How to Use This Manual

This manual describes the features and functions of EDGE 7.11. It is intended for application developers and system administrators and assumes that you have a basic understanding of programming and UNIX. For EDGE development and EDGE Client implementations, it assumes a basic knowledge of the Windows operating system.

EDGE Developer is used for EDGE development. EDGE Client is used for EDGE operations on Windows-based PCs. This manual describes both these interfaces.

The following is an overview of the contents of this manual:

Chapter 1, Introduction, provides an overview of project design, features that you can use to internationalize your projects, and information on how to use EDGE Developer to accomplish these tasks.
Chapter 3, Setting Up EDGE, describes required set up before using EDGE.

Chapter 2, Setting Up Projects, discusses required and optional features that can be modified for a project, including calendar, counters, intro screen, operator access, project options, and work schedules.

Chapter 5, Monitors, discusses the project and global monitors, which provide information about operator and queue activity.

Chapter 6, Correspondence, discusses the Correspondence function, which allows you to produce documents ranging from simple acknowledgment letters to complex preprinted forms, using data from your projects.

Chapter 7, Ad Hoc Reports, discusses the types of ad hoc reports, how to create them, and how to output them.

Chapter 8, EQL, discusses the EDGE Query Language (EQL).

Chapter 9, Archive, discusses the archive function, which allows you to save EDGE modules from the system or project level; to transfer modules to another system, another project, or another site; and to backup your projects.

Chapter 10, ACL, discusses the AdvantEDGE Command Language (ACL) and the commands that can be invoked.
Chapter 11, User Files and Fields, discusses how to create and use your own files and fields.

Chapter 12, EDGE Files and Fields, discusses EDGE system files and fields.

Chapter 13, Importing and Exporting, discusses how to import data to EDGE and export data from EDGE.

Chapter 14, Queues, discusses defining and seeding queues.

Chapter 15, Logic Flows & Tasks, discusses how to create logic flows, create tasks, and use the task scheduler.

Chapter 16, Verbs, discusses the actions that can be performed using verbs.

Chapter 17, EDGE DDE Capabilities, discusses Dynamic Data Exchange (DDE) processes.

Chapter 18, Message Queues, discusses message queues used for interprocess communication.

Chapter 19, User Screens, discusses user screens, including design issues.
Chapter 20, Screen Objects, discusses objects that are available for screens and their properties.

Chapter 21, ActiveX Controls, discusses how to extend the functionalities of EDGE screen objects and provide custom services using ActiveX controls.

Chapter 22, User Toolbars, discusses toolbars.

Chapter 23, EDGE Guide Functions, discusses the EDGE Guide area.

Chapter 24, Finishing Touches, discusses guide file access, guide options, hot keys, results, assembly, testing, and swapping.

Appendix Y, ASCII Chart, lists the ASCII characters 0–127.

Appendix Z, CALCULATE Functions, discusses each function used with the CALCULATE verb, shows the syntax, and provides examples.

Appendix AA, Format Codes, discusses codes that can be used with the FORMAT INPUT and FORMAT OUTPUT verbs to format date, time, and numeric values from internal to external format and from external to internal format.

Appendix AB, EDGE Environment Variables, discusses environment variables that can be used to modify EDGE’s behavior.
Appendix AC, Keyboard Commands in ASCII Operations, describes keyboard commands that can be used to perform actions in a guide.

Appendix AD, EDGE Directory Structure, discusses the directories and files EDGE uses.

Appendix AE, etww.ini File, discusses the ETWW.INI file’s sections and options.
If you need assistance, contact Pitney Bowes Technical Support.

Voice  +1 (800) 232 3343 (USA and Canadian customers only),
       or +1 (888) 507 5285 (international direct dial).
       +44 800840 0001 (European customers--press 1 for
       Technical Support, then 7)

Web    http://pitneybowes.com

Email  software.support@pb.com

Mail   Pitney Bowes
       One Global View, Troy, NY 12180

For online case management, go to the link http://www.pbin-
sight.com/support/online-support-services. The Pitney Bowes Soft-
ware Support & Maintenance Handbook can also be found by
searching at the same link.
Overview

EDGE Developer enables you to create and manage guides (scripts) to be used for sales campaigns, market research, customer service, credit and collections, order processing, and other telebusiness services.

This chapter provides an overview of the following:

- Understanding Guides and Projects.
- Project Design.
- Localizing EDGE.
- Getting Started.
- EDGE Developer Conventions.
- The User Interface.
- Menu Commands.
- Areas and Modules.
# Understanding Guides and Projects

A guide is a script used by an agent when talking with customers and consists of screens, queues, results, and logic.

A guide is part of a project. Other elements of a project include the following:

<table>
<thead>
<tr>
<th>Access and schedules</th>
<th>Access and schedules include features such as agent access, work scheduling, and preferred calling hours for customers.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database files</td>
<td>You can create your own files as well as use data from EDGE system files. Data can be imported to EDGE via file or tape, or can be entered directly. You can also import and export files to and from your mainframe computer. For example, you may purchase prospect tapes and transfer the data to a user-defined file.</td>
</tr>
<tr>
<td>Queues</td>
<td>Queues organize and manage your projects and give agents access to records in an orderly manner. The information can be ordered and maintained according to your project needs.</td>
</tr>
<tr>
<td>Reports</td>
<td>The information in any file on the system can be used to generate customer correspondence, sales reports, data analysis, or management reports.</td>
</tr>
</tbody>
</table>
EDGE also includes administrative features for system maintenance. The main components of EDGE administration are:

<table>
<thead>
<tr>
<th>ACL</th>
<th>ACL stands for AdvantEDGE Command Language. The ACL function allows you to perform functions that help you troubleshoot and manage problems.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security and monitoring</td>
<td>System security includes creating and maintaining user IDs and passwords. Monitoring allows an EDGE administrator to monitor an agent while in a guide and to view, input, or coach the agent by sending screen messages to the agent’s terminal.</td>
</tr>
<tr>
<td>System maintenance</td>
<td>System maintenance functions enable you to evaluate the performance of your system and guides. The reports allow you to view various components of the guide that relate to the speed and efficiency of your guide.</td>
</tr>
<tr>
<td>System setup</td>
<td>System setup functions include identifying information about the site, ports, printers, and I/O devices.</td>
</tr>
<tr>
<td>Tasks</td>
<td>Tasks allow you to perform batch processes on files. Tasks can be used to do a variety of functions such as seeding and clearing queues, printing reports and correspondence, and transferring records.</td>
</tr>
</tbody>
</table>
Version control  The version control facility saves a record of all changes made to tasks, logic flows, screens, and results. This makes it easy to try changes. If a change is not exactly what you needed, you can revert back to the previous version. When everything is working as desired, you can save your entire guide with a new version number then release it to operators.
Before you design a project, you should prepare a flow chart outlining the various paths an agent may need to follow during the course of a call. This flow chart should show the decision points and what happens at each point.

Thoughtful planning of your EDGE project will save you development time in the long run. By carefully designing your project, you can gain the following benefits:

- More productive and successful agents.
- Easier conversion of the project to the EDGE system.
- More useful information about your customers, products, and services.

Figure 1-1 shows the flow chart used with the Softsell project, which is used throughout this manual for examples.
Planning a Project

Each application of EDGE differs, but the following guidelines may help you in preparing your flow chart. Planning a project usually involves the following steps.

- Describe the Project Requirements.
- Design the Project.
- Plan the Files.
- Design the Guide.
- Design the Reports.
- Create the Guide.
- Modify the Guide.

**Figure 1-1. Sample Guide Design**
Describe the Project Requirements

This step helps you identify the boundaries, constraints, and objectives of the project. The following questions may help you with this process. You can also review your EDGE Developer training materials.

General Considerations

- What is the application of the project?
- What products or services are involved?
- What information will you be gathering? Information may include reports used, customer information such as main contact, status, etc.
- Are distinct sales campaigns a consideration, or are the same products offered all the time?

Calling Considerations

- Is this project outbound, inbound, or both?
- Will you use automatic dialers?
- If inbound, what type of calls will the agents be taking? Is a menu needed?
- If outbound, agents could be making different types of calls. What should be displayed to agents to show the type of calls being made?
• How many hours will agents spend on the phone? How many shifts per day?
• Will agents call across time zones?
• Will calling take place to other countries?
• Will callbacks be scheduled?
• How many times will the prospect or customer be called? Will the prospect be called at home if not reached at work?
• What dispositions or results will be used for inbound and outbound?
• How will previous contact information be displayed to agents? Do you need a special screen or can you use the EDGE History screen ($CALLHIST)?
• What types of calling lists will you use? Will they be time-ordered or first-in-first-out?

**Customer Considerations**

• Are you targeting existing customers or prospect lists?
• How many customers or prospects will be involved?
• Will agents handle their own customers, or will there be a customer pool?
If the agents handle their own customers, at what point is a customer assigned to an agent?

Will the same agent handle recalls (busies and no answers)?

What types of recalls will be made, and should they be grouped for reporting purposes? How many attempts should be made before a call is placed in a dead queue?

Will the agents call customers to confirm receipt of the product or monies?

If agents have their own customer base, how do you want to handle callbacks to customers when an agent is ill or on vacation?

Database Considerations

Will prospect or customer information come from existing files or from data entry? Will agents refer to the existing customer base during their presentations or are they building a new customer base?

Which files will be loaded or updated from existing files? How often?

What information will be used for the key to the files? What kind of searching capabilities will you need? What informa-
tion must you display to verify that the correct account has been found?

- How will you handle order cancellations or changes after the completed presentation? Will agents be able to update data?

- How will you store customer records? Will you be archiving data to tape, or updating files on your mainframe computer? If you are sending data to the mainframe, what type of information will you send?

**Correspondence and Report Considerations**

- What correspondence or preprinted forms will you need? (Include the name of the person responsible if someone else is preparing them).

- What performance or management reports will you need?

- How often (daily, weekly, monthly, etc.) will you need management reports and which ones will be needed when?

- Will you need to give anyone else information?

- Is the literature or fulfillment printed in batches or at time of request?

- Does the agent need to schedule a follow-up call after the material is sent?
**Design the Project**

Using simple flow chart symbols, you can graphically describe essential components of the project. Items that you might want to chart include the following:

- **Data/Files**: Identify the files needed and where the files originate. For example, you might import data from your company mainframe or from a tape.
- **Agents**: Identify the people responsible for providing specific information.
- **Reports**: Identify type and frequency of reports (for example, weekly, monthly).
- **Correspondence**: Identify correspondence needed (for example, fulfillment letters, sales lead sheets, preprinted forms) and frequency.
- **Interface issues**: Decide if communication to the mainframe is an issue. Are you going to import or export data from the mainframe?

**Plan the Files**

The files you need will depend on the project requirements and other constraints. Some of the components that affect file planning are described below.

- **EDGE files**: Review the EDGE files when considering your project files. You may be able to use data from the EDGE files.
Files you create are described in Chapter 11. Files created by EDGE are described in Chapter 12.

**Design the Guide**

Designing a flow diagram of the script or guide can be useful when you begin to create the guide in EDGE. Items that you might want to chart include the following:

- **Screens**: Indicate the type of screen, the text to appear, and the script branching considerations. Review this chart with others involved in the project.
**Design the Reports**

Report design can help you verify that the files are appropriate. You should make sure that the report details can be obtained from the project or EDGE files. Other design elements you might plan:

- Review the file layouts (project and system) to ensure that the report details can be obtained from the files.

- Decide how the reports are to be created; for example, you can use EDGE tools such as ad hoc reports, export data to other applications, or use Dynamic Data Exchange (DDE) to access other applications directly.

- Decide on the frequency (daily, weekly, monthly, special) for the reports. These decisions may affect the way you collect data.

**Paths**

Decide where and when a path might change. A decision to branch to another screen can be made before a screen is displayed, during the display of a screen, after exiting a screen, or from a field or function key.
Create the Guide

Instructions for creating the guide are too complex and varied to provide a prescriptive list. Instead, Table 1-1 contains a recommended checklist.

Table 1-1. Guide Creation Checklist

<table>
<thead>
<tr>
<th>Function</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>q Create results.</td>
<td>24</td>
</tr>
<tr>
<td>q Create screens.</td>
<td>19</td>
</tr>
<tr>
<td>q Write logic flows and attach.</td>
<td>15</td>
</tr>
<tr>
<td>q Set up guide options and agent access (for yourself only).</td>
<td>24</td>
</tr>
<tr>
<td>q Assemble the guide.</td>
<td>24</td>
</tr>
<tr>
<td>q Test the path of screens. For example, verify that logic operates when expected.</td>
<td>24</td>
</tr>
<tr>
<td>q Determine where hot keys are needed and create them.</td>
<td>24</td>
</tr>
<tr>
<td>q Test field validation, such as members of set and ranges.</td>
<td>11</td>
</tr>
<tr>
<td>q Note additional elements needed (for example, screens, logic, results) while testing the initial guide. Write as necessary.</td>
<td>24</td>
</tr>
<tr>
<td>q Continue assembling and testing the guide until it works.</td>
<td>24</td>
</tr>
<tr>
<td>q Clear test data, queues, and project monitor, if needed.</td>
<td>5, 14</td>
</tr>
</tbody>
</table>
Modify the Guide

Modification of your guide to respond to changing requirements should be anticipated.

Here are some of the items you might consider when making modifications:

<table>
<thead>
<tr>
<th>Screens</th>
<th>Are the screens clear and concise? Can they be easily read and understood by the agent?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calls</td>
<td>Are the test calls being filed properly? Are the result and queue settings correct at the end of the call?</td>
</tr>
<tr>
<td>Reports</td>
<td>Run some test reports and correspondence. Is the data correct?</td>
</tr>
</tbody>
</table>

Table 1-1. Guide Creation Checklist (continued)

<table>
<thead>
<tr>
<th>Function</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>q Load data and build cross-references, if needed.</td>
<td>11, 13</td>
</tr>
<tr>
<td>q Create access for all agents.</td>
<td>2</td>
</tr>
<tr>
<td>q Set up work schedule, calendars, and project monitor result groups.</td>
<td>2</td>
</tr>
<tr>
<td>q Seed queues.</td>
<td>14</td>
</tr>
<tr>
<td>q Release guide to agents (also referred to as swapping).</td>
<td>24</td>
</tr>
<tr>
<td>q Automate reports and other tasks.</td>
<td>15</td>
</tr>
</tbody>
</table>
Does it work? Test the project for every contingency. Ask someone who is unfamiliar with the project to participate in the testing.

Testing Test the project to see if data is being properly written to the files.

Agent test Give access to agents and test. Clear tested data before going live. Tested data includes both the data gathered by the system in specific EDGE files during the testing process, and changes that may have been made to user files and queues during testing.

Maintenance Continue to test and modify the guide throughout the life of the project. Add new screens, queues, or fields as the needs of the customer or marketplace dictate.
Localizing EDGE

EDGE can be modified to accept languages other than U.S. English and to support locale-specific formats. For information on localizing EDGE, see the EDGE 7.11 Localizing EDGE document.
# Getting Started

## Requirements

It is recommended that you use high resolution monitors (at least 800 x 600) for development. You should test your projects, however, on machines similar to what your agents will use.

## Installing EDGE Developer

To install EDGE Developer, follow the instructions that accompanied the software.

## Running EDGE Developer

To run EDGE Developer from the Start button, do either of the following:

- Select the shortcut to EDGE Developer.
- Run the etww.exe program.

Before you can log into ETW–W, you must set up the connection to the EDGE server. For more information about setting up the connections, see the section [Client Settings](#).

## Connecting to EDGE Developer

Once your connection setup is complete, you can connect to EDGE Developer by selecting the Connect from the File menu or by clicking the connection icon on the toolbar.
If you have not specified an EDGE logon on the Connection Setup tab, the EDGE Developer Connect screen is displayed. The screen is similar to the following:

![Connect Screen](image)

**Logging On**

To log on to EDGE Developer, enter your user ID, password, and workstation ID as appropriate. Press RETURN or ALT+O, or click OK to continue the log on process. To change the setup properties, click Setup. To quit, click Cancel.
To log off EDGE Developer, select Exit from the File menu or press ALT+X.
**EDGE Developer Conventions**

EDGE Developer provides a development interface to all functions of EDGE.

EDGE Developer follows Microsoft Windows standards for user interfaces. For example, you can have multiple development windows open, you can use drag and drop, and toolbars are available for common functions. This manual assumes you are familiar with Microsoft Windows operating systems.

**Terminology**

The following terminology is used with ETW–W.

**Drag and Drop**

Drag and drop is a Windows 95/98 method for copying objects. For example, to copy an ad hoc report, you drag it to the project where you want the copy.

**Objects**

Objects describe units of information, including cells, paragraphs, characters, and the documents in which they reside. For example, in EDGE Developer, both screens and list boxes are considered objects.
Properties

Properties define the appearance or state of an object. An object’s properties may consist of physical characteristics as well as its operational state. You can change the object itself by changing its properties. For example, you can modify a screen (object) by changing its layout (changing a property).

Progressive Disclosure of User Interface

The process of disclosing only those features that are usable at a particular time. This prevents the user from being overwhelmed by being shown too many things at once. The tabbed page organization helps with this by highlighting only those properties that are accessible to the user based on the selected object.

Tearoffs

Tearoffs are a feature of tabbed pages that have lists of items that can be modified. You can move the tab page off the form by dragging the tab to another area of the desktop. When torn off, the tab transforms into a list. Commands on the tab are not available when the tab is a tearoff. To return the tab to the form, click the close box. For an example of a tearoff, see Figure 1-2.
Using Uppercase and Lowercase Letters

Figure 1-2. Example of a Tearoff

EDGE on a UNIX system is case-sensitive, which means you can create a file called CUSTOMER and another file called customer. EDGE considers the CUSTOMER file as a different file from the customer file. This also means that you must use the correct case when specifying elements in EDGE.
However, EDGE on an NT system is not case-sensitive. This means that a file called CUSTOMER and another file called customer are treated as the same file.

Before you archive or unarchive modules from a UNIX system to an NT system or vice versa, please take case sensitivity into consideration.

Unless otherwise required, we recommend that uppercase letters be used for all EDGE elements (IDs). Using uppercase letters for IDs eliminates the need to remember the case and may eliminate the possibility of updating the wrong element.

If you are using DBL (Database Link), refer to the documentation for your database for possible naming restrictions.
The User Interface

The EDGE Developer user interface consists of the following:

- Browser.
- ETW–W Toolbar.
- Menus.
- Tabbed Pages.
- Property Pages.

Browser

Once a connection has been established, the Browser is displayed. This window lists all the projects to which you have development access.

You can choose to display them in one of the following formats:

- Icon View. You can select either large or small icons.
- Name View.
- Window View. Select this view by double-clicking a project icon.

To display an area from the Browser, click the project icon, then do one of the following:
Click the tool on the toolbar that represents the area. (For more information about the toolbar, see the section ETW–W Toolbar.)

Select the area from the View menu or press the shortcut keys on that menu.

The modules available in each area are listed in Table 1-2.

**Icon View**

When viewed by icon, the projects are displayed as icons on the Browser window. Figure 1-3 shows the Browser in icon view.

*Note: The EDGE icon indicates the EDGE level.*
When the Browser is viewed by name, the projects are displayed as a list. The cursor is positioned on the currently selected project and the following information about the project is listed.

- Project name
- Creator; user ID of person who created the project.
- Creation date.
- Description.

**Figure 1-4** shows the Browser in name view.

![Browser in Name View](image)

**Figure 1-4.  Browser in Name View**

**Areas Window**

The Areas window is displayed when you double-click a project icon on the Browser. When the window opens, the areas for the selected
The EDGE project are displayed. A plus (+) sign to the left of an item indicates it can be expanded to show subentries. When the item is expanded, the plus sign changes to a minus (-) sign. For example, Figure 1-5 shows the EDGE project expanded while the other projects are collapsed.

![Figure 1-5. Areas Window](image)

To display an area tabbed page from the Areas window, do any of the following:

- Double-click the area name.
- Click the tool on the toolbar that represents the area.
- Select the area from the View menu.
**Shortcut Menu on Titlebar**

Each window or dialog box in EDGE Developer has an icon on the left side of the titlebar. When you click the icon, a menu is displayed with the following commands:

- **Close**
  Closes the active dialog box. (You can also press ALT+C or CTRL+F4.)

- **Move**
  Changes the pointer to a tool that allows you to move the active dialog box to another area of the window. (You can also press ALT+M.)

- **Minimize**
  Minimizes the dialog box to an icon. (You can also press ALT+N to minimize.)

- **Next**
  Makes the next dialog box active. (You can also press CTRL+F6.)

- **Restore**
  Restores the icon to the maximized state. (You can also press ALT+R.)

**ETW–W Toolbar**

The ETW–W toolbar contains tools (icons) that provide the following general functions:

- Application development.
- System administration.
- Application production.
- General shortcuts.

The tools are identified in Figure 1-6.
Figure 1-6. ETW-W Toolbar

The toolbar is color-coded to visually group like functions:

- **Blue**: Application development.
- **Yellow**: System administration.
- **Orange**: Application production.
- **Green**: General shortcuts.

Toolbars may be dragged to a different location on the screen or positioned vertically. To resize, grab and drag the corner of the palette.

**Menus**

EDGE Developer has a set of standard menus that are always available. Each area also has its own menu, which is displayed between the View and Window menus when the area tabbed page is dis-
played and active. For example, in Figure 1-7 the Database menu is available because the Database area tabbed page is active.

Area menus contain commands for each of the modules in the area. Submenus are available from the main menu depending on the tab currently selected. For example, when the Files/Fields tab is selected, submenu items related to the Files/Fields module are displayed.

Also, some menu commands are only available when an item is selected.
**Tabbed Pages**

Once you select a project and an area, a tabbed page for that area is displayed. For example, if you select the SOFTSELL icon on the
Browser, then select the report icon on the development toolbar, a report tabbed page similar to the following is displayed:

More than one tabbed page can be opened at a time. The tabbed page contains the following areas:

- **Area name**: The name of the area.
- **Close button**: Closes the current window.
### Command buttons

Used to add, modify, or delete objects. Some commands are only available when an item on the tab page is selected. The following commands are available on all tab pages:

- **Add**: Adds an object.
- **Delete**: Deletes the selected object.
- **Modify**: Allows you to modify the selected object.

### List box

Lists the available items for the selected tab.

### Modules

The modules in the area. Modules are represented by tabs.

### Project

The name of the current project.

---

**List Box Shortcut Menu**

For each list box located on a tabbed page, a shortcut menu is available. When you click the icon, a shortcut menu allows you to do the following:

- **Add**: Adds an object.
- **Delete**: Deletes the selected object.
- **Modify**: Allows you to modify the selected object.
- **Refresh List**: Refreshes the items displayed in the list box.
**Keyboard Navigation**

Tabbed pages may be accessed via the keyboard by pressing the right or left arrow keys. The arrow keys move the focus from one tabbed page to another tabbed page. To access a specific tab page, use the TAB key. The focus will move to the list box selections. The arrow keys can be used to select an item from the list box. If you continue pressing the TAB key, the focus will continue to the next enabled item on the tab page eventually returning the focus back to the tab name.

**Property Pages**

Property pages list all the user accessible properties for the item. For example, if you select the Results tab and the result ERROR, a property page similar to the following is displayed:
To display the property page for an item, select it from the list then press the Modify button or double-click the item.

The property page contains the following:

- **Close button** Closes the current window. If properties have changed, you are prompted to save the object.
More than one property page can be opened at a time.

Depending on the property, a value can be specified in several ways. In the following example, the value for the Logic property can be specified by doing any of the following:

- Type the value in the area next to the property name.
- Select the value from a list of items by pressing the arrow and selecting a value.
- Select the value from a list of guide files and variables by pressing the three-dot button and selecting a field or variable.

**Specifying Property Values**

<table>
<thead>
<tr>
<th>Name of item</th>
<th>The name specified for the item. For example, ERROR is the name for a result.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project</td>
<td>The name of the current project.</td>
</tr>
<tr>
<td>Properties</td>
<td>Displays properties for the current object.</td>
</tr>
</tbody>
</table>

![Diagram of property specification](image)
**Selecting a Value**

When an arrow is displayed next to a property, it usually means the property value can be selected from a list of appropriate values. For example, in the following figure, the ad hoc reporting types are displayed for the Report type property.

To display the values for the property, press the arrow. To select a value, click the value. To change the value, press the arrow and select another value.

**Selecting Multiple Values**

For some properties, when the arrow is displayed, multiple values can be specified. An example of this type of property occurs on the Result Groups tab shown below.
To begin, click the arrow for the property. This displays a second arrow with a list box containing the current values for the property. If there are no values specified, the list box is empty. When the second arrow is pressed again, a list of the available values for the property is displayed. For example, for the Result Groups, a list of results is displayed.

<table>
<thead>
<tr>
<th>#</th>
<th>Title</th>
<th>Result(s)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ø</td>
<td>1 NONE</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>✓</td>
<td>✓</td>
<td>BAD CALL, ERROR, NO.SALE, SALE</td>
</tr>
</tbody>
</table>

To select values  Click the value. After the mouse is released, the value is listed next to the list box arrow. To select another value, click the arrow and select a value. When done, click outside the property. The values are listed, separated by commas, in the property area.

To delete values  Click the value on the list box, then press the DELETE or BACKSPACE key.

To display values  Press the arrow to display the values entered for the property.
Entering Multiple Values

For some properties, the arrow displays the values you have specified for the property. If you have not specified values for the property, the list box is empty. In the following example, three values have been specified for the Include from field property.

To modify values

Click the value on the list box, type the change, then press ENTER.

To add values

Type the value and press ENTER. The value is added to the list.

To delete values

Click the value on the list box, press the DELETE or BACKSPACE key, then press ENTER.

To display values

Press the arrow to display the values entered for the property.

To modify values

Click the value on the list box, type the change, then press ENTER.
Selecting a Field or Variable

Generally, when a three-dot button is displayed with a property it means you can specify a field or variable for the property. When you press the three-dot button, a dialog box similar to the following is displayed:

To select a field, expand the file by pressing the plus (+) sign next to the file name. Then double-click the fieldname or click the fieldname and press OK.

To select a variable, double-click the variable or click the variable and press OK. When a variable is selected, the sign for the variable is
Using Grids

Some properties are displayed on grids, which group properties that are related to each other. For example, in the following grid, the Field ID, Break/Total, and Options properties are related. If no values have been specified for the properties, the grid looks similar to the following:

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field ID</td>
</tr>
<tr>
<td>Break/Total</td>
</tr>
<tr>
<td>Options</td>
</tr>
</tbody>
</table>

Specifying Grid Values

To specify a value, position the cursor in the cell for the property by clicking in the cell. Enter the value or select the value from a list by selecting the arrow or three-dot button. After specifying the properties for the row, the grid is similar to the following:

<table>
<thead>
<tr>
<th>Field ID</th>
<th>Break/Total</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCT</td>
<td>Break</td>
<td>ACCOUNT NUMBER</td>
</tr>
</tbody>
</table>
Modifying Grid Values

To modify a value, click in the cell for the property and make the change by typing or selecting the value.

Adding or Deleting Grid Rows

To add or delete an entire row, move the cursor to the first column on the grid until it is displayed as a horizontal arrow similar to the figure below:

When the arrow is on the row you want to delete or add, click the right mouse button. A shortcut menu is displayed similar to the following:
To add a row after the selected row, click Add Row or press ALT+A. To delete the row, click Delete Row or press ALT+D.
Menu Commands

Menus allow you to perform commands. Area commands are described in the appropriate chapters. The following menu commands are described in this chapter:

- Edit Menu.
- File Menu.
- Help Menu.
- View Menu.
- Window Menu.

Edit Menu

The following commands are available from the Edit menu:

- **Add**: Adds an object based on the selected tab. For example, if the Screens tab is selected, the Add command opens the screen editor so you can add a new screen. The Add command is also displayed on appropriate tabs.

- **Copy**: Copies the selected text to the clipboard. You can also use CTRL+C.

- **Cut**: Copies the selected text to the clipboard and deletes from the current location. You can also use CTRL+X.

- **Delete**: Deletes the selected object. You can also use ALT+D or the delete key. A warning message will display.
Duplicate

Find

Not currently supported.

Only available in the logic or task editor. Searches for the specified text within a logic flow or task. When selected, a dialog box similar to the following is displayed:

The following can be specified:

- **From beginning of logic**: Searches from the beginning of the logic flow.
- **From cursor position**: Searches from the cursor position.
- **Match whole word**: Finds the text only if the whole word matches the specified text. If selected, a search for REC would not find RECORD.
- **Match case**: Finds the text only if the case matches. If selected, a search for rec would not find REC.
Wrap around search
If checked, when the search gets to the end of the logic flow, it starts again at the beginning.

When the text is found, the cursor is moved to the text on the logic editor. If the text cannot be found, a message is displayed:

Modify
Allows you to modify the selected object by opening the property page or editor for the object. You can also use ALT+M.

Paste
Pastes the contents of the clipboard to the location following the current cursor position. You can also use CTRL+V.

Replace
Not currently supported.

Undo
Undoes (reverses) the effect of the last operation. You can also use CTRL+Z. Not all operations can be undone.

**File Menu**

The following commands are available from the File menu:

Change User
Displays the EDGE Developer connect screen and allows you to log on with another user ID.

The current user ID is disconnected from the server.

Close Project
Closes the current project and returns to the Browser.

Close Window
Closes the active dialog box. If the object was modified, you are notified and allowed to save it, cancel the close, or exit without saving.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete Project</td>
<td>Deletes the currently active project.</td>
</tr>
<tr>
<td>Disconnect</td>
<td>Disconnects from the server.</td>
</tr>
<tr>
<td>Exit</td>
<td>Exits EDGE Developer. If an object was modified, you are notified and allowed to save it, cancel the close, or exit without saving.</td>
</tr>
<tr>
<td>List Projects</td>
<td>Lists all the projects on the system. You can also print the list.</td>
</tr>
<tr>
<td>Modify Project</td>
<td>Opens the Project Definition for the selected project so that the description can be modified. (No other property is modifiable after the project is created.)</td>
</tr>
<tr>
<td>New Project</td>
<td>Opens the Project Definition for a new project.</td>
</tr>
<tr>
<td>Print</td>
<td>Not currently supported.</td>
</tr>
<tr>
<td>Printer Setup</td>
<td>Allows you to specify the client printer.</td>
</tr>
<tr>
<td>Save Object</td>
<td>Saves the object, including any changes to the document. (The object is not closed when it is saved from the menu.)</td>
</tr>
<tr>
<td>Save Object As</td>
<td>Saves the current object under a new name and opens the editor or tab page associated with the object.</td>
</tr>
<tr>
<td>Set Server Printer</td>
<td>Specifies the server printer for the current login session. This property is optional.</td>
</tr>
<tr>
<td>Setup</td>
<td>Specifies the connection between your workstation and the server. For more information, see Chapter 3.</td>
</tr>
</tbody>
</table>
### Help Menu

The following commands are available from the Help menu:

- **About ETW for Windows**
  - Displays version and contact information.

- **Help Topics**
  - Not currently supported.

- **Learning ETW for Windows**
  - Not currently supported.

- **Server Patch**
  - Used for patches. One of the following can be specified:
    - **Install Server Patch**
      - Used to install a patch. For more information, see the Patch Installation Procedures that accompanied each patch.
    - **List Server Patch History**
      - Used to view a history of patches that have been applied to an EDGE installation.

  Patch history is not cleared by an upgrade. Therefore, if an EDGE installation has been upgraded, some of the patches listed may be for an older EDGE version.

### View Menu

The View menu allows you to open area tabbed pages for the selected project. For example, if the SOFTSELL project is selected
when you select Database, the Database area tabbed page is displayed. The following areas are available:

- Database.
- Queues. (Not available if EDGE is selected.)
- Guide.
- Tasks.
- Setup.
- Reports.

You can also open global level areas when a project is selected. The global areas that can be displayed are:

- Users.
- Server.
- Telephony.
- Tools.
- Tasks.
# Window Menu

The Window menu allows you to manage the window and display information about the server. The following commands are available.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ACL</strong></td>
<td>Transfers to AdvantEDGE Command Language (ACL) mode. This facility allows you to use the ACL commands and also the EDGE Query Language (EQL) commands.</td>
</tr>
<tr>
<td><strong>Arrange Icons</strong></td>
<td>Arranges the icons, which represent minimized dialog boxes, horizontally on the bottom of the EDGE Developer window.</td>
</tr>
<tr>
<td><strong>Browser</strong></td>
<td>Displays the Browser window, which displays all the projects on the system.</td>
</tr>
<tr>
<td><strong>Cascade</strong></td>
<td>Arranges the open dialog boxes in a cascade from the left top of the screen toward the bottom right. The currently active dialog box is on the top, with the most recently displayed dialog boxes in descending order.</td>
</tr>
<tr>
<td><strong>Display Bulletins</strong></td>
<td>Displays the bulletins, if any.</td>
</tr>
<tr>
<td><strong>Server Information</strong></td>
<td>Displays information about the server, such as the user ID and port.</td>
</tr>
<tr>
<td><strong>Tile Horizontal</strong></td>
<td>Arranges the dialog boxes horizontally on the window.</td>
</tr>
<tr>
<td><strong>Tile Vertical</strong></td>
<td>Arranges the dialog boxes vertically on the window.</td>
</tr>
</tbody>
</table>
In addition to the commands, a list of all open windows is displayed on this menu. To make a window active, select it from the list. (Dialog boxes are numbered in the order in which they were opened.)
**Areas and Modules**

EDGE Developer is organized into areas and modules. Areas include modules that are related to the area. For example, the Queues area includes the Queue Definition, Queue Seeding, and Queue Utilities modules.

Some areas are available only on the global or project level. For a listing of all areas and modules, see Table 1-2.

**Table 1-2. Areas and Modules**

<table>
<thead>
<tr>
<th>Area</th>
<th>EDGE (Global Level)</th>
<th>Projects (Project Level)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queues</td>
<td>Not applicable.</td>
<td>Queue Definitions. Queue Seeding. Queue Utilities.</td>
</tr>
</tbody>
</table>
### Areas and Modules (continued)

<table>
<thead>
<tr>
<th>Area</th>
<th>EDGE (Global Level)</th>
<th>Projects (Project Level)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Multiple Project Assemble/Swap.</td>
<td>Logics.</td>
</tr>
<tr>
<td></td>
<td>General (project selection, project operations).</td>
<td>Results.</td>
</tr>
<tr>
<td></td>
<td>Schemes.</td>
<td>Tables.</td>
</tr>
<tr>
<td></td>
<td>Styles.</td>
<td>Options.</td>
</tr>
<tr>
<td></td>
<td>Libraries/Functions.</td>
<td>OLE Objects.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>System Screens.</td>
</tr>
<tr>
<td>Tasks</td>
<td>Task Definition.</td>
<td>Task Definition.</td>
</tr>
<tr>
<td></td>
<td>Task Scheduler.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ACTZ</td>
<td>Agent (Operator) Access.</td>
</tr>
<tr>
<td></td>
<td>International ACTZ</td>
<td>Calendar.</td>
</tr>
<tr>
<td></td>
<td>ODBC Data Source.</td>
<td>Work Schedules.</td>
</tr>
<tr>
<td></td>
<td>Selection List.</td>
<td>Selection List.</td>
</tr>
<tr>
<td></td>
<td>Monitor.</td>
<td>Correspondence.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Monitor.</td>
</tr>
<tr>
<td>Users</td>
<td>Groups.</td>
<td>Not applicable.</td>
</tr>
<tr>
<td></td>
<td>Users.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ODBC Users.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bulletins.</td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>EDGE (Global Level)</td>
<td>Projects (Project Level)</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>Tools</td>
<td>Archive. Locator. 3270 Script. 3270 KeyMap.</td>
<td>Not applicable.</td>
</tr>
</tbody>
</table>
Overview

This chapter describes the following features that can be modified for a project:

- Counters.
- Schedules.
- Intro Screen.
- Operator Access.
- EDGE Client Options, Labels, and Schemes.
- Project Options.
- Version Control.
- Coded Project Selection.
- Server Images.
Counters

Counters can be used wherever you need to assign and keep track of numbers. You can use counters to assign unique sequential numbers as record IDs (primary keys) for records in a file, to count inventory items, or for any other purpose where you need to keep track of numbers. Counters can be created in the following ways:

- When the project is created. For more information, see the section Create Project Definitions.
- With the The EDGE Counter Page.
- With the COUNTER verb.

The list of counter names is kept in an EDGE file called EDGE_COUNTERS. There are no limits to the number of counters that can be created.

This section explains counters and how they are created with a project or with the Counter page.

Using Counters

Counters exist at the EDGE level, whether they are created with the Counter page or when a project is created. Consequently, counters can be shared between guides. This allows you, for example, to use counters to provide common keys for files shared by many projects.
### Setting Up Projects Counters

Depending on the database type for the counter, you can specify the direction of the counter; which can be either increment, decrement, or both. You can also specify a maximum or minimum value for the counter and that when the value is reached or exceeded, the counter is to cycle back to the minimum value (for incrementing counters) or maximum value (for decrementing counters).

### Project Counter

EDGE uses the project counter to assign numbers for system-assigned call events. When you create a project, you can accept the default name for the project counter, which is the same as the project name, or specify the name of another existing counter. If you accept the default project counter, the counter is not actually created until the first system-assigned call event is created. This can occur when testing a guide either in live mode, or in operations. You can also change the counter for an existing project using **Project Options** and specify a different project counter at that time.

### Deleting Counters

Counters can be deleted from the Counter page or using the **COUNTER** verb. You may want to delete a counter, for example, because you want to change its properties. Once you delete it, you can recreate it using different properties.

Deleting a counter takes effect immediately throughout EDGE, whether you delete it in test mode or live mode, in a project, or at the...
EDGE level. For this reason, it is recommended that if deletions are necessary, they be made when no other users are active.

Because counters exist at the EDGE level and the number of projects sharing the counter is unknown, when you delete a project, the project counter associated with that project is not deleted.

Be aware of the following:

- If you delete a project counter used to assign call IDs, the next time a call ID has to be assigned, EDGE will recreate the counter as a c-tree counter using the default settings. EDGE then increments the counter until it finds an available number. Depending on the number of records already in the file, this could take a significant amount of time.

- If you delete a counter other than a project counter, the next time an increment or decrement command is invoked for that counter, EDGE returns a status of -102100 (counter does not exist). It is up to your logic flow or task to handle this.

**Caching**

Counters manage multiple requests very efficiently. To further increase the efficiency of counters, you can specify that EDGE keep a cache of numbers in memory. (Caching is specified when you define the database device.)
The availability of caching depends on the database used to create the counters.

Caching has the following implications for counter cycling:

- If you specify cycling for a decrementing counter, you must enable caching before you create the counter. If you do not, the counter will not cycle. If it is decremented past its minimum value, an error is generated.

- The number of values cached must be smaller than the number of values being cycled. For example, if you have ten numbers in your cycle, your cache cannot be greater than nine. If you specify a larger cache, EDGE automatically reduces it to one less than the number of values being cycled.

**Using Offsets**

Offsets allow you to increment or decrement a counter by a value other than the amount specified when you created the counter. Offsets can be used, for example, to adjust counters being used to track quantities of inventory items. When you get a new shipment of an item, you can use the offset to readjust the counter for that item.

The availability of offsets depends on the database used to create the counters.

*Note: If you use counters for quantities, and wish to display the values in the counters, create a relational field using the Advanced User*
The EDGE Counter Page

The EDGE Counter page can be used to create a user-defined counter. Once a counter has been saved, you cannot modify any of its properties. This applies to counters however they are created: as part of a project, from the Counter page, or with the COUNTER verb.

You can use the default values for the properties by entering the database device name, leaving all other properties blank, then saving the counter. When you open the counter, the default values are displayed.

To create a counter using the Counter page, select the following:

EDGE / Database / Counters

When you add a counter, a page similar to the following is displayed:

Defined relationships:

RELATIONS(S) FOR SELECTING/SORTING/TOTALING
1.(01) TEDGE_COUNTERS;X;7;1;1
OUTPUT/DISPLAY FORMATTING
2.(01) G.1
Properties:

- **Allow counter cycling**: If checked, the counter is reset to its minimum or maximum value. This value is determined by the Direction property. By default, this property is cleared. This property is enabled only if the Database device name is an Oracle type.

- **Database device name**: The database device for which the counter is being created.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direction</td>
<td>The counter is to be incremented or decremented or both. Valid selections depend on the database type; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Both</td>
</tr>
<tr>
<td></td>
<td>Decremented</td>
</tr>
<tr>
<td></td>
<td>Incremented</td>
</tr>
<tr>
<td>Increment/Decrement amount</td>
<td>The value by which the counter is incremented or decremented; must be a number. The default is 1. For Sybase and Informix, 1 is the only value allowed.</td>
</tr>
<tr>
<td>Initial counter value</td>
<td>The initial value for the counter; must be a number. For an incrementing counter, the default is the minimum value. For a decrementing counter, the default is the maximum value.</td>
</tr>
<tr>
<td>Maximum counter value</td>
<td>The maximum value allowed for the counter.</td>
</tr>
<tr>
<td>Minimum counter value</td>
<td>The minimum value allowed for the counter.</td>
</tr>
</tbody>
</table>

DBL type counters create a table in the RDB which stores the current value of the counter.
Schedules

Overview

Schedules are defined through a combination of modules. You first define work schedules, such as normal working hours and holiday working hours, then apply the work schedules to monthly calendars. You can also specify the last day the schedule is to be used.

The Calendar feature works with Work Schedules to indicate the days and hours in which call events can be scheduled. This allows you to restrict the scheduling of call events to times that are appropriate to your work shifts and customer preferred calling hours. It can also be used to warn the agent when an outbound call is outside the preferred customer calling hours.

By default, the time in a work schedule is based on the customer’s local time. For example, if preferred calling hours are from 1 to 5 PM, an agent in California could place a call to a customer in New York at 11 AM and be within the preferred range.

The user or group time zone is used instead of the site time zone if the target queue is a time-ordered protected queue and country and/or phone codes are defined for the user or group. If a work schedule is defined for the associated user or group, then that work schedule is used instead of the default work schedule for the project.
Time zones can be associated with agents and agent groups by specifying the country and phone codes for the agent or agent group in the User Definition or Group Definition profiles. When scheduling callbacks and seeding calls to protected queues, EDGE automatically takes into account the preferred calling times in different time zones, as well as agent or agent group availability in their local time zone.

To determine the time zone associated with a user or group, EDGE checks the agent record for country and phone codes first, then the group record, and then the site record. EDGE uses the first valid data encountered.

The calendar and work schedule can affect the following:

- Make an Outbound Call.
- Scheduling a Callback.
- Seeding to a Time-Ordered Queue.
- Setting a Result with a system-arranged callback.

For the work schedule and calendar to be used as the basis for restricting scheduling or warning the agent, you may need to specify some additional settings as described under each situation.
**Make an Outbound Call**

You can display a warning message when an agent tries to make an outbound call that is outside the preferred calling hours. The warning is displayed for both the Make Call and Next Call commands on the Call Center screen in EDGE Client. The warning does not restrict the agent from making the call. The agent can continue to make the call.

To enable the warning, you must do the following:

- Specify the work schedule and calendar.
- Check the box to display the preferred calling hour warning message on the Project Options page.

You can restrict an agent from making an outbound call outside the preferred calling hours by using a logic flow to determine the value of the SYS field $INPCH. If the value is N, the call is outside the preferred calling hours. The logic flow should be attached to the before screen logic area for the first outbound screen and route to a user-defined screen that explains why the agent was restricted from making the call.
**Scheduling a Callback**

You can restrict an agent from scheduling a callback outside the preferred calling hours. If the date or time is outside the preferred calling hours specified for the work schedule and calendar, a message is displayed that suggests an acceptable date and time. The agent would see a message similar to the following:

```
NOT WITHIN PREFERRED CALLING HOURS. TRY 09/24/96 AT 09:00AM
```

If the agent tries to enter a date beyond the specified last day to schedule callbacks, the following message is displayed:

```
DATE BEYOND LAST CALLBACK DATE.
```

To restrict scheduling, you must do the following:

- Specify the work schedule and calendar.
- Specify the last day to schedule callbacks.

**Seeding to a Time-Ordered Queue**

You can restrict seeding of call events for a time-ordered queue to the preferred calling hours specified for the work schedule and calendar. To restrict seeding, you must:

- Specify the work schedule and calendar.
• Check the box to adjust the date and time values to conform to the work schedule and preferred calling hour, on the Assign Time-Ordered Queue Options page. For more information, see the section Time-Ordered Queue Properties.

**Setting a Result**

By setting a result, you can restrict the scheduling of a call event to days defined with a Work Schedule. When the result is set, the system only schedules the call event to a Calendar day associated with a work schedule that specifies a work shift or preferred calling hours. The system restricts the scheduling based on the entire day, not the preferred calling hours.

To restrict scheduling, you must do the following:

• Specify the work schedule and calendar.

• Specify the result as system-arranged based on work schedule. (For more information, see Results.)

**Work Schedule**

The work schedule is used to specify the work shifts and the customer preferred calling hours. Used in conjunction with a calendar, you can warn or restrict agents from making calls outside the preferred calling hours.
You can create a work schedule for any type of schedule, including non-work periods, such as holidays and weekends. You can restrict calls from being scheduled on non-working periods by creating a work schedule that does not identify any hours for the work shifts or preferred calling hours. A nonwork schedule can be used to identify days when calls should not be scheduled.

To create a work schedule, select the following:

Project / Setup / Work Schedules / Add

Work schedule names must be from one to four characters in length. Typically, the name reflects the kind of work or non-work schedule, such as WKDY for weekday, WKND for weekend, and HOLI for holiday.
**Properties:**

- **Customer preferred calling hours**: The time periods which are preferred for calling the customer; can include up to six time periods. This is used for outbound calls, scheduling callbacks, and queue seeding.

- **Daily work shifts**: The time periods agents work; can include up to six time periods. If time is displayed in 24-hour (international) mode, the AM/PM column is not displayed.

- **Description**: The type of work schedule.
Calendar

The Calendar page allows you to associate work schedules with specific days of the month. Optionally, you can assign the work schedule to a specific agent or group of agents, but not both. If you do not select a specific agent or group of agents, the work schedule applies to all agents.

If you want to restrict the scheduling of callbacks, you must associate a work schedule with each day in the month. You must do this for each month the project is in operation. If you do not, agents will be able to schedule callbacks for any day and hour.

You do not have to reassemble the guide when you create or change the calendar or work schedule hours. To change, make changes to the calendar and work schedules, then release the guide to operations.

Properties: (continued)

- International time: When checked, indicates the time is displayed in 24-hour mode. This property is display only; it is set in the EDGE Setup area using the Country Code module.

- Work schedule name: The name of the work schedule.
To create a calendar, select the following:

Project / Setup / Calendar / Add

For an example of a calendar, see Figure 2-1. When you define a calendar, you select the month the calendar is for. When you save the calendar for the first time, EDGE automatically names the calendar, using the first three letters of the month followed by the last two digits of the year, for example, AUG96. If a month is not specified, the current month is used. The month abbreviations for English are listed below.

- JAN • JUL
- FEB • AUG
- MAR • SEP
- APR • OCT
- MAY • NOV
- JUN • DEC

Note: Before creating a calendar, you should create Work Schedule or copy work schedules from an existing project. If you do not specify a work schedule in the calendar, the calendar is not saved.
To select the entire month, click here.

To select a particular week in the month, click the square to the left of the week.

To select the same day in each week in the month, click the week day name.

---

**Figure 2-1. Calendar**
**Associating a Work Schedule With Days**

To associate a work schedule with specific days on the calendar, first select the work schedule from the list of schedules. Then, select the applicable days:

- Specific days
  - Click the day.
- A week
  - Click the box in the column to the left of the week.
- Specific day for all weeks
  - Click the week day name.

**Assigning a Work Schedule to Agents**

After selecting the work schedule and work days, optionally you can assign the work schedule to a specific agent or group of agents. Select an agent from the User list, or select the group of agents from the Group list.

EDGE does not allow you to assign a specific agent and a group of agents to the same work schedule in a given calendar. For example, if you have already selected a group of agents, EDGE sets the group name to None when you attempt to select an agent from the User list. *None* indicates that you have not made a selection. If the User and Group lists are both set to None, the work schedule applies to all agents.
Figure 2-2 illustrates the EDGE interfaces that support agents and groups of agents that work in multiple time zones.
Setting the Last Day for Callbacks

The Last Day for Callback page allows you to set a date after which no more callbacks can be scheduled. This feature is useful in a project that has a specific end date. If the agent tries to schedule a call beyond the date set, a message is displayed that advises the agent and recommends a valid date.

Note: You do not have to reassemble the guide to change the last day to schedule callbacks. Simply make the change and swap the guide.

To set the last day to schedule callbacks, select the following:

Project / Setup / Project Info / Set Last Day for Callbacks

A page similar to the following is displayed:
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>The date displayed for the month, day, and year is cleared.</td>
</tr>
<tr>
<td>Month, Day, Year</td>
<td>The last callback month, day, and year.</td>
</tr>
<tr>
<td>Today</td>
<td>The last day is set to the current date.</td>
</tr>
</tbody>
</table>
Intro Screen

The Intro screen allows you to create a screen that is displayed during the time project files are being opened. Because the screen displays before the files are opened, the screen can contain text only. The Intro screen does not have to be assembled. You can turn it ON or OFF immediately.

To create an Intro screen, select the following:

Project / Setup / Project Info / Intro Screen

A page similar to the following is displayed:
Properties:

Auto dismiss intro screen
If checked, the screen is closed when the project files are opened. If the files open quickly, the agents may not see the screen.
If cleared, the agent must press RETURN to exit the screen.

Enable intro screen
When checked, the intro screen is displayed.

Intro screen text
The text to be displayed to the agents while the project is opening. The text does not have to be enclosed in quotation marks.
Operator Access

The Operator Access module is used to set up the following agent features:

- Operator access.
- EDGE Client options.
- EDGE Client labels.
- EDGE Client schemes.

To assign operator access, select the following:

Project / Setup / Operator Access

Operator access specifies the type of access an agent has for a project, such as the type of calls that can be handled.

Before operator access can be assigned, the agent must have a user ID.

In addition to assigning operator access to each agent, you must also do the following:

- Allow project access to the user ID. This is specified as part of User Setup
• Allow operator access to the project. This is specified as part of Project Options.

Note: If several agents are to have the same access settings, you can set one agent up, then use the Save Object As command on the File menu until all agents have been specified.

The Operator access page is similar to the following:
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow operator access</td>
<td>If checked, allows access to the project.</td>
</tr>
<tr>
<td>Groups</td>
<td>The group assigned to the user ID, if any. This is display only.</td>
</tr>
<tr>
<td>Inbound</td>
<td>When checked, allows agent to receive inbound calls.</td>
</tr>
<tr>
<td>Manual access only</td>
<td>When checked, allows access to manual queues only. When cleared, allows access to search queues as well as manual queues.</td>
</tr>
<tr>
<td>Manual queues</td>
<td>The queues available when the agent selects the Make Call command from the Call Center screen.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of the user ID from the User ID page. This is display only.</td>
</tr>
<tr>
<td>Operator access</td>
<td>The user ID of the agent whose access is being set up. To specify all agents, type an asterisk (*).</td>
</tr>
<tr>
<td>Outbound</td>
<td>When checked, allows agent to make outbound calls. (Outbound calls include direct dialing, taking call events from queues, and manually accessing call events in assigned queues.)</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search queues</td>
<td>The <strong>Queues</strong> to be searched when an agent selects the Next Call command from the Call Center screen. Queues are searched in the order they are entered here, unless the queues are assigned a <strong>Priority group</strong>. For information on the order in which calls are removed from a queue, see the section <strong>How Call Events are Removed from Queues</strong>.</td>
</tr>
<tr>
<td>Training</td>
<td>When checked, agent is in training mode.</td>
</tr>
<tr>
<td></td>
<td>When cleared, the agent is in live mode.</td>
</tr>
<tr>
<td></td>
<td>In training mode, the agent can use the guide without writing or gathering data. As a reminder to the agent, the word Training appears on the status line of guide screens.</td>
</tr>
<tr>
<td>View group queues</td>
<td>When checked, agent can view information in group queues.</td>
</tr>
<tr>
<td>View shared queues</td>
<td>When checked, the agent can view information in shared queues. (Shared queues are all queues except protected queues.)</td>
</tr>
</tbody>
</table>

*Note: This option is not intended for supervisors (users with a **Status level** of 30).*
Changing Operator Access

You can change an agent’s access at any time. Changes take place when the operator accesses the next call for the project. You do not have to reassemble the guide to have changes to operator access take effect.

If the agent is in a call and you clear the Allow operator access property, the agent is able to complete the call but cannot make another call in that project. If the agent tries to access any command other than View Callbacks or View Monitor, the following message is displayed:

YOU DO NOT HAVE ACCESS TO PROJECT: Project Name

The agent is then returned to the Project Selection screen in EDGE Client. The project name is displayed, but the agent is not allowed access to the project.

Deleting Operator Access

To delete the project from the list on the Project Selection screen, the operator access record for the agent must be deleted. (Clearing the

Properties: (continued)

View/select callbacks
When checked, the agent can display the list of scheduled callbacks. (Scheduled callbacks are displayed when the agent selects the View Callbacks command from the Call Center screen.)
To delete an operator access, select the following:

Project / Setup / Operator Access / the agent’s user ID / Delete

You can access the following command from the Operator Access menu on the Setup menu:

- List operator access.

The Operator Access report provides information about the access of each user for the project. For security reasons, user passwords are not displayed. You can send the report to any server or workstation printer or display it on the terminal.

The report is similar to the following:
**Properties:**

- **Access**
  - If the agent has access to the project, displays Y.

- **Callback display**
  - If the agent can display information related to the call, displays Y.

- **Inbound**
  - If the agent can receive inbound calls, displays Y.

- **Manual only**
  - If the agent can only make manual calls, displays Y.
Properties: (continued)

- **Manual queues**: The list of queues to be searched when the agent selects the Make Call command from the Call Center screen.
- **Outbound**: If the agent can make outbound calls, displays Y.
- **Search queues**: The list of queues to be searched when an agent selects the Next Call command from the Call Center screen.
- **Train**: If the agent has training access to the project, displays Y.
- **User ID**: The user ID of the agent.
EDGE Client Options, Labels, and Schemes

The EDGE Client environment can be set up at the following levels in EDGE:

- EDGE level.
- Project level.
- Operator access level.

For information about where the EDGE client environment is set up at these levels, see Table 3-1.

The order of precedence used is the following:

1. Operator access level.
2. Project level.
3. EDGE level.

For descriptions about EDGE Client options, labels, and schemes, see the section EDGE Client Environment Settings.
Project Options

The Project Options allows you to specify or change options for the project. These options should be set before the guide becomes operational and should not be reset after the guide goes to operations. To specify the options for the project, select the following:

Project / Setup / Project Info / Options

Properties:

- **Allow operator access**
  - When checked, agents have access to the project. Agents must also be given access to the project through Operator Access or Groups.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Counter</td>
<td>The list of Counters that can be used with the project. If blank, the default counter, which has the same name as the project, is used.</td>
</tr>
<tr>
<td>Created by</td>
<td>The user ID, date, and time the project was created. This is display only.</td>
</tr>
<tr>
<td>Description</td>
<td>The description from the Projects screen. This is display only.</td>
</tr>
<tr>
<td>Display preferred calling hour warning message</td>
<td>A warning message is displayed when an agent tries to Make an Outbound Call that is not within the specified preferred calling hours.</td>
</tr>
<tr>
<td>Gather guide performance</td>
<td>For ASCII applications, gathers screen response time in terms of the number of times each screen is accessed and the average time to access each screen. This property has no effect in EDGE Client.</td>
</tr>
<tr>
<td>Gather sales analysis data</td>
<td>Data is gathered on units and revenue. This data is displayed on the Project Monitor screens in operations.</td>
</tr>
</tbody>
</table>
Properties: (continued)

- **When checked, the system automatically writes a record to the REPORTING file at the end of each call. However, to update the REPORTING record with product, quantity, or dollar information before it is written you must use the **SR.TRANS** verb in a logic flow. If checked, but the called is saved to a result that does not update the **REPORTING File**, the data is not gathered.**

  If the setting of this option is changed, you should reassemble the guide.

- **Manual calling only**

  When checked, agents are restricted to outbound manual calling and scheduled callbacks cannot be retrieved.

- **Short description for agents**

  A one-line description to be displayed on the Project Selection screen in EDGE Client.
Version Control

Version control is a means to manage change. It ensures that only one person at a time can make changes to records, while providing a record of changes that were made, who made the changes, and when. If necessary, you can use the record of changes to revert to an earlier version.

For example, under version control, if you want to work on a logic flow, you can check it out while you make your changes. Once you check it out, no one else can work on that logic flow until you check it back in. When modifications to the logic flow are finished, you can check it back in, which increments its version number and records all the changes made to the record. The next person to work on the logic flow can now check it out. When it is checked back in, its version number is again incremented and all changes recorded. If at some future time, it is decided that an older version of the logic flow is really the correct one, you can revert to that version by checking it out then checking it back in, at which point it becomes the latest version.

The EDGE version control module provides the following tools:

- Checking Files In and Out.
- Reporting on Changes to Files.
• **Bumping Versions.**
• **Storage of Metadata.**

Version control can also be invoked from ACL. For more information, see the section `vercont`.

The areas and levels for which version control is provided are listed in Table 2-1. The table also lists the EDGE filename for the area.

**Table 2-1. Version Control Areas**

<table>
<thead>
<tr>
<th>Area</th>
<th>Level</th>
<th>EDGE Filename of Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic flows</td>
<td>Project</td>
<td>LF.SOURCE</td>
</tr>
<tr>
<td>Results</td>
<td>Project</td>
<td>RESULTS</td>
</tr>
<tr>
<td>Screens</td>
<td>Project</td>
<td>SCREENS</td>
</tr>
<tr>
<td>Tasks</td>
<td>EDGE and project</td>
<td>TASKS</td>
</tr>
</tbody>
</table>

Before you can use version control, you must set it up as described in the next section.

Version control can be set up for one or more projects. To set up access, select the following:

   EDGE / Server Area / Version Control Setup
To put all projects under version control, including any project created in the future, select (All). Otherwise, select the appropriate projects. The projects remain highlighted. If you wish to disable version control from a selected project, click on the item to remove the highlight.

Where available, EDGE uses the standard UNIX SCCS version control utility to manage its version control. On systems where SCCS is not available, for example, on Windows NT systems, EDGE can be configured to use the RCS utility. The RCS utility can be downloaded for free from the internet.
To configure EDGE to use RCS, set the `VERCONT_PGM` and `VERCONT_PATH` environment variables. If neither SCCS nor RCS is available, version control should not be used.

The first time version control is used, a directory called sccs (or rcs if using that version control utility) is created in the project directory. Within that directory, a subdirectory is created for each area that has records checked in. The subdirectory is given the name of the EDGE file that contains the record. Each record then becomes a file entry in the appropriate area directory. A sample structure using sccs is shown in Figure 2-3.

![Version Control Directory Structure](image)

**Figure 2-3.  Version Control Directory Structure**
Checking Files In and Out

Checking files in and out through version control provides the following major benefits:

- Facilitates multiuser environments by preventing concurrent changes by two or more people.
- Provides a history of changes.

Once a project is placed under version control, any time a record is opened for modification in one of the areas tracked by EDGE (logic flows, results, screens, or tasks), you can use the check in and out features. Checking in a record places an entry in the SCCS system and allows the record to be opened for modification by any user. Checking out a record gives you exclusive rights to changing that record.

Generally, a record is checked in only when you are satisfied it is working correctly.

Caution! When you assemble, EDGE uses the latest saved version of each logic flow, screen, and result, regardless of whether it is currently checked out or being modified. Therefore, before
swapping, be sure the correct versions were used for the assembly.

When you open a record that has previously been checked in, a page similar to the following is displayed:

**Properties:**

**Area**  
The version control area. Display only.
Properties: (continued)

Do not overwrite EDGE record

If checked and if the Check Out command button is selected, the record is marked as checked out in SCCS, but is not written into EDGE. This is intended to be used when you have modified a record without first checking it out, and now wish to check it in. (Once a record is checked in, it must be checked out before it can be checked in again.).

If the No Check Out command button is selected, this has no effect

Command buttons

The following command buttons are available on this page:

- Read Only: Records are opened in read-only mode and available to others. Any changes made are not saved unless you use the Save Object As command from the File menu.

- Check Out: Checks out the highlighted records and opens the record selected to be modified. If the record to be modified is not highlighted, it is not checked out.

- No Check Out: Opens the record without checking it out.

- Cancel: Cancels the current operation and returns to the previous page.
Properties: (continued)

- **Records**: The list of records in the current area that can be checked out. One or more records can be selected.

- **Versions**: The versions of the currently selected record. If more than one record is selected, only the most recent version can be checked out.

When you exit a record either that you have checked out or that has never been checked in (new records), a page similar to the following is displayed:

![Version Control Check In dialog box]

Properties:

- **Area**: The version control area. Display only.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comments</td>
<td>Information about the record being checked in. This is optional. These comments are displayed in the version control report.</td>
</tr>
<tr>
<td>Records</td>
<td>The records that are available to be checked in. This includes all the records you have checked out as well as any records that have never been checked in.</td>
</tr>
<tr>
<td>Undo check out</td>
<td>Cancels the previous check out.</td>
</tr>
<tr>
<td>Version</td>
<td>The version the currently selected record will be saved with. Display only.</td>
</tr>
</tbody>
</table>

### Opening Checked Out Records

If someone attempts to open a record you have checked out, a message similar to the following is displayed:

![ETW for Windows dialog](image)
Deleting Records

Once a project is placed under version control, any time a record is deleted from one of the areas tracked by EDGE (logic flows, results, screens, or tasks), the following happens:

- If the record is not already checked out, it is checked out.
- A blank version of the record is checked in.
- The EDGE record is deleted.

To restore a deleted record, do the following:

1. From EDGE Developer, select any record in the area that contains the deleted record.
2. From the list of records to check out, select the deleted record.
3. The record you originally selected in step 1 is displayed. Close it.
4. From the list of records to check in, select the version of the deleted record you want to restore. Do NOT select the latest version--this is an empty record.
5. Click OK.
6. At this point, the name of the record is displayed in the list of records you can edit.
Reporting on Changes to Files

The first time a record is checked into version control, it is copied to a file in the appropriate directory. (The directory structure is described in the section Setting up Version Control.) Each time it is checked in, changes are noted in the file, along with the version number and user ID of the person making the changes. You can view the current status of the records or compare two versions by Version module in the Reports area.

EDGE or Project / Reports area / Version tab

A page similar to the following is displayed:

![SOFTSELL (Reports) interface](image)
The following commands are available:

- Check-in/Check-out Report.
- Comparison Report.

**Check-in/Check-out Report**

When this command is selected, a page similar to the following is displayed:

![Image of Check-in/Check-out Report window]

**Properties:**

- **Area**: The area that contains the records on which to report; can be one of the following:
Properties: (continued)

- All
- Logics
- Results
- Screens
- Tasks

Latest version only
If selected, only the current status of the selected records is reported on.

Records
The records on which to report. To report on all records, select All.

If the specified Area is All, no records are listed in this box.

Report on
Determines the type of activity to report on; can be one of the following:

- Versions checked in
- Versions checked out
- All

Users
The IDs of users to report on. More than one user can be selected. To report on all activity, select All.

The report is similar to the following:
Comparison Report

This report can be used to compare two versions of a record. When the Comparison Report command is selected, a page similar to the following is displayed:
Only those properties that have changed are listed in the report. The report is similar to the following:

**Properties:**

- **Area**: The area that contains the records to compare; can be one of the following:
  - Logics
  - Results
  - Screens
  - Tasks

- **Record**: The record to be compared.

- **Version 1**: The version to compare.

- **Version 2**: The version to be compared against.
The properties that are compared depend on the area. The properties are listed in the following tables:

- Logics. See Table 2-2.
- Results. See Table 2-3.
- Screens. See Table 2-4.
• Tasks. See Table 2-5.

**Table 2-2. Logic Properties That Are Compared**

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last updated by</td>
</tr>
<tr>
<td>Last updated on</td>
</tr>
<tr>
<td>Last updated at</td>
</tr>
<tr>
<td>Last assembled by</td>
</tr>
<tr>
<td>Last assembled on</td>
</tr>
<tr>
<td>Last assembled at</td>
</tr>
<tr>
<td>Default file</td>
</tr>
<tr>
<td>Logic statements</td>
</tr>
</tbody>
</table>

**Table 2-3. Results Properties That Are Compared**

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result group</td>
</tr>
<tr>
<td>Callback arranged by</td>
</tr>
<tr>
<td>Added days for system arranged callbacks</td>
</tr>
<tr>
<td>Added hours for system arranged callbacks</td>
</tr>
<tr>
<td>Added minutes for system arranged callbacks</td>
</tr>
<tr>
<td>Reset recall count</td>
</tr>
</tbody>
</table>
### Table 2-3. Results Properties That Are Compared (continued)

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Send to queue</td>
</tr>
<tr>
<td>System arrangement (Calendar or Work)</td>
</tr>
<tr>
<td>Do NOT update HISTORY file</td>
</tr>
<tr>
<td>Do NOT update REPORTING file</td>
</tr>
<tr>
<td>Do NOT update Global and project monitors</td>
</tr>
<tr>
<td>Description</td>
</tr>
</tbody>
</table>

### Table 2-4. Screen Properties That Are Compared

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screen type</td>
</tr>
<tr>
<td>Title</td>
</tr>
<tr>
<td>Last updated</td>
</tr>
<tr>
<td>Rebound</td>
</tr>
<tr>
<td>Inactive</td>
</tr>
<tr>
<td>Default filename</td>
</tr>
<tr>
<td>Suppressed screen commands</td>
</tr>
<tr>
<td>Allowed screen commands</td>
</tr>
<tr>
<td>Function key logics</td>
</tr>
</tbody>
</table>
### Table 2-4. Screen Properties That Are Compared (continued)

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function key description</td>
</tr>
<tr>
<td>Function key route screen</td>
</tr>
<tr>
<td>Function key execute exit screen logic</td>
</tr>
<tr>
<td>Function key result</td>
</tr>
<tr>
<td>During screen logics</td>
</tr>
<tr>
<td>Before screen logics</td>
</tr>
<tr>
<td>Exit screen logics</td>
</tr>
<tr>
<td>Screen validation logics</td>
</tr>
<tr>
<td>Activation logic</td>
</tr>
<tr>
<td>Deactivation logic</td>
</tr>
<tr>
<td>Selection screen numbers</td>
</tr>
<tr>
<td>Selection screen text</td>
</tr>
<tr>
<td>Selection screen routes</td>
</tr>
<tr>
<td>Selection screen ranks</td>
</tr>
<tr>
<td>Selection screen results</td>
</tr>
<tr>
<td>Maximum number of selections</td>
</tr>
<tr>
<td>Include selections from</td>
</tr>
<tr>
<td>Exclude selections from</td>
</tr>
</tbody>
</table>
**Table 2-4. Screen Properties That Are Compared (continued)**

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Always show selection numbers</td>
</tr>
<tr>
<td>Rotate selections</td>
</tr>
<tr>
<td>Rotate first n choices</td>
</tr>
<tr>
<td>Translation tables</td>
</tr>
<tr>
<td>Translation table fields</td>
</tr>
<tr>
<td>Show title</td>
</tr>
<tr>
<td>Show row labels</td>
</tr>
<tr>
<td>Show function keys</td>
</tr>
<tr>
<td>Screen metric</td>
</tr>
<tr>
<td>Screen width</td>
</tr>
<tr>
<td>Screen height</td>
</tr>
<tr>
<td>Physical metric</td>
</tr>
<tr>
<td>Screen style</td>
</tr>
<tr>
<td>Function key style</td>
</tr>
<tr>
<td>Selection area style</td>
</tr>
<tr>
<td>Function key layout</td>
</tr>
<tr>
<td>Hot key layout</td>
</tr>
<tr>
<td>Hotspot table</td>
</tr>
</tbody>
</table>
### Table 2-4. Screen Properties That Are Compared (continued)

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects</td>
</tr>
<tr>
<td>Object type</td>
</tr>
<tr>
<td>Top</td>
</tr>
<tr>
<td>Left</td>
</tr>
<tr>
<td>Width</td>
</tr>
<tr>
<td>Height</td>
</tr>
<tr>
<td>Visible</td>
</tr>
<tr>
<td>Enabled</td>
</tr>
<tr>
<td>Object name</td>
</tr>
</tbody>
</table>

### Table 2-5. Task Properties That Are Compared

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last updated by</td>
</tr>
<tr>
<td>Last updated on</td>
</tr>
<tr>
<td>Last updated at</td>
</tr>
<tr>
<td>Last assembled by</td>
</tr>
<tr>
<td>Last assembled on</td>
</tr>
<tr>
<td>Last assembled at</td>
</tr>
</tbody>
</table>
Bumping is the process that is used to set all currently checked in records to the same version. Bumping usually occurs after all changes and testing have been finished, just before the project is released, but can be used at any time.

**Table 2-5. Task Properties That Are Compared (continued)**

<table>
<thead>
<tr>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title</td>
</tr>
<tr>
<td>Files</td>
</tr>
<tr>
<td>File abbreviations</td>
</tr>
<tr>
<td>Process all records in file</td>
</tr>
<tr>
<td>Process records in selection list</td>
</tr>
<tr>
<td>Process records</td>
</tr>
<tr>
<td>Before task ad hoc reports</td>
</tr>
<tr>
<td>After task ad hoc reports</td>
</tr>
<tr>
<td>Send before task reports to printer</td>
</tr>
<tr>
<td>Send after task reports to printer</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Chain to task</td>
</tr>
<tr>
<td>Task statements</td>
</tr>
</tbody>
</table>

**Bumping Versions**

Bumping is the process that is used to set all currently checked in records to the same version. Bumping usually occurs after all changes and testing have been finished, just before the project is released, but can be used at any time.
EDGE version control assigns numbers to records using the following format:

b.n

b   The bump version. This number is assigned when you perform a bump.

n   A sequential number that is set to 1 each time the version is bumped. This number is incremented each time a record is checked in.

To bump records at the project level, select the following:

   Project / Setup area / Version Bump

To bump records at the EDGE level, select the following:

   EDGE / Guide area / General tab / Version Bump

In either case, a page similar to the following is displayed:
Properties:

Area

The area to be bumped; can be one of the following:

- All
- Logics
- Results
- Screens
- Tasks
If a record is currently checked out, it cannot be bumped. A message similar to the following is displayed:

![Message](image)

**Properties: (continued)**

- **New version**  The version to bump to. This must be an integer greater than the highest checked in version in the selected area. For example, if there is a logic flow at version 1.2 and a screen at version 4.3 and All is selected as the area, the lowest version that can be specified is 5. If, in this example, Logics is the area selected, the lowest version that can be specified is 2. The highest number that can be specified is 99.

- **Records**  The records to be bumped. To bump all records in an area, select All.

  If the selected Area is All, no records are listed.

**Storage of Metadata**

Metadata is information about a record. Each project has a file called VER.CONT, in which current information about version control is saved. The information saved in this file is listed in Table 2-6.
### Table 2-6. Metadata in VER.CONT File

<table>
<thead>
<tr>
<th>Field</th>
<th>FieldName</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>ID</td>
<td>The record ID, which is formed by concatenating the EDGE area filename with the record ID. For example, if you have a logic flow called CHANGE.LBL, it has the record ID LF.SOURCE-CHANGE.LBL.</td>
</tr>
</tbody>
</table>
| 1     | STATUS    | The current status of the record; can be one of the following:  
  - D  The record has been deleted.  
  - I  The record is checked in.  
  - O  The record is checked out. |
| 2     | VERSION   | The current version number. |
| 3     | DATE.TIME | The date and time the record was last modified, in external format. For example, if the STATUS is I, this contains the date and time the record was checked in. |
| 4     | USER      | The ID of the user that performed the last modification. |
Coded Project Selection

Coded Project Selection allows agents to select a project from the Project Selection screen in EDGE Client by selecting the Project Code command, then entering the code for the desired project.

The following steps are required to set up coded project selection:

- Specifying Project Codes.
- Enabling Coded Project Selection.

You do not have to assemble the guide to use this feature.

Specifying Project Codes

The Coded Project Selection property sheet is used to specify codes for projects. You can associate more than one code with each project. For example, if you associated the Softsell project with the codes named REBATE and 800-2600, an agent could access the project with either code.

To create a code, select the following:

   EDGE / Telephony / Coded Project

A dialog box similar to the following is displayed:
Enabling Coded Project Selection

Coded Project Selection is enabled through the Telephony Configuration page. To display this page, select the following:

- EDGE / Telephony / Options / Configuration

A dialog box similar to the following is displayed:

**Properties:**

- **Coded project selection setup**: The code to be associated with the project selection; can be up to 10 characters in length.
  - Codes are case-sensitive. For example, the system will consider REBATE and rebate as different codes.
  - The code can include spaces. For example, TEST STUDY is a valid code.

- **Project name**: The project associated with the code.
Coded project selection.

Properties:

- Allow TSR into guide without operator access record
- Enable coded project selection
- Route to $MAIN screen at end of guide

For more information, see your EDGE Telephony Gateway documentation.

When checked, coded project selection is enabled.

When checked, agent is routed to the Project Selection screen at the end of the guide. Only available when Enable coded project selection is checked.
When the agent selects the Project Code command in the Project Selection screen in EDGE Client, a message box similar to the following is displayed:

![Project Code dialog box]

When cleared, agent is returned to the Call Center screen for the project.

Select project base on

For more information, see your EDGE Telephony Gateway documentation.

Telephony switch field

For more information, see your EDGE Telephony Gateway documentation.

Use telephony automatic inbound feature

For more information, see your EDGE Telephony Gateway documentation.

Use telephony automatic outbound feature

For more information, see your EDGE Telephony Gateway documentation.

Properties: (continued)
If the code entered by the agent has been associated with a project, the first screen of the project is displayed. If the agent enters an incorrect code, a message box similar to the following is displayed:

![Dialer Message](image)

User Access Error!

Message from host:
THE CALLER HAS DIALED:
THERE IS NO PROJECT DEFINED FOR THIS NUMBER.

OK
Graphic images for command buttons can be stored on the client or the server. Saving the graphic image on the server eliminates the need to copy and maintain graphic images on each EDGE Client.

If the server is used to store the image, EDGE Client will read the graphic data from the server when a screen containing the graphic is loaded. To minimize the amount of network traffic, each graphic image is stored in a cache on the agent’s PC.

To save graphic images on the server, select the following:

   Project / Guide / Server Images tab

A page similar to the following is displayed:
white text
This chapter describes areas that need to be set up in EDGE Developer; for example, creating user IDs.

The following areas are covered in this chapter:

- **Client Settings**.
- **Bypass EDGE Login through UNIX**.
- **Country Calling Setup**.
- **EDGE Client Environment Settings**.
- **Create Project Definitions**.
- **Server Setup**.
- **SuperServer Setup**.
- **User Setup**.
Client Settings

The Setup section is used to set preferences for your PC such as client/server connection information, log on options, debugging options, toolbar icon size, and caching. The information is saved in the ETWW.INI file.

To display the Setup page, select the following:

   EDGE / File menu / Setup

The tabs available on the Setup page are the following:

- Client Setup.
- Connection Setup.

Client Setup

When you add or modify the setup, a Client Setup page similar to the following is displayed:
Properties are described alphabetically within groups.

**Properties:**

At login

The At login group box contains properties that can be set to affect the log on process.
<table>
<thead>
<tr>
<th><strong>Properties:</strong> (continued)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Log on at startup</strong></td>
<td>Specifies whether the user is logged on automatically when the connection to the EDGE server is made. If checked, the user is automatically logged on to EDGE. The <strong>EDGE login</strong> information on the Connection Setup tab must be set in order to use this feature.</td>
</tr>
<tr>
<td><strong>Show splash screen</strong></td>
<td>If checked, the splash screen is displayed when the connection to EDGE occurs. The splash screen is cleared by the following.</td>
</tr>
<tr>
<td><strong>Cache lists</strong></td>
<td>Specifies whether the lists of objects are to be updated from the server whenever an object of that type is requested.</td>
</tr>
<tr>
<td><strong>Timer on splash screen (in seconds)</strong></td>
<td>The splash screen is cleared after the specified number of seconds has passed. The default is five seconds.</td>
</tr>
<tr>
<td><strong>Remove splash screen on mouse click</strong></td>
<td>The splash screen is cleared when the mouse is clicked.</td>
</tr>
</tbody>
</table>
Properties: (continued)

If checked, lists are read from the server when the user first logs on to EDGE, then again when the Refresh tool on the toolbar is clicked. Changes made on the local client are reflected in the list; for example, if you add a field, that field is added to your list and to the server, but not to any other client that has the cache lists box checked.

If cleared, lists are updated from the server as required. For example, if a field is added to a file, and then a screen object is defined that requires a fieldname from that file, the list of fields is updated.

By default, the box is cleared.

The state of this box can have an effect on performance. If cleared, each request for an object requires EDGE go out to the server to update the list.

If the lists of objects are fairly stable, that is, new objects are not being created, checking the box is recommended because of the increase in performance.

The following areas are affected:

Files.
Fields.
Logic flows.
Screens.
Styles.
### Client Settings

#### Properties: (continued)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Client trace</strong></td>
<td>The Client trace group box is used to set up a file to capture all the functions called by EDGE. Client trace is intended to assist EDGE Customer Care in debugging problems should they occur. It can have a significant impact on performance and disk space on the client PC and should not be used unless requested by EDGE Customer Care.</td>
</tr>
<tr>
<td>Append to client file</td>
<td>If checked, appends information to the file. If cleared, the file is cleared each time the trace starts.</td>
</tr>
<tr>
<td><strong>Path</strong></td>
<td>The name of a text file in which the trace is written to.</td>
</tr>
<tr>
<td><strong>Start client trace</strong></td>
<td>If checked, all the functions called by EDGE as determined by the trace level are written to the specified file.</td>
</tr>
<tr>
<td><strong>Trace level</strong></td>
<td>The level of detail provided by the trace, with one being the least detailed.</td>
</tr>
<tr>
<td><strong>EDGE</strong></td>
<td>The EDGE group box is used to set up the connection to EDGE Client for testing the guide.</td>
</tr>
<tr>
<td><strong>Password</strong></td>
<td>The password to the server, if any. For security purposes, the characters you type are shown as asterisks (*).</td>
</tr>
</tbody>
</table>
Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>The complete path to the EDGE client application. To browse for the correct path, click the three-dot button. This path is also used by the Project Operations command in the EDGE Guide area on the General tab.</td>
</tr>
<tr>
<td>Port</td>
<td>The port number used to launch EDGE Client from EDGE Developer using a socket connection.</td>
</tr>
<tr>
<td>Server login</td>
<td>The login name for the server.</td>
</tr>
</tbody>
</table>

Note: Before the guide can be tested, you must check the Allow multiple sessions property in the Users area.

Server trace

The Server Trace group is used to set up the UNIX stderr file and to start a trace on the server.

Server trace is intended to assist EDGE Customer Care in debugging problems should they occur. It should not be used unless requested by EDGE Customer Care.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>Path to the stderr file.</td>
</tr>
<tr>
<td>Start trace</td>
<td>If checked, all the commands that go to the server are written to stderr file specified by the path.</td>
</tr>
</tbody>
</table>
**Connection Setup**

The Connection Setup page is used to specify a connection between your workstation and the server.

To display the Connection Setup page, select the following:

EDGE / File menu / Setup / Connection Setup tab

A page similar to the following is displayed:

**Properties: (continued)**

**Toolbars**

The Toolbars group box is used to specify the size of icons on the EDGE Developer toolbar; can be one of the following:

- Large icons
- Small icons

It is recommended that you do not change the icon size while connected to EDGE Developer. Doing so causes a problem with the connection icon and option on the File menu; to correct this, revert to the original setting.
Properties are described alphabetically within groups.

**Properties:**

**EDGE login**  The EDGE login group box is used to specify user information to be used on the login screen. This is optional.

The group includes the following:

- **Password**  The password, if any, associated with the user ID.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ID</td>
<td>The user ID to log on to EDGE.</td>
</tr>
<tr>
<td>Workstation ID</td>
<td>The workstation ID setup for the current user. This is intended for use only when testing telephony features using a guide.</td>
</tr>
<tr>
<td>Workstation ID setup</td>
<td>For more information on workstation setup, see your EDGE Telephony Gateway documentation.</td>
</tr>
<tr>
<td>Server</td>
<td>The Server group box is used to specify server information. The group includes the following:</td>
</tr>
<tr>
<td>Connection timeout</td>
<td>The number of seconds to try to make a connection with the server before failing.</td>
</tr>
<tr>
<td>Login</td>
<td>The server login to be used when a connection is attempted.</td>
</tr>
<tr>
<td>Password</td>
<td>The server password, if any.</td>
</tr>
<tr>
<td>Setup files</td>
<td>The Setup files group displays the files that can be used for a connection. You can also add or delete a file.</td>
</tr>
<tr>
<td>Basic Connection Settings</td>
<td>These specify the IP address and port for the connection. You can specify a primary and secondary connection.</td>
</tr>
<tr>
<td>Primary Host</td>
<td>Hostname or IP address of the server and port (optional).</td>
</tr>
<tr>
<td>Properties: (continued)</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Secondary Host</td>
<td>Hostname or IP address of the secondary server and port (optional)</td>
</tr>
<tr>
<td>Swap Primary and Seco-</td>
<td>If clicked, the IP address for the primary server is swapped with the secondary server.</td>
</tr>
<tr>
<td>ndary servers button</td>
<td></td>
</tr>
<tr>
<td>Secure Communications</td>
<td>This group specifies that secure communications be used with the server. This is an optional feature and requires additional licensing.</td>
</tr>
<tr>
<td>Secure communications</td>
<td>When this checkbox is set (checked), the Private Key File and Cipher properties are enabled. By selecting this property EDGE Developer will establish a secure SSH connection to the EDGE server. If left unchecked, EDGE Developer will use a telnet or socket based connection to the EDGE Server. For information see Encrypted Communications.</td>
</tr>
<tr>
<td>using encryption check-</td>
<td></td>
</tr>
<tr>
<td>box</td>
<td></td>
</tr>
<tr>
<td>Private Key File</td>
<td>The filename of the RSA/DSA private key file for authentication</td>
</tr>
<tr>
<td>Cipher</td>
<td>The type of encryption cipher to be used.</td>
</tr>
</tbody>
</table>
**New Connection**

The following steps allow you to create a new file with connection information:

1. Click the Add command button on the Connection Setup tab.

2. Enter the name of the new INI file in the text box (next to the Use as default check box).

---

Note: The name cannot be ETWGEO.INI; this is restricted for internal use.

---

3. Enter the appropriate IP address and port (use port 23 for all Telnet sessions).

4. Enter the appropriate server login ID and password, if any.

5. To log on to ETW–W automatically when you connect to the server, enter the EDGE login user ID and password.

6. To use this setup the next time you connect to EDGE Developer, check the Use as default check box.

7. Click OK.
### Bypass EDGE Login through UNIX

The EDGE Login screen can be bypassed using the UNIX security system. Setup is required through UNIX and EDGE.

In addition to allowing security to be kept at the UNIX level, it also allows passwords to be created and maintained at the UNIX level.

#### UNIX Requirements

The following steps are the UNIX requirements for a user to bypass the EDGE login process:

1. The user ID must have a group ID of 57. (For more information, contact your system administrator.)

2. The user’s PATH variable must include the EDGESYS/bin directory. Enter this information in the user’s .profile file.

3. User information must be entered in the /etc/passwd and /etc/group files for the user.

#### EDGE Requirements

The following are the EDGE requirements for a user to bypass the EDGE login process:

1. Set the following two environment variables:
   - **EDGENAME**: The name of the EDGE login account.
   - **LOGPATH**: The server to log on to. The possible values are: edge, etw, and edgeo.
2. The user must have an EDGE user ID. The user ID determines the level of access the user has within EDGE. The EDGE user ID is case sensitive and must match the case of the UNIX login ID.

3. The password for the EDGE user ID is ignored when the UNIX bypass is used. The password is checked only when EDGE is directly logged on to.

4. Setting the following two environment variables are optional:

   **EDGETERM**
   
   For ASCII only. If the terminal type prompts the user to enter a type, fills in the specified terminal type.
   
   If not set, the user is prompted for the terminal type.

   **EDGESTATION**
   
   Overrides the default workstation set in the Port Definition record.
   
   If not set, uses the default workstation ID as defined in the Port Definition record.

5. To invoke EDGE and bypass the EDGE login screen, use the script called runedge located in the EDGESYS/bin directory. The user can enter the command at the shell prompt or it can be placed at the end of the user .profile file.
**Country Calling Setup**

This section is used to specify information for making international calls, and adding time zone information.

To display the Setup page, select the following:

EDGE / Setup area

A page similar to the following is displayed:
Country Code

The Country Code page is used to define and modify information for your country site and to add time zone information for callbacks to locations outside the United States.

System settings for the following site properties can be specified:

- Date and time format.
- Daylight savings time.
- Description.
- Monetary.
- Telephone number.

For example, the date format can be modified to reflect the international format of day/month/year. When the date format is modified, guides do not need to be reassembled. However the verbs.lst file located in the EDGE Developer installation folder must deleted and reinstalled for changes to take effect.

Note: For information on specifying languages, see the document Localizing EDGE.

If you delete a country code, EDGE checks the ACTZ file and deletes any entries that were defined for that country code. Depending on the size of your ACTZ file, this may take several minutes.
To display the Country Code page, select the following:

EDGE / Setup area / Country Code tab

When you select Add or Modify, a page similar to the following is displayed:
Properties are described alphabetically within groups.

**Properties:**

- **Country code name**
  - The name that identifies the country code.

- **Date and time format**
  - The format for the date and time. The following properties can be specified:
    - **Date Order**
      - The order in which to display the month, day, and year; can be one of the following:
        - DMY (Day, month, year).
        - DYM (Day, year, month).
        - MDY (Month, day, year).
        - MYD (Month, year, day).
        - YDM (Year, day, month).
        - YMD (Year, month, day).
    - **Date separator**
      - The date separator; can be a space, a single character, or a slash (/). The default is a slash (/). For example, if you specify the date separator as a period (.) and the date format as DMY, the default numeric date format would be dd.mm.yy.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Day/month style</th>
<th>The capitalization rules for displaying days of the week and months; can be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Days and months are displayed in English and with all uppercase letters. This is the default for systems upgraded from a release of EDGE prior to 3.5.</td>
</tr>
<tr>
<td>Capitalized</td>
<td>Days and months are displayed in the language specified in the language setup and with the capitalization of the server operating system. This is the default for new installations.</td>
</tr>
<tr>
<td>Uppercase</td>
<td>Days and months are displayed in the language specified in the language setup and with all uppercase letters.</td>
</tr>
<tr>
<td>Properties: (continued)</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Greenwich mean</td>
<td>The number of hours past Greenwich Mean Time for the site. From the drop-down list box, select the correct time zone. A sampling of cities are included for each time zone.</td>
</tr>
<tr>
<td>Time separator</td>
<td>The time separator; can be a space, a single character, or a colon ( : ). The default is a colon ( : ).</td>
</tr>
<tr>
<td>Use 24 hour time</td>
<td>If checked, specifies that a 24-hour clock is used. 13:00 would always refer to 1 o’clock in the afternoon.</td>
</tr>
<tr>
<td>Daylight savings time</td>
<td>The start and end dates of daylight savings time for three years. If two digit years are used, a base year of 30 is used. This means that the years between 30 and 99 are in the 20th century (1930 - 1999) and years less than 30 are in the 21st century (2000 - 2029). This property is not affected by the EDGE base year property in Site Setup.</td>
</tr>
<tr>
<td>Offset</td>
<td>The number of hours to be set ahead for daylight savings time.</td>
</tr>
<tr>
<td>Description</td>
<td>The description is an optional field that you can use to describe your country code.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Monetary</th>
<th>The monetary terms and symbols to use within EDGE. The following properties can be specified:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decimal separator</td>
<td>The symbol to use as a decimal separator when numbers are entered. For example, in the United States, this is a period ( . ). If no character is specified, a period is used.</td>
</tr>
<tr>
<td>Monetary symbol</td>
<td>The symbol for currency. For example, in the United States, this is a dollar sign ( $ ). If no monetary symbol is specified, $ is used.</td>
</tr>
<tr>
<td>Monetary term</td>
<td>The term for the primary rate of currency. For example, in the United States, this is the word dollar. If no term is specified, the word dollar is used.</td>
</tr>
<tr>
<td>Thousands separator</td>
<td>The character used to separate thousands. For example, in the United States, this is a comma ( , ). If no character is specified, a comma is used.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thousands separator term</td>
<td>The term for the character used to separate thousands. For example, in the United States, this is the word comma. If no term is specified, the word comma is used.</td>
</tr>
<tr>
<td>Telephone number</td>
<td>Specifies telephone number details. The following properties can be specified:</td>
</tr>
<tr>
<td></td>
<td>The maximum number of digits in a phone number, including area code. For example, in the United States, the maximum number is ten.</td>
</tr>
<tr>
<td></td>
<td>The minimum number of digits in a phone number, including area code. For example, in the United States, the minimum number is ten.</td>
</tr>
</tbody>
</table>
### ACTZ File Update

The ACTZ file contains records based on the U.S. area code/telephone number prefixes. During **Queue Seeding**, the ACTZ file can be used for telephone number validation.

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Leading digits</strong></td>
<td>The number of digits in the telephone number to use to validate against the ACTZ file. More than one value can be specified. For example, in the United States, the ACTZ file contains six- and three-digit codes. To validate six digits first, specify six. Then, to validate three digits in the cases where there is no match for six, specify three. (The format of your ACTZ file determines the values to be used here. For more information, contact the supplier of your file.)</td>
</tr>
<tr>
<td><strong>Prepend digits</strong></td>
<td>The digits to prepend to phone numbers when they are sent to the predictive dialer. For example, in the U.S., a 1 is often prepended. Only used with the AIT EDGEDial Plug-In.</td>
</tr>
</tbody>
</table>
The ACTZ page allows you to update information for a record or add a record to the ACTZ file. You should do so only when directed by EDGE Customer Care.

You can also update the ACTZ file with records from another file. For more information, see International ACTZ File Update.

To display the ACTZ page, select the following:

EDGE / Setup area / ACTZ tab

A page similar to the following is displayed:
Properties:

Area code
The name of the ACTZ record. This unique code represents the calling area.

Daylight savings information
If checked, indicates that daylight savings time is observed for the area covered by the record.

State information
The state information. The following must be specified:
International ACTZ File Update

The International ACTZ page allows you to update the ACTZ file with telephone information in another file. You can do the following:

- Define an Update Definition.
- Clear Processed ACTZs.
- Clear Processing Errors.

Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State code</td>
<td>The two letter code for the state or country.</td>
</tr>
<tr>
<td>State name</td>
<td>The name of the state or country.</td>
</tr>
<tr>
<td>State number</td>
<td>The state number. The numbers listed are based on the alphabetical listing for the state name.</td>
</tr>
<tr>
<td>Time zone information</td>
<td>The time zone information. The following must be specified:</td>
</tr>
<tr>
<td>Time zone code</td>
<td>The code for time zone. These codes are based on the time zone name.</td>
</tr>
<tr>
<td>Time zone name</td>
<td>The name for the time zone.</td>
</tr>
<tr>
<td>Time zone number</td>
<td>The number of the Greenwich Mean Time (GMT) zone.</td>
</tr>
</tbody>
</table>
• List ACTZ Processing Errors.

• Process ACTZ Updates.

To display the International ACTZ page, select the following:

EDGE / Setup area / International ACTZ tab

A page similar to the following is displayed:

The Add and Delete command buttons are not enabled on this page. To add or delete a country code, select the Country Code tab.
Define an Update Definition

You can add calling codes and time zone information to the EDGE ACTZ file. To do this, you must have a file that has records with the following information:

- Calling code (in the definition screen, this is referred to as the Area code/prefix field).
- Daylight savings time indicator.
- Location.
- Time zone.

Note: Before you can define an update definition, you must create the file definition and field definitions for your file.

When you define the ACTZ file for a country code, you specify the file and fields to be referenced when the ACTZ file is updated. After the fields have been filled in, you can update the ACTZ file by selecting the Process ACTZ Updates command.

When you select a country code, and select Modify, a page similar to the following is displayed:
Properties:

File
The name of the file that contains the time zone information.

Time zone field
The name of the field that contains the offset in hours from Greenwich Mean Time. All offsets must be positive numbers.

Area code/prefix field
The unique code that identifies the calling area. In the United States this is the area code, and sometimes, the prefix.

Daylight savings time field
The name of the field that specifies if the location uses daylight savings time. A Y (or 1) indicates the location uses daylight savings time.

Location field
The name of the field that contains descriptive text about the calling code, such as city. This property is optional.
Clear Processed ACTZs

The Clear Processed ACTZs command clears records from the ACTZ file. A country code must be selected before this command is enabled.

This command can be selected from the International ACTZ tab or menu. When this command is selected, the following is displayed:

Properties:

- All records: If checked, all the records are cleared.

[Diagram of Clear Processed ACTZs window]
**Properties: (continued)**

- **Records in selection list**
  If checked, allows you to specify the selection list that contains the names of records to clear.

- **Individual records**
  If checked, allows you to specify the names of records to clear.

- **Maximum error count**
  The number of errors that allowed during the clear process.

**Clear Processing Errors**

The Clear Processing Errors command clears the log file that was used by the update process. This command can be selected from the International ACTZ tab or menu.

To **List ACTZ Processing Errors**, select the command from the International ACTZ menu.

**List ACTZ Processing Errors**

The ACTZ processing errors displays the errors that occurred during processing, if any. The log can be displayed in a summary or log format. These commands can be selected from the International ACTZ menu.
Process ACTZ Updates

The Process ACTZ Updates command adds the records to the ACTZ file from the specified file. A country code must be selected before this command is enabled.

This command can be selected from the International ACTZ tab or menu. When this command is selected, the following is displayed:

Properties:

- All records: If checked, all the records are cleared.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Records in selection list</td>
<td>If checked, allows you to specify the selection list that contains the names of records to clear.</td>
</tr>
<tr>
<td>Individual records</td>
<td>If checked, allows you to specify the names of records to clear.</td>
</tr>
<tr>
<td>Maximum error count</td>
<td>The number of errors that allowed during the clear process.</td>
</tr>
</tbody>
</table>
**EDGE Client Environment Settings**

EDGE Client is the interface used to access an EDGE guide using Microsoft Windows 95/98. This section is used to set up the following areas of the EDGE Client environment:

- EDGE Client Options.
- EDGE Client Labels.
- EDGE Client Scheme.

*Note: In previous versions of EDGE, EDGE Client was referred to as GEO.*

The EDGE client environment can be set up at different levels in EDGE. Table 3-1 lists where in EDGE the EDGE Client environment can be set up.

<table>
<thead>
<tr>
<th>Level</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDGE level</td>
<td>EDGE / Server area / GEO Setup</td>
</tr>
<tr>
<td>Project level</td>
<td>Project / Setup area / Project Info tab / GEO Setup</td>
</tr>
<tr>
<td>Operator access level</td>
<td>Project / Setup area / Operator Access tab</td>
</tr>
</tbody>
</table>
**EDGE Client Options**

The GEO options page is used to set up access to features in EDGE Client such as modifying scheme selection.

To display the GEO options page at the EDGE level, select the following:

```
EDGE / Server area / GEO Setup
```

To display the GEO options page at another level, see Table 3-1.

A page similar to the following is displayed:
**Properties:**

- **Allow access modifications**

  If checked, the Access command from the Setup menu in EDGE Client is enabled. The Access command allows you to change the access to some of the commands in the View menu and toolbar in EDGE Client.
<table>
<thead>
<tr>
<th>Setting Up EDGE EDGE Client Environment Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Properties: (continued)</strong></td>
</tr>
<tr>
<td>Allow label modifications</td>
</tr>
<tr>
<td>Allow new window creation</td>
</tr>
<tr>
<td>Allow scheme modifications</td>
</tr>
<tr>
<td>Allow setup modifications</td>
</tr>
<tr>
<td>Case-sensitive member-of-set</td>
</tr>
</tbody>
</table>
### Properties: (continued)

**Data entry text conversion**

The Data entry text conversion specifies how alphabetic characters are to be saved in all fields; can be one of the following:

- **Blank or null**  
  If this property is left blank, the setting used is in the following order of precedence:
  
  1. Operator access level.
  2. Project level.
  3. EDGE level.
  
  4. **Guide Options 1.**

- **Capitalize**  
  The first character of each word is capitalized. All other letters are lowercase.

- **[Field format]**  
  Uses the capitalization specified in the field definitions.

- **Lowercase**  
  All letters are converted to lowercase.

- **None**  
  All fields are saved as entered by the agent. This is the default.

- **Uppercase**  
  All letters are converted to uppercase.
### EDGE Client Labels

**GEO labels** are the text on the command buttons in EDGE Client used to access standard areas in EDGE. The command buttons are located on the Call Center Screen and the Project Selection Screen. These screens are displayed in the section **System Screens**.

To display the GEO Labels page at the EDGE level, select the following:

```
EDGE / Server area / GEO Setup / GEO Labels tab
```

To display the GEO Labels page at another level, see Table 3-1.

A page similar to the following is displayed:

---

**Properties: (continued)**

| Menu items access | The options checked are enabled in the View menu and standard toolbar in EDGE Client. The options are described in the EDGE 7.11 Client Reference Manual. |

---

**Note:** If the *Data entry text conversion* property on the EDGE Client Options tab is set at the operator access, project, or EDGE level, it has precedence over the *Data entry case conversion* property on the Guide Options 1 tab.
The defaults used for the command buttons are listed in Table 3-2.

An ampersand (&) in the label indicates the following letter can be used with the ALT key to access that area. The letter can be entered in either upper or lower case. For example, &Auto-Receive is the default label for the auto-receive area. On that button, the A letter is underlined, indicating that ALT+A accesses the auto-receive area. You can also press ALT+a. In the case of View Call&backs, the letter b is under-
lined on the button, indicating `ALT+b` or `ALT+B` accesses the view call-backs area.

### Table 3-2. Default EDGE Client Labels

<table>
<thead>
<tr>
<th>Areas in EDGE</th>
<th>Default Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auto-dial</td>
<td>Auto &amp;dial</td>
</tr>
<tr>
<td>Auto-receive</td>
<td>&amp;Auto-receive</td>
</tr>
<tr>
<td>Make call</td>
<td>&amp;Make call</td>
</tr>
<tr>
<td>Next call</td>
<td>&amp;Next call</td>
</tr>
<tr>
<td>Open project</td>
<td>&amp;Open project</td>
</tr>
<tr>
<td>Receive call</td>
<td>&amp;Receive call</td>
</tr>
<tr>
<td>View callbacks</td>
<td>View &amp;callbacks</td>
</tr>
<tr>
<td>View monitor</td>
<td>&amp;View monitor</td>
</tr>
</tbody>
</table>
The GEO Scheme page allows you to apply schemes to screens in EDGE Client.

To display the GEO Scheme page at the EDGE level, select the following:

EDGE / Server area / GEO Setup / GEO Scheme tab

To display the GEO Schemes page at another level, see Table 3-1.

A page similar to the following is displayed:
Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| Scheme   | A list of schemes to select from to apply to screens. At the EDGE level, a scheme or the DEFAULT scheme *must* be selected, otherwise you will encounter assembly errors. If this property is left blank or set to None at any level other than the EDGE level, the option set for Scheme on the GEO Scheme page is used in the following order of precedence:  
1. Operator access level. The default is None.  
2. Project level. The default is None.  
3. EDGE level.  
For information on defining schemes, see the section Schemes. |
| Style    | A list of styles in the selected scheme. When a style is selected, its properties are displayed on this tab page. For information on defining style names, see the section Styles. |
Create Project Definitions

A project consists of one guide and all the associated User Files and Fields, Queues, Results, User Screens, Logic Flows, Ad Hoc Reports, and Correspondence for that project that you define. In addition, when you create a project, EDGE creates a set of screens to be used in that project.

After you create a project, you can set up project options such as counters, schedules, and operator access. For more information, see Chapter 2, Setting Up Projects.

To create a project, select New Project from the File menu. A page similar to the following is displayed:
Properties:

Counter: The name of the counter to use as the project counter. If you do not specify a name, EDGE uses a counter with the same name as the project. If there is currently no counter with that name, EDGE creates one using the defaults listed in Table 3-3.

To change Counters used for the project, select Options on the Project Info page in the Setup area.

Created by: The user ID of the person who created the project. This property is display only.

Created on: The project creation date.
**Properties: (continued)**

- **Description**
  A text description of the project. Up to six lines of text can be specified. This property is optional.

- **Project name**
  The name of the project. Do not use names ending in `.c` or `.cpp`. Doing so can cause problems at swap time.

- **Storage**
  The name of the directory in which to locate the project. Once you have created the project, you cannot modify the storage location.

**Table 3-3. EDGE-Created Counter Defaults**

<table>
<thead>
<tr>
<th>Property</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Counter name</td>
<td>project.name</td>
</tr>
<tr>
<td>Allow counter cycling</td>
<td>N</td>
</tr>
<tr>
<td>Database device name</td>
<td>DEFAULT</td>
</tr>
<tr>
<td>Direction</td>
<td>1</td>
</tr>
<tr>
<td>Increment/decrement amount</td>
<td>1</td>
</tr>
<tr>
<td>Initial counter value</td>
<td>1</td>
</tr>
<tr>
<td>Maximum counter value</td>
<td>$10^{27}$</td>
</tr>
<tr>
<td>Minimum counter value</td>
<td>1</td>
</tr>
</tbody>
</table>
Server Setup

The following can be set up on your server:

- General Setup.
- I/O Device Setup.
- Port Setup.
- Printer Setup.
- Server Communication Setup.

To display the Server Setup page, select the following:

EDGE / Server area

A page similar to the following is displayed:
Depending on the tab selected, additional menus may be added to the standard menus. For example, when the General tab is selected, a Server Options menu is added.

When you select the Server area, a Server menu is displayed with the following commands:

- List ad hoc reporting log
- Ad Hoc Reports

The Ad Hoc Reports log contains the name of every ad hoc report that was sent to the printer since the last time the log was cleared.
**General Setup**

The General page is used to do the following:

- Clean EDGE/tmp.
- Clear Ad Hoc Reporting Log.
- Set Up EDGE Client.
- Set Up Screen Defaults.
- Set Up Site Information.
- Set Up SuperServer.
- Set Up Version Control.

Guide performance reports

The guide performance report shows screen response time in terms of the number of times each screen is accessed and the average time to access each screen.

This command affects system performance. Before using it, we recommend you contact EDGE Customer Care. Guide performance reports are normally used at the request of EDGE Customer Care, for example, if you are having system performance problems.

To generate data for the reports, check the Gather guide performance check box in Project Options. Guide performance statistics are gathered only for ASCII terminals. They have no meaning for EDGE Client operations.
These commands are also available on the Server Options menu.

**Clean EDGE/tmp**

The Clean EDGE/tmp command clears the EDGE/tmp directory.

**Clear Ad Hoc Reporting Log**

The Clear Ad Hoc Reporting Log command clears the entries in the ad hoc reporting log.

The ad hoc reporting log contains the name of every Ad Hoc report that was sent to the printer since the last time the log was cleared.

**Clear Guide Performance File**


The guide performance file contains screen response time in terms of the number of times each screen is accessed and the average time to access each screen.
Set Up EDGE Client

The GEO Setup specifies the EDGE level environment settings for EDGE Client.

A page similar to the following is displayed:

![EDGE (GEO Setup) Window]

These properties and tabs are described in the section EDGE Client Environment Settings.
These properties can also be specified for projects and workstations. The settings for a workstation, if any, takes precedence over project and EDGE level specifications. The settings for a project, if any, takes precedence over EDGE level specifications.

For information on changing the EDGE Client environment for individual workstations, see your EDGE 7.11 Client Reference Manual.

Set Up Screen Defaults

The Screen Defaults command allows you to set up EDGE level defaults for screen development in EDGE Developer. These metrics are used during development. At runtime, the EDGE client determines the measurements based on the scheme in use. For this reason, it is important to test screens in an environment similar to that used by your agents.

A page similar to the following is displayed:
Properties are described alphabetically within groups.

**Properties:**

- **ASCII conversions**
  - The number of pixels used by a character in the font specified in the Font property. This is for ASCII use only.
  - **Column width**
    - The column width in pixels.
  - **Row height**
    - The row height in pixels.
**Properties: (continued)**

- **Default measurement**
  - The physical unit of measurement; can be one of the following:
    - Centimeters
    - Inches
    - Millimeters
    - Pixels
    - Points
  - This property is available only if the Default metric property is Physical.

- **Default metric**
  - The default unit of measurement for screens and objects on the screen; can be one of the following:
    - Physical
      - The physical measurement used, such as twips. If specified, select the actual unit using the Default measurement property.
    - Pixels
      - Pixel stands for picture element. A pixel is a single dot on the screen. The actual size depends on the physical monitor.
    - Row/Col
      - Rows and columns use the default font specified to determine height and width.

- **Font**
  - The characteristics of the default font.
    - Name
      - The name of the font.
    - Size
      - The size of the font.
**Setting Up EDGE Server Setup**

**Properties: (continued)**

- Screen height: The default height of screens. The value is based on the Default metric property.
- Screen width: The default width of screens. The value is based on the Default metric property.

---

**Set Up Site Information**

The Site Setup command allows you to identify information about your company’s physical location and other basic site information to be used by EDGE. Generally, site information is entered when you install EDGE at your company site.

**Note:** If you make changes to your site information, after ten days, you will need to enter a reconfiguration authorization code. The appropriate authorization code can be obtained from EDGE Customer Care. If you do not provide a reconfiguration code within ten days of the request, at the end of ten days, no one will be allowed to log on to EDGE, either in development or operations.

A page similar to the following is displayed:
Properties:

**Address**
Your street address. Two lines are provided.
This property is used only for display on the login screen.

**Area code/prefix**
The portion of your telephone number used to match with entries in the ACTZ file to determine your time zones. For example, in the United States, this is the area code and prefix.
Properties: (continued)

This information is used to adjust callback scheduling times based on the time zone and is critical for callback scheduling. Also, if you use the VOAD-DIAL verb, it checks this property to determine if the number being called is in the same location as the site.

City, state

Your city and state.

This property is used only for display on the login screen.

Country calling code

The Country Code for your site. The default is 011, which contains definitions for the United States.

During Queue Seeding, this property is compared to the country code in the Country Calling field or constant country code properties specified in the queue seeding instruction.

Dialer predial code

The codes, if any, that need to be dialed before the phone number. For example, in some companies you must dial 9 to get an outside line. In that case, you would enter a 9 for this property.

This property is used only by the VOAD auto-dialer.

The following codes can be specified in any combination:

- `n` Any digit.
- `;` The dialer is to pause until it receives a dial tone.
- `,` A half-second pause.
Properties: (continued)

The following code indicates that the dialer is to send a 9, wait for the dial tone, send a 1, then wait one second:

```
9;1,,
```

Note: These codes can be specified as part of `VOAD-DIAL`, `VOAD-DIAL-ATT`, and `VOAD-HANGUP` verbs. Codes in the verb take precedence over the verbs specified in the server setup.

| EDGE base year | The base year to use in EDGE if two digit year formats are used. The base year determines if a year entered as two digits is in the 20th or 21st century. Numbers greater than or equal to the base year are stored as years in the 20th century and numbers less than the base year are stored as years in the 21st century.

The base year has no effect if the year is entered with four digits.

For example, assume the base year is 30. This means that years entered as a two-digit number between 30 and 99 are in the 20th century (1930 - 1999) and years less than 30 are in the 21st century (2000 - 2029). Years entered as four digits, such as 2030, are stored as the year 2030. |
After the base year is changed, projects that are used in ASCII operations must be reassembled. All users should log out and back in to EDGE to ensure that everyone is using the new base year.

The default is 0 (all years are in the 20th century). To be effective, this property must be modified before the year 2000. Regardless of base year, the minimum date that can be entered in EDGE is 01/01/100.

**Hide password entry**
If checked, the user’s password is not displayed when entered at login.

**Hide user ID entry**
If checked, the user ID is not displayed when entered at login.

**LDAP URI**
Specifies the server used for LDAP Authentication. This value is ignored if the LDAP URI is specified in the EDGE User Access Definition.

**Limited login attempts**
The number of unsuccessful login attempts before the user is restricted from system access. Unsuccessful login attempts can be restricted from 1 to 9 attempts.

The default is unrestricted.

When the specified number of login attempts is reached, the port is locked; that is, login is disabled for the port. For information on releasing locked ports, see the property **Login disabled**.
SuperServer is used to set up socket connections, which provide an alternative method of communication between the EDGE Server and EDGE Client.

For information on this command, see the section SuperServer Setup.
**Set Up Version Control**

Version control ensures that only one person at a time can make changes to records, while providing a record of changes that were made, who made the changes, and when. For more information, see the section Version Control.

**I/O Device Setup**

I/O devices are used to specify devices used to transfer data. To define an I/O device, select the following:

EDGE / Server area / I/O Dev tab

When you select an existing device or add a new one, a page similar to the following is displayed:
Properties:

Description  A description of the device being defined. This property is optional.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Server path-name</th>
<th>The server pathname to the device being defined.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type</strong></td>
<td>The device type; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>File is used with modules such as archiving, importing, and exporting. It is recommended this <em>not</em> be used as a transport mechanism.</td>
</tr>
<tr>
<td></td>
<td>Internet Socket A transport mechanism used to communicate with servers typically located on another machine; used with the MSGQUEUE verb. For more information, see Chapter 18, Message Queues.</td>
</tr>
<tr>
<td></td>
<td>Message Queue A transport mechanism for interprocess communication; used with the MSGQUEUE verb. For more information, see Chapter 18, Message Queues.</td>
</tr>
<tr>
<td></td>
<td>MQSeries A transport mechanism that links EDGE and IBM’s MQSeries product; used with the MSGQUEUE verb. For more information, see Chapter 18, Message Queues. <em>MQSeries is not supported by EDGE on NT.</em></td>
</tr>
</tbody>
</table>
**Port Setup**

The Port Setup page is used to specify your PC connections to the server.

To set up a port, select the following:

EDGE / Server area / Ports tab

When you add or modify a port name, a page similar to the following is displayed:

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Named Pipe</td>
<td>A transport mechanism for communication between processes running on the same machine; used with the MSGQUEUE verb. For more information, see Chapter 18, Message Queues.</td>
</tr>
<tr>
<td>Old IPC Server</td>
<td>A transport system designed for UNIX IPC message queue servers that work with EDGE 3.6 or earlier.</td>
</tr>
</tbody>
</table>

Named pipes are not supported by EDGE on NT.
Properties:

Auto-dial ok  If checked, the VOAD-DIAL verb can open the auxiliary port.

Login disabled  If checked, you cannot log on to EDGE. If you have set the login attempts in Set Up Site Information to a specific number and that number is reached, EDGE disables login on that port and automatically checks this box. To enable login for the port again, clear the check box.

This property is useful only if your workstations are connected through serial ports.
**Printer Setup**

The Printer Setup page is used to add, modify, or delete EDGE printer definitions. You can also set a default server printer.

You must define separate printer definitions to print in different formats even if you are printing to the same printer. For example, define separate printer definitions for printing in portrait mode normal, por-

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port name</td>
<td>If you are creating a port name, you must use the same name as the name defined in UNIX for the port.</td>
</tr>
<tr>
<td>Printer name</td>
<td>The server printer name.</td>
</tr>
<tr>
<td>Set by user at login</td>
<td>If checked, the user can set the terminal type at login.</td>
</tr>
<tr>
<td>Station name</td>
<td>The name used to identify the port for telephony.</td>
</tr>
<tr>
<td>Terminal type</td>
<td>The terminal type for the port.</td>
</tr>
</tbody>
</table>
trait mode condensed, landscape mode normal, and landscape mode condensed.

To set up an EDGE printer, select the following:

EDGE / Server area / Printers tab

When you add or modify an EDGE printer definition, a page similar to the following is displayed:

![Edge Printer Setup Window]

**Properties:**

- **Description:** A text description of the current printer. This is optional.
- **Length:** The number of rows.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Printer command</td>
<td>The UNIX <code>lp</code> commands to execute when you print. You can use this, for example, to set up one EDGE printer to print from the default paper stack. You can set up a second EDGE printer using the same server printer, but with a command to print from the manual feed.</td>
</tr>
<tr>
<td>Printer name</td>
<td>The printer name to be displayed in EDGE; can contain up to ten alphanumeric characters or periods, but must start with an alphanumeric character. This is not the UNIX printer name. This EDGE printer name is displayed in the printers list box on the Printers tab page.</td>
</tr>
<tr>
<td>Server printer name</td>
<td>The UNIX printer name. This is not the EDGE printer name. To view a list of your UNIX printer names, from UNIX enter the following:</td>
</tr>
<tr>
<td></td>
<td><code>lpstat -t</code></td>
</tr>
<tr>
<td>Width</td>
<td>The number of columns.</td>
</tr>
</tbody>
</table>

### Default Printer Setup

To set a default server printer, select the Set Server Default Printer button on the Printers tab. When you select Set Server Default Printer, you are prompted to specify a UNIX printer.
Note: To set a server printer to use during your current login session, use Set Server Printer from the File menu.

**Printer Order**

There are many areas in EDGE Developer where you can specify a printer. EDGE uses the printer specified in the following order:

1. The printer specified in the Reports area (that is, if you are printing from Ad Hoc Reports or Correspondence).
2. The printer specified through the Set Defaults command on the Ad Hoc Reports tab (that is, if you are printing an Ad hoc report).
3. The printer specified for the current login session. (For more information, see the section File Menu.)
4. The printer specified in your user ID setup or group ID setup. (For more information, see the section User Setup.)
5. The printer specified in the Port Setup.
6. The default server printer. (For more information, see the section Default Printer Setup.)
The Server Communication Setup page is used to specify properties to enable transfer of data between a non-EDGE window and an EDGE window using the AT&T 6500 cluster controller.

To transfer data, use the `COMMUNICATE-(RECEIVE)` verb.

To display the Server Communication Setup page, select the following:

- EDGE / Server area / Server Comm Setup tab

When you add or modify, a page similar to the following is displayed:

![Property dialogue box for Communication process name: AT&T6500, Device: AT&T6500, Function: RECEIVE, Operator message: blank]

**Properties:**
- **Communication process name**: The name of the process, which is used by the `COMMUNICATE-(RECEIVE)` verb.
## Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device</td>
<td>The device. Currently, the only device is ATT6500.</td>
</tr>
<tr>
<td>Function</td>
<td>The function this device is to perform. Currently, the only function is RECEIVE.</td>
</tr>
<tr>
<td>Operator message</td>
<td>The message that is displayed on the agent’s screen when data is ready for transfer. For example, the message can explain what the agent must do to signal the server system.</td>
</tr>
</tbody>
</table>
**SuperServer Setup**

When the EDGE server is running, socket connections can be used as an alternative to telnet as a method of communicating between the EDGE Server and EDGE Client. A daemon process called SuperServer running on the EDGE server is responsible for listening and accepting connections to the specific socket ports specified in EDGE Developer.

Once SuperServer accepts the connection, it launches a separate program called Logon, which validates the user logon process. The Logon program executes EDGE’s .profile or profile.ksh file which allows full support for environment variables set in the profile, .user.profile, and the edge_country files.

SuperServer is started in one of the following ways:

- If validation of the system login and password is required, SuperServer must be started from ACL. This is the only way to validate system login and password. See the section `supersrvr` Utility.

- If no system login and password validation is required, SuperServer can be started from EDGE Developer or ACL. In this case, the environment variable `LOGON_NO_VALIDATION` must be added to the .profile or profile.ksh file. SuperServer is
Setting Up SuperServer

To set up SuperServer, select the following:

EDGE / Server area / General tab / SuperServer Setup

A page similar to the following is displayed:

then started from the Start SuperServer command button in the SuperServer Setup page.
Figure 3-1. **SuperServer Setup**

**Properties:**

- **Use telnet (disable SuperServer)**: Allows EDGE Server and EDGE Client to communicate via telnet only.
- If SuperServer is still running and you close the SuperServer Setup page, a message is displayed asking if you want to shut down the SuperServer.
Use socket
(enable SuperServer)

Allows agents to log in to EDGE Client using socket connections. If checked, telnet connections are still available.

If EDGE administrators want agents to only log in to EDGE Client using telnet, make sure SuperServer is stopped; otherwise, agents can log in via sockets.

To check if SuperServer is still running, look at the Status property or go to ACL and use the supersrvr Utility.

Base port

The port number SuperServer starts listening at. This value must be a positive integer. The port numbers for the other ports are automatically determined based on this value.

The default is 21064.

ACL port

Reserved.

ETWW port

Reserved.

GEO port

The port number for EDGE Client (Base port + 3). Use this port number in the Connection Setup screen in EDGE Client or EDGE Developer.

This is display only.
### supersrvr Utility

The supersrvr utility manages the socket connections between client workstations and EDGE. This utility is started at ACL and validates

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log file</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Status</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Start SuperServer</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Stop SuperServer</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
system login and password. This is the only way to validate system
login and password.

The supersrvr utility syntax is as follows:

```
supersrvr -? | -start {base_port {logfile_name}} | -stop {base_port} |
  -status {base_port}
```

**Note:** The pipe symbols ( | ) in the syntax indicate mutually exclusive
options. The curly braces ({ }) in the syntax indicate optional
parameters.

- `?-` Displays the list of parameters available for the super-
srvr utility.

- `start` Starts the SuperServer. The following additional
parameters can be specified:

  - `base_port` The number of the first port Super-
  Server is to listen at. If not specified, SuperServer starts listening at
default port number 21064.

  - `logfile_name` The fully qualified pathname to a log
  file. If specified, the `base_port` must
  also be specified. If `logfile_name` is not specified, no logs are created.

- `stop` Stops the SuperServer. EDGE itself is not stopped. The
  following additional parameter can be specified:
SuperServer Log File

SuperServer log file is used for troubleshooting when EDGE Client cannot connect via sockets.

The following are the most common types of messages returned by SuperServer and EDGE Developer that are written to the log file:

- Invalid port number for the base port.
- SuperServer fails to start.
- SuperServer fails to stop.
- SuperServer is already shut down.
- SuperServer is on a different port.
- SuperServer fails a query.

**base_port**

The port number SuperServer is to stop listening at. If not specified, the default port number 21064 is used.

**-status**

Queries the SuperServer and displays the name of the EDGE account and log file, if any. The following additional parameter can be specified:

**base_port**

The port number SuperServer is running on. If not specified, the default port number 21064 is used.
Periodically, you should clear this file since it is not done automatically.
User Setup

This area is used to define the following:

- Bulletins.
- Groups.
- ODBC Users. For information on ODBC, see the EDGE 7.11 ODBC Reference Manual.
- Users Tab.

To display the Users page, select the following:

EDGE / Users area

A page similar to the following is displayed:
Groups

You can create a group to define EDGE access for a group of users. By setting access at the group level, you can control and change access without having to modify each user ID. Groups are optional.

*Note:* Users may not belong to multiple groups.

To create or modify a group, select the following:

EDGE / Users / Groups tab
When you create a new group or modify an existing one, a page similar to Figure 3-2 is displayed.

**Group Options**

The Group Options page is used to define telephony settings, project access, users in the group, and login access for a group.

When the Group Options tab is selected, a page similar to the following is displayed:
Figure 3-2.  Group Definition

Properties:

ACL  If checked, the user IDs associated with this group have access to ACL.

Allow administrator access  If checked, the users in the group can log on to EDGE using EDGE Administrator.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow developer access</td>
<td>If checked, the users in the group can log on to EDGE using EDGE Developer and the Group Access Definition tab is enabled to allow you to Specify EDGE Access Areas.</td>
</tr>
<tr>
<td>Callset/Group number</td>
<td>The callset number or the group number. This is used with telephony only.</td>
</tr>
<tr>
<td>Country</td>
<td>Allows the developer or system administrator to associate the user group with a country code.</td>
</tr>
<tr>
<td></td>
<td>The country code must be defined in the Country Code or International ACTZ tab before EDGE can associate the country code with a time zone (see Country Calling Setup).</td>
</tr>
<tr>
<td></td>
<td>This property is optional. However, if you identify a country code, you must also identify a phone code before saving the user group record.</td>
</tr>
<tr>
<td>Description</td>
<td>A description of the group.</td>
</tr>
<tr>
<td></td>
<td>This property is optional.</td>
</tr>
<tr>
<td>Disable login</td>
<td>If checked, specifies user IDs associated with this group cannot be used to log on to EDGE.</td>
</tr>
<tr>
<td>Group name</td>
<td>The user defined group name.</td>
</tr>
<tr>
<td>Phone</td>
<td>Allows the developer or system administrator to associate the user group with a phone code containing the telephone area code and an optional prefix.</td>
</tr>
</tbody>
</table>
Properties: (continued)

- Example: Suppose that the telephone number is 949-784-6000. Valid phone codes:
  - Area code only: 949
  - Area code plus prefix: 949784

  Do not enter a separator character between the area code and prefix. The phone code must be defined in the ACTZ tab before EDGE can associate the phone code with a time zone (see ACTZ File Update).

  This property is optional. However, if you identify a phone code, you must also identify a country code before saving the user group record.

- Projects
  The projects the users in this group can access. This applies to EDGE Developer, as well as operations. To allow access to all projects, select all.

- Status level
  Status levels are used in ASCII, EDGE Client, and ODBC, but not in EDGE Developer.

  In ASCII, status levels determine the areas a user can access. The following status levels in ASCII are used:

<table>
<thead>
<tr>
<th>Status Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>A status level 10 can run reports, but not create or delete any.</td>
</tr>
<tr>
<td>20</td>
<td>A status level of 20 can create and run reports, but cannot delete anything.</td>
</tr>
</tbody>
</table>
Properties: (continued)

30 A status level of 30 can access and perform all functions in the checked areas in the Access Definition page.

In EDGE Client, status levels are used to distinguish between supervisors and agents. The following status levels in EDGE Client are used:

10 or 20 A status level of 10 or 20 are considered the same; an agent is normally a status level 10 or 20.

30 A status level of 30 gives you debugging options; a supervisor is normally a status level 30. For more information about debugging options, see your EDGE 7.11 Client Reference Manual.

In ODBC, status levels are used to determine access. For more information, see your EDGE ODBC Reference Manual.

Telephony device The telephony device for this group. This is used only with telephony.

Users The users associated with this group.
**Specify EDGE Access Areas**

The Access Definition page is used to specify the areas of EDGE the group can access. To enable the Access Definition tab, Allow developer access must be checked on the Group Options page similar to Figure 3-2.

The Access Definition page is similar to the following:
Properties:

- **All**: To allow access to all the modules for EDGE Developer, select All. All the check boxes are checked.

- **Global level**: The areas accessed at the global level (also called the EDGE level).
  - The areas checked can be accessed by the group.

- **Global/project level**: The modules accessed at both the global and project levels.
Users Tab

The Users tab allows you to do the following:

- Release All Locked User IDs.
- Release Locked User ID.
- Invoke User Monitor.

These are available as commands on the tab page or from the Users menu. To display the Users page, select the following:

EDGE / Users / Users tab

A page similar to the following is displayed:

Properties:

- Group name: The group ID you are specifying access for.
- None: To prevent access to any areas in EDGE, select None. All the check boxes are cleared.
- Project level: The areas accessed at the project level; can only be accessed from within a project.

The areas checked can be accessed by the group.
**Properties:**

**Release All Locked User IDs**

The Release All Locked User IDs command is used to release one or more locked users.

User IDs in use are always locked. The lock on the user ID is normally released when the user logs off. However, sometimes the lock on the user ID is not released; for example, if there was a power failure, the logoff was improper, or there is a frozen terminal condition.
**Properties: (continued)**

A locked user ID can also cause an error message to be displayed when an assembly is attempted, because the system shows a user ID logged on and currently using a project.

If a user ID is in active use, it cannot be unlocked.

**Release Locked User ID**

The Release locked user ID command is used to release one locked user.

**User Monitor**

The User Monitor command is used to view and access screens being displayed by another user who is in an operations or development guide in either test or live mode. The monitored user must be using an ASCII terminal, not EDGE Client.

**List User IDs**

This command provides a listing of all defined user IDs, similar to the following:
Create or Modify User IDs

This command is currently available only from the Users menu.

All EDGE users must have a user ID. The level of access and areas of workstation access are determined through the user ID. You can also create a Groups to define the EDGE access for a group of users.

A user ID can contain any combination of letters, numbers, and periods, up to ten characters. When you first install EDGE, the following user IDs are available by default:

- GST1-GST9.
The GSTn user IDs do not have passwords. The IMA user ID’s has a password and is intended for use by EDGE Customer Care.

The following tabs are available:

- User ID.
- Access.
- Access Definition.
- 3270. This tab describes the settings for 3270 users. For more information, refer to the EDGE 7.11 3270 Reference Manual.

Telephony. This tab describes the settings for telephony users. For more information, refer to the EDGE 7.11 Telephony Reference Manual.

All changes to the user ID password are recorded by EDGE. The date & time of the password change and the user ID that made the change is recorded in the USER.ID.FILE system file. An adhoc report called List Password Modification can be run from the Users menu when the users tab is selected. The report is similar to the following:
**User ID**

When you create or modify a user ID, a page similar to the following is displayed:
Properties:

Allow multiple sessions

A user can log on to EDGE multiple times with one user ID. This is intended for testing purposes.

There is no imposed limit to the number of logins a single user ID may incur using the multiple login feature. (You can specify up to the size of the c-tree record size limitation).
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Country</strong></td>
<td>Allows the developer or system administrator to associate the user with a country code. The country code must be defined in the Country Code or International ACTZ tab before EDGE can associate the country code with a time zone (see <strong>Country Calling Setup</strong>). This property is optional. However, if you identify a country code, you must also identify a phone code before saving the user record.</td>
</tr>
<tr>
<td><strong>Disable login</strong></td>
<td>If checked, the user cannot log on to EDGE.</td>
</tr>
<tr>
<td><strong>First name</strong></td>
<td>The first name of the user assigned to this user ID. During operations, this name is stored in the system field $FIRSTNAME.</td>
</tr>
<tr>
<td><strong>Full name</strong></td>
<td>The full name (first and last) of the user assigned to this user ID. Separate the names with a space. During operations, this name is stored in the system field $OPNAME.</td>
</tr>
<tr>
<td><strong>Group</strong></td>
<td>The group the user is in.</td>
</tr>
<tr>
<td><strong>Keep group settings</strong></td>
<td>The access settings for the <strong>Groups</strong> the user is in are to be used for this user ID.</td>
</tr>
<tr>
<td><strong>Last name</strong></td>
<td>The last name of the user assigned to this user ID. During operations, this name is stored in the system field $LASTNAME.</td>
</tr>
</tbody>
</table>
Password

The password for this user ID. The password can contain up to ten characters. The password is never displayed in the box; the box is always blank, whether there is a password assigned or not.

Note: The password is ignored if LDAP Authentication is being used. LDAP Authentication can be configured site-wide using Set Up Site Information or at a user-specific level using the Access tab. A password must be set when using LDAP authentication. Null passwords are not valid when using this method. Password Change Recording will not work because passwords are changed in the LDAP directory, not in EDGE.

To change an existing password to no password, enter the word NULL at the password prompt. The existing password is deleted and the user no longer needs to enter a password at login time.

A password is optional.

If a password is changed after the user ID was first created, EDGE will record the date & time of the password modification. An adhoc report can be run from the users menu to list all password modifications. See Password Change Recording for additional information.
Properties: (continued)

Phone

Allows the developer or system administrator to associate the user with a phone code containing the telephone area code and an optional prefix.

Example: Suppose that the telephone number is 949-784-6000. Valid phone codes:

Area code only: 949
Area code plus prefix: 949784

Do not enter a separator character between the area code and prefix. The phone code must be defined in the ACTZ tab before EDGE can associate the phone code with a time zone (see ACTZ File Update).

This property is optional. However, if you identify a phone code, you must also identify a country code before saving the user record.

Printer

The name of the server printer for the user.

User name

The user ID the user enters at log on.

This is display only.

Access

The Access tab allows you to specify user access privileges, such as access to ACL. When the Access tab is selected, a page similar to the following is displayed:
Properties:

- **ACL**
  - If checked, the user has access to ACL. For information on ACL, see Chapter 10.

- **Allow Administrator access**
  - If checked, the user can log on to EDGE using EDGE Administrator.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow Developer access</td>
<td>If checked, the user can log on to EDGE using EDGE Developer and the Access Definition page is enabled to allow you to specify the areas of EDGE Developer the user can access.</td>
</tr>
<tr>
<td>LDAP Authentication</td>
<td>EDGE 7.11 supports LDAP Authentication against LDAP-compliant servers, including OpenLDAP Server and Active Directory. When LDAP Authentication is used, the user password is stored in the LDAP directory, and the user password in EDGE is ignored. Optionally, the LDAP session can be secured using certificate files.</td>
</tr>
<tr>
<td>LDAP URI</td>
<td>Specifies the LDAP server. Specifying this is optional if the LDAP URI is specified on a site-wide basis in the Set Up Site Information screen. If this value is set, it overrides the value in the Set Up Site Information Screen. Example value: ldap://ldap.mydomain.com</td>
</tr>
<tr>
<td>Bind DN</td>
<td>Identifies the information necessary to specify this user in the LDAP directory.</td>
</tr>
</tbody>
</table>

**Example:**

<table>
<thead>
<tr>
<th>LDAP URI</th>
<th>ldap://ldap.mydomain.com</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bind DN</td>
<td>Specify user information</td>
</tr>
</tbody>
</table>

**Note:**
- LDAP URI is optional if specified on a site-wide basis in the Set Up Site Information screen.
- The user password in EDGE is ignored when LDAP Authentication is used.
- Certificate files can be used for securing the LDAP session.
Properties: (continued)

Secure LDAP  If checked, this specifies that LDAP is secured over an encrypted connection (recommended, since LDAP is being used to verify passwords). You will need to retrieve the certificate file (usually provided by the LDAP-compliant server or a certification authority) and identify the certificate file to be used in the configuration file $EDGE/.ldaprc. The TLS_CACERT directive can be used to specify the certificate file. Sample $EDGE/.ldaprc is listed below:

Sample .ldaprc:

```#
# LDAP Defaults
#
# See ldap.conf(5) for details
# This file should be world readable but not world writable.
BASE DC=edge,DC=usa,DC=aitgroupad,DC=com
URI ldap://oracle11g.pbi.global.pvt
#URI ldap://amber.usa.aitgroupad.com ldap://ldap-master.example.com:666
#SIZELIMIT 12
#TIMELIMIT 15
#DEREF never
TLS_CACERT /edge/edge785/EDGE/w2003ad.edge.usa.aitgroupad.com.cer TLS_REQCERT allow```
Operator Access
The projects the user can make or receive calls for.
This property is display only. For information on assigning access, see the section Operator Access.

Projects
The projects the user can access. This applies to EDGE Developer, as well as operations. To allow access to all projects, select All.

Status level
Status levels are used in ASCII, EDGE Client, and ODBC, but not in EDGE Developer.
In ASCII, status levels determine the areas a user can access. The following status levels in ASCII are used:

10 A status level 10 can run reports, but not create or delete any.
20 A status level of 20 can create and run reports, but cannot delete anything.
30 A status level of 30 can access and perform all functions in the checked areas in the Access Definition page.

In EDGE Client, status levels are used to distinguish between supervisors and agents. The following status levels in EDGE Client are used:

10 or 20 A status level of 10 or 20 are considered the same; an agent is normally a status level 10 or 20.
Access Definition

The Access Definition page specifies the areas of EDGE Developer the user can access. To enable the Access Definition tab, check Allow Developer access on the Access Definition page. When the Access Definition tab is selected, a page similar to the following is displayed:

Properties: (continued)

30  A status level of 30 gives you debugging options; a supervisor is normally a status level 30. For more information about debugging options, see your EDGE 7.11 Client Reference Manual.

In ODBC, status levels are used to determine access. For more information, see the EDGE 7.11 ODBC Reference Manual.

User name  The user ID the current setup is for.

This is display only.
Properties:

- **All**
  
  To allow access to all the modules for EDGE Developer, select All. All of the boxes are checked.

- **Global level**
  
  The areas accessed at the global level (also called the EDGE level).

- **Global/project level**
  
  The modules accessed at both the global and project levels.
Properties: (continued)

Note: Selecting access to the Guide Development area enables the Assembly toolbar for the user in EDGE Developer; otherwise it is disabled.

None
To prevent access to any areas in EDGE, select None. All the check boxes are cleared.

Project level
The areas accessed at the project level; can only be accessed from within a project.
Encrypted Communications

Overview

EDGE Client supports encrypted communications with the EDGE Server via the Secure Shell (SSH) protocol. SSH operates in a similar way to telnet except that it provides checks for proving the authenticity of an EDGE Server, encryption and authentication of all client/server communications, and server logon processing using strong public key cryptography.

Note: Encrypted communication is optional and requires additional licensing. The module can be enabled anytime after the installation or upgrade of EDGE. If you are using EDGE Developer in unencrypted mode, or you use EDGE Administrator or EDGE Backup, the telnet daemon must remain installed on the EDGE host. Otherwise, the telnet daemon can be disabled or removed.
Encrypted Communications

Requirements

In order for EDGE Client to communicate securely with EDGE Server, an SSH daemon must be installed and configured on the EDGE host. EDGE Client has been certified with OpenSSH v3.9 or greater, however any SSH server that conforms to the SSH v2 protocol specification should work.

Before connecting via SSH, the following steps should to be performed:

- Decide on the encryption cipher to use
- Decide on the authentication method to use
- Obtain the host’s key fingerprint

If a private/public key pair is to be used for server logon authentication, the following additional steps must be performed:

- Generate a public/private key pair (using PuTTYgen) for use with EDGE
- Add the public key to the list of authorized keys on the EDGE host
- Configure the server to ensure correct authentication
Encrypted Communications

Configuration

Decide on the encryption cipher

EDGE Client supports the following encryption ciphers:

- DES
- Triple DES
- Blowfish
- AES

Data Encryption Standard (DES) encryption uses a 56 bit key to encrypt data communications. This method has been around for a long time and it is considered insecure by today’s standards.

Triple DES encryption uses the same technique as DES except that it performs the cipher algorithm three times, thus providing 168 bit encryption. This is a very popular encryption algorithm, however it is slower when compared to Blowfish and AES.

Blowfish is a fast, free encryption cipher using 128 bit key length.

Advanced Encryption Standard (AES) - also known as Rijndel - has been adopted by the U.S. Government as an encryption standard and operates with a 128 bit, 192 bit, or 256 bit key. The exact key length used is determined by SSH server configuration.
Encrypted Communications

EDGE Client supports the following authentication methods:

- Password authentication
- Public key authentication

Password authentication is very similar to traditional telnet authentication where a username/password is sent to the server, except that encryption is set up before the username and password are sent, thus preventing plain-text passwords being sent 'over-the-wire'.

Public key authentication is more secure than password authentication as the key is not sent directly over the network, making it extremely difficult to 'spy' on the authentication process and allow a third party to gain unauthorized access to the host system. EDGE Client supports the following key types:

- RSA
- DSA

Each time EDGE Client connects to the EDGE Server, it verifies that it is communicating with a trusted host. It does this by comparing a system-wide key sent by the host with one cached on the client. However, the first time EDGE Client connects with the host, no such key has been cached and the host must be verified manually. Rather than comparing the entire key (which can be very long), a fingerprint
may be generated from the host key which is suitable for human verification.

**Note:** The following commands apply to OpenSSH. Other SSH server implementations may require different commands and provide a different output from that shown below.

**Note:** It is recommended that this procedure be performed either on the system console, or via an SSH session where the host has already been verified for its authenticity.

To obtain the host’s key fingerprint, log in to a terminal session as root and run the following command:

```
# ssh-keygen -l -f keyfile
```

Where `keyfile` is the path to either the hosts private or public key (e.g. `/etc/ssh/ssh_host_rsa_key`)

The output from this command should be similar to the following:

```
ssh_host_rsa_key.pub
```
The key length (1024) and the fingerprint (the colon-separated string) is used later when comparing with the fingerprint displayed in EDGE Client to verify that it is connecting to the same host.

See Host Authentication for more information on the procedures used by EDGE Client to verify host authenticity.
Private/public key authentication

Public key technology provides a significant improvement over ‘shared secret’ methods such as the username/password challenge/response used in traditional authentication methods. Most modern public key cryptography is based on prime number theory and the fact that it takes a considerable amount of time for computers to factorize the result of two large prime numbers multiplied together.

Before public key authentication may be used, a private/public key pair must be generated on one computer. The private key is held safe on the EDGE Client, while the public key is stored on the EDGE host. While the private key must be kept safe and secure, it does not matter if the public key is made known to a large number of people.

Rather than sending a key directly over the network, it is used to encrypt a piece of data. This piece of data may only be decrypted correctly with the matching public key and if so decrypted, it can be assumed that the data came only from someone holding the matching private key. This means that even if an unauthorized party were to ‘wire-tap’ and crack the communication link, they would still not be able to steal the key for malicious use later.
Generate a public/private key pair (using PuTTYgen)

PuTTYgen is a Windows tool used to create keys that are supported by EDGE Client. PuTTYgen is a freely available tool and may be downloaded from http://www.chiark.greenend.org.uk/~sgtatham/putty/.

After running PuTTYgen, select the key type and length to be generated. Any key length may be used, however a 1024 bit key is considered strong enough to be secure with today’s technology.

Click the ‘Generate’ button and move the mouse around until the bar is filled entirely. After enough randomness has been collected, it will take a few moments to generate the key pair.

Save the private key for use within EDGE Client by clicking the ‘Save private key’ button. By default, keys will be saved with a .PPK extension. The location of the key chosen here should be entered in the Connection Setup dialog.

Note: It is possible to save the private key encrypted with a passphrase to provide additional security. If a passphrase is used, this must be entered as the ‘Server password’ configuration item in the Connection Setup dialog.

Do not close PuTTYgen yet - the public key must be copied and used later while configuring the list of authorized keys on the EDGE host.
Add the public key to the list of authorized keys

Note: If either the private or public key is lost, the other side of the key is irretrievable and a new private/public key must be generated from scratch.

In order to verify the private key, the public key must be added to the list of authorized keys located on the EDGE host.

Note: The following commands apply to OpenSSH. Other SSH server implementations may require different commands and provide a different output from that shown below.

In order to add the key to the authorized keys list, open a terminal on the EDGE host (preferably via an SSH session where the host can be verified for its authenticity), and login as the ‘ima’ user.

Ensure that the .ssh directory exists by running the following command:

$ ls -d $EDGE/.ssh

If the directory does not exist, create it by using the following commands:

$ mkdir $EDGE/.ssh
$ chmod 700 $EDGE/.ssh

Edit the authorized_keys file by entering the following:

$ vi $EDGE/.ssh/authorized_keys

Enter insert mode (by typing ‘i’) and cut and paste the public key obtained above into the authorized_keys file, ensuring that the key exists on a single line by itself.

\textit{Note:} If your terminal emulator does not provide support for cut and paste, the public key should be saved to a local file and transferred to the EDGE host via FTP. From there, it may be added to the authorized_keys file by using the ‘:r’ command in vi.

It is possible to use the same private/public key pair by all agents running EDGE Client. However, if separate key pairs are to be used, each public key must be added to the authorized_keys file, with each key on a separate line.

Exit insert mode by entering <esc>, and save the file by entering ‘:wq<enter>’. Ensure that the authorized_keys file has the correct permissions by entering the following:

$ chmod 644 $EDGE/.ssh/authorized_keys
**Configure the server**

By default, OpenSSH requires certain permissions to be set on the EDGE home directory before it will validate against it’s list of authorized keys.

It may be necessary to remove the group-write permission on $EDGE in order for the authentication process to work correctly. To do this, open a terminal session, login as the user ‘ima’ and enter the following command:

$ chmod 755 $EDGE
Host Authentication

Before EDGE Client will communicate with the EDGE Server, the host running EDGE Server must be authenticated. This is done each time EDGE Client connects and so to avoid a message being displayed each time a user logs in, host keys are cached in the edgeo.ini file.

When a user logs in, the key sent from the host matched against any keys cached in edgeo.ini. If no matching entry can be found (based on the key type, hostname, and port number), a dialog similar to the following is displayed:
Both the server host name and the key fingerprint should be checked to ensure that this is indeed the authentic host running EDGE Server (see Obtain the host’s key fingerprint for information on how the host fingerprint may be displayed).

If the both the server and the fingerprint are deemed to be accurate, the user should click the 'Yes' button to continue. Clicking 'Yes' will store the key in edgeo.ini (under the [ssh-keys] section).
Note: If both primary and secondary servers are configured in EDGE Client, each host must be authenticated separately.

The key cached in edgeo.ini is not specific to the individual Windows PC that the verification is performed on. It may be more convenient to validate the host key on a single machine before distributing the edgeo.ini file to a large number of users.

Invalid Keys

Each subsequent time the user logs in to EDGE, the host key is validated against the cached version. If the keys do not match, a message similar to the following is displayed:

Unless the host's key is known to have changed, this message indicates that EDGE Client is connecting to an un-trusted host (maybe because of a ‘man-in-the-middle’ attack) and that the integrity of the system may have been compromised - security personnel should investigate the cause of the problem.
If it is known that the host's key has been changed by someone authorized to do so, the cached key must be manually removed from edgeo.ini before EDGE Client will connect via this host/port combination (See [ssh-keys] in the edgeo.ini section of this manual) host key format for details on identifying the correct line in edgeo.ini). After the offending key has been removed, the next time a user logs on they will be presented with a new 'Verify Host Key' dialog as described above).
This chapter describes the following monitors:

**Global Monitor**
Provides call activity information for all agents, groups, and projects. EDGE automatically collects data on agents when they log in to EDGE Client. This monitor is used to report general call statistics for all projects for all agents, groups, or a specific group. Global Monitor provides call activity information for all agents, groups, and projects. EDGE automatically collects data on agents when they log in to EDGE Client. (The other monitors track similar data, but present it in other formats.)

**Logon Monitor**
Provides logon and logoff information for agents and groups. In order for this data to be tracked, Logon Monitor must be enabled. This monitor is used to report total time for all agent's EDGE Client login sessions.
**Project Monitor**

Provides call activity information for all agents and groups within a specific project. EDGE automatically collects data on agents when they log in to EDGE Client.

This monitor is used to report on data similar to that tracked by the [Global Monitor](#), but for a specific project. Project Monitor has a built-in interface in EDGE Client called Operator Monitor that can be enabled for agents to see the activity data. Project Monitor can be customized to report on specific results or revenue information.

**User Monitor**

Allows a user to view and access the screens being displayed by another agent.

Unlike the other monitors, this does not produce a report, but instead allows real time online monitoring of what an agent is doing and seeing. Depending on settings, the person monitoring can enter data on or otherwise manipulate the agent’s screen.

*Note: User Monitor is available only for ASCII terminals. It is not available for EDGE Client.*
Global Monitor

The Global Monitor feature collects data on all agents for all projects in the system. You can produce monitor reports and you can initialize the monitor to delete older records. The Global Monitor data is collected automatically by EDGE.

A Global Monitor report can be produced with the following statistics (sorted by hour or group):

- The cumulative time worked (includes idle and guide time).
- The total number of calls.
- The number of calls per hour.
- The number of inbound calls.
- The number of outbound calls.
- The number of calls which were new call IDs.
- The number of calls that were not new call IDs.

To access the Global Monitor, select the following:

EDGE / Reports / Monitor

The following commands are available from both the tab sheet and the Monitor menu:
Initialize the Global Monitor

The Initialize Global Monitor command clears all the data that has accumulated since the last initialization. Global Monitor data is kept in a file that is continually updated as project activity occurs. This file continues to grow until you initialize it; therefore, you should make initialization part of your periodic maintenance.

You can also initialize the Global Monitor using the ACL INITIALMON command, which is described in Chapter 10.

Caution! Initializing the monitor while agents are in operations can corrupt the file and hang the agents’ workstations.

Global Monitor Reports

The Run Global Monitor command generates a report containing the monitor data. When you select Run Global Monitor, a dialog box similar to the following is displayed:

- Initialize the Global Monitor.
- Global Monitor Reports.
Properties:

Format   The type of report; can be one of the following:

   By group   Arranges the report by groups. See Figure 5-1. (Groups are described in the section Groups.)

   By group by hour   Arranges the report by groups, with a subtotal for each hour. See Figure 5-2.

   By hour   Arranges the report by hour. See Figure 5-3.

   By hour, by group   Arranges the report by hour, with a subtotal for each group. See Figure 5-4.

Limit to group   Limits the report to the specified group.

Send to   The destination of the report; can be one of the following:
The information shown on the Global Monitor report depends on the sort option you choose. The report headings that display for each sort option are described in Table 5-1.

**Note:** If a user ID is not assigned to a group, the user ID is shown as the group ID on Global and Project Monitor reports.

**Table 5-1. Global Monitor Headings**

<table>
<thead>
<tr>
<th>Heading</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calls</td>
<td>The total number of attempted dialings made since the last initialization.</td>
</tr>
<tr>
<td>Calls/Hour</td>
<td>The average number of calls per hour. The system computes the number by dividing the number of calls by hours worked.</td>
</tr>
<tr>
<td>First Calls</td>
<td>The number of call IDs that were accessed for the first time.</td>
</tr>
<tr>
<td>Group</td>
<td>The name of the group of users. User groups are described in Chapter 3.</td>
</tr>
</tbody>
</table>
Table 5-1.  Global Monitor Headings (continued)

<table>
<thead>
<tr>
<th>Heading</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hour</td>
<td>The hour in which the call activity occurred. This column is displayed if you specified a by hour format. The figures in the other columns apply to the specific hour displayed. Hours are represented from the beginning of the hour to the 59th minute of that hour, 12:00-12:59.</td>
</tr>
<tr>
<td>Hours Wkd</td>
<td>The duration of time in the guide or total time since last initialization. For reports by hour, you can get totals higher than one hour since all calls started in one period are counted in the total for that period. For example, a call starting at 7:55 and ending at 8:10 adds 15 minutes to the total for the hourly period 7:00-7:59.</td>
</tr>
<tr>
<td>Inbound</td>
<td>The number of inbound calls received.</td>
</tr>
<tr>
<td>Operator</td>
<td>The agent’s name. This column appears on reports using a format that includes the group.</td>
</tr>
<tr>
<td>Outbound</td>
<td>The number of outbound calls made.</td>
</tr>
<tr>
<td>Recalls</td>
<td>The number of recalled numbers. Any call that is not a First Call is counted as a Recall.</td>
</tr>
<tr>
<td>User ID</td>
<td>The ID of the agent assigned to the project. Only the first four digits of the User ID are displayed.</td>
</tr>
</tbody>
</table>
## Sample Global Monitor Reports

### Figure 5-1. Global Monitor by Group

<table>
<thead>
<tr>
<th>GROUP</th>
<th>USER</th>
<th>OPERATOR</th>
<th>HOURS</th>
<th>CALLS</th>
<th>IN-HOUR CALLS</th>
<th>OUT-HOUR CALLS</th>
<th>FIRST CALLS</th>
<th>RE-CALLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>GST6</td>
<td>GUEST</td>
<td></td>
<td>0.03</td>
<td>2</td>
<td>52.6</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>GST7</td>
<td>GUEST</td>
<td></td>
<td>0.02</td>
<td>3</td>
<td>106.9</td>
<td>3</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Guest</td>
<td></td>
<td></td>
<td>0.06</td>
<td>5</td>
<td>75.6</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>TOT:</td>
<td></td>
<td></td>
<td>0.06</td>
<td>5</td>
<td>75.6</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

(cr=again, X=exit)
### Figure 5-2. Global Monitor by Group, by Hour

<table>
<thead>
<tr>
<th>GROUP</th>
<th>USER</th>
<th>OPERATOR</th>
<th>HOUR</th>
<th>HOURS</th>
<th>CALLS/HOUR</th>
<th>IN-BOUND</th>
<th>OUT-BOUND</th>
<th>FIRST CALLS</th>
<th>RE-CALLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guest</td>
<td>GST6</td>
<td>GUEST</td>
<td>10 AM</td>
<td>0.03</td>
<td>2</td>
<td>52.6</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Guest</td>
<td>GST7</td>
<td>GUEST</td>
<td>10 AM</td>
<td>0.02</td>
<td>3</td>
<td>106.9</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>***</td>
<td></td>
<td></td>
<td></td>
<td>0.06</td>
<td>5</td>
<td>75.6</td>
<td>5</td>
<td>0</td>
<td>5</td>
</tr>
</tbody>
</table>

TOT: 0.06 5 75.6 5 0 5 0

(cr=again, X=exit):
Figure 5-3.  Global Monitor by Hour
<table>
<thead>
<tr>
<th>HOUR</th>
<th>GROUP</th>
<th>USER</th>
<th>OPERATOR</th>
<th>HOURS WKD</th>
<th>CALLS /HOUR</th>
<th>IN-BOUND</th>
<th>OUT-BOUND</th>
<th>FIRST CALLS</th>
<th>RE-</th>
<th>CALLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 AM</td>
<td>Guest</td>
<td>GST6</td>
<td>GUEST</td>
<td>0.03</td>
<td>2</td>
<td>52.6</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 AM</td>
<td>Guest</td>
<td>GST7</td>
<td>GUEST</td>
<td>0.02</td>
<td>3</td>
<td>106.9</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>***</td>
<td>***</td>
<td>***</td>
<td>***</td>
<td>0.06</td>
<td>5</td>
<td>75.6</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>TOT:</td>
<td></td>
<td></td>
<td></td>
<td>0.06</td>
<td>5</td>
<td>75.6</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

(cr=again, X=exit):

**Figure 5-4. Global Monitor by Hour, by Group**
**Logon Monitor**

The Logon Monitor feature tracks the logon time, logoff time, number of minutes worked, and the number of calls for all EDGE Client agents. You can produce monitor reports and you can initialize the monitor to delete older records.

*Note: Logon Monitor does not provide information on EDGE ASCII users or EDGE Developer developers.*

To access the Logon Monitor, select the following:

- **EDGE / Reports / Monitor tab**

The following commands are available from both the Monitor tab and the Monitor menu:

- **Setup Logon Monitor.**
- **Initialize Logon Monitor.**
- **Run Logon Monitor.**

**Setup Logon Monitor**

The Setup Logon Monitor command enables and disables the logon monitor. The Logon Monitor should be set up when no agents are logged into EDGE Client. Only agents who log in after the Logon Monitor is enabled are recorded.
When you select the Setup Logon Monitor command, a dialog box similar to the following is displayed:

![Setup Logon Monitor](image)

**Properties:**

<table>
<thead>
<tr>
<th>Disable</th>
<th>Enables the Logon Monitor. When selected, the agent logon and logoff time is not tracked. This is the default.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>Enables the Logon Monitor. When selected, a record is created in the EDGE LOGON.HIST file for each EDGE Client agent. The agent’s login time, logout time, number of minutes worked, and number of calls taken are tracked.</td>
</tr>
</tbody>
</table>

**Note:**  For Logon Monitor, a call is considered complete each time an agent routes to $END. If the guide does not route to $END, the call is not considered complete and the total number of calls will not be correct.
The Initialize Logon Monitor command allows you to determine how many days of monitor records to keep. For example, if you keep the last 7 days of records, all records older than 7 days are deleted from the LOGON.HIST file. The data deleted is determined by the `Keep records from last nnn days` property.

The Initialize Logon Monitor should be run when there are no agents using EDGE Client. If agents are logged in, their records are deleted immediately and no record is updated when they log out. You can also initialize the Logon Monitor using the ACL INITIALMON command, which is described in Chapter 10.

**Note:** The Initialize Logon Monitor command is disabled until the Logon Monitor is enabled using Setup Logon Monitor; if Logon Monitor is later disabled, the data can still be initialized.

When you select the Initialize Logon Monitor command, a dialog box similar to the following is displayed:
Monitors Logon Monitor

The dialog box indicates which records were last initialized. For example, "Logon monitor records for sessions prior to and including Friday, August 20, 1999 were removed by GST2 on Friday, September 10, 1999 at 12:20 PM. Select the minimum age of records to delete and press Ok.

Keep records from last nnn days

Properties:

The number of days of records to keep; must be a number from 0 to 999. All records prior to the specified number of days are deleted. If set to zero, all data is removed. The default is 7 days.

The dialog box indicates which records were last initialized. For example, "Logon monitor records for sessions prior to and including Monday, June 14, 1999 were removed ...". The Initialize Logon Monitor dialog box also identifies who last initialized the monitor and when it occurred. For example, "... removed by GST2 on September 10, 1999 at 12:20PM."

When OK is selected to initialize the monitor, the following dialog box is displayed:
When OK is selected from this confirmation dialog, the monitor data is deleted; if Cancel is selected, the data is not deleted.

Monitor data is kept in the LOGON.HIST file that continues to accumulate until the next initialization; therefore, you should make initialization part of your periodic maintenance.

You can also initialize the Logon Monitor using the ACL INITIALMON command, which is described in Chapter 10.

The Run Logon Monitor command generates a report that provides data on the logon and logoff time for all EDGE Client agents, since the last time the Logon Monitor was initialized.

You can also generate a Logon Monitor report using the ACL PRINT-MON command, which is described in Chapter 10.

Although the number of calls for each logon session is tracked in the LOGON.HIST file, it is not included in the reports. This data can be
reported on using an EDGE Ad Hoc report or other reporting method such as Crystal Reports.

**Note:** The Run Logon Monitor command is disabled until logon monitor is enabled using Setup Logon Monitor; if Logon Monitor is later disabled, Run Logon Monitor remains enabled so that previous data can be reported on.

When you select the Run Logon Monitor command, a dialog box similar to the following is displayed:

![Logon Monitor dialog box]

**Properties:**

Select  The criteria for the report; includes the following:
**Properties: (continued)**

- **User**  
  Specifies the user name for the report. Select a specific user name or (All) for all users. The default is All.

- **Group**  
  Specifies the group name for the report. Select a specific group name or (All) for all groups. The default is All.

- **Start date**  
  The beginning date of the report. The default is the last date the Logon Monitor was initialized.

- **End date**  
  The last date of the report. The default is today’s date.

- **Sort**  
  The method of sorting the report; can be one of the following:
  - **Date, Group, User**  
    Arranges the report by date, then by group and user. This is the default. See Figure 5-5.
  - **Group, Date, User**  
    Arranges the report by group, then by date and user. See Figure 5-6.
  - **User, Date**  
    Arranges the report by user then by date. See Figure 5-7.

- **Send to**  
  The destination of the report; can be one of the following:
  - **Screen**
  - **Server printer**
The information shown on the Logon Monitor report depends on the sort options. The report headings for each format option are described in Table 5-2.

**Table 5-2. Logon Monitor Headings**

<table>
<thead>
<tr>
<th>Report Heading</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>The name for a group of users. User groups are described in Chapter 3.</td>
</tr>
<tr>
<td>Logon Date</td>
<td>The date the agent logged on.</td>
</tr>
<tr>
<td>Logon Time</td>
<td>The time the agent logged on.</td>
</tr>
<tr>
<td>Logout Complete</td>
<td>Specifies whether the agent logged out successfully; &quot;Y&quot; if the agent logged out successfully or &quot;N&quot; if the agent improperly exited from EDGE Client.</td>
</tr>
<tr>
<td>Logout Date</td>
<td>The date the agent logged off.</td>
</tr>
<tr>
<td>Logout Time</td>
<td>The time the agent logged off.</td>
</tr>
<tr>
<td>Minutes Worked</td>
<td>The number of minutes the agent worked during the login session.</td>
</tr>
<tr>
<td>User ID</td>
<td>The user ID of the agent.</td>
</tr>
</tbody>
</table>
## Sample Logon Monitor Reports

![Logon Monitor](image)

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Group</th>
<th>Userid</th>
<th>Logout Date</th>
<th>Logout Time</th>
<th>Logout Complete</th>
<th>Minutes Worked</th>
</tr>
</thead>
<tbody>
<tr>
<td>06-21-99</td>
<td>16:37:59</td>
<td>GST2</td>
<td></td>
<td>06-21-99</td>
<td>16:38:16</td>
<td>Y</td>
<td>0.2</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:40:28</td>
<td>GST2</td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:38:20</td>
<td>GST2</td>
<td></td>
<td>06-21-99</td>
<td>16:38:28</td>
<td>Y</td>
<td>0.1</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:43:33</td>
<td>GST2</td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:46:51</td>
<td>GST4</td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:41:42</td>
<td>GST5</td>
<td></td>
<td>06-21-99</td>
<td>16:41:51</td>
<td>Y</td>
<td>0.1</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:41:57</td>
<td>IMA</td>
<td></td>
<td></td>
<td>N</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:45:54</td>
<td>FORLANG</td>
<td>GST2</td>
<td></td>
<td>N</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>06-21-99</td>
<td>16:49:50</td>
<td>GENERAL</td>
<td>GST5</td>
<td></td>
<td>N</td>
<td></td>
<td>0.0</td>
</tr>
</tbody>
</table>

***

Press RETURN to continue:

---

**Figure 5-5. Logon Monitor by Date, Group, User**
### Figure 5-6.  Logon Monitor by Group, Date, User

<table>
<thead>
<tr>
<th>Group</th>
<th>Date</th>
<th>Time</th>
<th>Userid</th>
<th>Logon Date</th>
<th>Logon Time</th>
<th>Logoff Date</th>
<th>Logoff Time</th>
<th>Complete</th>
<th>Worked</th>
</tr>
</thead>
<tbody>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:37:59</td>
<td>GST2</td>
<td>06-21-99</td>
<td>16:38:16</td>
<td>Y</td>
<td></td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:40:28</td>
<td>GST2</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:38:20</td>
<td>GST2</td>
<td>06-21-99</td>
<td>16:38:28</td>
<td>Y</td>
<td></td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:43:33</td>
<td>GST2</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>GST4</td>
<td>06-21-99</td>
<td>16:46:51</td>
<td>GST4</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>GST5</td>
<td>06-21-99</td>
<td>16:41:42</td>
<td>GST5</td>
<td>06-21-99</td>
<td>16:41:51</td>
<td>Y</td>
<td></td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>GST5</td>
<td>06-21-99</td>
<td>16:41:57</td>
<td>GST5</td>
<td>N</td>
<td></td>
<td></td>
<td></td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

**FORLANG** 06-21-99 16:45:54 GST2 N 0.0

**GENERAL** 06-21-99 16:49:50 GST5 N 0.0

Press any key to continue...
<table>
<thead>
<tr>
<th>Userid</th>
<th>Login Date</th>
<th>Login Time</th>
<th>Logout Date</th>
<th>Logout Time</th>
<th>Minutes Worked</th>
</tr>
</thead>
<tbody>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:37:59</td>
<td>06-21-99</td>
<td>16:38:16</td>
<td>Y</td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:38:20</td>
<td>06-21-99</td>
<td>16:38:28</td>
<td>Y</td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:40:28</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:43:33</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>GST2</td>
<td>06-21-99</td>
<td>16:45:54</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>FORLAN</td>
<td>06-21-99</td>
<td>16:45:54</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>GST4</td>
<td>06-21-99</td>
<td>16:46:51</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>GST5</td>
<td>06-21-99</td>
<td>16:41:42</td>
<td>06-21-99</td>
<td>16:41:51</td>
<td>Y</td>
</tr>
<tr>
<td>GST5</td>
<td>06-21-99</td>
<td>16:49:50</td>
<td></td>
<td></td>
<td>N</td>
</tr>
<tr>
<td>IMA</td>
<td>06-21-99</td>
<td>16:41:57</td>
<td></td>
<td></td>
<td>N</td>
</tr>
</tbody>
</table>

Press RETURN to continue.

Figure 5-7. Logon Monitor by User, Date
The Project Monitor feature collects data on all agents in a specified project. You can produce monitor reports and you can initialize the monitor to delete older records.

The Project Monitor is a tool to help evaluate the status of a project and measure the activity of agents. Project Monitor was designed to be initialized on a daily basis to let you generate reports showing daily project utilization.

A Project Monitor report can be produced with statistics (sorted by hour or group) in the following areas:

- Main, which consists of general call activity data. This includes all of the same statistics reported in the Global Monitor Reports, but only for the current project. This data is automatically tracked.
- Units/Revenue, which provides sales-type data that is tracked using the SR.TRANS verb. This data is only reported if set up as described in the section Displaying Units and Revenue.
- Result groups, which shows agent activity sorted by defined EDGE result groups. This data is only reported if set up as described in the section Run Project Monitor.
Project Monitor data can optionally be made available to agents in EDGE Client using the Define Operator Monitor command. Operator Monitor allows agents to view their performance and the average activity for all agents. The data is displayed in operations when the agent selects the View Monitor command on the Call Center screen.

To access the Project Monitor, select the following:

```
project / Reports / Monitor
```

The following commands are available from both the tab sheet and the Monitor menu:

- Clear Project to Date Monitor.
- Run Project Monitor.
- Define Operator Monitor.
- Initialize the Project Monitor.
- Run Project Monitor.

A sample Project Monitor report is shown in Figure 5-8. The available report formats are described in the section Run Project Monitor. The report headings are described in Table 5-3. The report commands shown at the bottom of the screen are described in Table 5-4.
Clear Project to Date Monitor

This command clears all of the accumulated monitor data from the project. It requests confirmation before it executes the clear.

When you select the Clear Project to Date Monitor command, a dialog box similar to the following is displayed:

*Figure 5-8. Sample Project Monitor Report*

This command clears all of the accumulated monitor data from the project. It requests confirmation before it executes the clear.

When you select the Clear Project to Date Monitor command, a dialog box similar to the following is displayed:
**Result Groups**

The Result Groups command allows you to accumulate call event results into groups of results to include in Project Monitor reports. (A result identifies the outcome or disposition of a call event. For information on defining results, see Chapter 24.)

A result group can have as few as one result or as many as 30 results.

Result groups should be defined before the project goes live. Results that are defined after the project goes live or are not assigned to a result group are listed under the category Other in result group reports.

If you modify an existing result group or change its order on the Result Groups screen, you must perform a partial assembly and swap the guide to the agents.

Once results have been calculated for a result group, changing the order of the result groups on the Result Groups screen causes erroneous data.
The command displays a screen similar to the following:

![Define Operator Monitor](image)

**Properties:**
- **Description**: Describes the result group. This property is optional.
- **Results**: The results assigned to this group. Up to 30 results can be specified.
- **Title**: The result group name; can be up to five characters.

**Define Operator Monitor**

Define Operator Monitor is used to specify the result groups, unit totals, revenue totals, and a two-line message to be displayed to agents when they select View Monitor, and to be displayed on the Project Monitor report.
While the agent is viewing the monitor, the system tracks the time as *Idle Time*. This time is shown on the Project Monitor report.

The Define Operator Monitor dialog box is similar to the following:

![Define Operator Monitor](image)

**Properties:**

- **A maximum of ten Result Groups totals may be displayed**: The result groups to include in the Operator Monitor display. For information on result groups, see the section Run Project Monitor.

- **A two-line message may be displayed**: A message of up to two lines to be included in the Operator Monitor display.

- **Display unit totals**: If selected, the unit totals appear in the Operator Monitor display. For information on how to create the unit data, see the section Displaying Units and Revenue.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display revenue totals</td>
<td>If selected, the revenue totals appear in the Operator Monitor display. For information on how to create the unit data, see the section Project Monitor Report Screen Commands.</td>
</tr>
</tbody>
</table>

**Displaying Units and Revenue**

The units and revenue information shown on the Project Monitor report is generated only if the following steps have been completed:

- The Gather sales analysis data flag must be set in Project Options. For information on Project Options, see Chapter 2.

- The SR.TRANS verb must be used in end of guide logic to update the quantity and amount data in the REPORTING file. For information on the SR.TRANS verb, see Chapter 16.

**Initialize the Project Monitor**

The Initialize Project Monitor command moves data from the current reporting file to the project-to-date file, then clears the current reporting file. Initializing does not clear data from the project-to-date file.

You can also initialize the Project Monitor using the ACL INITIALMON command, which is described in Chapter 10.
Initializing the Project Monitor should be part of a daily or weekly administrative task. If you do not initialize the Project Monitor on a regular basis, you may find it difficult to interpret the data. The Project Monitor should also be initialized before you make a project operational.

When you select Initialize, the date of the last initialization and user ID of the person who performed the last initialization are displayed in a dialog box:

![ETW for Windows dialog box]

Caution! Initializing the monitor while agents are in operations can corrupt the file and hang the agents’ workstations.

**List Result Groups**

The List Result Groups report lists the currently defined result groups. This command is currently available only from the menu. The report is similar to the following:
When you select the Run Project Monitor command, a dialog box similar to the following is displayed:
Properties:

Format  Specifies how the report is to be formatted; can be one of the following:

- **By group**: Arranges report by groups. (Groups are described in Chapter 3.)
- **By hour**: Arranges report by hour.
- **By group, by hour**: Arranges report by group, with a subtotal for each hour.
- **By hour, by group**: Arranges report by hour, with a subtotal for each group.

Limit to group  Limits the report to one group that you select.

Period  The period for the report; can be one of the following:
Daily

Shows activity for the current period. The current period includes data by agent and by project since the last time the Project Monitor was initialized. Current period data is not added to the project-to-date file until you initialize the project.

If you initialize the Project Monitor on a daily basis, the current period consists of daily data. If you initialize on a weekly basis, the current period represents weekly data.

Project-to-date

Shows cumulative activity since the project was first initialized. The project-to-date file is empty until you first initialize the project. It does not include data from the active current period.

Each time you initialize the Project Monitor, information is removed from the current reporting file and added to the project-to-date file. Initializing does not clear data from the project-to-date file.

Send to

The destination of the report; can be one of the following:

- Screen.
- Server printer.
The information shown on a Project Monitor report depends on the options you choose. The report options are described in the section Run Project Monitor. When the report is displayed on the screen, you can select different views by using the report screen commands listed at the bottom of the report.

The report headings possible for Project Monitor reports are described in Table 5-3. The report screen commands are listed in Table 5-4.
Note: If a user ID is not assigned to a group, the user ID is used as the group ID on Global and Project Monitor reports.

Table 5-3. Project Monitor Headings

<table>
<thead>
<tr>
<th>Report Heading</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td># Result Group</td>
<td>The number of calls that were assigned this result defined in Result Groups. This column is displayed if the result groups option is selected on the report display screen.</td>
</tr>
<tr>
<td>Avg Guide</td>
<td>The average time in minutes that an agent spent in the guide. This is measured as the time the agent spent before being routed to the Call Center screen. This column is displayed if the units/revenue option is selected on the report display screen.</td>
</tr>
<tr>
<td>Avg Idle</td>
<td>The average time the agent spent in the operations area of the system but not in the guide. If you are tracking this time, be sure to have agents log off the system during meal and break periods. This column is displayed if the units/revenue option is selected on the report display screen.</td>
</tr>
<tr>
<td>Calls</td>
<td>The total number of attempted dialings made by the agent since the last initialization. This column is displayed on the main Project Monitor report and on the Result Groups report.</td>
</tr>
</tbody>
</table>
### Table 5-3. Project Monitor Headings (continued)

<table>
<thead>
<tr>
<th>Report Heading</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calls, Hour</td>
<td>The average number of calls per hour. The system computes the number by dividing the number of calls by hours worked.</td>
</tr>
<tr>
<td>First Calls</td>
<td>The number of call IDs that were accessed for the first time.</td>
</tr>
<tr>
<td>Group</td>
<td>Project Monitor report: The name of the group of users. Result Groups report: The name of the result group.</td>
</tr>
<tr>
<td>Hour</td>
<td>The hour in which the call activity occurred. This column is displayed if you specified a by hour format. The figures in the other columns apply to the specific hour displayed. Hours represent the beginning of the hour to the 59th minute of that hour; for example, 10:00 am covers the time 10:00-10:59.</td>
</tr>
<tr>
<td>Hours Wkd</td>
<td>The time the user spent in the guide since the last initialization. For reports by hour, you can get totals higher than one hour since all calls started in one period are counted in the total for that period. For example, a call starting at 7:55 and ending at 8:10 adds 15 minutes to the total for the hourly period 7:00-7:59. This column is displayed on the main Project Monitor report and on the Result Groups report.</td>
</tr>
<tr>
<td>Inbound</td>
<td>The number of inbound calls received by the agent.</td>
</tr>
</tbody>
</table>
Table 5-3.  Project Monitor Headings (continued)

<table>
<thead>
<tr>
<th>Report Heading</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. Units</td>
<td>The number of units sold. This column is displayed if the units/revenue option is selected on the report display screen.</td>
</tr>
<tr>
<td>Operator</td>
<td>The name of the agent. This column is displayed on any report using a format that includes the group.</td>
</tr>
<tr>
<td>Other</td>
<td>Values for any result not defined as part of a result group.</td>
</tr>
<tr>
<td>Outbound</td>
<td>The number of outbound calls made by the agent.</td>
</tr>
<tr>
<td>Recalls</td>
<td>The number of recalled numbers. Any call that is not a First Call is counted as a Recall.</td>
</tr>
<tr>
<td>Result Group, Hour</td>
<td>The number of results per hour assigned to the result group. This column is displayed if the result groups option is selected on the report display screen.</td>
</tr>
<tr>
<td>Revenue</td>
<td>The value of the units sold. This column is displayed if the units/revenue option is selected on the report display screen.</td>
</tr>
<tr>
<td>Revenue/ Hour</td>
<td>The amount of revenue generated per hour. This column is displayed if the units/revenue option is selected on the report display screen.</td>
</tr>
<tr>
<td>Total</td>
<td>The total of every column on the report.</td>
</tr>
</tbody>
</table>
Project Monitor Report Screen Commands

In every Project Monitor report displayed on the screen, the commands listed Table 5-4 are displayed at the bottom of the screen, as shown in Figure 5-8.

Table 5-4. Project Monitor Report Screen Commands

<table>
<thead>
<tr>
<th>Code</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cr (RETURN)</td>
<td>next page</td>
<td>Displays the next page of the report.</td>
</tr>
<tr>
<td>1</td>
<td>main</td>
<td>Displays the main Project Monitor report screen.</td>
</tr>
</tbody>
</table>
Table 5-4. **Project Monitor Report Screen Commands (continued)**

<table>
<thead>
<tr>
<th>Code</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>units/revenue</td>
<td>Displays columns for No. Units, Units/Hour, Revenue, and Revenue/Hour. Additional information shown on the report depends on the format options you choose. A sample units/revenue report is shown in Figure 5-11.</td>
</tr>
<tr>
<td>3-n</td>
<td>result groups</td>
<td>Displays the report based on result groups. Additional information shown on the report depends on the format options you choose. Result groups are described in the section Run Project Monitor.</td>
</tr>
<tr>
<td>X</td>
<td>exit</td>
<td>Returns you to the main Project Monitor display screen.</td>
</tr>
</tbody>
</table>

The following are representative Project Monitor reports.
### Figure 5-9. **Current by Operator, by Group, by Hour**

<table>
<thead>
<tr>
<th>GROUP</th>
<th>USER</th>
<th>OPERATOR</th>
<th>HOUR</th>
<th>HOURS</th>
<th>CALLS</th>
<th>CALLS</th>
<th>IN-BOUND</th>
<th>OUT-BOUND</th>
<th>FIRST CALLS</th>
<th>RE-CALLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guest</td>
<td>GST6</td>
<td>GUEST</td>
<td>11 AM</td>
<td>0.16</td>
<td>3</td>
<td>18.4</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Guest</td>
<td>GST7</td>
<td>GUEST</td>
<td>11 AM</td>
<td>0.79</td>
<td>2</td>
<td>2.5</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>***</td>
<td></td>
<td></td>
<td></td>
<td>0.95</td>
<td>5</td>
<td>5.2</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>TOT:</td>
<td></td>
<td></td>
<td></td>
<td>0.95</td>
<td>5</td>
<td>5.2</td>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
</tr>
</tbody>
</table>

(cr=next page, 1=main, 2=units/revenue, 3-5=result groups, X=exit)
### Figure 5-10. Project-to-Date by Source Queue

<table>
<thead>
<tr>
<th>SOURCE</th>
<th>HOURS</th>
<th>CALLS /HOUR</th>
<th>CALLS IN-</th>
<th>OUT-</th>
<th>FIRST</th>
<th>RE-</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL IN ONE</td>
<td>2.17</td>
<td>25</td>
<td>11.5</td>
<td>25</td>
<td>0</td>
<td>25</td>
</tr>
<tr>
<td>TOT</td>
<td>2.17</td>
<td>25</td>
<td>11.5</td>
<td>25</td>
<td>0</td>
<td>25</td>
</tr>
</tbody>
</table>
### Figure 5-11. Project-to-Date, by Operator, by Hour, Units/Revenue

<table>
<thead>
<tr>
<th>HOUR</th>
<th>HOURS</th>
<th>CALLS</th>
<th>AVG</th>
<th>AVG.</th>
<th>NO.</th>
<th>UNITS</th>
<th>REVENUE</th>
<th>REVENUE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>08 AM</td>
<td>0.03</td>
<td>5</td>
<td>0.38</td>
<td>0.07</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
<tr>
<td>09 AM</td>
<td>0.12</td>
<td>4</td>
<td>1.76</td>
<td>0.12</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
<tr>
<td>10 AM</td>
<td>0.05</td>
<td>5</td>
<td>0.56</td>
<td>0.07</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
<tr>
<td>11 AM</td>
<td>0.16</td>
<td>7</td>
<td>1.30</td>
<td>0.09</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
<tr>
<td>02 PM</td>
<td>0.11</td>
<td>2</td>
<td>3.32</td>
<td>0.06</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
<tr>
<td>03 PM</td>
<td>1.62</td>
<td>1</td>
<td>2.23</td>
<td>95.18</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
<tr>
<td>04 PM</td>
<td>0.05</td>
<td>1</td>
<td>0.30</td>
<td>2.96</td>
<td>0</td>
<td>0</td>
<td>$0.00</td>
<td>$0.00</td>
</tr>
</tbody>
</table>

TOT:   2.17  25  1.20  4.00  0   0.0  $0.00    $0.00

(cr=next page, 1=main, 2=units/revenue, 3-3=result groups, X=exit)
**User Monitor**

The User Monitor feature allows a user to view and access the screens being displayed by an agent who is in an operations or development guide in either test or live mode. The user who initiates user monitoring can view, input keystrokes, and send messages to the agent’s screen. This feature can be useful for training, guidance, or input validation.

In this section, the person who controls the monitor session is referred to as the *supervisor*, and the person being monitored is referred to as the *agent*.

*Note:* This feature is available for ASCII terminals; it is not available for use with EDGE Client.

**Modes**

User Monitor modes provide different capabilities for the supervisor. To switch between modes, the supervisor enters CTRL+T. The User Monitor modes are:

- **Coach Mode.**
- **Message Mode.**
- **View Mode.**
Coach Mode

In coach mode, the supervisor can view and enter input on the agent’s screen. Any keystroke entered from the supervisor’s terminal is displayed on the agent’s screen as input. If the agent has not been locked out, both the supervisor and the agent can enter keystrokes, even at the same time, in which case the keystrokes are intermixed. This mode can be useful in helping new agents. It can also be used to enter a password or code known only to the supervisor.

Message Mode

The message mode allows the supervisor to display a message on the agent’s screen. The message is displayed on the status line of the screen for both the supervisor and agent. As the supervisor types the message, it is displayed on the agent’s screen. The message can be any length and will scroll on both screens. When the supervisor presses ENTER, the message is cleared and the cursor returns to the beginning of the message bar. Only the supervisor can write messages. The agent can only view the message.

View Mode

In view mode the supervisor can view the agent’s screen, but cannot affect the screen by entering keystrokes. This is the mode in effect when a session starts.
Access Requirements

A supervisor must have the following user ID settings to be able to access user monitoring:

- In EDGE Developer, access to the Users area.
- In ASCII, access to the System Supervisor Control menu and a status level of at least 20.

Restrictions

A user monitoring session can only be started between two terminals of the same EDGE terminal type (for example, two VT100 terminals). All terminal types supported by the EDGE system, except EDGE Client, can run a monitor session. If an agent is running a guide under a terminal type that is incompatible with the supervisor’s terminal, the user ID is displayed, but cannot be monitored.

Begin User Monitor

To begin a User Monitor session, select the following:

Users area /Users tab /User Monitor

A dialog box similar to the following is displayed:
To continue, select Yes. If agents are logged on to ASCII terminals, the user selection screen is displayed. If no users are logged on, the wait screen is displayed; for more information, see the section When No Agents are Logged On.

**User Selection Screen**

The user selection screen displays a list of all agents on the system currently running guides, whether in the operations or development guide, or test or live mode.
Commands

/search Searches for the user ID specified. To begin the search, enter a slash (/). A prompt is displayed on the status line. Enter the user ID to be searched for.

q Quits the monitor session.

r Refreshes the user selection screen. If agents log on after the User Monitor session is started, they will be displayed when the screen is refreshed.

up/down arrows Moves the cursor selection up or down when you press the up and down arrow keys. The arrow key is also used to move to the next or previous page.
n Notifies the agent that a monitor session has been started. You must specify this option before starting the session with the agent. When notified, the message, “Monitored by: User ID” is displayed at the top right corner of the user's screen. *User ID* is the ID of the supervisor. When the session is ended, the message, “User ID ended monitoring,” is displayed on the user's screen.

ENTER Starts a monitor session with the selected agent.
Session Commands

While in a user monitoring session, the following commands are available:

CTRL+C Quits the monitor session and returns to the user selection screen. If the notify option was selected, the agent is notified that the session has ended.

CTRL+L Locks and releases the unlock when in coach mode. When locked out, the agent is locked out from entering keystrokes. When CTRL+L is pressed again, the lock is released and the agent is able to enter keystrokes. When the supervisor leaves coach mode or quits the session, the lock is automatically released.

CTRL+T Toggles between monitor modes. The mode is displayed on the error line at the right-hand side. The modes toggle in the following order: view, coach, and message.

Note: You can specify another key to use as the toggle key by setting the TANDEM_TOGGLE environment variable. For more information, see Appendix AB, EDGE Environment Variables.

When No Agents are Logged On

If there are no agents running a guide when User Monitor is selected, the message shown in Figure 5-12 is displayed.
Currently, there are no users logged into a guide. Every second, I will check again to see if users have logged in. You will go into monitoring mode when the first user logs in. If you wish to exit this wait mode, press <ctrl-C> or <break>. Once monitor mode is entered, you can exit at any time by pressing <ctrl-C> or <break>.

**Figure 5-12. User Monitor with No Users**

If you do not exit, the process is placed in a polling mode which waits for a user to log into a guide. When a user logs into a guide, the user selection screen is displayed.
Correspondence

Overview

The Correspondence module allows you to use text, user-defined fields, system fields, and data from your files to produce documents ranging from simple acknowledgment letters to complex preprinted forms. Correspondence allows you to:

- Access data in up to 200 files per report.
- Perform IF-ELSE comparisons for special text.
- Enter and format text.
- Extract partial field values.
- Specify margins, spacing, and page length of correspondence.
- Format labels.
- Send print commands to the printer.
<table>
<thead>
<tr>
<th>Truncating or Rounding Numeric Fields</th>
</tr>
</thead>
</table>

In default configuration, numeric field values in Correspondence are rounded off if they contain more decimal places than allowed by the field definition. In Ad Hoc Reports and EQL, however, numeric field values with too many decimal places are truncated, not rounded off.

Correspondence can be set up to truncate numeric values by setting an environment variable called CORRPRINT_TRUNC_AS_EQL. This environment variable is described in Appendix AB.
Creating Correspondence

The Correspondence module is available at the project level only. To open Correspondence, select the following:

Project / Reports area / Correspondence tab

A page similar to the following is displayed:

![Correspondence Tab](image)

The Correspondence tab allows you to create new correspondence, or to delete, modify, or print existing correspondence. When you
click either the Add or Modify button, a page similar to the following is displayed:

![Image of the SOFTSELL (Correspondence) - ORDER window]

**Properties:**

- **Command box**
  - The area of the screen where text and fields are typed. For information on how to enter data here, see the section [Basic Editing](#).
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correspondence</td>
</tr>
<tr>
<td>File</td>
</tr>
<tr>
<td>Last modified by</td>
</tr>
<tr>
<td>Last modified on</td>
</tr>
<tr>
<td>Left margin</td>
</tr>
<tr>
<td>Page length</td>
</tr>
<tr>
<td>Print line numbers</td>
</tr>
<tr>
<td>Printer</td>
</tr>
<tr>
<td>Right margin</td>
</tr>
</tbody>
</table>
Basic Editing

A correspondence report is created by entering fields, line commands, and text in the Correspondence page. The following indicate the rules for using Correspondence.

**Fields**

To specify a field, enter the field ID enclosed in brackets, for example, [ACCT]. To specify the internal value of the field, precede the field ID with an equal sign, for example, [=ACCT]. The fields are displayed formatted according to the field definition unless you use Field Format Codes to specify the display format.

To fix the exact position of a field on the page at a specific point, use the > character; for example, >[ACCT]. The field is always printed in the position on the page indicated by the >, with no adjustments made based on proportional fonts.

To specify a single value of a multivalue field, enter the value number following the field name separated by a comma, for example, [PRICE,1]. To print all values, enter an asterisk: [PRICE,*]. If the correspondence contains more than one multivalued field on the same line, for example, [x,*] [y,*], the last value of the multivalued field x and the first value of the multivalued field y, will be on the same line. To ensure they are on different lines, put them on separate lines in the correspondence.
For information on how multivalues are formatted, see the section *Multivalue Display Commands*.

If you use a field from a file other than the file defined for the correspondence, you must identify the file. To do this, use a *File abbreviation* as a prefix to the field ID; for example, [C-ACCT].

System fields can also be placed in a correspondence.

**Line Commands**

Line commands allow you to control the way the correspondence is formatted and printed.

**Text**

Text is displayed as you have typed it. Do not enclose text in quotation marks or brackets unless you want those characters in your correspondence.
Field Format Codes

Field format codes are used to display fields in a format different from the field definition. The codes are entered in the text of your correspondence.

Note: If you use a particular field format for several reports, it may be more efficient to create a Synonym field.

Field format codes are listed below and described in the following sections.

- Date Format Codes.
- Group Extract Code.
- Justification and Length Codes.
- Mask Character Codes.
- Text Extract Code.
Caution! Formatting fields may have an effect on IF-ELSE Commands. For example, a numeric field that is formatted with dollar signs ($500) will be compared as alphanumeric data, not numeric. In this case, $60 will be considered greater than $500. It is recommended that if you do comparisons of numeric, date, or time fields, you use an equal sign with the field ID ([=ID]) to force appropriate comparisons. (For an explanation of the use of equal signs and brackets, see the section Basic Editing.)

Date Format Codes

Date format codes allow you to format the date in a correspondence. Date format codes and examples are listed in Table 6-1.

Note: Be sure to enter an equal sign (=) before the field name; for example, [=ORD.DATE;D].

The date format codes listed in Table 6-1 are based on the Country Code set to United States. Other country settings could change the
result of the date format codes. The Country Calling Code is specified in the Site Setup screen.

**Table 6-1. Date Format Codes**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>[=field;D]</td>
<td>Formats date as Mon DD YYYY.</td>
<td>Apr 15 1997</td>
</tr>
<tr>
<td>[=field;D-]</td>
<td>Formats date as MM-DD-YYYY.</td>
<td>04-15-1997</td>
</tr>
<tr>
<td>[=field;D/]</td>
<td>Formats date as MM/DD/YYYY.</td>
<td>04/15/1997</td>
</tr>
<tr>
<td>[=field;DD]</td>
<td>Formats date as DD.</td>
<td>15</td>
</tr>
<tr>
<td>[=field;DMA]</td>
<td>Formats date as Month.</td>
<td>April</td>
</tr>
<tr>
<td>[=field;DM]</td>
<td>Formats date as MM.</td>
<td>04</td>
</tr>
<tr>
<td>[=field;DY]</td>
<td>Formats date as YYYY.</td>
<td>1997</td>
</tr>
</tbody>
</table>
Group Extract Code

The group extract code allows you to extract a group of text or data that is separated by a delimiter. The group extract code is shown in Table 6-2 and examples are listed in Table 6-3.

Table 6-2. Group Extract Code

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[field;G{m}*n]</td>
<td>Extracts text or numbers from a field that is divided by delimiters. Spaces can be delimiters.</td>
</tr>
<tr>
<td></td>
<td>{m}  The number of groups to skip before extracting; always begins from left to right; defaults to zero.</td>
</tr>
<tr>
<td></td>
<td>*    The delimiter symbol.</td>
</tr>
<tr>
<td></td>
<td>n    The number of groups to display; always begins from left to right.</td>
</tr>
</tbody>
</table>
### Table 6-3. Group Extract Formatting Examples

<table>
<thead>
<tr>
<th>Group Extract Code</th>
<th>Data Displayed by Field Definition</th>
<th>Data Displayed by Group Extract Code</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[field;G/1]</code></td>
<td>04/02/1997</td>
<td>04</td>
<td>Skips zero groups separated by a slash and extracts one group, up to the first delimiter (/).</td>
</tr>
<tr>
<td><code>[field;G1/2]</code></td>
<td>04/02/1997</td>
<td>02/1997</td>
<td>Skips one group separated by a slash and extracts two groups.</td>
</tr>
<tr>
<td><code>[field;G2-1]</code></td>
<td>714-622-6200</td>
<td>6200</td>
<td>Skips two groups separated by a dash and extracts one group.</td>
</tr>
<tr>
<td><code>[field;G1 1]</code></td>
<td>ABC DEF GHI</td>
<td>DEF</td>
<td>Skips one group separated by a space and extracts one group.</td>
</tr>
</tbody>
</table>
Correspondence Field Format Codes

Justification and length codes allow you to change the justification and length of data for a correspondence. Justification and length codes are listed in Table 6-4 and examples are shown in Table 6-5.

**Table 6-4. Justification and Length Codes**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[field;L]</td>
<td>Left-justifies the data.</td>
</tr>
<tr>
<td>[field;R]</td>
<td>Right-justifies the data.</td>
</tr>
<tr>
<td>[field;L#n]</td>
<td>Displays n characters, counting from the left.</td>
</tr>
<tr>
<td>[field;R#n]</td>
<td>Displays n characters, counting from the right.</td>
</tr>
</tbody>
</table>

**Table 6-5. Justification and Length Formatting Examples**

<table>
<thead>
<tr>
<th>Justification and Length Code</th>
<th>Data by Field Definition</th>
<th>Displayed Data</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>[field;L]</td>
<td>123ABC</td>
<td>123ABC</td>
<td>Left-justifies the data.</td>
</tr>
<tr>
<td>[field;R]</td>
<td>123ABC</td>
<td>123ABC</td>
<td>Right-justifies the data.</td>
</tr>
<tr>
<td>[field;L#3]</td>
<td>123ABC</td>
<td>123</td>
<td>Displays three characters, counting from the left.</td>
</tr>
<tr>
<td>[field;R#3]</td>
<td>123ABC</td>
<td>ABC</td>
<td>Displays three characters, counting from the right.</td>
</tr>
</tbody>
</table>

Justification and Length Codes
The mask character codes allow you to justify fields, format time, display fields as uppercase or lowercase, and display only alphabetic characters or numbers. Mask character codes are shown in Table 6-6 and examples are listed in Table 6-7.

**Table 6-6. Mask Character Codes**

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[field;MCA]</td>
<td>Shows only alphabetic characters.</td>
</tr>
<tr>
<td>[field;MCL]</td>
<td>Shows characters as lowercase.</td>
</tr>
<tr>
<td>[field;MCN]</td>
<td>Shows only numbers.</td>
</tr>
<tr>
<td>[field;MCT]</td>
<td>Capitalizes the first letter of each word.</td>
</tr>
<tr>
<td>[field;MCU]</td>
<td>Shows characters as uppercase.</td>
</tr>
<tr>
<td>[field;ML]</td>
<td>Left-justifies the field.</td>
</tr>
<tr>
<td>[field;MR]</td>
<td>Right-justifies the field.</td>
</tr>
<tr>
<td>[field;MTH]</td>
<td>Shows time with AM or PM.</td>
</tr>
<tr>
<td>[field;MT]</td>
<td>Shows time as a 24-hour clock.</td>
</tr>
<tr>
<td>Mask Character Code</td>
<td>Data by Field Definition</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>[field;MCA]</td>
<td>ABC123</td>
</tr>
<tr>
<td>[field;MCL]</td>
<td>ABC CORP</td>
</tr>
<tr>
<td>[field;MCN]</td>
<td>ABC123</td>
</tr>
<tr>
<td>[field;MCT]</td>
<td>abc CORP</td>
</tr>
<tr>
<td>[field;MCU]</td>
<td>abc corp</td>
</tr>
<tr>
<td>[field;ML]</td>
<td>123ABC</td>
</tr>
<tr>
<td>[field;MR]</td>
<td>123ABC</td>
</tr>
<tr>
<td>[field;MTH]</td>
<td>48000</td>
</tr>
<tr>
<td>[field;MT]</td>
<td>48000</td>
</tr>
</tbody>
</table>
**Text Extract Code**

The text extract code allows you to extract part of a field value for a correspondence. The text extract code is shown in Table 6-8 and examples are listed in Table 6-9.

### Table 6-8. Text Extract Code

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[field;T{m,n}]</code></td>
<td>Extracts data from a field starting at position <code>m</code> for <code>n</code> number of characters.</td>
</tr>
<tr>
<td><code>{m}</code> is optional. If not included, the extract begins at the first position from the left or right, depending on the field justification.</td>
<td></td>
</tr>
</tbody>
</table>

### Table 6-9. Text Extract Formatting Examples

<table>
<thead>
<tr>
<th>Text Extract Code</th>
<th>Data by Field Definition</th>
<th>Displayed Data</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>[field;T3]</code></td>
<td>123ABC</td>
<td>123</td>
<td>Extracts and shows three characters starting from the left in a left-justified field.</td>
</tr>
<tr>
<td><code>[field;T3]</code></td>
<td>123ABC</td>
<td>ABC</td>
<td>Extracts and shows three characters starting from the right in a right-justified field.</td>
</tr>
</tbody>
</table>
### Table 6-9. Text Extract Formatting Examples (continued)

<table>
<thead>
<tr>
<th>Text Extract Code</th>
<th>Data by Field Definition</th>
<th>Displayed Data</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>[field;T3,4]</td>
<td>123ABC</td>
<td>3ABC</td>
<td>Extracts four characters starting with the third character from the left in a left-justified field.</td>
</tr>
<tr>
<td>[field;T3,4]</td>
<td>123ABC</td>
<td>123A</td>
<td>Extracts four characters starting with the third character from the right in a right-justified field.</td>
</tr>
</tbody>
</table>
System Fields

Correspondence can include the value stored in any of the fields in the SYS file listed in Table 6-10. To use them in correspondence, you must put the field inside brackets [ ] and include the dollar sign ($) before the field name. (The dollar sign distinguishes system fields from user-defined fields; see the section Fields for more information.) You can also use these fields in Ad Hoc Reports and EQL statements.

The following system fields are affected by the Date Offset property in the Task Scheduler.

- $CDATE
- $CMONTH
- $DATE
- $DAY
- $DOW
- $MONTH
- $YEAR

Table 6-10. System Fields

<table>
<thead>
<tr>
<th>System Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[$CDATE]</td>
<td>The current date formatted Month DD, YYYY (August 04, 2004).</td>
</tr>
<tr>
<td>[$CMONTH]</td>
<td>The name of the current month (for example, January, February).</td>
</tr>
<tr>
<td>[$CORRID]</td>
<td>The name of the correspondence.</td>
</tr>
</tbody>
</table>
### Table 6-10. System Fields (continued)

<table>
<thead>
<tr>
<th>System Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[$DAY]</td>
<td>The number of the current day of the month (1-31).</td>
</tr>
<tr>
<td>[$DOW]</td>
<td>The name of the current day of the week (Monday, Tuesday, etc.).</td>
</tr>
<tr>
<td>[$INDEX]</td>
<td>A counter whose value is increased each time the field is printed.</td>
</tr>
<tr>
<td>[$MONTH]</td>
<td>The number of the current month (1-12).</td>
</tr>
<tr>
<td>[$PAGE]</td>
<td>The current page number of the correspondence. This is a total of all the pages printed for all the records.</td>
</tr>
<tr>
<td>[$PROJECT]</td>
<td>The project ID.</td>
</tr>
<tr>
<td>[$RPAGE]</td>
<td>The relative page number of the record. This starts at 1 for each new record.</td>
</tr>
<tr>
<td>[$YEAR]</td>
<td>The current year.</td>
</tr>
</tbody>
</table>
Line Commands

Line commands allow you to control the way the final document is formatted and printed. They are identified by a backslash (\) in the first position of the line. All the fields and text that follow the line command are affected by the line command until an opposing line command ends the command. For example, the command \JUSTIFY justifies text at both right and left margins. To end justification, you must use the \NOJUSTIFY command.

Line commands must follow these formatting rules:

- Commands begin with a backslash (\) in the first position on a line, followed by the command name with no spaces after the backslash.
- The command name may be uppercase or lowercase.
- Only one command per line is allowed.
- If the command has a short name, it may be used.
- Numeric arguments in commands behave as follows:
  A value preceded by a plus (+) or minus (-) sign is added or subtracted to the current value of that function.
  A value without a + or - sign sets the function to that value.
Some examples of numeric arguments are shown in Table 6-11.

**Table 6-11. Line Commands with Numeric Arguments**

<table>
<thead>
<tr>
<th>Line Command Syntax</th>
<th>Example</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\COMMANDn</td>
<td>\LMARGIN 10</td>
<td>Sets the left margin at column 10.</td>
</tr>
<tr>
<td>\COMMAND +n</td>
<td>\LMARGIN +10</td>
<td>Adds 10 to the current left margin value.</td>
</tr>
<tr>
<td>\COMMAND -n</td>
<td>\LMARGIN -10</td>
<td>Subtracts 10 from the current left margin value.</td>
</tr>
</tbody>
</table>

Line commands affect the functions listed below. Commands for each function are described in the following sections.

- Field Formatting Commands.
- IF- ELSE Commands.
- Label Formatting Commands.
- Line Formatting Commands.
- Margin Setup Commands.
- Multivalue Display Commands.
- Page Setup Commands.
• Printer Operation Commands.
• Read File Commands.

The formatting commands listed in Table 6-12 are automatically set when you create correspondence. To use different formatting, use the corresponding line command.

### Table 6-12. Default Line Command Settings

<table>
<thead>
<tr>
<th>Default Setting</th>
<th>Description</th>
<th>To Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>\BMARGIN 0</td>
<td>The bottom margin is set to zero. (No blank lines are displayed at the bottom of the page.)</td>
<td>\BMARGIN n</td>
</tr>
<tr>
<td>\BOT 0</td>
<td></td>
<td>\BOT n</td>
</tr>
<tr>
<td>\BM 0</td>
<td></td>
<td>\BM n</td>
</tr>
<tr>
<td>\FORMAT \FMT</td>
<td>Format mode is on. Fields that do not have format codes specified are displayed according to their field definition. Fields that have format codes specified are displayed according to their format code.</td>
<td>\NOFORMAT \NFMT</td>
</tr>
<tr>
<td>\LENGTH 66</td>
<td>The page length is 66 lines.</td>
<td>\LENGTH n</td>
</tr>
<tr>
<td>\LMARGIN 0</td>
<td>The left margin is set at 0.</td>
<td>\LMARGIN n</td>
</tr>
<tr>
<td>\NOFILL</td>
<td>Lines are not packed. Each line is printed as it is shown on the screen.</td>
<td>\FILL</td>
</tr>
</tbody>
</table>
### Table 6-12. Default Line Command Settings (continued)

<table>
<thead>
<tr>
<th>Default Setting</th>
<th>Description</th>
<th>To Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>\NOJUSTIFY \NOADJUST \NA</td>
<td>Margins are unjustified or ragged.</td>
<td>\JUSTIFY \ADJUST n \AD n</td>
</tr>
<tr>
<td>\NOVERT \NV</td>
<td>Vertical compression mode is off. Any blank lines are printed.</td>
<td>\VERT</td>
</tr>
<tr>
<td>\PAGE 1</td>
<td>Page numbering begins at 1.</td>
<td>\PAGE n \BP n</td>
</tr>
<tr>
<td>\RMARGIN 132</td>
<td>The right margin is set at 132.</td>
<td>\RMARGIN n \RM n</td>
</tr>
<tr>
<td>\SPACE 1</td>
<td>Lines are single-spaced.</td>
<td>\SPACE n</td>
</tr>
<tr>
<td>\TMARGIN 0 \TOP 0 \TM 0</td>
<td>The top margin is set to 0. (No blank lines are displayed at the top of the page.)</td>
<td>\TMARGIN n \TOP n \TM n</td>
</tr>
</tbody>
</table>
Field Formatting Commands

A field formatting command affects the format of fields until another field formatting command is encountered. The field formatting commands are listed in Table 6-13.

Table 6-13. Field Formatting Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FORMAT</td>
<td>Turns on format mode; any specified field format codes are enabled. Fields that do not have format codes specified are displayed according to their field definition. Fields that have format codes specified are displayed according to their format code. FORMAT is the default setting for field format.</td>
</tr>
<tr>
<td>NOFORMAT</td>
<td>Turns off format mode; any specified field format codes are disabled. Fields that do not have format codes specified are displayed as they are stored in the record. Fields that have format codes specified are displayed according to their format code.</td>
</tr>
</tbody>
</table>

IF-ELSE Commands

IF-ELSE commands allow you to include data on a conditional basis. IF-ELSE commands are described in Table 6-14.
**Note:** Multiple IF-ELSE commands can adversely affect the speed of correspondence production.

### Table 6-14. IF-ELSE Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| `\IF [field] relational operator [field] or “literal text”` | Begins the IF statement. The field in the expression can be a multivalue field [field,value position] or a field with a format code [field;code]. One of the following relational operators can be specified:  
  - `=` equal to  
  - `#` not equal to  
  - `>` greater than  
  - `<` less than  
  - `>=` greater than or equal to  
  - `<=` less than or equal to |
| `\ELSEIF [field] relational operator [field] or “literal text”` | Specifies an alternate IF case. |
| `\ELSE`                                      | Specifies an alternative when the IF or ELSEIF case is not met. |
| `\ENDIF`                                     | Ends the IF command. |
Label Formatting Commands

Label formatting commands allow you to format labels or label-like correspondence. For example, `\LABEL 3 5 1 26 3 21` specifies three labels across the page, five rows per label, one row between labels, and so on. (You must specify all the arguments (a–f) in the label definition.) When invoked, this command automatically changes the document to `\NOFILL` mode.

The label formatting commands are listed in Table 6-15.

**Table 6-15. Label Formatting Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>\LABEL a b c d e f</code></td>
<td>a  Number of labels (records) across the page.</td>
</tr>
<tr>
<td></td>
<td>b  Number of rows on each label.</td>
</tr>
<tr>
<td></td>
<td>c  Number of rows between labels (vertical spacing).</td>
</tr>
<tr>
<td></td>
<td>d  Number of characters per label (width of label).</td>
</tr>
<tr>
<td></td>
<td>e  Number of characters between labels (horizontal spacing).</td>
</tr>
<tr>
<td></td>
<td>f  Number of labels per page.</td>
</tr>
<tr>
<td><code>\ENDLABEL</code></td>
<td>Turns off the label command.</td>
</tr>
</tbody>
</table>
Line formatting commands allow you to specify line format options for correspondence. The line formatting commands are listed in Table 6-16.

Table 6-16. Line Formatting Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
</table>
| \*               | Begins comment lines or any lines you don’t want to print on the correspondence. A space must follow the *.
| \BR              | In \FILL mode, causes a break line. A new paragraph follows.                                          |
| \FILL (\FI)      | Begins fill mode and packs each line until full. Prints any overflow on following lines.               |
| \LINE-LENGTH n   | Determines the line length by using the left margin value plus the number specified. It then sets the right margin. The number specified is the number of characters allowed for each line. For example: If the left margin is 10 and you specify \LL 60, the right margin is set to 70. |
| \NOFILL          | Turns off fill mode. Each line is printed as a single line. Default.                                  |
| \NOVERT          | Turns off vertical compression mode.                                                                  |
| \ROW n           | Skips forward to the text row number specified. The top and bottom margins of the correspondence are protected areas. This command applies to text rows only. |
Margin Setup Commands

Table 6-16. Line Formatting Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\SPACE n</td>
<td>Spaces down the number of lines specified.</td>
</tr>
<tr>
<td>\SP n</td>
<td></td>
</tr>
<tr>
<td>\SPACING n</td>
<td>Sets the spacing between lines to the number specified.</td>
</tr>
<tr>
<td>\LS n</td>
<td>The default is single spacing.</td>
</tr>
<tr>
<td>\VERT</td>
<td>Turns on vertical compression mode. Does not print blank lines.</td>
</tr>
<tr>
<td>\</td>
<td>Introduces a backslash as the first character in a line of text.</td>
</tr>
<tr>
<td></td>
<td>This alerts the system to treat the third backslash as text, not as a line command.</td>
</tr>
</tbody>
</table>

Margin setup commands allow you to specify margins for correspondence. The margin setup commands are listed in Table 6-17.

Table 6-17. Margin Setup Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\ADJUST x</td>
<td>Justifies text based on x setting; x can be one of the following:</td>
</tr>
<tr>
<td>\AD x</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>Left-justify only.</td>
</tr>
<tr>
<td>R</td>
<td>Right-justify only.</td>
</tr>
<tr>
<td>C</td>
<td>Center between margins.</td>
</tr>
</tbody>
</table>
Table 6-17. Margin Setup Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>In \FILL mode, adjusts both margins and left justifies break lines. Same as \JUSTIFY.</td>
</tr>
<tr>
<td>BMARGIN n</td>
<td>Sets the bottom margin to the number of lines specified.</td>
</tr>
<tr>
<td>BOT n</td>
<td></td>
</tr>
<tr>
<td>BM</td>
<td></td>
</tr>
<tr>
<td>INDENT n</td>
<td>Indents text the number of spaces specified from the left margin.</td>
</tr>
<tr>
<td>IN n</td>
<td></td>
</tr>
<tr>
<td>JUSTIFY</td>
<td>Justifies text to left and right margins. Same as \ADJUST B.</td>
</tr>
<tr>
<td>LMARGIN n</td>
<td>Sets the left margin to the number specified.</td>
</tr>
<tr>
<td>LM n</td>
<td></td>
</tr>
<tr>
<td>NOJUSTIFY</td>
<td>Turns off \ADJUST mode. Margins are ragged or as typed. Default.</td>
</tr>
<tr>
<td>NA n</td>
<td></td>
</tr>
<tr>
<td>NOVERT</td>
<td>Prints all blank lines. Default.</td>
</tr>
<tr>
<td>NV</td>
<td></td>
</tr>
<tr>
<td>RMARGIN n</td>
<td>Used in \FILL mode to set the right margin to the number specified.</td>
</tr>
<tr>
<td>RM n</td>
<td></td>
</tr>
<tr>
<td>TEMPINDENT n</td>
<td>Temporarily indents text from the left margin. Affects only the line following the command.</td>
</tr>
<tr>
<td>TI n</td>
<td></td>
</tr>
<tr>
<td>TMARGIN n</td>
<td>Sets the top margin to the number of lines specified.</td>
</tr>
<tr>
<td>TOP n</td>
<td></td>
</tr>
<tr>
<td>TM n</td>
<td></td>
</tr>
</tbody>
</table>
Multivalue Display Commands

Multivalue display commands allow you to specify how multivalues are to be displayed. The multivalue display commands are listed in Table 6-18.

Caution! These commands are intended to be used in \FILL (\Fi) mode only. If you are not in fill mode, data will print as an extremely long line unless your printer wraps automatically. In this case, however, EDGE will not know how many lines of output the printer is printing.

Table 6-18. Multivalue Display Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\VMCR</td>
<td>Changes all value marks into carriage return/linefeed characters; that is, a value mark acts as a paragraph mark.</td>
</tr>
<tr>
<td>\VMDEFAULT</td>
<td>The default case. If a multi-valued field is specified without an explicit value reference, all the values are printed and no error occurs.</td>
</tr>
<tr>
<td>\VMSP</td>
<td>Changes all value marks to spaces.</td>
</tr>
</tbody>
</table>

Page Setup Commands

Page setup commands allow you to specify paging options for correspondence. The page setup commands are listed in Table 6-19.

Note: When the document exceeds a page, a page break occurs. A system-forced or soft page break appears as a single dashed line on
the screen. A page break controlled by a command (a hard break) appears as a double dashed line.

### Table 6-19. Page Setup Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\LENGTH n \PL n</td>
<td>Sets page length to the number of rows specified.</td>
</tr>
<tr>
<td>\PAGE n \BP n</td>
<td>Advances to the first line of the next page and assigns the next relative page number based on the number specified. The default page number is +1.</td>
</tr>
</tbody>
</table>

Printer operation can be optimized by using printer escape commands, form feed commands, and the Printer Setup function in EDGE.

#### Printer Escape Commands

Printer escape commands can be embedded in your correspondence. They are used to perform special printer functions and are dependent on the type of printer you have. Refer to your printer manual for a list of supported escape commands. You can use printer escape commands and UNIX line printer commands.
The printer configuration you set through the Set Server Printer command in Server Setup applies when you print your correspondence. To set the server printer, select Set Server Printer on the File menu.

Table 6-20 lists some basic ASCII printer escape commands. Before using a command, refer to your printer manual to make sure that these sequences are supported.

### Table 6-20. Printer Escape Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>ASCII</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\b</td>
<td>8</td>
<td>Backspace.</td>
</tr>
<tr>
<td>\</td>
<td>92</td>
<td>Backslash.</td>
</tr>
<tr>
<td>\d</td>
<td>127</td>
<td>Delete.</td>
</tr>
<tr>
<td>\e {optional printer codes}</td>
<td>27</td>
<td>Escape; can be followed by optional printer codes, if supported. For example, some printers use the following sequence to start bold: \e(s3B. If no code follows the \e, causes the printer to quit printing.</td>
</tr>
<tr>
<td>\f</td>
<td>12</td>
<td>Form feed; go to the top of a new page.</td>
</tr>
<tr>
<td>\n</td>
<td>10</td>
<td>New line.</td>
</tr>
<tr>
<td>\r</td>
<td>13</td>
<td>Carriage return.</td>
</tr>
<tr>
<td>\s</td>
<td>32</td>
<td>Space.</td>
</tr>
<tr>
<td>\t</td>
<td>9</td>
<td>Tab.</td>
</tr>
</tbody>
</table>
Form Feed Commands

Using the \FORMFEED command rather than line counting can increase the speed of printing. If \FORMFEED is not used, the printer uses the \LENGTH command to determine the top of the next page. If you use \FORMFEED, it should be at the top of your correspondence, before the text, to further streamline the printing process. The form feed commands are described in Table 6-21.

Note: Use the \FORMFEED command to improve printing performance.

Table 6-21. Form Feed Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\FORMFEED \FF</td>
<td>Signals the printer that a new page or form should start.</td>
</tr>
<tr>
<td>\NOFORMFEED \NFF</td>
<td>Turns off the \FORMFEED command and returns to using \LENGTH to determine</td>
</tr>
<tr>
<td></td>
<td>the top of a page.</td>
</tr>
</tbody>
</table>
Read File Commands

Read file commands allow you to read specific records from a file to include in your correspondence. The read file commands are listed in Table 6-22.

Table 6-22. Read File Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\FILE filename {abbr}</td>
<td>Used with the \READ command to access data in another file. The file abbreviation can be used in place of the filename when referencing fields.</td>
</tr>
<tr>
<td>\READ</td>
<td>Reads the next record from the primary file.</td>
</tr>
<tr>
<td>\READ {file-ID field-ID}</td>
<td>Reads the file using the field value from the primary file. You may use the file ID or the file abbreviation. For example: \READ TRANS {INV.NO] reads the Transaction file looking for a Record ID value equal to the value of INV.NO in the primary file.</td>
</tr>
<tr>
<td>\READ {file-ID “value”}</td>
<td>Reads the file based on a literal value. You may use the file ID or the file abbreviation. The literal value must be in quotes. For example: \READ MSG “HOLIDAY” reads the MESSAGE file looking for a Record ID equal to “HOLIDAY.”</td>
</tr>
</tbody>
</table>
Ad hoc reports are individual reports that you create using data in EDGE. Reports can be run one at a time from the Ad Hoc Report page or several can be run at once using the Batch Reports page. Ad hoc reports can also be run from within the Tasks Area.

This chapter describes the following:

- Ad Hoc Reports.
- Batch Reports.
- Selection Lists.

The Reports area also has tabs for Correspondence and Monitors.
Ad Hoc Reports

The following types of ad hoc reports can be created:

Columnar  A columnar report presents columns of data. The number of columns depends on the fields you select. You can adjust the spacing between columns and change the order of the columns in the report.

Form  A form report allows you to create a report to print on a pre-established form, or to create your own form. You can place data at specific locations on the report by specifying row and column coordinates.

Freeform  A freeform report allows you to directly enter EQL statements to create your ad hoc report.

Labels  A label report allows you to specify the width, height, spacing, and other characteristics of labels.

Selection List  A selection list is a list of record IDs from a file and is created from specifications you provide. It does not produce a report, but can be used in functions such as ad hoc reports, correspondence, and queue seeding.

You can access Ad Hoc Reports from the EDGE Reports area or from the Reports area of any project. Reports created in the EDGE Reports area are available in all projects; reports created in a project’s Reports area apply to that project only.
When the Reports dialog box appears, click the Ad Hoc Report tab to bring it to the front as shown below.

![Reports dialog box]

The Ad Hoc Report tab allows you to create new ad hoc reports, or to delete, modify, or print an existing ad hoc report. From this tab, you may also set defaults for report headers and footers, printer, and terminal.

**Special Fields**

You can add information to an ad hoc report by using the special fields shown in Table 7-1. These fields supply the current value for the command to the report. For example, if the report contains the option \DATE, the current date is shown on the report when it is run.
Using backslash commands can reduce the time you spend modifying individual reports and selection criteria.

**Table 7-1. Special Fields**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\DATE</td>
<td>The current date in the format specified in your Country Code option. This field is affected by the Date offset parameter in the Task Scheduler.</td>
</tr>
<tr>
<td>\ENTRY1 - \ENTRY9</td>
<td>The storage fields for data requested during run-time. They are specified in the Prompt option.</td>
</tr>
<tr>
<td>\FILE</td>
<td>The name of the file used in the report.</td>
</tr>
<tr>
<td>\PROJECT</td>
<td>The name of the project where the report was created. Global-level reports show EDGE as the project name.</td>
</tr>
<tr>
<td>\REPORT</td>
<td>The ad hoc report name.</td>
</tr>
<tr>
<td>\SITE</td>
<td>The information entered under Report Heading on the Site Setup screen. See Chapter 3, Setting Up EDGE.</td>
</tr>
<tr>
<td>\TIME</td>
<td>The current system time in format HH:MM.</td>
</tr>
<tr>
<td>\TITLE</td>
<td>The report title as defined on the Title/Prompts tab.</td>
</tr>
<tr>
<td>\USER</td>
<td>The user ID of the person running the report.</td>
</tr>
<tr>
<td>$BOH</td>
<td>The current hour, formatted HH:00:00 (14:00:00).</td>
</tr>
<tr>
<td>$BOM</td>
<td>The date of the first day of the current month, formatted MM/01/YY (08/01/97).</td>
</tr>
</tbody>
</table>
Before creating ad hoc reports, you may wish to establish some default settings for them. You can set defaults for report headers and footers, printer, and terminal. To create one set of defaults for all projects, use the Set Defaults button on the EDGE Ad Hoc Report tab. To create project-by-project defaults, use the Set Defaults button on each project’s Ad Hoc Report tab. When you click the Set Defaults button, the Ad Hoc Report Defaults dialog box appears as shown in below.

### Table 7-1. Special Fields (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$BOQ</td>
<td>The date of the first day of the current quarter, formatted MM/DD/YY (07/01/97). The possible months are 01, 04, 07, and 10.</td>
</tr>
<tr>
<td>$BOW</td>
<td>The date of the beginning of the current week (Sunday), formatted MM/DD/YY (08/04/97).</td>
</tr>
<tr>
<td>$BOY</td>
<td>The date of the first day of the current year, formatted 01/01/YY (01/01/97).</td>
</tr>
<tr>
<td>$NOW</td>
<td>The current 24-hour time, formatted HH:MM:SS (09:32:55). See also $BOH and $TIME.</td>
</tr>
<tr>
<td>$TODAY</td>
<td>The current date, formatted MM/DD/YY (08/01/97).</td>
</tr>
</tbody>
</table>
Properties:

Centered When checked, specifies that the corresponding text (heading or footing) is centered on report pages. The default is cleared.

Footing The default text to display in the footing on reports. This can include up to four lines of text, backslash commands, and EQL footing options positioned on the bottom of the page. A blank line is inserted between the last row of the report and the first line of the footing.
Properties: (continued)

**Heading**  
The default text to display in the heading on reports. This can be up to four lines of text, backslash commands, and EQL heading options positioned at the top of the page. A blank line is inserted between the heading and the beginning of the report.

**Icons**  
The icons displayed in the center of the dialog box represent information that can be included in the heading or footing area of a report. To include icon information, position the cursor in the heading or footing text box and then click the desired icon.

*Figure 7-1* gives a brief description of each icon.

**Printer Length**  
The number of text lines per page for the printer. The default is 60 lines.

**Printer ID**  
The name of the printer to be used to print reports. There is no default.

**Width**  
The number of characters per text line for the printer. The default is 80 characters.

**Terminal Length**  
The number of text lines displayed on the screen. The default is 24 lines. The minimum is 24 lines; the maximum is the size of the screen.
**Properties: (continued)**

**Width**  The number of characters per text line displayed on the screen. The default is 80 characters. The minimum is 80 lines; the maximum is the size of the screen.

---

**Figure 7-1.  Ad Hoc Report Heading and Footing Icons**
Creating Ad Hoc Reports

The basic steps for creating or modifying an ad hoc report are as follows:

- Entering a Report Name.
- Selecting a Report Type and Source File.
- Setting Report Properties.
- Running Ad Hoc Reports.

When you click on Add or Modify on the Ad Hoc Report tab of the Reports dialog box, the Ad Hoc Instruction dialog box appears as shown below.
In this dialog box, you can enter the report name, and then use the Report Display tab to choose a report type and source file. Once you have set up the report in this manner, you can select report properties using the other tabs in the Ad Hoc Instruction dialog box. You can then run the report by clicking the Run Ad Hoc Report button on the Ad Hoc Report tab of the Reports dialog box.
Entering a Report Name

You must enter a name for the report in the Ad Hoc instruction text box before any other work can be accomplished. The tabbed pages of the dialog box do not become active until a name is entered.

The report name must be no more than 10 characters in length with no spaces. It must begin with a letter and be alphanumeric (although dots may also be used).

Selecting a Report Type and Source File

After entering a report name, you must select a report type and source file. You can select these properties from the Report Display tab. To select a report type, click the desired type in the Report type list box. To select a source file, click the desired file in the File list box.
Setting Report Properties

The tabbed pages in the Ad Hoc Instruction dialog box allow you to set the properties of an ad hoc report. Depending on the type of report selected, you may have as few as three or as many as eight tabbed pages to work with as indicated in Table 7-2.

Table 7-2. Availability of Ad Hoc Instruction Tabs

<table>
<thead>
<tr>
<th>Tab</th>
<th>Report Type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Columnar</td>
</tr>
<tr>
<td>Column Display</td>
<td>✓</td>
</tr>
<tr>
<td>Display Options</td>
<td>✓</td>
</tr>
<tr>
<td>Headings/Footings</td>
<td>✓</td>
</tr>
<tr>
<td>Label Setup</td>
<td></td>
</tr>
<tr>
<td>Output Device</td>
<td>✓</td>
</tr>
<tr>
<td>Record Selection</td>
<td>✓</td>
</tr>
<tr>
<td>Report Display</td>
<td>✓</td>
</tr>
<tr>
<td>Title/Prompts</td>
<td>✓</td>
</tr>
</tbody>
</table>

As you make selections in each tabbed page, instructions for building your report appear in the text box above the tabs.
The Column Display tab allows you to identify the fields for the columns of a report. This tab is available for columnar and label reports.

Double-click a grid cell to display this drop-down arrow. Click the drop-down arrow to see the drop-down list box.

**Properties:**

**Break/Total**
- Specifies whether a field requires a break line or total. To select either option, click Break or Total in the Break/Total drop-down list box.
- If you select Break, a break is made each time the value for the field changes. Generally, this is used for fields that you also sort.
Display Options Tab

Properties:  (continued)

Field ID
The ID of each field to be included as a column in the report. To add a field, click the corresponding Field ID in the Field ID drop-down list box.

Options
The break line delimiter; can be text, backslash commands, or EQL group option codes. If no delimiter is specified, asterisks (****) are used.

If you select Total, the field is totaled. A total line is used when you want to show the total on a numerical field when a break occurs.

The fields are displayed in the report columns in the order in which they are listed here.

The Display Options tab allows you to control report information and formatting. This tab is available for columnar and label reports. It is similar to the following:
Properties:

- **Check ...% of records**
  The percentage of records to be checked for the report. When selected, only the percentage of records specified are read. Out of the percentage read, only those that meet your query requirements are displayed. This property is saved with the report and used each time the report is run.
  
  There is no default.

- **Check ... records**
  The number of records to be checked for the report. When selected, only the number of records specified are read. Out of the number read, only those that meet your query requirements are displayed. This property is saved with the report and used each time the report is run.
  
  There is no default.
**Properties: (continued)**

**Display maximum of … records**

The maximum number of records to display in the report. This property is saved with the report and used each time the report is run.

There is no default.

**Grand total line display**

The text to appear on the grand total line of the report. This option applies when you have identified a field to be totaled in the Column Display tab.

There is no default.

You can specify the following types of information for the grand total display: text, backslash commands, or EQL grand total options.

**Line label**

When checked, specifies that the lines in the report are labeled. The line labels appear along the left edge of the report page. Report lines are labeled according to the following types:

<table>
<thead>
<tr>
<th>Line Type</th>
<th>Line Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title/heading</td>
<td>T</td>
</tr>
<tr>
<td>Record</td>
<td>R</td>
</tr>
<tr>
<td>Break</td>
<td>Number</td>
</tr>
<tr>
<td>Total</td>
<td>G</td>
</tr>
</tbody>
</table>

The default is cleared.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text file format</td>
<td>This property is only enabled if the Text file format property is checked. When checked, specifies to save report data in text file format. Also enables the Line label property. The default is cleared.</td>
</tr>
<tr>
<td>Generate selection list</td>
<td>When checked, specifies that a selection list will be created. If you abort the report display while the report is being generated, you will also abort the creation of the selection list. The default is cleared.</td>
</tr>
<tr>
<td>Default list</td>
<td>When selected, the temporary selection list will be created in the $EDGE/tmp directory and called process_ID.ers0. This property is enabled only if the Generate selection list property is checked. This is the default setting when the Generate selection list property is checked.</td>
</tr>
<tr>
<td>User named list</td>
<td>When selected, the selection list will be created with the name you specify in the corresponding text box. This property is enabled only if the Generate selection list property is checked.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Page format</th>
<th>Display record count</th>
<th>When checked, the total number of records that matched the report’s criteria is displayed.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>The default is cleared.</td>
</tr>
<tr>
<td>Display total &amp; break lines</td>
<td>When checked, specifies that only total and break lines are displayed, and detail lines are not.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is cleared.</td>
</tr>
<tr>
<td>Double space detail lines</td>
<td>When checked, specifies that detail lines are double-spaced on the report.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is cleared.</td>
</tr>
<tr>
<td>Remove default ID field</td>
<td>When checked, specifies that the record ID does not appear as the first column in the report.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is cleared.</td>
</tr>
<tr>
<td>Remove field heading</td>
<td>When checked, specifies that the column headings from the field definition do not appear in the report.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is cleared.</td>
</tr>
<tr>
<td>Remove page heading</td>
<td>When checked, specifies that no heading or footing appears on the page.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is cleared.</td>
</tr>
</tbody>
</table>
The Headings/Footings tab allows you to specify heading and footing information for an individual report. This tab is available for columnar and label reports.

When you specify heading and footing information on this tab, it overrides the settings on the Set Default tab. For a description of the properties and icons, see Setting Defaults.

Properties: (continued)

Remove paging

When checked, specifies to allow you to scroll through the report on the screen rather than moving from page to page by pressing the RETURN key.

The default is cleared.
**Label Setup Tab**

The Label Setup tab allows you to specify properties for labels. This tab is available only for label reports.

**Properties:**

- **Characters across label**
- **Characters between labels**
- **Compress null fields**
- **Indent from left margin**

The number of characters across one label.

The number of characters in the space between two labels (horizontal spacing).

When checked, specifies to eliminate blank lines created by fields that contain no value.

The number of characters to indent from the left side of the page.
Output Device Tab

Properties: (continued)

- **Labels across page**: The number of labels across the width of the page.
- **Lines between labels**: The number of lines between the labels (vertical spacing).
- **Lines per label**: The number of lines within each label.

The Output Device tab allows you to set terminal and printer specifications, and to specify information for transferring a file to the client and launching an application such as Excel or FileMaker. The destination of the report is determined at run time; for more information, see the section *Running Ad Hoc Reports*.

This tab is available for all report types.
Properties:

**Printer**  The characteristics of the printer.
- **Length**  The number of text lines per page for the printer. The default is 60 lines.
- **Printer ID**  The name of the printer to be used to print reports. There is no default.
- **Width**  The number of characters per text line for the printer. The default is 80 characters.

**Terminal**  The characteristics of the terminal.
- **Length**  The number of text lines displayed on the screen. The default is 24 lines.
**Record Selection Tab**

The Record Selection tab, shown below, allows you to select records for your report from individual record IDs, the source file, or a selection list. You can also specify conditions that must be met before the report is generated. The following are the options available:

- **Workstation file**: The file path and type of application to send the report to. It is recommended that if you send a report to the workstation, you create it in **Text file format**.
- **Launch application**: The name of the application to launch after the report is saved.
- **Path name**: The path, including filename, of where to send the report.
- **Prompt when run**: If checked, the path and application can be specified at run time.

**Properties:** (continued)

- **Width**: The number of characters per text line displayed on the screen. The default is 80 characters.

**Note:** The Ad Hoc Report Defaults dialog box determines the default condition of the properties in this section. The settings you make on this tab for a particular report, however, supersede the defaults.
record can be used. This tab is available for columnar, label, and selection list reports.

Properties:

Condition  The manner in which the corresponding field ID is compared to another field ID or value; can be any of the following:
Properties: (continued)

- beginning with
- containing
- ending with
- equal to
- greater/equal to
- greater than
- less/equal to
- less than
- not beginning with
- not containing
- not ending with
- not equal to

There is no default.

Conditions

The extent of the comparison, as follows:

- **Advanced**  Must meet the detailed conditions specified. When this condition is specified, the Type property is enabled.
- **All**     Must meet all conditions specified.
- **Any**    Must meet at least one of the conditions specified.
- **None**   Inverts the conditions specified. This is the default.
### Ad Hoc Reports

**Setting Report Properties**

<table>
<thead>
<tr>
<th><strong>Properties:</strong> (continued)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Field</strong></td>
<td>If checked, the entry in the Field ID/Value property is a field name. This property is display only.</td>
</tr>
<tr>
<td><strong>Field ID</strong></td>
<td>The field to be compared. There is no default.</td>
</tr>
<tr>
<td><strong>Field ID/Value</strong></td>
<td>The field or value used to compare with the Field ID. This can be a field, a value, or left null. If using a field, select from the pull-down list box. Anything that is typed is assumed to be a value. If specifying a value, do not enclose in quotation marks. EDGE Developer automatically adds them for you. If a user is specifying a date, they should use a standard date format not an EDGE internal date. The default is null.</td>
</tr>
<tr>
<td><strong>Records</strong></td>
<td>Specifies which records to compare; can be one of the following: <strong>All records</strong>  <strong>From selection list</strong>  <strong>Individual records</strong>  The default is All records.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From selection list</td>
<td>If you specify <code>From selection list</code>, a text box appears so you can enter the name of the selection list. If you specify <code>Individual records</code>, a list box appears so you can enter names of the desired records.</td>
</tr>
<tr>
<td>Type</td>
<td>The type of advanced record selection; can be any of the following:</td>
</tr>
<tr>
<td></td>
<td><code>where</code></td>
</tr>
<tr>
<td></td>
<td><code>and</code></td>
</tr>
<tr>
<td></td>
<td><code>or</code></td>
</tr>
<tr>
<td></td>
<td><code>and no</code></td>
</tr>
<tr>
<td></td>
<td><code>or no</code></td>
</tr>
<tr>
<td></td>
<td><code>and where</code></td>
</tr>
<tr>
<td></td>
<td><code>or where</code></td>
</tr>
<tr>
<td></td>
<td><code>and where no</code></td>
</tr>
<tr>
<td></td>
<td><code>or where no</code></td>
</tr>
<tr>
<td></td>
<td><code>and where (</code></td>
</tr>
<tr>
<td></td>
<td><code>or where (</code></td>
</tr>
<tr>
<td></td>
<td><code>and where ( no</code></td>
</tr>
<tr>
<td></td>
<td><code>or where ( no</code></td>
</tr>
<tr>
<td></td>
<td><code>) and where</code></td>
</tr>
<tr>
<td></td>
<td><code>) or where</code></td>
</tr>
<tr>
<td></td>
<td><code>) and where no</code></td>
</tr>
<tr>
<td></td>
<td><code>) or where no</code></td>
</tr>
</tbody>
</table>
Report Display Tab

The Report Display tab allows you to select a report type and a source file, and to select fields for sorting. This tab is available for all report types.

Properties: (continued)

) and where (  
) or where (  
) and where ( no  
) or where ( no  

There is no default.

This property is enabled only when the Advanced condition type property is selected.

The Report Display tab allows you to select a report type and a source file, and to select fields for sorting. This tab is available for all report types.
**Title/Prompts Tab**

The Title/Prompts tab allows you to specify the report title, to display instructions to persons running the report, and to request information before the report is run. This tab is available for all report types.

**Properties:**

- **File**: The source file from which to obtain information for the report. Click on the drop-down arrow to choose a file from the drop-down list box.

- **Last modified**: The time, date, and user ID of the last modification to this file.

- **Report type**: The report type of the report; can be one of the following:
  - Columnar.
  - Form.
  - Freeform.
  - Label.
  - Selection list.
  
  The default is columnar.

- **Sort by**: Specifies how the records are to be sorted.
  
  - **By field**: The name of the field by which to sort.
  - **Reverse**: When checked, the corresponding field is sorted in descending order. The default is cleared (ascending order). This property is optional.
**Properties:**

- **Default value**
  The initial value for the prompt. If a value is entered, the user can accept it or override it with a different value. You can use backslash commands such as \DATE and \TIME as default values.

- **Instruction**
  The instructions for the user running the report.

- **Prompt**
  Text used to request information before the report is run.
  You can establish up to nine user input prompts. The information the user enters is stored in variables \ENTRY1 through \ENTRY9. These variables can be used anywhere in the report where backslash commands are accepted.

- **Report title**
  The title of the report. The default is the ad hoc report ID.

- **Type**
  The type of data expected when the prompt is answered; can be one of the following:
**Properties: (continued)**

<table>
<thead>
<tr>
<th>A</th>
<th>Alphanumeric.</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>Date.</td>
</tr>
<tr>
<td>N</td>
<td>Numeric.</td>
</tr>
<tr>
<td>T</td>
<td>Time.</td>
</tr>
</tbody>
</table>
You can run an ad hoc report by using any of the following methods:

- Clicking the Run Ad Hoc Report button on the Ad Hoc Report tab of the Reports dialog box. To make this button active, you must first select the report you would like to run from the list box.

- Clicking the Run Ad Hoc Report command from the Ad Hoc submenu on the Reports menu. To make this command active, you must first select the report you would like to run from the list box on the Ad Hoc Report tab of the Reports dialog box.

- Entering an XQUERY command in ACL or a task.

- Running a batch of reports together using the Batch Report tab of the Reports dialog box. For more information on batch reporting, see the section Batch Reports.

Regardless of the method chosen, ad hoc reports can be sent to a printer or to a terminal screen. In addition, columnar and label reports can be output to a workstation file by specifying the filename on the Output Device Tab.

When you use the Run Ad Hoc Report command, either by clicking on the button on the Ad Hoc Report tab, or by clicking the command
from the Ad Hoc submenu, the Run Report dialog box appears as shown below.

Properties:

Client file  The file path and type of application to send the report to. Values are displayed only if they were specified on the Output Device Tab. The properties are display only, unless the Prompt when run box on the Output Device Tab is checked. The following properties are displayed:

Path name  The path, including filename, to which the report is sent.

Application  The path name for the application to be launched.
Properties: (continued)

**Output**
- Send output to client file: When selected, specifies that the report will be sent to the specified application. This property can only be selected if a path name and launch application have been specified on the Output Device tab.
- Send output to screen: When selected, specifies that the report will be sent to the screen.
  - This is the default setting.
- Send output to server printer: When selected, specifies that the report will be sent to the printer.

**Run sample**
- When checked, specifies to run a sample report using the specified parameters.
  - The default is cleared.

**Check**
- Specifies to check only a specified number or percentage of records. Enter a value in the corresponding text box. This property is enabled only if the Run sample property is checked.
  - % of records: Checks only the specified percent of records.
  - This is the default setting.
**Properties: (continued)**

- **Display maximum of … records**
  - The maximum number of records to display for the report.

- **User prompts**
  - The prompts to be answered for the report, and their corresponding default answers. This is display only. A value is displayed for this property only if at least one prompt was specified on the Titles/Prompts tab. Click on the Next button to view additional prompts.

- **records**
  - Checks only the specified number of records.
Listing Ad Hoc Reports

The List Ad Hoc Reports command from the Ad Hoc submenu of the Reports menu creates a list of current reports similar to that shown in Figure 7-2. This list can be viewed on the screen or printed.

Figure 7-2. Ad Hoc Report List
**Batch Reports**

A batch report is an ordered list of up to 30 individual ad hoc reports grouped under a single name. You can run all these reports at one time by using batch reporting, which is available from both the EDGE level and the project level. To open batch reporting, select the Reports area to view the Reports dialog box. In the Reports dialog box, click the Batch Report tab so it moves to the front as shown below.
The Batch Report tab allows you to create new batch reports, or to delete, modify, and run existing batch reports. When you click on the Add or Modify button, the Batch Report dialog box appears and is similar to the following:

Properties:

- **Ad Hoc Report**: The reports that are to be included in the batch. Click in each cell, then click on the pull-down arrow to select an ad hoc report from the pull-down list box.
- **Batch report name**: The name assigned to the batch of reports.
- **Report #**: The sequence number of the report in the batch. This is display only.
Once you have defined the batch report, you can run it by clicking the Run Batch Report button. When you click the Run Batch Report button, the Run Batch dialog box appears, and is similar to the following:

![Run Batch dialog box]

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Batch report name</td>
<td>The name assigned to the batch of reports. This is display only.</td>
</tr>
<tr>
<td>Options</td>
<td>If checked, specifies that prompts are displayed before running the batch report. The default is cleared.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH:MM</td>
<td>The time, in hours and minutes, at which the batch report should be run. This property is active only when the Specify time property is selected.</td>
</tr>
<tr>
<td>Run right now</td>
<td>Specifies that the batch report should be run immediately. This is the default setting.</td>
</tr>
<tr>
<td>Seconds</td>
<td>The number of seconds by which to delay running the batch report. This property is active only when the Seconds delay property is selected.</td>
</tr>
<tr>
<td>Seconds delay</td>
<td>If checked, specifies that the report should be run after a specified number of seconds.</td>
</tr>
<tr>
<td>Specify time</td>
<td>Specifies that the report should be run at a specific time.</td>
</tr>
<tr>
<td>Reports in batch</td>
<td>The reports included in the batch. This is display only.</td>
</tr>
</tbody>
</table>
Selection Lists

When you run a selection list, the list is created and saved in the system. You can use selection lists in many EDGE functions, such as Ad Hoc Reports, Correspondence, and ACL. In Ad Hoc Reports, storing commonly used selection criteria in selection lists saves you time when running reports. Use the Record Selection Tab to indicate the selection list to be used for a report when you are creating the report.

To view information about the selection lists stored on your system, select a project and then select the Reports area to view the Reports dialog box. In the Reports dialog box, click the Selection List tab to move it to the front. A page similar to the following is displayed:
The Selection List tab allows you to list the details of, as well as display the contents of, a specified selection list. You can also use this tab to delete a specified selection list. However, the Add and Modify commands are disabled, since selection lists are added, defined and modified from within the report or correspondence itself.

To list selection list detail, click on the desired selection list, then click on the List Detail button. You may print the list to your client screen, or to the printer server. If you print the list to your screen, a report similar to the following is displayed.
To display the content of a selection list, click the desired selection list, then click the Display Content button. A page similar to the following is displayed:
All the field information for the specified selection list is displayed on this page.
Overview

EDGE Query Language (EQL) is a powerful and flexible query language that allows you to manipulate data stored in EDGE files. EQL is a data retrieval and report generator language that specifies actions to be performed to create a report. An EQL statement can be created through Ad Hoc Reports or the AdvantEDGE Command Language (ACL).

This chapter describes the use of EQL through ACL commands.

You can use EQL to do the following:

- Produce custom reports from data files.
- Display any file or data element to the screen.
- Generate labels and forms.
**EQL Notation**

The following conventions are used in the EQL clause syntax in this chapter. They should not be typed as part of the EQL statement.

{} Indicates optional information.

| Indicates a choice between the items shown.

COUNT Words in uppercase are EQL reserved words.

*filename* Words in italics indicate values you supply.

The following conventions are also used in the EQL clause syntax in this chapter. When applicable, they must be typed as part of the EQL statement.

( ) Designates column and row positions for text or fields. For example, (1,3) specifies the first column and third row of the output.

[ ] Indicates text to appear in the report or indicates options. The brackets do not appear in the report. For example, a column heading for a list of companies could be specified as [Prospective Customers:]. Or, a group option specified as ['U'] suppresses underlining on the break line.
Types of Reports

The purpose of EQL is to manipulate, retrieve, and display formatted and unformatted data for reporting purposes. A report can be a printed columnar report, a listing displayed on a terminal, a count of records, or a label. The types of reports you can create with EQL are described below:

**Columnar Report**
The default report type. This report is frequently used with financial information and is characterized by columns and rows of data.

**Count Report**
Counts the number of records that meet the specified criteria.

**Dump Report**
Lists the field values of the requested records. The record ID is shown at the beginning of each record, followed by the data for each field.

**Form Report**
Generates information formatted to be printed on pre-existing forms.

**Noformat Report**
Frequently used to format data to download to another application such as a spreadsheet.

Columnar Report

A columnar report is characterized by columns and rows of data. It is the default type of report generated by EQL. The width of the output, the number of fields, field length, and printer and screen size determine whether the report is presented horizontally or vertically. A
sample EQL statement for a columnar report is shown below, followed by the report in Figure 8-1.

**Syntax**

```
SELECT FROM filename {fieldname_1 fieldname_2…} {EQL clauses}
```

- `filename` The name of the file from which to select.
- `fieldname_n` The names of the fields to select. If no field names are specified, only the record ID is included in the report.
- `EQL clauses` The appropriate EQL clauses, such as Selection or General Options. EQL clauses are described in the section EQL Clauses.

**Sample EQL Statement**

```
SELECT FROM CUST.LIST COMPANY CITY ST WHERE ST = “CA” OPTIONS I
```
**EQL Types of Reports**

Figure 8-1. **Columnar Report**

A count report returns the number of records that meet your selection criteria. A sample EQL statement for a count report is shown below, followed by the report in Figure 8-2.

**Syntax**

```
SELECT COUNT FROM filename {EQL clauses}
```

- `filename`: The name of the file from which to select.

---

**Count Report**

<table>
<thead>
<tr>
<th>COMPANY NAME</th>
<th>CITY</th>
<th>ST</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMITH BROTHERS</td>
<td>Long Beach</td>
<td>CA</td>
</tr>
<tr>
<td>HICKORY SAUSAGE</td>
<td>San Fransico</td>
<td>CA</td>
</tr>
<tr>
<td>JONES INC.</td>
<td>Mission Viejo</td>
<td>CA</td>
</tr>
<tr>
<td>AQUA CHEMICAL</td>
<td>Anywhere</td>
<td>CA</td>
</tr>
<tr>
<td>DC PLUMBING</td>
<td>Culver City</td>
<td>CA</td>
</tr>
<tr>
<td>COFFMAN SYSTEMS</td>
<td>Cerritos</td>
<td>CA</td>
</tr>
<tr>
<td>SMITH &amp; WESSON</td>
<td>Carmel</td>
<td>CA</td>
</tr>
<tr>
<td>ACME TRADING</td>
<td>Santa Fe Springs</td>
<td>CA</td>
</tr>
<tr>
<td>SMITH BROTHERS</td>
<td>Long Beach</td>
<td>CA</td>
</tr>
<tr>
<td>DAMAGED, INC.</td>
<td>Edge City</td>
<td>CA</td>
</tr>
<tr>
<td>1ST NATIONWIDE BANK</td>
<td>Brea</td>
<td>CA</td>
</tr>
<tr>
<td>SEARS</td>
<td>Santa Ana</td>
<td>CA</td>
</tr>
<tr>
<td>C &amp; S Percision</td>
<td>Duarte</td>
<td>CA</td>
</tr>
<tr>
<td>COFFMAN SYSTEMS, INC.</td>
<td>Cerritos</td>
<td>CA</td>
</tr>
<tr>
<td>AUTOMOBILE CLUB</td>
<td>Burbank</td>
<td>CA</td>
</tr>
<tr>
<td>ABC COMPANY</td>
<td>Lodi</td>
<td>CA</td>
</tr>
<tr>
<td>TESTING TEMPS</td>
<td>Irvine</td>
<td>CA</td>
</tr>
</tbody>
</table>

Press any key to continue...
Types of Reports

**EQL Types of Reports**

**Sample EQL Statement**

```plaintext
SELECT COUNT FROM CUST.LIST
```

![Figure 8-2. Count Report](image)

**Dump Report**

A Dump report shows the stored data for each field in the record. The record ID is the first value shown. Data for each field is displayed to the right of the line number. The line number corresponds to the field position in the file. A sample EQL statement for a dump report is shown below, followed by the report in Figure 8-3.

**Syntax**

```plaintext
SELECT DUMP FROM filename {EQL clauses}
```

- **filename** The name of the file from which to select.
- **EQL clauses** The appropriate EQL clauses, such as Selection. EQL clauses are described in the section EQL Clauses.
Sample EQL Statement

SELECT DUMP FROM CUST.LIST WHERE CITY=“LONG BEACH”

Figure 8-3.  Dump Report

Form Report

A Form report allows you to display data on preprinted or user-created forms. You can define the properties of the form using form options (FOPTIONS), which are described in the section Form Options Clause. A sample EQL statement for a form report is shown below, followed by the report in Figure 8-4.
Note: If you use the TO ASCIIFILE option in a form report, you should specify the terminal and printer page lengths. To do this, use the EQL general option S, which is described in Table 8-5.

Syntax

SELECT FROM filename FOPTIONS {fieldname_1} {fieldname_2…} {EQL clauses}

filename  The name of the file from which to select.
fieldname_n The names of the fields to select. At least one field name must be specified.
EQL clauses The appropriate EQL clauses, such as Selection or General Options. EQL clauses are described in the section EQL Clauses.

Sample EQL Statement

SELECT FROM CUST.LIST FOPTIONS FL[10], FPP[1] (1,1) [Supplier:] (12,1) COMPANY (30,1) [Contact:] (40,1) LAST.NAME (32,2) [Phone:] (40,2) PHONE (24,9) [Other Details:] (40,9) SAL (44,9) FIRST.NAME (51,9) MI OPTIONS I HEADING ['MSTCL’STANDARD SUPPLIERS LIST’] FOOTING [FILE: ‘FE’PAGE:’P’]
Label Report

Figure 8-4.   Form Report

A Label report allows you to print information in a label format. You can define the properties of the label using label options (LOPTIONS), which are described in the section Label Options Clause. A sample EQL statement for a label report is shown below, followed by the report in Figure 8-5.

Syntax

SELECT LABEL FROM filename {fieldname_1 fieldname_2…} LOPTIONS {EQL clauses}

filename       The name of the file from which to select.
**EQL Types of Reports**

*Sample EQL Statement*

```eql
SELECT LABEL FROM CUST.LIST ACCT.NO COMPANY LAST.NAME
OPTIONS C, I LOPTIONS B[2], C[25], M[5], P[2], R[5], S[5], Y
```

*Figure 8-5. Label Report*

```plaintext
COLUMBUS SERVICES
Saith

PASTA PRODUCTS
Guine

STEVEN & JIM'S BAR
Valiensi

C & S PERCISION
Seltzer

COFFMAN SYSTEMS, INC.
Helgren

MARY'S PLACE
Mahoney
```

*fieldname* The names of the fields to select. If no field names are specified, only the record ID is included in the report.

*EQL clauses* The appropriate EQL clauses, such as Selection. EQL clauses are described in the section EQL Clauses.
Noformat Report

A Noformat report is frequently used to format data to download to another application such as a spreadsheet. You can define the properties of the report using Noformat options (NFOPTIONS), which are described in Noformat Options Clause. The NOFORMAT command uses a tab as the default field separator. A sample noformat report and EQL statement are shown in Figure 8-6.

Syntax

SELECT NOFORMAT FROM filename {fieldname_1 fieldname_2…} NFOPTIONS {EQL clauses}

filename

The name of the file from which to select.

fieldname_n

The names of the fields to select. If no field names are specified, only the record ID is included in the report.

EQL clauses

The appropriate EQL clauses, such as Selection. EQL clauses are described in the section EQL Clauses.

Sample EQL Statement

SELECT NOFORMAT FROM CUST.LIST NFOPTIONS E[#, F[-], R[*], T

Figure 8-6. Noformat Report
EQL Identifiers

EQL identifiers include the names of selection lists, fields, files, and other references used in EQL. Identifiers must meet the following requirements. Examples of valid EQL identifiers are shown in Table 8-1; invalid identifiers are shown in Table 8-2.

- Cannot be the same as a reserved word. (See Table 8-3.)
- Must contain at least one alpha character.
- Cannot begin with a number.
- Cannot begin with a delimiter. (For example, -,.,:, )
- Cannot begin with a dollar sign ($).

Table 8-1. Valid EQL Identifiers

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Reason</th>
</tr>
</thead>
<tbody>
<tr>
<td>$AB_D:1F</td>
<td>Begins with a dollar sign, has at least one alpha character, and the delimiter is not at the beginning.</td>
</tr>
<tr>
<td>QEF@ABC</td>
<td>Contains at least one alpha character.</td>
</tr>
<tr>
<td>MYSAMPLE</td>
<td>Different from the EQL reserved word SAMPLE.</td>
</tr>
<tr>
<td>ABC:123</td>
<td>Contains at least one alpha character and the delimiter is not at the beginning.</td>
</tr>
<tr>
<td>Identifier</td>
<td>Reason</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>$124_3</td>
<td>Does not include an alpha character.</td>
</tr>
<tr>
<td>12ABC</td>
<td>Begins with a number.</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>Same as an EQL reserved word.</td>
</tr>
<tr>
<td>-XYZ</td>
<td>Begins with a delimiter.</td>
</tr>
</tbody>
</table>
EQL Reserved Words

EQL reserved words are words that cannot be used as identifiers. In default condition, they can be uppercase or lowercase. If you set the environment variable EQL_CASE_SENSITIVE to a non-zero value, then the EQL reserved words must be in uppercase. For a description of environment variables, see Appendix AB.

EQL reserved words are described in Table 8-3. The clauses to use them are described in the EQL Clauses.

Table 8-3. EQL Reserved Words

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A heading and footing option that displays the account number.</td>
</tr>
<tr>
<td>AND</td>
<td>A logical operator that joins Selection clauses. Used optionally with WHERE.</td>
</tr>
<tr>
<td>ASCIIFILE</td>
<td>Used in a Target clause with an absolute or relative UNIX pathname to store a report.</td>
</tr>
<tr>
<td>AVERAGE</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>B</td>
<td>An option, whose meaning depends on the use. If used in a group, heading, or footing clause, displays a break value.</td>
</tr>
</tbody>
</table>
### Table 8-3. EQL Reserved Words (continued)

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BY</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>A relational operator used to locate values within a range in the set denoted by &lt; &gt;.</td>
</tr>
<tr>
<td>C</td>
<td>A general option that suppresses default column headings.</td>
</tr>
<tr>
<td></td>
<td>A group option that counts the number of records in the group.</td>
</tr>
<tr>
<td></td>
<td>A heading and footing option that centers data on the line.</td>
</tr>
<tr>
<td></td>
<td>A label option used with a variable to specify the width of the label.</td>
</tr>
<tr>
<td>COUNT</td>
<td>A report type that displays the number of records satisfied by a query.</td>
</tr>
<tr>
<td>D</td>
<td>A group and grand total option that prevents a break line on a single detail line.</td>
</tr>
<tr>
<td></td>
<td>A heading and footing option that displays the current date.</td>
</tr>
<tr>
<td>Reserved Word</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>DATA</td>
<td>The default output for a Source clause signifying the data portion of the file.</td>
</tr>
<tr>
<td>DESC</td>
<td>A Sort clause option that sorts the records in descending order.</td>
</tr>
<tr>
<td>DET-SUP</td>
<td>A group and grand total option that prevents a break line on a single detail line and inserts a page break after each group.</td>
</tr>
<tr>
<td>DICT</td>
<td>The output for a Source clause signifying the dictionary information from the file rather than the data.</td>
</tr>
<tr>
<td>DISPLAY</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>DLIST</td>
<td>A system list assigned to each user at login. Data copied to DLIST is stored until the user logs off or overwrites it with other data. Data on DLIST can be retained from one project to another and from system level to project level.</td>
</tr>
<tr>
<td>DSPLST</td>
<td>A general option that generates an output report in an EQL statement that includes a TO LIST or TO DLIST clause.</td>
</tr>
</tbody>
</table>
Table 8-3.  **EQL Reserved Words (continued)**

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUMP</td>
<td>A type of report displayed on the terminal that shows the data or dictionary components of records in a file.</td>
</tr>
<tr>
<td>E</td>
<td>A heading and footing option that right-justifies data on the line. A noformat option to specify the character for the end of report separator.</td>
</tr>
<tr>
<td>EQ</td>
<td>A relational operator meaning &quot;equal to&quot;. Used in the Selection clause. A synonym is &quot;=&quot;.</td>
</tr>
<tr>
<td>F</td>
<td>A heading and footing option that displays the file name, optionally left-justified n spaces. A noformat option to specify the character for the field separator.</td>
</tr>
<tr>
<td>FIELD</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>FL</td>
<td>A form option used with a variable to specify the number of lines per form.</td>
</tr>
<tr>
<td>FOOTING</td>
<td>Text that appears at the bottom of the report. The default footing for reports comes from the information identified in the Site Configuration screen. Use footing options to create a new footing.</td>
</tr>
</tbody>
</table>
### Table 8-3.  EQL Reserved Words (continued)

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FOPTIONS</td>
<td>The special options available with form reports. See FL and FPP.</td>
</tr>
<tr>
<td>FORMAT</td>
<td>The options used to define a label report. The reserved words LOPTIONS and FORMAT are synonyms and can be used interchangeably.</td>
</tr>
<tr>
<td>FPP</td>
<td>A form option used with a variable to specify the number of forms per page.</td>
</tr>
<tr>
<td>FROM</td>
<td>Used in the Source clause to indicate that DATA or DICT information is being retrieved from a file.</td>
</tr>
<tr>
<td>GE</td>
<td>A relational operator meaning &quot;greater than or equal to.&quot; Used in a Selection clause. A synonym is &quot;&gt;=&quot;.</td>
</tr>
<tr>
<td>GRAND-TOTAL</td>
<td>Used to specify a grand total line as the last record on the report. You can add text to the line or accept the default of multiple asterisks.</td>
</tr>
<tr>
<td>GROUP</td>
<td>Used on sorted records to create a group on which to break. By grouping on sorted data, a break line and an optional subtotal are shown for the field.</td>
</tr>
<tr>
<td>GT</td>
<td>A relational operator meaning &quot;greater than&quot;. Used in a Selection clause. A synonym is &quot;&gt;&quot;.</td>
</tr>
<tr>
<td>H</td>
<td>A general option that suppresses the default report heading.</td>
</tr>
<tr>
<td>Reserved Word</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>HEADING</td>
<td>Text that appears at the top of the report. The default heading for reports comes from the information identified in the Site Configuration screen. Use heading options to create a new heading.</td>
</tr>
<tr>
<td>I</td>
<td>A general option that suppresses the first column (by default the record ID). A synonym is ID-SUP.</td>
</tr>
<tr>
<td>ID-SUP</td>
<td>A general option that suppresses the first column (by default the record ID). A synonym is I.</td>
</tr>
<tr>
<td>IN</td>
<td>A special relational field used in Selection clauses to specify membership in a set of values denoted by &lt; &gt;.</td>
</tr>
<tr>
<td>INTO</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>KEY</td>
<td>The record ID. Unless specified otherwise, the first column on a report is a listing of KEYS. If used in the Output clause, the KEY appears as a column with the file name as the column heading. Use OPTIONS I to suppress the record ID column as the first column of the report.</td>
</tr>
<tr>
<td>L</td>
<td>A group option that suppresses a blank line before a break.</td>
</tr>
<tr>
<td></td>
<td>A grand total option that prints a blank line before the grand total.</td>
</tr>
<tr>
<td></td>
<td>A heading and footing option that specifies a new line.</td>
</tr>
<tr>
<td>Reserved Word</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>LABEL</td>
<td>A report type formatted according to LOPTIONS and represented by the keyword LABEL in the query.</td>
</tr>
<tr>
<td>LE</td>
<td>A relational operator meaning &quot;less than or equal to&quot;. Used in a Selection clause. A synonym is &quot;(\leq)&quot;.</td>
</tr>
<tr>
<td>LIST</td>
<td>A selection list designated in either a Source clause or Target clause.</td>
</tr>
<tr>
<td>LOPTIONS</td>
<td>The options used to define a label report. The reserved words LOPTIONS and FORMAT are synonyms and can be used interchangeably.</td>
</tr>
<tr>
<td>LP</td>
<td>A Target clause option to direct output to a line printer.</td>
</tr>
<tr>
<td>LT</td>
<td>A relational operator meaning &quot;less than&quot;. Used in a Selection clause. A synonym is &quot;(&lt;)&quot;.</td>
</tr>
<tr>
<td>M</td>
<td>A heading and footing option that centers text on the line.</td>
</tr>
<tr>
<td>N</td>
<td>A label option used with a variable to specify the column number of the left margin.</td>
</tr>
<tr>
<td>NE</td>
<td>A relational operator meaning &quot;not equal to&quot;. Used in a Selection clause. Synonyms are NOT and #.</td>
</tr>
<tr>
<td>NFOPTIONS</td>
<td>Report options used with the Noformat report type.</td>
</tr>
</tbody>
</table>
### Table 8-3.  EQL Reserved Words (continued)

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NI</td>
<td>A general option that formats the output for a workstation report.</td>
</tr>
<tr>
<td>NOFORMAT</td>
<td>A report type formatted according to NFOPTIONS and represented by the keyword NOFORMAT in the query.</td>
</tr>
<tr>
<td>NOPAGE</td>
<td>A general option that suppresses page scrolling on the terminal. A synonym is N.</td>
</tr>
<tr>
<td>NOT</td>
<td>A relational operator meaning &quot;not equal to&quot;. Used in a Selection clause. Synonyms are NE and #.</td>
</tr>
<tr>
<td>NOTIN</td>
<td>A relational operator used to compare a value to values in a set denoted by &lt; &gt;. Synonyms are NE, NOT, and #.</td>
</tr>
<tr>
<td>OF</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>ON</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>The General Options clause.</td>
</tr>
<tr>
<td>OR</td>
<td>A logical operator used to combine two Selection clauses.</td>
</tr>
<tr>
<td>ORDER</td>
<td>Synonym of SORT.</td>
</tr>
<tr>
<td>P</td>
<td>A group option that inserts a page break after a break.</td>
</tr>
<tr>
<td></td>
<td>A grand total option that inserts a page break after the grand total.</td>
</tr>
</tbody>
</table>
**Table 8-3.  EQL Reserved Words (continued)**

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A heading and footing option used to display the current page number, optionally left-justified n spaces. A label option used with a variable to specify the number of labels across the page.</td>
<td>A group option that suppresses paging until the break for detail lines occurs.</td>
</tr>
<tr>
<td>R</td>
<td>A general option that displays the number of records that validate the query. A label option used with a variable to specify the number of rows on a label. A noformat option to specify the character for the record separator.</td>
</tr>
<tr>
<td>RSAMPLE</td>
<td>A general option used with a variable to specify the number or percentage of records to test.</td>
</tr>
<tr>
<td>S</td>
<td>A heading and footing option that displays the site number. A general option used with variables to specify the terminal width and length, and printer width and length. A label option used with a variable to specify the horizontal spacing between labels.</td>
</tr>
<tr>
<td>Reserved Word</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------</td>
</tr>
<tr>
<td>SAMPLE</td>
<td>A general option used with a variable to limit the output of a query to the number of records specified.</td>
</tr>
<tr>
<td>SELECT</td>
<td>Begins every EQL sentence.</td>
</tr>
<tr>
<td>SORT</td>
<td>Used to sort or order the selected records. Synonym of ORDER.</td>
</tr>
<tr>
<td>SUM</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>T</td>
<td>A heading and footing option that displays the time and date.</td>
</tr>
<tr>
<td></td>
<td>A general option that displays the time it took to run the query.</td>
</tr>
<tr>
<td></td>
<td>A noformat option that displays a record type code in column one.</td>
</tr>
<tr>
<td>THE</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>TO</td>
<td>Used to identify the Target clause in the query.</td>
</tr>
<tr>
<td>TOTAL</td>
<td>A reserved word that is not a command but can be used in the EQL statement for clarity.</td>
</tr>
<tr>
<td>U</td>
<td>A group and grand total option that inserts a double line before the break or grand total value.</td>
</tr>
</tbody>
</table>
### Table 8-3. EQL Reserved Words (continued)

<table>
<thead>
<tr>
<th>Reserved Word</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIQUE</td>
<td>A heading and footing option that displays the user name.</td>
</tr>
<tr>
<td>UNIQUE</td>
<td>Used with SORT BY KEY to eliminate duplicate keys in the report.</td>
</tr>
<tr>
<td>USING</td>
<td>Used in a Source clause to access data from a LIST or DLIST.</td>
</tr>
<tr>
<td>V</td>
<td>A group option that displays the current break value.</td>
</tr>
<tr>
<td>WHERE</td>
<td>Used in the Selection clause to begin the comparison process.</td>
</tr>
<tr>
<td>Y</td>
<td>A label option that vertically compresses the data by removing blank lines from the label.</td>
</tr>
</tbody>
</table>
Every EQL statement begins with a Source clause and is completed by other required and optional clauses in any order. The clauses are listed below, with the Source clause listed first and the other clauses in alphabetical order. Each clause is described in detail in the following sections:

- Source Clause.
- Form Options Clause.
- General Options Clause.
- Group Clause.
- Headings and Footings Clause.
- Label Options Clause.
- Noformat Options Clause.
- Output Clause.
- Selection Clause.
- Sort Clause.
- Target Clause.
- Total Clause.
**Source Clause**

Every EQL statement begins with a Source clause. This clause specifies the report type and the file or selection list to use as the source for the report data. If your report type is other than columnar, the appropriate report type keyword must follow the SELECT keyword. If no report type keyword follows SELECT, a columnar report is presented.

For files, this clause consists of the keyword FROM, followed by the file name. If you want to select the dictionary part of the file, include the keyword DICT before the file name. The dictionary describes field attributes such as field position and display format. If you do not specify DICT, the DATA part of the file is the default. The DATA portion is the actual field value.

You can specify a selection list as part of the Source clause by including the keyword USING, followed by either the keyword LIST and the selection list name or the keyword DLIST. DLIST is a temporary list assigned to each user by the system. DLIST is described in the section **Target Clause**.
**Syntax**

```
SELECT {report_type} FROM {DICT} filename {USING LIST
selection_list_1, selection_list_2,...} {USING DLIST}
```

- **report_type**  The type of report to create; can be one of the following:
  - COLUMNAR
  - COUNT
  - DUMP
  - FORM
  - LABEL
  - NOFORMAT

- **filename**  The name of the file from which to select.

- **selection_list_n**  The names of the selection lists to use.
  When using more than one selection list, you can eliminate duplicate records by doing a SORT BY on the KEY and using the UNIQUE option. See the example below.

**Examples**

- SELECT LABEL FROM CUST.FILE
- SELECT NOFORMAT FROM DICT CUST.FILE
- SELECT FROM CUST.FILE USING DLIST
- SELECT FROM CUST.FILE USING LIST SELST, PROD.LIST SORT BY KEY
**Form Options Clause**

For form reports, this optional clause allows you to specify the number of lines per form and the number of forms per page. Other EQL clauses can also be used in form reports. For example, the Output clause with row and column numbers for each field can be used to design a form. Form options are described in Table 8-4.

**Syntax**

FOPTIONS FL \[n\], FPP \[n\]

**Examples**

FOPTIONS FL [12], FPP [1]

<table>
<thead>
<tr>
<th>Table 8-4. Form Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option</td>
</tr>
<tr>
<td>FL[n]</td>
</tr>
<tr>
<td>FPP[n]</td>
</tr>
</tbody>
</table>

**General Options Clause**

This optional clause allows you to specify format and output options for the report. The General Options clause can be used with other EQL clauses, including the Heading and Footing clause and the grand total option. General options are listed in Table 8-5.
**Syntax**

OPTIONS `general_option_1, {general_option_2,...}`

The General options are listed in Table 8-5.

**Examples**

OPTIONS 3, C, I,

OPTIONS RSAMPLE[10], S[20,10,80,30]

**Table 8-5. General Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td># (integer)</td>
<td>Number of blank lines between records on the report. Default = 0 (single spacing) Specify as 1 for double spacing, 2 for triple spacing, etc.</td>
</tr>
<tr>
<td>C</td>
<td>Suppresses the default column headings.</td>
</tr>
<tr>
<td>D or DET-SUP</td>
<td>Suppresses detail data used by the Total and Group clauses. Only subtotals and totals are displayed.</td>
</tr>
<tr>
<td>DSPLST</td>
<td>Generates an output report in an EQL statement that includes a TO LIST or TO DLIST clause.</td>
</tr>
<tr>
<td>H</td>
<td>Suppresses the default report heading.</td>
</tr>
</tbody>
</table>

*Note*: If you abort the report display while the report is being generated you will also abort the creation of the selection list.
### Table 8-5. General Options (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I or ID-SUP</td>
<td>Suppresses the first column (by default the record ID).</td>
</tr>
<tr>
<td>N or NOPAGE</td>
<td>Suppresses page scrolling on the terminal.</td>
</tr>
<tr>
<td>NI</td>
<td>Formats the output for a workstation report.</td>
</tr>
<tr>
<td>R</td>
<td>Displays the number of records that validate the query.</td>
</tr>
<tr>
<td>RSAMPLEPLE[n]</td>
<td>Number of records to test or percentage of records to test. When this many records have been tested against your query, your query is terminated. This option limits the scope of your query. See also the SAMPLE[n] option below.</td>
</tr>
<tr>
<td>S[#,#,#,#]</td>
<td>Specifies the terminal width, terminal length, printer width and printer length. The defaults are 80, 24, 132, and 60.</td>
</tr>
<tr>
<td>SAMPLE[n]</td>
<td>Number of records that satisfy your query. When this number is reached, no more records are tested. This option limits the output of your query to the number of records you specify. See also the RSAMPLE[n] option above.</td>
</tr>
<tr>
<td>T</td>
<td>The time it took to run the query. Time is displayed in three ways:</td>
</tr>
<tr>
<td></td>
<td>System time: System resource time.</td>
</tr>
<tr>
<td></td>
<td>CPU time: CPU time to execute user functions.</td>
</tr>
<tr>
<td></td>
<td>Real time: Difference between the time the query began and the time it completed.</td>
</tr>
</tbody>
</table>
**Group Clause**

This optional clause allows you to group or arrange sorted data for output. If the Total clause is used in the same report, a subtotal of the data is produced. The Group clause does not sort the data; it groups data according to the KEY or the fields you specify and produces a break line between each group. You can also specify the row and column coordinates for a form report, text to be printed on the subtotal line, and options to control the break line. Group options are described in Table 8-6.

---

**Note:** If you use the Group clause, include a Sort clause or use an existing sorted list so the grouped data is in a logical sorted order. Although it is not required, you get the best results if you sort by the same fields and in the same order by which you group.

---

**Syntax**

GROUP BY \{(column, row)\} KEY | fieldname_1 {fieldname_2,...} {\[text:'group options']}

- **column**
  - The column number on the form report at which to position the first character of the field or text.

- **row**
  - The row number on the form report at which to position the first character of the field or text.

- **fieldname_n**
  - The fields by which you want to group. A break line is displayed between each group.
**text** The text to display on the group break line.

**Examples**

GROUP BY DATA ['L'] SORT BY DATE
SORT BY TERR [DESC], COMPANY GROUP BY TERR [TERRITORY:'V']

**Group Options**

The group options listed in Table 8-6 allow you to further define the display of the group. The options must be inside square brackets and enclosed within single quotation marks. Within the brackets, you can group the options within single quotation marks or put single quotation marks around each option, for example: ['BLP'] or ['B”L”P'].

**Table 8-6. Group Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>['B']</td>
<td>Allows a break value in the heading. Use with the heading option B.</td>
</tr>
<tr>
<td>['C']</td>
<td>Displays in the break line the number of records that are part of the current group.</td>
</tr>
<tr>
<td>['D']</td>
<td>Prevents a break on a single detail line.</td>
</tr>
<tr>
<td>['L']</td>
<td>Suppresses a blank line before a break.</td>
</tr>
</tbody>
</table>
Table 8-6. Group Options (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[’P’]</td>
<td>Inserts a page break after a break.</td>
</tr>
<tr>
<td>[’R’]</td>
<td>Suppresses paging until the break for detail lines occurs.</td>
</tr>
<tr>
<td>[’U’]</td>
<td>Inserts a double line before the break value.</td>
</tr>
<tr>
<td>[’V’]</td>
<td>Displays the current break value.</td>
</tr>
</tbody>
</table>

This optional clause allows you to identify text to be used in the heading at the top of the page and in the footing at the bottom of the page. If you do not specify a heading or footing, the system uses the information defined on the Site Configuration in the section Set Up Site Information.

Within the brackets shown in the syntax below, you can group the options within single quotation marks or put single quotation marks around each option, for example: [’CLS’] or [’C‘L‘S’].

Syntax

HEADING [ ‘heading_options’ | text | text’heading_options’ | ‘heading_options’text ]
FOOTING [text'footing_options' | 'footing_options'text | 'footing_options'|text]

  text     The text that to display in the heading or footing of the report.

  options  The heading and footing options are listed in Table 8-7.

**Examples**

FOOTING [FILE: 'FE'PAGE:'P']

  HEADING ['MSLTC']

FOOTING [SITE:'S'USER:'U'TIME/DATE:'T']

  HEADING [Company Contacts for 'DC']

**Table 8-7. Heading and Footing Options**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>The port number.</td>
</tr>
<tr>
<td>A</td>
<td>The account number.</td>
</tr>
<tr>
<td>B</td>
<td>The break number. Use with the group option B.</td>
</tr>
<tr>
<td>C</td>
<td>Centers data on the line. Generally used after the data it affects.</td>
</tr>
<tr>
<td>D</td>
<td>The current date.</td>
</tr>
</tbody>
</table>
Label Options Clause

This optional clause allows you to define properties for label-type reports.

Syntax

FORMAT | LOPTIONS label_option_1,..., label_option_6

Label options are listed in Table 8-8.

Table 8-7. Heading and Footing Options (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Right justifies data following this option on the line.</td>
</tr>
<tr>
<td>Fn</td>
<td>The file name optionally followed by n spaces.</td>
</tr>
<tr>
<td>L</td>
<td>Specifies a new line in the heading or footing.</td>
</tr>
<tr>
<td>M</td>
<td>Centers text on the line. Must appear before the data it affects.</td>
</tr>
<tr>
<td>Pn</td>
<td>The current page number optionally left-justified n spaces.</td>
</tr>
<tr>
<td>S</td>
<td>The site name.</td>
</tr>
<tr>
<td>T</td>
<td>The time and date.</td>
</tr>
<tr>
<td>U</td>
<td>The user name.</td>
</tr>
</tbody>
</table>
Notes

You must specify all label options except Y, which is optional.

The reserved words FORMAT and LOPTIONS are synonyms and can be used interchangeably.

Examples

```
SELECT LABEL FROM CUST.LIST ACCT.NO COMPANY PHONE OPTIONS I
LOPTIONS B[2], C[25], M[0], P[2], R[5], S[5], Y
```

Table 8-8. Label Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>B[#]</td>
<td>The vertical spacing between labels.</td>
</tr>
<tr>
<td>C[#]</td>
<td>The width of the label in number of characters.</td>
</tr>
<tr>
<td>M[#]</td>
<td>The column number of the left margin.</td>
</tr>
<tr>
<td>P[#]</td>
<td>The number of labels across the page.</td>
</tr>
<tr>
<td>R[#]</td>
<td>The number of rows on a label.</td>
</tr>
<tr>
<td>S[#]</td>
<td>The horizontal spacing between labels.</td>
</tr>
<tr>
<td>{Y}</td>
<td>Vertically compresses the data by removing blank lines.</td>
</tr>
</tbody>
</table>

Noformat Options Clause

This optional clause is used in noformat reports to specify field and record separators and to display the record type.
Note: If you use the NFOPTIONS reserved word, you must specify at least one of the noformat options listed in Table 8-9.

Syntax

NFOPTIONS noformat_opt1, {noformat_opt2,...}

Noformat options are listed in Table 8-9.

Examples

NFOPTIONS E[#, F[-], R[*], T
NFOPTIONS T

Table 8-9. Noformat Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E[char]</td>
<td>The end of report separator.</td>
</tr>
<tr>
<td>F[char]</td>
<td>The field separator.</td>
</tr>
<tr>
<td>R[char]</td>
<td>The record separator.</td>
</tr>
<tr>
<td>T</td>
<td>Each record or line begins with a type code in column one. The codes are listed below:</td>
</tr>
</tbody>
</table>
Output Clause

This optional clause defines the names, order, and positions of fields and text to be included in the report. The fields appear in the order entered in the clause. Fields designated using the Group clause or grand total option are also displayed in the report.

The column headings for the fields are from the field definition specified when the field was created. If no column heading was defined when the field was created, the field name is displayed.

You can specify row and column placement in a form report for each field or for text.

When no fields are specified, the only output is a listing of record IDs that satisfy the query.

---

Table 8-9. Noformat Options (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>number</td>
<td>Break line</td>
</tr>
<tr>
<td>G</td>
<td>Total line</td>
</tr>
<tr>
<td>R</td>
<td>Record</td>
</tr>
<tr>
<td>T</td>
<td>Title or heading</td>
</tr>
</tbody>
</table>
**Syntax**

\{(\texttt{column, row}) [\texttt{text}]\} | \{(\texttt{column, row})\} KEY | \texttt{fieldname}_1, \{\texttt{fieldname}_2...\}

- \texttt{column} The column number on the form report at which to position the first character of the field or text.
- \texttt{row} The row number on the form report at which to position the first character of the field or text.
- \texttt{text} Text that you want to appear at the specified location on the form report.
- \texttt{fieldname}_n The fields that you want to include in the report, or the fields that you want to display in the form report at the specified column and row settings.

**Examples**

(5,5) [Contact:] (5,15) LAST.NAME

FIRST.NAME, LAST.NAME, CITY, STATE

**Selection Clause**

This optional clause specifies record selection criteria, using logical or relational operators. This clause begins with the keyword WHERE, followed by the selection expression. Multiple selection clauses separated by AND or OR can be used in one EQL statement.
Note: When using relational operators, the two items being compared should be of the same data type (alphanumeric, numeric, date, or time). Unexpected results could occur if the compared items are not the same data type.

When a field is compared to a string, the data type for the comparison is determined by the justification of the field and the contents of the string.

- If the field is left-justified, the data type of the comparison is alphanumeric, regardless of the contents of the string.
- If the field is right-justified, the data type of the comparison is determined by the contents of the string. If the string contains only digits, the comparison is numeric; otherwise, the comparison is alphanumeric.

Syntax

WHERE fieldname_1 relational_operator_1 {"string_1" | fieldname_2 | selection_list_1} {AND | OR WHERE. fieldname_3 relational_operator_2 {"string_2" | fieldname_4 | selection_list_2}

fieldname_n The name of the field you are comparing for selection.
Examples

- A comparison string that contains quotation marks must be enclosed in single quotation marks:

  `WHERE NAME = 'The "Blue Bird"'`

- A comparison that applies to more than one field must be repeated:

  `WHERE AGE.1 IN <"12","16","18"> OR AGE.2 IN <"12","16","18">`

- A comparison between two fields can be made using the operators `=`, `#`, `<`, `>`, `<=`, and `>=`. The two fields being compared should be of the same type or unexpected results could occur.
WHERE AGE.1 < AGE.2

- A comparison between a field and a selection list can be made using the operator IN:
  WHERE AGE.1 IN LIST SEL.LIST

- The following two comparisons are equivalent:
  WHERE AGE IN <"12","16","18">
  WHERE AGE = "12" OR AGE = "16" OR AGE = "18"

- The following two comparisons are equivalent:
  WHERE AGE BETWEEN <"11","19">
  WHERE AGE >= "12" AND AGE <= "18"

- The following three comparisons are equivalent:
  WHERE AGE NOTIN <"12","16","18">
  WHERE AGE # <"12","16","18">
  WHERE AGE NE "12" AND AGE NE "16" AND AGE NE "18"

### Table 8-10. Relational Operators

<table>
<thead>
<tr>
<th>Relational Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>Not equal to. Synonyms are NE and NOT.</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than. Synonym is LT.</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to. Synonym is LE.</td>
</tr>
</tbody>
</table>
This optional clause allows you to indicate the fields by which to sort. The first field indicated is the primary sort key, the second field indi-

<table>
<thead>
<tr>
<th>Relational Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equal to. Synonym is EQ.</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than. Synonym is GT.</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to. Synonym is GE.</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>Member of a range of values in the set denoted by &lt; &gt;, excluding the named values.</td>
</tr>
<tr>
<td>EQ</td>
<td>Equal to. Synonym is = .</td>
</tr>
<tr>
<td>GE</td>
<td>Greater than or equal to. Synonym is &gt;=.</td>
</tr>
<tr>
<td>GT</td>
<td>Greater than. Synonym is &gt;.</td>
</tr>
<tr>
<td>IN</td>
<td>Member of the set denoted by &lt; &gt;. Synonyms are = and EQ. Can also be used with a selection list name to indicate membership.</td>
</tr>
<tr>
<td>LE</td>
<td>Less than or equal to. Synonym is &lt;=.</td>
</tr>
<tr>
<td>LT</td>
<td>Less than. Synonym is &lt;.</td>
</tr>
<tr>
<td>NE</td>
<td>Not equal to. Synonyms are # and NOT.</td>
</tr>
<tr>
<td>NOT</td>
<td>Not equal to. Synonyms are # and NE.</td>
</tr>
<tr>
<td>NOTIN</td>
<td>Not member of the set denoted by &lt; &gt;. Synonyms are #, NE, and NOT.</td>
</tr>
</tbody>
</table>
cated is the secondary sort key, and so on. If no Sort clause is specified, the records are sorted in ascending order based on the record ID. To order the output in descending order, include [DESC] after the sort field.

For numeric fields, a numeric sort is performed. For alphanumeric fields, the sort is based on the ASCII collating sequence for each character. For right-justified alphanumeric fields, leading spaces are added to the field before the sort is performed.

Note: If you use a Group clause, you get the best results if you sort by the same fields and in the same order by which you group.

**Syntax**

```
{SORT BY | ORDER BY} {KEY | fieldname_1} {,fieldname_2...} [[DESC]]
```

fieldname_n  The name of the field by which you want to sort.

**Notes**

SORT BY and ORDER BY are synonyms and can be used interchangeably.
You can use the SORT BY clause to order a selection list by the sorted field instead of by record key. For more information, see the example below and the section Target Clause.

**Examples**

```
SORT BY PRICE, QUANTITY
SORT BY ORG.DATE [DESC]
SORT BY STATE TO LIST SEL.LIST FIELD STATE
ORDER BY TERR [DESC]
```

The following two clauses are equivalent:

```
ORDER BY TERR, COMPANY
ORDER BY TERR ORDER BY COMPANY
```

**Target Clause**

The Target clause specifies where to direct the output of the report. If no output is specified, the default is the screen. With the Target clause, you can specify the output device for the report as the printer, an ASCII file, a selection list, or the DLIST. Target clause options are described in Table 8-11.

You can use an optional phrase, FIELD `fieldname`, following the selection list name. This saves the field to the selection list instead of the record keys.
Using a selection list as the target suppresses the output report. If you want to generate an output report and a selection list, use the general option DSPLST.

You can specify more than one selection list as the target by using multiple TO LIST clauses. If you use a SORT BY clause, all selection lists are affected.

**Syntax**

TO ASCIIFILE [UNIX_pathname]

TO DLIST

TO LIST listname

TO LIST listname_1 {FIELD fieldname} {TO LIST listname_2}

TO LP

UNIX_pathname The path name for the UNIX file to which you want to write the selected data.

listname_n The name of the selection list.

fieldname The name of the field to write to the selection list. If no fieldname is specified, the record ID is written.
Examples

SELECT FROM STATUS WHERE DATE >= "05/01/97" TO DLIST
SELECT FROM CUST.FILE WHERE ST NE "CA" TO LIST ST.LST FIELD ST
SELECT FROM CUST.FILE SORT BY STATE TO LIST SEL.LIST FIELD STATE
TO LIST SEL.LIST2
SELECT FROM XYZ TO ASCIIFILE [/usr/tmp/xyz]

Table 8-11. Target Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TO ASCIIFILE [UNIX path name]</td>
<td>The UNIX path can be either absolute or relative. In either case, you must have access to the UNIX path. If the file already exists, EQL overwrites it. If the file does not already exist, EQL creates it. If you use the TO ASCIIFILE option in a form report, you should specify the terminal and printer page lengths. To do this, use the general option S, which is described in Table 8-5.</td>
</tr>
<tr>
<td>TO DLIST</td>
<td>DLIST is space assigned to you by the system each time you log on. When you use the TO DLIST target option, a list of record IDs or fields that satisfy the query is written to DLIST. The values in the DLIST are saved until you log off the system or you overwrite the existing DLIST.</td>
</tr>
</tbody>
</table>
**Total Clause**

This optional clause allows you to identify the numeric fields to total. You can include the row and column coordinates for a form report. The fields you specify in the Total clause are also displayed as part of the report. If the Group clause is used in the same report, a subtotal of the data is produced.

*Note:* The Total clause applies only to numeric fields. Do not total on alphanumeric, date, or time fields.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TO LIST <code>listname</code></td>
<td>A list of record IDs that satisfy the query is written to the selection list for the current project. The list can be used as the source for another query.</td>
</tr>
<tr>
<td>TO LIST <code>listname</code> FIELD <code>fieldname</code></td>
<td>The values of the named field that satisfy the query are saved to the selection list. Only the specified field is saved to the selection list; the record keys are not. The resulting selection list is sorted by the key values. You can change the sort order by using <code>SORT BY</code>.</td>
</tr>
<tr>
<td>TO LP</td>
<td>Output is directed to the printer that is assigned to the user.</td>
</tr>
</tbody>
</table>
**Syntax**

TOTAL ON \{(column,row)\} KEY | fieldname_1 \{,fieldname_2,...\}

- **column** The column number on the form report at which to position the first character of the field or text.
- **row** The row number on the form report at which to position the first character of the field or text.
- **fieldname_n** The field on which you want to total.

**Examples**

SORT BY PRICE TOTAL ON PRICE

(1,3) [TOTAL BALANCE:] TOTAL ON (15,3) PRICE

**Grand Total Clause**

The grand total option allows you to include text on the grand total line. The grand total is based on the field used in the Total clause. You must use the Total clause to use the grand total option. If you do not include text for the grand total line, the default is a series of asterisks (***), followed by the grand total amount.

The grand total options are listed in Table 8-12. You can also use general options with the grand total option. General options are listed in Table 8-5.
Syntax

GRAND-TOTAL {{text:{'grand_total_options'}}}

  text  The text to be displayed on the grand total line.

Examples

  TOTAL ON TOT.AMT SORT BY ACCT GRAND-TOTAL ['U']
  TOTAL ON BALANCE SORT BY STATE GRAND-TOTAL [GRAND TOTAL
  BALANCE IS:]

Grand Total Options

The options listed in Table 8-12 allow you to further define the display of the grand total. The options must be inside square brackets and enclosed within single quotation marks. Within the brackets, you can group the options within single quotation marks or put single quotation marks around each option, for example: ['LP'] or ['L’ ‘P'].

Table 8-12. Grand Total Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>['L']</td>
<td>Prints a blank line before the grand total line.</td>
</tr>
<tr>
<td>['P']</td>
<td>Inserts a page break after the grand total line.</td>
</tr>
<tr>
<td>['U']</td>
<td>Inserts a double line before the grand total line.</td>
</tr>
</tbody>
</table>
Archive allows you to copy modules from the EDGE level or project level to a specified file or tape backup. These modules can then be transferred to another system, another project, or another site. Examples of modules include project data files and EDGE user IDs. For a listing of modules that can be archived, see Table 9-1.

Archive is useful at sites that have separate development and production EDGE systems. Once a project is finalized on the development system, it can be archived to the production system.

To archive modules you must define an archive instruction that specifies the modules to be stored. After the instruction is defined, to actually store the modules, use the Begin Archive command. To transfer the modules to another system, project, or site, use the Begin Unarchive command. The unarchive options are defined within the archive instruction; therefore, the requirements of the
unarchive need to be understood before defining the archive instruction.

The `archive` and `UNARCHIVE` can also be done using ACL commands.

The following are described in this chapter:

- Defining an Archive Instruction.
- Beginning the Archive.
- Beginning the Unarchive.

Note: There are modules of EDGE that cannot be archived, such as site information. To save all the information associated with EDGE, use the backup utilities provided with your operating system.

To display the Archive page, select the following:

EDGE / Tools area / Archive tab

A page similar to the following is displayed:
Defining an Archive Instruction

The archive instruction specifies which EDGE and project modules to archive. An archive instruction does not automatically archive an entire system or project; the modules to be archived must be identified in the archive instruction.

When you add or modify an instruction, a page similar to the following is displayed:
**Properties:**

- **Archive instruction name**: The name of the archive instruction.
- **Block size**: The number of characters in a destination block. Block size is meaningful only when the archiving to tape.

You can get this information from your system administrator. The default is 5120.
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th>Description</th>
<th>A description of the archive instruction, which can include notes or instructions for processing the archive instruction.</th>
</tr>
</thead>
<tbody>
<tr>
<td>File/Area</td>
<td>The file or area to be archived; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td>(All)</td>
<td>Archives all screens, logic flows, results, tables, queue definitions, result groups, embedded expressions, and toolbars.</td>
<td></td>
</tr>
<tr>
<td>Logics</td>
<td>Archives the logic flows specified in the Items property.</td>
<td></td>
</tr>
<tr>
<td>Screens</td>
<td>Archives the screens specified in the Items property.</td>
<td></td>
</tr>
<tr>
<td>Expressions</td>
<td>Archives all embedded expressions.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This property is only available if the module is Guide, Global database, or Project database.</td>
<td></td>
</tr>
<tr>
<td>File option</td>
<td>Specifies whether fields, or fields and data are to be archived; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td>Fields</td>
<td>Archives the field definitions for the file.</td>
<td></td>
</tr>
<tr>
<td>Fields and data</td>
<td>Archives all of the field definitions and data stored in the file.</td>
<td></td>
</tr>
</tbody>
</table>
Properties: (continued)

- **Format**: Generic is the only value. Display only.
- **Items**: The specific logic flows or screens to be archived. To select all logic flows or screens, select (all). Multiple logics or screens can be selected by holding down the CTRL key while clicking on the individual items.
  
  This property is only available if the module is Guide and logics or screens is specified in the File/Area property.

- **Last modified date**: The date the archive instruction was last modified. Display only.

- **Module**: The module to be archived. The modules that can be selected are described in Table 9-1. A maximum of 120 modules can be included in one archive instruction.
  
  To archive an individual logic flow or screen, select Guide, then specify the logic flow or screen name in the Items property.

- **Number of modules**: The number of modules in the archive instruction. A maximum of 120 modules is allowed per archive instruction. Display only.

- **Path**: The destination path of the archive data. This is extracted from I/O device definition. Display only.
Project

The location of the module to be archived. For modules in projects, select the project name. If the module is located at the EDGE level, the system automatically sets this property to EDGE.

Server I/O device

The destination (I/O device) for the archived data. For information on defining an I/O device, see the section I/O Device Setup.

Unarchive options

The unarchive option for the module. For an example of how the different options work, see Figure 9-1. Can be one of the following:

- Leave (formerly Add)
  Adds only new records to the target system. For example, if a file does not exist on the target system, the archived file is added to the system. If the file already exists on the target system, only new records are added to the file.

For example, if you are copying logic flows, and want to keep any logics that might exist on the destination system, specify this option. Only the logic flows that do not exist (that is, do not have the same name) are added to the destination system.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear (formerly Replace)</td>
<td>Clears the existing module on the target system and replaces it with the archived module. The entire contents of the existing module are replaced. For example, if the module being archived is a file, when the file is unarchived, the entire existing file is replaced. For example, if you are copying logic flows, and want to completely replace all of the logics that exist on the destination system, specify this option. All of the logic flows that exist on the destination are deleted, and are replaced with these new ones.</td>
</tr>
<tr>
<td>Update (formerly Overwrite)</td>
<td>Adds new modules and records to the target system and overwrites existing records that have the same ID as the archived records. For example, if the file does not exist on the target system, the file is added to the system. If the file exists on the target system, new records are added to the module and existing records are overwritten with archived records that have the same ID.</td>
</tr>
</tbody>
</table>
Properties: (continued)

For example, if you are copying logic flows and want to add all new logics, update any logics that might also exist on the destination system, and leave the rest of the logics alone, specify this option.
### Unarchive options

#### Archived Logics

- **Add**
  - Logics C, D

- **Replace**
  - Logics C, D

- **Overwrite**
  - Logics C, D

<table>
<thead>
<tr>
<th>Unarchive options</th>
<th>Archived Logics</th>
<th>Logics on the Destination System</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Before Unarchive</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logics A, B, C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unchanged: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logics C, D</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unchanged: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Logics A, B, C</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Unchanged: None</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Figure 9-1. Sample Unarchive Results
### Table 9-1. Archive Modules

<table>
<thead>
<tr>
<th>Module</th>
<th>Level</th>
<th>Data Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>3270</td>
<td>EDGE</td>
<td>All keymaps, play configurations, and scripts.</td>
</tr>
<tr>
<td>Ad hoc reports</td>
<td>Project</td>
<td>All ad hoc report records and ad hoc batch reports. If unarchive is performed on a different platform, you must resave all ad hoc reports to ensure correct syntax.</td>
</tr>
<tr>
<td>EDGE counters</td>
<td>EDGE</td>
<td>All counters used by EDGE, including project counters.</td>
</tr>
<tr>
<td>Guide</td>
<td>Project</td>
<td>Screens, logic flows, results, tables, queue definitions, result groups, embedded expressions and toolbars. The source code only is saved; therefore after unarchiving, EDGE forces a full assembly of the guide.</td>
</tr>
<tr>
<td>Global database</td>
<td>EDGE</td>
<td>A user-defined global database file. If data is archived, associated cross-reference data is also archived.</td>
</tr>
<tr>
<td>Global monitor</td>
<td>EDGE</td>
<td>All global monitor records.</td>
</tr>
<tr>
<td>Global guide</td>
<td>EDGE</td>
<td>Modified EDGE system screens, ActiveX definitions, External libraries and functions.</td>
</tr>
<tr>
<td>Import/Export</td>
<td>Project</td>
<td>Import and export instructions.</td>
</tr>
<tr>
<td>Instructions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O device</td>
<td>EDGE</td>
<td>All I/O device configurations.</td>
</tr>
</tbody>
</table>
### Table 9-1.  Archive Modules (continued)

<table>
<thead>
<tr>
<th>Module</th>
<th>Level</th>
<th>Data Affected</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operator access</td>
<td>Project</td>
<td>Operator access definitions. All user IDs must be resaved after unarchiving.</td>
</tr>
<tr>
<td>Project correspondence</td>
<td>Project</td>
<td>All project correspondence documents.</td>
</tr>
<tr>
<td>Project database</td>
<td>Project</td>
<td>A single project database file. If you select data and fields to archive, the cross-reference data is also archived. To archive call information, you must archive the STATUS File and HISTORY File.</td>
</tr>
<tr>
<td>Project monitor</td>
<td>Project</td>
<td>All operator and source monitor records, including the project-to-date monitors. If the guide is also archived, the data records and result groups are stored.</td>
</tr>
<tr>
<td>Seeding instructions</td>
<td>Project</td>
<td>All seeding instructions for queues.</td>
</tr>
<tr>
<td>Styles and schemes</td>
<td>EDGE</td>
<td>All styles and schemes.</td>
</tr>
<tr>
<td>Tasks</td>
<td>Project</td>
<td>All task codes are archived. All tasks must be reassembled after unarchiving.</td>
</tr>
<tr>
<td>User IDs</td>
<td>EDGE</td>
<td>All user IDs on the system.</td>
</tr>
</tbody>
</table>
Beginning the Archive

The Begin Archive command performs the selected archive instruction.

To start the command, select an instruction and select the Begin Archive command or select the Begin Archive command from the menu while an instruction is open. (You can also begin the archive by using the ACL ARCHIVE command.) When you begin the archive, a page similar to the following is displayed:

```
Archive

Instruction ID: SOFTSELL
I/O Device ID: tmp_file
Archive location: File: /tmp/file_361a
Format: Generic
Start time: 09:22AM
Current activity: Archive completed

Account/Project ID:
File name:
Record count:
Elapsed time (HH:MM:SS): 00:00:00

Press ENTER to continue:
```
### Displayed Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account/Project ID</td>
<td>The ID of each account or project is displayed as it is archived.</td>
</tr>
<tr>
<td>Archive location</td>
<td>The path to the device where the data is being stored.</td>
</tr>
<tr>
<td>Current activity</td>
<td>The type of archive activity being performed.</td>
</tr>
<tr>
<td>Elapsed time</td>
<td>The time since the archive process began.</td>
</tr>
<tr>
<td>File name</td>
<td>The files associated with the project or account are displayed as they are archived.</td>
</tr>
<tr>
<td>Format</td>
<td>The format of the instruction; Generic is the only value.</td>
</tr>
<tr>
<td>I/O Device ID</td>
<td>The ID of the I/O device.</td>
</tr>
<tr>
<td>Instruction ID</td>
<td>The name of the archive instruction.</td>
</tr>
<tr>
<td>Record count</td>
<td>The number of records in the file is displayed as the file is archived.</td>
</tr>
<tr>
<td>Start time</td>
<td>The time the archive process began.</td>
</tr>
</tbody>
</table>
**Beginning the Unarchive**

The Begin Unarchive command performs the selected unarchive instruction, which unpacks the stored information according to the unarchive options defined in the archive instruction.

Unarchiving can also be done in ACL using the `UNARCHIVE` command. This command allows you to override settings in the archive instruction, such as the project name or the unarchive option for restoring modules. If you unarchive a project on the same platform as the archive platform, you must use the ACL UNARCHIVE command to override the project name.

---

**Caution!** If a DataBase Link file is to be unarchived, the outcome may be different from unarchiving a c-tree file. For more information, see the EDGE 7.11 Database Link Reference Manual.

---

**Before Unarchiving**

Unarchiving can substantially alter your system and project modules, especially when the instruction uses the replace or overwrite option. This means, for example, that data in your current files will be replaced with data stored when the instruction was archived. As a precaution, if you plan to unarchive to a system or project that currently exists, we recommended you back up the destination system or project before beginning the unarchive.
Be careful about unarchiving between UNIX and NT platforms as the NT platform is not case sensitive. Objects created in UNIX versions of EDGE that use the same name but different cases (for example, First and first) will not work correctly.

Although you can perform an unarchive through the Task Scheduler, you should not do so until you have successfully performed the process several times using the menu selection. It is possible to destroy data by selecting certain unarchive options.

After the unarchive is complete, you may need to do the following:

- If you unarchive an EDGE project created with a prior version of EDGE, you should upgrade the project to your latest version of EDGE. To upgrade a project, see your upgrade procedures.

- If you unarchive the STATUS file, you must rebuild the queues. For information on rebuilding queues, see Chapter 14, Queues.

- If you unarchive a guide, you must assemble the guide to build the object code. (The archive function saves only source code.)

- If you unarchive ad hoc reports to a different platform from the archived platform, you must resave them to ensure correct syntax.
The Unarchive Command

To start the Begin Unarchive command, select an Archive instruction and select the Begin Unarchive command. (You can also select the Begin Unarchive command from the menu while an instruction is open.)

A dialog box similar to the following is displayed:

The Unarchive Options dialog box allows you to change the following:

- Server I/O device; change this setting if you have moved the archive file to a new location. For example, if you originally
archived to an I/O device that pointed to a file on your system, then later copied the file to tape. Specify a new I/O device name that points to the tape drive.

- Block size; change this setting if you have changed the settings on your tape drive and to use a different block size.

To continue the unarchive process, select Unarchive. To cancel, select Cancel.

If you select Unarchive, the unarchive process begins and a page similar to the following is displayed:
Properties:

**Account/Project ID**
- The ID of each account or project being unarchived.

**Archive Location**
- The path to the device where the data is stored.

**Creation Time/Date/User**
- The time and date the archive was performed and the user ID of the user who performed the archive.

**Current Activity**
- The unarchive activity currently being performed; each activity is displayed as it is performed.

**Description**
- The description provided in the archive instruction.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed Time</td>
<td>The time elapsed since the unarchive process began.</td>
</tr>
<tr>
<td>File Name</td>
<td>The files associated with the project or account being unarchived.</td>
</tr>
<tr>
<td>Format</td>
<td>The format of the instruction; Generic is the only value.</td>
</tr>
<tr>
<td>I/O Device ID</td>
<td>The ID of the I/O device.</td>
</tr>
<tr>
<td>Instruction ID</td>
<td>The ID of the archive instruction.</td>
</tr>
<tr>
<td>Record Count</td>
<td>The number of records in the file being unarchived.</td>
</tr>
<tr>
<td>Started At</td>
<td>The time the unarchiving began.</td>
</tr>
</tbody>
</table>
Listing Archive Instructions

To list available archive instructions, select one of the following from the Archive menu:

- List current archive instruction.
- List all archive instructions.

The report is similar to the following:

```
List Archive Instructions

I/O Device : ARCHIVE (Flat File, /dev/rmt0)
Block Size : 5120
Format : Generic
Last Modified : 12:53:29 PM 03/23/1998 by GST6
Total Modules :

Description :

This is for documentation of EDGE.
```
Overview

ACL (AdvantEDGE Command Language) provides an interface between EDGE and the UNIX operating system. ACL provides the following features:

- Command Line Editing using Insert Mode or Edit Mode.
- Online Help with summary and detailed information.
- Security Features for specific ACL commands.

You can invoke ACL commands, which provide common functions at the command line level. You can also invoke any UNIX command by preceding the command with an exclamation point (!—this is sometimes referred to as a bang sign). (Note that EDGE ACL commands are not preceded by an exclamation point.)

The EDGE ACL commands are listed in Table 10-1. They are described later in this chapter. The commands should be entered in the case shown.
Many of the ACL commands can be used in tasks by using them with the EXECUTE verb. These commands are noted in the table.

**Table 10-1. EDGE ACL Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANALYZELOGIC</td>
<td>Counts the number of times a verb is used in a logic flow.</td>
<td></td>
</tr>
<tr>
<td>ARCHIVE</td>
<td>Invokes an EDGE archive instruction.</td>
<td>Y</td>
</tr>
<tr>
<td>CHANGEPTR</td>
<td>Allows you to change the default printer.</td>
<td></td>
</tr>
<tr>
<td>CHECKFILE</td>
<td>Checks the condition of EDGE files.</td>
<td></td>
</tr>
<tr>
<td>chkdb2xref</td>
<td>Checks records in a database file against the related cross-reference file to ensure the cross-reference entries are correct.</td>
<td></td>
</tr>
<tr>
<td>chkxref2db</td>
<td>Checks cross-reference entries against the related database file to ensure the entries are valid.</td>
<td></td>
</tr>
<tr>
<td>CLEANTMP</td>
<td>Deletes files in the EDGE temporary work directory.</td>
<td></td>
</tr>
<tr>
<td>CLEAR-FILE</td>
<td>Clears all records from a file.</td>
<td></td>
</tr>
<tr>
<td>CLEARUSER</td>
<td>Clears locked user IDs.</td>
<td></td>
</tr>
<tr>
<td>compact</td>
<td>Condenses an EDGE file.</td>
<td>Y</td>
</tr>
<tr>
<td>COPY</td>
<td>Copies records.</td>
<td></td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
<td>Execute</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>COPYLIST</td>
<td>Copies a selection list.</td>
<td></td>
</tr>
<tr>
<td>copy.proj</td>
<td>Copies an EDGE project.</td>
<td>Y</td>
</tr>
<tr>
<td>CORRPRINT</td>
<td>Prints correspondence reports.</td>
<td>Y</td>
</tr>
<tr>
<td>COUNT</td>
<td>Counts fields in a record or records in a file.</td>
<td></td>
</tr>
<tr>
<td>crestore</td>
<td>Restores an EDGE file.</td>
<td>Y</td>
</tr>
<tr>
<td>ctreescp</td>
<td>Copies a c-tree file.</td>
<td>Y</td>
</tr>
<tr>
<td>ctrlchar</td>
<td>Removes control characters from a file.</td>
<td></td>
</tr>
<tr>
<td>DELETELIST</td>
<td>Deletes a selection list.</td>
<td></td>
</tr>
<tr>
<td>DELHELP</td>
<td>Deletes user-defined online help.</td>
<td></td>
</tr>
<tr>
<td>DELREC</td>
<td>Deletes fields in a record or records in a file.</td>
<td></td>
</tr>
<tr>
<td>DELRECS</td>
<td>Obsolete. Use DELREC.</td>
<td>Y</td>
</tr>
<tr>
<td>DUMP</td>
<td>Displays records to the screen.</td>
<td></td>
</tr>
<tr>
<td>dump_shadow</td>
<td>Displays SmartMerge tracking records to the screen.</td>
<td></td>
</tr>
<tr>
<td>edgeio</td>
<td>Imports data from or exports data to an I/O device file.</td>
<td>Y</td>
</tr>
<tr>
<td>EDIT</td>
<td>Edits records.</td>
<td></td>
</tr>
<tr>
<td>EDITHELP</td>
<td>Creates and modifies online help for commands.</td>
<td></td>
</tr>
</tbody>
</table>
Table 10-1. EDGE ACL Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDITLIST</td>
<td>Edits a selection list.</td>
<td></td>
</tr>
<tr>
<td>EXECTASK</td>
<td>Executes a task.</td>
<td>Y</td>
</tr>
<tr>
<td>filecheck</td>
<td>Checks for fragmentation in a file.</td>
<td>Y</td>
</tr>
<tr>
<td>FILEINFO</td>
<td>Returns information about the file.</td>
<td></td>
</tr>
<tr>
<td>FIXFILE</td>
<td>Rebuilds files.</td>
<td></td>
</tr>
<tr>
<td>GOTO</td>
<td>Returns to EDGE.</td>
<td></td>
</tr>
<tr>
<td>INITIALMON</td>
<td>Initializes the local, project or global monitor.</td>
<td></td>
</tr>
<tr>
<td>LIST</td>
<td>Lists record IDs in a specified file.</td>
<td></td>
</tr>
<tr>
<td>LISTFILES</td>
<td>Lists all files in the account.</td>
<td>Y</td>
</tr>
<tr>
<td>LISTLISTS</td>
<td>Lists selection lists.</td>
<td></td>
</tr>
<tr>
<td>LOOKUP</td>
<td>Looks up a specific string in a file.</td>
<td>Y</td>
</tr>
<tr>
<td>LS</td>
<td>Synonym for LISTFILES.</td>
<td></td>
</tr>
<tr>
<td>LSREC</td>
<td>Lists record IDs in a specified file.</td>
<td></td>
</tr>
<tr>
<td>LSTAPE</td>
<td>Lists I/O devices.</td>
<td></td>
</tr>
<tr>
<td>LSTAPELOCKS</td>
<td>Lists I/O devices currently in use.</td>
<td></td>
</tr>
<tr>
<td>MOVE</td>
<td>Moves a record.</td>
<td></td>
</tr>
<tr>
<td>MULTI.ASSEM</td>
<td>Invokes a multiproject assembly instruction.</td>
<td>Y</td>
</tr>
<tr>
<td>PASSWD</td>
<td>Sets a password for an ACL command.</td>
<td></td>
</tr>
</tbody>
</table>
### Table 10-1. EDGE ACL Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRINTMON</td>
<td>Prints a local, global or project monitor report.</td>
<td>Y</td>
</tr>
<tr>
<td>proclook</td>
<td>Displays UNIX processes.</td>
<td>Y</td>
</tr>
<tr>
<td>Qclear</td>
<td>Clears specified call events from EDGE files.</td>
<td>Y</td>
</tr>
<tr>
<td>Qdelete</td>
<td>Deletes a queue definition.</td>
<td>Y</td>
</tr>
<tr>
<td>Qrebuild</td>
<td>Rebuilds all or specified queues.</td>
<td>Y</td>
</tr>
<tr>
<td>Qseed</td>
<td>Seeds call events.</td>
<td>Y</td>
</tr>
<tr>
<td>Qtrans</td>
<td>Transfers or deletes call events from one or more queues.</td>
<td>Y</td>
</tr>
<tr>
<td>RMREC</td>
<td>Synonym for DELREC.</td>
<td></td>
</tr>
<tr>
<td>RMTAPELOCKS</td>
<td>Removes a lock from an I/O device.</td>
<td></td>
</tr>
<tr>
<td>SECURE</td>
<td>Assigns a specific security level to an ACL command.</td>
<td></td>
</tr>
<tr>
<td>SELECT</td>
<td>Produces an EQL statement.</td>
<td></td>
</tr>
<tr>
<td>SETSTDERR</td>
<td>Specifies stderr destination.</td>
<td></td>
</tr>
<tr>
<td>set_tracking</td>
<td>Enables record tracking for SmartMerge operations.</td>
<td></td>
</tr>
<tr>
<td>SHOWSECURITY</td>
<td>Shows security levels required to use ACL commands.</td>
<td></td>
</tr>
</tbody>
</table>
Accessing and Exiting ACL

Before you can access ACL, your user ID must have been granted permission to ACL. To grant permission, use the Users Tab tool.

Before entering ACL, be sure you are at the correct level (EDGE or project) for the command you wish to invoke. If you are at the project level, be sure you are in the correct project.

Table 10-1. EDGE ACL Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
<th>Execute</th>
</tr>
</thead>
<tbody>
<tr>
<td>smart_merge</td>
<td>Performs enhanced importing and exporting to/from multiple I/O devices.</td>
<td>Y</td>
</tr>
<tr>
<td>TAPEFWD</td>
<td>Moves tape forward to next tape file.</td>
<td></td>
</tr>
<tr>
<td>TAPERead</td>
<td>Displays tape data in octal.</td>
<td></td>
</tr>
<tr>
<td>TAPERew</td>
<td>Rewinds tapes.</td>
<td></td>
</tr>
<tr>
<td>UNARCHIVE</td>
<td>Transfers archived modules from one project to another.</td>
<td>Y</td>
</tr>
<tr>
<td>vercont</td>
<td>Invokes the EDGE version control process.</td>
<td>Y</td>
</tr>
<tr>
<td>vfiles</td>
<td>Performs multiple filechecks.</td>
<td>Y</td>
</tr>
<tr>
<td>XQUERY</td>
<td>Executes ad hoc reports.</td>
<td>Y</td>
</tr>
<tr>
<td>xrefall</td>
<td>Builds a cross-reference file.</td>
<td>Y</td>
</tr>
</tbody>
</table>
**EDGE on UNIX**

To access ACL from EDGE Developer, select ACL from the Window menu.

To exit ACL, enter one of the following in uppercase or lowercase:

- EXIT
- MENU
- GOTO {screen.name}
- OFF

All the commands return you to the screen from which you entered ACL.

You can also exit ACL by pressing **CTRL+D**.

**EDGE on Windows 2000**

If you are running EDGE on Windows 2000, ACL is available using telnet. Because of differences in interfaces between UNIX and NT, do **not** use the ACL feature from within EDGE Developer or EDGE Client.

To access ACL from a telnet session, use the following procedure:

1. From the Windows Start menu, select the Run... command. Enter the following:

   `telnet hostname`
where *hostname* is the name of your NT server.

2. At the login prompt, enter the name of your EDGE account (this is usually edge).

3. If prompted for a password, enter the appropriate password.

4. A prompt similar to the following is displayed:

   a(cl), e(tw), edg(eo), UPGRADE, sh(ell)<CR>=sh

   To go to ACL, enter a.

5. A prompt similar to the following is displayed:

   Enter Project Name:

   Enter the project name. To work at the EDGE level, press RETURN without entering anything.

6. The ACL prompt is displayed.

   **Note:** To go to another project at ACL, you must log out of the telnet session and log back in.

---

**Wildcards**

Some commands provide for the use of wildcards, such as an asterisk (*). If a command supports the use of wildcards, to use a wildcard character as an actual character, you must precede the character with a backslash (\). For example, to dump a record from the ACTZ
Command Line Editing

When you access ACL, you are in insert mode at the command line. The ACL command line prompt is similar to the following:

[INS, 101]->

The prompt has the following information:

<table>
<thead>
<tr>
<th>INS</th>
<th>Indicates the mode. Can be INS (insert mode) or EDT (edit mode).</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Indicates the number in the ACL history stack where the current command is to be stored.</td>
</tr>
</tbody>
</table>

ACL History

When you enter a command at the ACL command line, a copy of the command is placed in the ACL history stack. Each command is numbered. The first command that is entered is command 1. The next command is command 2, and so on.

The ACL history stack can contain up to 100 commands. If a command is entered and there are already 100 commands in the stack, the lowest numbered command is deleted and the latest command is added to the stack.
The stack is associated with the port on which it was created. All commands created on that port are stored in the same stack. If you are on a network, this means that when you log off, then log on again, you may have different commands saved in your stack.

You can access commands in the history stack when you are in editor mode. A stack pointer keeps track of the command you are currently accessing. To search the stack and copy a command to the command line, use the ACL history stack edit commands, which are listed in Table 10-2.

**Insert Mode**

When you are in insert mode, you can enter any UNIX or EDGE ACL command. To switch from insert mode to edit mode, press ESCAPE key.

**Edit Mode**

When you are in edit mode, you can recall a command from the ACL history stack and invoke it as is, or first edit it, then invoke it. For example, if you have entered a long ACL command and need to make a minor change, you can recall the command, make the change, then invoke it.
Note: Editing the command is usually easier than retyping the command.

The editor commands are listed in Table 10-3. These commands can be performed only from edit mode, not from insert mode. The editor commands are case-sensitive. Type only the commands. Do not press ENTER.

To switch from edit mode to insert mode, use one of the following editor commands:

A a C
cw dh i

Table 10-2. ACL History Stack Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>/text</td>
<td>Searches the command history stack for the specified text. If the string is found, moves the stack pointer to that line in the command history stack and copies the command to the command line. Remains in edit mode. When you specify the command, do not put a space between the slash (or question mark) and the text. After you have entered the text, execute the search by pressing ENTER, the down arrow, or CTRL+J.</td>
</tr>
<tr>
<td>?text</td>
<td></td>
</tr>
</tbody>
</table>
### Table 10-2. ACL History Stack Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dh</td>
<td>Lists the command history stack, then switches to insert mode.</td>
</tr>
<tr>
<td>{n}G</td>
<td>Moves the stack pointer to line (n) in the command history stack and copies that command to the command line. If (n) is not specified, the last command in the stack is copied. Remains in edit mode.</td>
</tr>
<tr>
<td>{n}j</td>
<td>Moves the stack pointer down (n) lines in the command history stack and copies that line to the command line. If (n) is not specified, (n=1) is assumed. If there are fewer than (n) lines following the current line, moves the stack pointer to the last line in the stack and copies that line. Remains in edit mode.</td>
</tr>
<tr>
<td>{n}k</td>
<td>Moves the stack pointer up (n) lines in the command history stack and copies that line to the command line. If (n) is not specified, (n=1) is assumed. If there are fewer than (n) lines preceding the current line, moves to the first line in the stack and copies that line. Remains in edit mode.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>^</td>
<td>Moves the cursor to the beginning of the line. Remains in edit mode.</td>
</tr>
<tr>
<td>$</td>
<td>Moves the cursor to the last character of the line. Remains in edit mode.</td>
</tr>
<tr>
<td>.</td>
<td>Repeats the last editing command. Remains in edit mode.</td>
</tr>
<tr>
<td>{n}~</td>
<td>Toggles (n) characters on the line between upper case and lower case, starting with the character under the cursor. If (n) is not specified, toggles one character. Remains in edit mode.</td>
</tr>
<tr>
<td>A</td>
<td>Moves the cursor to the end of the line, then switches to insert mode.</td>
</tr>
<tr>
<td>a</td>
<td>Moves the cursor one character to the right on the line, then switches to insert mode.</td>
</tr>
<tr>
<td>{n}b</td>
<td>Moves the cursor (n) words to the left, and leaves the cursor on the first character of that word. If (n) is not specified, moves left one word. If there are fewer than (n) words to the left, moves the cursor to the first character of the first word. Remains in edit mode.</td>
</tr>
<tr>
<td>C</td>
<td>Deletes all characters to the right of the cursor, including the character under the cursor, then switches to insert mode.</td>
</tr>
</tbody>
</table>

Table 10-3. ACL Line Edit Commands
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cw</td>
<td>Deletes all characters to the right of the cursor up to the first space, including the character under the cursor, then switches to insert mode. If there are no spaces to the right of the cursor, deletes all characters up to the end of the line.</td>
</tr>
<tr>
<td>D</td>
<td>Deletes all characters to the right of the cursor, including the character under the cursor. Remains in edit mode.</td>
</tr>
<tr>
<td>dw</td>
<td>Deletes all characters to the right of the cursor up to the first space, including the character under the cursor. If there are no spaces to the right of the cursor, deletes all characters up to the end of the line. Remains in edit mode.</td>
</tr>
<tr>
<td>{n}h</td>
<td>Moves the cursor left n characters. If n is not specified, moves left one character. If there are fewer than n characters to the left, moves to the first character. Remains in edit mode.</td>
</tr>
<tr>
<td>i</td>
<td>Switches to insert mode; cursor remains on the current character.</td>
</tr>
<tr>
<td>{n}l</td>
<td>Moves the cursor right n characters. If n is not specified, moves right one character. If there are fewer than n characters to the right, moves to the last character. Remains in edit mode.</td>
</tr>
<tr>
<td>u</td>
<td>Undoes the last change. Remains in edit mode.</td>
</tr>
</tbody>
</table>
Online Help

Online help is available for ACL commands in both summary and detailed format. The commands used to invoke help are listed in Table 10-4.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>{n}x</td>
<td>Deletes n characters, starting with the character under the cursor. If n is not specified, deletes the character under the cursor. All characters to the right of the deleted characters are moved left. Remains in edit mode.</td>
</tr>
<tr>
<td>{n}w</td>
<td>Moves the cursor n words to the right, and leaves the cursor on the first character of that word. If n is not specified, moves right one word. If there are fewer than n words to the right, moves to the last character of the last word. Remains in edit mode.</td>
</tr>
</tbody>
</table>
To create your own online help for any command that is invoked within the ACL environment, use the `EDITHELP` command. For more information, see the section `EDITHELP`.

**Table 10-4. ACL Commands to Invoke Help**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>?</td>
<td>Displays a one-line description of the ACL commands. See Figure 10-1.</td>
</tr>
<tr>
<td>??</td>
<td>Displays a detailed description of the ACL commands. See Figure 10-2.</td>
</tr>
<tr>
<td>command?</td>
<td>Displays a one-line description of the specified command.</td>
</tr>
<tr>
<td>command??</td>
<td>Displays a detailed description of a specific command.</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ACL</td>
<td>Command line editor for Edge system environment.</td>
</tr>
<tr>
<td>CHANGEPTR</td>
<td>Changes the default printer.</td>
</tr>
<tr>
<td>CHECKFILE</td>
<td>Checks for the sanity of an EDGE file.</td>
</tr>
<tr>
<td>CLEANTMP</td>
<td>Remove temporary files from EDGE account.</td>
</tr>
<tr>
<td>CLEARUSER</td>
<td>Clears a locked EDGE user id.</td>
</tr>
<tr>
<td>COPY</td>
<td>Copies a record to another record.</td>
</tr>
<tr>
<td>COUNT</td>
<td>Very fast count of records in a file.</td>
</tr>
<tr>
<td>DELETELIST</td>
<td>Deletes the specified selections lists.</td>
</tr>
<tr>
<td>DELHELP</td>
<td>This is to delete a help record.</td>
</tr>
<tr>
<td>DELREC</td>
<td>Deletes a record from a file.</td>
</tr>
<tr>
<td>DUMP</td>
<td>Dumps the contents of a record to the screen.</td>
</tr>
<tr>
<td>EDIT</td>
<td>Edit an EDGE record.</td>
</tr>
<tr>
<td>EDITHELP</td>
<td>Add/modify user defined online help.</td>
</tr>
<tr>
<td>EDITLIST</td>
<td>Edits the specified selections lists.</td>
</tr>
<tr>
<td>EXIT</td>
<td>Exit to the EDGE System Control Menu.</td>
</tr>
<tr>
<td>FILEINFO</td>
<td>Displays some information about a file.</td>
</tr>
<tr>
<td>FIXFILE</td>
<td>Fixes a corrupted file. Might delete some records.</td>
</tr>
<tr>
<td>GOTO</td>
<td>Go back where left off in EDGE.</td>
</tr>
</tbody>
</table>

*Figure 10-1. Sample One-Line ACL Help Descriptions*
Figure 10-2. Sample Detailed Help Description

Security Features

An important element of ACL is its security features:

- Users can be granted or denied access to ACL through the User ID Definition.
- Individual ACL commands can be protected by requiring a password or by giving access to users based on their security level with one of the following commands:
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SECURE</strong></td>
<td>Secures a specified command by protecting the command based on the security level of the user ID.</td>
</tr>
<tr>
<td><strong>PASSWD</strong></td>
<td>Password-protects a specific command.</td>
</tr>
</tbody>
</table>
Alphabetical Listing of Commands

This section describes all the EDGE ACL commands in alphabetical order.

Conventions

The following headings are used to describe each command:

- Syntax.
- Interactive Prompts.
- Notes.

Syntax

The syntax for each command is described under the heading Syntax. This describes all the areas and properties that can be used, in the order that they must be entered.

In the definitions, the following conventions are used:

- Italics indicate variable properties.
- The pipe symbol ( | ) indicates that exactly one of the enclosed properties can be specified.
- Braces ( {} ) indicate optional properties.
In the following example of a syntax statement, replace the word *filename* with the name of the file you wish to list. You can optionally enter a second filename. The `-c` and `-n` are optional, and only one can be selected, as is true for the `dict` and `data` properties. If you specify those properties, you would enter them exactly as shown.

```
LSREC {-c |-n} { data|dict } filename1 {filename2}
```

**Interactive Prompts**

Many of the commands can be used interactively by entering just the command name. Syntax for the interactive prompts is described under the heading Interactive Prompts. If there are no interactive prompts, the heading is not used.

The commands may have different features depending on whether the complete command is entered or interactive prompts are used. For example, if you use the command line, you can enter just the filename and the file must be in the current directory. However, if you use interactive prompts and the following prompt is displayed, you can enter a complete pathname to the directory containing the file:

```
Dir [default 'data']:
```

In the description of the interactive prompts, the properties are described only if they differ from the syntax description.
Notes

The notes section provides additional information, such as the level where the command must be executed.
ANALYZELOGIC

The ANALYZELOGIC command counts the number of times a verb is used in a logic flow.

Syntax

ANALYZELOGIC -A | -S list_ID | -I logic_IDs

- A   Analyzes all logic flows in the project.
- I logic_IDs  Analyzes the specified logic flow.
- S list_ID  Analyzes all the logic flows based on the specified selection list.

There are two steps to creating a report with the output.

1. Execute the ANALYZELOGIC command from ACL at the project level.
2. Create and run an EQL report that displays the verb counts. A sample EQL statement and output are provided in this section.

When you execute the ANALYZELOGIC command, the logic flows being processed are displayed on the screen. For an example, see Figure 10-3. The results of the command are sent to a file named LFREPORT. This file is cleared and regenerated each time the command is executed.
Figure 10-3.  ANALYZELOGIC Display

You can display the results of the ANALYZELOGIC command, by creating an ad hoc report or running an EQL statement. The LFREPORT file is a project level file. The key to the file is Logic_ID*Verb; that is, the name of the logic flow, followed by an asterisk, followed by the name of a verb. The file contains the following fields:

- **LFID**: The name of the logic flow containing the verb.
- **VERB**: The verb that is being counted.
The following EQL statement displays all the fields in the LFREPORT file.

```
SELECT FROM LFREPORT LFID VERB CNT
```

For a sample of the report, see **Figure 10-4**.

![Figure 10-4. Logic Flow Report](image)

- **CNT** The number of times the verb is located in the specified logic flow.

For a sample of the report, see **Figure 10-4**.
The ARCHIVE command invokes an EDGE archive instruction.

Syntax

ARCHIVE instruction {-B n} {-F} {-M n} {-P pathname} {-V}

- instruction
  The name of the archive instruction as created through the archive module. To specify more than one instruction, enter the instruction names separated by a space. The instructions are executed in the order entered and one is completed before the next begins.

  You must create the archive instructions before executing the ARCHIVE command.

- B n
  The block size. The default is 4000. This property overrides the block size, if any, specified in the archive instruction. The block size property is meaningful only when the destination is a tape.

- F
  Displays all the modules being processed as the archive instruction is executed. This option requires a you to press ENTER at the end of each screen display.

  This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.
The ARCHIVE command must be invoked at the system level.

-M \{n\} Specifies the progress (milestone) of the process is to be displayed by the -V option every n records. The default is 100 if n is not specified.

-P pathname The UNIX pathname to the target storage. If not specified, the I/O device identified in the archive instruction is used.

-V Displays all the messages returned by the command, scrolling the messages if necessary.

Notes

The ARCHIVE command must be invoked at the system level.
The `CHANGEPTR` command allows you to change the default server printer for the current session only.

**Syntax**

```plaintext
CHANGEPTR printer ID
```

- **printer ID**
  
  The name of the printer to be assigned as the default. The printer ID must exactly match the ID that is defined in the printer setup.

**Interactive Prompt**

If the printer ID is not specified, the following prompt is displayed:

```
Printer id [must fill]:
```

**Notes**

This command changes the default server printer only while you are in ACL. When you exit and return to EDGE Developer, the printer is reset to its previous value. To change the default server printer in EDGE Developer, use the Set Server Printer command in `Server Setup` on the File menu.
The CHANGEPTR command can be invoked from the EDGE or project level.
The CHECKFILE command performs c-tree functions to validate the integrity of a file and reports its status.

This command is for c-tree files only.

**Syntax**

CHECKFILE file1 {file2…}

*filen* The names of files to check.

**Interactive Prompts**

If you do not specify the filename as part of the command, the following prompts are displayed:

Dir [default 'data']:
Filename(s) [must fill]:

**Notes**

You can use CHECKFILE to check if a file has been corrupted. For example, if the system goes down while an EDGE process is running, a file may become corrupted. By using CHECKFILE, you can determine its condition.
The command verifies both the data and the dict areas of the file. You can specify the dict area of the file only by using the interactive prompts.

You can also check for file corruption by using the Check File for Corruption command on the Database Maintenance tab. You can also use the filecheck command.

The CHECKFILE command must be invoked from the level that contains the files to check.
chkdb2xref

The chkdb2xref command checks each record in a specified database file against the related cross-reference file to ensure the cross-reference entries are correct. If specified, the command can add missing entries to the cross-reference file.

Syntax

chkdb2xref {-?|-h} {-f} {-k n} {-m n} {-v} filename

-? | -h  Displays the command usage to the standard out file.

-f  Specifies missing entries are to be written to the cross-reference file.

-k n  Specifies to skip the first n records in key order in the database file before starting to check.

-m n  Specifies the progress (milestone) of the process is to be displayed by the -v option every n records. The default is 100 if this option is not specified.

-v  Displays all the messages returned by the command, scrolling the messages if necessary.

filename  The name of the database file on which to run the command.
Notes

The chkdb2xref command reads each record in the database file with a lock and does not release the lock until the record is verified (and optionally fixed) for all the cross-reference entries for that record. This allows the command to be run during live operations. If a record that is needed is locked by an operator, the command waits for the lock to be released.

If an entry is missing in the cross-reference file and the -f option is not specified, a message similar to the following is sent to the standard out file:

    Missing xref word 'word' from record 'record.ID'

If an entry is missing in the cross-reference file and the -f option is specified, a message similar to the following is sent to the standard out file:

    Missing xref word 'word' from record 'record.ID'
    Adding [record.ID]:<word> xref entry

The chkdb2xref command can be executed from ACL or from the EXECUTE verb in either a logic flow or task.

For information on checking the cross-reference file against the database file, see the chkxref2db command.
The chkxref2db command checks each cross-reference entry against the related database file to ensure the entries are valid. The command can also be specified to remove extra entries from the cross-reference file.

**Syntax**

```
chkxref2db {-?| -h} {-f} {-k n} {-m n} {-v} filename
```

- `-?| -h` Displays the command usage to the standard out file.
- `-f` Specifies extra entries are removed from the cross-reference file.
- `-k n` Specifies to skip the first `n` entries in key order in the cross-reference file before starting to check.
- `-m n` Specifies the progress (milestone) of the process is to be displayed by the `-v` option every `n` records. The default is 100 if this option is not specified.
- `-v` Displays all the messages returned by the process, scrolling the messages if necessary.

`filename` The name of the database file on which to run the command.
Notes

The chkxref2db command retrieves each entry in the cross-reference file and locks the corresponding database record. The lock is not released until the entry is verified (and optionally fixed) for all the cross-reference entries for that record. This allows the command to be run during live operations. If a record that is needed is locked by an operator, the command waits for the lock to be released.

If a cross-reference entry is not in the database file and the -f option is not specified, a message similar to the following is sent to the standard out file:

   Bad xref word 'word' from record 'record.ID'

If a cross-reference entry is not in the database file and the -f option is specified, a message similar to the following is sent to the standard out file:

   Bad xref word 'word' from record 'record.ID'
   Removing [record.ID]:<word> xref entry

The chkxref2db command can be executed from ACL or the EXECUTE verb in either a logic flow or task.

For information on checking the database file against the cross-reference file, see the chkdb2xref command.
The CLEANTMP command clears the entries in the EDGE temporary work directory located in /.../edge/EDGE/tmp.

**Syntax**

CLEANTMP

**Notes**

Temporary work files are created in this directory when someone logs into EDGE. If an operator does not log off properly, these files remain in the directory and occupy disk space.

You should have all users log off before performing this command. If you invoke the command when users are logged on, not all entries in the directory are cleared.

The CLEANTMP command can be invoked from the EDGE or project level.
Cleaning up /usr6/edgedev/EDGE/tmp
1- Deleting <37466.EBF0>
2- Deleting <37596.EBF0>
3- Deleting <37596.ERS0>
4- Deleting <48948.EBF0>
5- Deleting <48948.ERS0>
6- Deleting <49046.EBF0>
7- Deleting <49046.ERS0>
8- Deleting <49058.EBF0>
9- Deleting <49058.ERS0>
10- Deleting <67680.EBF0>
11- Deleting <67680.ERS0>
12- Deleting <67704.EBF0>
13- Deleting <67704.ERS0>
14- Deleting <67706.EBF0>
15- Deleting <67706.ERS0>
16- Deleting <88314.edgevi>
17- Deleting <CEFserver_T.b>
18- Deleting <CEFserver_T.c>
19- Deleting <nohup.out>

<page 1> Press <cr> to continue. s = scroll, q = quit item, Q = quit all:

white text
The CLEAR-FILE command clears all records from a specified file.

**Syntax**

CLEARFILE {DICT} *filename*

- **DICT** Indicates the DICT portion of the file is to be cleared.
- **filename** The name of the file to be cleared.
### CLEARUSER

The CLEARUSER command clears locked user IDs.

**Syntax**

`CLEARUSER {user_ID1 {user_ID2 ... }}`

- `user_IDn` User ID to be cleared.

**Interactive Prompts**

If you do not specify the user ID as part of the command, the following prompt is displayed:

`User id(s) [must fill]:`

**Notes**

Be very careful with this command. Clear a locked user ID only when the user is not on the system. Clearing a user ID that is in use can cause system problems. For example, it may allow another user to log in using the released user ID. In this case, the system may not recognize the valid user. It can also cause problems with Project Monitor reports.
You can also clear locked users through the Release Locked User ID command or Release All Locked User IDs command on the Users tab.

The CLEARUSER command can be invoked from the EDGE or project level.
**compact**

The compact command condenses a file by eliminating space or fragmentation in the file. It cannot be used on cross-reference files nor on the EDGE queues file.

This command is for c-tree files only.

**Syntax**

compact file1 {file2…}

  
  *filen*  
  Name of file to be compacted.

**Notes**

The command first copies all the valid records in the file to a new file, deletes the original file, then renames the new file to the original filename.

You must have enough space on the partition to hold both the original and new files. If you are compacting a list of files, you need at least as much disk space as required by the largest file in the list of files.
You can also compact files by using the Compact File command on the Database Maintenance tab at the EDGE or project level.

The files that are to be compacted must not be in use.

When you invoke compact, the following prompt is displayed:

Before proceeding, make sure that you have enough space available on this partition. Do you want an estimate of the space needed for each file? (y/n/q)[y]

To have the system determine the estimate, press ENTER. A report similar to the following is produced:
To continue, press ENTER. The following prompt is displayed:

Backup original file(s) to tape before compaction? (y/n/q)[y]

To back up your files, press ENTER. You are strongly encouraged to do this. If anything goes wrong, your files may be damaged or deleted.

When the process is completed, a report similar to the following is displayed:
Check the error log file, using the path given in the report. To read the error file, enter the following at the ACL prompt:

```plaintext
!cat error.log.path
```

The compact command must be invoked from the level that contains the files to be compacted.
The COPY command allows you to copy records from one file to another, or within the same file. You must be in the project that contains the files.

**Syntax**

COPY {{-F} {dict} srcfile srcrecord_id {targetrecord_id | targetfile}} | -F srcfile -S list targetfile

- **-F** Specifies the copy is to a different file. If not specified, the record is copied to the source file.
- **dict** Copies the dictionary portion of the file. If not specified, data records are copied.
- **srcfile** Name of the source file.
- **srcrecord_id** ID of the source record. To copy all records, enter an asterisk (*).
- **targetrecord_id** ID of the target record. If the record already exists, it is overwritten. This property is valid only if copying to a record in the same file.
**targetfile**  Name of the target file. Must be used with -F. The record is copied with the same ID as the source. (To copy a record to a different file and change its ID, use the interactive prompts.)

It is recommended that both files have the same layout.

**-S**  Specifies the IDs of the source records are in a selection list.

**list**  Name of the selection list that contains the IDs of the records to copy.

---

**Interactive Prompts**

If you do not specify properties, the following series of prompts is displayed:

- Dir [default 'data']:
- Source filename [must fill]:
- Source item(s) [must fill]:
- Dir [default 'data']:
- Target filename :
- Target item :
**Notes**

To use a selection list with the interactive prompts, enter the following at the Source item(s) prompt and do not specify IDs for the Target item prompt:

`s list`

To copy from files that are not in the current project, use the interactive prompts and specify the path name at the Dir prompt. For example, to copy from a system level file specify `/usr/edge/EDGE data` at the Dir prompt.

To copy field definitions, it is recommended you use EDGE Developer copy features to copy the appropriate field definitions. This ensures the field definitions are created correctly.
The COPYLIST command allows you to copy one selection list to a second selection list. You must be in the project that contains the list.

**Syntax**

COPYLIST \{source\_list destination\_list\}

\textit{source\_list} \hspace{1cm} The name of the source selection list.

\textit{destination\_list} \hspace{1cm} The name of the destination selection list.

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

Source selection list [must fill]:
Destination selection list [must fill]:

**Notes**

The destination selection list is created if it does not already exist. You cannot display this selection list through the List Detail command on the Selection List tab. However, you can use the list with any command that accepts a selection list.
To show all selection lists whether created in ACL or in ad hoc reporting, use the `LISTLISTS` command.
The copy.proj command copies all source components of one project to another project. It does not copy the object code.

**Syntax**

copy.proj {-c} -s source_proj -d destination_proj

- **-c** Clears the operator access in the destination project.
- **-s source_proj** The name of the source project.
- **-d destination_proj** The name of the destination project, which must already exist. If there is existing data in the destination project, it will not be available after the copy.

**Notes**

The copy.proj command must be invoked from the system level.

Both projects must exist in EDGE before the command is performed. Also, before invoking the copy, make sure the destination partition has adequate space for the copy.
The source project is closed to operations during the copy. A log of copy activity is created in the $EDGE/tmp directory with the name copyprojpid, where pid is the process ID used in the copy.

This command makes an exact duplicate of the source project. You should check the following areas, which may refer to the source project name, and change the name to the destination project, if appropriate:

- **Operator access.** By default, each user who has been given access to the operations area of the source project is given access to the operations area of the destination project. To clear all operator access to the destination project, use the -c option.

- **Project counter.** By default, the name of the counter, if any, specified in the source project is copied to the destination. To use the name of the current project as the name of the project counter, do not specify anything in the project counter property in the project options area. If the name of the project counter is blank, EDGE automatically uses the name of the project as the project counter.

- **Project options.**

- **User IDs.** The name of the destination project is copied to the supervisor project access for each user who has supervisor access to the source project.
After the project is copied and any necessary changes made, you must perform a full assembly.

If the copy.proj command fails, the message "$EDGE/tmp/copyprojxxx (where xxx is the process ID) is sent to the standard error file.

**Relational Database Notes**

Files are copied exactly to the destination project. When a file that uses a relational database is copied to the destination project, the file in the destination project is marked as having been created in EDGE or imported, as the case may be, exactly as specified in the source project. The destination file points to the same relational database table as the source file. EDGE, however, is not aware of this.

A problem arises if the file was created in the source account (not imported), and you then delete the file from one project and not the other. EDGE deletes the database table being pointed to, but does not delete the file in the second project. The file now points to a non-existent table. To prevent this, if you delete a file from a project that has been copied to or from another project, you must delete the file from both projects.

If the file in the source was originally imported, it is marked as imported in the destination. You cannot delete imported files, so the problem described above does not arise.
The CORRPRINT command prints correspondence reports.

**Syntax**

CORRPRINT *correspondence* {-A} {-B} {-C n} {-I record.IDs} {-P {printer}} {-Q} {-S listname}

- *correspondence*  
  The name of the correspondence to be printed.

- **-A**  
  Prints a correspondence document for each record in the default file.

- **-B**  
  Does not wait on error or require a response. If there is an error, a message is briefly displayed on the terminal. Can be used with the task scheduler.

- **-C n**  
  Prints the number of copies specified by *n*.

- **-I record.IDs**  
  Prints correspondence for the specified record IDs. The record IDs must be separated by a comma; for example, -I 100, 101, 102.

- **-P {printer}**  
  Sends output to the printer. If the printer name is not specified, the report is printed on the default printer. Your printer can be set up for EDGE using Printer Setup.
Notes

The CORRPRINT command must be invoked from the project that contains the Correspondence reports. If the command is executed from the task scheduler, you should use the -B option to ensure the task does not wait.
COUNT

The COUNT command counts the number of records in a specified file.

Syntax

COUNT {{ data|dict } filename1 { filename2 ...}}

data Specify records in the data section of the file are to be counted. This is the default.
dict Specify records in the dict section of the file are to be counted.

filename The name of the file to be counted. To specify the name of more than one file, separate the filenames with spaces.

Interactive Prompts

If no properties are specified, the following prompts are displayed:

Dir [default 'data']:
Filename(s) [must fill]:
Notes

The COUNT command must be invoked from the level that contains the files to count.
crestore

The crestore command enables you to restore backups made during the compact process. You can use this, for example, if the compact was interrupted or did not complete.

Syntax

crestore

Notes

Run crestore under the same directory (EDGE or project) as the backup. If you do not, you may overwrite files in the wrong project.

When you invoke crestore, the following prompt is displayed:

Enter tape path ( [q] to quit )
( <RETURN> for default /dev/rmt0 ) =>

To quit, enter q. To accept the default tape path, press ENTER. Otherwise, enter the tape path where the tape is mounted.

If you continue, the following prompt is displayed:

Please insert your backup tape and press return when ready
To continue, mount the backup tape and press ENTER. To cancel at this point, press the ESC key.
The ctreecp command copies records from one file to another file in the same project.

**Syntax**

ctreecp {-B} source.file destination.file

- **-B** Sets up the command to allow it to be run in the UNIX background. (This does not cause the command to run in the background. To run in the background, the command must be specified at the UNIX level and the & flag used.)

- **source.file** Name of the file that contains the records to copy. To copy field definitions, enter the filename prefixed by F_, for example, F_CUSTOMERS.

- **destination.file** Name of the file into which the records are copied. If you are copying field definitions or cross-reference records, use the appropriate prefix.

   If the specified file does not exist, ctreecp creates it.

**Notes**

If you copy the field definitions, you must rebuild them. For more information, see the section Fields.
The ctreecp command must be invoked from the level that contains the files to copy.
The `ctrlchar` command checks for and optionally removes control characters in an EDGE file.

**Syntax**

```plaintext
ctrlchar {-options} filename {record1 {record2…}}
```

**options**  
The following options can be specified:

- `-I` Includes the record ID in the control character search and removal.
- `-Q` Disables any record messages other than failures.
- `-R` Removes any control characters found in records.
- `-V` Displays all record messages.
- `-8` Removes eight-bit characters.

**filename**  
The name of the file to check.

**record**  
The name of the record to check.

**Notes**

If you do not specify options, the `ctrlchar` utility checks for control characters only; it does not remove them, nor does it check for eight-bit characters.
Control characters are characters with an ASCII value of 0 through 31, with the exception of the attribute, value, and sub-value marks, which have ASCII values of 30, 29, and 28, respectively. Eight-bit characters are characters with an ASCII value greater than 127.

You might use this command, for example, to eliminate possible control characters after importing a file.

You can also use the Check for Control Characters command and Remove Control Characters command on the Database Maintenance tab in the level that contains the files to check.
**DELETELIST**

The DELETELIST command deletes one or more selection lists.

**Syntax**

DELETELIST \{list1 \{list2 ...\}\}

- \(lstmt\) The name of the list to delete.

**Interactive Prompts**

If no properties are specified, the following prompt is displayed:

Selection list id(s) [must fill]:

**Notes**

The DELETELIST command must be invoked from the level that contains the lists to delete.
**DELHELP**

The DELHELP command deletes any online help you have created for ACL or UNIX commands.

**Syntax**

DELHELP command

*command* The command for which the online help is to be deleted.

**Interactive Prompts**

If no properties are specified, the following prompt is displayed:

Commands :

**Notes**

Only online help that you have defined can be deleted. EDGE system online help supplied is not deleted.

The DELHELP command can be invoked from the EDGE or project level.
**DELREC**

The DELREC command allows you to delete records or field definitions from a file. It also deletes associated records from the cross-reference file, if any. You can delete specific records or definitions, or the deletion can be based on a selection list.

**Syntax**

```
DELREC { data|dict } filename { record_id1 {record_id2…} } {-S listname}
```

- **data** Specifies that records in the data section of the file are to be deleted. This is the default.
- **dict** Specifies that records in the dict section of the file are to be deleted.
- **filename** The name of the source file.
- **record_id** The name of the record to delete. To delete all the records in the file, enter an asterisk (*).
- **listname** The name of the selection list that contains the names of the records to delete.

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

```
Dir [default 'data']:
```
Notes

The DELREC command must be invoked from the level that contains the records to delete.
**DELRECS**

The DELRECS command is obsolete and is retained for backward compatibility only. To delete records, use the DELREC command.
The DUMP command allows you to display records to the screen.

**Syntax**

DUMP \{dict | data\} filename \{record_id1 \{record_id2…\}\} {-S listname}

- **data** Specifies that records in the data section of the file are to be displayed. This is the default.
- **dict** Specifies that records in the dict section of the file are to be displayed.
- **filename** Name of the file.
- **record_id** Name of the record to display. To display all the records in the file, enter an asterisk (*).
- **listname** The name of the selection list that contains the names of the records to display.

**Interactive Prompts**

If a property is missing, the following prompts are displayed:

- Dir [default 'data']:
- Filename [must fill]:
- Item(s) [must fill]:
Notes

The DUMP command must be invoked from the level that contains the records to display.
dump_shadow

The dump_shadow command displays SmartMerge tracking records to the screen.

Syntax

dump_shadow {-f DB_filename}

DB_filename Dumps records related to specified file; if not used, all records are displayed.

Notes

SmartMerge tracking records are kept in a file called S_SHADOW.
edgeio

The edgeio command is used to import (receive) data from or export (send) data to an I/O device file.

**Syntax**

```
edgeio {options} instruction
```

The following options can be specified:

- **-a**
  Appends the export file to the end of an existing file. This property cannot be used with the -o option. This option cannot be performed through the Export tab.

- **-b n**
  Uses a block size of n. The block size must be a multiple of the record length. For example, if the record length is 250, the block size could be 250, 500, 750, and so on. The default is 512.

- **-c counter**
  When the file is imported, you can specify the initial value to be used to assign record IDs. This has effect only if there is no record ID field specified in the import instruction. This is valid for import only.

- **-e**
  Performs an export. If not specified, the system assumes the instruction is to import.
-f Displays information in full screen format. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-i /O device Uses the specified I/O device instead of the device specified in the instruction. Cannot be used with -p option.

-l listname The listname of the saved selection list for record IDs to be exported.

-m {n} Specifies the progress (milestone) of the process is to be displayed by the -v option every n records. The default is 100 if n is not specified.

-o Overwrites the existing file when exporting. This is the default.

-p path-name Uses the path of the device instead of I/O device ID. This property cannot be used with the -i option.

-r Rewinds I/O device first.

-s n Skips n records from the tape device when the file is imported.

-t Test mode. Does not write to an EDGE file or device.
### edgeio

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-v</code></td>
<td>Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.</td>
</tr>
<tr>
<td><code>-z</code></td>
<td>Displays the records in the specified I/O device, one at a time.</td>
</tr>
</tbody>
</table>

- `instruction` The name of the import or export instruction. The instruction must be created from the Import or Export data sheet, as appropriate.

### Notes

You must have created Import Instructions or Export Instructions before using this command.

The edgeio command can be invoked from the EDGE or project level.

### Example

```bash
dedgeio -e EXP.RPT
```
**EDIT**

The EDIT command allows you to edit records in a file.

**Syntax**

```
EDIT {dict | data} filename { record_id1 {record_id2…} } {-S selection_list}
```

- **data** Specifies that records in the data section of the file are to be edited. This is the default.
- **dict** Specifies that records in the dict section of the file are to be edited.
- **filename** Name of the file.
- **record_idn** ID of the record to edit. To edit all the records in the file, enter an asterisk (*).

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

Dir [default 'data']:
Filename [must fill]:
Item(s) [must fill]:
Notes

Editing is done using full-screen editing commands, which are listed in Table 10-5. These commands are based on the UNIX vi editor. For more information on these commands, see any standard UNIX vi documentation.

Saving and Exiting

To save your work, you must be in edit mode. If you are in replace or input mode, to return to edit mode, press ESC. Then press the colon (:). The cursor drops to the bottom of the page. You can then enter one of the following:

- **q** Quits without saving. If you have made changes that have not been saved you are warned. You can cancel the quit and save the changes or use q! to quit unconditionally.
- **q!** Quits without saving. You are not warned that you have made changes.
- **w** Saves (writes) and remains in edit mode.
- **wq!** Saves (writes) and quits edit mode.
Table 10-5. Full-screen Editing Commands

<table>
<thead>
<tr>
<th>Operation</th>
<th>Keystroke</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backspace</td>
<td>backspace</td>
<td>Moves cursor from right to left on the line.</td>
</tr>
<tr>
<td>Change case</td>
<td>{n}~</td>
<td>Toggles n characters on the line between upper case and lower case, starting with the character under the cursor. If n is not specified toggles one character. Remains in edit mode. The cursor is positioned one character beyond the number specified (which includes spaces), or at the last character on the line.</td>
</tr>
<tr>
<td>Copy lines</td>
<td>{n}YL</td>
<td>Copies (yanks) n lines from the cursor position.</td>
</tr>
<tr>
<td>Copy words</td>
<td>{n}YW</td>
<td>Copies (yanks) n words from the cursor position. (To paste the text, use the P command.)</td>
</tr>
<tr>
<td>Delete characters</td>
<td>{n}X</td>
<td>Deletes n characters from the cursor position to the right.</td>
</tr>
<tr>
<td>Delete lines</td>
<td>{n}CL</td>
<td>Deletes n number of lines. Changes to insert mode.</td>
</tr>
<tr>
<td></td>
<td>{n}DL</td>
<td>Deletes n number of lines. Remains in edit mode.</td>
</tr>
<tr>
<td>Delete words</td>
<td>{n}CW</td>
<td>Deletes n number of words. Changes to insert mode.</td>
</tr>
<tr>
<td>Operation</td>
<td>Keystroke</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Delete words</td>
<td>{n}DW</td>
<td>Deletes (n) number of words. Remains in edit mode.</td>
</tr>
<tr>
<td>Go left</td>
<td>{n}H</td>
<td>Moves the cursor left (n) characters.</td>
</tr>
<tr>
<td>Go right</td>
<td>{n}L, space-bar</td>
<td>Moves the cursor right (n) characters; can also press SPACEBAR.</td>
</tr>
<tr>
<td>Go to line</td>
<td>{n}G</td>
<td>Moves cursor to line (n). Can also press \text{G}\ ENTER, then the line number.</td>
</tr>
<tr>
<td>Go to beginning</td>
<td>0 (zero)</td>
<td>Moves cursor to the beginning of the current line.</td>
</tr>
<tr>
<td>Go to end</td>
<td>A</td>
<td>Moves the cursor to the end of the line, then changes to insert mode.</td>
</tr>
<tr>
<td>Go to end &amp; edit</td>
<td>$</td>
<td>Moves the cursor to the last character of the line. Remains in edit mode.</td>
</tr>
<tr>
<td>Go to next word</td>
<td>{n}W</td>
<td>Moves the cursor to the beginning of the (n)th word on the line. Can only be used on the line in which the cursor is positioned.</td>
</tr>
<tr>
<td>Go up lines</td>
<td>{n}K</td>
<td>Moves the cursor up (n) lines.</td>
</tr>
<tr>
<td>Go down lines</td>
<td>{n}J</td>
<td>Moves the cursor down (n) lines.</td>
</tr>
<tr>
<td>Insert line</td>
<td>O</td>
<td>Inserts a line after the current line and changes to insert mode.</td>
</tr>
<tr>
<td>Operation</td>
<td>Keystroke</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Next occurrence</td>
<td>N</td>
<td>Used with the search command to show the next occurrence of the value found.</td>
</tr>
<tr>
<td>Paste text</td>
<td>P</td>
<td>Pastes text copied by the YL or YW command.</td>
</tr>
<tr>
<td>Replace mode</td>
<td>R</td>
<td>Replaces single character.</td>
</tr>
<tr>
<td>Search</td>
<td>/text</td>
<td>Searches for the specified text.</td>
</tr>
<tr>
<td>Substitute</td>
<td>S</td>
<td>Enters input mode; you can substitute the characters.</td>
</tr>
<tr>
<td>Undo</td>
<td>U</td>
<td>Undoes the previous command and moves the cursor back to the previous position.</td>
</tr>
</tbody>
</table>
EDITHELP

The EDITHELP command allows you to create and modify your own online help for UNIX shell scripts or to add to the detailed help for EDGE ACL commands.

Syntax

EDITHELP command

command    The command for which online help is to be edited.

Interactive Prompts

If no properties are specified, the following prompt is displayed:

Commands :

Notes

Editing is done using full-screen editing commands, which are listed in Table 10-5.

You can specify both summary and detailed help messages. The screens for each are presented automatically. Summary help is limited to 54 characters. There is no limit for detailed help.
The EDITLIST command allows you to add, modify, or delete records in a selection list. Editing is done using full-screen editing commands, which are listed in Table 10-5.

**Syntax**

```
EDITLIST list1 {list2 … }
```

listn The list to be edited.

**Notes**

The EDITLIST command can be invoked from the EDGE or project level.
**EXECTASK**

The EXECTASK command executes a task.

**Syntax**

EXECTASK *project.name task.name*

- *project.name*: The name of the project that contains the task. If the task is at the global level, specify the word EDGE.
- *task.name*: The name of the task to execute.

**Notes**

The EXECTASK command is intended to be used in tasks that are run by the task scheduler.

When the task specified by EXECTASK is finished, the user invoking the command is logically logged off the port by EDGE. Any subsequent command or function that is issued may fail if the command checks for user privileges.

Because the task scheduler handles logging off and on ports each time a new task is invoked, the logoff by EXECTASK has no effect on the next task in the task scheduler. In addition, EXECTASK checks for
a user and if it does not find one, it logs on a dummy user. This allows you to specify multiple EXECTASKs in a task.

To execute a task from another task, you can also use chaining.

For more information on tasks, see Chapter 15.
The filecheck command provides diagnostic information about a file. This command is for c-tree files only.

**Syntax**

```
filecheck {options} {data|dict} filename
```

*options*  The following options can be specified:

- `-l`  Opens the file in EXCLUSIVE mode so that no one else can change the file while it is in the middle of being filechecked.

- `-m`  Displays summary only.

- `-s`  Suppresses the check for consistency between the data and index portions of the file.

*data*  Specifies the data section of the file is to be checked. This is the default.

*dict*  Specifies the dict section of the file is to be checked.

*filename*  The name of the file to be checked.
**Notes**

The filecheck command checks for consistency between the data and index portions of the file, checks for fragmentation, and indicates if the file should be compacted.

The -l option directs filecheck to open the file it is checking in EXCLUSIVE mode, so that no one else can change the file while it is in the middle of being filechecked. However, in the event there are already locks on the file, filecheck cannot open the file in EXCLUSIVE mode, and it fails with the following error message:

```
Could not properly open filename.
```

That is, filecheck using the -l option fails if someone has a record in the file locked. Depending on the file, if filecheck is run when there are live users, this error may occur. If it does, you can either rerun the filecheck without the -l option, or rerun it at a later time.

If the -l option is not used and filecheck is run while there are live users, the following message is displayed if the file is updated during the checking process:

```
WARNING: File filename has been updated during the filecheck process.
```

If this message is displayed and if filecheck reports errors, run filecheck again.
You can use this command to determine if the file is corrupted. The total number of records plus the number of resource nodes should always equal the total number of active nodes. If there is a discrepancy, the file is most likely corrupted. If there is an inconsistency or if the file is corrupted, you can fix the file by using the **FIXFILE** command.

You must perform this command from the level where the file is located. If the command does not function, make sure you are in the correct project or directory.

You can also check files by using the Check File for Corruption command on the **Database Maintenance** tab in the level that contains the files to check.
The following information is displayed:

- Total active nodes.
- Resource nodes.
- Total deleted nodes.
- Total B-tree nodes.
- Total space wasted.
- Percent fragmentation in file.
- Total space that may be restored.
- Current file size.
- Total space needed for this file.
- Total number of records.
- Total number of records for each secondary key that has data.
The FILEINFO command displays information about a specified file.

**Syntax**

```plaintext
FILEINFO { data|dict } filename
```

- **data** Specifies that information about the data section of the file is to be displayed. This is the default.
- **dict** Specifies that information about the dict section of the file is to be displayed.

**filename** The name of the source file.

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

```
Dir [ default 'data']:
Filename(s) [must fill]:
```

**Notes**

This command returns the full pathname, the record count, the amount of disk space (bytes) used, and the average disk space per record, similar to the following:
The FILEINFO command can be invoked from the EDGE or project level.
The FIXFILE command rebuilds files that have been corrupted.

This command is for c-tree data files only.

**Syntax**

```
FIXFILE { data|dict } filename1 {filename2 ... }
```

- **data** Specifies that the data section of the file is to be rebuilt. This is the default.
- **dict** Specifies that the dict section of the file is to be rebuilt.
- **filenamen** The names of files to fix.

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

```
Dir [ default 'data']: 
Filename(s) [must fill]:
```

**Notes**

To determine if a file is corrupted, you can use the filecheck command.
Before using this command, ensure that no one is using the file, then back up the file.

_Note:_ If the file is large, this command may take several hours to complete.

The FIXFILE command can be invoked from the EDGE or project level.
The GOTO command takes you from ACL back to EDGE.

**Syntax**

GOTO

**Notes**

You are returned to the window from which you entered ACL.

The GOTO command can be invoked from the EDGE or project level.
INITIALMON

The INITIALMON command initializes the logon, project, or global monitor, depending on the options specified.

Syntax

INITIALMON -L {-A n} | -P | -S { -V }

- A n  The age in days of Logon Monitor records to delete. For example, if seven is specified, all records with a start date seven or more days ago are deleted. The default is zero which deletes all records. Only supported with the -L option.

- L  Initializes Logon Monitor. Must be run from the EDGE level.

- P  Initializes Project Monitor. Must be run from the project level.

- S  Initializes Global Monitor. Must be run from the EDGE level.

- V  Displays all the messages returned by the process, scrolling the messages if necessary while the command processes.

Notes

To initialize the logon monitor or global monitor, the INITIALMON command must be invoked while EDGE is selected from the Browser. To initialize the project monitor, the command must be invoked from the appropriate project.
Examples

INITIALMON -P -V  This command initializes the project monitor and displays all messages returned by the process.

INITIALMON -L -A 7 -V  This command initializes the logon monitor, deleting records more than seven days old and displays all messages returned by the process.
**LIST**

The LIST command displays a list of record IDs in a specified file.

**Syntax**

```
LIST { { data|dict } filename record_id1 {record_id2…}}
```

- **data** Specifies that record IDs in the data section of the file are to be displayed. This is the default.
- **dict** Specifies that record IDs in the dict section of the file are to be displayed.
- **filename** Name of the file to be displayed.
- **record_id** IDs of the records to display. You can specify the exact IDs or use wildcards as follows:
  - To display all records, enter an asterisk (*).
  - To display records with common characters, specify those characters with an * in the location where the characters differ. For example, to display all record IDs that start with TEST, enter TEST*. To display all record IDs that end with TEST, specify *TEST.
To indicate any single character, specify a question mark (?) at the location of the character. You can specify more than one ?. For example, to display all record IDs that start with any character followed by the number 1, specify ?1. To display all record IDs that contain three characters with A as the middle character, specify ?A?.

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

- Dir [default 'data']:
- Filename [must fill]:
- Item(s) [must fill]:

**Notes**

The LIST command can be invoked from the EDGE or project level.
LISTFILES

The LISTFILES command displays a list of files in the current account or in a specified directory.

Syntax

LISTFILES \{pathname\}

pathname      The path to the directory that contains the files to list. To indicate the current directory, enter a period (.).

Interactive Prompts

If no property is specified, the following prompt is displayed:

Path [ default ' . ' ]:

Notes

The LISTFILES command can be invoked from the project or system level.
Example

ACL - Account: EDGE port: 40 Type 'goto' to exit.
<current directory>

.access
3270_PLAY
ACCESS
ACTCONF
ACTZ
ACTZ_ERRORS
ADHOC
AHBATCH
AHLOG
ARCHINST
ASSEM_INST
ASSEM_LOG
ASSEM_Pholds
ASSEM_WORK
BP_LOCAL
BPTASK
BULLETINS
CALLSET
CELLS_FILES
CELLS_FILES1
CELLS.ZLPC
COMM_INFO
CELLS_FILES2
Cells.ZATT
Cells.ZLPC
DB_Device
DB_INF
DIAler_STATUS
EDGE_COUNTERS
DBM
DBMINFO
COMM_INFO
EMAIL
EPASSWD
EXPORT.DEF
EXPORTINFO
EMAIL
EPASSWD
EXPORT.DEF
EXPORTINFO
EXT. FUNCTION
EXT.LIBRARY
F_ASEM.Pholds
F_LOCATIONS
F_TPLOG
F_ZATT
F_ZCEN
F_ZLPC
GLOBAL_CORRESP
GROUP
ICoSMERGE
ID_FILE
IMPORT.DEF
IMPORTINFO
INFTMDEF
IODEVICE
ISDINDEX
ITEM_DESC
KEY_MAP
LIST.ACPS
LIST.LOCS
LIST.ZIPS
LOCATIONS
LOG_CONTROL
LOGON.HIST
MD
MONITOR.INIT
ODBC.DSN
ODBC_INFO
PDW_QUEUE
PDW_QUEUE1
PDW_QUEUE2
PDW_QUEUE3
PDW_QUEUE4
PDW_XFER
PERFORMANCE
PFiLES
PORT.DEFINITION
PORTISON
PORTWORK
PRINTERS
PROJ.PARAM
PROJECTS
RECORDED.CALLS

<page 1> Press <cr> to continue, s = scroll, q = quit item, Q = quit all: }
LISTLISTS

The LISTLISTS command displays the names of selection lists created at the current level.

Syntax

LISTLISTS list1 {list2 ...}

listn  The selection list names to display. You can specify the exact names or use wildcards as follows:

To display all lists, enter an asterisk (*).

To display list names with common characters, specify those characters with an * in the location where the characters differ. For example, to display list names that start with LIST, enter LIST*. To display list names that end with LIST, specify *LIST.

To indicate any single character, specify a question mark ( ? ) at the location of the character. You can specify more than one ?. For example, to display list names that start with any character followed by the number 1, specify ?1. To display list names that contain three characters with A as the middle character, specify ?A?.

Interactive Prompts

If no properties are specified, the following prompt is displayed:
Selection list id(s) [must fill]:

**Notes**

If no selection lists exist that match the criteria, the following message is displayed:

No such lists

The LISTLISTS command can be invoked from the EDGE or project level.

For information on editing the items in the list, see the EDITLIST command.
LOOKUP

The LOOKUP command searches for specific characters in an EDGE file. Once the characters have been found, the record ID and the location of the first reference are displayed.

Syntax

LOOKUP string { data|dict } filename record_id1 {record_id2 ...}

string The characters to search for.
data Specifies the data section of the file is to be searched. This is the default.
dict Specifies the dict section of the file is to be searched.
filename The file in which to search.
record_id n IDs of records in which to search. You can specify the exact IDs or use wildcards as follows:

To search all records, enter an asterisk (*).

To search records with common characters in their name, specify those characters with an * in the location where the characters differ. For example, to search all records whose names start with TEST, enter TEST*. To search all records whose names end with TEST, specify *TEST.
To search records where a single character differs, specify a question mark (\( ? \)) at the location of the character. You can specify more than one ?. For example, to search all records whose names start with any character followed by the number 1, specify ?1. To search all records whose names contain three characters with A as the middle character, specify ?A?.

**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

- String to look for [must fill]:
- Dir [default ‘data’]:
- Filename [must fill]:
- Item(s) [must fill]:

**Notes**

The LOOKUP command can be invoked from the EDGE or project level.

**Example**

```
LOOKUP LOOKUP HISTORY *
```
The LS command is a synonym for LISTFILES.
**LSREC**

The LSREC command displays a list of all the record IDs in a file.

**Syntax**

```
LSREC {-c | -n } { data|dict } filename1 {filename2}
```

- `-c` Displays a columnar list of the record IDs. This is the default.
- `-n` Displays a line number for each record ID.
- `data` Displays record IDs in the data section of the file. This is the default.
- `dict` Displays record IDs in the dict section of the file.
- `filename...` The names of the files to be displayed.

**Interactive Prompts**

If no properties are specified, the following prompt is displayed:

```
Dir [default 'data']:
Filename(s) [must fill]:
```

**Notes**

The LSREC command can be invoked from the EDGE or project level.
**LSTAPE**

The LSTAPE command displays the EDGE-defined I/O devices and the paths for those devices.

**Syntax**

LSTAPE

**Notes**

The LSTAPE command can be invoked from the project or system level.

You cannot page this command. If you have defined more I/O devices than can be displayed on the screen, the list will scroll.
LSTAPELOCKS

The LSTAPELOCKS command lists locks on I/O devices in use on the system.

Syntax

LSTAPELOCKS

Notes

If there are no tape locks, the following message is displayed:

No current tape locks on the system.

The LSTAPELOCKS command can be invoked from the EDGE or project level.
The MOVE command moves one record to another record in the same file, or to a record in another file, then deletes the source record.

**Syntax**

```
MOVE { {-f}{ data|dict } srcfile record_id1 {record_id2 | { data|dict } targetfile }}
```

- `-f` Specifies the move is to a different file. If not specified, the record is moved to the source file.
- `data` Specifies to use the data section of the file. This is the default.
- `dict` Specifies to use the dict section of the file.
- `srcfile` The name of the source file for the move.
- `record_id1` The ID of the source record to be moved.
- `record_id2` The ID of the target record. If this record exists, it is overwritten with no warning. This property is valid only if moving to a record in the same file.
- `targetfile` Name of the target file. This is valid only if `-f` is specified. The record is created with the same ID as the source record. To copy a record to a different file and change its ID, use the interactive prompts.
**Interactive Prompts**

If no properties are specified, the following prompts are displayed:

- Dir [default 'data']:
- Source filename [must fill]:
- Source item(s) [must fill]:
- Dir [default 'data']:
- Target filename:
- Target item:

**Notes**

You must be in the project that contains the files.
The MULTI.ASSEM command invokes a previously written multiproject assemble instruction. You must create the instruction through the Multiple Project Assembly/Swap tab sheet in the EDGE Guide area.

**Syntax**

MULTI.ASSEM *instruction_ID* {-B}

*instruction_ID*  The ID of the multiproject assemble instruction.

-B  Executes the assembly without displaying prompts.

**Notes**

To run the MULTI.ASSEM command, you must enter ACL from the EDGE level. If you attempt to run it from a project level, an error similar to the following is displayed:

THIS PROCESS MUST BE PERFORMED FROM THE EDGE LEVEL.

You can also perform multiproject assemblies from Multiple Project Assembly/Swap tab sheet in the EDGE Guide area.
PASSWD

The PASSWD command can be used to password any ACL command, including user-defined commands.

Syntax

PASSWD (command1 {command2…})

commandn  The command to be passworded. If several commands are specified, they will all have the same password.

Interactive Prompts

After you invoke the command, the following prompts are displayed:

Commands:

Enter the names of the commands to be passworded. If you specified the name when you invoked PASSWD, this prompt is not displayed.

Please enter new password:

Enter the password. It can contain up to 14 characters. To delete a password, press ENTER for the new password.

Reenter password to confirm:
Retype the password exactly as you entered it.

If the command had a password previously, after you enter the new password information, you are requested to enter the old password.

**Notes**

When a passworded command is invoked, the following is displayed:

```
Cmd passworded, enter password (14 chars max):
```

If you password-protect a command, that command is added to the list of commands displayed by ACL help, if it is not already in the list. You can add help messages by using the EDITHELP command, which is described on page 1018. To delete the command from the help list, use the DELHELP command.

If the password for a command is forgotten, to remove the password, you must contact EDGE Customer Care.
The PRINTMON command prints a logon, global, or project monitor report.

Syntax

PRINTMON monitor {options}

**monitor** Indicates the type of report; can be one of the following:

- **LMON** Logon Monitor.
- **PMON** Project Monitor by Operator/Hour.
- **QMON** Project Monitor by Source Queue.
- **SMON** Global Monitor.

**options** The following options can be specified:

- **-C** Prints a cumulative report, from the last project-to-date initialization to the present.
  
  *Only used with Project Monitor.*

- **-D** Prints a report from the last time the project monitor was initialized to the present.
  
  *Only used with Project Monitor.*

- **-E date** The end date for the logon monitor records to include in the report.
only used with Logon Monitor.

-F Displays information in full screen format. This is the default. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-G group ID The EDGE group ID on which the report is to be based.

-P Sends the report to the default printer. This also turns off full screen mode. To use this command in the task scheduler, specify this option.

-R n The sorting for the report; can be one of the following:

1  By group.
2  By hour.
3  By group and hour.
4  By hour and group.
5  By user and date. Only used with Logon Monitor.

-S date The start date for the logon monitor records to include in the report.

Only used with Logon Monitor.
Notes

To display the system level report, the command must be invoked at the system level. To display a project level report, the command must be invoked at the project level.

If neither the -P or -T option are specified, the report is displayed on the terminal.

Example

PRINTMON PMON -D -R 3 -T This command runs the project monitor for all days since it was last initialized, sorted by group then hour. The report displays on screen.
PRINTMON LMON -S 10/12/99 -E 10/22/99 -R 1

This command reports the logon monitor data for the period starting on 10/12/99 and ending on 10/22/99. The report is sorted by group.
The proclook command displays UNIX processes.

**Syntax**

`proclook (option)`

- `option` Specifies the processes to display. One of the following can be specified:
  - `-d` Displays detached processes; that is, processes not attached to a terminal, such as daemons, servers, or runaway processes.
  - `-e` Displays EDGE processes; that is processes with EDGE as the user.
  - `-f` Displays defunct processes; that is processes that are finished, but not yet released by the parent process.
  - `-l {time}` Displays processes using more CPU time in minutes than specified in the option. The default is five minutes if `time` is not specified.
  - `-o` Displays all processes that are over one day old.
  - `-p pid` Displays all processes associated with the specified pid (process ID), both child processes and parent processes.
  - `-t term` Displays all processes associated with the specified terminal.
-u user  Displays processes for the specified user. Specifying EDGE as the user is the same as using the -e option.

**Notes**

If no option is specified, the command displays EDGE processes (-e option), detached processes (-d option), defunct processes (-f option), and processes over one day old (-o option).

The following information is displayed in the order listed:

- **User ID**  Displayed by all options except -e and -u.
- **Process ID**  Displayed by all options.
- **Parent ID**  Displayed by all options.
- **Terminal**  Displayed by all options except -t.
- **Start Time**  Displayed only by -t option.
- **CPU time**  Displayed by all options.
- **Command**  Displayed by all options.

**Example**

```
proclook -p 35090
```
<table>
<thead>
<tr>
<th>USER</th>
<th>PROCESS ID</th>
<th>PARENT ID</th>
<th>TERMINAL</th>
<th>CPU TIME</th>
<th>COMMAND</th>
</tr>
</thead>
<tbody>
<tr>
<td>edge</td>
<td>32892</td>
<td>1</td>
<td>-</td>
<td>13:34</td>
<td>SCREEN</td>
</tr>
<tr>
<td>edge</td>
<td>35090</td>
<td>32892</td>
<td>pts/10</td>
<td>0:18</td>
<td>/bin/ksh</td>
</tr>
</tbody>
</table>

Press any key to continue, q = quit item, Q = quit all:
Qclear

The Qclear command clears (deletes) specified call events from the following files:

- HISTORY
- queues
- RESPONSES
- STATUS
- STATUSID

Syntax

Qclear {options} {queue}

options  The following options can be specified:

-F Displays information in full screen format. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-H Suppresses all output to the screen. If -F or -V is specified after the -H, -H has no effect.
**Notes**

This command must be invoked from the project that contains the queue.

- **-K** Unlocks all locked call events. You should be cautious when using this option and not use it during live operations, because it could unlock call events in use by operators.

- **-L listname** Uses a selection list to select the queue names.

- **-M \{n\}** Specifies the progress (milestone) of the process is to be displayed by the -V option every n records. The default is 100 if n is not specified.

- **-P** Sends the output to the printer.

- **-V** Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.

**queue** The name of the queue to be cleared. If not specified, the -L option must be used.
The Qdelete command deletes a queue definition.

Syntax

Qdelete {options} queue

options  The following options can be specified:

- \( F \)  Displays information in full screen format. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

- \( H \)  Suppresses all output to the screen. If \( F \) or \( V \) is specified after the \( H \), \( H \) has no effect.

- \( K \)  Unlocks all locked call events. You should be cautious when using this option and not use it during live operations, because it could unlock call events in use by operators.

- \( L \) listname  Uses a selection list to select the queues to delete.

- \( M \) \( n \)  Specifies the progress (milestone) of the process is to be displayed by the \( V \) option every \( n \) records. The default is 100 if \( n \) is not specified.

- \( P \)  Sends the output to the printer.
Qdelete deletes the queue definition, not the data in the queue.

You can use this only when the queue is clear of all call events. If the queue contains any call events, Qdelete will fail.

This command must be invoked from the project that contains the queue.

- `queue`  
  The name of the queue to be deleted. If not specified, the -L option must be used.

- `-V`  
  Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.
Qrebuild

The Qrebuild command rebuilds all or specified queues. When a queue is rebuilt, the call events in the queues file are deleted and recreated by reading the STATUS file.

Syntax

Qrebuild {options} queue

options The following options can be specified:

-A Rebuilds all queues.

-B Moves active call events that were started more than 24 hours ago to the specified queue. The starting time is determined from the START field in the STATUS file. When the call event is moved, the following occurs:
- The call event is made inactive.
- The scheduled date and time are changed to the current date and time.

-F Displays information in full screen format. This is the default. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-H Suppresses all output to the screen.
This command must be invoked from the project that contains the queue.

If neither the -N nor the -O option is specified, all non-queued call events are moved to the ERROR queue.
This command should not be used during operations because the queue will appear empty.

If running a Qrebuild in a task after manually copying new dates and times into the STATUS File SCHED.DATE and SCHED.TIME fields, the STATUS file INDEX field must be cleared in order for the Qrebuild to work correctly.
**Qresched**

The Qresched command processes a time-ordered protected queue to ensure that calls are scheduled during preferred calling hours defined for the assigned agent or group. Specify the -Q option to interactively approve each rescheduled call. If you omit this option, calls are rescheduled automatically.

**Syntax**

Qresched { -FHPQV? } { -M # } queue-name

The following options can be specified:

- **-F**  Displays information in full-screen format. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

- **-H**  Suppresses all output to the screen. Should be used when executing a task from the task scheduler.

- **-M {n}**  Specifies the progress (milestone) of the process is to be displayed by the -V option every n records. The default is 100 if n is not specified.

- **-P**  Sends the output to the printer.
-Q  Interactively queries the user to approve each rescheduled call.

-V  Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.

-?  Prints command line usage to stderr.

queue-name  The name of the time-ordered protected queue.

**Notes**

- This command must be invoked from the project that contains the queue.
- Do not use the -Q option when you run this command within a task.

**Examples**

Qresched -F -M 100 -Q CAMPAIGN
The Qseed command seeds call events into a queue based on an existing queue seed instruction.

Note: If the target queue is a time-ordered protected queue and country and/or phone codes are defined for the user or group, the user or group time zone is used instead of the site time zone. Also, if a work schedule is defined for the associated user or group, then that work schedule is used instead of the default work schedule for the project.

To determine the time zone associated with a user or group, EDGE checks the agent record for country and phone codes first, then the group record, and then the site record. EDGE uses the first valid data encountered.

Syntax

Qseed {options} instruction

options The following options can be specified:

-C Allows multiple call events per call ID.

-D Deletes all call events based on the call ID. Then creates a new call event for the call ID.
-E n  Stops the seeding when the error count reaches n errors.

-F  Displays information in full screen format. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-H  Suppresses all output to the screen. Should be used when executing a task from the task scheduler.

-M {n}  Specifies the progress (milestone) of the process is to be displayed by the -V option every n records. The default is 100 if n is not specified.

-N n  Seeds only n call events specified.

-P  Sends the output to the printer.

-U  Unconditionally seeds call events to the specified queue. If there is already a record for the same call event, it is overwritten.

-V  Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.

**instruction**  The name of the queue seed instruction created through the queue seeding function.
Notes

This command must be invoked from the project that contains the queue.
The Qtrans command transfers or deletes call events from one or more queues.

**Syntax**

Qtrans source destination {options}

*source*  
The source queue; can be one of the following:
- `-F queue` Specifies the queue from which the call events are to be transferred or deleted.
- `-U` Specifies that call events are to be transferred or deleted from any queue.

*destination*  
The destination queue; can be one of the following:
- `-D` Deletes the call events specified by the *source*.
- `-T queue` Transfers the call events specified by the *source* to the specified *queue*.

*options*  
The following options can be specified:
- `-A` Transfers or deletes all the records in the specified queue; must be used in combination with the -F option. You cannot use -A with -U.
<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-B n</strong></td>
<td>Transfers call events where the scheduled date and time in the STATUS record is ( n ) minutes after the current time.</td>
</tr>
<tr>
<td><strong>-C date*time</strong></td>
<td>The new callback date and time. This is intended to be used with the -I option. ( Date ) can be entered in external format, for example, 12/06/95. ( Time ) can be entered in either external or internal format. ( Date ) and ( time ) must be separated by an asterisk (*).</td>
</tr>
<tr>
<td><strong>-E</strong></td>
<td>Restricts access according to the queue access (open, closed, open by schedule). Used in combination with the -F option.</td>
</tr>
<tr>
<td><strong>-H</strong></td>
<td>Suppresses all output to the screen. Should be used when executing from the task scheduler.</td>
</tr>
<tr>
<td><strong>-I call event</strong></td>
<td>Transfers or deletes the specified call event; must be used in combination with the -F option. You cannot use -I with -U. If the call event contains spaces, it must be enclosed in quotation marks.</td>
</tr>
<tr>
<td><strong>-J</strong></td>
<td>Locks call events that are transferred. This option is intended for telephony.</td>
</tr>
<tr>
<td><strong>-K</strong></td>
<td>Unlocks call events. (Locked call events cannot be transferred.) Before using this option, ensure no users are in the guide.</td>
</tr>
<tr>
<td><strong>-L listname</strong></td>
<td>Uses a selection list to transfer call events.</td>
</tr>
</tbody>
</table>
### Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>-M {n}</strong></td>
<td>Specifies the progress (milestone) of the process is to be displayed by the -V option every ( n ) records. The default is 100 if ( n ) is not specified.</td>
</tr>
<tr>
<td><strong>-N n</strong></td>
<td>Transfers a maximum of ( n ) number of call events. Used with the -F option to limit the number of call events to be transferred.</td>
</tr>
<tr>
<td><strong>-P</strong></td>
<td>Sends the output to the printer.</td>
</tr>
</tbody>
</table>
| **-Q** | Queries the user for the action to be taken on each call event processed. If the -M option is also specified, this option is ignored. One of the following prompts is displayed, depending on whether the destination is -D or -T:  
  
  **Delete xxx. Answer (Y/N/Q/A)**  
  **Transfer xxx. Answer (Y/N/Q/A)**  
  
  The following responses are valid:  
  
  **A** Yes to all.  
  **N** No.  
  **Q** Quit.  
  **Y** Yes. |
This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-S Displays process in full screen mode.

-V Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.

-W Increments the predictive dialer count stored in the PD.COUNT field in the STATUS file. This field tracks the number of times the call event is sent to the dialer.

**Notes**

This command must be invoked from the project that contains the queues.

The following are commonly used options:

- To transfer or delete all call events, use the -A option.
- To transfer or delete a single call event, use the -I option.
- To transfer or delete call events based on a selection list, use the -L option.
For information on transferring queues using the Transfer Call IDs command on the Queue Utilities tab, see Chapter 14.

**Examples**

Qtrans -F INCOMPLETE -T DEAD -L NEW.SALES
Qtrans -U -D
The RMREC command is a synonym for DELREC. For more information, see DELREC.
The RMTAPELOCKS command removes a tape lock from an I/O device.

Syntax

RMTAPELOCKS {device_id1 {device_id2...}}

device_idn    ID of I/O device that is locked.

Interactive Prompts

If no device ID is specified, the following prompt is displayed:

I/O device(s) name [must fill]:

Notes

An I/O device may become locked if a transfer fails or if the device is not set up properly.

Before using this command you should use the LSTAPELOCKS command to verify that the I/O device is locked.

This command can be invoked from the EDGE or project level.
The SECURE command assigns a specific security level to a command. Once secured, the command can only be used by users with a security level equal to or greater than the level assigned to the command.

### Syntax

SECURE \( \{nn \ command1 \{command2 \ldots \}\} \)

- **nn**  
  The security level. The highest level that can be assigned is 50.

- **commandn**  
  The names of the commands to secure. To change all commands, enter an asterisk (*).

### Interactive Prompts

If no properties are specified, the following prompts are displayed:

- New security level [default '0']:
- Commands [must fill]:

SECURE
Notes

The highest security that can be assigned is 50. The highest user security is 30. Thus, although you can assign a security higher than 30, no one can access that command for any reason, including changing the security level. If you need to remove a security level greater than 30, contact EDGE Customer Care.

If a user attempts to execute a command that has a higher security, the following message is displayed:

You do not have proper security level to run this command.

This command can be invoked from the EDGE or project level.
SELECT

The SELECT command produces an EQL statement.
The SETSTDERR command allows you to direct all stderr output (from your process only) to another destination.

**Syntax**

```
SETSTDERR {filename}
```

*filename* UNIX filename where the output is to be directed; can include the full path name.

**Interactive Prompts**

If no properties are specified, the following prompt is displayed:

```
File name [must fill]:
```

**Notes**

This command can be invoked from the EDGE or project level. The default destination is /dev/null.

If you are using EDGE Developer, this command affects the stderr only while you are in ACL. When you exit and return to EDGE Developer, stderr is reset to its previous value.
set_tracking

The set_tracking command enables record tracking for SmartMerge operations.

**Syntax**

set_tracking -f filename {-n | -y | -v value}

- **-f filename** The name of the database file to track.
- **-n** Disables all tracking and saving; equivalent to -v 0 option.
- **-y** Enables all tracking and saving; equivalent to -v 7 option.
- **-v value** Controls specific tracking and saving options. The following values are available:
  0   Disables all tracking and saving. This is equivalent to the -n option.
  1   Tracks when and where a record was last updated.
  2   Tracks whether a record has been deleted.
  4   Saves deleted records. (Currently not available.)

**Notes**

To enable a combination of options, add the values of the options you want. For example, to enable all options, add 1 + 2 + 4 for a -v
value of 7. To enable tracking of updated and deleted records only, add 1 + 2 for a -v value of 3. A -v value of 3 is recommended for SmartMerge operations. Saving deleted records cannot be enabled unless tracking deleted records is also enabled. Because of this requirement, -v values of 4 and 5 are illegal and cause the error message “Invalid flag combination.”

Note: When you disable tracking or saving for a file, any tracking information or deleted records that had been saved are removed.

Example

set_tracking -f CUST.LIST -v 3
SHOWSECURITY

The SHOWSECURITY command displays your security level, and if specified, the security level of ACL commands and whether or not the commands are password-protected.

Syntax

SHOWSECURITY {command1 {command2 ...}}

  commandn  The name of the command to display. To display all commands, enter an asterisk (*).

Interactive Prompts

If no properties are specified, the following prompt is displayed:

  Commands:

Notes

This command can be invoked from the EDGE or project level.
Example

```
[INS.22]--> SHOWSECURITY SETSTDERR
  User security level: 30
  SETSTDERR : level 00 not password protected

[INS.23]--> ⬤
```
The smart_merge command is used to import (receive) data from or export (send) data to external sources. The default is to perform an import; the -e option must be specified to perform an export.

**Syntax**

smart_merge -options instruction_ID

- **options**
  - All the options available for the edgeio command can also be used with the smart_merge command. The edgeio options are described in the section edgeio. In addition to the edgeio options, the following options are available with the smart_merge command:
  - `-x external` The name of the external source from which records are being imported or to which records are being exported, for example, PC1, IMAPC, or BOBSMAC. If no external source is specified, the default is EDGE.
smart_merge

You must have created a regular import or export instruction and a SmartMerge import or export instruction before using this command.

The smart_merge command can be invoked from the EDGE or project level.

**Example**

```
smart_merge -e -x PC1 EXP.RPT
```

**Notes**

-nd Forces a complete export of all selected records. This option applies to exporting only and is necessary only if record tracking of updates is enabled.

`instruction_ID` The name of the SmartMerge import or export instruction.
The TAPEFWD command moves the tape forward to the header record of the next file.

**Syntax**

TAPEFWD \{device_id1 \{device_id2 ... \}\}

\textit{device_idn} \quad \text{ID of I/O device to move forward.}

**Interactive Prompts**

If no device ID is specified, the following prompt is displayed:

I/O device(s) name [must fill]:

**Notes**

It the rewind pathname is specified in the I/O device, the tape is always rewound and this command has no effect.
TAPEREAD

The TAPEREAD command reads a tape and displays an octal dump of its data on the screen. You can use this, for example, to verify record data before adding a new file or modifying an existing file.

Syntax

TAPEREAD device_id

device_id ID of I/O device to read.

Interactive Prompts

If no device ID is specified, the following prompt is displayed:

I/O device(s) name [must fill]:

Notes

This command can be invoked from the EDGE or project level.
**TAPEREW**

The TAPEREW command rewinds tapes.

**Syntax**

TAPEREW {device_id1 {device_id2 ... }}

device_idn      ID of I/O device to rewind.

**Interactive Prompts**

If no device ID is specified, the following prompt is displayed:

I/O device(s) name [must fill]:

**Notes**

This command can be invoked from the EDGE or project level.
The UNARCHIVE command transfers archived modules from one project to another project.

Syntax

UNARCHIVE [-D device | -P pathname] {options}

-D device  Specifies the I/O device where the archived data is to be found. The name of the device must be the same as specified in the archive instruction. Either this or the -P option must be specified.

-P pathname  Specifies the pathname to the archived data. Either this or the -D option must be specified.

options  The following options can be specified:

-A  Prompts user to substitute a project name. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-B n  Uses a block size of n. The default is 4000.

-F  Displays information in full screen format. Should not be used from the task scheduler, because the system waits until prompts are acknowledged.
-I \textit{instruction} Verifies the instruction associated with archived data matches the specified \textit{instruction}. The unarchive process is canceled if the instruction does not match.

-N Scans the archive, but does not load. Does not alter system.

-Q Queries user to confirm the archive instruction before beginning to unarchive. Displays the archive instruction description and index. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

-S a Prompts user to selectively and manually restore files. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

The -S a option allows the user to override the archive settings to add/overwrite/replace/ignore; can be one of the following:

A Add.

I Ignore.

O Overwrite.

R Replace.
The UNARCHIVE command restores data previously archived. If data already exists, the archived data overwrites that data. For more information on archiving and unarchiving, see Chapter 9, Archive.

The UNARCHIVE command must be invoked from the EDGE level.

**Examples**

```
UNARCHIVE -P /EDGE/tmp/sales.arc -A -F
```

**Notes**

`-V` Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.
The vercont command invokes the EDGE version control process.

**Syntax**

vercont command -f filename { -r record ID | -a } options

*command* The action to invoke; can be one of the following:

- **-b** Bumps specified records.
- **-d** Compares differences between two versions. Must use two -v options.
- **-D** Causes a blank version of the record to be checked into SCCS.
- **-g** Gets a read-only version of the specified records.
- **-i** Checks in the specified records, if they have been checked out by the current user, or if they have never been checked in.
- **-li** Lists the records available to be checked in; that is, the currently checked out records and records that have never been checked in.
- **-lo** Lists the records available to be checked out; that is, the currently checked in records.
-lv Displays the versions for a specified record.
-o Checks out the specified records.
-pb Displays a report of both checked in and checked out records.
-pi Displays a report of checked in records.
-po Displays a report of checked out records.
-u Undoes a checkout.

-a All records. Not valid with -d or -lv commands.

-f filename The name of the version control area. This is required. The file can be any file in the project. However, only the files listed in Table 2-1 can be managed through EDGE Developer. For any other files you place in version control, you must continue to use the ACL vercont command.

-r record ID The names of the records to be processed. To specify all records, use -a. Either this option or -a is required for all commands except -li and -lo. These options have no effect on the -li and -lo commands.

options Information to use in processing. The options that can be used depend on the command; the following options are available:

-c comments Comments. Optional. Valid only with the -b and -i commands. Should be the last option on a line as all text following the -c is assumed to be comments.
-n Retrieves record in read-only mode. Optional. Valid only with the -o command.

-v version Version of the record. Valid with the -b, -d, and -o commands. Required when used with -b, indicates version to bump to. When used with -d, two -v options are required. Optional when used with -o and indicates version to check out; if not specified, latest version is checked out.

-F Bumps all records with a version less than the specified bump version, regardless of whether there are checked out records for the specified file. Optional.

If not specified, records are bumped only if the following are true:

- There must be no checked-out records in the specified file.
- There must be no records with a version higher than the specified version to bump to.

Valid only with the -b command with -a option.

-L Reports on only the latest version of the records. Optional. Valid only with -pb and -pi commands.
When you invoke EDGE version control from ACL, you have access to all the records and files in the level you entered ACL from. For example, if you entered from the EDGE level, you have access to all the files at the EDGE level. If you entered from a project, you have access to all files in that project. However, except for the version control areas set up by EDGE, EDGE is not aware of the status of these files under version control; they can be edited as usual by anyone with access to them.

For information on EDGE version control, see the section Version Control.

-\texttt{U user ID} Reports on only activity for the specified user. Optional. Valid only with -pb, -pi, and -po commands.
vfiles

The vfiles command executes the filecheck command on multiple files.

Syntax

vfiles -P projectname | -A {-F filename} {-S selection.list} {-M} {> logfile}

-P projectname  The name of the project that contains the files to be checked. If not specified, the -A option must be specified.
-A  Checks every file in every project and at the global level.
-F filename  Name of file to check.
-S selection.list  The name of the selection list that contains the filenames to check.
-M  Displays a summary instead of detailed listing.
> logfile  Redirects output to logfile.

Notes

If the -P option is specified and neither the -F nor the -S selection.list option is specified, every file in the project is checked.
This command can be very helpful, for example, when recovering from a system crash. If you suspect that any of your files have been affected by a crash, run vfiles to check the integrity of those files and act accordingly.

If you encounter any problems you are not familiar with, contact EDGE Customer Care for assistance.

If running the command against many files, you can use the > option to redirect the output to a file to view later.

Note: If this command is performed on many files, it may take a long time to run.

The vfiles command can be invoked from the EDGE or project level.
The XQUERY command executes Ad Hoc Reports or Ad Hoc Batch Reports.

Syntax

XQUERY \{options\} report.name \{time\}

The following options can be specified:

- **-A** Sends output to a slave printer (auxiliary port on terminal). This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

- **-B** Executes an ad hoc batch of reports.

- **-E printerID** The name of the printer to which the output is directed.

- **-L** Sends output to the default printer.

- **-N** Disables prompting for paging and completion.

- **-P** Prompts for properties before batch mode. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.
-T Sends output to the terminal. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.

**report.name** Specifies the name of the ad hoc report or batch report to be executed. To execute an ad hoc batch of reports, use the -B property.

**time** Specifies when a batch report is to be run. Must be specified with the -B option. You can enter the time in HH:MM format or as the number of seconds from the current time. If you enter the number of seconds, they are added to the current time and the time is truncated to the minute.

**Notes**

If you do not specify either the -E option or the -L option, you are prompted for the destination of the report.

**Example**

XQUERY -L LIST.PHONE
The `xrefall` command builds a cross-reference file.

**Syntax**

```
xrefall filename {-M n} {-V}
```

- `filename` The name of the file on which to build the cross-reference.
- `-M {n}` Displays the progress (milestone) of the process at every `n` records processed per file. The default is 100 if `n` is not specified. This option should not be used from the task scheduler, because it causes the system to wait until prompts are acknowledged.
- `-V` Displays all the messages returned by the process, scrolling the messages if necessary while the command is being processed.

**Example**

```
xrefall CUST.LIST -V
```

Messages similar to the following are displayed:

```
Clearing cross-reference file X_CUST.LIST
441 Cross Reference records built
```
Overview

This chapter discusses the following:

- Design Considerations for Files.
- File and Field Commands.
- Files.
- Fields.
- Cross-Reference Files.
- Records.
- Database Maintenance.

Files are logical groupings of data. The files you need depend on what your guide is designed to do. For example, if you are selling products, you might put data about your customers into a file called CUSTOMERS, data about your products into a file called PRODUCTS,
and data about the products purchased by customers into a third file called ORDERS.

Files are made up of records. Each record in turn is made up of fields. Every record in a file has exactly the same fields in exactly the same order as every other record in that file. The length of a field, however, can vary from record to record.

By defining the fields, you define the contents of the record. The only requirement is that every record must have a unique record ID by which to identify the record. For example, you may define a file to contain your customer records. You can create a unique account number to identify each customer, which can then be used as the record ID. You then determine what other information you need to store for your customers. This information dictates what fields should be defined for your record.

Once you define these fields, you have defined how records in your customer file look. See Figure 11-1.

Note: In addition to the files you create, EDGE creates its own set of files (called EDGE Files) whenever you create a new project.
### CUSTOMER File

<table>
<thead>
<tr>
<th>Fields</th>
<th>Name</th>
<th>Address</th>
<th>ZIP</th>
</tr>
</thead>
<tbody>
<tr>
<td>cust001</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cust002</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cust003</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cust004</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cust005</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cust006</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 11-1. File, Fields, and Records**
Design Considerations for Files

The following is a list of things to consider before designing your files:

- Decide where the files are needed.
- Decide on the database under which each file is to be created.
- Determine the fields needed for each file and the characteristics for each field.
- Determine how records are to be added to each file.

Deciding Where Files are Needed

You can create files on two levels:

- EDGE Level.
- Project Level.

Files created at the EDGE level are referred to as global files and are available to all projects. Files created at the project level are referred to as project files and are available only to the project in which they are created.

Files that can be used by several projects should be created at the EDGE level. For example, if several projects use the same customer list, one customer file can be created at the EDGE level. On the other
### Deciding on the Database

EDGE files reside in databases. The default database is called c-tree. Optional modules are available that allow you to create files using supported EDGE databases.

For more information on optional databases, see the EDGE 7.11 DataBase Link Reference Manual.

### Determining the Fields

Files consist of records, which in turn consist of fields. Each field in a record represents one piece of data. For example, in a customer file, you might have fields for the first name, last name, address, postal code, and phone number.

The following are some points to consider:

- What are the characteristics of the field? For example, is it a date or a number? If it contains letters, should they be in uppercase or lowercase?
- Is the field to be used for data entry? If so, how is the data to be validated?
- How should the field be formatted for display?
- Will the data be used in more than one way? If so, should a synonym field be defined?
How will a record be identified? What are the characteristics of the record ID?

Will you need alternative ways to access the file? You can define alternative keys, called secondary keys. The values in secondary keys do not have to be unique. (Secondary keys currently can be used with the GET-KEY verb.)

Do your agents need the ability to look up records by cross-referencing on data?

Determining How to Add Records

The way records are added depend on the individual situation. You might use one or more of the following:

- Records can be added individually through logic flows in guides.
- You can import records in batches from external sources.
- You can create records in batches using tasks.
File and Field Commands

The Files/Fields tab contains all file and field commands such as add, modify, and delete. This is available at the EDGE and project level.

To display the Files/Fields tabbed page, select the following:

EDGE or project / Database area

A tabbed page similar to the following is displayed:
### Commands

**Add Field**
- Creates a new field for the selected file.

**Assign Field Positions**
- Assigns each field a position in the record and builds the secondary key index for the selected file.

**Build Cross-Reference File**
- Builds the cross-reference data from the file records, based on the cross-reference file definition.

**Copy Fields**
- Copies fields defined at the EDGE or project level to any other file.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Redefine Cross Reference File</strong></td>
<td>Creates or modifies a cross-reference file definition for the selected file.</td>
</tr>
<tr>
<td><strong>Delete Cross-Reference File</strong></td>
<td>Deletes the cross-reference file for the selected file.</td>
</tr>
<tr>
<td><strong>Delete Fields</strong></td>
<td>Deletes a field from the selected file.</td>
</tr>
<tr>
<td><strong>Delete Records</strong></td>
<td>Deletes records from the selected file.</td>
</tr>
<tr>
<td><strong>Modify Field</strong></td>
<td>Modifies the selected field.</td>
</tr>
<tr>
<td><strong>Show Guide Files Only</strong></td>
<td>Specifies which files are available to modify; can be one of the following:</td>
</tr>
<tr>
<td><strong>Show Guide Show All Files</strong></td>
<td>Displays only files that are defined in Guide File Access.</td>
</tr>
</tbody>
</table>

This command is available only at the project level.
Files

Creating Files

You must create the file at the level at which you intend to use it: EDGE Level or Project Level.

Whether creating a new file or modifying an existing file, the following tabbed pages define the file:

- **File Information**: Defines the file. The properties available differ based on whether the file is being created at the EDGE Level or Project Level.
- **File Specs**: This tab is only available for files that are linked to relational databases.
- **Index Specs**: This tab is only available for files that are linked to relational databases.
- **Hints**: This tab is only available for files that are linked to relational databases.

For information on linking files to relational databases, see the EDGE 7.11 DataBase Link Reference Manual.

**EDGE Level**

To create a global file at the EDGE level, select the following:

- EDGE / Database area / Files/Fields tab
When you add a file, a page similar to the following is displayed:

Properties:

- **Database device**: The name of the database device to use in building this file. All defined database devices are listed. For information on creating a database device, see the EDGE 7.11 DataBase Link Reference Manual.

The following database devices are standard:
### Properties: (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Ctree</strong></td>
<td>The c-tree database, which is supplied with EDGE.</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>This is currently set to the c-tree database.</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>The detailed description of the file. This is an optional field.</td>
</tr>
<tr>
<td><strong>File name</strong></td>
<td>The name of the file.</td>
</tr>
<tr>
<td><strong>Project access</strong></td>
<td>The list of available projects in EDGE. To allow all projects access to the file, select the item (All).</td>
</tr>
<tr>
<td><strong>Selected projects</strong></td>
<td>The projects that have been selected to have access to this file.</td>
</tr>
<tr>
<td></td>
<td>To select a project, highlight the desired project and drag it to the selected projects area. To delete a project from the selected projects area, highlight the project and press the DELETE key.</td>
</tr>
</tbody>
</table>
**Project Level**

To create a file at the project level, select the following:

project / Database area / Files/Fields tab

When you add a file, a page similar to the following is displayed:
Properties:

Database device

The name of the database device to use in building this file. All defined database devices are listed. For information on creating a database device, see the EDGE 7.11 DataBase Link Reference Manual.

The following database devices are standard:

- Ctree The c-tree database, which is supplied with EDGE.
Copying a File

To copy a file, use the Drag and Drop method.

Deleting a File

To delete a file, from the Database area click the file to delete, then select the Delete command. You are asked to verify the deletion:

To cancel the deletion, click Cancel. To delete the file, click OK.

Properties: (continued)

- **Default**
  - This is currently set to the c-tree database.

- **Description**
  - The detailed description of the file. This is an optional property.

- **File name**
  - The name of the file.
**Fields**

Fields define the data in a file. (Before you can add fields for a file, you must create the file.) If using c-tree as your database, a maximum of 2100 fields can be defined. If using a database other than c-tree, the database may have a smaller maximum.

To create a field for a file, select the file or any other field in the file from the Files/Fields tab, then select the Add Field command.

To display the fields that have been created for a file, expand the list by clicking the plus sign (+) associated with the file on the Files/Fields tab. Once fields are displayed, they can be modified. To modify a field, select the field from the Files/Fields tab, then select the Modify Field command.

Whether creating a new field or modifying an existing field, the following tabbed pages define the field:

- **General Tab.** Used to define the basic characteristics of a field.
- **Validation Tab.** Used to define the field validation settings. This tab is available for all field types except Relational.
- **Assign Relationship Tab.** Used to define the relation for Relational fields. This tab is only available for relational fields.
The following tabs are only available for files that are linked to relational databases. For more information, see the EDGE 7.11 DataBase Link Reference Manual.

- Index specs
- Virtual
- Database Info
- Constraints

**General Tab**

The General tab specifies the characteristics of a field. To access the General tab, select the following:

EDGE or project / Database area / Files/Fields tab

When you use the Add field command, a page similar to the following is displayed:
Properties:

- **Column name**: Used with relational databases. For more information, see your DataBase Link optional module documentation.
- **Data type**: The type of data the field can contain; can be one of the following:
  - **Alpha-numeric**: Can contain any combination of characters including numbers, letters, and symbols such as a percent, decimal, or dash.
Properties: (continued)

This is the default data type.

Date Can contain only valid dates. For example, you cannot input February 30 into a date field. Data is stored in EDGE internal date format; that is, the number of days since December 31, 1967.

Note: The minimum date that can be entered in EDGE is 01/01/100.

Numeric Can contain only digits and one decimal point.

Time Can contain only valid times, consisting of digits, colons (:), and the characters AM or PM; all other characters are ignored.

Data is stored in EDGE internal time format; that is, the number of seconds since midnight.

The data type provides some validation checking. For example, if you specify a numeric field, only digits and a decimal point can be entered for that field. For more information about validation checking, see Validation Tab.

Once a field has been saved, the data type cannot be changed.
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Field name</td>
<td>The name of the field. Once a field has been saved, the field name cannot be changed.</td>
</tr>
<tr>
<td>Field position</td>
<td>The order of the field in the record. This property is display only.</td>
</tr>
<tr>
<td></td>
<td>This property is blank until field positions have been assigned. You can specify field positions for primary fields only; record ID fields always have the field position zero (0), and other field types do not have field positions. For more information on assigning field positions, see Assign Field Positions.</td>
</tr>
<tr>
<td></td>
<td>Once a field position is defined, it cannot be changed. In addition, if the field is in a file other than a c-tree file, once the field position has been assigned, the field definition can no longer be deleted.</td>
</tr>
<tr>
<td>File name</td>
<td>The name of the file for the field.</td>
</tr>
<tr>
<td></td>
<td>This property is display only.</td>
</tr>
<tr>
<td>Formatting</td>
<td>The characteristics for displaying the data in the field. The available formatting options vary by data type.</td>
</tr>
<tr>
<td></td>
<td>With an input field, the data entered by the agent is displayed according to field formatting. How the data is written to the record, however, is determined by the properties in guide options. In default configuration, data is stored as it was typed regardless of field formatting.</td>
</tr>
</tbody>
</table>
Properties: (continued)

For example, if an agent types Ca into an alphanumeric field formatted as uppercase, the data is displayed as CA, but, by default, it is written to the record as Ca.

The data may also be stored using the Data entry case conversion property located on the Guide Options 1 tab.

The following formatting options are available for alphanumeric data:

- **Capitalize**: The first letter of each word is capitalized. If the word begins with a non-letter character, that character is ignored. For example, 4ever is redisplayed as 4Ever.
- **Lower case**: All letters are displayed in lowercase.
- **None**: Data is displayed exactly as it is entered. This is the default.
- **UPPER CASE**: All letters are displayed in uppercase.

The following formatting options are available for dates:

- **MMM DD YY**: The date is displayed with a three-character month abbreviation, the day of the month, and the final two digits of the year.
Properties: (continued)

**MM/DD/YY**  The date is displayed in numeric format, with the month, day, and two digits of the year separated by the character specified in the country code for your site.

**MM-DD-YY**  The date is displayed in numeric format, with the month, day, and two digits of the year separated by dashes ( - ).

**MMM DD YYYY**  The date is displayed with the three-character month abbreviation, the day of the month, and the full year.

This is the default.

**MM/DD/YYYY**  The date is displayed in numeric format, with the month, day, and four digits of the year separated by the character specified in the country code for your site.

If a two-digit year is specified, the century is determined by the setting of the **EDGE base year** To specify a year in another century, use a date format with a 4-digit year and enter the entire year. For example, when the current year is 1996, to specify the year 2004, this value must be entered in a date field with the year formatted to four digits.

The order of the month, day, and year depend on settings in the **Country Code** for your site.
The following formatting options are available for numeric data:

- **$ and comma**
  The value is displayed using the currency symbol and thousands separator.

- **$ only**
  The value is displayed using the currency symbol.

- **Comma only**
  The value is displayed using the thousands separator.

- **None**
  The value is displayed exactly as it is entered. This is the default.

The currency and thousands separator symbols depend on settings in the Country Code for your site.

When Numeric data type is specified, an additional property is displayed, from which you can select the number of decimal places; specify a value zero to four.

If a number is entered with fewer decimal values than specified, zeros are appended to the right of the decimal characters, as required. For example, if two decimal places are specified, and 20.1 is entered, it is redisplayed as 20.10.

The following formatting options are available for time data:

- **AmPm**
  The value is displayed using 12-hour clock format:
### Properties: (continued)

<table>
<thead>
<tr>
<th>Time Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:00 AM</td>
<td>Midnight.</td>
</tr>
<tr>
<td>11:59 PM</td>
<td>One minute before midnight.</td>
</tr>
<tr>
<td>12:01 AM</td>
<td>One minute after midnight.</td>
</tr>
</tbody>
</table>

If this formatting is specified and a time less than or equal to 12:59 is entered without an AM or PM indicator, AM is assumed. If a time greater than 12:59 is entered, it is assumed to be military time and is converted to PM as appropriate.

#### AmPm & Secs

The value is displayed using 12-hour clock format and includes the seconds:

<table>
<thead>
<tr>
<th>Time Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12:00:00 AM</td>
<td>Midnight.</td>
</tr>
<tr>
<td>11:59:59 PM</td>
<td>One second before midnight.</td>
</tr>
<tr>
<td>12:00:01 AM</td>
<td>One second after midnight.</td>
</tr>
</tbody>
</table>

#### 24 Hour

The value is displayed using 24-hour clock format:

<table>
<thead>
<tr>
<th>Time Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00</td>
<td>Midnight.</td>
</tr>
<tr>
<td>23:59</td>
<td>One minute before midnight.</td>
</tr>
</tbody>
</table>
The value is displayed using 24-hour clock format, and includes the seconds:

<table>
<thead>
<tr>
<th>Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00</td>
<td>Midnight.</td>
</tr>
<tr>
<td>23:59:59</td>
<td>One second before midnight.</td>
</tr>
<tr>
<td>00:00:01</td>
<td>One second after midnight.</td>
</tr>
</tbody>
</table>

The time separator symbol depends on settings in the **Country Code** for your site.

The **Heading** property is used to provide an alternative column heading for ad hoc reports. The default column heading is the field name.
**Properties:** (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Justification</td>
<td>The way in which a field is to be justified for ad hoc reports; can be one of the following:</td>
</tr>
<tr>
<td>Left</td>
<td>Causes the data to be displayed from the left-most position of the field. If the data contains more characters than specified by the field length, the extra characters are wrapped to the next line, as necessary.</td>
</tr>
<tr>
<td>Right</td>
<td>Causes the data to be displayed from the right-most position of the field. If the data contains more characters than specified by the field length, the extra characters overwrite the field to the left.</td>
</tr>
</tbody>
</table>

The default is Left. Typically, numeric data is right-justified and alphanumeric data is left-justified.

Justification also affects sorting and selections. For this reason, it is recommended that dates, time, and numeric data be specified as right-justified.
Properties: (continued)

Length

The number of characters in the field. All characters are counted; for example, if the field is numeric with two decimal places, include one character for the decimal point.

This property determines the number of characters that are displayed. For example, for Ad Hoc reporting, the length property determines the width of the column.

When a field is attached to a non-multiline input object on a screen, the length also establishes the total number of characters (width) that can be input by an agent in a guide. When the maximum number of characters is entered, the cursor automatically moves to the next field. If you do not want the cursor to move automatically, you can change the F_AUTO_RETURN environment variable. (For information on environment variables, see Appendix AB.)

When a field is attached to a multiline input object on a screen, the length establishes the width of the object only. The number of characters that can be entered is determined by the Storage width property.

For c-tree files, fields are not limited to a maximum number of characters. The number of characters a field can hold is only limited when being input from a guide screen; in that case only as many characters as defined in the length property can be input. If modifying data using verbs or by manually editing the record at ACL, there is no limit to the number of characters that can be input.
**Properties:**  (continued)

**Multi-valued**

The default is 10.

When checked, specifies that more than one value can be stored in this field.

For example, on an order entry form, the product number, quantity, and price might all be multivalued. Then, each item the customer orders causes a product number value to be added to the product number field, a quantity value to be added to the quantity field, and a price value to be added to the price field.

Values are referenced by using the field name, followed by a comma and the value position. For example, to access the third value in the PROD.NO field, reference it as follows:

```
PROD.NO,3
```

The value position can be specified using a number or a variable containing a number.

This property is optional.

**On-line help**

The text to be displayed during operations if the agent presses Help while the cursor is on this field. Allows up to four lines of text

You might use online help, for example, to display the type of data to be entered, the valid range of values, or an explanation of what the field is intended for.
**Properties: (continued)**

This property is optional.

**Of type**

This is assigned when you need to process information in an existing field in some way. For example, assume you define a primary field called NAME which contains a first name and a last name separated by a space. You can then define a relational field called FIRST.NAME that extracts the first name from the NAME field.

If Relational is specified for the Type property, the Of type property is displayed, which allows the relationship to be defined. The type can be one of the following:

- Advanced user defined.
- Arithmetic expression.
- File translate.
- Group extract.
- Text extract.

After you select a relationship, you can specify the exact relationship characteristics from the Assign Relationship tab. For more information, see Assign Relationship Tab.

**Protected**

When checked, specifies that the field is display only. This property is set only for fields in EDGE Files.

**Secondary key field**

When checked, specifies that the field is an alternative key to use to retrieve records.
The values in the secondary key fields associate the secondary key field with the record ID. This provides very fast access to the record. For example, if you want to retrieve records based on telephone numbers as well as the account number, you can specify the telephone number field as a secondary key. Then, when the agent enters a customer’s telephone number, the appropriate record can be retrieved quickly by looking at the secondary key to get the account number.

Secondary key fields are used with the GET-KEY verb.

Secondary keys can only be defined for primary fields.

You can define up to 20 secondary key fields per file.

For c-tree files, the secondary keys are kept in the data_x portion of the file. For more information, see Appendix AD.

As records are added to or deleted from your files, EDGE updates the data_x portion of the file.

If you add or delete a secondary key field, you must build the secondary key index by assigning field positions. For more information, see Assign Field Positions.

For relational database type files, see your DataBase documentation, which is available separately as an optional module.

For a multiline input object, this property determines the total number of characters that can be entered.
### Properties: (continued)

**Type**  
To allow an unlimited number of characters to be entered for the multiline input object, use the value zero or null.

For information on using this with relational database fields, see the EDGE 7.11 DataBase Link Reference Manual.

The type of field that is being defined; can be one of the following:

- **Primary**: Has a field position and defined characteristics. The default field type.
- **Record ID**: The primary key for a record; must contain unique data for each record, and must not be null. Record IDs are always field position 0.
- **Relational**: Processes data in another field for display in ad hoc reporting; cannot be used in logic, tasks, or on screens. For more information, see the section Assign Relationship Tab.
Validation Tab

The Validation tab specifies additional criteria to be used to validate the field during data entry. The following types of validation can be specified:

- File validation.
- Member of Set.
- Patterns.
- Range.

Properties:  (continued)

Synonym

Defines an alternative name for a field; used to specify different display characteristics for a field, to give a field a shorter name, or to provide different validation criteria. For example, you can define a numeric field to display right-justified for one report, then define a synonym for that field that displays left-justified for another report.

When specified, a To property appears so the field for which this is an alternative name can be selected. Synonym fields have the same field positions as the To field.

Once a field has been saved, the data type cannot be changed.
The validation is performed in the order shown in Figure 11-2.
**Figure 11-2. Order of Validation**
To access the Validation tab, select the following:

EDGE or project / Database area / Files/Fields tab

When you use the Add field command and select the Validation tab, a page similar to the following is displayed:
Properties:

Default value: The value to be assigned during operations to an input field object before it is displayed on a screen if the associated field is currently empty ("""). EDGE treats the default value as if it were obtained from the associated field.

To specify multiple values as the default if the field is multivalued, separate the values with semicolons; for example:
red;green;blue

Red is the first value, green the second, and blue the third.

The default value is validated against the field type during assembly (it is not validated by the Scan option). For example, if the field type is numeric, the default value must be numeric. If not, an assembly error occurs.

Error message The text to display if the agent enters a value for the field that does not meet the validation requirements. A different error message can be specified for each type of validation.

This text is displayed at the bottom of the agent’s screen and replaces the standard EDGE error message. For example, if the data the agent enters is validated with member of set and the data does not match, the EDGE error message is:

Not a member of set.

You can replace this with a more specific message, such as:

You must enter either a T or F.

You must specify the literal text; you cannot specify a field or variable.
Properties: (continued)

File validation

File validation uses the values in another file to verify the data or to provide a cross-reference capability. The following properties can be defined:

Field

The field in the record specified by the Record property, against which the data is to be validated. To specify a specific value in a multi-valued field, append a comma and the value position after the field name; for example, CODES, 4.

If a field is specified and a record is not, the field entry is ignored.

This property is optional.

File

The name of the file to be used, against which the data is to be validated.

Locate

The type of validation; can be one of the following:

Yes

The entry must match a record ID in the specified file. If the Record property is specified, the entry must match a field in that record.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>The entry must not match a record ID in the specified file. If the Record property is specified, the entry must not match any field in that record.</td>
</tr>
<tr>
<td>X-Ref</td>
<td>Allows the agent to press <code>CTRL+X</code> and query the cross-reference file. If there is no cross-reference file for the specified file, this property is ignored. If the Locate value is X-Ref, the validation is against the record in the specified file and the validation must be true in order to be valid.</td>
</tr>
<tr>
<td>Record</td>
<td>Specifies the name of the record to be searched. For example, if you want to search the record called MAY.PROD, enter the characters MAY.PROD. If the Field property is also specified, only that field is searched in the specified record. If the Field property is not specified, all fields in the record are searched. If you specify the Locate property as X-Ref, the Record property is ignored. This property is optional.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

If the data does not meet the criteria, the specified error message, if any, is displayed. If the Error message property is empty, the following EDGE error message is displayed:

```
NOT FOUND IN FILE
```

#### Member of Set

The values used to validate the entry. To create this list, enter one member on each line. Type the member and press ENTER, then repeat for each member. For example, to validate a Yes response, you can specify the following as a set:

- YES
- Yes
- yes
- Y
- y

If the data does not meet the criteria, the specified error message, if any, is displayed to the agent. If the Error message property is empty, the following EDGE error message is displayed:

```
NOT A MEMBER OF SET
```

#### Patterns

The patterns used to validate the entry. For a list of the valid pattern codes, see Table 11-1.
The pattern codes can be combined to form exactly the pattern you need. To validate, the entry must contain exactly the number of characters specified in the pattern and of the exact type in the exact order. For example, the following pattern can be used to validate a social security number:

3N"-"2N"-"4N

If the data does not meet the criteria, the specified error message, if any, is displayed to the agent. If the Error message property is empty, the following EDGE error message is displayed:

DOES NOT MATCH PATTERN

The Range validation is used to verify that data is within a specified range. The range can be either numeric or alphabetic. For example, to test that a number is between 1 and 100, you can specify the range as from 1 to 100. Both 1 and 100 would fall within the range.

From     Specifies the starting number or letter.
To       Specifies the ending number or letter.
Ranges for dates and times can be specified in external or internal format, depending on the setting of the Enhanced range validation property in guide options. If the property is checked, enter the ranges using external format; for example, from June 1, 1998, to June 30, 1998. If the property is cleared, enter the ranges using internal format; for example, from 10016 to 10045. (The agent enters the date or time in external format, regardless of the setting; EDGE converts the data if necessary.)

If a time range is specified, be sure it does not span midnight. EDGE converts all time to the number of seconds since midnight, so a range from 8:30 p.m. to 2:30 a.m., for example, would always fail.

If the data does not meet the criteria, the specified error message, if any, is displayed to the agent. If the Error message property is empty, the following EDGE error message is displayed:

```
OUT OF RANGE
```

When checked, specifies that data is required for the field when it is associated with an object on the screen. The agent does not have to select the object for this to have effect; can be one of the following:

- **None** No data is required.
Properties: (continued)

Upon exiting object

Data must be supplied to the object associated with this field before the agent can leave the screen. In addition, if the agent selects an object associated with this field, data must be supplied before the agent can leave the object.

Upon exiting screen

Data must be supplied to the object associated with this field before the agent can leave the screen.

Required entry validation does not occur if the object is invisible, disabled, or both invisible and disabled on a screen. For example, assume there is an object PHONE defined as a required entry field on the screen FIRST. The object is invisible and disabled; therefore, the required entry validation does not occur.

Required for DB write/update

When checked, specifies that the field must have a value (not be empty) to write or update the record. Checking this property places a not null constraint on the field. If the field is empty, the record is not written and an error is generated to the $DBERRNO field. This property can be used with any database type.

Unlike the other validation properties (for example, Member of set), this property is validated at the time a record is written, not when a value is entered on the screen.
Properties: (continued)

When a field is linked to a pre-existing RDB column, if the column has a not null constraint, this property is automatically checked.

Checking this property for fields in files with data is not recommended because the not null constraint may already be violated.

Table 11-1. Pattern Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nA</td>
<td>Tests for n alphabetic characters. Can also specify n As; that is, you can specify either 3A or AAA.</td>
</tr>
<tr>
<td>nN</td>
<td>Tests for n numeric characters. Can also specify n Ns; that is, you can specify either 3N or NNN.</td>
</tr>
<tr>
<td>nX</td>
<td>Tests for n characters of any value. Can also specify n Xs; that is, you can specify 3X or XXX.</td>
</tr>
<tr>
<td>&quot;string&quot;</td>
<td>Tests for exact characters specified between quotation marks. For example, to test that the entry starts with an asterisk followed by nine characters, you can define the pattern as &quot;*&quot;9X.</td>
</tr>
</tbody>
</table>

Assign Relationship Tab

The Assign Relationship tab specifies the relationship for relational type fields. This tab is only available for relational fields.
The properties available on the tab depend on the type of relationship specified in the Of type property on the General Tab. The following types of relationships are defined on the Assign Relationship tab:

- Advanced User-Defined.
- Arithmetic Expression.
- File Translate.
- Group Extract.
- Text Extract.

**Advanced User-Defined**

The Advanced user-defined relationship allows you to specify relationships based on processing codes. When you use the Add field command for an advanced user-defined relational file and select the Assign Relationship tab, a page similar to the following is displayed:
To specify more than one code for a property, press ENTER at the end of the line. The codes that can be used are described in Appendix AA.
**Arithmetic Expression**

The Arithmetic expression relationship allows you to form an arithmetic expression using existing fields. When you use the Add field command for an arithmetic expression relational file and select the Assign Relationship tab, a page similar to the following is displayed:
Properties:

Arithmetic expression

The desired expression. You can use any combination of literals, fields, variables, and the following arithmetic operators:

- + Adds two values.
- - Subtracts two values.
- * Multiplies two values.
- / Divides the first value by the second value.
For example, to add the values in field 1 and field 2, then multiply by field 3, use the following:

\[(\text{FIELD1} + \text{FIELD2}) \times \text{FIELD3}\]

**File Translate**

The File translate relationship allows you to access data in a second file and display it using Ad hoc reporting. When you use the Add field command for a file translate relational file and select the Assign Relationship tab, a page similar to the following is displayed:
Properties:

Use the data from field

The field in the current file that contains values that are the record IDs of the translation file.

of this file to read a record from file

The name of the translation file. This file’s record IDs correspond to the data in the specified field in the current file.

and extract the data from field

The field in the translation file that contains the desired data.
File translate can be used, for example, to translate the product numbers in your INVENTORY file to their corresponding product descriptions in the PRODUCT file.

**Group Extract**

The Group extract relationship extracts characters from a field, based on a known delimiter character in the field. When you use the Add field command for a group extract relational file and select the Assign Relationship tab, a page similar to the following is displayed:
Properties:

Use data from field  The name of the field from which to extract data.

Skip ___ groups  The number of delimited groups to skip. If you want the first group, skip 0 (zero) groups.

separated by character  The delimiting character. It can be any printable character.
For example, if the FULL.NAME field in your CUST.LIST file contains the last and first name of your customers separated by a comma, you can extract the last name by skipping zero groups, specifying the separator character as a comma, and extracting one group.

**Text Extract**

The Text Extract relationship extracts characters from a specified location in a field. When you use the Add field command for a text extract relational file and select the Assign Relationship tab, a page similar to the following is displayed:
**Properties:**

- Use the data from field
- Starting with character position
- extract __ characters
- The name of the field from which to extract the characters.
- The position of the first character to be extracted.
- The number of characters to extract.
<table>
<thead>
<tr>
<th>Assign Field Positions</th>
</tr>
</thead>
</table>

For example, if your telephone number field contains both the area code and the telephone number, you can extract the area code by specifying the starting position as 1 and extract 3 characters.

The Assign Field Positions command on the Files/Fields tab is used to assign field positions. This command is also used to build the secondary key index file when you change an existing field to a secondary key.

When the Assign Field Positions command is selected, a dialog box similar to the following is displayed:
Properties:

Assign field When clicked, assigns the selected field in the Unassigned field list to the field position specified in the Next available position property. This property is only available if there are fields that do not have assigned field positions.

Assigned fields The list of all primary fields in the file that currently have defined field positions, as well as their position numbers. The list is in position number order. The record ID (position 0) is not listed.

This property is display only.
Properties: (continued)

- **Automatic**: When clicked, automatically assigns field position numbers in alphabetical order of the field names. This command is only available if there are fields that do not have assigned field positions.

- **Next available position**: Shows the first unassigned position in the file. This is display only.

- **Unassigned fields**: The list of all primary fields in the file that do not currently have assigned field positions.

  To assign a field position, select the field from this list, specify the field position in the Next available position property, then click the Assign Field button. The default is the next available position. You cannot assign a position number that has been assigned previously.

**Note:** Field position has an impact on performance. It is recommended that if possible, fields that will be accessed more frequently be assigned lower-numbered positions.

**Exiting Assign Field Positions**

When the Assign field positions dialog is exited, the following prompt displays:
Selecting Yes causes the following actions:

- Assigns field positions, if any have been added.
- Builds the dictionary records.
- Updates secondary keys, if any.

If there are secondary keys, before updating the dictionary, EDGE displays the following warning:

**Copy Fields**

You can copy fields from files on the EDGE or project level to any other file.
The two ways to copy fields are:

**Drag and Drop method**
Copies a field definition to another file using the same field name. The field position is not assigned. If a field with that name currently exists in the file, you will receive an error.

Before the file can be used, field positions must be assigned and the file rebuilt.

**Copy Fields command**
Copies one or more field definitions and allows you to rename fields and to specify whether to keep or remove the assigned field position numbers.

For c-tree files, if field positions are assigned, the file is rebuilt when the command is exited. This means the file is ready to be used.

When the Copy Fields command is selected from the Files/Fields tab, a page similar to the following is displayed:
User Files and Fields

Properties:

Field names  The field names to be copied. Only highlighted fields will be copied.

To select more than one field, hold down the SHIFT or CONTROL key and click the left mouse button.

From file  The file where the fields to be copied are located.

From project  The location of the file containing the fields to be copied; can be a project or EDGE.
### Delete Fields

The Delete Fields command on the Files/Fields tab deletes specified field definitions and data in those fields. If the field is cross-referenced, the cross-reference data is also deleted.

When you click the Delete Fields command, a dialog box similar to the following is displayed:

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove primary field positions</td>
</tr>
<tr>
<td>If checked, removes the position number from the copied field. If not checked, previously assigned field position numbers are used.</td>
</tr>
<tr>
<td>(Field positions must be assigned and the file rebuilt before it can be used. For more information, see the section Assign Relationship Tab.)</td>
</tr>
<tr>
<td>Rename fields</td>
</tr>
<tr>
<td>If checked, you are prompted, as each field is copied, to change the default field name.</td>
</tr>
<tr>
<td>To file</td>
</tr>
<tr>
<td>The file to copy the fields to.</td>
</tr>
<tr>
<td>To project</td>
</tr>
<tr>
<td>The location of the file in which to copy the field; can be a project or EDGE.</td>
</tr>
</tbody>
</table>

**Delete Fields**
To delete a field, select the field, then click the Delete command. The following prompt is displayed:

**Properties:**

- **Delete**: When selected, any highlighted fields will be deleted.
  
  To select more than one field, hold down the **SHIFT** key and click the left mouse button.
  
- **File name**: The filename of the file from which the fields are being deleted.
  
  This property is display only.
  
To delete a field, select the field, then click the Delete command. The following prompt is displayed:
Click OK to delete the field or Cancel to return to the Database tab. If the field is a primary field, the data in the file for that field is also deleted; however, the field position in the record is not deleted. This ensures that fields that follow the deleted field still can be accessed by their field definitions.

If the field is a synonym or relational field, only the field definition is deleted.

If the specified file has been assigned guide access, after the deletion is complete, a full assembly of your guide may be required.

**List File Fields**

The List File Fields command provides a listing of the fields for the selected file. This command is found on the Files/Fields menu when the Database area is active.

The report is sorted by record position within field type.
### Display Columns

<table>
<thead>
<tr>
<th>Field ID</th>
<th>The name of the field.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rec Pos</td>
<td>The position of the field within the record.</td>
</tr>
<tr>
<td>FT</td>
<td>The field type; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>I          Record ID.</td>
</tr>
<tr>
<td></td>
<td>P          Primary.</td>
</tr>
<tr>
<td></td>
<td>R          Relational.</td>
</tr>
</tbody>
</table>
### Display Columns (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>Synonym</td>
</tr>
<tr>
<td>Sk</td>
<td>A Y in this column indicates the field is a secondary key.</td>
</tr>
<tr>
<td>Synonym Fld ID</td>
<td>The name of the primary field for which this is a synonym.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The type of data; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>D Date.</td>
</tr>
<tr>
<td></td>
<td>N Numeric.</td>
</tr>
<tr>
<td></td>
<td>T Time.</td>
</tr>
<tr>
<td></td>
<td>X Alphanumeric.</td>
</tr>
<tr>
<td>MV</td>
<td>Indicates if the field is multivalued; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>N Field is not multivalued.</td>
</tr>
<tr>
<td></td>
<td>Y Field is multivalued.</td>
</tr>
<tr>
<td>Just</td>
<td>The justification; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>L Field is left-justified.</td>
</tr>
<tr>
<td></td>
<td>R Field is right-justified.</td>
</tr>
<tr>
<td>Length</td>
<td>The length of the field, in characters.</td>
</tr>
<tr>
<td>Dec Pl</td>
<td>The number of decimal places.</td>
</tr>
</tbody>
</table>
Display Columns (continued)

Fmt
Indicates the type of formatting; the meaning of the code depends on the data type, as indicated in Table 11-2.

Report Heading
The heading for reports.

Range
The defined range for validation.

Patterns
The defined pattern for validation.

Table 11-2. Formatting Codes

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format Code</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>1</td>
<td>DD MMM YY.</td>
</tr>
<tr>
<td>D</td>
<td>2</td>
<td>MM/DD/YY.</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td>MM-DD-YY.</td>
</tr>
<tr>
<td>D</td>
<td>4</td>
<td>DD MMM YYYY.</td>
</tr>
<tr>
<td>N</td>
<td>1</td>
<td>None.</td>
</tr>
<tr>
<td>N</td>
<td>2</td>
<td>$ and comma.</td>
</tr>
<tr>
<td>N</td>
<td>3</td>
<td>$ only.</td>
</tr>
<tr>
<td>N</td>
<td>4</td>
<td>Comma only.</td>
</tr>
<tr>
<td>T</td>
<td>1</td>
<td>24-hour clock format.</td>
</tr>
<tr>
<td>T</td>
<td>2</td>
<td>AmPm.</td>
</tr>
<tr>
<td>T</td>
<td>3</td>
<td>Seconds.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Format Code</td>
<td>Meaning</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>T</td>
<td>4</td>
<td>AmPm and seconds.</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
<td>None.</td>
</tr>
<tr>
<td>X</td>
<td>2</td>
<td>UPPERCASE.</td>
</tr>
<tr>
<td>X</td>
<td>3</td>
<td>lowercase.</td>
</tr>
<tr>
<td>X</td>
<td>4</td>
<td>Capitalize.</td>
</tr>
</tbody>
</table>
Cross-Reference Files

Cross-referencing refers to retrieving records based on information in fields other than the record ID. For example, if you set up the Last Name field in your customer file as a cross-reference field, and an agent knows the customer’s last name, but not the account number, the cross-reference feature can be used to retrieve all records with the indicated last name. From that list, the appropriate record can be selected.

The data for the cross-reference fields is kept in an index-only file with the name X_file.name. For example, if you build a cross-reference file for your CUSTOMERS file, the cross-reference file is called X_CUSTOMERS.

The cross-reference file can be set up so that only records with exact matches are listed or so that records with similar sounds are listed. (Retrieving records based on similar sounds uses Soundex Codes, which are described in the next section.) Within a single cross-reference file, you can set up some fields for exact matches, some for soundex matches, and some for both, if desired.

The cross-reference feature is activated when the agent in operations presses CTRL-X or in EDGE Client, clicks the Lookup icon ( ).
As records are added to or deleted from your file, EDGE automatically updates the associated cross-reference file. EDGE also manages the search of the cross-reference file. After the agent makes a selection from the presented list, the specified record is read and objects updated with the information from the file. (This is equivalent to the agent entering the data into the field.)

**Soundex Codes**

Soundex codes provide a way of searching for similar sounding words. A soundex code is produced by converting the consonants in a word to numeric values, based on the sounds of the letters.

A soundex code consists of the first letter of the word, plus numeric values for the next three consonants or combination of consonants. Vowels and the consonants H, W, and Y are ignored when producing soundex codes. If there are fewer than three numeric values, zeros are appended to the right, as required.

More than one letter may reduce to the same numeric value. This means that words with a similar arrangement of consonants may have similar soundex codes, regardless of the spelling. For example, SMITHS, SMITHSON, and SANDS all reduce to the same soundex code: S532.

If two adjacent consonants reduce to the same numeric value, only one instance of the value is used.
The values used by EDGE to produce soundex codes are listed in Table 11-3.

Note: Currently, soundex codes are available only for American English sounds.

### Table 11-3. Soundex Values

<table>
<thead>
<tr>
<th>Value</th>
<th>Letter</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>A, E, I, O, U, H, W, Y</td>
</tr>
<tr>
<td>1</td>
<td>B, F, P, V</td>
</tr>
<tr>
<td>2</td>
<td>C, G, H, K, Q, S, X, Z</td>
</tr>
<tr>
<td>3</td>
<td>D, T</td>
</tr>
<tr>
<td>4</td>
<td>L</td>
</tr>
<tr>
<td>5</td>
<td>M, N</td>
</tr>
<tr>
<td>6</td>
<td>R</td>
</tr>
</tbody>
</table>

To create or modify a cross-reference file, select the file from the Database tab, then select (Re)Define Cross Reference File command from the Files/Fields tab. After a cross-reference file has been defined, the command is named Redefine Cross Reference File.
The (Re)Define Cross Reference File contains the following tabs:

<table>
<thead>
<tr>
<th>Tab</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File Information</strong></td>
<td>Defines the cross-reference file characteristics.</td>
</tr>
<tr>
<td><strong>File specs</strong></td>
<td>This tab is only available with relational databases.</td>
</tr>
<tr>
<td></td>
<td>For more information, see the DataBase Link optional module documentation.</td>
</tr>
<tr>
<td><strong>Index specs</strong></td>
<td>This tab is only available with relational databases.</td>
</tr>
<tr>
<td></td>
<td>For more information, see the DataBase Link optional module documentation.</td>
</tr>
</tbody>
</table>

**File Information**

When the (Re)Define Cross Reference File command is selected from the Files/Fields tab, a page similar to the following is displayed:
Properties:

Database type: The database type of the cross-reference file. This property cannot be changed once the cross-reference file is created.

Description: The description that was specified when the file was created. This property is display only.

Display fields: The names of the fields to be listed when the agent requests a cross-reference list.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File name</strong></td>
<td>The filename of the file for which the cross-reference is being defined. This property is display only.</td>
</tr>
<tr>
<td><strong>Minimum word length</strong></td>
<td>The minimum number of characters a word is to contain before it is used to build a cross-reference record or is considered for matching. For example, if you specify 4 as the minimum word length, words such as Inc or The will not be used to build a cross-reference record. However, a minimum length of 4 will also eliminate words and acronyms such as IMA.</td>
</tr>
<tr>
<td><strong>Regular fields</strong></td>
<td>The names of the fields to be used to build cross-reference records. Up to the first 64 characters in each word in the field is used to build a record, unless the word is below the minimum word length or is in the list of words to ignore. (Words are characters delimited by spaces or end of field marks.) The data supplied by the agent must be an exact match. (The agent can supply wildcards or partial matching criteria and still have a match. For more information on how the agent uses cross-referencing, see the EDGE 7.11 Client Reference Manual.)</td>
</tr>
<tr>
<td><strong>Soundex fields</strong></td>
<td>The names of the fields to be used to build cross-reference records using soundex codes. The data supplied by the agent must match a soundex code for there to be a match.</td>
</tr>
</tbody>
</table>
Words to ignore

The words that are not to be cross-referenced. These are generally words that are so common that a match is meaningless. For example, if you are cross-referencing on names of businesses, you may indicate the word Company is not to be cross-referenced. Words that are below the minimum word length are also excluded and do not need to be listed here.

Note: The same field can be specified as both a regular field and a soundex field, if desired.

To create the cross-reference data from the data in a file’s records, select the file from the Files/Fields tab, then select the Build Cross-reference command. As the cross-reference file is built, a screen similar to the following is displayed:
Delete Cross-Reference File

To delete a cross-reference file, select the file from the Files/Fields tab, then select the Delete Cross Reference command. The following prompt is displayed:

Build Cross-reference: File CUSTOMER

Data Records Processed: 30
Cross-Reference Records Processed: 12

Building Cross Reference file completed successfully. Press RETURN to continue.
To delete the cross-reference file, select OK; to cancel and return to the Files/Fields tab, select Cancel.
Records

Records can be added to your files through agent input using the guide you have designed, or you can import records from existing files into your EDGE files. You can also export records from EDGE files to other applications. For more information on importing and exporting files, see Chapter 13.

Delete Records

The Delete Records command deletes specified records from the current file. When records are deleted, the associated cross-reference records are also deleted. To delete records, select the following:

- Project or EDGE / Database area /Files/Fields tab, Delete Records command

A dialog box similar to the following is displayed:
Properties:

All records  Deletes all data records and clears the file. The field definitions are not affected.

Specific record IDs  Deletes the specified records. Enter the names of the records in the list box.

Using selection list  Deletes the records specified in the selection list. (for information on selection lists, see Chapter 7.) When this button is clicked, a box in which to enter the name of the selection list is displayed.
After the specified records have been deleted, a report similar to the following is displayed:

```
[ACL - Account: SOFTSELL port:_14 Type 'goto' to exit.]

Please Wait... Processing records...

"2" deleted.
Processing completed.
Total records processed = 1
Total deleted = 1
Total not found or locked = 0

Press ENTER to continue:
```
Database Maintenance

The Database area includes a module for maintaining EDGE and project level databases. To use with EDGE level databases, select the following:

   EDGE / Database area / Maintenance tab

To use with project level databases, select the following:

   Project / Database area / Maintenance tab

A page similar to the following is displayed:
Check for Control Characters

The commands on the Maintenance page can also be run from ACL.

The Check for Control Characters command checks for control character and eight-bit characters from a specified file and notifies you of any occurrences.

The report is similar to the following:

- n control characters found in record "ID"
- x records found with control characters
Press ENTER to continue:

The characters that are checked for are those with ASCII values in the following ranges:

0-27, 31, 127-251, and 255.

For a complete listing of ASCII characters, see Appendix Y. Also see the description of the ACL command `ctrlchar`.

**Remove Control Characters**

The Remove Control Characters command removes control character and eight-bit characters from a specified file and notifies you of any occurrences.

The characters that are checked for are those with ASCII values in the following ranges:

0-27, 31, 127-251, and 255.

For a complete listing of ASCII characters, see Appendix Y. Also see the description of the ACL command `ctrlchar`.

**Compact File**

The Compact File command eliminates fragmentation in an EDGE file. When files are frequently updated they can become fragmented. Also, when records are deleted from a file, the file is not reduced in physical size. The compact process reorganizes the data for more
efficient usage of disk space and recovers the space from deleted records.

Associated cross-reference files are not compacted.

**Caution!** You should not use Compact File on a file that is in use.

This feature is for c-tree files only.

**Before Starting**

The compact process creates a temporary file and copies all the records into this file. When the compaction is completed, the original file is deleted and the compacted file is renamed to the original file-name. Because the file is duplicated, be sure you have sufficient disk space before beginning the process. EDGE can calculate the required space, if desired. When you invoke the compact command, a message similar to the following is displayed:
As an additional safeguard, you can back up the file before the compact process. The following prompt is displayed:

Backup original files to tape before compaction? (y/n/q)[y]

If you respond y, you are prompted for the tape path:

Enter tape path ([q] to quit)
(<RETURN> for default xx/xx) =>

The following prompt is displayed:

Enter path to your tmp directory ([q] to quit)
Check File for Corruption

The file is then compacted.

Also see the description of ACL command compact.

The Check File for Corruption function checks files for corruption and fragmentation. This feature checks for corruption and determines if the file should be compacted. The information returned is not accurate if the file is in use. This feature is for c-tree files only.
Properties:

Filename in EDGE/tmp:
The file to which to send the output; can be any valid filename. This is available only if Send output to a file is checked.

Send output to a file
When checked, specifies to send the results to a file. If an output file is specified, the output is sent to a file rather than displayed.

Summary format
When checked, specifies to return a summary of the results; can be checked for a summary or not checked for a detailed result. A summary report is displayed in Figure 11-3. A detailed report is displayed in Figure 11-4.
Figure 11-3. Summary Format

The following information is returned:

- The percentage of fragmentation for the file.
- Total active nodes: the number of records counted based on the data.
- Total resource nodes: a c-tree generated number which is always 2.
- Total items: the number of records counted based on the index.
In addition to the information included in the summary format, the detail format includes the following:

- Total deleted nodes: unused nodes that c-tree maintains to be reused.
- Total b-tree nodes: nodes that c-tree maintains internally.
- Total space wasted due to records with bad header.
- Total space that may be restored after compaction.
• Current file size.
• Total space needed for this file.
• Total number of data records based on the index.

Also see the description of the ACL command filecheck.
EDGE Files and Fields

Overview

EDGE files are files that EDGE uses to record information about projects, such as call status. You can use the information in EDGE files in combination with your project files. You can query the EDGE files, and in many cases, you can directly write information into EDGE files. (This chapter describes EDGE files; for information on user files, see Chapter 11.)

EDGE files are created automatically when a project is created and are listed in Table 12-1.

Table 12-1. EDGE Files

<table>
<thead>
<tr>
<th>System Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HISTORY File</td>
<td>Contains call event history.</td>
</tr>
<tr>
<td>REPORTING File</td>
<td>Contains call information that is used for Sales Analysis reporting.</td>
</tr>
<tr>
<td>RESPONSES File</td>
<td>Saves date, time and queue information.</td>
</tr>
</tbody>
</table>
**Record IDs**

EDGE files use values referred to as call events as their record IDs. As shown in Figure 12-1, a call event has the following components:

- **Call ID**: A unique value that usually relates to your user files in some way. When you generate a call event, you can specify the value of the call ID or allow EDGE to assign the call ID.

- **Delimiter**: The default delimiter is a tilde (~).

- **Event index**: A sequential number assigned by EDGE.

---

**Table 12-1. EDGE Files (continued)**

<table>
<thead>
<tr>
<th>System Filename</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATUS File</td>
<td>Contains information related to the most recent activity of each call event.</td>
</tr>
<tr>
<td>STATUSID File</td>
<td>Keeps track of locked call IDs.</td>
</tr>
<tr>
<td>SYS File</td>
<td>Stores information about the call event while it is active.</td>
</tr>
<tr>
<td>TEMP File</td>
<td>Contains user-defined fields.</td>
</tr>
</tbody>
</table>
Figure 12-1. Components of a Call Event

The call event value can be generated by one of the following methods:

- The ASSIGN-NEW-CALL-ID verb, which generates a call event consisting of just a call ID.

- The CALL-EVENT verb, which generates a call event consisting of a call ID, a delimiter, and an event index number.

- Queue Seeding, which generates call events consisting of just the call IDs, unless the property Allow multiple call events per call ID is enabled, in which case it generates call events consisting of the call ID, a delimiter, and a sequential event index number for each subsequent instance of a call ID (instead of overwriting the call ID).

- Automatically by EDGE, if a call event has not been assigned (or a call event record read using the GET-CALL-RECORD verb), when a call is ended using the $END function.
After the call event value is generated, records are created in the EDGE files using that call event as the record ID.

We recommend you use your customer file record ID as the call ID component of the call event. For example, if your customer file uses an account number as its record ID, you could specify the customer account number when generating a call even.

*Note:* The **REPORTING File** does not use the call event as its record ID. It uses a combination of port number, date, and time as the key.

**Modifying EDGE Files**

Fields in the SYS file are used by EDGE to update corresponding fields in the HISTORY, REPORTING, and STATUS files. (For a list of SYS file fields that are saved to other files, see the description of the **SYS File**.)

To update these files, you can use logic to update the appropriate fields in the SYS file. Then, when the call is filed, the corresponding fields in the other files are updated.

You should not update other EDGE files directly except in cases where there is no corresponding field in the SYS file. For example, there is no **COMMENTS** field in the SYS file, so if you need to update the **COMMENTS** field in the STATUS file, you do so directly by copying into STATUS-COMMENTS.
If you update a field in an EDGE file that is also in the SYS file, that field is overwritten by the value in the SYS file when the call event is filed.

Do not add your own fields to the HISTORY, REPORTING, STATUS, and SYS files. During an upgrade these files are replaced. If you have added fields to an EDGE file, they will be lost.

**Note:** You can add fields to the RESPONSES and TEMP file; these files are not overwritten during an upgrade.

The SYS file is cleared at the end of a call (when routing to the $END function). If you need to save information from the SYS file that is not saved in an EDGE file, you must save it in one of your user files before the end of the call.
The HISTORY file maintains the running history of the call event. When a call event is filed, the HISTORY record for that call event is updated. By default, ten contacts per call event are stored. (You can change the default in Guide Options.)

All fields in the HISTORY file, except the field TCALLS, are multivalued.

- **CALL.ID** The CALL.ID field displays up to 15 characters of the CALLEVENT field.

- **CALLEVENT** The CALLEVENT field contains the call ID and event index. This field is the record ID for the HISTORY file.

- **CALLID** The CALLID field displays the call ID. This is a relational field where the information is extracted from the CALLEVENT field.

  This field can only be used in Ad Hoc Reports.

- **CBDATE** The CBDATE field is a synonym for SCHED.DATE.

- **CBTIME** The CBTIME field is a synonym for SCHED.TIME; however, the formatting of HH:DD PM is used; for example, 06:28 PM instead of 18:28.

- **COMMENTS** The COMMENTS field contains a history of agent comments. To store agent comments in the HISTORY file, copy the COMMENTS field from the STATUS file (using a logic flow).
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>The DATE field contains a history of the dates the call event was accessed. The dates are displayed in MM-DD-YY format.</td>
</tr>
<tr>
<td>END</td>
<td>The END field contains a history of the times the calls ended. The time is displayed in 24-hour format.</td>
</tr>
<tr>
<td>EVENTINDEX</td>
<td>The EVENTINDEX field displays the six-digit index number for this call record. This is a relational field where the information is extracted from the CALLEVENT field. This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td>LCONTACT</td>
<td>The LCONTACT field is not currently used by EDGE. This field can be used to track any user-defined data; data in this field is only overwritten when new data is specified for this field. To place data in this field, copy the specified data to this field (using a logic flow); the data is automatically written during end of call processing.</td>
</tr>
<tr>
<td>LSCREENS</td>
<td>The LSCREENS field contains a history of the names of the screens presented when the call event was accessed.</td>
</tr>
<tr>
<td>OPID</td>
<td>The OPID field contains a history of the user IDs of the agents who handled the call event.</td>
</tr>
<tr>
<td>PORT</td>
<td>The PORT field contains a history of the ports used by the agents who handled the call event.</td>
</tr>
<tr>
<td>QUEUE</td>
<td>The QUEUE field contains a history of the specific queues the call event was placed in, for example, CALL-BACK/OP1 or COMPLETED.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>RESULT</td>
<td>The RESULT field contains a history of the results that were assigned when the call event was filed.</td>
</tr>
<tr>
<td>S.D</td>
<td>The S.D field is a synonym for SCHED.DATE.</td>
</tr>
<tr>
<td>S.T</td>
<td>The S.T field is a synonym for SCHED.TIME.</td>
</tr>
<tr>
<td>SCHED.DATE</td>
<td>The SCHED.DATE field contains a history of the scheduled callback dates. The dates are displayed in MM-DD-YY format.</td>
</tr>
<tr>
<td>SCHED.TIME</td>
<td>The SCHED.TIME field contains a history of the scheduled callback times. The times are displayed in 24-hour format; for example, 18:28 for 6:28 PM.</td>
</tr>
<tr>
<td>START</td>
<td>The START field contains a history of the time the calls started. The times are displayed in 24-hour format.</td>
</tr>
<tr>
<td>TCALLS</td>
<td>The TCALLS field contains the total number of times this call event has been accessed. (This field is not multivalued).</td>
</tr>
</tbody>
</table>
The REPORTING file contains information used to generate reports for Sales Analysis.

The record ID of the REPORTING file consists of the agent's port number, and the time and date the call was filed. The components of the record ID are delimited by asterisks (*). This differs from other EDGE files, which use the call ID as the record ID. In the REPORTING file, the call ID is a field in the file.

The date and time used in the record IDs of the REPORTING file are not formatted; EDGE internal date and time are used. For example, the record ID of a call event filed on 20 July 1994 at 10:57 A.M. on port 7_0 is displayed as follows:

```
7_0*9698*39459
```

The following fields in the REPORTING file are multivalued:

- PRODUCTS
- AMOUNT
- QTYS

All other fields are single-valued.
To write records to the REPORTING file, enable the Gather Sales Analysis Data property in Project Options. To update the data, use the SR.TRANS verb in the end of guide logic.

To prevent the REPORTING file being updated, do one of the following:

- To prevent records being written to the REPORTING file for the entire guide, clear the Gather sales analysis data check box in project options.

- To prevent a record being written for the current call event, set the $SA.SAVE field in the SYS file to N. (The $SA.SAVE field is reset to Y after each call.)

- To prevent records being written for specific results, check the REPORTING file box in the Do NOT update section of the Result Definitions screen.

**AMOUNT** The AMOUNT field contains the total dollar amount by product type. This is a multivalued field.

**CALL.ID** The CALL.ID field contains the call event that generated the call.

**DATE** The DATE field displays the date the record was filed. The date is displayed in MM-DD-YY format. This is a relational field where the information is extracted from the ID field. This field can only be used in Ad Hoc Reports.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DURATION</td>
<td>The DURATION field contains the length of the call in seconds.</td>
</tr>
<tr>
<td>FIRST.CALL</td>
<td>The FIRST.CALL field indicates if this is the first time the call event was accessed; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>1 This is the first call for the call event.</td>
</tr>
<tr>
<td></td>
<td>0 The call event was previously accessed.</td>
</tr>
<tr>
<td>HOUR</td>
<td>The HOUR field contains the hour when the call ID was filed. The time is displayed in 24-hour format.</td>
</tr>
<tr>
<td>ID</td>
<td>The ID field is the record ID for the REPORTING file. It consists of the agent's port number and the time and date the call was filed, delimited by asterisks (*).</td>
</tr>
<tr>
<td>IN.OUT</td>
<td>The IN.OUT field indicates the direction of the call; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>1 The call was inbound.</td>
</tr>
<tr>
<td></td>
<td>0 The call was outbound.</td>
</tr>
<tr>
<td>PORT</td>
<td>The PORT field displays the port used by the agent who handled the call event. This is a relational field where the information is extracted from the ID field.</td>
</tr>
<tr>
<td></td>
<td>This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td>PRODUCTS</td>
<td>The PRODUCTS field contains the product IDs for all products ordered by the customer. The IDs must be specified by using the SR.TRANS verb. This is a multivalued field.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>QTYS</td>
<td>The QTYS field contains the total quantity of each product ordered. This is a multivalued field.</td>
</tr>
<tr>
<td>QUEUE</td>
<td>The QUEUE field contains the specific name of the queue the call event was placed in, for example, CALL-BACK/OP1 or COMPLETED.</td>
</tr>
<tr>
<td>RESULT</td>
<td>The RESULT field contains the name of the result that was set when the call was filed.</td>
</tr>
<tr>
<td>TIME</td>
<td>The TIME field displays the time the record was filed. The time is displayed in 24-hour format. This is a relational field where the information is extracted from the ID field.</td>
</tr>
<tr>
<td></td>
<td>This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td>TOTAL.AMT</td>
<td>The TOTAL.AMT field contains the total amount for all products purchased. This field is equivalent to the total of the values in the AMOUNT field.</td>
</tr>
<tr>
<td>TOTAL.QTY</td>
<td>The TOTAL.QTY field contains the total quantity of all the products ordered. This field is equivalent to the total of the values in the QTYS field.</td>
</tr>
<tr>
<td>USER</td>
<td>The USER field contains the user ID of the agent who handled the call.</td>
</tr>
</tbody>
</table>
### RESPONSES File

The RESPONSES file saves the date, time and queue information. Fields can be added to this file, and these fields can be populated during **Queue Seeding** or from **Logic Flows**.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DATE</strong></td>
<td>The *DATE field displays the date the call event was accessed. This is a relational field based on the DATE field of the STATUS file. This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td><strong>DISP</strong></td>
<td>The *DISP field displays the specific name of the latest queue the call event was placed in. This is a relational field based on the QUEUE field of the STATUS file. This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td><strong>END.TIME</strong></td>
<td>The *END.TIME field displays the time the call ended. This is a relational field based on the END field of the STATUS file. This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td><strong>START.TIME</strong></td>
<td>The *START.TIME field displays the time the call started. This is a relational field based on the START field of the STATUS file. This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td><strong>CALL.ID</strong></td>
<td>The CALL.ID field contains up to 15 characters of the CALLEVENT field.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>CALLEVENT</td>
<td>The CALLEVENT field contains the call ID and event index number. This field is the record ID for the RESPONSES file.</td>
</tr>
<tr>
<td>CALLID</td>
<td>The CALLID field displays the call ID for this call record. This is a relational field based on the CALLEVENT field.</td>
</tr>
<tr>
<td>EVENTINDEX</td>
<td>The EVENTINDEX field displays the six-digit event index for this record. This is a relational field based on the CALLEVENT field.</td>
</tr>
</tbody>
</table>

This field can only be used in [Ad Hoc Reports](#).
The STATUS file tracks the current status of all call events. The data stored in the STATUS file is information pertinent to the call, such as date and time, agent ID, and port number.

A record is created in the STATUS file when a new call event is created through logic flows or when a queue is seeded, if the record does not currently exist. A record is updated whenever the call event is accessed.

You can manually modify only the following STATUS file fields:

- ALTPHONE
- COMMENTS
- RCNT
- SCHED.DATE
- SCHED.TIME
- STARTINDEX
- TELNO
- TP.PRIO
Caution! Do not add your own user fields to the STATUS file. Each time a call event is accessed, its record in the STATUS file is formatted to EDGE's current requirements and any values in user-defined fields are deleted. During an upgrade, user-defined fields themselves, as well as the data, are deleted.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALTINDEX</td>
<td>The ALTINDEX field indicates the value position of the phone number in the ALTPHONE field that is currently at the dialer.</td>
</tr>
<tr>
<td></td>
<td>This field is used by telephony.</td>
</tr>
<tr>
<td>ALTPHONE</td>
<td>The ALTPHONE field contains the list of all phone numbers. This is a multivalued field.</td>
</tr>
<tr>
<td></td>
<td>This field can be populated using a task or logic to provide alternate phone numbers for telephony.</td>
</tr>
<tr>
<td></td>
<td>This field is used by telephony.</td>
</tr>
<tr>
<td>CALL.ID</td>
<td>The CALL.ID field contains the call ID and event index number.</td>
</tr>
<tr>
<td>CALLEVENT</td>
<td>The CALLEVENT field contains the call ID and event index number. This field is the record ID for the STATUS file.</td>
</tr>
<tr>
<td>CALLID</td>
<td>The CALLID field displays the call ID for this call record. This is a relational field where the information is extracted from the CALLEVENT field.</td>
</tr>
<tr>
<td></td>
<td>This field can only be used in Ad Hoc Reports.</td>
</tr>
</tbody>
</table>
CBDATE  The CBDATE field is a synonym for SCHED.DATE.

CBTIME  The CBTIME field is a synonym for SCHED.TIME.

CODE    The CODE field indicates the current status of this call record; can be one of the following:

A       The record is currently active.

D       The record is currently not active.

COMMENTS The COMMENTS field contains user-defined information. For example, it might contain a description of the call event.

The COMMENTS field is not automatically transferred to the HISTORY file. If you wish to keep the comments in the HISTORY file, you must copy them using a logic flow.

COUNTRY The COUNTRY field contains the Country Code for the site. This value can be specified as part of queue seeding instructions. If not specified in Queue Seeding, the value specified in the Site Information property sheet is used.

DATE    The DATE field contains the date the call event was accessed. The date is displayed in MM-DD-YY format.

DIALS   The DIALS field is a synonym for TP.COUNT.

This field is used by telephony.

DISP    The DISP field is a synonym for QUEUE.
EMSG  The EMSG field is for internal use.
END  The END field contains the end time of the call. The time is displayed in 24-hour format.
EVENTINDEX  The EVENTINDEX field displays the six-digit index number for this call record. This is a relational field where the information is extracted from the CALLEVENT field. This field can only be used in Ad Hoc Reports.
INDEX  The INDEX field contains the following information depending on the queue type:
  FIFO  Contains a randomly generated packet number.
  Time  Contains a packet number indicating the scheduled date and time. If rescheduling a call in a task by copying a new date and time into the SCHED.DATE and SCHED.TIME fields, null ("") must be copied into this field before running Qrebuild.
ITZC  The ITZC field is the numeric designation representing the time zone for this call event, based on Greenwich Mean Time (GMT). This is determined by area code. If the call event is generated by queue seeding, the value is automatically determined by EDGE. If the call event is created through logic, you can copy the telephone number to the TELNO field in the STATUS file before the call is ended and EDGE will determine the ITZC value.
You can use the value in this field, for example, to generate reports based on time zones, or create a selection list of call IDs within specific time zones for distribution to agents. For example, you can select all the records in time zone 7 (Chicago, Dallas) or zone 6 (New York, Miami).

A complete listing of GMT zones is presented on the Country Code Definition property sheet. When your site is configured, information about your site's zone is determined from these choices.

**LASTSCREEN**

The LASTSCREEN field contains the name of the last screen presented when the call event was accessed.

**OPID**

The OPID field contains the user ID of the agent who handled the call event.

**PD.COUNT**

The PD.COUNT field is a synonym for TP.COUNT. This field is used by telephony.

**PORT**

The PORT field contains the name of the port used by the agent who handled the call event.

**QNAME**

The QNAME field contains the general ID of the queue the call event was placed in, for example, CALLBACK/U or COMPLETED.

**QUEUE**

The QUEUE field contains the specific ID of the queue the call event was placed in, for example, CALLBACK/OP1 or COMPLETED.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QUEUE.TYPE</td>
<td>The QUEUE.TYPE indicates the type of queue this call event was placed in; can be one of the following: 1 Time ordered. 2 FIFO (first in, first out). This field can only be used in Ad Hoc Reports.</td>
</tr>
<tr>
<td>RCNT</td>
<td>The RCNT field contains the recall count for the call event. This field indicates the number of times the call event has been accessed. You can reset the RCNT field to zero as part of defining a result or through a logic flow. This can be used, for example, in a logic flow to restrict the number of attempts on a call ID.</td>
</tr>
<tr>
<td>RESULT</td>
<td>The RESULT field contains the name of the result that was assigned when the call event was filed. Before a call event can be filed, you must have specified a result.</td>
</tr>
<tr>
<td>SCHED.DATE</td>
<td>The SCHED.DATE field contains the scheduled callback date. The date is displayed in MM-DD-YY format.</td>
</tr>
<tr>
<td>SCHED.TIME</td>
<td>The SCHED.TIME field contains the scheduled callback time. The time is displayed in 24-hour format.</td>
</tr>
<tr>
<td>SRC.QUEUE</td>
<td>The SRC.QUEUE contains the name of the first queue this call event was assigned to. If the call event was created by queue seeding, that queue is placed in this field. If the call event was created by logic, the name of the default queue specified in Guide Options is placed in this field.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>START</td>
<td>The START field contains the time the call started. The time is displayed in 24-hour format.</td>
</tr>
<tr>
<td>STARTINDEX</td>
<td>The STARTINDEX field contains the value position of the first phone number to be dialed from the ALT-PHONE field, using the telephony feature enhanced alternate phone number selection. This field is used by telephony.</td>
</tr>
<tr>
<td>TCALLS</td>
<td>The TCALLS field contains the total number of times this call event has been accessed.</td>
</tr>
<tr>
<td>TELNO</td>
<td>The TELNO field contains the telephone number used to access the customer associated with this call event. The value for this field is automatically filled in if the call event is created by queue seeding. If you generate a call event through logic, you must copy the telephone number from the customer file to this field. This field is used to determine the time zones in cases where there can be more than one time zone for a country code.</td>
</tr>
<tr>
<td>TP.COUNT</td>
<td>The TP.COUNT field contains the total number of dial attempts made by the predictive dialer, including those resulting in a non-connect status. This field is used by telephony.</td>
</tr>
</tbody>
</table>
You must define the predictive dialer codes that increment this count. If a code is not explicitly defined, and that code is returned by the predictive dialer, the TP.COUNT field is not incremented. (For more information on predictive dialing, see the telephony documentation.)

**TP.LCALL**

The TP.LCALL field contains the number of times this call event was dialed by the predictive dialer prior to being returned as a non-connect or connect.

This field is used by telephony.

**TP.LDATE**

The TP.LDATE field contains the date this call event was last returned by the predictive dialer as resulting in a non-connect.

This field is used by telephony.

**TP.LTIME**

The TP.LTIME field contains the time this call event was last returned by the predictive dialer as resulting in a non-connect.

This field is used by telephony.

**TP.PRIOR**

The TP.PRIOR field contains the priority assigned to an InterVoice predictive dialer call.

This field is used by telephony.

**TZC**

Reserved.

**USES DST**

Reserved.
The STATUSID file keeps track of locked call IDs.

If necessary, you can unlock a call ID by using the Rebuild Queues command on the Queues Utility tab or by using the Qrebuild utility with the -K option.

You can also unlock call IDs by deleting the corresponding record from the STATUSID file and changing the CODE field in the STATUS file to D. To clear the STATUS records, from the Database area, select the STATUS file, then click the Delete Records command.

It may be necessary to unlock call IDs, for example, if your system crashes while the call ID is locked.

Note: This file does not keep track of individual call events that may be locked.

| CALLID | The CALLID field contains the call ID of this record. This field is the record ID for the STATUS file. |
| OPID | The OPID field contains the ID of the agent who locked the call ID. |
| CODE | The CODE field always contains the letter A indicating the call is active. |
**SYS File**

The SYS file holds data temporarily while the call is in progress. Fields in the SYS file are prefixed with dollar signs ($).

At the end of the call, the data shown in Table 12-2 is automatically stored in the HISTORY and STATUS files, then the file is cleared. If you want to store data from fields other than those listed in the table, you must copy the data to fields in a user-defined file before the end of the call.

**Note:** If specified in end of guide logic, the value of $QUEUE is ignored; that is, it is not written to the STATUS or HISTORY file. $QUEUE can be populated in any logic flow other than one designated as end of guide logic.

**Table 12-2. SYS File Fields Copied to HISTORY and STATUS Files**

<table>
<thead>
<tr>
<th>SYS Fields</th>
<th>STATUS Fields</th>
<th>HISTORY Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CALLEVENT</td>
<td>CALLEVENT</td>
<td>CALLEVENT</td>
</tr>
<tr>
<td>$COUNTRY</td>
<td>COUNTRY</td>
<td>COUNTRY</td>
</tr>
<tr>
<td>$OPID</td>
<td>OPID</td>
<td>OPID</td>
</tr>
<tr>
<td>$PORT</td>
<td>PORT</td>
<td>PORT</td>
</tr>
</tbody>
</table>
Table 12-2. SYS File Fields Copied to HISTORY and STATUS Files (continued)

<table>
<thead>
<tr>
<th>SYS Fields</th>
<th>STATUS Fields</th>
<th>HISTORY Fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>$QUEUE</td>
<td>QUEUE</td>
<td>QUEUE</td>
</tr>
<tr>
<td>$RESULT</td>
<td>RESULT</td>
<td>RESULT</td>
</tr>
<tr>
<td>$TCBDATE</td>
<td>SCHED.DATE</td>
<td>SCHED.DATE</td>
</tr>
<tr>
<td>$TCBTIME</td>
<td>SCHED.TIME</td>
<td>SCHED.TIME</td>
</tr>
</tbody>
</table>

In the following descriptions, a Y in the User Update entry indicates you can modify the field directly:

$AUTODELAY

The $AUTODELAY field specifies the number of seconds to pause before performing the action specified by the $AUTOFKEY. During the pause, the Call Center screen is displayed with no commands enabled.

The default is 4 seconds.

User Update: Y

$AUTOFKEY

The $AUTOFKEY field specifies a Call Center command to perform when an agent routes to $END. EDGE automatically invokes the $AUTOEND function instead of $END.
When the agent exits the call, the $AUTOEND screen displays for the number of seconds specified by $AUTODELAY, then the specified command is performed. This is used in logic flows to streamline guide usage by requiring agents to press fewer keys.

For example, if at the end of each call, the agent makes the next available call in queue, specify 3 (Next Call) for this variable before ending the call. Thus, the agent does not have to select Next Call from the Call Center screen, but EDGE automatically selects the next call and routes the agent to the first outbound guide screen.

The $AUTOFKEY field can contain one of the following:

1  Receive Call command.
2  Make Call command.
3  Next Call command.
4  View Callbacks command.
6  View Monitor command.
8  Exit command.

If $AUTOFKEY contains an invalid value, $AUTOFKEY is cleared and the agent routes to the Call Center screen.

User Update: Y
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$AVAIL</td>
<td>The $AVAIL field is a synonym for $TP.AVAIL. This field is used by telephony.</td>
</tr>
<tr>
<td>$CALL.ID</td>
<td>The $CALL.ID field is a synonym for $CALLID.</td>
</tr>
<tr>
<td>$CALLEVENT</td>
<td>The $CALLEVENT field contains the call ID, delimiter, and event index number of the call event currently being processed.</td>
</tr>
<tr>
<td>$CALLID</td>
<td>The $CALLID field contains the call ID portion of the call event. This field is extracted from $CALLEVENT.</td>
</tr>
<tr>
<td>$CAMPAIGN</td>
<td>The $CAMPAIGN field is a synonym for $TP.ACDGRP. This field is used by telephony.</td>
</tr>
<tr>
<td>$CAMPAIGNID</td>
<td>The $CAMPAIGNID field is a synonym for $TP.ACDGRP.</td>
</tr>
<tr>
<td>$CEDELIM</td>
<td>The $CEDELIM field contains the call event delimiter.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$COUNTRY</td>
<td>The $COUNTRY field contains the country code for the site as defined in the section Set Up Site Information.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$CURR.DATE</td>
<td>The $CURR.DATE field contains the current system date. This field is updated each time the agent accesses a new screen. The date is displayed in MM-DD-YY format.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$CURR.TIME</td>
<td>The $CURR.TIME field contains the current system time. This field is updated each time the agent accesses a new screen. The time is displayed in 24-hour format.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$DATE</td>
<td>The $DATE field is a synonym for $CURR.DATE.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$DBERRNO</td>
<td>The $DBERRNO field is a multivalued field that contains an error code if a file operation fails as indicated by an N in the $READ.OK field. The data in this field is valid only if $READ.OK is N.</td>
</tr>
</tbody>
</table>
|         | $DBERRNO has the following values:
The status code. For a list of codes, see the EDGE 7.11 DataBase Link Reference Manual.

The database type; can be one of following:

1. c-tree
2. Oracle
3. Informix
4. Sybase
6. SQL Server

Reserved for future use.

Reserved for future use.

The number of rows processed.

Reserved for future use.

Offset of the error into the SQL statement.

ODBC state code (5 letters/digits).

User Update: N

$ENTRY

The $ENTRY field contains the current data that was entered for the object that has the focus, if that object is an input field.
If the agent presses return without entering any new data, the contents of $ENTRY depend on the settings of the SET_DENTRY_NULL environment variable, as follows:

0 (zero)  $ENTRY contains the contents of the field.

1 (one)   $ENTRY contains a null ("").

This field should be used only in logic that is attached to an object.

User Update: Y

$ERROR.MSG

The $ERROR.MSG field can be used to display a user-defined error message on the bottom of the agent's screen. No response is necessary from the agent.

To display a message using $ERROR.MSG, use a logic flow to copy the message into the field. Then attach the logic flow at the point you want it to be displayed. You do not have to define a place for the field on your screen. When the logic is executed, the message is automatically displayed when the logic flow is exited.

If $ERROR.MSG is set in a logic flow with a route specified, or in a logic flow that is attached to function key or hot key with a route specified, the route is canceled.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$EVENT</td>
<td>The $EVENT field contains the name of the event that was triggered.</td>
</tr>
<tr>
<td>$EVENTINDEX</td>
<td>The $EVENTINDEX field contains the call event index number for the $CALLEVENT.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$FIELDNAME</td>
<td>The $FIELDNAME field contains the name of the object that invoked the logic. For check boxes, command buttons, list boxes, and radio buttons, this is the name specified in the Value field property. For input fields, it is the name specified in the Field name property. This field contains the filename and the fieldname separated by a hyphen; for example, CUSTOMER-ACCOUNT. This field should be used only in a logic flow that is attached to an object.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$FIRSTNAME</td>
<td>The $FIRSTNAME field contains the first name of the agent, as specified on the First name property on the User Definition.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$FKEY</td>
<td>The $FKEY field contains the number of the last function key used by the agent.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
</tbody>
</table>
| $GEO   | The $GEO field indicates if EDGE Client is currently being used; can be one of the following:  
  Y   Yes; indicates EDGE Client is being used.  
  N   No; indicates EDGE Client is not being used.  
User Update: N  |
| $GROUP | The $GROUP field contains the group assignment of the agent as defined on the User Definition.  
User Update: N  |
| $HOURDIFF | The $HOURDIFF field contains a number representing the difference in hours between the agent’s time and the customer’s time. For example, if the customer’s time is one hour ahead, this field contains +1; if one hour behind, it contains -1.  
User Update: N  |
| $ID | The $ID field contains the ID of the current record on the task list. This field is used for task maintenance.  
User Update: N  |
| $INPCH | The $INPCH indicates if the call is in preferred calling hours; can be one of the following:  
  Y   Yes; the call is within preferred calling hours. For inbound calls, the field always contains a Y.  
User Update: N  |
N  No; the call is not within preferred calling hours.

Preferred calling hours are defined in Work Schedule.

User Update: N

$I0$  The $I0$ field indicates if the call is inbound or outbound; can be one of the following:

I  Inbound call.

O  Outbound call.

User Update: N

$ISDN\text{CALING}$  The $ISDN\text{CALING}$ field is a synonym for $TP.ANI$.

This field is used by telephony.

User Update: N

$ISDN\text{CALLED}$  The $ISDN\text{CALLED}$ field is a synonym for $TP.DNIS$.

This field is used by telephony.

User Update: N

$ISDN\text{OTHER}$  The $ISDN\text{OTHER}$ field is a synonym for $TP.DAT\text{A}$.

This field is used by telephony.

User Update: N

$ISDN\text{MODE}$  If Y, indicates that automated telephony is currently turned on.
$LASTCBDATE
The $LASTCBDATE field contains the date of the last callback as specified on the setup for the project. The field is displayed in MMM DD YY format.
User Update: N

$LASTNAME
The $LASTNAME field contains the last name of the agent as defined on the User Definition.
User Update: N

$LASTSCREEN
The $LASTSCREEN field contains the name of the screen the agent was on prior to the current screen. If the agent is on the first user screen of the guide, $LASTSCREEN is empty. If the agent routes to the screen she or he is currently on, $LASTSCREEN is not changed.
User Update: N

$LB.CLICK
The $LB.CLICK specifies why the list box object logic is being called; can be one of the following:

0  No click.
1  Single click.
2  Double click.

This field is only used with the list box object.
User Update: N
$LB.SELCHANGE  The $LB.SELCHANGE indicates if the selection has changed in the list box object; can be one of the following:

- null  Initial value.
- 0    No change has occurred.
- 1    Yes, the selection has changed.

This field is only used with the list box object.
User Update: N

$LCALLIDS  The $LCALLIDS field contains the list of locked Call IDs. (For information on locking call IDs, see the description of the CALL-EVENT verb.)
User Update: N

$LEVENTS  The $LEVENTS field contains the list of locked call events. (For information on locking call events, see the description of the CALL-EVENT verb.)
User Update: N

$LIVETEST  The $LIVETEST field indicates the mode the guide is being executed in; can be one of the following:

- L    Live mode.
- T    Test/training mode.

User Update: N
### $LOC.DIST
The $LOC.DIST field contains the distance from the customer’s location to the location found in the Locator search. This is a multivalued field and can have up to ten occurrences of up to three digits each.

This field is used by Locator.

User Update: N

### $LOC.FIELD1
The $LOC.FIELD1 field can be used to display information associated with the location found in the Locator search. This is a multivalued field.

This field is used by Locator.

User Update: N

### $LOC.FIELD2
The $LOC.FIELD2 field can be used to display information associated with the location found in the Locator search. This is a multivalued field.

This field is used by Locator.

User Update: N

### $LOC.FIELD3
The $LOC.FIELD3 field can be used to display information associated with the site located. This is a multivalued field.

This field is used by Locator.

User Update: N
$LOC.FIELD4 The $LOC.FIELD4 field can be used to display information associated with the location found in the Locator search. This is a multivalued field.

This field is used by Locator.

User Update: N

$LOC.IDS The $LOC.IDS field contains the record IDs of the location found in the Locator search. This is a multivalued field.

This field is used by Locator.

User Update: N

$LOCKED The $LOCKED field indicates if the record has been locked by another user; can be one of the following:

Y Yes; the record is locked.
N No; the record is not locked.
"" (null); no records have been read.

User Update: N

$LOGICERR The $LOGICERR field contains the error number for the runtime logic error, followed by appropriate arguments. For information on the errors, see the UNIX file /usr/include/errno.h. This is a multivalued field.

This field is used by the CALCULATE verb.
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>User Update</th>
<th>System Monitor Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>$LOGICLINE</td>
<td>The $LOGICLINE field contains the line number in the logic flow where the error occurred. This field is used by the <strong>CALCULATE</strong> verb.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>$LOGICNAME</td>
<td>The $LOGICNAME field contains the name of the logic flow in which the runtime error occurred. This is used by the <strong>CALCULATE</strong> verb.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>$MDACH</td>
<td>The $MDACH field contains the number of calls per hour.</td>
<td>N</td>
<td>This field is used by the Project Monitor.</td>
</tr>
<tr>
<td>$MDADH</td>
<td>The $MDADH field contains the average dollars per hour.</td>
<td>N</td>
<td>This field is used by the Project Monitor.</td>
</tr>
<tr>
<td>$MDATE</td>
<td>The $MDATE field contains the current date in MM-DD-YY format.</td>
<td>N</td>
<td>This field is used by the Project Monitor.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDAUH</td>
<td>The $MDAUH field contains the average units per hour.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDCLS</td>
<td>The $MDCLS field contains the total number of calls.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDCPH</td>
<td>The $MDCPH field contains the number of calls per hour.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDDOLS</td>
<td>The $MDDOLS field contains the total number of dollars.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDDPH</td>
<td>The $MDDPH field contains the dollars per hour.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDHRS</td>
<td>The $MDHRS field contains total number of hours.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>User Update</td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>$MDMSG</td>
<td>The $MDMSG field can be used for a message that you define.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDRAC</td>
<td>The $MDRAC field contains the average calls per hour by result group.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDRGN</td>
<td>The $MDRGN field contains the names of the record groups. This is a multivalued field.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDRTC</td>
<td>The $MDRTC field contains the total number of calls by record group.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDRYC</td>
<td>The $MDRYC field contains the calls per hour by record group for the agent requesting the report.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This field is used by the Project Monitor.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDTCS</td>
<td>The $MDTCS field contains the total number of calls. This field is used by the Project Monitor. User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDTHS</td>
<td>The $MDTHS field contains the total hours. This field is used by the Project Monitor. User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDTIME</td>
<td>The $MDTIME field contains the current time. This field is used by the Project Monitor. User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDUNITS</td>
<td>The $MDUNITS field contains the total number of units. This field is used by the Project Monitor. User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MDUPH</td>
<td>The $MDUPH field contains the number of units per hour. This field is used by the Project Monitor. User Update: N</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MMSG</td>
<td>Reserved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MSUBJ</td>
<td>Reserved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MTIME</td>
<td>Reserved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MUFROM</td>
<td>Reserved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$MUNAME</td>
<td>Reserved.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OBJECT</td>
<td>The $OBJECT field contains the name of the ActiveX object that caused an event to be triggered.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OLDQUEUE</td>
<td>The $OLDQUEUE field contains the name of the queue the call event was placed in the last time it was accessed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OLDRESULT</td>
<td>The $OLDRESULT field contains the name of the result specified for the call event the last time it was accessed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OPDEV</td>
<td>The $OPDEV field indicates the operational mode the user is in; can be one of the following:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>D Development mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>O Operations mode.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OPID</td>
<td>The $OPID field contains the user ID of the agent who handled the call event.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$OPNAME</td>
<td>The $OPNAME field contains the full name of the agent as specified in the Full name property on the User Definition.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>User Update</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td><code>$PCBDATE</code></td>
<td>The <code>$PCBDATE</code> field contains the callback date specified for the call event the last time it was accessed. The <code>$PCBDATE</code> is displayed in MMM DD YY format.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td><code>$PCBTIME</code></td>
<td>The <code>$PCBTIME</code> field contains the callback time specified for the call event the last time it was accessed. The <code>$PCBTIME</code> is displayed in HH:MM AMPM format.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td><code>$PORT</code></td>
<td>The <code>$PORT</code> field contains the name of the port used by the agent who handled the call event.</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td><code>$PREV.ENTRY</code></td>
<td>The <code>$PREV.ENTRY</code> field contains the data previously contained in the object with the current focus, if that object is an input field. This field has meaning only if the data in that object has changed. This field can be used only in individual field logic.</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td><code>$PROJECT</code></td>
<td>The <code>$PROJECT</code> field contains the name of the project.</td>
<td>N</td>
<td></td>
</tr>
</tbody>
</table>
$QUEUE

The $QUEUE field contains the specific name of the queue the call event was placed in, for example, CALLBACK/OP1 or COMPLETED.

User Update: Y

$READ.OK

The $READ.OK field contains the result of the last file operation; can be one of the following:

- **Y** Yes; the operation was successful.
- **N** No; the operation was unsuccessful. In this case, the record workspace is cleared.

The following EDGE verbs set $READ.OK:

- DELETE-RECORD.
- GET-KEY.
- READ-RECORD.
- RELEASE-RECORD.
- WRITE-RECORD.

When $READ.OK is set to N, an error code is returned in $DBERRNO.

The data in this field is valid only immediately after the file operation.

User Update: N
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Update Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>$RESULT</td>
<td>The $RESULT field contains the name of the result currently set for the call event. This field is updated each time Results are set and may change over the course of a call.</td>
<td>Y</td>
</tr>
<tr>
<td>$SA.SAVE</td>
<td>The $SA.SAVE field indicates whether a record should be written to the REPORTING File for the current call at the end of the call; can be one of the following:</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Y Yes; indicates a record is to be written. At the beginning of each call, EDGE sets $SA.SAVE to Y.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N No; indicates a record is not to be written. You must set this value.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If you do not want a record written to the REPORTING file, you must set this field to N using a logic flow.</td>
<td></td>
</tr>
<tr>
<td>$SCREEN</td>
<td>The $SCREEN field contains the name of the screen currently displayed.</td>
<td>N</td>
</tr>
<tr>
<td>$SCRNAMES</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>$SCRTIMES</td>
<td>Reserved</td>
<td></td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
<td>User Update</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>$SELNO</td>
<td>The $SELNO field contains the numbers of the current choices from a selection screen. (For more information, see the section Cell Properties.)</td>
<td>Y</td>
</tr>
<tr>
<td>$SELTXT</td>
<td>The $SELTXT field contains the text associated with selection numbers of the current choices from a selection screen. (For more information, see the section Cell Properties.)</td>
<td>Y</td>
</tr>
<tr>
<td>$SITETZ</td>
<td>The $SITETZ field contains the for the site as defined in the site setup.</td>
<td>N</td>
</tr>
<tr>
<td>$SYSEMSG</td>
<td>The $SYSEMSG field contains system error messages.</td>
<td>Y</td>
</tr>
<tr>
<td>$TCBDATE</td>
<td>The $TCBDATE field contains the customer callback date.</td>
<td>Y</td>
</tr>
<tr>
<td>$TCBTIME</td>
<td>The $TCBTIME field contains the customer callback time.</td>
<td>Y</td>
</tr>
</tbody>
</table>
$THEIRDATE

The $THEIRDATE field contains the current date at the customer site as determined by the TELNO and COUNTRY fields in the STATUS file.

This field is updated during the call as follows:

- At the beginning of the call.
- On entry to the Set Callback screen.

The date is calculated only on outbound calls. If the call is inbound, this field is set to your current date.

The date is displayed in MM-DD-YY format.

User Update: N

$THEIRTIME

The $THEIRTIME field contains the current time at the customer site as determined by the TELNO and COUNTRY fields in the STATUS file.

This field is updated during the call as follows:

- At the beginning of the call.
- On entry to the Set Callback screen.

The time is calculated only on outbound calls. If the call is inbound, this field is set to your current date.

The time is displayed in 24-hour format.

User Update: N

$TIME

The $TIME field is a synonym for $CURR.TIME.
$TOPCHOICE  
The $TOPCHOICE field contains the number of selections to rotate on the display. This field is used only with selection screens. (For more information, see the section Main Properties.)

User Update: Y

$TP.ACDGRP  
The $TP.ACDGRP field contains one of the following:

- With inbound telephony, contains the ACD group the agent is assigned to.
- With predictive dialing, contains the callset the agent is assigned to.

This field is used by telephony.

User Update: N

$TP.AGENTSTATUS  
The $TP.AGENTSTATUS field is a multivalued field that contains the current state of the agent. It is updated dynamically as the agent’s phone status changes. The following values are used:

1  The current state of the agent; can be one of the following:
   outbound status changed.
signed on
signed off
idle
available
make busy
wrap-up

2 The event type; can be one of the following:
    S Solicited
    U Unsolicited

3 The reason the state changed. This is determined by the gateway.

4 The time of the event in seconds.
   This field is used by telephony.

User Update: N

$TP.AGENT-TYPE The $TP.AGENT-TYPE field contains the Agent type specified when logging in.
   This field is used by telephony.

User Update: N

$TP.ALTNAME The $TP.ALTNAME field contains the alternate telephony ID for the switch.
   This field is used by telephony.

User Update: Y
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>User Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>$TP.ANI</td>
<td>The $TP.ANI field contains the telephone number from which the call originated.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>This field is used by telephony.</td>
<td></td>
</tr>
<tr>
<td>$TP.AVAIL</td>
<td>The $TP.AVAIL field specifies the available mode to be used for the next Make Agent Available command.</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>This field is used by telephony.</td>
<td></td>
</tr>
<tr>
<td>$TP.CALLHISTORY</td>
<td>The $TP.CALLHISTORY field is a multivalued field that contains the history of all call activities that occur from the start of the call to the $END screen. Each change in the call state is appended to the end of the $TP.CALLHISTORY field. The following values are used:</td>
<td>Y</td>
</tr>
<tr>
<td>1</td>
<td>The current state of the call; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>alerting</td>
<td></td>
</tr>
<tr>
<td></td>
<td>connected</td>
<td></td>
</tr>
<tr>
<td></td>
<td>disconnected</td>
<td></td>
</tr>
<tr>
<td></td>
<td>conferenced</td>
<td></td>
</tr>
<tr>
<td></td>
<td>held</td>
<td></td>
</tr>
<tr>
<td></td>
<td>retrieved</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>The event type; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The event type; can be one of the following:</td>
<td></td>
</tr>
</tbody>
</table>
S Solicited  
U Unsolicited  

3 The reason the call was disconnected; can be one of the following:  
A All other cases.  
B Abandoned.  
C Abandoned in queue.  
D Abandoned while ringing an agent.  
E Terminated by transfer.  

4 The call ID. This is provided by the gateway or CTI middleware.  

Note: The $TP.CALLHISTORY SYS field is automatically reduced in size by half when it reaches a maximum of seven (7) kilobytes; the oldest data is discarded. This prevents possible communication problems between the server and EDGE Client if the agent does not route to $END between phone calls.  

This field is used by telephony.  
User Update: Y
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$TP.CALLID</td>
<td>The $TP.CALLID field contains the ACD callid in hexadecimal format. This field is used by telephony.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$TP.CALLSET</td>
<td>The $TP.CALLSET field is a synonym for $TP.ACDGRP. This field is used by telephony.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$TP.CALLSTATUS</td>
<td>The $TP.CALLSTATUS field contains the current call state for the agent; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>alerting, conferenced, connected, disconnected, held, retrieved, transferred</td>
</tr>
<tr>
<td></td>
<td>This field is used by telephony.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$TP.DATA</td>
<td>The $TP.DATA field contains the data passed from a TELEPHONY verb command that has the properties Send misc data 1 and Send misc data 2. This is a multivalued field.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$TP.DEVICE</td>
<td>The $TP.DEVICE field contains the name of the telephony device associated with this agent.</td>
</tr>
<tr>
<td>$TP.DIGITS</td>
<td>The $TP.DIGITS field contains the collected digits associated with a call.</td>
</tr>
<tr>
<td>$TP.DNIS</td>
<td>The $TP.DNIS field contains the telephone number dialed.</td>
</tr>
<tr>
<td>$TP.HBRDEST</td>
<td>The $TP.HBRDEST field contains the destination for a host based routing request. This field is only used in tasks.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$TP.II_DIGITS</td>
<td>The $TP.II_DIGITS field contains information on the originating phone line. For example, these digits can indicate if the call originated from a cellular phone, pay phone, a jail, a hotel, and so on. The II-digits are passed by the phone network to the switch. The value stored in $TP.II-DIGITS is standard across the phone network. A list of the possible values can be found in BellCore's Local Exchange Routing Guide (document number TR-EOP-000085). This field can be used to route or provide special treatment for an incoming call. The $TP.II_DIGITS field is available following a successful Check for a Call or Wait for a Call command and for host based routing. This field is used by telephony. User Update: N</td>
</tr>
<tr>
<td>$TP.IMITATE</td>
<td>The $TP.IMITATE field specifies whether a telephony user is being impersonated. This allows the process requesting a telephony operation to be a different process from the one the operation is applied to. This enables telephony functionality to be used in tasks, or to log agents in from a supervisor’s station. This field is used by telephony. User Update: Y</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$TP.IMPLICIT-AVAIL</td>
<td>The $TP.IMPLICITAVAIL field controls the make agent available status. At the $INWAIT screen, EDGE automatically makes an agent available to receive a call. Some gateways will disconnect the call when this occurs. This feature can be turned off by setting it to &quot;N&quot; at the end of the Guide. This field is used by telephony. User Update: Y</td>
</tr>
<tr>
<td>$TP.IMPLICIT-UNAVAIL</td>
<td>The $TP.IMPLICITUNAVAIL field controls the make agent unavailable status. If the agent cancels out of the auto-receive mode at the Call Center screen, EDGE automatically will make the agent unavailable. Some gateways will hang up the current call. This feature can be turned off by setting it to &quot;N&quot;. This field is used by telephony. User Update: Y</td>
</tr>
<tr>
<td>$TP.INACD</td>
<td>The $TP.INACD field is a synonym for $TP.INVDN. This field is used by telephony. User Update: N</td>
</tr>
<tr>
<td>$TP.INEXT</td>
<td>The $TP.INEXT field contains the inbound extension associated with an agent or specifies which agent to manipulate if used in a task. If used in a task, the field $TP.IMITATE must also be specified. This field is used by telephony.</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$TP.INVDN</td>
<td>The $TP.INVDN field contains the extension, ACD queue, or the vector directory number (VRU port number) indicating where the call was received from. This field is used by telephony.</td>
</tr>
<tr>
<td>$TP.LOCATION</td>
<td>The $TP.LOCATION field contains the T-Server switch ID to which the call is to be routed when the call's destination (the party being dialed) is located on a connected switch of the same type. The switch ID can be set by copying a value into this field, or by setting the value in a TELEPHONY verb property. This field is also used in the filename for data transferred between switches to ensure uniqueness in filenames.</td>
</tr>
<tr>
<td>$TP.LOGONTOACD</td>
<td>The $TP.LOGONTOACD field is passed to the Avaya Pro-Active contact server while logging on to the dialer. The value for this field initially comes from the user id setup. This field is used by telephony.</td>
</tr>
</tbody>
</table>
$TP.OUTEXT

The $TP.OUTEXT field contains the outbound extension associated with an agent or specifies which agent to manipulate if used in a task. If used in a task, the field $TP.IMITATE must also be specified.

This field is used by telephony.

User Update: Y only in task.

$TP.REASON

The $TP.REASON field contains either ‘Inbound’ or ‘Outbound’ when the Avaya Proactive Contact system generates the AGTCallNotify event to change the Agent status as determined by $TP.AGENTSTATUS.

This field is used by telephony

User Update: Y

$TP.SETTYPE

The $TP.SETTYPE field contains the Aspect phoneset type associated with an agent.

This field is used by telephony.

User Update: N

$TP.SWINFGROUP

The $TP.SWINFGROUP field contains the Rockwell agent information groups if any have been specified in the workstation ID properties. Obsolete. Retained for backward compatibility.

This field is used by telephony.

User Update: Y
**$TP.SWLOGIN** The $TP.SWLOGIN field contains the switch login ID associated with an agent.

This field is used by telephony.

User Update: N

**$TP.SWNUM** The $TP.SWNUM field contains the Rockwell agent sign-on ID. Obsolete. Retained for backward compatibility

This field is used by telephony.

User Update: Y

**$TP.SWPASS** The $TP.SWPASS field contains the switch login password associated with an agent.

This field is used by telephony.

User Update: N

**$TP.SWSUPNUM** The $TP.SWSUPNUM field contains the Rockwell supervisor sign-on ID. Obsolete. Retained for backward compatibility

This field is used by telephony.

User Update: Y

**$TP.UNAVAIL** The $TP.UNAVAIL field specifies the unavailable mode to be used for the next Make Agent Unavailable command.

This field is used by telephony.
$TP.USER

The $TP.USER field specifies the user ID of the agent to manipulate. If used in a task, the field $TP.IMITATE must also be specified.

This field is used by telephony.

User Update: Y

$TP.VRU

The $TP.VRU field indicates whether the current call has been through the VRU; can be one of the following:

N  Call did not come through the VRU.
Y  Call came through the VRU.

This field is used by telephony.

User Update: N

$TP.WORKCLASS

The agent workclass used when logging on to Avaya Proactive Contact. This value comes from the user id setup and can be one of the following:

B  Blend
O  Outbound
I  Inbound
M  Managed
P  Person-to-Person
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$TPHYDEV</td>
<td>This field is used by telephony. The $TPHYDEV field is a synonym for $TP.DEVICE. User Update: Y</td>
</tr>
<tr>
<td>$UAPCODE</td>
<td>The $UAPCODE field is no longer supported. It is retained for backward compatibility. User Update: N</td>
</tr>
<tr>
<td>$UAPCOUNT</td>
<td>The $UAPCOUNT field is no longer supported. It is retained for backward compatibility.</td>
</tr>
<tr>
<td>$UAPHIST</td>
<td>The $UAPHIST field is no longer supported. It is retained for backward compatibility.</td>
</tr>
<tr>
<td>$UAPMATCHID</td>
<td>The $UAPMATCHID field is no longer supported. It is retained for backward compatibility.</td>
</tr>
<tr>
<td>$UAPSCORE</td>
<td>The $UAPSCORE field is no longer supported. It is retained for backward compatibility.</td>
</tr>
<tr>
<td>$UNAVAIL</td>
<td>The $UNAVAIL field is a synonym for $TP.UNAVAIL. This field is used by telephony. User Update: Y</td>
</tr>
<tr>
<td>$USER.ID</td>
<td>The $USER.ID field is a synonym for SOPID. User Update: N</td>
</tr>
<tr>
<td>$USESDEST</td>
<td>Reserved.</td>
</tr>
</tbody>
</table>
$VAL

The $VAL field contains the position of the value of the field associated with the object that currently has the focus, if that object is an input field.

$VAL can be used in logic flows to locate data in a specific record when the input data was from a multivalued field. $VAL can also be used in logic to match the contents of several multivalued fields.

For example, when taking an order you can display the product number in PROD,$VAL, the description in DESC,$VAL, and the price in PRICE,$VAL.

This field can be used only in individual field logic.

User Update: N

$VQDATE

The $VQDATE date contains the view queue date specified from the ASCII $QVIEW screen. The date is displayed in MM DD YY format. This field is not populated in EDGE Client; it is used only in ASCII operations when F4 (View/Select Callbacks) on $OP is pressed.

User Update: N

$VQID

The $VQID field contains the name of the queue being viewed specified from the ASCII $QVIEW screen. This field is not populated in EDGE Client; it is used only in ASCII operations when F4 (View/Select Callbacks) on $OP is pressed.

User Update: N
<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$VQS</td>
<td>The $VQS field contains the number of the choice from the ASCII $QVIEW screen. This field is not populated in EDGE Client; it is used only in ASCII operations when F4 (View/Select Callbacks) on $OP is pressed.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$VQTIME</td>
<td>The $VQTIME field contains the queue time specified for the ASCII $QVIEW screen. The time is displayed in HH:MM AMPM format. This field is not populated in EDGE Client; it is used only in ASCII operations when F4 (View/Select Callbacks) on $OP is pressed.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>$WINCLOSING</td>
<td>The $WINCLOSING field specifies that a window is closing because of a click on the close box; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Y Yes; indicates the window is closing.</td>
</tr>
<tr>
<td></td>
<td>N No; indicates a window is not to be closed.</td>
</tr>
<tr>
<td></td>
<td>To keep the window open, set this value to N.</td>
</tr>
<tr>
<td></td>
<td>User Update: Y</td>
</tr>
<tr>
<td>$WINDOW</td>
<td>The $WINDOW field contains the name of the current window.</td>
</tr>
<tr>
<td></td>
<td>User Update: N</td>
</tr>
<tr>
<td>Field</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$WINDOWLIST</td>
<td>The $WINDOWLIST field contains a list of open windows in EDGE Client.</td>
</tr>
<tr>
<td>$WSID</td>
<td>The $WSID field contains the workstation ID settings.</td>
</tr>
<tr>
<td></td>
<td>This field is used with telephony.</td>
</tr>
<tr>
<td>$XCALLEVENT</td>
<td>The $XCALLEVENT field is a synonym for $CALLEVENT.</td>
</tr>
<tr>
<td>$XCALLID</td>
<td>The $XCALLID field is a synonym for $CALLID, except it has length 20, allowing up to 20 characters to be displayed.</td>
</tr>
</tbody>
</table>
TEMP File

The TEMP file has no predefined fields. You define fields in it as you need them.

The fields you define can be used to store data during a call or task, but EDGE does not write the data to the file. The fields are cleared at the end of the call or task. This means, for example, that if you copy a value to the TEMP file in one task, then chain to a second task, any data in the TEMP file from the first task will have been cleared.

To save the data, use the COPY verb to copy the data to a field in one of your files before the end of the call.

Generally, you use the TEMP file to store data that is pertinent to the flow of the guide, but that does not need to be saved after the call or task is completed. For example, you can use a TEMP file field to display instructions to the agent.

The TEMP file is an alternative to using Variables to store information.
Overview

The importing and exporting modules allow you to move data between EDGE and non-EDGE applications. For example, you may want to use data from your company’s mainframe files to populate EDGE files; you may want to receive data from another company site; or you may want to copy data from EDGE to your mainframe files to be used in spreadsheet applications.

The Import module allows you to bring data from a text file into an EDGE database file at either the project or EDGE level. The Export module allows you to copy fields from an EDGE database file to a text file.

You can import or export data from within EDGE Developer, from a task using the EXECUTE verb with the edgeio command, or from ACL using the edgeio command.

For importing to or exporting from multiple database files, you can use the SmartMerge command.
Preparing to Import Data

Before you begin the import process, the following steps must be performed:

1. Create an I/O device. For information on creating I/O devices, see the section I/O Device Setup.
2. Get the format of the source data from the provider of the file.
3. Create the file and fields into which the data is to be imported. For information on creating files, see Chapter 11, User Files and Fields.
4. Create the Import Instructions.

We also recommend you do the following; these steps are described in the section Import Commands:

5. Display the import instructions and verify all the definitions.
6. Display the import records and compare to the format.
7. Do a test import and check errors.
8. After you finish the import process, we recommend you list the processing errors and resolve them, if any.

Preparing to Export Data

Before you begin the export process, the following steps must be performed:
1. Create an I/O device. For information on creating I/O devices, see the section I/O Device Setup.

2. Verify the format of the destination file with the receiver of the file.

3. Create the Export Instructions.

4. If only certain records are to be exported, create a selection list. For information on creating selection lists, see Chapter 7, Ad Hoc Reports.

We also recommend that you do the following; these steps are described in the section Export Commands:

5. Display the export instructions and verify all the definitions.

6. Display export records and compare to format.

7. Do a test export and check errors.

8. After you finish the export process, we recommend you list the processing errors and resolve them.
Import Instructions

An import instruction is a set of properties that tells EDGE about data to be imported from one file and how it is to be saved in an EDGE file.

To define an import instruction, select the following:

EDGE or project / Database area / Import tab

Select an existing definition, or click the Add button. The following property sheets are available:

- General Property Sheet.
- Selection Criteria Property Sheet.
- Add Fields Property Sheet.

General Property Sheet

The General Property Sheet is used to define the properties for the output device and is similar to the following:
**Properties:**

**Add/update**

Indicates how the records are to be processed into the destination file and its associated cross-reference file, if any; can be one of the following:

- **Add**
  
  Adds records to the file. If an imported record has the same record ID as an existing record, an error is recorded in the log file, if Log errors is checked.

- **Both**
  
  Updates specified fields in records in the destination file that match imported records. Adds new records to the destination file when there are no matching records.
Properties: (continued)

**Update**
Updates specified fields in records in the destination file that match imported records. If the record ID of an imported record does not match the name of a record in the destination file, an error is recorded in the log file, if *Log errors* is checked.

**Block size**
The number of characters in a source block. The block size must be a multiple of record size. Block size is meaningful only when the data to be imported is on tape.

This property is displayed only if Fixed length record format is selected.

**Clear file**
Specifies whether the destination file and associated cross-reference file, if any, should be cleared before source records are imported.

**Data format**
The format of the source data; can be one of the following:

- ASCII labeled
- ASCII unlabeled
- EBCDIC labeled
- EBCDIC unlabeled

The default data format is EBCDIC labeled. The data is converted to ASCII when it is imported.
**Properties:** (continued)

**Error limit**  
The number of errors that can be encountered on the import before the import process is stopped. The default error limit is 100. If the number of errors exceeds the limit, the import is stopped and the following message is displayed:

```
Process aborted. Errors exceeded limit.
```

The records that were imported before the error limit was reached are left in the destination file.

**Field mark**  
The character that indicates the end of a field. You can type in any character, or you can select one of the following predefined field marks:

- Formfeed.
- Linefeed.
- Return.
- Space.
- Tab.

The character you specify as a field mark should not be any character that could possibly be in the data. For example, if there are spaces in your data, do not select space as the field mark.

You cannot specify the same character for both a record mark and a field mark.

This property is displayed only if Variable length record format is selected.
**Importing and Exporting**

**Import Instructions**

**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>From device</strong></td>
<td>The name of the I/O device to be used as the source of the data to be imported. (For information on creating an I/O Device, see the section I/O Device Setup.)</td>
</tr>
<tr>
<td><strong>Into</strong></td>
<td>The name of the destination file.</td>
</tr>
<tr>
<td><strong>Log errors</strong></td>
<td>Specifies whether errors that are encountered during the import process should be written to the log file. (The log file is called TAPE.ERRORS and is in the current directory. For information on listing the log file, see the section Import Commands.)</td>
</tr>
<tr>
<td><strong>Record format</strong></td>
<td>The format of the source records; can be one of the following:</td>
</tr>
<tr>
<td>Fixed length</td>
<td>Every record has the same number of characters. Fixed length fields are usually padded with space characters, which may cause matching problems in EDGE. You can remove these padding characters with the Trim option.</td>
</tr>
<tr>
<td>Variable length</td>
<td>The number of characters can vary by record. The Record mark property indicates the end of record character. The Field mark property indicates the end of field character.</td>
</tr>
<tr>
<td></td>
<td>The default record format is Fixed Length.</td>
</tr>
</tbody>
</table>
Selection Criteria Property Sheet

The Selection Criteria property sheet is used to specify criteria that source records must meet to be imported. When you click the Selection Criteria tab, a screen similar to the following is displayed:

Properties: (continued)

Record mark
The character that indicates the end of a record. You can type in any character, or you can select one of the following predefined record marks:
- Formfeed.
- Linefeed.
- Return.
- Space.
- Tab.

The character you specify as a record mark should not be any character that could possibly be in the data. For example, if there are commas in your data, do not select comma as the record mark.

This property is displayed only if Variable length record format is selected.

Record size
The number of characters in the source record.

This property is displayed only if Fixed length record format is selected.

Selection Criteria Property Sheet

The Selection Criteria property sheet is used to specify criteria that source records must meet to be imported. When you click the Selection Criteria tab, a screen similar to the following is displayed:

Properties: (continued)

Record mark
The character that indicates the end of a record. You can type in any character, or you can select one of the following predefined record marks:
- Formfeed.
- Linefeed.
- Return.
- Space.
- Tab.

The character you specify as a record mark should not be any character that could possibly be in the data. For example, if there are commas in your data, do not select comma as the record mark.

This property is displayed only if Variable length record format is selected.

Record size
The number of characters in the source record.

This property is displayed only if Fixed length record format is selected.
**Properties:**

- **Clear ID file**
  Specifies whether the file specified in the ID File property is cleared before the import starts.

- **From device**
  The name of the I/O device being used for the import. This property is display only.

- **ID file**
  The name of an existing file into which the source record IDs can be saved. Only record IDs are added; no data is saved in the ID file. You can use this ID file, for example, to create a list of records that were imported.
  This property is optional.
## Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input character conversion</td>
<td>Source characters that require special processing; for example, control characters in the source data.</td>
</tr>
<tr>
<td>Ignore</td>
<td>Characters that are not to be copied to the destination file. The Ignore characters are processed before a Replace is performed.</td>
</tr>
<tr>
<td>Replace</td>
<td>Characters that are to be converted before being output to the destination file.</td>
</tr>
<tr>
<td>With</td>
<td>Characters that are to replace corresponding source characters in the Replace property. If a source character in Replace has no corresponding character in With, it is not copied to the destination file.</td>
</tr>
<tr>
<td>For information on how to specify characters, see Table 13-1.</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>The length of the field to be used for selection. The property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td>Read every</td>
<td>Specifies which records are to be read. For example, if you specify 3, every third record is read. The default is to read every record.</td>
</tr>
<tr>
<td>Record format</td>
<td>The record format of the source records. This property is display only.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Record limit</td>
<td>The maximum number of records to be written to the destination file. If the source file contains fewer records, no error is noted.</td>
</tr>
<tr>
<td>Record type</td>
<td>The criteria records must meet before they can be selected for import.</td>
</tr>
<tr>
<td>Equal to</td>
<td>The values the record must match if it is to be imported. For example, if you want to import records only from California, specify CA.</td>
</tr>
<tr>
<td>Length</td>
<td>The length of the field to be used for selection. This property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td>Not equal to</td>
<td>The values the record must not be equal to if it is to be imported. For example, if you do not want to import any records from California, specify CA in this property.</td>
</tr>
<tr>
<td>Position</td>
<td>The position of the field to be used for selection. This property is displayed only if Variable length record format is selected.</td>
</tr>
<tr>
<td>Start</td>
<td>The first position of the field to be used for selection. This property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td>Skip</td>
<td>The number of records to be skipped before starting the import.</td>
</tr>
</tbody>
</table>
Add Fields Property Sheet

The Add Fields property sheet is used to specify the data characteristics of the import fields.

To import the record ID of a file, you must include the record ID field in the Add Fields property sheet. If you do not, EDGE will assign a sequential number as the record ID for the file.

Before you add fields, specify the following properties on the General property sheet:

- Fill the Into property with the name of the file into which the data is to be imported.
- If you are importing data from a fixed length record, specify the record size and block size.

Table 13-1. Input Character Conversion

<table>
<thead>
<tr>
<th>Characters to Check</th>
<th>How to Enter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control character</td>
<td>Enter ^code (caret, character); for example, ^c.</td>
</tr>
<tr>
<td>Escape character</td>
<td>Enter \code (backslash, character); for example, \G.</td>
</tr>
<tr>
<td>Hexadecimal value</td>
<td>Enter \0x (backslash, zero, lowercase x), followed by the hexadecimal value; for example, \0x9A.</td>
</tr>
<tr>
<td>Octal value</td>
<td>Enter \0 (backslash, zero), followed by the octal value; for example \0377.</td>
</tr>
<tr>
<td>Printable character</td>
<td>Enter the character itself; for example, z.</td>
</tr>
</tbody>
</table>
### Properties:

- **Comment**: The description of the field. The default comment is the Heading property in the field definition.
- **Decimals**: The number of decimal places to retain in numeric data.
- **End**: The last position of a field in the record. This property is displayed only if Fixed length record format is selected.
- **Field**: The name of the destination field. When you specify the field name, the type is filled in automatically.
- **Format**: The data format. Available formats depend on the Type specified. For a description of the formats, see Table 13-2.
**Properties:** (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Justification</strong></td>
<td>Specifies how the data is to be justified in the record; can be one of the following:</td>
</tr>
<tr>
<td>Left</td>
<td>Data is placed in the field starting at the leftmost character. If necessary, it is padded to the right with spaces.</td>
</tr>
<tr>
<td>Right</td>
<td>Data is placed in the field starting at the rightmost character. If necessary, it is padded to the left with spaces.</td>
</tr>
<tr>
<td><strong>Length</strong></td>
<td>The number of characters in the field. This property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td><strong>Limit</strong></td>
<td>The highest number to be placed in a replicate field. This property is available only if Type is Replicate.</td>
</tr>
<tr>
<td><strong>Literal</strong></td>
<td>The literal value to be placed in the field. This property is available only if the Type is literal.</td>
</tr>
<tr>
<td><strong>Position</strong></td>
<td>The position of the field in the record. This property is displayed only if Variable length record format is selected.</td>
</tr>
<tr>
<td><strong>Start</strong></td>
<td>The first position of the field in the record. This property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td><strong>Trim</strong></td>
<td>Specifies how space characters are to be processed before writing the data; can be one of the following:</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excess</td>
<td>Removes all spaces to the left of the first nonspace character and to the right of the last nonspace character, and all but one space between any two nonspace characters.</td>
</tr>
<tr>
<td>Leading</td>
<td>Removes all spaces to the left of the first nonspace character.</td>
</tr>
<tr>
<td>None</td>
<td>Removes no spaces.</td>
</tr>
<tr>
<td>Surrounding</td>
<td>Removes all spaces to the left of the first nonspace character and to the right of the last nonspace character.</td>
</tr>
<tr>
<td>Trailing</td>
<td>Removes all spaces to the right of the last nonspace character.</td>
</tr>
</tbody>
</table>

**Type**

The data type of the destination field and is used by EDGE to determine how to copy the data. Specific formats for each type are listed in Table 13-2. Type can be one of the following:

- **Alphanumeric**
  - The field can contain any characters.
- **Counter**
  - The field is to contain a sequential number, starting at 1 (one) for the first record. The counter is incremented by one for each record read. (The number is not part of the source data.)
Importing and Exporting

Properties:  (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Date</td>
<td>The field contains a date. If a two-digit year is specified, the century is determined by the setting of EDGE base year.</td>
</tr>
<tr>
<td>Literal</td>
<td>The destination field is to contain a specified literal value. Every record contains the same value. (The literal is not part of the source data.)</td>
</tr>
<tr>
<td>Numeric</td>
<td>The field contains numeric characters. To view the results of various settings for numeric types, see Table 13-3.</td>
</tr>
<tr>
<td>Replicate</td>
<td>The destination field is to contain a sequential number, starting at 1 (one) for the first record, up to a specified limit. When the limit is reached, the counter is reset to 1 and incremented for each succeeding record until the limit is again reached. (The number is not part of the source data.) For example, if you specify 10 as the limit, the records are numbered as they are imported, starting with 1 and going up to 10, then starting over again at 1. If the file has 100 records, when the import is finished there will be 10 records with the number 1, 10 records with the number 2, and so on.</td>
</tr>
</tbody>
</table>
Properties: (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>The field contains a time value. The source data is not validated to ensure it meets the characteristics of the specified type.</td>
</tr>
<tr>
<td>Value position</td>
<td>The position of a multivalued field in which to place the data; can be one of the following:</td>
</tr>
<tr>
<td>n</td>
<td>Copies the data into value position n and overwrites any existing data.</td>
</tr>
<tr>
<td>B</td>
<td>Inserts the data as the first value; existing data is shifted to the next higher value position.</td>
</tr>
<tr>
<td>E</td>
<td>Appends the data as a value following the last value.</td>
</tr>
</tbody>
</table>

This property is available only if the field is defined as multivalued.

Update if blank

If checked and the source field is null or contains nothing but spaces, specifies the source data is to be copied into the destination field. If not checked and the source field is null or contains nothing but spaces, the destination field is not updated.
**Table 13-2. Import Format Options by Data Types**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alphanumeric</td>
<td>Capitalize</td>
<td>Capitalizes the first letter of each word. A word is defined as a group of characters delimited by spaces, a punctuation mark, or end-of-line mark. If the word begins with a nonletter character, the first letter character is capitalized; for example, 4ever is converted to 4Ever.</td>
</tr>
<tr>
<td></td>
<td>Lowercase</td>
<td>Converts all letters to lowercase.</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>Does no conversion of the source data.</td>
</tr>
<tr>
<td></td>
<td>Uppercase</td>
<td>Converts all letters to uppercase.</td>
</tr>
<tr>
<td>Date</td>
<td>None</td>
<td>The source is not converted. The date is displayed in EDGE internal format.</td>
</tr>
<tr>
<td></td>
<td>DD-MM-YY</td>
<td>The date is in international numeric format with a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>DD-MM-YYYY</td>
<td>The date is in international numeric format with a four-digit year.</td>
</tr>
<tr>
<td></td>
<td>DD-MMM-YY</td>
<td>The date is in international format with a three-letter abbreviation for the month and a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>DD-MMM-YYYY</td>
<td>The date is in international format with a three-letter abbreviation for the month and a four-digit year.</td>
</tr>
</tbody>
</table>
Table 13-2. Import Format Options by Data Types (continued)

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MM-DD-YY</td>
<td>The date is in US numeric format with a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>MM-DD-YYYY</td>
<td>The date is in US numeric format with a four-digit year.</td>
</tr>
<tr>
<td></td>
<td>MMM-DD-YY</td>
<td>The date is in US format with a three-letter abbreviation for the month and a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>MMM-DD-YYYY</td>
<td>The date is in US format with a three-letter abbreviation for the month and a four-digit year.</td>
</tr>
<tr>
<td></td>
<td>month-DD-YY</td>
<td>The date is in US format with the month spelled out and a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>month-DD-YYYY</td>
<td>The date is in US format with the month spelled out and a four-digit year.</td>
</tr>
<tr>
<td></td>
<td>YYMMDD</td>
<td>The date is in numeric format starting with a two-digit year and no separator characters.</td>
</tr>
<tr>
<td></td>
<td>YYYYMMDD</td>
<td>The date is in numeric format starting with a four-digit year and no separator character.</td>
</tr>
</tbody>
</table>
**Table 13-2. Import Format Options by Data Types (continued)**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>UNIX Internal</td>
<td>This is the EDGE internal format, which is the number of days since December 31, 1967.</td>
</tr>
<tr>
<td></td>
<td>User defined</td>
<td>A format that you define. You can use the following characters:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DD Day of the month.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MM Month as a number</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MMM Month as a three-letter abbreviation.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>YYYY Complete year.</td>
</tr>
<tr>
<td></td>
<td>Numeric</td>
<td>None</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The source is not converted. The Decimals property has no effect on the data.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, MMM YYYY would produce a date of Oct 2001.</td>
</tr>
</tbody>
</table>
### Table 13-2. Import Format Options by Data Types (continued)

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit precision</td>
<td>The decimal point for numbers is explicit and is stored in the data. The value is truncated to the number of decimal places specified in the <code>Decimals</code> property.</td>
</tr>
<tr>
<td></td>
<td>Implicit precision</td>
<td>The decimal point for numbers is implicit. To specify the number of decimal places, use the <code>Decimals</code> property. If the stored data contains a decimal point, the Implicit format has no effect.</td>
</tr>
<tr>
<td></td>
<td>Truncate</td>
<td>The value is truncated to the number of decimal places specified in the <code>Decimals</code> property.</td>
</tr>
<tr>
<td>Time</td>
<td>None</td>
<td>The source is not converted. The data is displayed in EDGE internal format.</td>
</tr>
<tr>
<td></td>
<td>HH:MM</td>
<td>The hours and minutes in 24-hour format; for example, 16:05.</td>
</tr>
<tr>
<td></td>
<td>HH:MM:SS</td>
<td>The hours, minutes, and seconds in 24-hour format; for example, 16:05:30.</td>
</tr>
<tr>
<td></td>
<td>HH:MMampm</td>
<td>The hours and minutes in 12-hour format; for example, 04:05pm.</td>
</tr>
</tbody>
</table>
**Table 13-2. Import Format Options by Data Types (continued)**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HH:MM:SSampm</td>
<td>The hours, minutes, and seconds in 12-hour format; for example, 04:05:30pm.</td>
</tr>
<tr>
<td></td>
<td>User defined</td>
<td>A format that you define. You can use the following characters:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HH    Hour.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MM    Minutes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SS    Seconds.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ampm  Time in 12-hour format.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>:      Delimiter; can be any character.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>For example, HH.MM would produce a time of 16.05.</td>
</tr>
</tbody>
</table>
### Table 13-3. Results of Various Settings for Decimals and Format

<table>
<thead>
<tr>
<th>Source</th>
<th>Decimals</th>
<th>Destination Format</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2.123</td>
<td>0</td>
<td>2</td>
<td>2.123</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>2.123</td>
<td>2</td>
<td>2.12</td>
<td>2.123</td>
<td>2.12</td>
<td>2.12</td>
</tr>
<tr>
<td>2.123</td>
<td>3</td>
<td>2.123</td>
<td>2.123</td>
<td>2.123</td>
<td>2.123</td>
</tr>
<tr>
<td>2.123</td>
<td>4</td>
<td>2.1230</td>
<td>2.123</td>
<td>2.1230</td>
<td>2.1230</td>
</tr>
<tr>
<td>2123</td>
<td>0</td>
<td>2123</td>
<td>2123</td>
<td>2123</td>
<td>2123</td>
</tr>
<tr>
<td>2123</td>
<td>2</td>
<td>2123.00</td>
<td>2123.00</td>
<td>2123.00</td>
<td></td>
</tr>
<tr>
<td>2123</td>
<td>3</td>
<td>2123.000</td>
<td>2123.000</td>
<td>2123.000</td>
<td></td>
</tr>
<tr>
<td>2123</td>
<td>5</td>
<td>2123.00000</td>
<td>0.02123</td>
<td>2123.00000</td>
<td></td>
</tr>
</tbody>
</table>
## Import Commands

The following commands are available on the Import menu:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Begin Import.</strong></td>
<td>Begins the import process. For details, see the next section.</td>
</tr>
<tr>
<td>Clear All Processing Errors.</td>
<td>Clears the error processing logs for all import definitions in the current project.</td>
</tr>
<tr>
<td>Clear Processing Errors.</td>
<td>Clears the error processing log for the currently selected import definition. If no import definition is selected, this option is not available.</td>
</tr>
<tr>
<td>List All Import Device Formats.</td>
<td>Lists all the import definitions for the current project.</td>
</tr>
<tr>
<td>List All Import Processing Errors.</td>
<td>This command lists the error processing logs for all the import definitions in the current project.</td>
</tr>
<tr>
<td>List Current Import Device Format.</td>
<td>This command lists the currently selected import definition.</td>
</tr>
<tr>
<td>List Current Import Processing Errors.</td>
<td>This command lists the error processing log for all the currently selected import definitions. If no import definition is selected, this command is not available.</td>
</tr>
</tbody>
</table>
Begin Import

For importing, you can have EDGE assign a sequential number to each record as the record ID if you do not specify the record ID as one of the fields. When you begin the import, EDGE prompts you for the number at which to start assigning record IDs.

When the Begin Import command is selected, a dialog box similar to the following is displayed:

List Import Device Records.

This command displays the source records. This is the same as clicking Yes at the Display tape records prompt when you begin the import.

Note: For errors to be logged, you must specify Log Errors in the instruction before you execute the import.
Properties:

- **Begin import**: Begins the import, using the properties checked. If the record ID is not specified as one of the destination fields, the import process requests the initial value for the record IDs; see Figure 13-1. If the record ID is specified, this prompt is skipped.

- **Display tape records before starting**: Displays the first record to be imported. (The list is the same as selecting the List device records command.)

- **Rewind tape before starting**: Rewinds the tape. Meaningful only if your data is on tape and the specified path is to a rewind tape drive.

- **Test only**: Tests the import; does not actually write any records.
**Figure 13-1. Import Prompt for Record ID Value**
Export Instructions

An export instruction is a set of properties that tells EDGE about data to be exported from an EDGE file and how it is to be saved in a text file.

To define an import instruction, select the Import tab on the Database tab sheet. Select an existing definition, or click the Add button. The following property sheets are available:

- General Property Sheet.
- Add Fields Property Sheet.

General Property Sheet

The General Property Sheet is used to define the properties for the input device and is similar to the following:
**Properties:**

**Block size**
- The number of characters in a destination block. The block size must be a multiple of record size.
- This property is displayed only if Fixed length record is selected.

**Data format**
- The format of the destination data. You can specify one of the following:
  - ASCII unlabeled
  - EBCDIC unlabeled
  - EBCDIC unlabeled
- The default is EBCDIC unlabeled. If you need labeled tapes, see the section **Creating a Tape Label**.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Error limit** | The number of errors that can be encountered in the export before the export process is canceled. The default is 100. If the number of errors exceeds the limit, the export is stopped, and the following message is displayed:  
  Process aborted. Errors exceeded limit. |
| **Field mark** | The character that indicates the end of a field. You can type in any character, or you can select one of the following predefined field marks:  
  Formfeed.  
  Linefeed.  
  Return.  
  Space.  
  Tab.  
  You cannot specify the same character for both a record mark and a field mark.  
  This property is displayed only if Variable length record format is selected. |
| **From** | The name of the source file. |
| **Log errors** | Specifies whether errors that are encountered during the export process should be written to the log file.  
  (The log file is called TAPE_ERRORS and is in the current directory. For information on listing the log file, see the section Export Commands.) |
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Record format</strong></td>
<td>The format of the export records; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Fixed length</strong> Every record has the same number of characters.</td>
</tr>
<tr>
<td></td>
<td><strong>Variable length</strong> The number of characters can vary by record.</td>
</tr>
<tr>
<td></td>
<td>The default is Fixed Length.</td>
</tr>
<tr>
<td><strong>Record mark</strong></td>
<td>The character that indicates the end of a record. You can type in any character, or you can select one of the following predefined record marks:</td>
</tr>
<tr>
<td></td>
<td><strong>Formfeed</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Linefeed</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Return</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Space</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Tab</strong></td>
</tr>
<tr>
<td></td>
<td>You cannot specify the same character for both a record mark and a field mark.</td>
</tr>
<tr>
<td></td>
<td>This property is displayed only if Variable length record format is selected.</td>
</tr>
<tr>
<td><strong>Record size</strong></td>
<td>The number of characters to be written to an export record.</td>
</tr>
<tr>
<td></td>
<td>This property is displayed only if Fixed length record format is selected.</td>
</tr>
</tbody>
</table>
The Export process does not provide a way of specifying labeled tapes. If you need a label, you can do one of the following:

- Write the label using other software, then run the export process.
- Create two export instructions. Have the first export instruction create a label by using the literal data type to specify the label data. Have the second instruction export the data.

If you choose to write a label using one of these methods, be sure to specify the non-rewind tape path when creating your I/O device.

The Add Fields property sheet is used to specify the data characteristics of the export fields.

Before you add fields, specify the following properties on the General Property Sheet:
- In the From property, specify the name of the file from which the data is to be exported.

- If you are exporting data to a fixed length record, specify the record size and block size.

**Properties:**

- **Comment**: A description of the field. The default comment is the Heading property in the field definition.

- **Decimal**: The number of decimal places to retain in numeric data.

- **End**: The last position of a field in the record. This property is displayed only if Fixed length record format is selected.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>The name of the source field.</td>
</tr>
<tr>
<td>Format</td>
<td>Formats for each data type.</td>
</tr>
<tr>
<td>Justification</td>
<td>How the data is to be justified in the record; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Left</strong> Data is placed in the field starting at the leftmost character. If necessary, it is padded to the right with spaces.</td>
</tr>
<tr>
<td></td>
<td><strong>Right</strong> Data is placed in the field starting at the rightmost character. If necessary, it is padded to the left with spaces.</td>
</tr>
<tr>
<td></td>
<td>This property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td>Length</td>
<td>The number of characters in the field. This property is displayed only if Fixed length record format is selected.</td>
</tr>
<tr>
<td>Limit</td>
<td>The highest number to be placed in a replicate field. Replicate fields are described in the Replicate Type section later in this chapter.</td>
</tr>
<tr>
<td>Literal</td>
<td>The literal value to be placed in the field. Literal fields are described in the Literal Type section later in this chapter.</td>
</tr>
<tr>
<td>Position</td>
<td>The position of the field in the record. This property is displayed only if Variable length record format is selected.</td>
</tr>
</tbody>
</table>
Properties: (continued)

Start
The first position of the field in the record. This property is displayed only if Fixed length record format is selected.

Trim
How space characters are to be processed before writing the data; can be one of the following:

- **Excess**
  Removes all spaces to the left of the first nonspace character and to the right of the last nonspace character, and all but one space between any two nonspace characters.

- **Leading**
  Removes all spaces to the left of the first nonspace character.

- **None**
  Removes no spaces.

- **Surrounding**
  Removes all spaces to the left of the first nonspace character and to the right of the last nonspace character.

- **Trailing**
  Removes all spaces to the right of the last nonspace character.

Type
The data type of the source field; this is used by EDGE to determine how to copy the data. Specific formats for each type are listed in Table 13-4; can be one of the following:

- **Alphanumeric**
  The field can contain any characters.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Counter</td>
<td>The field is to contain a sequential number, starting at one for the first record. The counter is incremented by one for each record read. (The number is not part of the source data.)</td>
</tr>
<tr>
<td>Date</td>
<td>The field contains a date. If a two-digit year is specified, the century is determined by the setting of <code>EDGE base year</code>.</td>
</tr>
<tr>
<td>Literal</td>
<td>The destination field is to contain a specified literal value. Every record contains the same value. (The literal is not part of the source data.) For exporting, <code>EDGE</code> supports the following escape sequences:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>\</td>
</tr>
<tr>
<td></td>
<td>BELL</td>
</tr>
<tr>
<td></td>
<td>FORMFEED</td>
</tr>
<tr>
<td></td>
<td>NEWLINE</td>
</tr>
<tr>
<td></td>
<td>RETURN</td>
</tr>
<tr>
<td></td>
<td>TAB</td>
</tr>
<tr>
<td></td>
<td>vertical tab</td>
</tr>
<tr>
<td>Numeric</td>
<td>The field contains numeric characters. To view the results of various settings for numeric types, see <code>Table 13-3</code>.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replicate</td>
<td>The destination field is to contain a sequential number, starting at one for the first record, up to a specified limit. Whenever the limit is reached, the counter is reset to one and incremented for each succeeding record until the limit is again reached. (The number is not part of the source data.) For example, if you specify 10 as the limit, the records are numbered as they are exported, starting with 1 and going up to 10, then starting over again at 1. If the file has 100 records, when the export is finished there will be 10 records with the number 1, 10 records with the number 2, and so on.</td>
</tr>
<tr>
<td>Time</td>
<td>The field contains a time value. The source data is not validated to ensure it meets the characteristics of the specified type.</td>
</tr>
<tr>
<td>Value</td>
<td>The value in a multivalued field to use as the source of the data. This prompt is displayed only if the field is defined as multivalued.</td>
</tr>
</tbody>
</table>
Table 13-4. Export Format Options by Data Types

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alphanumeric</td>
<td>Capitalize</td>
<td>Capitalizes the first letter of each word. A word is defined as a group of characters delimited by spaces, a punctuation mark, or end of line mark. If the word begins with a nonletter character, the first letter character is capitalized; for example, 4ever is converted to 4Ever.</td>
</tr>
<tr>
<td></td>
<td>Lowercase</td>
<td>Converts all letters to lowercase.</td>
</tr>
<tr>
<td></td>
<td>None</td>
<td>Does no conversion of the source data.</td>
</tr>
<tr>
<td></td>
<td>Uppercase</td>
<td>Converts all letters to uppercase.</td>
</tr>
<tr>
<td>Date</td>
<td>None</td>
<td>The source is not converted. The data is displayed in EDGE internal format.</td>
</tr>
<tr>
<td></td>
<td>DD-MM-YY</td>
<td>The date is in international numeric format with a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>DD-MM-YYYY</td>
<td>The date is in international numeric format with a four-digit year.</td>
</tr>
<tr>
<td></td>
<td>DD-MMM-YY</td>
<td>The date is in international format with a three-letter abbreviation for the month and a two-digit year.</td>
</tr>
<tr>
<td></td>
<td>DD-MMM-YYYY</td>
<td>The date is in international format with a three-letter abbreviation for the month and a four-digit year.</td>
</tr>
<tr>
<td>Data Type</td>
<td>Format</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MM-DD-YY</td>
<td>The date is in US numeric format with a two-digit year.</td>
<td></td>
</tr>
<tr>
<td>MM-DD-YYYY</td>
<td>The date is in US numeric format with a four-digit year.</td>
<td></td>
</tr>
<tr>
<td>MMM-DD-YY</td>
<td>The date is in US format with a three-letter abbreviation for the month and a two-digit year.</td>
<td></td>
</tr>
<tr>
<td>MMM-DD-YYYY</td>
<td>The date is in US format with a three-letter abbreviation for the month and a four-digit year.</td>
<td></td>
</tr>
<tr>
<td>month-DD-YY</td>
<td>The date is in US format with the month spelled out and a two-digit year.</td>
<td></td>
</tr>
<tr>
<td>month-DD-YYYY</td>
<td>The date is in US format with the month spelled out and a four-digit year.</td>
<td></td>
</tr>
<tr>
<td>YYMMDD</td>
<td>The date is in numeric format starting with a two-digit year and no separator characters.</td>
<td></td>
</tr>
<tr>
<td>YYYYMMDD</td>
<td>The date is in numeric format starting with a four-digit year and no separator characters.</td>
<td></td>
</tr>
</tbody>
</table>
This is the EDGE internal format, which is the number of days since December 31, 1967.

A format that you define. You can use the following characters:
- DD: Day of the month.
- MM: Month as a number.
- MMM: Month as a three-letter abbreviation.
- mont: Month spelled out.
- YY: Last two digits of year.
- YYYY: Complete year.
- -: Delimiter; can be any character.

For example, MMM YYYY would produce a date of Oct 2001.

The source is not converted. The Decimal property has no effect on the data.
### Table 13-4. Export Format Options by Data Types (continued)

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Explicit precision</td>
<td>The decimal point for numbers is explicit and is stored in the data. The value is truncated to the number of decimal places specified in the Decimals property.</td>
</tr>
<tr>
<td></td>
<td>Implicit precision</td>
<td>The decimal point for numbers is implicit. To specify the number of decimal places, use the Decimal property. If the data contains a decimal point, the Implicit format has no effect.</td>
</tr>
<tr>
<td></td>
<td>Truncate</td>
<td>The value is truncated to the number of decimal places specified in the Decimals property.</td>
</tr>
<tr>
<td>Time</td>
<td>None</td>
<td>The source is not converted. The data is displayed in EDGE internal format.</td>
</tr>
<tr>
<td></td>
<td>HH:MM</td>
<td>The hours and minutes, in 24-hour format; for example, 16:05.</td>
</tr>
<tr>
<td></td>
<td>HH:MM:SS</td>
<td>The hours, minutes, and seconds, in 24-hour format; for example, 16:05:30.</td>
</tr>
<tr>
<td></td>
<td>HH:MM:ampm</td>
<td>The hours and minutes, in 12-hour format; for example, 04:05pm.</td>
</tr>
</tbody>
</table>
### Table 13-4. Export Format Options by Data Types (continued)

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Format</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>HH:MM:SSampm</td>
<td>The hours, minutes, and seconds, in 12-hour format; for example, 04:05:30pm.</td>
</tr>
<tr>
<td>User defined</td>
<td>User defined</td>
<td>A format that you define. You can use the following characters:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- HH Hour.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- MM Minutes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- SS Seconds.</td>
</tr>
<tr>
<td></td>
<td>amp m</td>
<td>- Time in 12-hour format.</td>
</tr>
<tr>
<td></td>
<td>:</td>
<td>- Delimiter; can be any character.</td>
</tr>
<tr>
<td></td>
<td>For example,</td>
<td>For example, HH.MM would produce a time of 16.05.</td>
</tr>
<tr>
<td></td>
<td>HH.MM</td>
<td></td>
</tr>
</tbody>
</table>
# Export Commands

The following commands are available on the Export menu:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Begin Export</strong></td>
<td>This command begins the export process.</td>
</tr>
<tr>
<td>Clear All Processing Errors</td>
<td>This command clears the error processing logs for all export definitions in</td>
</tr>
<tr>
<td></td>
<td>the current project.</td>
</tr>
<tr>
<td>Clear Processing Errors</td>
<td>This command clears the error processing log for the currently selected</td>
</tr>
<tr>
<td></td>
<td>export definition. If no export definition is selected, this command is</td>
</tr>
<tr>
<td></td>
<td>not available.</td>
</tr>
<tr>
<td>List All Export Device Formats</td>
<td>This command lists all the export definitions for the current project.</td>
</tr>
<tr>
<td>List Current Export Device Format</td>
<td>This command lists the currently selected export definition.</td>
</tr>
<tr>
<td>List Export Device Records</td>
<td>This command displays the source records. This display is the same as</td>
</tr>
<tr>
<td></td>
<td>clicking Yes at the Display tape records prompt when you begin the export.</td>
</tr>
<tr>
<td>List All Export Processing Errors</td>
<td>This command lists the error processing logs for all the export definitions</td>
</tr>
<tr>
<td></td>
<td>in the current project.</td>
</tr>
</tbody>
</table>
Importing and Exporting

Export Commands

Note: For errors to be logged, you must specify Log Errors in the instruction before you execute the export.

Begin Export

When the Begin Export command is selected, a dialog box similar to the following is displayed:

![Dialog box showing export options]

**Properties:**

- **Begin export**: Begins the export, using the properties checked.
- **Export name**: The name of the file to export.
The source file is read and statistics similar to the following are displayed:

**Properties:**

- **Rewind tape before starting.** When checked, specifies to rewind the tape before export. Meaningful only if the specified path is to a rewind tape drive.

- **Selection list** The selection list to use in selecting records to export; enabled only when the Use selection list property is checked.

- **Use selection list** When checked, allows you to specify the name of the selection list to use in selecting records to export.
Importing and Exporting

Export Commands

- **Export date to Device**
  - Transfer records from file to I/O device
  - Mode: **LIVE**
  - I/O Device ID: **CUSTOMERS**
  - File: **CUST.LIST**
  - Format: **ASCII**

- **Record ID:**
  - Records transferred: 30
  - Records w/ errors: 0

- **Total records processed:** 30 (updated every 100 records)

- **Elapsed time (HH:MM:SS):** 00:00:10
  - Transfer rate: 18 0.00 records per minute

- **Transfer complete**
- **Press ENTER to continue:**

*white text*
SmartMerge provides additional importing and exporting functionality for EDGE. It is similar to the standard EDGE import/export feature, with the following additional features:

- SmartMerge can import into or export from more than one external database file at a time.
- Parent-child selection relationships among files can be defined to fully utilize the SmartMerge export function.
- Each external source or target is identified by a unique name.
- A record header uniquely identifies each record in the various SmartMerge files.
- A copy of each exported record is maintained for each external source to which the record is exported. If the same record is later imported, the imported version is compared to the saved version so that only modified or new records are actually added to the EDGE database during the import process. Records can be modified by EDGE or any of the external sources.
- Optional Record Tracking can exclude unchanged records and deleted records from the import and export process.
SmartMerge is ideal to handle very large database files where only a portion of the file is modified or updated at a time because it processes only records that have been changed or added. SmartMerge is also suited for importing and exporting records among multiple internal and external files.

**Record Tracking**

Record tracking can be used with SmartMerge to perform these tracking and saving options:

- Track when and in what system (EDGE or external system) a record was last updated.
- Track whether a record has been deleted.
- Save a copy of deleted records. (This option is not functional in the current version of SmartMerge.)

Record tracking saves time, communication costs, and file space by excluding unchanged records from the export process. Record tracking also maintains the accuracy of your database by excluding deleted records from the import process. Record tracking is especially useful for large databases that are relatively stable.

Record tracking is enabled by using the `set_tracking` command.
Tracking information is stored in an EDGE file called S_SHADOW. S_SHADOW file records consist of the fields listed in Table 13-5.

### Table 13-5. S_SHADOW Records

<table>
<thead>
<tr>
<th>Field Position</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Record identifier. This is always R.</td>
</tr>
<tr>
<td>2</td>
<td>Filename.</td>
</tr>
<tr>
<td>3</td>
<td>Record ID.</td>
</tr>
<tr>
<td>4</td>
<td>Status of the record, either E for existing or D for deleted.</td>
</tr>
<tr>
<td>5</td>
<td>For existing records, the date and time the record was last modified. For deleted records, the date and time the record was deleted.</td>
</tr>
<tr>
<td>6</td>
<td>Name of the system on which the file was updated, either EDGE or an external system.</td>
</tr>
</tbody>
</table>

Sample S_SHADOW file records are shown below:

```
R ACT 1 E 04/04/97 07:17 EDGE
R ACT 2 D 04/03/97 11:31 EDGE
```

To display records in this file, use the ACL command `dump_shadow`. 
SmartMerge Importing

SmartMerge importing is similar to the standard EDGE import feature, with the following added functionality:

- Records can be imported into multiple database files
- Records can be compared to the versions that were exported to merge only the changed fields.

The steps for performing a SmartMerge import are listed below and described in the following sections.

1. **Create a Standard Import Definition** for each file that is affected by the import.
2. **Create a SmartMerge Import Definition**.
3. **Begin the SmartMerge Import**.

As records are read from the import file, they are merged into the database on a field-by-field basis. For records that were created on EDGE, only the fields that have been modified since they were exported are written back to the EDGE database. To determine if a record has been modified, the record is compared to the record that was saved by the export process.

Several special cases of imported records are described below:
• If a record that was exported from EDGE then deleted in EDGE is modified by the external database, that record will be added to the EDGE database if the record is included in the import file.

• If a record that was exported from EDGE is imported using an import definition that does not include every field of the record, any unspecified field is added to the EDGE database with the value it contained when it was originally exported, even if that field has since been modified by EDGE.

• If a new record is imported into EDGE from an external database using an import definition that does not include every field of the record, any unspecified field is left blank, as EDGE cannot determine the value of the field.

• If the same field in a record has been modified in EDGE and in the external database, the merging process retains the EDGE version of the field and discards the external database changes. If you want the external database changes to be retained instead, use the DB Device Override module to specify this.

Create a Standard Import Definition

Create a standard import definition for each file into which you are importing data.
The following properties, which are defined on the Import Selection Criteria property sheet, are ignored by the SmartMerge command; however, valid information must be entered into all required properties:

- Skip.
- Read every.
- Record type.
- Input character conversion.

The following standard import definition properties are overridden by the corresponding values in the SmartMerge import definition; however, valid information must be entered for these properties:

- Data format.
- From device.
- Error limit.
- Log errors.
- Record format.
- Record limit.

For information on creating import definitions, see the section Import Instructions.
Note: If your import data is alphanumeric, it is strongly recommended that you select the trim option trailing. (Trailing removes all spaces to the right of the last nonspace character. This ensures that trailing spaces do not adversely affect the success of field comparisons.)

Create a SmartMerge Import Definition

The SmartMerge import module is used to set up SmartMerge import definitions. When you select this module, a property sheet similar to the following is displayed:

Properties:

Data format The format of the source data; can one of the following:
### Properties: (continued)

- **ASCII labeled**
- **ASCII unlabeled**
- **EBCDIC labeled**
- **EBCDIC unlabeled**

If the data is in EBCDIC format, it is converted to ASCII when imported.

The default is ASCII unlabeled.

- **Error limit**

  The number of errors that can be encountered by SmartMerge before the import process is stopped. The default error limit is 100. If the number of errors exceeds the limit, the import is stopped and the following message is displayed:

  > Process aborted. Errors exceeded limit.

  Records that were imported before the error limit was reached are left in the destination file.

- **From device**

  The name of the I/O Device instruction to be used as the source of the data to be imported.

- **Header**

  Unique tag given to records in the import files. When importing, you must use the same header that was defined in the export process.

- **Import name**

  The names of the import instruction used to do the actual importing.
Begin the SmartMerge Import

Properties: (continued)

- **Log errors**: Specifies whether errors that are encountered during the import process should be written into the log file. (The log file is called TAPE.ERRORS and is in the current directory.)
- **Record format**: The format of the source records. For SmartMerge, all records have a fixed format and this is display only.
- **Record limit**: The maximum number of records to be written to the destination file. If the source file contains fewer records, no error is noted. Blank specifies no limit.

The Begin Import command on the Import tab sheet executes the selected SmartMerge import instruction. A dialog box similar to the following is displayed:
**Properties:**

- **External database name**
  - If the records being imported are from an external database, enter the name of the database. SmartMerge uses this information to determine which records need to be updated.

- **Rewind tape before starting**
  - Rewinds the tape. Meaningful only if your data is on tape and the specified path is to a rewind tape drive.

- **SmartMerge import name**
  - The name of the SmartMerge definition.

- **Test mode**
  - Tests the import; does not actually write any records.
**SmartMerge Exporting**

SmartMerge exporting is similar to the standard EDGE export feature, with the following added functionality:

- Can include records from multiple database files.
- Can select records based on matching parent-child relationships field values from selected records.
- Can store a copy of each exported record.
- With record tracking, can include only the records that have changed since the last export to a particular external source.

The steps for performing a SmartMerge export are listed below and described in the following sections.

1. **Create a Standard Export Definition** for each file that is affected by the export.
2. **Create a SmartMerge Export Definition**.
3. **Enable Record Tracking**, if desired. (Ideally, record tracking should be enabled when a file is created.)
4. **Begin the SmartMerge Export**.

**Create a Standard Export Definition**

Create an export definition for each file from which you are exporting data.
The following standard export properties are overridden by the corresponding values in the SmartMerge export definition; however, valid information must be entered into all required properties:

- Block size.
- Data format.
- Error limit.
- Log errors.
- Record format.
- To device.

For information on creating export definitions, see the section Export Instructions.

Create a SmartMerge Export Definition

The SmartMerge Export command is used to set up SmartMerge export definitions. When you select this command, a property sheet similar to the following is displayed:
**Properties: General**

- **Block size**
  The number of characters in a destination block.

- **Data format**
  Specifies whether the data in the destination file is to be written in ASCII or EBCDIC format. The default is ASCII.
### Properties: General (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Error limit</strong></td>
<td>The number of errors that can be encountered in the export process before it is stopped. The error limit value applies to the entire export process; it is not a per file value. The default error limit is 100. If the number of errors exceeds the limit, the export is stopped and the following message is displayed: Process aborted. Errors exceeded limit.</td>
</tr>
<tr>
<td><strong>From device</strong></td>
<td>The name of the I/O device.</td>
</tr>
<tr>
<td><strong>Log errors</strong></td>
<td>Specifies whether errors that are encountered during the export process should be written into the log file. (The log file is called TAPE.ERRORS and is in the current directory.)</td>
</tr>
<tr>
<td><strong>Record format</strong></td>
<td>Must be fixed length; this is display only.</td>
</tr>
<tr>
<td><strong>Record limit</strong></td>
<td>The maximum number of records to be written to the output device. If the source file contains fewer records, no error is noted. If this field is empty, there is no limit on the number of records to be exported.</td>
</tr>
</tbody>
</table>

### Properties: Export Definition

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Export name</strong></td>
<td>The name of a standard export definition. This definition is the parent definition.</td>
</tr>
</tbody>
</table>
### Properties: Export Definition (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header</td>
<td>A tag that is prefixed to each record in the standard export definition. The header for each standard export definition must be unique and cannot be a prefix of another header. For example, the headers EX and EXPORT are not considered unique; the headers ET and EX are considered unique.</td>
</tr>
<tr>
<td>Selection list</td>
<td>The name of the selection list used to select records from the file being exported by this definition. To select all records from the file enter the word ALL. If this property is left blank, only those records selected from the parent-child relationships are exported.</td>
</tr>
</tbody>
</table>

### Properties: Parent/Child

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Child field</td>
<td>The field from the child file to compare with the parent field to select associated child records.</td>
</tr>
<tr>
<td>Export name (containing child)</td>
<td>The name of a standard export definition. This definition is a child definition.</td>
</tr>
<tr>
<td>Parent field</td>
<td>The field from the parent file that is used to select associated records from the child files. This field is associated with the currently selected export name in the export definition property.</td>
</tr>
</tbody>
</table>
**Begin the SmartMerge Export**

When a SmartMerge export instruction is selected on the Export tab sheet, the Begin Export command executes that SmartMerge command. A dialog box similar to the following is displayed:

![Dialog box for SmartMerge export options](image)

**Properties:**
- **External database name:** If the records being exported are to be used to update an external database, enter the name of the database. SmartMerge uses this information to determine which records need to be updated.

---

**Note:** When specifying a parent or child record ID as the field name, you can use the word KEY instead of the actual record ID name.
When a SmartMerge Begin Export command is executed, EDGE performs the following functions:

- EDGE saves a copy of each exported record to support the import merge feature. If these records are later imported, only the fields that have changed are written to the EDGE database.

**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rewind tape before starting</td>
<td>Rewinds the tape. Meaningful only if your data is on tape and the specified path is to a rewind tape drive.</td>
</tr>
<tr>
<td>Test mode</td>
<td>Tests the export; does not actually write any records.</td>
</tr>
<tr>
<td>Turn off data checking</td>
<td>Turns off Record Tracking for the current export.</td>
</tr>
<tr>
<td>Write Mode</td>
<td>Determines how to update the file; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Append to existing file.</td>
</tr>
<tr>
<td></td>
<td>Overwrite existing file.</td>
</tr>
</tbody>
</table>

**Note:** If the same record is exported to more than one external database, a separate copy is saved for each database.
• If Record Tracking is enabled, only records that have been modified since the last export to the external system are exported.
External DB Override

The external DB override command is used to specify the names of external databases whose changes are to override changes made on the EDGE server.

When you select this option, a list of currently defined overrides is presented. To specify an external database override, select an existing definition, or click New. A dialog box similar to the following is displayed:

![Dialog box for specifying external database override]

Properties:
- **External database device**: The name of the I/O device whose changes are to override the EDGE server changes.
**Importing and Exporting External DB Override**

When a new record is created, EDGE automatically creates a record ID for it using the SmartMerge definition and external database name separated by a pipe symbol (|).

### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Import names</td>
<td>The names of the import instruction used with SmartMerge.</td>
</tr>
<tr>
<td>SmartMerge import name</td>
<td>The name of the SmartMerge definition.</td>
</tr>
</tbody>
</table>

When a new record is created, EDGE automatically creates a record ID for it using the SmartMerge definition and external database name separated by a pipe symbol (|).
Queues

Overview

The dictionary defines a queue as a file or line of people waiting their turn. In EDGE, a queue is an ordered list of telephone numbers, customer account numbers, or any other unique identifier waiting to be processed.

Queues allow you to organize and manage your projects and give agents access to records in an orderly manner. You can create queues with special characteristics, such as being ordered by date and time (which is referred to as a time-ordered queue). You can use a time-ordered queue, for example, to allow your agents to make calls to customers based on a schedule.

The entries in queues are referred to as call events. Each call event is associated with a record in your data file, usually a customer record. You define which call events go into a queue and how the call events are removed. EDGE handles the actual access.
You can initially place call events in queues through the queue seeding process, which creates call events from your data files based on criteria you specify. You can also use a guide or task to add or remove a call event from a queue as appropriate.

This chapter discusses the following:

- Design Considerations for Queues.
- Queue Definitions.
- Queue Seeding.
- Queue Utilities.
Queues Design Considerations for Queues

Queues are an integral part of an EDGE project and form the basis for managing your events.

Queues are defined at the project level. A project might have queues for callbacks, orders, wrong numbers, and finished events. For example, if an agent retrieves a call event, makes a call, then gets a busy signal or no answer, the call event can be assigned to a queue named RECALL, with a result to automatically reschedule the call event for a later time. If the call event were completed successfully, it might be assigned to a queue named FINISHED.

The following is a list of things to consider before designing your queues:

- Which Queues to Create.
- Who Can Access Each Queue.
- How to Organize Queues.
- How to Add Call Events to Queues.
- How Call Events are Removed from Queues.

Which Queues to Create

As you determine your Project Design, you must consider the types of events the agents will handle, where the events come from, and
how the results will be handled. The answers to these questions determine the types of queues you will need.

A call event can reside in only one queue at a time, but there can be multiple call events for a customer. Each of these related call events can reside in a different queue.

Queues do not necessarily have to be designed or implemented all at once. For example, as your business changes, you may find that you have need of a queue for a special campaign that will run for a limited time. EDGE allows for this; you can add or delete queues at any time.

You can design a queue so that all agents can be given access to call events in that queue. You can also define a queue so that access to it is restricted to specific agents or specific groups of agents. Queues that are restricted to specific agents or specific groups are referred to as protected queues. Protected queues are commonly used when agents work on commission or have their own customers.

Agents can add call events to any queue, but can only remove call events from a queue if they have explicitly been given access to the queue.

For example, if you have several agents who are to work only on initial contacts, you can create a group queue containing call events for
the initial contacts, define a group with the desired agents in it, and give access to the queue only to that specific group. No other groups or agents will be able to remove call events from that queue. (For information on setting up agent groups, see Chapter 3.)

You create a protected queue by the way you name the queue. To create a protected queue to which specific agents will be given access, append /U to the queue name when you create the definition. To create a protected queue to which groups of agents will be given access, append /G to the queue name.

Note: Remember that only the agent assigned to an agent-protected queue can retrieve call events from that queue. If other agents need access to those call events, the call event must be transferred to another queue.

Unless a queue is specifically restricted when it is defined, any agent can be given access to it. However, no one can access any queue until given explicit access to it.

**How to Organize Queues**

Call events can be organized within queues in the following ways:

- First-in, first-out order (FIFO).
- No specific order (randomly ordered).
- By time and date (time-ordered).
FIFO queues can be used, for example, to manage cold calls, follow-ups to orders, or any time the action should be done in sequence.

Randomly ordered queues might be used for events that need some maintenance, such as a queue that handles wrong numbers.

Time-ordered queues (Most recent and Next available) can be used to manage scheduled appointments. The time can be set automatically by EDGE, manually by the agent, or using logic.

Call events can be added to queues in any of the following ways:

- By Queue Seeding. Queue seeding takes information from database records, creates a call event for each record, and places the call event in a queue. Queue seeding can be used, for example, when you import records from your mainframe and you want to use EDGE to make the records available to your agents.

- By specifying a queue in the Results for a call.

- By specifying a queue name in the $QUEUE system field.

- By using the CALL-EVENT verb. The CALL-EVENT verb enables you to manage call events, including adding call events to queues.
**How Call Events are Removed from Queues**

Before we discuss how to list your queues so that the call events are accessed in the proper order, let’s see what occurs when an agent requests a call event using the Next Call button on the Call Center screen; EDGE automatically performs the following steps:

1. Selects a call event from a queue (as described in the following section).

2. Removes the call event from its queue. Removing the call event from a queue enables the agent to work with the record associated with the call event.

3. Updates fields in the **SYS File** ($CALLEVENT, $CALLID, $CEDELIM, $COUNTRY, $EVENTINDEX, $OLDQUEUE, $OLDRESULT, $PCBDATE, $PCBTIME, $SITETZ, $THEIRDATE, $THEIRTIME).

4. Initializes some fields in the **STATUS File** (QUEUE, QNAME, RESULT, LASTSCREEN, SCHED.DATE, SCHED.TIME).

5. Updates other fields in the STATUS file (CODE, DATE, OPID, PORT, START).

6. Routes to the first outbound screen as defined in the **Guide Options** and displays the record (typically a customer data record) associated with the call. Only the fields on the screen that are in the file from which the record was queue seeded display automatically.
Accessing Call Events in the Correct Order

There are several considerations in determining which call event will be selected when an agent selects Next Call from the Call Center screen:

- The **Ordering** of the queue.
- The **Removal** strategy of the queue.
- The **Priority group** of the queue.

It is easiest to understand how EDGE determines which call event to retrieve by discussing guides that do not use priority groups separately from guides that use them.

**Note:** For the purpose of the next sections, the term valid call event is used to specify call events in a queue that are ready to be removed based on the queue’s definition. For example, in time-ordered queues, valid call events are those that are scheduled for the current time or for earlier than the current time. For a FIFO queue, all events are valid; when the FIFO queue is empty, it no longer has valid call events.
Guides Not Using Priority Groups

When priority groups are not used EDGE performs the following steps to determine which call event to retrieve:

1. EDGE selects the first queue from the Search queues defined in the agent’s operator access. Typically time-ordered queues are listed first, then the FIFO queues and None queues.

2. If the queue has a valid call event, it is removed.

   If the queue is time ordered, and there are several valid call events (have a scheduled time before or at the current time), the Removal strategy of the queue determines which call event will be retrieved first.

3. The call event is processed by the agent using the guide.

4. When the agent exits the calls and returns to the Call Center screen and again selects the Next Call button, EDGE again selects the first queue and determines if it contains any valid call events.

5. If so, that event is removed, and steps 3 and 4 are repeated until there are no longer valid call events in the queue.

6. When a queue no longer has valid events, EDGE moves on to the next queue in the agent’s Search queue list and repeats steps 2-5. This process repeats with all of the queues in the Search queue list.
7. When there are no valid calls in any of the queues in the agent’s Search queue listing, the agent receives an error message.

Note: Each time the agent selects Next Call, all queues from the top of the Search Queue list are checked in sequence for valid calls. A FIFO queue would have to be empty before the queue following it in the list is accessed. This is why time-ordered queues are typically placed first in the queue listing.

Guides Using Priority Groups

1. EDGE retrieves the list of Search queues defined in the agent’s operator access. Typically time-ordered queues are listed first, then the FIFO queues and None queues.

2. EDGE verifies whether a Priority group has been assigned to the first queue in the list. If no priority group is assigned, see the section Guides Not Using Priority Groups.

If a priority group is defined for the queue, EDGE determines if the next queue in the list has the same priority group number. If so, EDGE determines whether the next queue in the list has the same priority number, and so on, until EDGE reaches a queue that uses a different number.

For example, Figure 14-1 shows two lists of search queues for two agents. The lists contain the same queues with the same priority groups, but are listed in slightly different order. For
Agent 1, the queue CBACK/U is not listed sequentially with the other queues assigned a priority group number of 1. Therefore, it is not considered as part of the same subset.

3. EDGE now has a subset of the search queue list that has been defined as having the same priority. EDGE will work within that subset of queues until none of the subset queues have valid call events.

4. From this subset of queues (those that sequentially have the same priority group), EDGE randomly selects a queue.

5. If the queue has a valid event, it is removed.

   If the queue is time ordered, and there are several call events that are currently scheduled (have a scheduled time before or at the current time), the Removal strategy of the queue determines which call event will be retrieved first.

6. The agent processes the removed call event in the guide, then returns to the Call Center screen and again selects Next Call.

7. The next step varies depending on whether the default behavior is used or the environment variable SHUFFLE_ALL=1 has been set:

   If the default behavior is used (the environment variable SHUFFLE_ALL is not set), EDGE stays with the same queue until it no longer has valid call events. In the example shown in Figure 14-1, for Agent 2, if queue CBACK_WEST is
selected first, the other callback queues will not be accessed until all the callbacks from CBACK_WEST are dialed (assuming the queue is open and has valid call events).

Then EDGE moves to the next queue in the list and one-by-one removes all of its call events for the agent, until it no longer has valid call events.

And so on until EDGE has processed the entire subset list of queues.

When none of the subset list (all queues with the same priority level) have valid call events, EDGE moves on to the next queue in the Search queue list, and steps 2-7 are repeated.

If the environment variable SHUFFLE_ALL is set to 1, EDGE again takes the subset list of queues and randomly picks a queue.

Once the queue is picked, a valid call event is selected from the queue, and is processed by an agent.

Each time the agent selects Next Call, a new queue is selected. In the example shown in Figure 14-1, for Agent 2, if queue CBACK_WEST is selected the first time, it may or may not be selected the next time the agent selects Next Call.
When none of the subset list have valid call events, EDGE moves on to the next queue (which is one that has a different priority group number), and steps 2-7 are repeated. In the example shown in Figure 14-1, for Agent 1, the subset that consists of CBACK_EAST and CBACK_WEST must no longer have valid call events before the next subset (which consists of EAST_FIFO and WEST_FIFO) is used.

8. When there are no valid calls in any of the queues in the agent’s Search queue listing, the agent receives an error message.
Figure 14-1. Grouping Search Queues by Priority Group
Queue Definitions

The Queue Definitions property sheet is used to define and modify queue definitions. A queue definition is needed for each queue you intend to use.

Whenever a queue definition is modified, the information is immediately available in operations.

To define a group-protected queue, append /G to the queue name; for example, CONTACT/G. (The /G counts as two characters out of the total of 10 characters allowed for the name.)

To define an agent-protected queue, append /U to the queue name; for example, CALLBACK/U. (The /U counts as two characters out of the total of 10 characters allowed for the name.)

When the first call event from a group is placed in a group-protected queue, the /G in the queue name is replaced with a /* followed by the group name; for example, CONTACT/*SOUTH. When the first call event from an agent is placed in an agent-protected queue, the /U in the queue name is replaced with a / followed by the user ID; for example, CALLBACK/JON.

To display the queue definition page, select the following:

Project / Queues area / Queue definitions tab
Adding or Modifying a Queue Definition

When you add or modify a queue definition, a page similar to the following is displayed:

Properties:

Access The options for call event availability to the agents; can be one of the following:

Closed The queue is closed and no call events can be removed from the queue; however, new call events can be added.

Open The queue is open and call events are available for removal from the queue.
Queues

Properties: (continued)

Opened by schedule

The queue is open and call events are available only during scheduled times. The queue is closed and call events cannot be removed from the queue outside of the scheduled times.

When Open by schedule is specified, the From/To grid is enabled to allow you to schedule the times the queue is to be open and the call events available. Specify up to three time periods.

The time periods must not span midnight; that is, do not indicate from 10:00 p.m. to 2:00 a.m. Instead, set up times from 10:00 p.m. to 11:59 p.m. and 12:00 a.m. to 2:00 a.m.

The times you specify in the boxes are based on the time at your site, not the customer’s time. For example, if the queue contains call events for customers in the Eastern time zone and the site is in the Pacific time zone, to open the queue to access calls to customers between 8 a.m. and 5 p.m. Eastern time, set the times from 5 a.m. to 2 p.m.

Description

The description of the queue. The description is intended for your own use. This property is optional.
**Properties:** (continued)

**Ordering**  The method for organizing call events in the queue; can be one of the following:

- **FIFO**  First in, first out ordering. Call events are added to the bottom of the queue and removed from the top of the queue.

- **None**  When the queue is seeded, each call event is assigned a random time and date and placed in the queue according to that time and date. The date range is from 0 to 16383 and the time range is from 0 to 86399.

- **Time**  The call events are organized by scheduled callback times. If more than one call is scheduled for the same time, the first one scheduled is the first available for removal at the scheduled time. For example, if 3 calls are scheduled for 3 p.m., the first call that was scheduled is the first call removed at 3 p.m., then the second, finally the third.

This property cannot be changed once you have saved the definition.

When a call event is added to a FIFO queue, if there is a date and time in the SCHED.DATE and SCHED.TIME fields, these fields are used in placing the call event in the queue. That is, the call event is placed according to the callback times, not at the bottom of the list in FIFO order.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority group</td>
<td>A number assigned to queues to group them together. The priority group number is used by EDGE when determining which queue (from the agent’s Search queue list) to pull calls from when an agent selects Next Call from the Call Center screen. This property does not override the priority of queue searches for agents. For example, in the list of search queues shown below, if PROSP.CA, PROSP.NY, and PROSP.TX have a priority group number of 1, any one of them can be accessed to provide the next call event. However, they would be accessed only if the RECALL/U queue had no calls. When you assign operator access, queues with the same priority group number must be listed contiguously for EDGE to recognize the association. For example, in the list above, if PROSP.CA and PROSP.TX are priority group 1 and PROSP.NY is priority group 2, EDGE will not recognize PROSP.CA and PROSP.OR as being in the same priority group.</td>
</tr>
</tbody>
</table>

![Queue List Example](image-url)
How priority groups work depends on whether the environment variable SHUFFLE_ALL=1 is set. The behavior varies as follows:

**SHUFFLE_ALL not set.** EDGE randomly selects a queue within the first priority group subset and stays with the same queue until it is empty or the agent logs out. EDGE moves to the next queue when original queue is empty or the agent logs out and back in.

**SHUFFLE_ALL=1.** EDGE randomly selects a queue within the first priority group subset, and selects a new queue from the same subset each time the agent selects Next Call.

This setting is useful, for example, if you have several queues with the same business priority, such as cold calls for multiple sales territories.

This property is optional.

For information on using priority groups, see the section Guides Using Priority Groups and Figure 14-1.

**Removal**

The method for removing call events from the queue; can be one of the following:
Properties: (continued)

Most recent
Removes the event scheduled for the time closest to the current time, and moves back in time retrieving available call IDs. Does not remove events later than the current time. Only available for time-ordered queues.

Next available
Removes the oldest scheduled event, and moves forward in time retrieving available call IDs. Does not remove events later than the current time. Only available for time-ordered queues.

Next
Removes the next event in the queue. This property is automatically used for FIFO queues. It can also be specified for time-ordered queues. With time-ordered queues, the oldest item (by date and time) is removed first.

Random
Generates a random time and date (using the date range of 0 to 16383 and the time range of 0 to 86399) and removes the event scheduled for that time or the next later time. This property is automatically used for queues with None as their ordering.

This removal strategy provides randomness to call event selection because the calls are placed in the queue in a random order and removed from the queue in a random order.
Deleting a Queue Definition

A queue definition can be deleted only if the corresponding queue is empty. If the queue is a protected queue, all the individual queues related to the protected queue must be empty.

You are then asked to verify the deletion:

Properties: (continued)

Send call IDs to new queue

The queue in which to place a call event after it has been removed from this queue. When a call event is removed from this queue, the following occurs:

- If the call event is set with a Results that specifies a queue destination, the call event is sent to that queue.
- If the result does not specify a queue, the call goes to the queue specified in this property.
- If no queue is specified and the queue is not set by the result, call events are returned to the queue from which they originated.

This property is optional.
Queue Definitions Menu

The Queue Definitions menu contains the following commands:

- **Clear Queue Problem Log.**
- **List Queue Definitions.**

To access the queue definitions menu, select the following:

Project / Queues area / Queue definitions tab

**Clear Queue Problem Log**

The Clear Queue Problem Log command clears the queue problem log, which contains any error that occurred when removing or placing a call from a queue. Before clearing, the errors should be checked to see if they indicate a problem with the queuing system, such as a design flaw in the ordering of each agent’s queues, or a corrupt EDGE system file.

This command is also available on the Queue Definitions tab.
When this option is selected, a prompt similar to the following is displayed:

![ETW for Windows dialog box](image)

When you select Yes, the log is cleared.

**List Queue Definitions**

The List Queue Definitions command displays the queues currently defined for this project. When list queue definitions is selected, a page similar to the following is displayed:
The following information is displayed:

Description  The optional description specified in the queue definition.

Next Queue   The queue calls taken from this queue are placed in if the result set for the call event does not specify a queue.
<table>
<thead>
<tr>
<th><strong>Ordering</strong></th>
<th>The method for ordering call events placed in the queue.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Queue ID</strong></td>
<td>The name of the queue.</td>
</tr>
<tr>
<td><strong>Removal Strategy</strong></td>
<td>The method for removing call events from the queue.</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>The defined access status (for example, open, closed) of the queue.</td>
</tr>
</tbody>
</table>
Queue Seeding

Queue seeding places call events into a queue as specified in the queue seeding instruction. It also creates a record in the STATUS File for each call event that is seeded and fills in the following fields:

- ALTPHONE.
- COUNTRY.
- INDEX (if appropriate).
- ITZC.
- SRC.QUEUE.
- TELNO.

If the target queue is a time-ordered protected queue and country and/or phone codes are defined for the user or group, the user or group time zone is used instead of the site time zone. Also, if a work schedule is defined for the associated user or group, then that work schedule is used instead of the default work schedule for the project.

To determine the time zone associated with a user or group, EDGE checks the agent record for country and phone codes first, then the group record, and then the site record. EDGE uses the first valid data encountered.
Telephone numbers are validated against telephone information specified in the Country Code Definition and the ACTZ file. The following chart depicts the validation done during queue seeding.

**Figure 14-2. Validation Process During Queue Seeding**

To display the queue seeding instruction, select the following:

Project / Queues area / Queue Seeding tab

When you add or modify a queue seeding instruction, a page similar to the following is displayed:
**Properties:**

- **Allow multiple call events per call ID**
  - Specifies whether a call event can be seeded if there is already a call event for the record in a queue. (The existence of call events is determined by records in the STATUS file.)
  - **Checked**  A call event is seeded even if there is already a call event for the record.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cleared</strong></td>
<td>If there is already a call event for the record, an error is reported in the log file and no call event is seeded for that record. If there is no call event, the call event is seeded. This property has no effect on using multiple call events outside of seeding. The default is cleared.</td>
</tr>
<tr>
<td><strong>Alternate phone number fields</strong></td>
<td>The fields in the record containing additional phone numbers that can be called if the number in the telephone number field is busy or does not answer. The specified alternate phone number fields can be multi-valued.</td>
</tr>
<tr>
<td><strong>Assign call ID from field</strong></td>
<td>The name of the field from which to assign call IDs. To have EDGE generate the call IDs, enter an * (asterisk). EDGE uses the Project Counter to generate call IDs. If blank, the record ID is used as the call ID.</td>
</tr>
<tr>
<td><strong>Constant country code</strong></td>
<td>The code for the country where the calls will be directed. If both this and the Country Code field properties are blank, the country specified in the site information is used. For more information, see the section Set Up Site Information.</td>
</tr>
</tbody>
</table>
Properties: (continued)

Copy field grid  The fields from the seeding data file that you would like copied to the RESPONSES File. Typically, data is copied to the RESPONSES File to maintain information on the original data in the record.

Copy field   The field from the file specified in the queue seeding instruction.

Do not specify the record ID as a source field.

Response field  Specifies the field in the RESPONSES File into which the source data is to be copied. If necessary, you can create additional fields in the RESPONSES file.

Do not specify the record ID as a destination field.

Value  The value position in the field in the Responses file into which the source is copied. For example, if the field is to be copied into the third position, enter 3. This property is required if copying into a multivalued field.

These properties are optional.

Country code field  The field that contains the code for the country where the calls will be directed. This property overrides the Constant Country Code property.
## Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File</td>
<td>Specifies the name of the file that contains the fields to be used in the seeding instruction. The file must be an EDGE file. If the calls are being directed to the country specified in the site configuration, leave this blank. This property must be specified before the fields can be specified.</td>
</tr>
<tr>
<td>Limit records to selection list</td>
<td>Specifies the name of a selection list that contains the record IDs to use in the seeding. If a selection list is specified, only those records listed in the selection list will be seeded. (To create the selection list, use Ad Hoc Reporting.)</td>
</tr>
<tr>
<td>Into queue</td>
<td>Specifies the name of the queue to be seeded. To seed to a protected queue, specify the name with the /U or /G as appropriate, then use the Distribution Properties tab to specify the exact queues. If a user or group protected queue is selected, the Distribution Properties tab becomes available. If a time-ordered queue is specified, the Time ordered queue tab becomes available.</td>
</tr>
<tr>
<td>Telephone number</td>
<td>The name of the field that contains the telephone number.</td>
</tr>
</tbody>
</table>
The Distribution tab is available only when the queue being seeded is a protected queue. If the queue is a group-protected queue, you can specify the groups for whom queues are to be created and seeded. If the queue is an agent-protected queue, you can specify the agents for whom queues are to be created and seeded.

To display the Distribution tab, select the following:

Project / Queues area / Queue seeding / Distribution tab

When you add or modify a seeding instruction which seeds into a user or group protected queue, a page similar to the following is displayed:
Properties:

By matching the value a specific field to a table

Seed the calls based on a field that is to be matched against values in the table. For example, if you want to create queues based on a field called REGIONS, you can specify the regions to select and the groups to which records with those regions are to be assigned. If a record has a value other than the ones specified in the table, no call event is created.

If selected, the following properties are available:
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Fieldname</th>
<th>The field that contains the group name or user ID to match in the table.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>The group or agent to assign the call event to.</td>
</tr>
<tr>
<td>Value</td>
<td>The value that is to be matched in the field.</td>
</tr>
</tbody>
</table>

**To specific group(s)**

Seed the calls evenly among all the specified agents or groups. Select the agent or group names in the presented list, and drag to the selected groups property.

If selected, the following property is available:

**Selected groups**

A list of defined groups or agents is displayed. To deselect a choice, highlight it in the list box, then press BACKSPACE or DELETE.

**Using the group name in a specific field**

Seed the calls to the queue for the agent or group specified in the field. The field is examined for each record seeded to determine which queue to place it in. For example, if the field GROUP is used, the value of GROUP is checked for each record and the call is seeded into the queue for the group specified in GROUP.

If selected, the following property is available:

**Fieldname**

The field that contains the group name or user ID.
Queues Queue Seeding

**Time-Ordered Queue Properties**

Note: The prompts show the word groups or operators, depending on the type of queue.

The Time-Ordered Queue property sheet specifies dates and times for the call events in the queue. This option is available only if the queue to be seeded is defined as a time-ordered queue.

If the target queue is a time-ordered protected queue and country and/or phone codes are defined for the user or group, the user or group time zone is used instead of the site time zone. Also, if a work schedule is defined for the associated user or group, then that work schedule is used instead of the default work schedule for the project.

To determine the time zone associated with a user or group, EDGE checks the agent record for country and phone codes first, then the group record, and then the site record. EDGE uses the first valid data encountered.

Note: The options **With random date** and **With random time** do not guarantee that call events are evenly distributed among the specified dates or times. For example, one date may have more call events assigned than another date.
When you select the Time-Ordered Queue tab, a page similar to the following is displayed:

**Properties:**

- Adjust date & time values to conform to the work schedule and/or preferred calling hours
- Specifies whether to adjust the scheduled dates and times according to the work schedule and preferred calling hours set for the project.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checked</td>
<td>Adjusts the scheduled dates and times according to the work schedule and preferred calling hours set for the project. Any date or time that falls outside the work schedule and preferred calling hours is scheduled for the beginning of the next work day.</td>
</tr>
<tr>
<td>Cleared</td>
<td>The scheduled dates and times are not adjusted; therefore, it is possible to schedule a call for a time when agents are not working.</td>
</tr>
<tr>
<td>Set schedule date</td>
<td>The source of the scheduled date; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>From field</td>
</tr>
<tr>
<td></td>
<td>Schedules the call events based on the date in the specified field.</td>
</tr>
</tbody>
</table>
Starting from date ___ using periodic value in field ___ or randomly over ___ workdays Assigns scheduled dates starting with the date given, and changing it either using the value given in the field or randomly over the specified number of workdays.

If a field is specified, the value in the field can be a number in the range 0-366, in which case that value is added to the starting date to get the scheduled date. Or, the value in the field can be of the form DAY-N, where DAY is the abbreviation of a day of the week in the local language, and N is the occurrence of that day to use.

For example, if the field contains WED-2, and the start date is the first of the month, the scheduled date is the second Wednesday of that month.
Properties: (continued)

With random date

Randomly distributes call events within the queue according to the specified beginning and ending dates. Multiple call events may be assigned to the same date.

The date can be in any valid format such as the following formats:

- 3/1/97
- 1 MAR 97
- Mar 1 1997

Set schedule local time

Specifies how the scheduled time is to be set. The time will be adjusted to the local time indicated by the area code; can be one of the following:

- From time value in field

Schedules the call events based on the time in the specified field.
Queue Seeding Menu

The Queue Seeding menu contains the following commands:

- List Seeding Error Detail.
- List Seeding Error Summary.
- Start Seeding Process.

List Seeding Error Detail

The List Seeding Error Detail command displays or prints all seeding errors.

Properties: (continued)

With random time

Randomly distributes call events within the queue according to the specified beginning and ending dates. Multiple call events may be assigned to the same date.

The time can be in any valid format such as the following formats:

1P
1 PM
1:00 P.M.
13:00
If there are errors, you should resolve them before you continue. If you do not resolve errors and conflicts, you may encounter problems when you try to use the queue. For example, if area codes are missing from telephone numbers, you might not be able to perform any outbound calling.

The possible errors are listed in Table 14-1.

The report is similar to the following:

```
DESCRIPTION OF SEEDING ERRORS

SEED....... CALL ID....... DESCRIPTION OF ERROR......................
INSTRUCTION

ALL.IN.ONE 1   CALL EVENT ALREADY EXISTS
ALL.IN.ONE 48  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 47  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 49  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 5   CALL EVENT ALREADY EXISTS
ALL.IN.ONE 50  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 51  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 52  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 46  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 45  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 44  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 43  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 42  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 41  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 40  CALL EVENT ALREADY EXISTS
ALL.IN.ONE 4   CALL EVENT ALREADY EXISTS
```
**List Seeding Error Summary**

The List Seeding Error Summary summarizes the seeding errors by error type. You can use this option to determine if errors occurred. If there are errors, you can then select the Display Seeding Error Detail option for a detailed listing of the errors.

The report is similar to the following:

![Report Preview](image)
Start Seeding Process

Start Seeding Process initiates the actual queue seeding. Before seeding your queues, be sure you have completed the following:

- Specified your host setup, including defining your Country Codes.
- Set up Calendar and Work Schedules.
- If you are seeding to protected queues, be sure you have set up the Distribution Properties.

When you select the start seeding command, a dialog similar to the following prompts you to enter the maximum number of errors before the seeding process is terminated:

![Queue Seeding Option](image)

When the seeding process is complete, a page similar to the following displays the number of call events seeded, the number of errors, the start seeding time, and the end seeding time.
**Queue Seeding**

Start Time: 1:16:53 PM  
No. Seeded: 72  
No. Errors: 23  
Message: 72 flushed  
End Time: 1:16:55 PM

Press RETURN to Continue:

---

**Table 14-1. Queue Seeding Errors**

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTEMPTED TO SCHEDULE A CALL BEYOND LAST CALLBACK DAY</td>
<td>The schedule date is beyond the last callback day.</td>
</tr>
<tr>
<td>CALL EVENT ALREADY EXISTS</td>
<td>A call event already exists for this call ID. To allow multiple call events, check the “Allow multiple call events” check box.</td>
</tr>
</tbody>
</table>
Table 14-1. Queue Seeding Errors (continued)

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INVALID AREA CODE/PREFIX FOR xxx</td>
<td>The value for the phone number or alternate phone number in the specified field does not exist in the ACTZ file. An ACTZ File Update is required.</td>
</tr>
<tr>
<td>INVALID COUNTRY CODE FOR xxx</td>
<td>The value for the country code in the specified field is invalid.</td>
</tr>
<tr>
<td>NULL CALL ID</td>
<td>The field used to create the call ID is empty.</td>
</tr>
<tr>
<td>NULL GROUP ID</td>
<td>A group-protected queue is being seeded and the group ID is empty.</td>
</tr>
<tr>
<td>NULL OPERATOR ID</td>
<td>An agent-protected queue is being seeded and the agent ID is empty.</td>
</tr>
<tr>
<td>NULL TELEPHONE NUMBER</td>
<td>The phone number field is empty.</td>
</tr>
<tr>
<td>PERIODIC DATE: EMPTY</td>
<td>The periodic date field is empty.</td>
</tr>
<tr>
<td>PERIODIC DATE: xxx</td>
<td>The periodic date field is invalid. The format for the date should be a number between 0 and 366 or in the format DAY-N where DAY is the local name for the day of the week (for example, Sunday could be entered as SUN in English or DOM in Spanish) and N is the week number. (The name can be in uppercase or lowercase.)</td>
</tr>
</tbody>
</table>
### Table 14-1. Queue Seeding Errors  (continued)

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHONE NUMBER IS EITHER LESS THAN xxx OR GREATER THAN yyy</td>
<td>The value for the phone number or alternate phone number in the specified field is invalid.</td>
</tr>
<tr>
<td>PHONE NUMBER NOT xxx DIGITS LONG</td>
<td>The value for the phone number or alternate phone number in the specified field is invalid.</td>
</tr>
<tr>
<td>PROBLEM WITH QUEUE: xxx UNABLE TO DELETE CALLID yyy</td>
<td>The queues file is corrupted or in serious need of a rebuild. A Qrebuild -A should probably be performed.</td>
</tr>
<tr>
<td>PROBLEM WITH QUEUE: xxx UNABLE TO LOCK STATUS RECORD for yyy</td>
<td>The queues file is corrupted or in serious need of a rebuild. A Qrebuild -A should probably be performed.</td>
</tr>
<tr>
<td>QUEUE ENTRY FOR CALL ID xxx HAD BAD FORMAT</td>
<td>The queues file is corrupted or in serious need of a rebuild. A Qrebuild -A should probably be performed.</td>
</tr>
<tr>
<td>TOO MANY ALTERNATE PHONE NUMBERS</td>
<td>The alternate phone numbers must take up no more than 3896 characters, including a delimiter character following each number.</td>
</tr>
</tbody>
</table>
Table 14-1. Queue Seeding Errors  (continued)

<table>
<thead>
<tr>
<th>Error Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>xxx CALL ID ALREADY EXISTS IN STATUS FILE</td>
<td>This message should only occur if a STATUS file entry was created in the middle of the seeding corresponding to the given call ID.</td>
</tr>
<tr>
<td>xxx NOT IN SEEDING FILE.</td>
<td>The record xxx does not exist in the customer file.</td>
</tr>
</tbody>
</table>
Queue Utilities

The Queue Utilities page and menu contain the following commands:

- Delete Call IDs.
- Rebuild Queues.
- Reschedule Call IDs.
- Transfer Call IDs.

The Queue Utilities menu also contains the following commands:

- List Callback Queue Call IDs by Date.
- List Queue Detail.
- List Queue Call ID Counts.
- List Queue Problem Log.

These queue utilities and reports are described in this section. To display the Queue Utilities page, select the following:

Project / Queues area / Queue Utilities tab

A page similar to the following is displayed:
Delete Call IDs

The Delete Call IDs command deletes call events from specified queues. The queue definition is not deleted. (The Qclear ACL command also deletes call events.)

When the call events are deleted, the associated records in the HISTORY, RESPONSES, STATUS, and STATUSID files are also deleted. User-defined records are not affected by this option.
To display the Delete Call IDs page, select the following:

Project / Queues area / Queue Utilities tab / Delete Call IDs command

A page similar to the following is displayed:

Properties:

Delete all call IDs

Specifications:

- Specifies whether to delete all the call IDs in one or all of the queues in the project. Specify which queues in the property Call IDs removed from queues.
**Properties: (continued)**

Delete selected call IDs  Specifies whether to delete only the call IDs listed or in the specified selection list.

Selection list  The name of the selection list containing the list of call events to be deleted.

Individual call IDs  The call events to be deleted. Enter the individual IDs or events, separated by a RETURN.

If a specified call event is not found in the specified queue, an error is returned.

Call IDs removed from queue(s)  The queue from which call IDs are to be deleted; can be one of the following:

All  Delete all or specified call IDs from all queue.

Queue name  Delete all or specified call IDs from this queue.

Protected queue  Delete all or specified call IDs from the specific group-protected or user-protected queue based on the value specified in the For which property.

For which The user ID or group for the protected queue specified in the Call IDs removed from queue property.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Unlock active call events</th>
<th>Specifies whether to delete call events, even if they are currently being used by another agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Checked</td>
<td>Call events are deleted, even if in use.</td>
</tr>
<tr>
<td>Cleared</td>
<td>Call events that are in use are not deleted.</td>
</tr>
</tbody>
</table>

*Note: You should not check the Unlock active call events box unless directed to do so by EDGE Customer Care.*

When a single queue is deleted, a page similar to the following is displayed:
When you delete all call IDs in all queues, a warning message similar to the following is displayed:

**Figure 14-3.  Clear a Queue Processing Screen**

When you delete all call IDs in all queues, a warning message similar to the following is displayed:

**Figure 14-4.  Clear All Queues Warning**
Rebuild Queues

The Rebuild Queues command rebuilds queues. For example, if your queues have been corrupted, perhaps because of a hardware failure, you can use this utility to recreate the queues. (The Qrebuild ACL command can also be used to rebuild queues.)

Caution! The rebuild process deletes all entries in the specified queues, then uses the entries in the STATUS file to rebuild the queues. Depending on the number of records in the STATUS File, this process may take several hours. Agents should not access the queues being rebuilt during this process.

Generally, you should perform this utility only when directed to do so by EDGE Customer Care.

To display the Rebuild Queues page, select the following:

Project / Queues area / Queue Utilities tab / Rebuild Queues command

A page similar to the following is displayed:
Properties:

- **Rebuild all queues**: Specifies whether to rebuild all queues.
- **Rebuild selected queues**: Specifies whether to rebuild only the specified queues.
- **Queue**: The queues to be rebuilt.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group/ User</td>
<td>If a protected queue is selected in the queue property, the group or user associated with the queue.</td>
</tr>
<tr>
<td>Place calls not currently in a queue into queue</td>
<td>The queue in which to place call events that are not currently in a queue. Typically these are calls that were left active by an agent (for example, by the agent breaking out of the guide during a call).</td>
</tr>
<tr>
<td>Unlock active call events</td>
<td>Specifies whether to rebuild call events, even if they are currently being used by another agent.</td>
</tr>
<tr>
<td>Checked</td>
<td>Call events are rebuilt, even if in use. The call ID is removed from the STATUSID File and the CODE field in the STATUS File is changed to D for deactivate.</td>
</tr>
<tr>
<td>Cleared</td>
<td>Call events that are in use are not rebuilt.</td>
</tr>
</tbody>
</table>

**Note:** You should not check the Unlock active call events box unless directed to do so by EDGE Customer Care.
After the queues are rebuilt, a page similar to the following is displayed:

![Rebuild Queues]

**REBUILD_QUEUES**

Current Activity: Rebuilding by a pass on the entire STATUS file.
Count: 144

Message: Total of 0 error.
Result: Total of 144 callids rebuilt.

Press RETURN to Continue:

If errors are reported, select List Queue Problem Log from the Queue Utilities menu to display the error log. This option is described in the section **List Queue Problem Log**.

**Reschedule Call IDs**

The Reschedule Call IDs command processes a time-ordered protected queue to ensure that calls are scheduled during preferred calling hours defined for the assigned agent or group. Optionally, this
form allows you to interactively confirm call rescheduling in the selected queue. (The Qresched ACL command can also be used to reschedule queues.)

To display the Reschedule Call IDs property sheet, select the following:

Project / Queues area / Queue Utilities tab / Reschedule Queue command

A page similar to the following is displayed:

![Reschedule Call IDs](image)

**Properties:**

- Confirm reschedules: Interactively prompts you to confirm or reject the new schedule on a call-by-call basis.
- Time-Ordered Queue: Name of the time-ordered protected queue
If interactive processing is requested, calls are identified by call ID. After the queues are rebuilt, a summary page displays to identify the following for each queue: queue name, count of calls in the queue, total rescheduling errors, and total number of calls rescheduled.

**Transfer Call IDs**

The Transfer Call IDs command transfers call events from one queue to another. (The Qtrans command can also be used to transfer call events.)

To display the Transfer Call IDs property sheet, select the following:

- Project / Queues area / Queue Utilities tab / Transfer Call IDs command

A page similar to the following is displayed:
Properties:

Destination queue

The destination queue for the call events being transferred:

To queue

Transfer the specified call events to this queue.

User/group

The user ID or group for the protected queue specified in the To queue property. This is only used with protected queues.

Source queue

The originating queue for the call events being transferred; can be one of the following:
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Any queue</td>
<td>Transfer the call event from whatever queue it is currently in.</td>
</tr>
<tr>
<td>From queue</td>
<td>Transfer the specified call events from this queue only. If a call event is not found in the specified queue, an error is returned.</td>
</tr>
<tr>
<td>User / group</td>
<td>The user ID or group for the protected queue specified in the From queue property. This is only used with protected queues.</td>
</tr>
<tr>
<td>Transfer all call IDs in a single queue</td>
<td>Specifies whether to transfer all the call IDs from a specific originating queue to a specific destination queue. Both a destination queue and source queue must be specified. This selection removes all call events from one queue and places them in another queue.</td>
</tr>
<tr>
<td>Transfer selected call IDs</td>
<td>Specifies whether to transfer the specified call IDs (either listed or in a selection list) to a specific destination queue. Specify the call IDs to be transferred in the properties Selection list or Individual Call IDs which are described below. This selection removes specified call events from one queue and places them in another queue; specify one of the following:</td>
</tr>
</tbody>
</table>
The List Callback Queue Call IDs by Date report lists call IDs within each queue by the scheduled callback date and time.

When this command is selected, a page similar to the following is displayed:

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection list</td>
</tr>
<tr>
<td>The name of the selection list containing the list of call events to be transferred.</td>
</tr>
<tr>
<td>Individual call IDs</td>
</tr>
<tr>
<td>The call events to be transferred. Enter the individual IDs or events, separated by a RETURN.</td>
</tr>
<tr>
<td>Unlock active call events</td>
</tr>
<tr>
<td>Specifies whether to transfer call events, even if they are currently being used by another agent</td>
</tr>
<tr>
<td>Checked</td>
</tr>
<tr>
<td>Call events are transferred, even if in use.</td>
</tr>
<tr>
<td>Cleared</td>
</tr>
<tr>
<td>Call events that are in use are not transferred.</td>
</tr>
</tbody>
</table>

Note: You should not check the Unlock active call events box unless directed to do so by EDGE Customer Care.
Properties:

Select queues

The queues that are to be included in the report; can be one of the following:

- **All**: All queues defined are included.
- **Queue**: The specified queues are included in the report.

Typically only time-ordered queues would be selected.

Include automatically scheduled callbacks

Specifies whether to include call events with automatically scheduled callbacks (EDGE system-assigned). For more information on automatically scheduling callbacks, see the discussion of **Results** in Chapter 24.

- **Checked**: Call events with system-assigned callbacks are included in the report.
Properties: (continued)

Cleared        Only call events with agent-assigned callbacks are included in the report.

Note: If a FIFO queue is selected, all callback date and times are system-assigned (set by EDGE to the date and time the call was seeded); therefore, for FIFO queues, the report is blank unless the property Include automatically scheduled callbacks is checked.

The report is similar to the following:
The following information is displayed.

- **Call ID**: The call ID.
- **Queue**: The queue the call ID is assigned to.
- **Sched CB date**: The scheduled call back date.
- **Scheduled time**: The scheduled call back time.
**List Queue Detail**

The List Queue Detail report lists each call event in each queue in the project in order by call IDs, similar to the following:

![Report Preview](image)

The following information is displayed:

- **Call ID**: The call ID.
- **Comments**: Comments from the STATUS file record for the call event.
List Queue Call ID Counts

The List Queue Call ID Counts report lists each queue in the project and the number of call events in each. The report is similar to the following:

<table>
<thead>
<tr>
<th>Queue</th>
<th>The queue the call ID is assigned to.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sched CB date</td>
<td>The scheduled call back date.</td>
</tr>
<tr>
<td>Scheduled time</td>
<td>The scheduled call back time.</td>
</tr>
</tbody>
</table>
### List Queue Problem Log

The List Queue Problem Log report lists a summary of errors encountered during any queue function other than seeding; for example, errors encountered by the `CALL-EVENT` verb are logged in this file.

<table>
<thead>
<tr>
<th>QUEUE</th>
<th>COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALL.IN.ONE</td>
<td>103</td>
</tr>
<tr>
<td>CALLBACK/DDG</td>
<td>1</td>
</tr>
<tr>
<td>CALLBACK/GST1</td>
<td>1</td>
</tr>
<tr>
<td>CALLBACK/GST7</td>
<td>5</td>
</tr>
<tr>
<td>DEAD</td>
<td>41</td>
</tr>
<tr>
<td>NO.SALE</td>
<td>4</td>
</tr>
<tr>
<td>RECALL</td>
<td>75</td>
</tr>
<tr>
<td>TOTAL</td>
<td>230</td>
</tr>
</tbody>
</table>

The following information is displayed:

- **Count**: The number of call events in the queue.
- **Queue**: The queue being counted.
The errors are listed in queue order. The report is similar to the following:

![Report Preview](image)

The following information is displayed:

<table>
<thead>
<tr>
<th>Log num</th>
<th>Queue</th>
<th>Date</th>
<th>Time</th>
<th>Call ID</th>
<th>User Port ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td></td>
<td>07-11-95</td>
<td>04:31PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>07-11-95</td>
<td>04:32PM</td>
<td>1</td>
<td>SF4</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>07-11-95</td>
<td>04:34PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>07-11-95</td>
<td>04:34PM</td>
<td>1</td>
<td>SF4</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>07-11-95</td>
<td>04:53PM</td>
<td>1</td>
<td>SF4</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>07-11-95</td>
<td>04:54PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>07-11-95</td>
<td>04:17PM</td>
<td>1</td>
<td>SF4</td>
</tr>
<tr>
<td>30</td>
<td></td>
<td>07-11-95</td>
<td>04:17PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>40</td>
<td></td>
<td>07-11-95</td>
<td>04:30PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>70</td>
<td></td>
<td>07-11-95</td>
<td>04:33PM</td>
<td>1</td>
<td>SF4</td>
</tr>
<tr>
<td>100</td>
<td></td>
<td>07-11-95</td>
<td>04:34PM</td>
<td>1</td>
<td>SF4</td>
</tr>
<tr>
<td>130</td>
<td></td>
<td>07-11-95</td>
<td>04:55PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>140</td>
<td></td>
<td>07-11-95</td>
<td>05:08PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>150</td>
<td></td>
<td>07-11-95</td>
<td>05:24PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>160</td>
<td></td>
<td>07-11-95</td>
<td>07:00PM</td>
<td>1</td>
<td>SF3</td>
</tr>
<tr>
<td>170</td>
<td></td>
<td>07-11-95</td>
<td>07:54PM</td>
<td>1</td>
<td>SF3</td>
</tr>
</tbody>
</table>
Err: The error number. For an explanation of the error codes less than 1000, see Table 14-2; for error codes 1000 or greater, see the description of the CALL-EVENT verb.

Call ID: The call ID that had the error.

User ID: The ID of the agent who handled the event that caused the error.

Port: The port on which the error occurred.

**Table 14-2. Queue Errors**

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Queue Definitions for the queue does not exist. EDGE tried to add/view/remove a call event from a queue for which there was no queue definition. This can happen if you delete a queue definition but forget to remove the queue ID from the operator access.</td>
</tr>
<tr>
<td>2</td>
<td>Call event locked and in use by another user. The system locks each call event while an agent is accessing it. If other users try to access the call event while it is locked, they will receive this error. If no agent has the call event active, the call event is probably in an error state. Most likely it is in the STATUSID File inappropriately, and should be cleared from this file using the file utilities.</td>
</tr>
</tbody>
</table>
The call event was not found in the queue at the recorded position. Each queued call event has a corresponding record in the status file. The record contains the name of the queue and position (index) within the queue. Time-ordered queues have an index composed of the scheduled date and time, while FIFO queues just have an integer index.

When a specific call event is requested from a queue, the system uses this information to physically locate and remove the call event from the queue. If it is unable to locate the call event in the queue at the recorded index position, this error is generated. If a problem occurs and corrupts certain queue-related files, this error will occur.

Calls exist in queue, but none are available. This error code currently has two possible meanings depending on whether the error was generated from operations or from an ACL command; can be one of the following:

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The call event was not found in the queue at the recorded position. Each queued call event has a corresponding record in the status file. The record contains the name of the queue and position (index) within the queue. Time-ordered queues have an index composed of the scheduled date and time, while FIFO queues just have an integer index. When a specific call event is requested from a queue, the system uses this information to physically locate and remove the call event from the queue. If it is unable to locate the call event in the queue at the recorded index position, this error is generated. If a problem occurs and corrupts certain queue-related files, this error will occur.</td>
</tr>
<tr>
<td>4</td>
<td>Calls exist in queue, but none are available. This error code currently has two possible meanings depending on whether the error was generated from operations or from an ACL command; can be one of the following:</td>
</tr>
</tbody>
</table>
In operations, this error code indicates that an agent attempted to get the next available call from a time-ordered queue, and there were no calls in the queue that met the date and time requirements. This can occur when selecting Next Call from the Call Center screen or when the CALL-EVENT command Get Next is executed in a logic flow. Not an error condition, this code causes normal system messages to appear to the user.

This message can also occur during a Qclear or Qrebuild command to indicate inconsistencies between the STATUS File and the queues file, however, these commands correct the problem during the same execution that creates the error, so there is no cause for concern. However, EDGE Customer Care should be contacted to attempt to determine how the queues file and STATUS file became inconsistent.

You can determine which problem the error indicates by checking the user ID and port the error was generated on.

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>In operations, this error code indicates that an agent attempted to get the next available call from a time-ordered queue, and there were no calls in the queue that met the date and time requirements. This can occur when selecting Next Call from the Call Center screen or when the CALL-EVENT command Get Next is executed in a logic flow. Not an error condition, this code causes normal system messages to appear to the user.</td>
<td></td>
</tr>
<tr>
<td>This message can also occur during a Qclear or Qrebuild command to indicate inconsistencies between the STATUS File and the queues file, however, these commands correct the problem during the same execution that creates the error, so there is no cause for concern. However, EDGE Customer Care should be contacted to attempt to determine how the queues file and STATUS file became inconsistent.</td>
<td></td>
</tr>
<tr>
<td>You can determine which problem the error indicates by checking the user ID and port the error was generated on.</td>
<td></td>
</tr>
</tbody>
</table>
### Table 14-2. Queue Errors (continued)

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Next available call event is beyond today’s date. This code is used internally to denote that using the specified removal strategy, the next call event in a time-ordered queue is later than the current date. Not an error condition unless it appears using the List Queue Problem Log command.</td>
</tr>
<tr>
<td>6</td>
<td>Next available call event is beyond the current time. Same as Error Code 5.</td>
</tr>
<tr>
<td>7</td>
<td>Transfer of a call event failed because it is not in the specified queue. The queues file entry for the call ID could not be deleted. This would typically only occur if there is an inconsistency between the STATUS File and the queues file. This can be resolved by running a Qrebuild.</td>
</tr>
<tr>
<td>8</td>
<td>Attempted to queue transfer a call event from a queue in which it did not exist. For example, the call event 1000 is in the queue NEW. If you execute a queue transfer and specify to transfer the call event 1000 from queue CALLBACK to queue OLD, an error 8 is generated because the call event 1000 is not in queue CALLBACK.</td>
</tr>
</tbody>
</table>
### Table 14-2. Queue Errors (continued)

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Status record for Next Available call event cannot be found. The system determines the Next Available call event by examining the queues, reading the <strong>STATUS File</strong> and flagging the records, as active. If, for some reason, the STATUS record does not exist for the call event, this error is generated and the next call event is requested. If the STATUS file has been corrupted or damaged (CLEAR-FILE, write by user to STATUS file using logic flow, etc.) this error code occurs. This is the reverse of error code 3.</td>
</tr>
<tr>
<td>10</td>
<td>Call event exists but is not queued. All queued call events should have a record in the <strong>STATUS File</strong> with a status code of “D” (in attribute 1). When the next available call event is determined but the status code for the call event is not a “D”, this error code is generated.</td>
</tr>
<tr>
<td>11</td>
<td>Queue ID in the <strong>STATUS File</strong> record does not match the ID of the queue from which the call event was removed. When the next call event is physically removed from a queue list, it is compared with the queue ID contained in the STATUS record. If they do not agree, an error is generated and the next call event is processed.</td>
</tr>
</tbody>
</table>
The system assumes that the information in the STATUS File record is always correct. Therefore, the queue list was in error (corrupted) and the call event is assumed to be in the queue list identified by the STATUS File record queue ID.

<table>
<thead>
<tr>
<th>Error Number</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>No reading from queue allowed (closed to agents). This code is used internally to inform the system to skip this queue when searching for the Next Available call event.</td>
</tr>
<tr>
<td>13</td>
<td>Unable to set process code on queue. It is not recommended that multiple queue utilities be performed on the same queue(s) at the same time. Even so, it is possible to instruct the system to seed, clear and rebuild and/or transfer call events in queues at the same time on two separate ports (processes). The system “flags” specific queues as being processed by queue utility on the port executing the utility so that other utility functions will not attempt to modify that queue. When the utility completes, the flags are normally removed and other processes can access these queue(s). If, for some reason, a utility hangs/breaks before it is finished, these flags may still be set. Other processes attempting to use any queues with these flags set will generate an error code.</td>
</tr>
</tbody>
</table>

Table 14-2. Queue Errors (continued)
Unable to build queue list. The system was unable to build a queue list because the SCHED.DATE and SCHED.TIME were blank. System fills in time and date with current values and continues. This error is created when the system crashes.

Trying to remove a call from a group protected queue when the agent is not assigned to the group.

Attempting to delete a call ID when the call record has been sent to a predictive dialer (in the PDW_QUEUE file).

Unable to locate queue during a queue transfer. This typically occurs when attempting to perform a queue transfer when swapping a guide.

These errors are generated by the CALL-EVENT verb.
Overview

Logic flows and tasks are constructed statements that describe actions or processes to be performed under specified conditions. Logic flows are attached to guides and performed within operations while tasks can be executed from within EDGE Developer or as background processes. While logic flows generally update one record, a task can update multiple records. Tasks allow you to specify selection lists and reports to be run before and after a task.

Logic and task editors are provided to allow you to create or modify statements for logic flows and tasks. Both editors allow you to construct statements in the same manner. The logic editor is used to construct logic flows within a guide, while the task editor specifies tasks that are performed outside of a guide.

Within the task area, there is a task scheduler that allows you to assign tasks to be run at specific times of the day, month, and year. Once set up and started, the scheduler will automatically run tasks.
according to the schedule you have specified. For more information, see the section Task Scheduler.

The following topics are described in this chapter:

- Constructing Statements.
- Logic and Task Editors.
- Logic Flows.
- Tasks.
- Task Scheduler.
Constructing Statements

When creating a logic flow or task, you construct statements describing the conditions under which to perform actions. Statements are created by using the Logic and Task Editors.

**EDGE Statements**

The following types of statements are available:

- ALWAYS.
- COMMENT.
- LABEL.
- IF-THEN-ELSE.

**ALWAYS**

The ALWAYS statement performs an action in all cases. When you begin a statement with ALWAYS, you must follow with a verb. When the verb action is completed, the statement is finished. You cannot use AND or OR to continue. (For more information on verbs, see Chapter 16.)

The syntax for statements beginning with ALWAYS is:

```
ALWAYS verb
```

The following types of statements can follow ALWAYS:
- ALWAYS.
- IF.
- LABEL.
- COMMENT.

**Examples**

The statement below copies zero to the variable @TOTAL. (For information on variables such as @TOTAL, see the section Variables.)

```
ALWAYS COPY the value 0 into @TOTAL
```

The statement below multiplies the value in #COST by the value in C-ITEMS and stores the result in @SUM.

```
ALWAYS MULTIPLY the value of #COST to the value of C-ITEMS store result in @SUM
```

The statement below assigns a new call ID using the value in field ACCT.NO.

```
ALWAYS ASSIGN-NEW-CALL-ID from the value of ACCT.NO
```

**COMMENT**

The COMMENT statement explains or describes the statements in the logic flow or task. Comments also allow you to provide a trail of
changes made to the code. You can enter a comment at any line in
the logic flow or task, including between the IF-THEN-ELSE state-
ments.

When you select the COMMENT term to begin a line, you are allowed
to enter a line of text. You do not have to enter quotation marks with
the comment. After you press OK, the comment is displayed pre-
ceded by two dashes (--), which indicates the line is a comment. You
can continue the comment for more than one line by selecting COM-
MENT repeatedly.

Using comments does not affect the performance of the logic flow or
task.

The syntax for statements beginning with COMMENT is:

    --Comment.

The following types of statements can follow COMMENT:

- ALWAYS.
- IF.
- LABEL.
- COMMENT.
Examples

The following comment describes part of the code.

- - This code extracts the numeric part of PART.NUM
- - and stores it in @NUM

The following comment indicates when and by whom the code was first created.

- - This code was created on April 15, 1997 by Joan Smith.

LABEL

The LABEL statement allows you to name a line number in the logic flow or task. Used in combination with the GOTO verb, you can bypass statements and go directly to the label. If you do not use the GOTO verb, the logic flow continues to the next statement. You can also create a loop statement by using a label.

Label statements should be completed by routing to another label or using the EXIT verb. If you do not exit the label, the logic flow continues to the next statement.

EDGE identifies the label name by enclosing it in braces ({}). These symbols are automatically generated by EDGE if using the Property Assisted or Quick Edit modes. The braces are considered part of the
label name and must be specified when using the label in a GOTO statement.

The syntax for LABEL statements is:

    LABEL {name}

The following types of statements can follow LABEL:

- ALWAYS.
- IF.
- LABEL.
- COMMENT.

Examples

The statement below determines the value of @SORT. If the value is D, the statements following the label {DESCENDING} are performed. If the value is not D, the statements following the label {ASCENDING} are performed.

    IF COMPARE if the value @SORT is = the value "D"
        THEN GOTO label {DESCENDING}
        ELSE GOTO label {ASCENDING}

    LABEL {ASCENDING}
    ALWAYS SORT-VALUES in PRICE with data values
        in TYPE order ascending type alphanumeric
    ALWAYS EXIT
LABEL {DESCENDING}
ALWAYS SORT-VALUES in PRICE with data values
   in TYPE order descending type
   alphanumeric
ALWAYS EXIT

The statement below checks the value in AREA.CODE. If the value is 714, it goes to the label {LOCAL NUMBER}. If the number is not 714, it goes to the label {LONG DISTANCE}.

   IF COMPARE if the value AREA.CODE is = the value 714
       THEN GOTO label {LOCAL NUMBER}
       ELSE GOTO label {LONG DISTANCE}

\textit{IF}

The IF statement is used to begin an IF-THEN-ELSE construction that tests a condition and performs actions based on the result of the condition. For more information on the types of conditions that may be used, see the section \textit{Condition Terms}.

The IF term can be used with the \textit{Conjunction Terms, AND or OR}, to test for more than one condition or for alternative conditions.

IF a condition is true, THEN statements are performed. IF a condition is false, ELSE statements are performed. When the IF statement is followed by AND or OR, a compound condition is tested.
Although the IF-THEN-ELSE statement can be constructed in a variety of ways, the following rules must be met:

- At least one THEN statement or one ELSE statement must follow the IF condition.
- THEN and ELSE statements must be grouped together, as shown in the examples.

The syntax for statements beginning with IF is:

```
IF condition
```

The following types of statements can follow IF:

- **AND**.
- **OR**.
- **THEN**.
- **ELSE**.
- **COMMENT**.

**Examples**

The statement below determines if the value of @TOTAL is greater than zero or the value in @RESULT is equal to INTEREST. If either condition is true, the value in TOTAL is copied to GR.TOTAL, Y is copied to @BROCHURE, and you are routed to the THANKS screen. If neither
condition is true, null is copied to @MSG and you are routed to the NO.SALE screen.

    IF COMPARE if the value @TOTAL is > the value 0
    OR COMPARE if the value @RESULT is = the value "INTEREST"
    THEN COPY the value TOTAL into GR.TOTAL
    THEN COPY the value "Y" into @BROCHURE
    THEN ROUTE to screen "THANKS"
    ELSE COPY the value "" into @MSG
    ELSE ROUTE to screen "NO.SALE"

The statement below determines if the ACCT.NO is null. If ACCT.NO is null, the message "Enter the Account ID" is copied to @MSG, and the operator is routed to the ACCT.NO field on the GET.ID screen. If ACCT.NO is not null, the operator is routed to the ORDER screen.

    IF COMPARE if the value ACCT.NO is = the value ""
    THEN COPY the value "Enter the Account ID" into @MSG
    THEN ROUTE to screen "GET.ID" field ACCT.NO
    ELSE ROUTE to screen "ORDER"

THEN

The THEN term is part of the IF-THEN-ELSE construction and specifies the actions to be taken if the conditions are met. THEN statements must be grouped together.
The syntax for statements beginning with THEN is:

```
THEN verb
```

The following terms can follow THEN:

- ALWAYS.
- IF.
- THEN.
- ELSE.
- LABEL.
- COMMENT.

**ELSE**

The ELSE term is part of the IF-THEN-ELSE construction and specifies the actions to be taken if the conditions are not met. ELSE statements must be grouped together.

The syntax for statements beginning with ELSE is:

```
ELSE verb
```

The following terms can follow ELSE:

- ALWAYS.
Conjunction Terms

Conjunction terms allow you to add additional or alternative conditions to the IF statement. When conjunction terms are used in an expression, EDGE evaluates the expression from left to right. Conjunction terms include the following:

- AND.
- OR.

**AND**

The AND term can be used to form compound conditions by combining two or more comparisons. When you use AND to create a compound condition, all the conditions must be true for the THEN statement to be performed. If one condition is false, the entire statement is considered false, and the ELSE statements are performed. For more information on using conditions, see the section Condition Terms.

The syntax for statements beginning with AND is:
AND condition

The following terms can follow AND:

- AND.
- OR.
- THEN.
- ELSE.
- COMMENT.

Examples

The statement below determines if the value for @SEX is equal to FEMALE and the value for @AGE is greater than 18. If both statements are true, you go to label {SELL}. If @SEX is not equal to FEMALE, or @AGE is equal to or less than 18, the statement is false and you go to the label {SORRY}.

```
IF COMPARE if the value @SEX = "FEMALE"
    AND COMPARE if the value @AGE > 18
    THEN GOTO label {SELL}
    ELSE GOTO label {SORRY}
```

The statement below determines if the values for @INTEREST and @NEW are Y. If both statements are true, you are routed to the NAME field on the NEW.CUST screen. If @INTEREST is not Y, or @NEW is not Y, the statement is false and you go to the label {NO.INT}.
IF COMPARE if the value @INTEREST = "Y"
   AND COMPARE if the value @NEW = "Y"
THEN ROUTE to screen "NEW.CUST" field
   NAME
ELSE GOTO label {NO.INT}

**OR**

The OR term can be used to add alternative conditions to the IF statement. When you use OR to create an alternative condition, only one of the conditions must be true for the THEN statement to be performed. If all the conditions are false, the ELSE statements are performed. For more information on using conditions, see the section Condition Terms.

The syntax for statements beginning with OR is:

    OR condition

The following terms can follow OR:

- **AND.**
- **OR.**
- **THEN.**
- **ELSE.**
- **COMMENT.**
**Examples**

The statement below determines if the value for `@SALARY` is greater than or equal to 50,000 or `@SAVINGS` is greater than or equal to 70,000. If either is true, you are routed to the INVEST screen. If neither are true, you are routed to the OTHER screen.

```
IF COMPARE if the value @SALARY is >= 50000
  OR COMPARE if the value @SAVINGS
    >= 70000
THEN ROUTE to screen "INVEST"
ELSE ROUTE to screen "OTHER"
```

The statement below determines if the value for `@INTEREST` is equal to Y or `@NEW` is equal to Y. If either is true, you are routed to the GET.INFO screen. If neither are true, you are routed to the GOODBYE screen.

```
IF COMPARE if the value @INTEREST = "Y"
  OR COMPARE if the value @NEW = "Y"
THEN ROUTE to screen "GET.INFO"
ELSE ROUTE to screen "GOODBYE"
```

**Condition Terms**

Condition terms are used to compare values or evaluate if a condition exists in a logic flow or task statement. The condition term always follows an `IF` statement or the conjunction terms `AND` or `OR`.

The following rules govern whether a comparison is true:
• The comparison is based on the internal format of the data, not on the display format.

• Uppercase and lowercase letters are treated as different letters. This is because EDGE bases the comparison on the ASCII value of the letter. For example, B is not the same as b, and EDGE is not the same as edge.

• The ASCII value is used to determine whether a character falls within a range of values. For example, z is not in the range of A to Z because z has an ASCII value of 122 and A to Z has an ASCII value of 65 to 90. For a listing of ASCII values, see Appendix Y.

• For numeric fields, null (""") and zero are treated as the same.

• For alphanumeric fields, null (""") is less than any other character.

• Numeric values are compared numerically. Combined alphanumeric values are compared alphabetically. Blank spaces within a numeric value are treated as an alphanumeric value and therefore are compared alphabetically.

Condition terms include the following:

• COMPARE.

• COMPARE-MULTI-VALUES.
Logic Flows & Tasks

Constructing Statements

- MEMBER-OF-SET.
- PATTERN.
- RANGE.

**COMPARE**

The COMPARE term compares two values to determine if a condition exists. There are six comparison terms which are represented by mathematical symbols.

The syntax for the COMPARE condition is:

```
IF COMPARE value {symbol} value
```

When the COMPARE term is selected using the Property Assisted mode, a dialog box similar to the following is displayed:

![Dialog box example](image)
**Properties:**

- **If the value**
  The first value in the comparison; can be a number, a literal, or a field or variable containing the value.

- **Is**
  The symbol representing the type of comparison; can be one of the following types:
  - Equal to (\(=\)).
  - Less than (\(<\)).
  - Greater than (\(>\)).
  - Less than or equal to (\(\leq\)).
  - Greater than or equal to (\(\geq\)).
  - Not equal to (\(#\)).

- **The value**
  The second value in the comparison; can be a number, a literal, or a field or variable containing the value.

**Examples**

In the following statement, if the value of @TOTAL is 8 and the value of ITEM.TOT is 10, the condition is true because 8 is less than 10.

IF COMPARE if the value @TOTAL is less than the value ITEM.TOT
In the following statement, if the value of SUM is 2, the condition is false because 2 is not equal to zero.

IF COMPARE if the value SUM is = the value 0

In the following statement, if the value of @LINE is 55, the condition is true because 55 is greater than 50.

IF COMPARE if the value @LINE is > the value of 50

**COMPARE-MULTI-VALUES**

The COMPARE-MULTI-VALUES term compares values in a multivalued field with a single value. There are six types of comparisons which are represented by mathematical symbols.

The syntax for the COMPARE-MULTI-VALUES condition is:

IF COMPARE-MULTI-VALUES {condition} values {symbol} value

When the COMPARE-MULTI-VALUES term is selected using the Property Assisted mode, a dialog box similar to the following is displayed:
**Properties:**

**If**

The condition for the comparison to be considered a match; can be one of the following:

- **All**: All the values must match the second value.
- **Any**: At least one value must match the second value.
- **None**: None of the values can match the second value.

**Of the values**

The values to be compared; can be a literal, a field, a multivalued field, or a variable containing the values.

**Are**

The symbol representing the type of comparison; can be one of the following types:

- **Equal to (=)**.
- **Less than (<)**.
Properties: (continued)

- Greater than (>).
- Less than or equal to (<=).
- Greater than or equal to (>=).
- Not equal to (#).

The value The value to be compared against; can be a number, a literal, or a field or variable containing the value.

Examples

In the statement below, if the values in NO.SEL are 1, 2, 3, 4, the condition is true because all of the values are greater than zero.

IF COMPARE-MULTI-VALUES if all of the values NO.SEL are > the value 0

In the statement below, if the values of NO.SOLD are 1, 4, 5, 12, the condition is true because 12 is greater than 10.

IF COMPARE-MULTI-VALUES if any of the values NO.SOLD are >= the value 10

In the statement below, if the values of COLORS are Blue, Yellow, and Red, the condition is true because none of the values are equal to Green.

IF COMPARE-MULTI-VALUES if none of the values COLORS are = the value GREEN
**MEMBER-OF-SET**

The MEMBER-OF-SET term determines if a value is included in a set of values. If the value is included in the set, the comparison is true and the actions starting with THEN are performed. If the value is not included in the set, the comparison is false and the actions starting with ELSE are performed.

The syntax for the MEMBER-OF-SET condition is:

IF MEMBER-OF-SET value {symbol}values

When the MEMBER-OF-SET term is selected using the Property Assisted mode, a dialog box similar to the following is displayed:

Properties:

- **If the value**
  - The value to be compared; can be a number, a literal, or a field or variable containing the value.
Properties: (continued)

Is a member of The members to be compared against; must be entered on the screen using the following format:

The list must begin and end with a slash (/), and all the values must be separated by a slash (/).

Examples

In the statement below, if the value of @COLOR is BLUE, the condition is true because BLUE is a member of the set.

IF MEMBER-OF-SET if the value @COLOR is a member of /BLUE/YELLOW/RED/

In the statement below, the condition is false because MR is in capital letters and considered different from Mr, which is in uppercase and lowercase letters.

IF MEMBER-OF-SET if the value "MR" is a member of /Mr/Ms/Miss/

PATTERN

The PATTERN term determines if a single value matches a text or number pattern. If the value matches the pattern, the comparison is true and the actions starting with THEN are performed. If the value does not match the pattern, the comparison is false and the actions starting with ELSE are performed.
The syntax for the PATTERN condition is:

```
IF PATTERN value {code}
```

When the PATTERN term is selected using the Property Assisted mode, a dialog box similar to the following is displayed:

![Dialog Box](image)

**Properties:**

- **If the value**
  
The value to be compared against the pattern; can be a number, a literal, or a field or variable containing the value.

- **Matches the pattern**
  
The pattern to be matched; must be identified using codes described in Table 15-1.

EDGE identifies the pattern by enclosing it in braces (`{}`). These symbols are automatically generated by the Property Assisted or Quick Edit modes.
Table 15-1. Pattern Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Pattern Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Alphabetic characters; can be shown as AA or 2A.</td>
</tr>
<tr>
<td>N</td>
<td>Numbers; can be shown as NNN or 3N.</td>
</tr>
<tr>
<td>X</td>
<td>Alphanumeric characters; can be shown as XXXX or 4X.</td>
</tr>
<tr>
<td>Literals</td>
<td>Literals; must be enclosed in quotation marks. For example, symbols such as hyphens (-) and slashes (/) must be enclosed in quotation marks.</td>
</tr>
</tbody>
</table>

Examples

In the statement below, if the value of the field TEL.NO is 714-622-6200, the condition is true because the value of the telephone number matches the pattern.

    IF PATTERN if the value TEL.NO matches the pattern {3N"-"3N"-"4N}

In the statement below, the condition is not true because IMA contains 3 alphabetic characters and the pattern is for 2 alphabetic characters.

    IF PATTERN if the value "IMA" matches the pattern {2N}
RANGE

The RANGE term determines if a value is within a numeric or letter range. If the value falls within the range, the condition is true and the actions starting with THEN are performed. If the value is outside the range, the condition is false and the actions starting with ELSE are performed. You identify the value to be examined and the beginning and ending value in the range. The values can be positive or negative.

The syntax for the RANGE condition is:

IF RANGE value value

When the RANGE term is selected using the Property Assisted mode, a dialog box similar to the following is displayed:
Examples

In the statement below, if the value of NO.SOLD is 4, the condition is true because 4 falls within the range of 2 to 7.

IF RANGE if the value NO.SOLD is in the range from 2 to 7

Properties:

If the value  The value to be examined; can be a number, a literal, or a field or variable containing the value.

In the range from  The beginning number or letter in the range; can be a number from 1 to $2^{32}$, or up to 4 letters enclosed in quotation marks.

To  The end number or letter in the range; can be a number from 1 to $2^{32}$, or up to 4 letters enclosed in quotation marks.
In the statement below, the condition is false because the value ABC does not fall within the range of AZZ to MMM since ABC comes before AZZ.

   IF RANGE if the value "ABC" is in the range from "AZZ" to "MMM"

In the statement below, if the value of NUMBER is 100, the condition is false because 100 is greater than 99.

   IF RANGE if the value NUMBER is in the range from 1 to 99
Logic and Task Editors

The logic and task editors allow you to create or modify statements for logic flows and tasks. Both editors allow you to construct statements in the same manner. The logic editor is used to construct logic flows within a guide, while the task editor specifies tasks to be performed outside of a guide. For more information, see the sections Logic Flows, Tasks, and Task Scheduler.

As statement lines are entered, the editor displays the statement line in a color that identifies the status of the line. The colors indicate the following:

- **Black**: The line is being edited.
- **Blue**: The line is valid.
- **Green and italicized**: The line is a comment line.
- **Red and bold**: The line has a syntax error.

**Note:** The first time you add or modify a logic flow or task after installing or upgrading EDGE, the system must install the list of available verbs from the server. This process takes a few minutes and is performed only once.
**Edit Modes**

You can use three editing modes to create a statement. The editing modes provide a wide range of assistance from displaying all the selections and properties that can be used to construct a statement to allowing you to enter the statement directly in a syntax form. You can use any or all of these modes in a logic flow or task. The following edit modes are available:

- **Property Assisted.**
- **Freeform Assisted.**
- **Quick Edit.**

**Property Assisted**

The property assisted editor mode provides a verb properties dialog box that helps you construct a statement line for a logic flow or task. To display the verb properties dialog box, move the cursor to the editor area, then do one of the following:

- Double-click.
- Press F4.
- Select the properties tool on the toolbar.

A dialog box similar to the following is displayed:
To assist in constructing logic flow or task statements, the verb properties dialog box provides the following:

- **Edit Tools.**
- **Upper List Box.**
- **Lower List Box.**

**Edit Tools**

The following edit tools are provided at the top of the verb properties dialog box and from the Logics menu to add, delete, or move from line to line within logic flow or task statements:

- **Next line** - Applies changes, if any, to the current line and selects the next line. If there is no next line, no action is taken.

- **Previous line** - Applies changes, if any, to the current line and selects the previous line. If there is no previous line, no action is taken.
### Upper List Box

The upper list box on the verb property dialog box lists the available statements that can be used to start a logic flow or task. For more information, see [Constructing Statements](#).

### Lower List Box

The lower list box on the verb property dialog box provides acceptable condition terms or verbs that can be used based upon the selection used in the upper list box.

After selecting a condition term or verb, the valid properties associated with the selected term or verb are displayed. For more information, see [Conjunction Terms](#), [Condition Terms](#), and [Chapter 16, Verbs](#).

<table>
<thead>
<tr>
<th>Action</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Insert below</td>
<td>Applies changes, if any, to the current line and inserts and selects a blank line below the current line.</td>
</tr>
<tr>
<td>Insert above</td>
<td>Applies changes, if any, to the current line and inserts and selects a blank line above the current line.</td>
</tr>
<tr>
<td>Delete line</td>
<td>Deletes the currently selected line.</td>
</tr>
</tbody>
</table>
**Freeform Assisted**

The freeform assisted editor mode provides a prompt for the verb syntax. After entering a statement name followed by a space, the editor prompts you for the next required entry. Prompts are only displayed for required properties. For optional properties, you must type the first word of the property. An example of a freeform assisted line is displayed in Figure 15-1.

**Using Freeform Assisted Mode**

To use the freeform assisted mode, do the following:

1. Type the statement name (for example, ALWAYS or LABEL) followed by a space. The statement name can be entered in uppercase or lowercase.

2. Type the next appropriate term followed by a space. For example, for the ALWAYS statement, type the verb name or if the LABEL statement is used, type the label name.

   A prompt is displayed for the next property.
3. Type the value for the first property followed by a space. The
editor will prompt you for the next required entry. For optional
properties, you must type the first word of the property.
Continue until all the properties are specified.

4. When all the properties have been specified, press RETURN.
5. Continue creating statements until your logic flow or task is complete.

6. Select the close button to save the logic flow or task.

Quick Edit

Quick edit mode allows you to enter the statement terms and property values separated by semi-colons without using prompts. An example of this mode is displayed Figure 15-2.

Using Quick Edit Mode

To use this mode, do the following:

1. Check the Quick Edit box on the Logic or Task Editor.

2. Type the statement name (for example, ALWAYS or LABEL) followed by a space.

3. Type the next appropriate term followed by a semi-colon (;). For example, for the ALWAYS statement, type the verb name or if the LABEL statement is used, type the label name.

4. Type each property value separated by a semi-colon (;). The properties must be entered in the order displayed by the Property Assisted verb properties dialog box. If a property is not specified, it still requires a semi-colon as a placeholder.

5. When all the properties have been specified, press RETURN. The line is returned with the proper verb syntax filled in.
6. Continue creating statements until your logic flow or task is complete.

7. Select the close button to save the logic flow or task.

Figure 15-2. Quick Edit Mode
Logic Flows

Logic flows are constructed statements that are performed within a Guide.

This section includes the following:

- Logic Flow Development Steps.
- Logics Tab.
- Logics Menu Commands.
- Logic Flow Search.

Logic Flow Development Steps

The following steps are involved when developing a logic flow:

- Create your logic flow. For information, see the section Constructing Statements. For information on the types of edit modes available, see the section Logic and Task Editors.
- Specify the Default file (optional).
- Verify the logic flow and correct any errors.
- Attach the logic flow to the guide. Logic flows can be attached to screens using Function Keys, Logic Properties, the Logic property of objects, Hot Keys, and End-of-guide logic.
- Test the guide as described in the section Getting the Guide Ready.

**Logics Tab**

Logic flows are created at the project level. To display the logic page, select the following:

Project / Guide area / Logics tab

When the Logics tab is selected, the existing logic flows are displayed. The Logics tab is similar to the following:
When you add or modify a logic flow, a page similar to the following is displayed:

![Logic Editor](image)

**Properties:**

- **Default file**
  The default file for the logic flow. If a default file is specified, you do not have to use the filename or file abbreviation when using fields in the logic flow.

  The default is the Guide assembly default file.
Parameter Definition

Clicking the Logic Parameters button in the Logic flow editor brings up the Parameter Definition for the logic flow (see figure below). If you define parameters for a logic flow, it can only be called by the

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Last edited</strong></td>
</tr>
<tr>
<td><strong>Line</strong></td>
</tr>
<tr>
<td><strong>Logic name</strong></td>
</tr>
<tr>
<td><strong>Logic Parameters</strong></td>
</tr>
<tr>
<td><strong>Quick edit</strong></td>
</tr>
</tbody>
</table>
**CALL-LOGIC** verb (not the CALL verb, which is meant to call logic flows without parameters). Logic flows with parameter definitions cannot be called from anything other than a logic flow (for example, they cannot be attached to function keys).

**Properties:**

- Logic flow: The name of the logic flow
Properties: (continued)

| Parameter | The text to be displayed for this parameter in the CALL-LOGIC verb properties. If not specified, the parameter is displayed as Parameter n, where n is the position. |
|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------
| Name      | Indicates how this parameter is to be used; can be one of the following:                                                                                                                   |
| Usage     | Input only: this parameter can be used to pass values into the logic flow.                                                                                                                   |
|           | Input and output: this parameter can be used to pass values into and back from the logic flow to the calling logic flow.                                                                    |
|           | Output only: this parameter can only be used to pass values back to the calling logic flow.                                                                                                  |
| Position  | The order of this parameter in the logic flow; this property is display only                                                                                                                  |
| Return    | A checkbox which specifies whether the logic flow returns a value.                                                                                                                          |

Logics Menu Commands

The following commands are available from the Logics menu:

- Delete line. Editor only.
- Insert below. Editor only.
List Logics

The List Logics command displays a listing of logic flows for the project. You can print the list to a client printer or view it on screen. When selected, a report similar to the following is displayed:
### Report Fields:

- **Changed since by**: The user ID of the person who saved the last changes to the logic flow.
- **Created on**: The date the logic flow was created.
- **ID**: The name of the logic flow.
- **Last assem on**: The date the logic flow was last assembled.
- **Last assem by**: The user ID of the person who last assembled the logic flow.

<table>
<thead>
<tr>
<th>ID</th>
<th>Created on</th>
<th>Last assem on</th>
<th>Changed by</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD</td>
<td>07/05/96</td>
<td></td>
<td>GST7</td>
</tr>
<tr>
<td>CALC</td>
<td>07/09/96</td>
<td></td>
<td>GST7</td>
</tr>
<tr>
<td>CHK.DATE</td>
<td>07/09/96</td>
<td></td>
<td>GST7</td>
</tr>
<tr>
<td>GST.RBC</td>
<td>07/09/96</td>
<td></td>
<td>GST7</td>
</tr>
</tbody>
</table>
The list logic code command allows you to print logic flow code. The listing is sent to the server printer. You can specify to print all the logic flows, logic flows within a selection list, or individual logic flows. When selected a dialog box similar to the following is displayed:

**List Logics Code**

Verify

The verify logic command allows you to determine if the logic flow is correct. When you verify the logic flow, the names of the fields,
files, and other objects used in the logic are verified. If a file or field does not exist, you will receive a message.

Verify displays only the first error it encounters. After fixing an error, you should repeat the command until there are no more errors.

The Verify command can be initiated in the following ways:

- Select the Verify icon on the ETW–W Toolbar while in the editor or when a logic flow is selected on the Logics tab.
- Select the Verify command button on the Logics tab while a logic flow is selected.
- Select the Verify Logic command from the Logics menu while in the editor or when a logic flow is selected on the Logics tab.

If you are in the editor and the verify command finds an error, it displays a message similar to the following:

![ETW for Windows dialog box](image)

- Field ID: ID ... not found in guide file: TEMP
- There is an error in line 1 of "GET.REC" of project "SOFTSELL".

OK
If you are not in the editor and the verify command finds an error, it displays a message similar to the following:

To open the editor and make corrections, select Yes. To quit the verify without making corrections, select No.

If the logic flow is correct, the following message is displayed:

**Logic Flow Search**

Specific text strings can be found while in the logic editor using the following:

- The **Find** command from the Edit Menu.
- **CTRL+F**.
Tasks

Tasks are constructed statements which can be executed immediately or as background processes. To automate background processes, see the section Task Scheduler.

This section describes the following:

- Task Development Steps.
- Tasks Area.
- File Access Tab.
- MultiRecord Tab.
- Task Commands.
- Task Search.

Task Development Steps

Defining a task involves the following steps:

- Create the task. For more information, see the section Constructing Statements. For information on the types of edit modes available, see the section Logic and Task Editors.
- Specify the File Access.
- Specify the records to be processed using the MultiRecord tab (optional).
• Specify the selection lists and reports to be run before and after the task using the MultiRecord tab (optional).

• Identify the next task to be run using the MultiRecord tab (optional).

• Assemble the task and correct any errors.

• Run the task or schedule the task for a specified time. To automate the task, see the section Task Scheduler.

**Tasks Area**

Tasks can be created at the EDGE or project level. You must define a task at the level you intend to use it. To display the task definition page, select the following:

EDGE or a project / Task area

When the task area is selected, a list of existing tasks is displayed on the Task Definition page. At the EDGE level, the Task area contains an additional tabbed page: Task Scheduler.

The Task Definition tab is similar to the following:
When you add or modify a logic flow, a page similar to the following is displayed:
**Properties:**

- **Chain to**: The name of the next task to be run after this task is completed. This name is specified on the **MultiRecord** tab. This property is display only.

- **Default file**: The default file for the task. The filename is specified on the **File Access** tab. This property is display only.
## File Access

The File Access tab is used to specify the files that can be accessed by the task. The default file for the task is the first file listed.

When you select the File Access tab, a page similar to the following is displayed:

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Last edited</td>
<td>The user ID, date and time when the task is saved. This property is display only.</td>
</tr>
<tr>
<td>Line</td>
<td>The number of the selected line followed by the total number of lines in the task. This property is display only.</td>
</tr>
<tr>
<td>Task name</td>
<td>The name of the task. After the task has been saved, this property is display only. The task name must be specified before the task editor can be used.</td>
</tr>
<tr>
<td>Quick edit</td>
<td>When checked, you can enter the statement terms and property values separated by semi-colons without using prompts. For more information, see the section Using Quick Edit Mode.</td>
</tr>
</tbody>
</table>
Properties:

Abbreviation: The abbreviation used to identify the file in this task; can be any alphanumeric sequence up to four digits.

For example, the abbreviation for CUST.LIST could be C, CU, C1, CUST, or any other combination. If the abbreviation was C, you can specify a field such as NAME in a task as either CUST.LIST-NAME or C-NAME.
Properties: (continued)

To access more than one record from the same file, identify the file with more than one abbreviation. For example, CUST.LIST could be abbreviated as C1 and C2. Each abbreviation must be specified as a separate entry.

This property is optional.

Description

The description of the task. A brief description is a useful reference, especially when more than one guide developer is working on the same project.

This property is optional.

File

The files that can be accessed by the task. There must be at least one file specified, up to a maximum of 30 files.

The default file for the task is the first file in the list. When a field from the default file is used in the task, the field does not have to be prefixed with the file abbreviation or name when used in the task.

Fields that are not in the default file must be prefixed with the file abbreviation or filename when used in the task.
The MultiRecord tab is used to identify the records to be processed, reports and selection lists to be run before or after the task, and the next task to run.

When you select the MultiRecord tab, a page similar to the following is displayed:

**Properties: (continued)**

For example, if INV.LIST is not the first file listed (that is, it is not the default), any field in the INV.LIST file, such as KEY, must be referenced as INV.LIST-KEY. If INV.LIST has an abbreviation of IN, the field could also be referenced as IN-KEY.

Title

A short title for the task. The title is displayed when using the command List Tasks, which identifies all the tasks in the project.

This property is optional.
**Properties:**

**After processing task**
- The ad hoc reports and selection lists to be processed after the task is run. To print the ad hoc reports, check the Send to printer check box. See also Send to printer property.
- This property is optional.

**Before processing task**
- The ad hoc reports and selection lists to be processed before the task is run.
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chain to next task</strong></td>
<td>If the task is based on a selection list, you may want to ensure the list is current by processing it before processing the task. The next task to be run. The specified task is run after the current task is completed. If the current task is terminated by any means, the chain to next task is run. You can also execute a task from another task with the EXECUTE-TASK verb or with the EXECTASK command. This property is optional.</td>
</tr>
<tr>
<td><strong>Process task using</strong></td>
<td>The record used in the task; can be one of the following:</td>
</tr>
<tr>
<td>No record IDs</td>
<td>Runs the task without reading specific records.</td>
</tr>
<tr>
<td>All IDs in the file</td>
<td>Runs the task for each ID in the file.</td>
</tr>
<tr>
<td>Selection list</td>
<td>Runs the task for each ID in the selection list. The selection list can be run before the task is processed. When selected, a box is displayed that allows you to enter the selection list. If the selection list does not exist, a runtime error is generated.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

Individual IDs  Runs the task for each ID specified. When selected, a box is displayed that allows you to enter the IDs.

If the `EXECUTE` verb is used in the task, do not process the task with All IDs or a selection list unless you want to invoke the `EXECUTE` command for each record in the file or list.

Send to printer  Specifies to print the report.

If the Task Scheduler is to be used to run this task, you should check the Send to printer check box for both ad hoc reports and selection lists. If you do not, the request fails and the next line of the task is executed.

**Task Commands**

The following commands are available from the Task Definitions page as well as from the Task Definition menu:

- **Assemble**.
- **Assemble All Tasks**.
- **Delete line**. Editor only.
- **Insert above**. Editor only.
- **Insert below**. Editor only.
- **List Task Assembly Errors**.
• **List Task Code.**

• **List Tasks.**

• **Make Data Server.** This command is used with telephony. For more information, see your Telephony documentation.

• **Make Host Based Router.** This command is used with telephony. For more information, see your Telephony documentation.

• **Next line.** Editor only.

• **Previous line.** Editor only.

• **Run Task.**

• **Verify Task.**

**Assemble**

The Assemble command is used to assemble the selected task into an executable program that can be run. A task cannot be run until it has been assembled. The assembly process verifies that file and field names used in the task exist in the project. After the process is finished the number of errors is displayed. If there is an error, you can display it from the **List Task Assembly Errors** command.

Before a task can be assembled, you must specify the **File Access** properties.
Tasks should be reassembled under any of the following conditions:

- After modifying the code in the task.
- If the task was copied from another task.
- If the file access is modified.

Tasks do not have to be reassembled after making changes on the multi-record tab.

**Assemble All Tasks**

The Assemble All Tasks command is used to assemble all the tasks in the project. As each task is assembled its name and the number of errors, if any, are displayed.

To display task errors, select the **List Task Assembly Errors** command.

**List Task Assembly Errors**

The List Task Assembly Errors command is used to display or print the first assembly error found when the task was assembled. The listing shows the task name, location of the error, and error description. After fixing the error, you should assemble the task again to check for further errors.
**List Task Code**

The List Task Code command allows you to print task code. The listing is sent to the server printer. You can specify to print all the tasks, tasks in a selection list, or individual tasks. When this command is selected, a dialog box similar to the following is displayed:

![Print Task Code](image)

**List Tasks**

The List Tasks command allows you to list or print the tasks for the project. The listing includes the name of the task, title, date and time the task was created, and the user ID of the person who created the task.
The listing is sent to the client printer. When displayed, the list is similar to the following:

![Report Preview](image)

**Run Task**

The Run Task command runs the last assembled version of the selected task. You cannot run a task while editing the task definition. While the task is running, a screen similar to Figure 15-3 displays
information about the task. When the task is completed, you can exit the screen by pressing ENTER.

To run a task, select the task, then select the Run command button or select Run Task from the Task Definitions menu.

You can also run a task from the task scheduler. Using the task scheduler allows you to specify a date and time at which to run the task. For more information, see the section Task Scheduler.
Figure 15-3. Run Task Command

Properties:

- **End Time**: The time the task ended.
- **Error Mesg**: The outcome of the process. If the task is successful, the message “Process Complete” is displayed. If not successful, an error message is displayed.
- **Primary File**: The name of the default file.
- **Record ID**: The record ID being processed. At the end of the task, the record ID of the last record processed is displayed.
- **Record count**: The number of records processed in the task.
Properties:  (continued)

Start Time    The time the task started.
Task ID    The name of the task.

Verify Task

The Verify Task command checks the names of the files, fields, screens, reports, results, queues and other objects used in the task. If a component does not exist, you will receive a message.

Verify displays only the first error it encounters. After fixing an error, you should repeat the command until there are no more errors.

The Verify command can be initiated in the following ways:

- Select the Verify icon on the ETW–W Toolbar while in the editor or when a task is selected on the Task Definition tab.
- Select the Verify command button on the Task Definition tab while a task is selected.
- Select the Verify Task command from the Task Definition menu while in the editor or when a task is selected on the Task Definition tab sheet.

Task Search

Specific text strings can be found within a task using the following:
- The Find command from the Edit Menu.
- Ctrl+F while in the task editor.

For more information on using the find command, see the section Edit Menu.
**Task Scheduler**

The Task Scheduler can be used to run tasks based on a schedule. You can schedule a task to be performed on a specific day, at a specific time. You can also schedule tasks to be performed on a periodic basis. Any task, whether at the project or EDGE level, can be run from the scheduler.

Once set up and started, the scheduler automatically runs tasks according to the schedule you have specified.

The Task Scheduler is available for setup only at the EDGE level. To display the tasks on the scheduler, select the following:

EDGE / Task area / Task Scheduler tab

A page similar to the following is displayed:
Schedule a Task

The task scheduler allows you to define tasks to be run from the scheduler. When a task is defined and the scheduler has been started, the task is run based on the specified schedule. The schedule list displays the currently scheduled tasks.

Note: Only the Account/Project and Task properties are required for input. If all other properties are left blank, the default is for the task to run on the hour (for example, at 10:00), of every hour, of every day, of every month.
The following restrictions are for tasks being run from the task scheduler:

- Do not include verbs that display to the screen such as DEBUG and DISPLAY-VALUES. If the task scheduler encounters a request to display to the screen, the request fails and the next line of the task is executed.

- If using MultiRecord, send all ad hoc reports and selection lists to the printer (do not display them).

**Note:** If you have a task that you do not want to delete from the Scheduler, but that you want to temporarily suspend, schedule it for a specific date in the past. For example, to suspend a weekly task, modify the schedule, so that it is set to run only on yesterdays date, such as August 23; in this case, the Days of the week property should be cleared. You will have a year to either reschedule the task or delete it before it runs again.

The task scheduler contains the following tabbed pages:

- **Date.**
- **Time.**

The Account/Project and Task properties can be specified while displaying either the Date or Time tabbed page. These properties are described under the Date tab.
Date

The Date tab allows you to specify the dates to run the task. To display the date page, select the following:

EDGE / Task area / Task Scheduler tab / Date tab

When you select the Date tab, a page similar to the following is displayed:
Properties:

**Account/Project**

The account or project that contains the task; can be one of the following:

- **EDGE** For EDGE tasks, select EDGE.
- **Project name** For project tasks, select the project name.
**Properties: (continued)**

**Date offset**

Offsets the system date to a previous or future date in ad hoc reports and correspondence run from the task scheduler. For example, you may want to run a report on Monday morning that uses Sunday as a system date. In this case, the date offset would be -1.

The Date offset property can be one of the following:

- **0 or blank**
  - No offset.

- **n**
  - An integer offsets the date to the next n date. For example, an offset of 1 for a task that runs on Sunday sets the system date to Monday.

- **- n**
  - A negative integer offsets the date to the previous n date. For example, an offset of -1 on a task that runs on Monday sets the system date to Sunday.

The date offset is applied to ad hoc reports and correspondence as follows:

- Tasks in which the offset is set.
- Reports and selection lists specified in Before processing task and After processing task properties on the MultiRecord tab.
Properties: (continued)

- Tasks specified in the Chain to next task property on the MultiRecord tab.

The following ad hoc fields are affected by the date offset:

\DATE command.

Date Icons.

The following fields in correspondence are affected by the date offset:

[$CDATE]
[$CMONTH]
[$DATE]
[$DAY]
[$DOW]
[$MONTH]
[$YEAR]

The EQL reserved words D and T are not affected by the date offset value.

Other than modifying the dates in the specified ad hoc and correspondence fields, the date offset has no effect on the task.

After the task and any chained task is completed, the ad hoc and correspondence fields are reset to the previous value.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Days of the month</th>
<th>The days of the month the task will be run.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific dates</td>
<td>Select one or more dates for the task to run. To clear a date, select it again.</td>
</tr>
<tr>
<td>All</td>
<td>Select the All button to select all dates. To clear a date, select that date.</td>
</tr>
<tr>
<td>None</td>
<td>Select None to clear all set days and have the task run using the Days of the week property. If the Days of the week property is not set, the task will run every day.</td>
</tr>
</tbody>
</table>

This property can be used in combination with the days of the week property. For example, selecting 1 in this property and FRI in the Days of the week property schedules the task for the 1st of the month as well as every Friday.

<table>
<thead>
<tr>
<th>Days of the week</th>
<th>The days of the week the task will be run.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specific days</td>
<td>Select one or more days for the task to run. To clear a day, select it again.</td>
</tr>
<tr>
<td>All</td>
<td>Select the All button to select all days. To clear a day, select that day.</td>
</tr>
</tbody>
</table>

If you select an invalid day, for example, 31 for the month of February, the system will ignore it.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Select None to clear all set days and have the task run using the <strong>Days of the month</strong> property. If the Days of the month property is not set, the task will run every day.</td>
</tr>
<tr>
<td></td>
<td>This property can be used in combination with the <strong>Days of the month</strong> property. For example, selecting 1 in this property and FRI in the <strong>Days of the week</strong> property schedules the task for the 1st of the month as well as every Friday.</td>
</tr>
<tr>
<td>Last modified</td>
<td>The date, time, and user ID of the person who last modified the schedule.</td>
</tr>
<tr>
<td></td>
<td>This property is display only.</td>
</tr>
<tr>
<td>Months</td>
<td>The months the task will be run.</td>
</tr>
<tr>
<td>Specific months</td>
<td>Select one or more months for the task to run. To clear a month, select it again.</td>
</tr>
<tr>
<td>All</td>
<td>Select the All button to select all months. To clear a month, select that month.</td>
</tr>
<tr>
<td>None</td>
<td>Select None to clear all set months and have the task run using the default, which is every month; this is equivalent to selecting All.</td>
</tr>
<tr>
<td>Schedule name</td>
<td>The name of the schedule.</td>
</tr>
</tbody>
</table>
The Time tab allows you to specify the times to run the task. To display the time page, select the following:

EDGE / Task area / Task Scheduler tab / Time tab

When you select the Time tab, a page similar to the following is displayed:
Properties:

Hours  The hours property specifies the hour the task will be run, represented by a 24 hour clock setting. To specify an hour, select the number for the hour.

Specific hours  Select one or more hours for the task to run. To clear a hour, select it again.

All  Select the All button to select all hours. To clear an hours, select that hour.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Select None to clear all set hours and have the task run using the default, which is every hour; this is equivalent to selecting All.</td>
</tr>
</tbody>
</table>

The following are examples of 12-hour and 24-hour equivalents, respectively:

<table>
<thead>
<tr>
<th>Time</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Midnight</td>
<td>0</td>
</tr>
<tr>
<td>6 AM</td>
<td>6</td>
</tr>
<tr>
<td>Noon</td>
<td>12</td>
</tr>
<tr>
<td>6 PM</td>
<td>18</td>
</tr>
<tr>
<td>11 PM</td>
<td>23</td>
</tr>
</tbody>
</table>

**Minutes**

The Minutes property specifies the minute at which the task will be run, represented by 60 positions.

- **Specific minutes**: Select one or more minutes of the hour for the task to run. To clear a setting, select it again.
- **All**: Select the All button to select every minute. To clear a minute, select that minute.
- **None**: Select None to clear all set minutes and have the task run using the default, which is on the hour (0).
Deleting a Schedule

When you delete a schedule from the task scheduler it no longer has any effect. To delete a schedule select the name of the schedule on the Task Schedule tab sheet, then select the Delete command.

Note: Tasks specified as part of the schedule are not deleted.

Task Scheduler Commands

The following commands are available from the Task Scheduler menu or as command buttons on the Task Scheduler tab:

- Assign Printer.
- Scheduler Status.
- Start Task Scheduler.
- Stop Task Scheduler.

Assign Printer

The Assign Printer command allows you to assign the printer for all the tasks run through the task scheduler. The default printer is based on the user ID of the person modifying or starting the scheduler. It can be changed to any printer on the system. When selected, a dialog box similar to the following is displayed:
Scheduler Status

The scheduler status command displays information about the task process, including the current status of the scheduler, when it was started and by whom.

When the scheduler status command is selected and the task scheduler is running, a message similar to Figure 15-4 is displayed.

```
Please Wait... Task scheduler is running.
Started at 02:23:15PM Jul 11 96 by GST7
Process running on Port ID -87024 as User ID TASKCRON
Scheduler was last "awake" at 02:23:15PM Jul 11 96
Press ENTER to continue:
```

Figure 15-4. Task Scheduler is Running

When the scheduler status command is selected and the task scheduler is not running, a message similar to Figure 15-5 is displayed.
Figure 15-5. **Task Scheduler is Not Running**

**Start Task Scheduler**

The Start Task Scheduler command starts the task scheduler. After the task scheduler is started, the system continually checks the schedule for a task to be run. When the task scheduler is started, the system creates a background process, TASKCRON, that monitors the scheduler. For each task, **Task Scheduler Log** files are created in the $EDGE/bckgrnd directory. The files are appended with a PID number based on the time (in internal format) when the task was finished, for example, stderr.27814.

Once started, you can monitor the status of the scheduler through the **Scheduler Status** command. For information on having the task scheduler start automatically when your system is booted, contact **EDGE Customer Care**.

**Stop Task Scheduler**

The Stop Task Scheduler command stops the task scheduler. When you select this command, the system immediately begins to stop the
task scheduler. Once stopped, all task processing stops. When the task scheduler is stopped, a message similar to Figure 15-6 is displayed.

![Shutting down the task scheduler]

Figure 15-6. *Stop Task Scheduler Message*

**If the System Shuts Down**

If the host system unexpectedly shuts down, the task scheduler process, TASKCRON, shuts down, but EDGE may not recognize that the process has stopped. To determine if the TASKCRON process is running, enter the following command from ACL.
Task Scheduler

Task Scheduler Log

All UNIX processes are displayed. If TASKCRON is displayed, the task scheduler process is running. If it is not displayed, stop the task scheduler and then start the scheduler.

The Task Scheduler Log lists the tasks that were run. The log allows you to verify that the tasks were actually performed at the specified time. We recommend that you list the directory in order to monitor the progress of the task scheduler. The directory is located at the system level or the */edge/EDGE directory and is called bckgrnd.

To list the log, enter the following command at the EDGE level from ACL.

```
!ps -fuedge | pg
```

If the task scheduler is running, the following files will be listed:

- `stderr.nnnn`
- `stdout.nnnn`

You will see these files for each task performed from the task scheduler.

`stderr.nnnn` Contains errors reported by the task. Some items displayed in this file appear as errors, but actually are normal, such as the following:
could not open file(1) SFILES. Error code 12 8-6.

If you get a signal error, notify EDGE Customer Care.

stdout.nnnn When paging the stdout file you should see a simulation of the task as if it were running manually. The task is not actually running, it is only simulating a run of the task. Any errors that the task had at the time it ran are displayed. This process may take a significant amount of time.

**Clear the Task Scheduler Log**

Usually, the directory that contains the log is cleared when you stop the task scheduler. If the directory is too large, the logs may not be cleared when shutting down the task scheduler. It is very important to watch the bckgrnd directory and verify that the items are getting cleared. If the directory is not monitored and cleared, it can take up a tremendous amount of disk space.
Verbs allow you to perform a variety of actions, such as:

- Call ID/Event and Record ID Processing.
- Data Formatting and Conversion.
- Field and Variable Processing.
- Financial and Mathematical Calculations.
- Guide Processing.
- Record Processing.
- Special Processing.
- Telephony and Communication Functions.
- Relational Database Processing.

Most verbs can be used in both logic flows performed in operations, and tasks executed as background processes. Each verb description
identifies whether the verb can be used in logic flows, tasks, or both. Logic and task editors are provided to allow you to construct statements for logic flows or tasks. For information on logic flows and tasks, see Chapter 15, Logic Flows & Tasks.

Verbs are composed of various properties. Special syntax rules for verb properties must be followed to create valid logic flows or tasks. For more information, see the section Verb Property Syntax Rules.

This chapter lists each verb and its properties. For a quick reference guide of EDGE verbs, see the charts Verb Summary and Verbs by Function.
Verb Property Syntax Rules

Verbs are composed of various properties. Properties can be specified by entering values or selecting from list boxes. Generally, the entry is the literal value or the location of the value. For example, when you use the ADD verb you can enter the number itself or the field or variable containing the number. Entry for each property is required unless it is identified as optional.

The following verb property syntax rules must be adhered to when entering property values to create valid logic flows or tasks:

- **Files and fields**
  - The names of files and fields are entered without quotation marks unless they are to be taken as literals.

- **Multivalued field or variable**
  - Multivalued fields are entered without quotation marks. In cases where just one value is to be used, the **Value position** can be specified by separating the field name and value position by a comma.
  - If a multivalued field or variable is stored to a single value field or variable, you will receive a scan or assembly error.

- **Value position**
  - The value position of a multivalued field or variable can be entered as a number, or a field or variable containing the number.
<table>
<thead>
<tr>
<th>Verbs</th>
<th>Syntax Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In logic flows, you can use a local, call, or global variable that contains</td>
</tr>
<tr>
<td></td>
<td>the number of the value position. In tasks, you can use a local variable only.</td>
</tr>
<tr>
<td></td>
<td>The position number must be separated from the multivalued field or variable</td>
</tr>
<tr>
<td></td>
<td>name by a comma: COMMENTS,1 or COMMENTS,@NUM. Generally, a position number can</td>
</tr>
<tr>
<td></td>
<td>be used wherever a multivalued field or variable is specified.</td>
</tr>
<tr>
<td><strong>Literal</strong></td>
<td>A literal value can contain any alphanumeric characters and must be enclosed in</td>
</tr>
<tr>
<td></td>
<td>quotation marks; for example, &quot;Hello.&quot;</td>
</tr>
<tr>
<td></td>
<td>Window and screen names used in a logic flow or task must be enclosed in</td>
</tr>
<tr>
<td></td>
<td>quotation marks.</td>
</tr>
<tr>
<td><strong>Numbers</strong></td>
<td>Numbers can be entered with decimal places, but cannot contain any other</td>
</tr>
<tr>
<td></td>
<td>symbols. Numbers are not enclosed in quotation marks.</td>
</tr>
<tr>
<td><strong>Variables</strong></td>
<td>Variables must start with an @, #, or % symbol. The length of a variable name</td>
</tr>
<tr>
<td></td>
<td>is unlimited. (In debug mode, only ten characters of the name are displayed.</td>
</tr>
<tr>
<td></td>
<td>For more information on debug mode, see the description of the DEBUG verb.)</td>
</tr>
<tr>
<td></td>
<td>Note that variables in logics are verified at runtime.</td>
</tr>
<tr>
<td></td>
<td>Values stored in variables are stored on a per user basis and cannot be shared</td>
</tr>
<tr>
<td></td>
<td>among users.</td>
</tr>
<tr>
<td></td>
<td>The following types of variables can be used:</td>
</tr>
<tr>
<td>Verb Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>@variables</td>
<td>Local variable. The value for the variable is retained only while in the specific logic flow or task. At the end of the logic flow or task, the variable is initialized to null. Local variables can be used in logic flows and tasks. There is no maximum number of local variables.</td>
</tr>
<tr>
<td>#variables</td>
<td>Call variable. The value for the variable is retained only while in the call, from $OP through $END. Call variables are initialized to null before $OP is opened. Call variables can be used in logic flows, but not in tasks. The maximum is 1024 call variables per user per guide.</td>
</tr>
<tr>
<td>%variable</td>
<td>Global variable. The value for the variable is retained for each user throughout all projects on the system while the user is logged on. Global variables are only available in operations mode. When the user logs off EDGE or an XGUIDE process changes, global variables are initialized to null.</td>
</tr>
</tbody>
</table>
Global variables can be used in logic flows, but not in tasks. They cannot be used with separate XGUIDES.

The maximum is 1024 global variables per user per system.
Table 16-1 provides a brief description of each verb and indicates if the verb is available in logic, tasks, or both. Verbs listed as optional are described in documentation for the optional module.

Table 16-1. Verb List

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>3270-MONITOR</td>
<td>Monitors a 3270 session.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>3270-PLAY</td>
<td>Plays a 3270 script or starts a passthru session. An optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ADD</td>
<td>Sums two numbers.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AND-LISTS</td>
<td>Compares two lists and produces a list whose values are in both lists.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>APPEND</td>
<td>Adds a word or value to the end of a field, number, or word.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>APPEND-VALUE</td>
<td>Adds a value to a multivalued field or variable.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ASK</td>
<td>Displays a message and waits for a response.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ASSIGN-NEW-CALL-ID</td>
<td>Specifies how the call ID will be assigned or created.</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALCULATE</td>
<td>Performs complex mathematical, financial, and statistical expressions.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>CALL</td>
<td>Calls a logic flow from another logic flow.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>CALL-LOGIC</td>
<td>Like CALL, but used to call logic flows that take parameters, and return values.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>CALL-EVENT</td>
<td>Manages functions associated with call events and call IDs.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>CALL-EXTERNAL-FUNCTION</td>
<td>Calls a function that was written outside of EDGE.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>CASES</td>
<td>Invokes the case-based reasoning library. An optional verb.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>CHANGE-CALL-ID</td>
<td>Modifies an existing call ID.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>CHARACTER-CONVERSION</td>
<td>Formats alphanumeric data.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>CLEAR-RECORD</td>
<td>Removes data from the workspace of the current record.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>COMBINE</td>
<td>Merges values, numbers, or text into a multivalued field.</td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>
Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMUNICATE-</td>
<td>Extracts data from the server and copies it to user-defined fields.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>(RECEIVE)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE-KEY</td>
<td>Extracts data from a file or string. An optional verb.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>COMPOUND-INTEREST</td>
<td>Derives a compound interest amount.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>CONVERT</td>
<td>Changes a specified ASCII decimal code to another ASCII decimal code.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>COOKIE</td>
<td>Reads and writes internet cookie information on an agent’s PC. An optional verb.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>COPY</td>
<td>Duplicates a field, text, or value and stores the value in another field or variable.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>COUNT</td>
<td>Determines the occurrences of a value within a string of values.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>COUNT-VALUES</td>
<td>Determines the number of values stored in a multivalued field.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>COUNTER</td>
<td>Defines actions for a counter object.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Verb</td>
<td>Description</td>
<td>Logic</td>
<td>Task</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>DATA-CONFERENCE</td>
<td>Starts or stops electronic conferencing between two users.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>DATA-TRANSFER</td>
<td>Transfers data from a UNIX file to an EDGE file.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DATE-FORMAT</td>
<td>Formats a date for display as either internal or external format.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DECRYPT</td>
<td>Performs decryption of a field, variable or string literal.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DB-TRACE</td>
<td>Controls the SQL trace facility of an Oracle database session. An optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DDE</td>
<td>Allows EDGE Client to communicate with Windows applications that support DDE.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>DEBUG</td>
<td>Tests logic flow or task statements.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DELETE-RECORD</td>
<td>Removes a record from a user-defined file and clears the workspace.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DELETE-VALUE</td>
<td>Removes one value in a multivalued field.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
### Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIGEST</td>
<td>Takes a field, variable or literal value and applies a hash function to it.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DISPLAY-CROSS-REFERENCE</td>
<td>Displays a list of cross-reference records for the file.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>DISPLAY-VALUES</td>
<td>Displays the values in a field on the terminal.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DIVIDE</td>
<td>Divides one number by another number.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EDGE</td>
<td>Allows advanced EDGE developers to build systems that can be used to administer EDGE installations and guides. An optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EMAIL</td>
<td>Sends and receives e-mail.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ENCRYPT</td>
<td>Performs encryption on a field, variable or string literal.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EXECUTE</td>
<td>Performs commands within a logic flow or task statement.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EXECUTE-ADHOC</td>
<td>Executes a single ad hoc report or a batch of ad hoc reports.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
### Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXECUTE-STORED-PROCEDURE</td>
<td>Allows Oracle stored procedures to be executed from EDGE. An optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EXECUTE-TASK</td>
<td>Executes a task from a guide or from another task.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>EXIT</td>
<td>Quits the logic flow or task.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FIND-VALUE</td>
<td>Locates the value position of a specified value in a multivalued field or variable.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FLAT-FILE-ACCESS</td>
<td>Allows access to a flat file for reading or writing on the server or client workstation.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FORMAT-INPUT</td>
<td>Specifies the format in which the data will be stored.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FORMAT-OUTPUT</td>
<td>Specifies the format for the data when displayed or printed.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FORMAT&amp;CONCAT</td>
<td>Builds a formatted, multivalued string.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>GET-CALL-RECORD</td>
<td>Retrieves the value of an existing call ID.</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
### Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET-CROSS-REFERENCE-DATA</td>
<td>Retrieves a list of formatted information based on the display fields identified for the cross-reference file.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>GET-KEY</td>
<td>Retrieves the key (record ID) based on the primary key, secondary key, or cross-reference value.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GET-NEXT-ELEMENT</td>
<td>Retrieves the next entry from the open selection list or field and stores the entry in a field or variable.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GET-PROPERTY</td>
<td>Retrieves the properties of an ActiveX property.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>GET-RECORD-TSTAMP</td>
<td>Retrieves the timestamp for a record in an EDGE database file.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GOTO</td>
<td>Moves to the specified label and continues processing the logic flow or task.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>GROUP-EXTRACT</td>
<td>Finds and extracts a group or groups of data separated by a delimiter.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>INSERT-VALUE</td>
<td>Places a value at a specified position in a multivalued field.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Verb</td>
<td>Description</td>
<td>Logic</td>
<td>Task</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>INTEGER</td>
<td>Changes a decimal number to a whole number.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>IRA</td>
<td>Calculates the return on an individual retirement account.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>LOAD-LIST</td>
<td>Transfers a list of call events contained in a field or selection list into a field or variable.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>LOCATE-BY-ACP</td>
<td>Locates locations using the telephone area code. An optional verb.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>LOCATE-BY-ZIP</td>
<td>Locates locations using the zip code number. An optional verb.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>LOG</td>
<td>Logs messages to a specified log file.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>LOOP</td>
<td>Performs loops within a logic flow or task.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>MENU</td>
<td>Adds or modifies a menu item.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>MESSAGE</td>
<td>Displays a message using a standard Windows message box.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>MONEY-CONVERSION</td>
<td>Formats a number with a currency format.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>MSGQUEUE</td>
<td>Sends messages to and receives messages from a server.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Verb</td>
<td>Description</td>
<td>Logic</td>
<td>Task</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>MULTIPLY</td>
<td>Multiplies a number by another number.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>OPEN-LIST</td>
<td>Opens a list at the beginning of the list.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>OPERATOR-ACCESS</td>
<td>Sets options associated with operator access.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>PASSWORD</td>
<td>Verifies a password for a specified user ID.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PICTURE-FORMAT</td>
<td>Defines a pattern for storing values.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>POWER</td>
<td>Raises a number by a specific power.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PRINT-CORRESPONDENCE</td>
<td>Prints a correspondence document.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>PRINT-VALUES</td>
<td>Sends the values of a single or multivalued field to a printer.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>QUEUE</td>
<td>Defines the scheduling and status of a queue.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>QUOTES</td>
<td>Surrounds a value with quotation marks.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RANDOM</td>
<td>Computes a random number between 1 and a specified number.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
**Table 16-1. Verb List (continued)**

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ-RECORD</td>
<td>Reads a record from a user-defined file.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RECORDING</td>
<td>Records agents’ conversations. Optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>REFRESH</td>
<td>Updates EDGE client screens with the current values of any updated screen objects.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>RELEASE-RECORD</td>
<td>Unlocks a record that was read with a lock.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>REMOVE</td>
<td>Deletes a value from a multivalued field based on the value, not the position, in the field.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>RETURN-VALUE</td>
<td>Returns a value to the calling logic flow. Can only be used by a logic flow called by the CALL-LOGIC verb, and whose parameter definition specifies that it returns a value.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>ROLL</td>
<td>Moves values in a multivalued field to the next value position.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ROUTE</td>
<td>Positions the operator’s cursor to an input field on the same or a different screen.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Verb</td>
<td>Description</td>
<td>Logic</td>
<td>Task</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>SCREEN-SAVER</td>
<td>No longer used. Retained for backward compatibility.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SEND-TO-VOAD</td>
<td>No longer used. Retained for backward compatibility.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SET-CLASSIFIER</td>
<td>Specifies the printer and number of copies to print.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SET-PROPERTY</td>
<td>Allows you to temporarily set some of the properties for an object when a guide is performed.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SHOW-MESSAGE</td>
<td>Displays a message to the operator.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>SLEEP</td>
<td>Pauses the logic flow or task for a specified number of seconds.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SORT-VALUES</td>
<td>Orders the contents of a multivalued field in ascending or descending order.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SOUNDEX</td>
<td>Converts a string to its soundex equivalent.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SPECIAL-SOUNDEX</td>
<td>Composes a soundex-type combination based on a field and a single character.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
### Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL-GENERAL</td>
<td>Executes an SQL statement that is not a SELECT statement. An optional verb.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>SQL-SELECT</td>
<td>Specifies a select query to read data from relational database tables or c-tree files.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>SQUARE-ROOT</td>
<td>Calculates the square root function.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>SR.TRANS</td>
<td>Stores information from user-defined fields to the REPORTING file for use with Sales Analysis Project Monitor.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>STRING-LENGTH</td>
<td>Determines the length of a string.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>SUBTRACT</td>
<td>Computes the difference between two numbers.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>SUM</td>
<td>Adds the numbers in a multivalued field.</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>TEKNEKRON-COMPLETE</td>
<td>Indicates the call has been completed to the TEKNEKRON equipment.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>TEKNEKRON-DIAL</td>
<td>Identifies information needed to dial the call to the TEKNEKRON equipment.</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td>Verb</td>
<td>Description</td>
<td>Logic</td>
<td>Task</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>TELEPHONY</td>
<td>Identifies the telephony device, callset, and specific commands and arguments. An optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TERMINATE-TASK</td>
<td>Stops a task.</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>TEXT-EXTRACT</td>
<td>Determines a value by specifying a value to search, the starting position, and the number of characters to extract.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TIME-FORMAT</td>
<td>Converts time data to an internal or external format.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TRANSACTION</td>
<td>Enables transaction mode, which provides database consistency. An optional verb.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TRANSFER-RECORD</td>
<td>Copies an impression of a record from one file to another file.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TRIM</td>
<td>Removes extraneous, leading, or trailing spaces from data.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>UAP</td>
<td>No longer used. Retained for backward compatibility.</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>VOAD-DIAL</td>
<td>No longer used. Retained for backward compatibility.</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>
### Table 16-1. Verb List (continued)

<table>
<thead>
<tr>
<th>Verb</th>
<th>Description</th>
<th>Logic</th>
<th>Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOAD-DIAL-ATT</td>
<td>No longer used. Retained for backward compatibility.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>VOAD-HANGUP</td>
<td>No longer used. Retained for backward compatibility.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>WINDOW</td>
<td>Creates, activates, opens, closes, positions, sizes, and displays windows.</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>WRITE-RECORD</td>
<td>Creates or updates a record in a user-defined file.</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
## Verbs by Function

The following sections provide a listing of verbs related by function.

### Call ID/Event and Record ID Processing
- ASSIGN-NEW-CALL-ID
- CALL-EVENT
- CHANGE-CALL-ID
- GET-CALL-RECORD
- GET-KEY
- GET-NEXT-ELEMENT
- LOAD-LIST
- OPEN-LIST

### Data Formatting and Conversion
- CHARACTER-CONVERSION
- CONVERT
- DATE-FORMAT
- FORMAT-INPUT
- FORMAT-OUTPUT
- FORMAT&CONCAT
- INTEGER
- MONEY-CONVERSION
- PICTURE-FORMAT
- QUOTES
- TIME-FORMAT
- TRIM

### Field and Variable Processing
- APPEND
- INSERT-VALUE
Verbs

<table>
<thead>
<tr>
<th>Verbs by Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPEND-VALUE</td>
</tr>
<tr>
<td>COMBINE</td>
</tr>
<tr>
<td>COPY</td>
</tr>
<tr>
<td>COUNT</td>
</tr>
<tr>
<td>COUNT-VALUES</td>
</tr>
<tr>
<td>DELETE-VALUE</td>
</tr>
<tr>
<td>FIND-VALUE</td>
</tr>
<tr>
<td>GROUP-EXTRACT</td>
</tr>
<tr>
<td>REMOVE</td>
</tr>
<tr>
<td>ROLL</td>
</tr>
<tr>
<td>SORT-VALUES</td>
</tr>
<tr>
<td>SOUNDEX</td>
</tr>
<tr>
<td>SPECIAL-SOUNDEX</td>
</tr>
<tr>
<td>STRING-LENGTH</td>
</tr>
<tr>
<td>TEXT-EXTRACT</td>
</tr>
</tbody>
</table>

Financial and Mathematical Calculations

| ADD                  |
| CALCULATE            |
| COMPOUND-INTEREST    |
| DIVIDE               |
| INTEGER              |
| IRA                  |
| MONEY-CONVERSION     |
| MULTIPLY             |
| POWER                |
| RANDOM               |
| SQUARE-ROOT          |
| SUBTRACT             |
| SUM                  |

EDGE 7.11 Developer Reference Manual 1018
### Guide Processing

<table>
<thead>
<tr>
<th>Verb</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASK</td>
<td>OPERATOR-ACCESS</td>
</tr>
<tr>
<td>CALL</td>
<td>QUEUE</td>
</tr>
<tr>
<td>DATA-CONFERENCE</td>
<td>REFRESH</td>
</tr>
<tr>
<td>DISPLAY-CROSS-REFERENCE</td>
<td>ROUTE</td>
</tr>
<tr>
<td>DISPLAY-VALUES</td>
<td>SET-PROPERTY</td>
</tr>
<tr>
<td>LOCATE-BY-ACP</td>
<td>SR.TRANS</td>
</tr>
<tr>
<td>LOCATE-BY-ZIP</td>
<td>SHOW-MESSAGE</td>
</tr>
<tr>
<td>MENU</td>
<td>WINDOW</td>
</tr>
<tr>
<td>MESSAGE</td>
<td></td>
</tr>
</tbody>
</table>

### Record Processing

<table>
<thead>
<tr>
<th>Verb</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLEAR-RECORD</td>
<td>RELEASE-RECORD</td>
</tr>
<tr>
<td>DELETE-RECORD</td>
<td>TRANSACTION</td>
</tr>
<tr>
<td>FLAT-FILE-ACCESS</td>
<td>TRANSFER-RECORD</td>
</tr>
<tr>
<td>READ-RECORD</td>
<td>WRITE-RECORD</td>
</tr>
</tbody>
</table>

### Special Processing

<table>
<thead>
<tr>
<th>Verb</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>3270-MONITOR</td>
<td>GET-CROSS-REFERENCE-DATA</td>
</tr>
<tr>
<td>Verbs by Function</td>
<td></td>
</tr>
<tr>
<td>-------------------</td>
<td>---</td>
</tr>
<tr>
<td>3270-PLAY</td>
<td>GET-RECORD-TSTAMP</td>
</tr>
<tr>
<td>AND-LISTS</td>
<td>GOTO</td>
</tr>
<tr>
<td>CASES</td>
<td>LOG</td>
</tr>
<tr>
<td>COOKIE</td>
<td>LOOP</td>
</tr>
<tr>
<td>COUNTER</td>
<td>PASSWORD</td>
</tr>
<tr>
<td>DEBUG</td>
<td>PRINT-CORRESPONDENCE</td>
</tr>
<tr>
<td>EDGE</td>
<td>PRINT-VALUES</td>
</tr>
<tr>
<td>EXECUTE</td>
<td>SET-PRINTER</td>
</tr>
<tr>
<td>EXECUTE-ADHOC</td>
<td>SLEEP</td>
</tr>
<tr>
<td>EXECUTE-TASK</td>
<td>TERMINATE-TASK</td>
</tr>
<tr>
<td>EXIT</td>
<td></td>
</tr>
</tbody>
</table>

**Telephony and Communication Functions**

<table>
<thead>
<tr>
<th>Verbs by Function</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CALL-EXTERNAL-FUNCTION</td>
<td>MSGQUEUE</td>
</tr>
<tr>
<td>COMMUNICATE-(RECEIVE)</td>
<td>RELEASE-RECORD</td>
</tr>
<tr>
<td>DATA-TRANSFER</td>
<td>TEKNEKRON-COMPLETE</td>
</tr>
<tr>
<td>DDE</td>
<td>TEKNEKRON-DIAL</td>
</tr>
<tr>
<td>EMAIL</td>
<td>TELEPHONY</td>
</tr>
</tbody>
</table>
Relational Database Processing

- COMPOSITE-KEY
- DB-TRACE
- EXECUTE-STORED-PROCEDURE

SQL-GENERAL
SQL-SELECT
The 3270-MONITOR verb monitors 3270 activity. For more information, see the documentation for the 3270 Gateway, which is available separately as an optional module.
The 3270-PLAY verb can be used to play a script which gets data from or puts data to a 3270 application screen or to start a passthru session. For more information, see the documentation for the 3270 Gateway, which is available separately as an optional module.
The ADD verb adds two numbers and stores the result in a field or variable.

**Properties:**

- **Add the value**: The number to add; can be a number, or a field or variable containing the number.
- **To the value**: The number to be added; can be a number, or a field or variable containing the number.
- **Store result in**: The location in which to store the result of the addition; can be a field or variable. The result can be stored back to one of the fields or variables containing the value to be added.
**Notes**

If one of the values is not numeric, the following results occur:

- If a null value (""") is added, it is treated as a zero (0). For example, null added to 2 results in 2.
- If a non-numeric value is added, the add is not done. For example, ABC added to 2 results in null.

**See Also**

CALCULATE verb.
The AND-LISTS verb compares two lists and produces a list whose values are in both lists.

Properties:

- **List 1**: The first list to be compared; can be a literal, or a multivalued field or variable containing the list. The items in the list must be delimited by value marks.

- **List 2**: The second list to be compared; can be a literal, or a multivalued field or variable containing the list. The items in the list must be delimited by value marks.

- **Store result in**: The location in which to store the resulting list; can be a multivalued field or variable. The items in the list are delimited by value marks.
Notes

The AND-LISTS verb can be used, for example, with the GET-KEY verb. The GET-KEY verb could be used first to produce a list of record IDs based on one set of criteria, such as last name = "SMITH", then used again to produce a list based on a second set of criteria, such as ZIP code = "92714". The two lists could then be used as input to the AND-LISTS verb to produce a list that meets both sets of criteria. In our example, the resulting list would contain all the record IDs with the last name of SMITH and ZIP code of 92714. In effect, this allows you to refine cross-referencing in a logic flow or task.

See Also

GET-KEY verb.
APPEND

The APPEND verb attaches a value to the end of another value and stores the result in a field or variable.

Properties:
- Append to the value of: The value that is to be followed by the value specified in the value of property; can be a number, a literal, or a field or variable containing the value.
- The value of: The value that goes at the end of the first value; can be a number, a literal, or a field or variable containing the value.
- Store result in: The location in which to store the resulting value; can be a field or variable.
**Notes**

This verb can be used, for example, to build record IDs; or to combine first and last names to create a full name field. To append two values with a space between, append a literal space to the end of the first value. Then append the second value to the previous result. The space separates the two values.

**See Also**

The CONCAT string function in the `CALCULATE` verb.
The APPEND-VALUE verb appends a value to the last position in a multivalued field or variable.

Properties:

- Append the value: The value to be appended to the last position of a multivalued field; can be a number, a literal, or a field or variable containing the value.

- To the end of the: The multivalued field or variable to which the specified value is appended; must be a multivalued field or variable.

The resulting value is stored back to this multivalued field or variable.
Notes

If the value of the target field or variable is an empty string, then the source value will append to it without adding a value mark. Use the INSERT-VALUE verb to add an empty string to any specific value position.

This verb could be used, for example, to store alternate phone numbers for a contact. You could also use this verb to store contact dates.

See Also

INSERT-VALUE and ROLL verbs.
The ASK verb displays a message and waits for the agent to respond.

**Properties:**

<table>
<thead>
<tr>
<th>Ask with prompt</th>
<th>The message to be displayed to the agent; can be a number, a literal, or a field or variable containing the message.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store result in</td>
<td>The logic flow or task waits until the agent responds. In EDGE Client, the message is displayed in a dialog box. In ASCII, the message line is displayed on the status line at the bottom of the screen.</td>
</tr>
</tbody>
</table>
Properties: (continued)

The number of characters, including spaces, should be 70 characters or less. If the message exceeds 70 characters, the remaining characters are truncated. The agent is reserved a minimum of 9 characters to return a response. If the message is less than 70 characters, the response can be longer.

For example, if the message is 10 characters, the response can be a maximum of 69 characters. After the line is filled, any keystroke returns the agent to the screen.

Store result in
The location in which to store the answer entered by the agent; can be a field or variable.

Conceal input
Specifies whether to conceal (not display) the agent’s input; can be one of the following:

- Yes The agent’s answer is concealed. This option can be useful, for example, to ask for a password or other confidential data.
- No The agent’s input is displayed.

Notes

Be cautious if you use this verb in a task. The task waits until someone answers the prompt and no other processing will occur while
the prompt is displayed. If this verb is run from the task scheduler, it is ignored and the next task statement is executed.

See Also

DISPLAY-VALUES verb.
ASSIGN-NEW-CALL-ID

The ASSIGN-NEW-CALL-ID verb creates a call ID based on a value and temporarily stores the value in $CALLID.

Properties:

From the value  The value to be used as the call ID; can be a number, a literal, a field or variable containing the value, or an asterisk (*) for a system-assigned number.

Notes

You can have the call ID reflect a customer account or other significant field by specifying that field as the value.

To have the system assign a sequential number, enter an asterisk (*) as the value. The system begins assigning numbers based on the cur-
rent number in the project counter. If records have not previously been assigned, the number is 1 (one). For more information about counters, see Chapter 2.

The following is an example of a logic flow that checks for a call ID which matches the value in ACCT.NO. If $CALLID is null, a new call ID is created using the ASSIGN-NEW-CALL-ID verb.

ALWAYS GET-CALL-RECORD with call ID of ACCT.NO
IF COMPARE if the value $CALLID is = the value ""
   THEN ASSIGN-NEW-CALL-ID from the value of ACCT.NO
   THEN COPY the value PHONE into S-TELNO

See Also
For information on modifying an existing call ID, see the CHANGE-CALL-ID verb.
The CALCULATE verb enables you to create complex expressions including:

- Arithmetic
- Logarithmic
- Array
- Statistical
- Bessel
- String
- Financial
- Trigonometric

The CALCULATE verb can be used in both logic and tasks; however, error handling properties are only available in logic flows.

This section includes the following:

- Calculation Results.
- Using the Keypad.
- Constructing Expressions.
- Testing Field and Variable Values.
**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calculate expression</td>
<td>The expression computed when the logic flow or task is performed. The expression can include operands, operators, and functions. The keypad may be used. For more information, see the section <strong>Using the Keypad</strong>. For more information on creating an expression, see the section <strong>Constructing Expressions</strong>.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the result of the calculation, which can be a number, an array, or a list; can be stored in a field or variable.</td>
</tr>
<tr>
<td>On error call logic flow</td>
<td>The name of the logic flow to call if there is an error; can be a literal, or a field or variable containing the name of the logic flow.</td>
</tr>
</tbody>
</table>
Properties: (continued)

If a logic flow is specified for this property and the Return on error property is yes, the specified logic flow is executed before continuing with the next line of the current logic flow.

This property is available only from a logic flow. This is an optional entry.

Return on error

Specifies whether to return to the logic flow if an error occurs. This property is available only from a logic flow; can be one of the following:

Yes  If an error occurs and a logic flow is specified, the called logic flow is executed before continuing with the next line of the current logic flow.

If an error occurs and a logic flow is not specified to be called, the CALCULATE verb statement is exited and the next line of the logic flow is executed.

No  If an error occurs, the logic flow is exited. There will not be a message that the logic flow has quit. If the error occurs in a logic flow that was called from another logic flow, only the called logic is exited.

Trace expression

Specifies whether to create a log of intermediate results when the CALCULATE verb is executed; can be one of the following:
Calculation Results

Properties: (continued)

- Yes  A file is created called calc.pid, where pid is the ID of the process currently executing on this port. The file is stored in the EDGE tmp directory. To find the pid, use the UNIX ps (process status) command.

- No  A log is not created.

Precision

During expression evaluation, all numbers and results of sub-expressions are stored internally using the maximum precision provided by the server. After an expression is evaluated, the result is returned with the standard logic precision, which is four decimal places. If the result is contained in a field, it is displayed according to the field format. While testing in the CALCULATE verb, a result can contain up to 11 places.

Error Handling

If an error occurs while the expression is being evaluated, the following system fields are updated:

- $LOGICERR  Contains the CALCULATE status code in the first value position and the UNIX system error number in the second value position, if appropriate.
$LOGICLINE Contains the line number in the logic flow where the error occurred.

$LOGICNAME Contains the name of the logic flow in which the runtime error occurred.

For a list of status codes, see Table 16-2.

Table 16-2. CALCULATE Status Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>Successful.</td>
</tr>
<tr>
<td>101</td>
<td>Syntax error.</td>
</tr>
<tr>
<td>102</td>
<td>Missing close parenthesis.</td>
</tr>
<tr>
<td>103</td>
<td>Missing close bracket.</td>
</tr>
<tr>
<td>104</td>
<td>Missing open parenthesis.</td>
</tr>
<tr>
<td>105</td>
<td>Missing open bracket.</td>
</tr>
<tr>
<td>106</td>
<td>Need open parenthesis.</td>
</tr>
<tr>
<td>107</td>
<td>Too few arguments.</td>
</tr>
<tr>
<td>108</td>
<td>Too many arguments.</td>
</tr>
<tr>
<td>109</td>
<td>Unterminated string.</td>
</tr>
<tr>
<td>110</td>
<td>Wrong argument type.</td>
</tr>
<tr>
<td>111</td>
<td>Invalid element.</td>
</tr>
</tbody>
</table>
**Using the Keypad**

To begin creating an expression or modifying an existing expression, select the three-dot button on the Calculate expression property. A screen resembling a calculator keypad is displayed. For an example, see Figure 16-1. While the keypad screen is active, the menu bar is disabled.

---

**Table 16-2. CALCULATE Status Codes (continued)**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>112</td>
<td>Database open failed.</td>
</tr>
<tr>
<td>113</td>
<td>Database read failed.</td>
</tr>
<tr>
<td>114</td>
<td>Too many symbols.</td>
</tr>
<tr>
<td>115</td>
<td>Domain error.</td>
</tr>
<tr>
<td>116</td>
<td>Unrecognizable error.</td>
</tr>
<tr>
<td>117</td>
<td>Division by zero.</td>
</tr>
<tr>
<td>118</td>
<td>Overflow.</td>
</tr>
<tr>
<td>119</td>
<td>Underflow.</td>
</tr>
<tr>
<td>120</td>
<td>Partial loss of significance.</td>
</tr>
<tr>
<td>121</td>
<td>Complete loss of significance.</td>
</tr>
<tr>
<td>122</td>
<td>Out of range error.</td>
</tr>
</tbody>
</table>
Figure 16-1.  CALCULATE Keypad

The keypad includes the following:
• Arithmetic and Embedded Functions.
• Cancel and OK Buttons.
• Display Area.
• Edit Buttons.
• File/Fields.
• Numeric Keypad.
• Separator Marks.
• Variables.

**Arithmetic and Embedded Functions**

Arithmetic functions are near the bottom left corner of the keypad. Several of the most common arithmetic and logarithmic functions are included. Use the drop down box to select a function. The function name and syntax will be entered in the display area. For more information regarding the types of arithmetic and embedded functions available, see Appendix Z.
**Cancel and OK Buttons**

The Cancel and OK buttons are at the bottom right corner of the keypad.

- **Cancel** Clears any expression that has not been previously saved and closes the keypad screen.
- **OK** Checks the syntax of the expression before exiting the screen. If the syntax is valid, the window is exited. If the formula is not valid, an error message is displayed. The message warns that the formula is not valid and asks if you want to save and exit anyway. Responses to the message can be one of the following:
  - **Cancel** The expression is not saved, but it continues to be displayed. The keypad window is not exited.
  - **OK** The expression is saved and the keypad window is exited.

**Display Area**

The large rectangular area at the top of the calculator keypad displays the expression. The expression can be entered from buttons on the keypad or from the keyboard. When you select a button to enter a function, the appropriate separator marks are entered for you. When you enter an expression from the keyboard, you must enter the name of the function and any required separator marks. Arrow
keys can be used to move around the expression. If the expression is long, the scroll bars automatically become active.

**Edit Buttons**

The edit buttons are located at the top right corner under the display area of the keypad and at the bottom left corner below the arithmetic functions. These buttons allow you to do a variety of editing functions. For a complete listing of the buttons and descriptions, see Table 16-3.

**Table 16-3. Edit Buttons**

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Clr" /></td>
<td><strong>Clr</strong>: Clears the entire display area. The clear action can be reversed if the Undo button is pressed immediately after the action.</td>
</tr>
<tr>
<td><img src="image" alt="Tab" /></td>
<td><strong>Tab</strong>: Positions the cursor or insertion point. If the expression contains parentheses, commas, or brackets, the cursor is positioned to the right of the next parenthesis, comma, or bracket. If none of these separator marks are present, the cursor is positioned to the right of the last character. Arrow keys can be used to move around the expression from the keyboard.</td>
</tr>
</tbody>
</table>
Undo: Removes the last entry, when selected the first time. If selected again, restores the entry that was removed. Continues to toggle between removing and restoring the last entry.

( ): Identifies the smallest expression to which the text belongs, based on the current insertion point or selected text. For example, in the expression \((6 \times 3) / 4\), if the cursor is positioned within the parentheses, the expression \((6 \times 3)\) is highlighted. Select again to see the next level of nesting. The action does not alter the expression.

<<: Highlights the previous (to the left) expression based on the current cursor position or the selected text.

>>: Highlights the next (to the right) expression based on the current cursor position or the selected text.

Syntax: Verifies the syntax of the entire expression. If there is an error, a message box reports the type of error, displays an error message, and highlights the expression. If there is no error, there is no response.

Test: Evaluates the entire expression. If the expression contains only constants, the result is displayed. For example, \(9 / 3\) returns the value 3. If the expression contains elements that are not constants, such as fields or variables, the value displayed in the expressions and values area is used.

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Undo</td>
<td>Undo: Removes the last entry, when selected the first time. If selected again, restores the entry that was removed. Continues to toggle between removing and restoring the last entry.</td>
</tr>
<tr>
<td>( )</td>
<td>( ): Identifies the smallest expression to which the text belongs, based on the current insertion point or selected text. For example, in the expression ((6 \times 3) / 4), if the cursor is positioned within the parentheses, the expression ((6 \times 3)) is highlighted. Select again to see the next level of nesting. The action does not alter the expression.</td>
</tr>
<tr>
<td>}</td>
<td>&lt;&lt;: Highlights the previous (to the left) expression based on the current cursor position or the selected text.</td>
</tr>
<tr>
<td>&gt;&gt;</td>
<td>&gt;&gt;: Highlights the next (to the right) expression based on the current cursor position or the selected text.</td>
</tr>
<tr>
<td>Syntax</td>
<td>Syntax: Verifies the syntax of the entire expression. If there is an error, a message box reports the type of error, displays an error message, and highlights the expression. If there is no error, there is no response.</td>
</tr>
<tr>
<td>Test</td>
<td>Test: Evaluates the entire expression. If the expression contains only constants, the result is displayed. For example, (9 / 3) returns the value 3. If the expression contains elements that are not constants, such as fields or variables, the value displayed in the expressions and values area is used.</td>
</tr>
</tbody>
</table>
The file, field, table, and Sys$ drop down boxes are located at the top right corner under the edit buttons. They can be used to specify the following:

- **File**
  
  Displays the available files. After selecting a file, the fields are available in the Field drop down box.

  The files that are displayed vary depending on the logic flow or task in which the CALCULATE verb is being used. In logic, only files available to the project are listed. In tasks, only files identified as available for access are listed.
Field Displays the available fields after a file selection is made. When a field is selected, the name is displayed inside single quotation marks, preceded by the file name and a dash. For example, 'PROD-BILL.AMT'.

Table Displays a list of EDGE tables available to the Array LOOKUP function. This function can be used to return a row of a table as an array. For more information on array functions, see Appendix Z. When a table name is selected, the name is displayed inside quotation marks. For example, “CODES.” For information on creating tables, see Appendix Z.

Sys$ Displays a list of the fields available in the SYS file. When selected, the field name is displayed inside single quotation marks.

**Numeric Keypad**

The numeric pad is on the left side of the keypad. The number pad includes:

- The numbers 0 (zero) through 9.
- The division sign (/).
- The multiplication sign (*).
- The subtraction sign (-).
- The addition sign (+).
The sign for a negative number (- sign).

- The decimal sign ( . ).

**Separator Marks**

The separator marks are at the left side of the keypad, below the numeric keypad. These buttons allow you to enter brackets, parentheses, and commas and to create complex mathematical expressions. For a description of the buttons, see Table 16-4.

**Table 16-4. Separator Marks**

<table>
<thead>
<tr>
<th>Mark</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[</td>
<td>Adds both a left and a right bracket and sets the insertion point between the brackets. To insert a left bracket only, enter the left bracket sign from the keyboard.</td>
</tr>
<tr>
<td>]</td>
<td>Adds a right bracket and positions the cursor to the right of the bracket. You can also enter the right bracket from the keyboard.</td>
</tr>
<tr>
<td>{</td>
<td>Adds both a left and a right parenthesis and sets the insertion point between the parentheses. To insert a left parenthesis only, enter the left parenthesis sign from the keyboard.</td>
</tr>
</tbody>
</table>
Variables

The Local, Call, and Global variable buttons are located on the right side of the keypad. When you select a variable button, the sign for the variable is displayed inside single quotation marks. The cursor is positioned after the sign so you can enter the name of the variable.

<table>
<thead>
<tr>
<th>Local</th>
<th>Enters '@'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Call</td>
<td>Enters '#'</td>
</tr>
<tr>
<td>Global</td>
<td>Enters '%'</td>
</tr>
</tbody>
</table>

Constructing Expressions

Expressions can include functions, operands, and operators. The standard precedence rules apply when expressions are evaluated:

- Expressions are evaluated from left to right.
- Multiplication and division expressions are evaluated before addition and subtraction expressions.

### Table 16-4. Separator Marks

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>')</td>
<td>Adds a right parenthesis and positions the cursor to the right of the parenthesis. You can also enter the right parenthesis from the keyboard.</td>
</tr>
<tr>
<td>', ...</td>
<td>Adds a comma followed by a space. This feature is useful in separating arguments inside an array expression. For example, [4, 3, 7].</td>
</tr>
</tbody>
</table>
Expressions within parentheses are always evaluated first.

Examples

1 + 2 = 3
1 + 2 * 3 = 7
(1 + 2) * 3 = 9
1 + (2 - 6) * (10 / 2) = -19
COUNT(1,2,4,6,.) = 4

Operands

Operands can be any of the following:

- Arrays.
- Fields and Variables.
- Numbers.
- The Result of Functions or Other Expressions.

Arrays

Arrays are lists of operands separated by commas (,) and enclosed in brackets [ ]. An array can include any number of operands, including another array. The result of using arrays in expressions depends on the number of arguments that the function takes.
When an array is specified in a function that takes a single argument, the result is a new array in which the operation has been performed on each element in the original array. For example, the square root function (SQRT) applied to an array results in the square root of each element in the array.

- `SQRT([4, 16, 25, 100])` results in `[2, 4, 5, 10]`
- `ABS([-3, 4, -18, 0, -7])` results in `[3, 4, 18, 0, 7]`
- `[-1, 2, 3, 4]` results in `[-1, -2, -3, -4]`

When an array is specified in a function that takes a fixed number of arguments, the result is a new array in which each element is the result of the function applied to the corresponding element in the original array with the other argument. In the power (POW) function, for example, each element in the array is raised by the power of the second argument.

- `POW([2, 4, 5, 10], 2)` results in `[4, 16, 25, 100]`
- `POW(2, [2, 4, 5, 10])` results in `[4, 16, 32, 1024]`
- `[1, 2, 3, 4] * 2` results in `[2, 4, 6, 8]`

If a function includes more than one array, the result is a new array with a size equal to the size of the smallest array, where each ele-
ment is a result of the function applied to the corresponding elements of the arguments.

\[
\begin{align*}
\text{POW([2, 3, 5], [1, 2, 3, 4])} & \quad \text{results in [2, 9, 125]} \\
\text{POW([1, 2, 3, 4], [2, 3, 5])} & \quad \text{results in [1, 8, 243]} \\
[1, 5, 15, 20] + [3, 6, 9, 12] & \quad \text{results in [4, 11, 24, 32]}
\end{align*}
\]

If a function includes a variable number of arguments and one of the arguments is an array, each element of the array is considered as a separate argument.

\[
\begin{align*}
\text{SUM([1,2,3,4],5,6)} & \quad \text{is equivalent to SUM(1,2,3,4,5,6) = 21} \\
\text{AVG([1,2,3,4],5,6)} & \quad \text{is equivalent to AVG(1,2,3,4,5,6) = 3.5} \\
\text{CONCAT(["a", "b" ],"c")} & \quad \text{is equivalent to CONCAT("a","b","c") = "abc"}
\end{align*}
\]

**Fields and Variables**

System and project fields and global, call, and local variables can be used as operands. When the CALCULATE verb is executed, the value of the field or variable is used in the expression.

If a field is multivalued, you can specify a value position by entering the field name followed by a comma, followed by the position num-
ber for the element. For example, FIELD,3 will access the third item in FIELD. You can also specify the value position with a field or variable which contains the number. Multivalued fields can also be accessed by using array formulas. For more information about array formulas, see Appendix Z.
Note: Fields and variables that are preceded by a file name or include a position number should be enclosed in single quotation marks. This is to eliminate ambiguities that the minus (-) and comma (,) characters might cause.

AGE Where AGE is a field in the default file.
'TEMP-VAL' Where VAL is a field in the TEMP file.
'T-VAL' Where VAL is a field in the file abbreviated as T.
'CUST-AGE' Where AGE is a field in the CUST file.
$CALLID Where CALLID is a field in the SYS file.
@TMP1 Where TMP1 is a local variable.
%TOTAL Where TOTAL is a global variable.
'TEMP-VAL,3' = 47 Where 'TEMP-VAL' contains 1^]123 Baseline Street^]47^]1234

Numbers

Numbers must begin with a digit, a decimal point, or a negative sign. Positive numbers cannot be preceded by a plus sign.

The number of digits displayed after the decimal point depends on the machine performing the calculation and the format associated
with the field where the result is stored. Because different machines may store floating point numbers with more or less precision, the results of examples shown here may vary slightly from what you experience on your system. The minimum number of digits returned for an expression is four.

8.0 Is a valid number.
+5 Is invalid because the plus sign is not allowed.
7.2.5 Is invalid because only one decimal point is allowed.

The Result of Functions or Other Expressions

An operand can be the result of a function or expression. For example, in the expression SQRT(19 * 7), the operand for the square root function is the expression 19 * 7. The expression (19 * 7) is evaluated first. The result of that expression, that is, 133, is passed to the square root function.

Operators

The four operators for addition, subtraction, multiplication, and division are +, -, *, and /. The precedence rules for arithmetic expressions require that multiplication and division be calculated before addition or subtraction.

5 + 3 = 8
200 * 1.25 = 250
Functions

Functions allow a transformation to be applied to a number of arguments. For more information about CALCULATE functions, see Appendix Z.

When a function is selected from the drop down box, the function is displayed as follows:

- Function name.
- Left parenthesis.
- The appropriate number of commas for the argument.
- Right parenthesis.

The cursor is positioned before the first comma in the function. After entering that argument, the Tab button can be used to move to the next argument. For functions with only one argument, the cursor is positioned between the parentheses.

Some functions take a fixed number of arguments. For example, the ROUND function takes two arguments, the number to be rounded and the number of decimal places to be rounded to.
Some functions take a variable number of arguments. For example, the SUM function can take any number of arguments. Each argument can be any valid operand.

Some functions require the arguments be of a certain type. For example, the ELEMENT function must be passed an array as the first argument and a number as the second argument.

Examples

\[
\text{IP}(10000, \frac{\text{RATE}}{\text{PERIODS}}, \ '@\text{TOTAL.PMTS}') = 1944.82
\]

where IP stands for Interest Paid, @RATE = .09, @PERIODS (per year) = 12, and @TOTAL.PMTS = 48.

\[
\text{ROUND}('\text{TOT.YR}', 2) = 389.38
\]

where ROUND stands for rounding a number, TOT.YR = 389.379.

\[
\text{FV}('\text{LOAN}', \frac{\text{RATE}}{12}, \ '@\text{TERM}') = 21.46
\]

where FV stands for Future Value, LOAN = 1500, @RATE = 12%, 12 stands for the number of periods, @TERM = 36.

Testing Field and Variable Values

To test an expression that contains fields and variables, set the value for fields and variables in the expressions and values area. To enable the expressions and values area, perform the following steps:

1. Select the Value button.
2. To display the fields and variables that are used in the expression, select the Test button. Each field and variable is displayed on its own line.

3. Change the values for the fields and variables as appropriate and select the apply button.

4. When you select the Test button again, the expression is tested using the values entered in the expressions and values.

The expressions and values area remains active as you continue to test and add to the expression. The scroll bars automatically become active if more variables are used in the expression. The following properties are used with the expressions and values area:

- **Apply**: Applies a new value when the expression is tested.
- **Clear**: Clears the values.
- **Expressions**: Displays the fields and variables used in the expression. This is activated by the test button.
- **Values**: Displays the value for the expression. Select the value to enter a new value.
The CALL verb transfers control to another logic flow.

**Properties:**
- **Call logic**  The name of the logic flow to be performed; can be a literal, or a field or variable containing the name of the logic flow.

**Notes**

The logic flow that identifies the logic flow to be performed is referred to as the calling logic. The logic flow that is identified to be performed is referred to as the called logic.
After the called logic is completed, the calling flow is continued, unless the called logic routes to a screen. If the called logic routes to a screen, the calling logic is not continued.

If the name of the called logic is in a field or variable, it cannot be verified during the scan of the logic flow. If at runtime the name is invalid or does not exist, the calling logic continues to execute as if the CALL had not been made. A message stating that the call statement could not be performed is sent to stderr.

The called logic can access data in the workspace and in the #variables and %variables. You cannot use @variables to share data between the logic flows because the values are retained only for the logic flow in which they occur.
The CALL-LOGIC verb transfers control to a logic flow that takes parameters. Unlike the CALL verb, this verb allows the calling logic flow to pass input, output or input/output parameters to the called logic flow, and receive a return value from it (see the RETURN-VALUE verb). The CALL-LOGIC verb allows more modular design of logic flows, making EDGE code easier to read. Without parameter passing, the calling logic flow communicates with the called logic flow using EDGE variables. When seeing a logic flow call another logic flow using the CALL verb, it is not clear which EDGE variables are modified by the called logic flow, but this interaction is made clearer by the parameter list and return value in the CALL-LOGIC verb.
**Properties:**

- **Logic flow**: The name of the logic flow to be performed; can be a literal, or a field or variable containing the name of the logic flow.

- **Input parameters**: The input parameters to be used with the logic flow; Each parameter can be a literal, or a field or variable containing a value. To specify each parameter, click the three-dot button which displays a grid of the parameters in the procedure.

- **Output parameters**: The output parameters to be used with the logic flow; can be a field or variable, but not a literal. To specify, click the three-dot button which displays a grid of the parameters in the logic flow.
Notes

The logic flow that identifies the logic flow to be performed is referred to as the calling logic. The logic flow that is identified to be performed is referred to as the called logic.

After the called logic is completed, the calling flow is continued, unless the called logic routes to a screen. If the called logic routes to a screen, the calling logic is not continued.

If the name of the called logic is in a field or variable, it cannot be verified during the scan of the logic flow. If at runtime the name is invalid or does not exist, the calling logic continues to execute as if the CALL had not been made. A message stating that the call statement could not be performed is sent to stderr.

Properties: (continued)

Input/output parameters

The input/output parameters to be used with the logic flow; each parameter can be a field or variable. To specify, each parameter click the three-dot button which displays a grid of the parameters in the logic flow.

Store status code in

The location in which to store the return value of the logic flow (the value specified by the Value to return property of the last RETURN-VALUE verb call in the called logic flow); can be a field or variable, but not a literal.
The called logic can access data in the workspace and in the #variables and %variables. Additionally, unlike the CALL verb, the called logic can access parameters (&variables) passed in by the calling logic flow. This is useful, because you cannot use @variables to share data between the logic flows because the values are retained only for the logic flow in which they occur.

The number and type of parameters is defined in the logic flow’s Parameter Definition.
The CALL-EVENT verb manages commands associated with call IDs and call events. Call events provide a method to relate multiple event records to a single customer record. A summary of the CALL-EVENT verb commands is listed in Table 16-5.
Properties:

- **Call event command**
  The available CALL-EVENT commands are listed in Table 16-5.

- **Call ID(s)/event(s)**
  The call ID used in the command; can be a field, variable, or literal. Most commands support multiple call events (from a multi-valued field or variable); the following commands only act on a single call event.
  - Locate Call Events.
  - Select Call Events.

The following commands do not support this property:

  - Check for Available Call.
  - Deselect Call Event.
  - Get Next Call Event.

- **Store status in**
  The location in which to store the result of the process, which can be one or more status codes. Can be stored in a field, multivalued field or variable. For status codes, see Table 16-7.

Notes

The **HISTORY File** is updated at $END with data from the last selected call event. If more than one call event was selected before $END, only
the last call event is reflected in the HISTORY file. Project monitoring data is not updated.

For more information on call events, see Chapter 14. For information about EDGE fields, see Chapter 12.

The actions that take place as a result of the CALL-EVENT commands are described in Table 16-6. The result may vary if the call event specified in the command is not currently selected. If the result of the action is "maybe," see the specific CALL-EVENT command for additional information.

Table 16-5. CALL-EVENT Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel Scheduled Events</td>
<td>Removes one or several call events from the queues.</td>
</tr>
<tr>
<td>Check for Available Call</td>
<td>Checks if an agent has a call available in any of the search queues identified in operator access.</td>
</tr>
<tr>
<td>Create Call Events</td>
<td>Creates a new call event for the call IDs.</td>
</tr>
<tr>
<td>Create Call IDs</td>
<td>Creates call IDs.</td>
</tr>
<tr>
<td>Delete Call Events</td>
<td>Deletes call events.</td>
</tr>
<tr>
<td>Delete Call IDs</td>
<td>Deletes call IDs.</td>
</tr>
</tbody>
</table>
### Table 16-5. CALL-EVENT Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deselect Call Event</td>
<td>Deselects and updates the currently selected call event.</td>
</tr>
<tr>
<td>Get Next Call Event</td>
<td>Retrieves the next call event available.</td>
</tr>
<tr>
<td>Locate Call Events</td>
<td>Locates all the call events associated with a specified call ID or call IDs.</td>
</tr>
<tr>
<td>Lock Call Events</td>
<td>Locks specified call events to prevent access.</td>
</tr>
<tr>
<td>Lock Call IDs</td>
<td>Locks specified call IDs to prevent access.</td>
</tr>
<tr>
<td>Schedule Call Events</td>
<td>Schedules one or more call events.</td>
</tr>
<tr>
<td>Select Call Events</td>
<td>Reads the STATUS, HISTORY, and RESPONSES records for the specified call event, sets the $CALLEVENT field to the specified value. Deselects the previously selected call event.</td>
</tr>
<tr>
<td>Unlock Call Events</td>
<td>Releases the lock on the specified call events.</td>
</tr>
<tr>
<td>Unlock Call IDs</td>
<td>Releases the lock on the specified call IDs.</td>
</tr>
</tbody>
</table>
### Table 16-6. CALL-EVENT Actions

<table>
<thead>
<tr>
<th>Call Event Command</th>
<th>Current Values Updated(^1)</th>
<th>Previous Values Updated(^2)</th>
<th>Locked Call Events Updated(^3)</th>
<th>Locked Call IDs Updated(^4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check for Available Call</td>
<td>No.</td>
<td>No.</td>
<td>No.</td>
<td>No.</td>
</tr>
<tr>
<td>Create Call Events</td>
<td>Yes.</td>
<td>Yes. With null.</td>
<td>Yes.</td>
<td>No.</td>
</tr>
<tr>
<td>Create Call IDs</td>
<td>Yes.</td>
<td>Yes. With null.</td>
<td>Yes.</td>
<td>Yes.</td>
</tr>
<tr>
<td>Get Next Call Event</td>
<td>Yes.</td>
<td>Yes.</td>
<td>Maybe.</td>
<td>No.</td>
</tr>
<tr>
<td>Locate Call Events</td>
<td>No</td>
<td>No.</td>
<td>No.</td>
<td>No.</td>
</tr>
<tr>
<td>Lock Call Events</td>
<td>No.</td>
<td>No.</td>
<td>Yes.</td>
<td>No.</td>
</tr>
<tr>
<td>Lock Call IDs</td>
<td>No.</td>
<td>No.</td>
<td>No.</td>
<td>Yes.</td>
</tr>
<tr>
<td>Select Call Events</td>
<td>Yes.</td>
<td>Yes.</td>
<td>Yes.</td>
<td>No.</td>
</tr>
</tbody>
</table>
If the specified call event is currently selected, the current values are updated as follows: \$CALLEVENT, \$CALLID, and \$EVENTINDEX fields in the SYS file and the CALLEVENT, CALLID, and EVENTINDEX fields in the STATUS, HISTORY, and RESPONSES files.

If the specified call event is currently selected, the previous values for the call event are updated for the \$OLDQUEUE, \$OLDRESULT, \$PCBDATE, and \$PCBTIME fields in the STATUS file.

If locked call events are unlocked, the \$EVENTS field is updated in the SYS file to include only locked events.

If a call IDs is locked, the \$LCALLIDS field is updated in the SYS file.

<table>
<thead>
<tr>
<th>Call Event Command</th>
<th>Current Values Updated?</th>
<th>Previous Values Updated</th>
<th>Locked Call Events Updated?</th>
<th>Locked Call IDs Updated?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unlock Call IDs</td>
<td>No.</td>
<td>No.</td>
<td>No.</td>
<td>Yes.</td>
</tr>
</tbody>
</table>

1 If the specified call event is currently selected, the current values are updated as follows: $CALLEVENT, $CALLID, and $EVENTINDEX fields in the SYS file and the CALLEVENT, CALLID, and EVENTINDEX fields in the STATUS, HISTORY, and RESPONSES files.

2 If the specified call event is currently selected, the previous values for the call event are updated for the $OLDQUEUE, $OLDRESULT, $PCBDATE, and $PCBTIME fields in the STATUS file.

3 If locked call events are unlocked, the $EVENTS field is updated in the SYS file to include only locked events.

4 If a call IDs is locked, the $LCALLIDS field is updated in the SYS file.
Most of the CALL-EVENT commands return a status code in a user-defined field or variable. The codes are listed in Table 16-7.

**Table 16-7. CALL-EVENT Status Codes**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Applies to Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>The command was completed successfully.</td>
<td>All commands but Check for Available Call, Create Call Events, Create Call IDs, and Locate Call Events.</td>
</tr>
<tr>
<td>1000</td>
<td>No more call events can be created for this call ID. A maximum of 1,000,000 call events can be created for a call ID. For example, ABC<del>000001 through ABC</del>999999.</td>
<td>Create Call Events.</td>
</tr>
<tr>
<td>1001</td>
<td>Call event or ID is locked by another user.</td>
<td>Lock Call Events and Lock Call IDs.</td>
</tr>
<tr>
<td>1002</td>
<td>Call event or ID was already locked. (Warning only).</td>
<td>Lock Call Events and Lock Call IDs.</td>
</tr>
<tr>
<td>1003</td>
<td>One or more call events for the call ID is locked.</td>
<td>Lock Call IDs.</td>
</tr>
<tr>
<td>1004</td>
<td>One or more call events for the call ID already exist. (Create Call ID)</td>
<td>Create Call IDs.</td>
</tr>
</tbody>
</table>
The Cancel Scheduled Event command removes one or more call events from the queues.

**Properties:**

There are no properties specific to this command.

---

**Table 16-7. CALL-EVENT Status Codes (continued)**

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
<th>Applies to Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>1005</td>
<td>The call event does not exist.</td>
<td>Cancel Scheduled Events, Delete Call Events, Lock Call Events, Schedule Call Events, Select Call Events and Unlock Call Events.</td>
</tr>
<tr>
<td>1006</td>
<td>Call event or ID is null.</td>
<td>Create Call IDs, Deselect Call Event.</td>
</tr>
<tr>
<td>1007</td>
<td>Call event or ID was not locked. (Warning only).</td>
<td>Unlock Call IDs and Unlock Call Events.</td>
</tr>
<tr>
<td>1008</td>
<td>Telephone number is null or invalid.</td>
<td>Schedule Call Events.</td>
</tr>
<tr>
<td>1100</td>
<td>Not in the specified queue.</td>
<td>Reserved for future use.</td>
</tr>
<tr>
<td>1101</td>
<td>Invalid queue specified. The queue does not exist.</td>
<td>Cancel Scheduled Events, and Schedule Call Events.</td>
</tr>
<tr>
<td>1103</td>
<td>Invalid date or time. (Command failed).</td>
<td>Select Call Events.</td>
</tr>
</tbody>
</table>
Notes

This command does the following:

- The call event is removed from the queue in which it was scheduled.
- The QUEUE, QNAME, and INDEX fields in the STATUS records are set to null.
- The SCHED.DATE and SCHED.TIME fields in the STATUS records remain populated.

This command does not delete the record of the call event in the STATUS, HISTORY, and RESPONSES files. For example, this command allows you to save the information on the contact, without having to schedule a call.

Check for Available Call

The Check for Available Call command determines if an agent has a call available in any of the search queues defined for that agent.

Properties:

- There are no properties specific to this command.
**Call Event**

**Notes**

This command does not require any input values nor does it update any fields. This command does not read the call event record in the STATUS file.

This command returns a value based on the state of the queues for the agent at the time of the command. If the queues are shared with other agents, the state may change at any time.

A supervisor might use this command to see if more queues need to be opened for the agents. In this case, the supervisor must have the same queues as the agents, because the command does not allow you to specify the user ID for the queues. If the result of the command equals N, the supervisor could open more queues using the QUEUE verb.

The Create Call Event command creates a new call event for each specified call ID.

**Properties:**

There are no properties specific to this command.
Notes

Specify * as the call event for a system-generated call ID.

This command does the following:

• Creates a STATUS, HISTORY, and RESPONSES record for each call event created.
• Locks each call event. (The specified call IDs are not locked).
• Selects the last call event created.

If call events already exist, this command creates a call event with the next unused index. To create multiple events for the same call ID, specify a multivalued field that contains multiple occurrences of the same call ID.

If the call IDs do not exist before using this command, they are created. Subsequent command usage with the same IDs will create call events.

The call ID remains selected after creation. If multiple call IDs are created, the last call ID in the list remains selected.
Note: To allow immediate access to the newly created call events by other agents, use the Unlock Call Events command. If you do not use this command, the locks are automatically released at the end of the call.

Create Call IDs

The Create Call IDs command creates the zeroth call event for each specified call ID value. The call ID created for the zeroth event does not include a tilde (~) or event index.

Properties:

There are no properties specific to this command.

Notes

Specify * as the call event for a system-generated call ID.

This command does the following:

- Creates a STATUS record for each call ID created.
- Locks all of the created call IDs.
- Selects the call event associated with the last call ID created.
Usually, the call ID is based on an account number or some other unique number in a customer file. This allows you to relate your customer file with the call ID used in EDGE files.

The call ID remains selected after creation. If multiple call IDs are created, the last call ID in the list remains selected.

**Note:** To allow immediate access to the newly created call events by other agents, use the Unlock Call Events command. If you do not use this command, the locks are automatically released at the end of the call.

The Delete Call Events command deletes the call event and all associated records.

**Properties:**

There are no properties specific to this command.

**Notes**

This command does the following:

- Deletes the records associated with the call events in the STATUS, HISTORY and RESPONSES files.
- Deletes the queue entry if the call events were scheduled.
Deselects the call events before deleting them, if they were selected.

Removes deleted call events from the list of locked call events if any of the specified call events were locked.

To remove the call event from a queue, but retain the record in the STATUS, HISTORY, and RESPONSES files use the Cancel Scheduled Events command.

Delete Call IDs

The Delete Call IDs command deletes all the call IDs and call events associated with the call ID.

Properties:

There are no properties specific to this command.

Notes

This command does the following:

- Deletes the records associated in the STATUS, HISTORY and RESPONSES files for all the call events associated with the call ID.
- Deletes all of the queue entries for the call events associated with the call ID.
Deselect Call Event

If one of the call events to be deleted is currently selected, the call event is deselected before it is deleted.

- Removes deleted call IDs and call events from the list of locked call IDs and events and if any of the specified call IDs or events were locked.

The Deselect Call Event command deselects the currently selected call event and optionally updates the EDGE file records associated with the call event.

**Properties:**

- **Update record**
  - Specifies whether to update information associated with the record. Can be a literal, or a field or variable containing a Y or N.
  - **Y** The current record is written to the HISTORY and RESPONSES files and the fields listed in Table 16-8 and Table 16-9 are updated. (To have the HISTORY record rolled, the call event must be selected at the end of the call.)
  - **N** The fields are not updated.

Selecting Y does not place the call event in a queue. To schedule the call into a queue use Schedule Call Events.
Table 16-8. STATUS Field Updates

<table>
<thead>
<tr>
<th>STATUS Field</th>
<th>Update with</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>The date when the call event was selected.</td>
</tr>
<tr>
<td>END</td>
<td>The current time.</td>
</tr>
<tr>
<td>LASTSCREEN</td>
<td>The name of the current screen.</td>
</tr>
<tr>
<td>RESULT</td>
<td>The value for $RESULT, if set.</td>
</tr>
<tr>
<td>START</td>
<td>The time when the call event was selected.</td>
</tr>
</tbody>
</table>

If populated by the user other STATUS fields are updated. Any changes made are updated. (The data for these fields must be copied to the STATUS fields before the update occurs.)

Note: Although the TELNO field is updated (if populated), the ITZC field is not automatically updated. Use Schedule Call Events to set a callback and update the ITZC field.

Table 16-9. SYS Fields Updated

<table>
<thead>
<tr>
<th>SYSTEM Fields</th>
<th>Updated with</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CALLEVENT</td>
<td>Null.</td>
</tr>
<tr>
<td>$CALLID</td>
<td>Null.</td>
</tr>
<tr>
<td>$EVENTINDEX</td>
<td>Null.</td>
</tr>
<tr>
<td>$OLDQUEUE</td>
<td>Null.</td>
</tr>
</tbody>
</table>
This command does the following:

- Deselects the currently selected call event by setting $CALLEVENT and $CALLID to null.

- Does not unlock the record. (To unlock, use the Unlock Call Event command). (This indicates that the STATUS file CODE field remains A).

- If the Update Record property is Y, the fields in both Table 16-8 and Table 16-9 are updated.

If you route to $END after executing this command, but before selecting or creating another call event, a new system-assigned call ID is created.

You can also deselect a call event by using the Select Call Events command on another call event. The previously selected call event is

---

**Table 16-9. SYS Fields Updated (continued)**

<table>
<thead>
<tr>
<th>SYSTEM Fields</th>
<th>Updated with</th>
</tr>
</thead>
<tbody>
<tr>
<td>$OLDRESULT</td>
<td>Null.</td>
</tr>
<tr>
<td>$PCBDATE</td>
<td>Null.</td>
</tr>
<tr>
<td>$PCBTIME</td>
<td>Null.</td>
</tr>
</tbody>
</table>

---

**Notes**
**Get Next Call Event**

automatically deselected, but fields in the STATUS, SYS, HISTORY, and RESPONSES files are not updated.

The Get Next Call Event command retrieves and locks the next available call event in the agent's search queue.

**Properties:**

There are no properties specific to this command.

**Notes**

This command does the following:

- Retrieves and locks the next call event from the first non-empty search queue assigned to the agent.

This function is the same as pressing F3 from the $OP or $END screen. Just as for F3, the retrieved call event is locked when this command is issued.

The call event value is placed in the field specified in the Store status in property, as well as the SYS fields $CALLID and $CALLEVENT.
**Locate Call Events**

The Locate Call Events command finds the call event that matches the specified criteria.

**Properties:**

- **Queue(s)**
  - The queue to be searched; can be a literal, a field or variable containing the queue ID, or * for all queues.
  - Specify only a single queue or * for all queues. If a multivalued list of queues is specified, only the first queue in the list is searched.

- **Date(s)**
  - The schedule date to be located; can be a literal, a field or variable containing the dates, * for all dates, or null ("") for FIFO queues. If entered as a literal, the date must be in internal format. Cannot be left blank.
  - Specify only a single date, * for all dates, or null for FIFO queues. If a multivalued list of dates is specified, only the first date in the list is used.
  - (To convert a date to internal format, see the DATE-FORMAT verb.)

- **Time(s)**
  - The scheduled time to be located; can be a literal, a field or variable containing the times, * for all times, or null ("") for FIFO queues. If entered as a literal, the time must be in internal format. Cannot be left blank.
  - Specify only a single time, * for all times, or null for FIFO queues. If a multivalued list of times is specified, only the first time in the list is used.
Notes

The call events that meet the criteria are stored in the field or variable specified in the Store status in property; the call event value is *not* stored in $CALLID or $CALLEVENT. If there are no matches, the result is null.

A date or time criteria specified for a FIFO queue is ignored. All call events for the call ID in the queue are returned.

**Lock Call Events**

The Lock Call Events command locks specified call events to prevent access.

**Properties:**

<table>
<thead>
<tr>
<th>Wait for lock flag</th>
<th>Specifies whether to wait when the call event is locked by another agent; can be a literal, or a multivalued field or variable containing a Y or N for each call event.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Wait for the other agent to release the lock on the call event, then lock the call event.</td>
</tr>
<tr>
<td>N</td>
<td>Do not wait; returns a status code indicating the call event is locked.</td>
</tr>
</tbody>
</table>

To convert time to internal format, see the TIME-FORMAT verb.)
CALL-EVENT

Notes

This command does the following:

- Sets the CODE field in the STATUS file to A for each specified call event. (The call IDs are not locked).

This command prevents other agents from accessing the specified call events. Only the agent who locks the call events has access. If you do not unlock the call events when you are finished, they are unlocked at $END.

Lock Call IDs

The Lock Call IDs command locks specified call IDs and prevents access to all associated call events.

Properties:

Wait for lock flag

- Specifies whether to wait when the call ID is locked by another agent; can be a literal, or a multivalued field or variable containing a Y or N for each call event.

Y Wait for the other agent to release the lock on the call event, then lock the call event.

Properties: (continued)

If there are more call events than values, the last value is used for the remaining call events.
Properties: (continued)

N  Do not wait; returns a status code indicating the call event is locked.

If there are more call events than values, the last value is used for the remaining call events.

Notes

This command sets the CODE field in the STATUSID file to A for each specified call ID.

This command prevents other agents from accessing the specified call IDs and associated call events. Only the agent who locks the call IDs has access. If you do not unlock the call IDs when you are finished, the call IDs are unlocked at $END.

EDGE checks the STATUSID file to determine if a call ID is locked. If the CODE field for the call ID is set to A, all corresponding call events are locked.
Schedule Call Events

The Schedule Call Events command adds a call event to a queue at a specified date and time, according to the site’s or the customer’s time zone.

Properties:

Queue(s) The queues in which to schedule the call events; can be a literal, a field or variable containing the queue names, or * to use the current value from the QUEUE field in the STATUS file.

To schedule multiple call events to a single queue, enter a single queue name in this prompt. To schedule multiple call events to different queues, enter a queue name to correspond to each call event.

Date(s) The dates to schedule the call events, can be one of the following:

Literal The date must be in internal format.

* Uses the current SCHED.DATE in the STATUS file. Use only if the call event has been previously scheduled; otherwise SCHED.DATE will be null.

"" For FIFO queues, enter "" for the current system date.

Time(s) The times to schedule the events; can be one of the following:

Literal The time must be in internal format.
CALL-EVENT

Properties: (continued)

* Uses the current SCHED.TIME in the STATUS file. Use only if the call event has been previously scheduled; otherwise SCHED.TIME will be null.

"" For FIFO queues, enter "" for the current system time.

Use customer time zone

Specifies whether to schedule the call event based on the customer’s time zone; can be a literal, or a field or variable containing the value; can be one of the following:

Y The call is scheduled based on the customer’s time zone.

N The call is scheduled based on the site’s time zone.

The following are required when scheduling based on a customer’s time zone:

• The STATUS field TELNO must include the telephone number for the customer.

• The STATUS field COUNTRY must include the country code for the customer. If the field is null, the site’s country code is used.
### Properties: (continued)

- The Date and Time Properties must be based on the customer’s date and time. Do not specify to use SCHED.DATE and SCHED.TIME, because the values in these fields are adjusted to your site date and time and will not change.

If the schedule is based on the customer’s time zone, the following STATUS fields are updated:

- COUNTRYITZC
- DATEQNAME
- USESDSTQUEUE

The following SYS fields are updated:

- $HOURDIFF$OLDRESULT
- $QUEUE$PCBDATE
- $OLDQUEUE$PCBTIME
- $THEIRDATE$THEIRTIME

### Notes

This command does the following:

- Assigns specified call events to queues.
- Updates the QUEUE, SCHED.DATE, and SCHED.TIME fields in the STATUS file for each call event.
- If scheduled according to the customer time zone, updates additional fields. See fields listed on this and the previous page.

You can schedule call events that are unscheduled or currently scheduled. If currently scheduled, the call event is rescheduled with the values entered for the Queue, Date, and Time prompts. If you specify an asterisk (*) in the Date or Time prompts, the system uses the current value in the STATUS record for the call event.

If the number of values in Queues, Dates, and Times prompts is less than the number of call events in the list, the remaining call events are moved to the queue identified in the corresponding STATUS record.

---

**Caution!** A call event can be scheduled in only one queue at a time. If you schedule a call event to a second queue, the call event is canceled from the first queue.

---

To schedule call events to a FIFO (First-In-First-Out) queue, enter the null value (""”) to the date and time settings. If you do not enter a value, you will receive an assembly error. If you do specify a date and time, it will be ignored and the current system date and time is used.
The Schedule Call Events command can be used to move calls from one queue to another. To do so, use the Locate option to retrieve a call event list containing all call events in a specific queue. Then use Schedule Call Events, specifying the new queue name. Enter an asterisk (*) for the date and time properties to use the values of SCHED.DATE and SCHED.TIME.

**Select Call Events**

The Select Call Events command selects a call event so it can be updated.

**Properties:**

There are no properties specific to this command.

**Notes**

This command does the following:

- Reads the STATUS, HISTORY, and RESPONSES records for the specified call event.
- Sets the SYS fields $CALLID and $CALLEVENT to the specified value.
- Locks the call event.
- If a previous call event was selected, an implicit Deselect command is done with update set to "N".
This command can be used to pull calls from another agent’s protected queue (for example, CALLBACK/GST4).

**Caution!** Unlike the Get Next Call Event command, this command does not remove the call event from the queue.

**Unlock Call Events**

The Unlock Call Events command releases the lock on the specified call events.

**Properties:**

There are no properties specific to this command.

**Notes**

This command does the following:

- Unlocks the specified call events by putting a D in the CODE field of the STATUS record.
- If the specified call event is selected, it is deselected before it is unlocked and $CALLEVENT is set to null.
- Deletes the specified call events from $LEVENTS.
- The STATUS record fields are not updated.
Unlock Call IDs

The Unlock Call IDs command releases locks on the specified call IDs.

Properties:

There are no properties specific to this command.

Notes

This command does the following:

- Unlocks the specified call IDs by deleting the call IDs from $LCALLIDS.
- If the specified call ID is selected, it is deselected before it is unlocked and $CALLID and $CALLEVENT are set to null.
- Deletes the STATUSID records for the specified call IDs.
The CALL-EXTERNAL-FUNCTION verb calls a function that was written outside of EDGE. The function can be in a shared library on the server, in a DLL on the client, or a method in an ActiveX control. Both the library and the function must be defined in EDGE.

**Properties:**

- **Library name**: The EDGE name for the library or ActiveX control that contains the desired function; can be a literal, or a field or variable containing the name.

- **Function name**: The name of the function or ActiveX method; can be a literal, or a field or variable containing the name.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>For object</td>
<td>The name of the screen object.</td>
</tr>
<tr>
<td></td>
<td>This property is available only if the Library name is the name of ActiveX Controls.</td>
</tr>
<tr>
<td>Store status in</td>
<td>The location in which to store the result of the call; can be a field or variable. For a list of status codes, see Table 16-10.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the return value of the function; can be a field or variable.</td>
</tr>
<tr>
<td>{additional properties}</td>
<td>Additional properties may be displayed, depending on the definitions of the library or ActiveX configuration. If the parameter is defined as input only, can be a literal, field, or variable. Otherwise, can be a field or variable.</td>
</tr>
</tbody>
</table>

When you specify a function or method, any arguments in its parameter list are added to the properties sheet. For functions in server libraries, the CALL-EXTERNAL-FUNCTION verb is limited to 50 arguments. For functions in DLLs or for ActiveX methods, the CALL-EXTERNAL-FUNCTION verb is limited to 16 arguments.
Notes

If there are no functions listed when you click the Function name drop down box, verify that the following have been done:

- The Library name is specified in Libraries/Functions.
- If the Function name is a method for an ActiveX control, ensure that in the control’s configuration, the Show in verb properties sheet box is checked for each method.

If the debug mode is specified in the library or function definition, the function is not actually called; the debug values specified for the function are returned.

CALL-EXTERNAL-FUNCTION supports external function parameters that contain embedded system delimiters. If none of your external function parameters use embedded system delimiters, you may disable the embedded system delimiter support to improve performance. To disable support for embedded system delimiters, set the CEF_SERVER_ENCODE environment variable to "N". For more information on setting environment variables, see Appendix AB, EDGE Environment Variables.
### Table 16-10. Status Codes

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Function was loaded successfully.</td>
</tr>
<tr>
<td>null</td>
<td>If the result is null, it indicates the return value of the function is void.</td>
</tr>
<tr>
<td>1</td>
<td>Failed to open EXT.LIBRARY file.</td>
</tr>
<tr>
<td>2</td>
<td>Failed to open EXT_FUNCTION file.</td>
</tr>
<tr>
<td>3</td>
<td>Unknown client type.</td>
</tr>
<tr>
<td>-101</td>
<td>Too many arguments. For a server-based function, the maximum number of arguments is 50. For all other functions, the maximum is 16.</td>
</tr>
<tr>
<td>-102</td>
<td>Thunking error. Thunking is the process of converting a win16 API call to a win32 API call or vice versa.</td>
</tr>
<tr>
<td>-103</td>
<td>DLL not found. The DLL pointed to by the library was not found.</td>
</tr>
<tr>
<td>-104</td>
<td>Procedure not found. The specified function was not found in the DLL.</td>
</tr>
<tr>
<td>-105</td>
<td>Internal error. Something is wrong with the values being passed; for example, the characters cannot be handled by the communication protocol.</td>
</tr>
</tbody>
</table>
The CASES verb invokes the CBR Link module and allows you to search and retrieve information from a case library. For more information, see the documentation for the CBR Link module, which is available separately as an optional module.
The CHANGE-CALL-ID verb names an existing call ID to a specified value.

**Properties:**

- **To the value**: The value to use for the new call ID; can be a number, a literal, or a field or variable containing the value.

**Notes**

This verb updates the corresponding records in the STATUS, HISTORY, and RESPONSES files. If the call ID has associated call events, the call ID portion of the key is also changed.

Before using this verb, you must have the value of the call ID to be changed in $CALLID. To do so, use the **GET-CALL-RECORD** verb with
the value of the call ID to be changed or the CALL-EVENT verb with the Select Call Event command.

When using a value from a file as the call ID, use the READ-RECORD verb to get the new value.

See Also

The GET-CALL-RECORD verb, and the Select Call Event command for the CALL-EVENT verb.
The CHARACTER-CONVERSION verb converts data to a specified format.

**Properties:**

- **Convert the value in:** The value to be converted; can be a number, a literal, or a field or variable containing the value.
- **Using the format:** The type of conversion. See Table 16-11.
- **Store result in:** The location in which to store the converted value; can be stored back to the source field or variable, or to another field or variable.
**Notes**

This verb can also be used to convert imported data to another format.

This conversion changes the way the data is stored. For example, a dollar amount such as $1,928.87 can be converted to the numeric value of 192887. For a description of the formats and an example, see Table 16-11.

The conversions in Table 16-11 are based on the following example:

```
ABC,$12.59,xyz
```

**Table 16-11. Character Conversion Types**

<table>
<thead>
<tr>
<th>Types</th>
<th>Description</th>
<th>Converted to</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha Only</td>
<td>Upper and lowercase letters only.</td>
<td>ABCxyz</td>
</tr>
<tr>
<td>Lower Case</td>
<td>All characters, with uppercase letters converted to lowercase.</td>
<td>abc,$12.59,xyz</td>
</tr>
<tr>
<td>Non-Alpha</td>
<td>All characters except letters.</td>
<td>,$12.59,</td>
</tr>
<tr>
<td>Non-Numeric</td>
<td>All characters except numbers.</td>
<td>ABC,.,xyz</td>
</tr>
<tr>
<td>Numeric Only</td>
<td>Numeric data only, no $ or decimals.</td>
<td>1259</td>
</tr>
<tr>
<td>Upper Case</td>
<td>All characters, with lowercase letters converted to uppercase.</td>
<td>ABC,$12.59,XYZ</td>
</tr>
</tbody>
</table>
See Also

The following verbs also convert formats: DATE-FORMAT, FORMAT-INPUT, FORMAT-OUTPUT, PICTURE-FORMAT, TIME-FORMAT
The CLEAR-RECORD verb removes data from the workspace for the current record.

Properties:

For file       The file to be cleared from the workspace; must be a filename.

Notes

If you are accessing more than one record at a time, this verb allows you to clear the record from the workspace before accessing the next record. For information on accessing more than one record at a time, see Guide File Access in Chapter 24.
The CLEAR-RECORD verb does not delete data from a record in a file. To delete a record, use the DELETE-RECORD verb.

To save the information in the workspace, use the WRITE-RECORD verb before clearing the record.
The COMBINE verb combines values into a multivalued field without duplicating identical values.

**Properties:**

- **The values**
  - The first set of values to be combined; can be a number, a literal, or a multivalued field or variable containing the values to be combined.

- **With the values**
  - The second set of values to be combined; can be a number, a literal, or a multivalued field or variable containing the values to be combined.

- **Store result in**
  - The location in which to store the combined values; can be a multivalued field or variable.
In the following example, two lists of colors are combined, resulting in one list.

List 1: RED, BLUE
List 2: BLUE, RED, YELLOW
Result: RED, BLUE, YELLOW

You can also use the COMBINE verb to eliminate duplicate values from one multivalued list. To do so, enter null (""") as the first value to be combined. Enter the list containing the duplicate values as the second value in the verb.
The COMMUNICATE-(RECEIVE) verb performs processing with the AT & T 6500 controller.

Notes

For information on this verb, contact EDGE Customer Care.
The COMPOSITE-KEY verb allows you to extract data from a file or string to get the values for the composite key. For more information, see the documentation for EDGE DataBase Link, which is available separately as an optional module.
The COMPOUND-INTEREST verb returns the value of a fixed amount plus interest for a specified number of payment periods.

**Properties:**

- **Principal**: The amount to be borrowed; can be a number, or a field or variable containing the value. Decimals may be used, but do not include a dollar sign ($).

- **Annual percentage rate**: The annual percentage rate; can be a number with decimal places, or a field or variable containing the number. Do not include the percent sign (%); for example, enter 8.5 for 8.5%. 

Notes

The COMPOUND-INTEREST verb returns the total amount of the initial value plus interest (future value).

See Also

CALCULATE verb.
The CONVERT verb changes an ASCII decimal code to another ASCII decimal code.

Properties:
- **String to convert**
  The string to be converted; can be a field or variable containing the string. The converted string is stored back to this field or variable.
- **Convert character**
  The ASCII decimal code for the character being converted; can be a literal, or field or variable containing the ASCII code from 1 to 255. For a listing of ASCII codes, see Appendix Y.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>To character</strong> (decimal)</td>
<td>The ASCII decimal code to replace the converted character; can be a literal, or field or variable containing the ASCII code from 1 to 255.</td>
</tr>
</tbody>
</table>

The following codes are not converted to the character for the ASCII decimal equivalent. Instead, they are converted as follows:

- 28 and 252: Converts to a sub-value mark.
- 29 and 253: Converts to a value mark.
- 30 and 254: Converts to an attribute mark.

For a listing of ASCII codes, see Appendix Y.

### Notes

The converted decimal code is put back to the field or variable which was specified for the string to convert.

*Note: The ASCII decimal codes from 1 through 31 should be avoided because they represent control characters that can adversely affect the screen display or task.*

The CONVERT verb can be used, for example, to convert a string delimited by a character to a multivalued field, without having to use the APPEND-VALUE verb. A delimited string is text or numbers sepa-
rated by characters. For example, A;B;C;1;2;3 is a string delimited by semicolons. This string can be converted to a multivalued field in which each character occupies a value position.

*With International Character Sets*

For customers using the ISO 8859-1 international character set, the characters represented by the ASCII decimal codes for 28, 29, 30, 252, 253, and 254 are not converted to international characters.

*Note:* The ASCII decimal code for the delimiter in an EDGE multivalued field is 253.
The COOKIE verb writes and reads persistent client state HTTP cookie information on an agent’s PC. For more information, see the documentation for EDGE Internet, which is available separately as an optional module.
The COPY verb makes a copy of a value and stores it in another field or variable.

Properties:

Copy  The value to be copied; can be a number, a literal, a field or variable containing the value, or special syntax to copy a record.

To copy a single value from a multivalued field or variable, include the position number, separated by a comma. For example:

@VAR,3

To copy a record, enter the following syntax with the filename as the name of the file to be copied:
Notes

When copying multivalued fields or variables, the result of the copy will depend on whether the value position is specified for either or both of the multivalued fields. For example, if a multivalued field is copied to another multivalued field, without specifying value positions for either field, the first field completely overwrites the second field.
If you copy a multivalued field or variable to a single value field without specifying a value position, you will receive an assembly error.

For the results of copy actions with multivalued fields, see Table 16-12. The results are for @OLD, based on the values for @OLD and @NEW.

To copy null into a field, enter quotation marks with no space between the marks (""). Copying a null value to a field ensures that the field is cleared of values.

You might use COPY to:

- Copy data from a field in one file to a field in another file. You must be sure to WRITE the record to save the data.
- Clear a field, multivalued field, or variable by copying null ("" ) to the field or variable.
- Copy a message to the $ERROR.MSG field for display to the screen.
- Copy a record from one file to an identical file. You must read the record before the copy and write it after the copy to have the record saved.

**Table 16-12. Copy Verb Examples**

<table>
<thead>
<tr>
<th>Value</th>
<th>Pos</th>
<th>@OLD</th>
<th>@NEW</th>
<th>Copy @NEW to @OLD</th>
<th>Copy @NEW, 1 to @OLD</th>
<th>Copy @NEW to @OLD, 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>Y</td>
<td>Y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>Z</td>
<td>Z</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1. A
2. B
3. C
The COUNT verb counts the occurrences of a value within a string of values.

**Properties:**

- **The occurrences of**: The value to be counted; can be a number, a literal, or a field or variable containing the value.
- **Within the values of**: The string to be counted; can be a number, a literal, or a field or variable containing the value.
- **Store result in**: The location in which to store the number of occurrences, which is zero (0) if no matches are found; can be a field or variable.
Notes

This verb counts exact match occurrences only. For example, assume you want to count the occurrences of BLUE in a multivalued field containing the following values,

- BLUE
- BLUEGREEN
- YELLOWGREEN
- BLUEGRAY

In this example, the count is 1 because the BLUE appears once. The other entries are not exact matches.

In a single-valued field, there is only one string. In a multivalued field, each value is considered an occurrence.
COUNT-VALUES

The COUNT-VALUES verb counts the number of values in a multivalued field.

Properties:

Count the values in

Store result in

Notes

Null values are counted as a value. For example, the value A]B]]C]D is counted as five because there is a null value between B and C.
The COUNTER verb defines actions for a counter object.

![Verb Properties](image)
**Properties:**

- **Counter command**: The command to perform; can be one of the following:
  - Create Counter Command.
  - Current Counter Command.
  - Decrement Counter Command.
  - Delete Counter Command.
  - Increment Counter Command.

- **Counter name**: The name of the counter; can be a literal, or a field or variable containing the name of the counter.

- **Store status in**: The location in which to store the status of the command; can be a field or variable. For a list of status codes, see Table 16-13.

---

**Table 16-13. COUNTER Verb Status Codes**

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful operation.</td>
</tr>
<tr>
<td>-101000</td>
<td>Soft connection error. The operation can be tried again.</td>
</tr>
<tr>
<td>-102100</td>
<td>The specified counter does not exist.</td>
</tr>
<tr>
<td>-102101</td>
<td>The counter record does not exist in the EDGE_COUNTERS file.</td>
</tr>
<tr>
<td>Status</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>-102102</td>
<td>Failed to connect to the database while opening the counter.</td>
</tr>
<tr>
<td>-102103</td>
<td>The counter already exists.</td>
</tr>
<tr>
<td>-102110</td>
<td>The minimum value specified was not less than the maximum value.</td>
</tr>
<tr>
<td>-102111</td>
<td>The start value is outside of the range for the min and max values.</td>
</tr>
<tr>
<td>-102112</td>
<td>Failed to open EDGE_COUNTERS file.</td>
</tr>
<tr>
<td>-102115</td>
<td>An invalid step value was specified.</td>
</tr>
<tr>
<td>-102116</td>
<td>An invalid counter type was specified.</td>
</tr>
<tr>
<td>-102117</td>
<td>An invalid counter command was specified.</td>
</tr>
<tr>
<td>-102118</td>
<td>A DB_DEVICE was not specified for the counter.</td>
</tr>
<tr>
<td>-102119</td>
<td>An error occurred while retrieving counter information.</td>
</tr>
<tr>
<td>-102120</td>
<td>An error occurred while parsing the counter information.</td>
</tr>
<tr>
<td>-102121</td>
<td>An error occurred while deleting the counter's record.</td>
</tr>
<tr>
<td>-102122</td>
<td>Counter value is less than zero.</td>
</tr>
<tr>
<td>-102123</td>
<td>Maximum counter value exceeded.</td>
</tr>
<tr>
<td>-102124</td>
<td>An error occurred while retrieving the counter value.</td>
</tr>
<tr>
<td>-nnn</td>
<td>A database-specific error. For more information, check your database</td>
</tr>
<tr>
<td></td>
<td>documentation.</td>
</tr>
</tbody>
</table>
**Notes**

You can use the counter to assign unique sequential numbers as record IDs (primary keys) for records in a file, to count inventory items, or for any other purpose where you need to keep track of numbers. Counters can also be created through the EDGE Counter screen. For more information, see Chapter 2.

Counters are currently available with c-tree and all of the relational databases accessed with EDGE’s DataBase Link optional module. For more information on implementation and use of counters, see Chapter 2.

When you select a command, prompts for additional properties are displayed. For information on these properties, see the description of each command on the following pages.

The Create Counter command creates a counter and assigns a name to it. Once the counter is created, it cannot be modified.

**Properties:**

- **Database device ID**
  - The database device for which the counter is being created; can be a literal, or a field or variable containing the name. (For information on specifying database device names, see Chapter 3.)
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direction</td>
<td>Specifies whether the counter is to be incremented or decremented or both. Valid selections depend on the database type and can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>B</strong> Can be both incremented and decremented. For c-tree or Oracle.</td>
</tr>
<tr>
<td></td>
<td><strong>D</strong> Can only be decremented. For c-tree or Oracle.</td>
</tr>
<tr>
<td></td>
<td><strong>I</strong> Can only be incremented. For all databases.</td>
</tr>
<tr>
<td></td>
<td>The default is I.</td>
</tr>
<tr>
<td>Increment/decrement</td>
<td>The value by which the counter is incremented or decremented; can be a number, or a field or variable containing the number.</td>
</tr>
<tr>
<td></td>
<td>The default is 1. For Sybase, 1 is the only value allowed.</td>
</tr>
<tr>
<td>Minimum counter value</td>
<td>The value below which the counter cannot be decremented; can be a number, or a field or variable containing the value. If you have not specified cycling and this value is reached, an error occurs.</td>
</tr>
<tr>
<td></td>
<td>For an incrementing or both counter, the default is 1.</td>
</tr>
<tr>
<td></td>
<td>For a decrementing counter, the default is (-10^{26}), that is, 10 followed by 26 zeroes.</td>
</tr>
<tr>
<td></td>
<td>For c-tree, the error message, &quot;Counter name MIN count exceeded,&quot; is sent to stderr.</td>
</tr>
</tbody>
</table>
Properties: (continued)

**Note:** The Sparc platform has a default value of \(-10^{22}\).

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum counter value</td>
<td>The value above which the counter cannot be incremented; can be a number, or a field or variable containing the value. If you have not specified cycling and this value is reached, an error occurs. For an incrementing or both counter, the default is (10^{27}), that is, 10 followed by 27 zeroes. For a decrementing counter, the default is -1. For c-tree, the error message is “Counter name MAX count exceeded.”</td>
</tr>
</tbody>
</table>

**Note:** The Sparc platform has a default value of \(-10^{22}\).

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow counter cycling</td>
<td>Specifies whether the counter is to be reset to its minimum or maximum value when it reaches its limit, can be a literal, or a field or variable containing the indication; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Y</strong> Resets back to the minimum for incrementing counters, or the maximum for decrementing counters.</td>
</tr>
<tr>
<td></td>
<td><strong>N</strong> Does not reset. If you attempt to increment past the limit, a counter error is returned.</td>
</tr>
</tbody>
</table>
Notes

This command creates a counter when testing in live mode or in live operations. The counter is not created in test mode.

The Current Counter command returns a value which varies depending on the database type. See the notes below for details. This value is dynamic and can change frequently, depending on the activity on your system. This value should not be used as a basis for creating IDs, because it is not unique.

Properties: (continued)

The default is N.

This property is only available with Oracle counters.

Note: Caching has an impact on counter cycling. For more information on caching, see the section Counters.

Initial counter value

The initial value for the counter; can be a number, or a field or variable containing the number.

For an incrementing counter, the default is the minimum value. For a decrementing counter, the default is the maximum value.
If the Current Counter command is used with an Oracle counter, at least one decrement or increment counter command must be executed during the current session by the user invoking this command; otherwise Oracle returns an error.

*Caution!* If a guide is swapped, the decrement or increment command must be invoked after the swap before invoking this command, even if the user is still in the current session.

**Properties:**

- **Store result in** The location in which to store the current value of the counter; can be a field or variable.

**Notes**

The value returned as the current counter varies as follows:

- **c-tree** The current value of the counter.
- **Informix** The highest value in the table associated with the counter.
- **Oracle** The last value that was returned by an increment or decrement command done by this process.
- **SQL Server** The highest value in the table associated with the counter.
### Decrement Counter Command

The Decrement Counter command decrements a specified counter.

#### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using offset</td>
<td>The value by which the counter is to be decremented; can be a number, or a field or variable containing the offset.</td>
</tr>
<tr>
<td></td>
<td>The offset replaces the value specified when the counter was created, for this operation only.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the value of the counter after the decrement; can be a field or variable.</td>
</tr>
</tbody>
</table>

#### Notes

With Oracle, the first decrement (or increment) returns the initial counter value. The Decrement Counter command is not available with Sybase or Informix counters.
### Delete Counter Command

The Delete Counter command deletes a specified counter.

**Properties:**
There are no properties specific to this command.

### Increment Counter Command

The Increment Counter command increments a specified counter.

**Properties:**
- **Using offset**
  The value by which the counter is to be incremented; can be a number, or a field or variable containing the offset.
  The offset replaces the value specified when the counter was created, for this operation only.
  This property is only available for c-tree counters.
- **Store result in**
  The location in which to store the value of the counter after the increment; can be a field or variable.

**Notes**

With Oracle, the first increment (or decrement) returns the initial counter value.
The **DATA-CONFERENCE** verb starts and stops an electronic conference between two users.

### Properties:

- **Data conference mode**
  - Specifies whether to start or stop the data conference; can be one of the following:
  - **On** Starts the data conference.
  - **Off** Stops the data conference.

- **User to conference with**
  - The user ID of the conferenced person; can be a literal, or a field or variable containing the user ID.
When a data conference is started, the screen displayed on the initiator’s terminal is displayed on the conferenced person’s terminal. If the initiator changes screens, the screen also changes on the conferenced person’s terminal.

To start a conference, both users must be in a guide, but the conferenced person does not have to be in the same guide as the initiator. Either the initiator or the conferenced person can stop the conference.

The DATA-CONFERENCE verb is only available for ASCII terminals. Because of EDGE Client’s client/server architecture, the DATA-CONFERENCE verb is not functional for EDGE Client workstations.

**Display Modes**

There are three display modes available while using the DATA-CONFERENCE verb. The modes can be toggled in the following order: view, coach, and message. To toggle between modes, press CTRL+T.

**Properties: (continued)**

Store status in The location in which to store the status code; can be stored in a field or variable. See the status codes in Table 16-14.
The display mode is identified on the right-hand side of the error line. The modes are:

- **Coach**: Both the initiator and conferenced person can input to the screen, even at the same time.
- **Message**: The conferenced person can send a message to the initiator which is displayed on the error line, below the function keys.
- **View**: The conferenced person can view, but not enter data to the initiator's screen. The data conference begins in this mode.

**Commands**

After the conference is initiated, the conferenced person can use control commands to switch between modes and to lock out or enable the initiator. The commands are:

- **CTRL+C**: Quits the monitor session.
- **CTRL+L**: Locks and releases the unlock when in coach mode. When locked out, the initiator is prevented from entering keystrokes. When CTRL+L is pressed again, the lock is released and the initiator is able to enter keystrokes.
- **CTRL+T**: Toggles between display modes. The mode being used is displayed on the right-hand side of the error line. For a list of available modes, see Display Modes.
Example

A company wants all orders to be authorized by a supervisor. When an order is taken, the operator presses a function key which performs a logic flow that does the following:

- ASKs for the user ID to be conferenced.
- Sets DATA-CONFERENCE to that user ID.
- ROUTEs to the current screen.

The order screen is displayed on the supervisor’s terminal. The supervisor does the following:

- Presses `CTRL+T` to toggle to coach mode.
- Presses a function key that ASKs for the authorization code and conceals the entry.
- Enters a code which checks the code and if valid, allows the supervisor to continue. If not valid, an error message is displayed.
- Presses `CTRL+T` to toggle to message mode and inform the operator that the order can be completed.

The operator presses a function key which stops DATA-CONFERENCE and completes the order.
Error Checking

To take action if the conference does not start, check the status code immediately after invoking DATA-CONFERENCE. If the status code is greater than zero, the conferenced party is either already in conference or not logged in. You may want to return to the original screen and try to conference with another party.

Table 16-14. DATA-CONFERENCE Status Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful.</td>
</tr>
<tr>
<td>1</td>
<td>Invalid Process ID (pid). No such pid or pid is in use.</td>
</tr>
<tr>
<td>2</td>
<td>Valid process, but not in guide.</td>
</tr>
<tr>
<td>3</td>
<td>User is already conferenced.</td>
</tr>
<tr>
<td>4</td>
<td>Unknown error.</td>
</tr>
<tr>
<td>5</td>
<td>No such user.</td>
</tr>
<tr>
<td>6</td>
<td>User is not in EDGE.</td>
</tr>
</tbody>
</table>
The **DATA-TRANSFER** verb copies data from a UNIX file to workspace, or copies data from workspace to a UNIX file.
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>Specifies whether to export or import the data; can be one of the following:</td>
</tr>
<tr>
<td>Export</td>
<td>Copies data from the workspace to the file specified in the I/O device. When exporting, to get data to the workspace to transfer to a file, the record must be read.</td>
</tr>
<tr>
<td>Import</td>
<td>Copies data from the file specified in the I/O device to the workspace. When importing, to save the transferred data to an EDGE file, the record must be written.</td>
</tr>
<tr>
<td>Device ID</td>
<td>The I/O device name; can be a literal, or a field or variable containing the name of the device.</td>
</tr>
<tr>
<td>Full path name</td>
<td>The path name; can be a literal, or a field or variable containing the path name.</td>
</tr>
<tr>
<td>Block size</td>
<td>The block size identifies the number of characters to be imported; must be a number from 1 to 32767.</td>
</tr>
<tr>
<td></td>
<td>To determine the block size, multiply the number of records by the number of characters in one record. For example, for a file of 100 records with 80 characters per record, the block size should be 8000.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max. number of retries</td>
<td>The maximum number of times the transfer will be tried; must be a number from 0 to 100.</td>
</tr>
<tr>
<td></td>
<td>The default is 1.</td>
</tr>
<tr>
<td>Seconds between retries</td>
<td>The number of seconds to wait between attempts to transfer; must be a number from 0 to 100.</td>
</tr>
<tr>
<td></td>
<td>The default is 1.</td>
</tr>
<tr>
<td>Store status code in</td>
<td>The status code for the transfer; can be stored in a field or variable. For a listing of status codes, see Table 16-15.</td>
</tr>
<tr>
<td>Store status text in</td>
<td>The status message; can be stored in a field or variable. For a listing of status messages, see Table 16-15.</td>
</tr>
<tr>
<td>Delete file after import</td>
<td>Specifies whether to delete the file after it is imported to EDGE. Only available with the import option. Can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes The file is deleted after it is imported.</td>
</tr>
<tr>
<td></td>
<td>No The file is not deleted.</td>
</tr>
</tbody>
</table>
Properties: (continued)

File update mode

Indicates the type of mode for the transfer. This property is only available with the export option. One of the following can be specified:

- **Append**
  Adds data to the end of the file referenced within the I/O device.

- **Create**
  Creates a file if one does not exist. If used when a file already exists, you will receive an error and the logic flow or task is immediately exited.

- **Create or append**
  Creates a file if one does not exist or appends new data to the existing file.

- **Create or overwrite**
  Creates a file if one does not exist or overwrites an existing file.

- **Overwrite**
  Overwrites the existing file.

Fields and positions

The names of the fields to import data to or export data from. If the field is not in the default file, the field name must include the file abbreviation. From 1 to 50 fields can be specified.

Notes

You can use this verb only with fixed-length (flat) ASCII files.
To save the imported data, you must write it to a record. To export data, you must have data in the workspace; to get data to the workspace you must read a record.

**Table 16-15. DATA-TRANSFER Status Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>Successful transfer.</td>
</tr>
<tr>
<td>1</td>
<td>Device does not exist in the IODEVICE file.</td>
</tr>
<tr>
<td>2</td>
<td>Device is not a file device.</td>
</tr>
<tr>
<td>3</td>
<td>Cannot read/open device or timeout error.</td>
</tr>
<tr>
<td>4</td>
<td>Cannot write/open device or timeout error.</td>
</tr>
<tr>
<td>5</td>
<td>File is busy on timeout.</td>
</tr>
<tr>
<td>6</td>
<td>Specified record does not exist.</td>
</tr>
<tr>
<td>7</td>
<td>File already existed - cannot CREATE.</td>
</tr>
<tr>
<td>8</td>
<td>File does not exist - cannot OVERWRITE.</td>
</tr>
<tr>
<td>9</td>
<td>File does not exist - cannot APPEND.</td>
</tr>
</tbody>
</table>
DATE-FORMAT

The DATE-FORMAT verb formats and stores a date in either internal or external format, based on the date order option specified for your site.

Properties:

Format the date

The date to be formatted, which can be in internal or external format; can be one of the following:
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number</strong></td>
<td>A number representing the date in internal format. The date is stored as the number of days between that date and the zero date, which is December 31, 1967. Dates before 12/31/67 are stored as negative numbers; dates after 12/31/67 are positive numbers.</td>
</tr>
<tr>
<td><strong>Literal</strong></td>
<td>The numbers representing the month, day and year. The order of the elements (month, day, and year) is determined by the date order specified in Country Codes. US order is M D Y. International order is D M Y. For more information about specifying the date order, see Country Codes in Chapter 3.</td>
</tr>
<tr>
<td><strong>Field or variable</strong></td>
<td>A field or variable containing the date in either internal or external format. Numbers can be separated by any non-numeric, non-alphabetic separator.</td>
</tr>
<tr>
<td><strong>To format</strong></td>
<td>The type of format in which to store the date, which must be the opposite of the type entered as input; can be one of the following: If the date to be formatted is in external format.</td>
</tr>
</tbody>
</table>
Unlike field formatting, which only affects the display of the date, the DATE-FORMAT verb changes the date to the selected format and stores it in that format. The verb also provides more format options than the field definition formats.
To format an external formatted date to another external format, do the following:

- Use the DATE-FORMAT verb to format the external date to an internal date.

- Then use the DATE-FORMAT verb to format the internal date to an external date.

You can use the DATE-FORMAT verb for a variety of reasons, including:

- Display a date in different format than provided in the field definition for the date.

- Display the actual value of the internal date.

- Convert dates from an imported file or to a file to be exported.

- Use in correspondence when the field definition formats are not appropriate.

For information on how EDGE handles two-digit years, see the discussion of EDGE base year.
### Table 16-16. DATE-FORMAT Examples

<table>
<thead>
<tr>
<th>Format #</th>
<th>Based on MDY Date Order and CAPS Style</th>
<th>Stored Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>MMM DD YYYY</td>
<td>Feb 21 1995</td>
</tr>
<tr>
<td>2.</td>
<td>MM/DD/YYYY</td>
<td>02/21/1995</td>
</tr>
<tr>
<td>3.</td>
<td>MM-DD-YYYY</td>
<td>02-21-1995</td>
</tr>
<tr>
<td>4.</td>
<td>MM/DD/YY</td>
<td>02/21/95</td>
</tr>
<tr>
<td>5.</td>
<td>MM-DD-YY</td>
<td>02-21-95</td>
</tr>
<tr>
<td>6.</td>
<td>Date of month (1-31)</td>
<td>21</td>
</tr>
<tr>
<td>7.</td>
<td>Day of week (Mon = 1, Sun = 7)</td>
<td>2</td>
</tr>
<tr>
<td>8.</td>
<td>Day of week (Monday - Sunday)</td>
<td>Tuesday</td>
</tr>
<tr>
<td>9.</td>
<td>Month (1-12)</td>
<td>2</td>
</tr>
<tr>
<td>10.</td>
<td>Month (January - December)</td>
<td>February</td>
</tr>
<tr>
<td>11.</td>
<td>Quarter</td>
<td>1</td>
</tr>
<tr>
<td>12.</td>
<td>Year (YYYY)</td>
<td>1995</td>
</tr>
<tr>
<td>13.</td>
<td>Year (YY)</td>
<td>95</td>
</tr>
</tbody>
</table>
The DB-TRACE verb allows you to control the SQL trace facility of an Oracle session. For more information, see the documentation for the DataBase Link, which is available separately as an optional module.
The DDE verb allows EDGE Client to communicate with Windows applications that support DDE. For more information, see Chapter 17.
The DEBUG verb can be used to enter debug mode to test a guide or a task.

Properties:
None.

Notes
You can use DEBUG multiple times in a logic flow or task. Some typical places to use the DEBUG verb are:

- After lengthy calculations to check the data.
- At the beginning of a logic flow to determine if the guide is performing the logic flow.
- At the end of the guide.
Before a message is stored to the $ERROR.MSG field to check that the field is null.

Before an IF condition to check values.

Debug features vary depending on whether debug is invoked while testing a guide or from a task. Both operations and task features are described in this section.

**Debug in Guides**

Debug mode allows you to interrupt the guide to display and set values. You do not need to remove DEBUG statements before you swap the guide to operations. The DEBUG verb is ignored in operations and will not be seen by agents.

You can invoke debug mode in the following ways:

- By placing a DEBUG verb in a logic flow. When the DEBUG verb is encountered, a message similar to the following is displayed:

  ![Debug Logic Window](image)

  - By pressing CTRL+K anywhere in a guide.
When debug mode is entered, a screen similar to the following is displayed:

![Debug Screen](image)

You can display and set values in the following areas:

- File records.
- System fields ($fields$).
- Global variables ($%name$).
- Call variables ($#name$).
- Local variables (@name). To view values for local variables, you must use a DEBUG verb in the logic flow that contains the variable; you cannot use CTRL+K because local variables are stored only for the duration of the logic flow.

To change a value, double-click the line that contains the value. A dialog box similar to the following is displayed:

The value set through debug mode is not permanently stored, but is retained as follows:

- `#variables` and `fields` Through $END, then reset to null.
- `%variables` For the entire session until logoff.
- `@variables` Throughout the current logic flow, then reset to null.
**Debug in Tasks**

In a task, the DEBUG verb can be used to examine and update the values in fields at any point in the task process. The DEBUG verb can also be used to ensure data is properly stored and written to fields during the execution of a task. The DEBUG verb can be used multiple times within the same task.

*Note:* When invoked, DEBUG requires a response from the user, consequently, it should not be executed from the task scheduler. If the DEBUG verb is encountered by the task scheduler, it is ignored and the next task statement is executed.

When the DEBUG verb is encountered in a task, the prompt displays the name of the task and the line number of the DEBUG statement. You can enter one of the following commands:

- **A** Aborts the task.
- **I** Continues with the task and ignores all DEBUG statements in the task for future records.
- **N** Bypasses debug mode for this DEBUG statement.
- **Y** Invokes debug mode.

When you enter DEBUG mode from a task, the screen is cleared to display a list of file names and @ variables similar to the following:
The files are based on the File Access for the task.

To examine or edit the current values in a file, enter the number next to the file or @variable. To quit DEBUG mode, enter Q.

After you select a file or @variable, the current value is displayed. The fields are displayed according to their record position. If the field is multivalued, the number of values is displayed on the right side of the screen. If the data cannot be displayed on a single line on the screen, a plus (+) followed by the number of undisplayed bytes is displayed on the right side of the screen.
The possible actions when examining a record are listed in Table 16-17.

**Table 16-17. Record Display Commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Allows you to EDIT (temporarily change) record data.</td>
</tr>
<tr>
<td>F</td>
<td>Returns to the FILE menu to select another file.</td>
</tr>
</tbody>
</table>
If you enter an E (edit), a prompt similar to the following is displayed:

```
ENTER LINE#: LINE#,value#: (I)D: (R)ETURN:
```

To change the value of an entry while in the task, enter one of the following:

- **Line number**: Edits the field in the specified line.
- **Line number,pos**: Edits the value on the specified line and position.
- **I**: Edits the record ID value.
- **R**: Returns to the field display.

When you edit a field, the current data is displayed at the bottom of the screen, followed by a prompt to enter a replacement value. For
example, if the field in position 10 contains a telephone number, the prompt is similar to the following:

```
10: 2139266653
REPLACE WITH:  
```

To temporarily change the value, enter a new value or clear the current value by entering spaces. To permanently change the value, you must write the record.

If a line number is entered beyond the last line of the current record, the following message is displayed:

```
LINE # BEYOND END OF RECORD. (C)ONTINUE?
```

If you enter C, the value is entered at the specified line.
The DECRYPT verb is an optional verb that requires additional licensing. The module can be enabled anytime after the installation or upgrade of EDGE.

For more information on how EDGE encrypts and decrypts data, please refer to documentation for the 'ENCRIPT' verb.

DECRYPT performs decryption of a field, variable, or string literal (known as “ciphertext” data) using one of the supported ciphers (listed below) and stores the “plaintext” result into a field or variable. Before performing the cipher, the input value is first decoded to binary using Base64 or Hex decoding.
<table>
<thead>
<tr>
<th>Cipher</th>
<th>The name of the cipher to use. May be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES128</td>
<td>128-bit AES</td>
</tr>
<tr>
<td>AES192</td>
<td>192-bit AES</td>
</tr>
<tr>
<td>AES256</td>
<td>256-bit AES</td>
</tr>
<tr>
<td>BF</td>
<td>Blowfish</td>
</tr>
<tr>
<td>DES</td>
<td>56-bit Data Encryption Standard</td>
</tr>
<tr>
<td>DES-X</td>
<td>Extended Data Encryption Standard</td>
</tr>
<tr>
<td>DES3</td>
<td>Triple-DES</td>
</tr>
<tr>
<td>CAST</td>
<td>CASE-128</td>
</tr>
<tr>
<td>IDEA</td>
<td></td>
</tr>
<tr>
<td>RC2</td>
<td></td>
</tr>
<tr>
<td>RC5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Encoding</th>
<th>The type of text-to-binary encoding to be used on the input of the cipher. May be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base64</td>
<td></td>
</tr>
<tr>
<td>Hex</td>
<td></td>
</tr>
</tbody>
</table>
**DECRYPT**

<table>
<thead>
<tr>
<th>Store result in</th>
<th>A field or variable which will contain the resulting plaintext.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store status in</td>
<td>If less than zero, indicates that an error occurred while decrypting the data. Otherwise, the length of the output string is returned.</td>
</tr>
</tbody>
</table>

Example:

ALWAYS COPY the value “9Y0TIYfF9t+Rt++n1Mmf8Q==” into @CipherText
ALWAYS DECRYPT using cipher "DES3" cipher text @CipherText with passphrase "mysecret" encoding as base64 store result in @PlainText store status in @Status
IF COMPARE if the value @Status is >= the value 0 THEN MESSAGE title $PROJECT text @PlainText buttons 2 default button 1 icon 1 store result in @NULL

The output of the DECRYPT verb can be verified in a shell using the OpenSSL package as follows:

```
$ echo '9Y0TIYfF9t+Rt++n1Mmf8Q==' | openssl enc -des3 -nosalt -d -pass pass:mysecret -a
Hello World!
```
The DELETE-RECORD verb removes a specific record from a user-defined file and clears the record workspace.

Properties:

- **From the file**: The name of the file containing the record to be deleted; must be the file name. Do not enclose in quotation marks.
- **Delete record ID**: The value of the record ID to be deleted; can be a number, a literal, or a field or variable containing the value.
- **Store status in**: The location in which to store the flag; can be stored in a field or variable.
DELETE-RECORD

Properties: (continued)

Do not wait for lock

Specifies whether to wait to delete the record if it is locked by another operator; can be one of the following:

Yes

- The $LOCKED field is set to Y.
- The record is not deleted.
- The logic flow or task continues to the next step.

No

- The logic flow or task waits until the record is unlocked before continuing.
- After the record is unlocked, it is deleted.
- If the record is not locked, it is deleted without any wait.

Notes

The DELETE-RECORD verb deletes all the information in the specified record. It also clears the workspace of the current record, which may not be the record specified to be deleted.
See Also

For information on deleting records from EDGE files, see the Delete Call IDs or Delete Call Events commands for the CALL-EVENT verb.

For more information on the results of locking, see the READ-RECORD verb, Table 16-27.
The DELETE-VALUE verb removes one value from a multivalued field.

Properties:
- Multivalued field: The value to be deleted; can be a multivalued field or variable.
- At position: The position number of the value to be deleted; can be a number, or a field or variable containing the number.

Notes
After the value is deleted, the values following the deleted value are shifted. For example, if value 2 is deleted, value 3 becomes value 2, value 4 becomes value 3, and so on.
The DIGEST verb takes a field, variable or literal value and applies a hash function using one of the algorithms listed below. The resulting value is sometimes called a checksum, or message digest and is often used to ensure integrity of data while in transit or being stored (this could be caused by corruption over a network, or by malicious purposes).

Because the output of a digest can result in binary data, EDGE also performs an ASCII encoding of the hash before storing the result. Users have the choice of either Base64 or hexadecimal encoding.

EDGE supports the following message digest algorithms:

- MD2, MD4, and MD5
- RIPEMD-160
- SHA (or SHA-0) and SHA-1

Message Digest (MD) algorithms, particularly MD5, are a popular hash function used in cryptography and for verifying that large files have been downloaded over the Internet successfully.

RACE Integrity Primitives Evaluation Message Digest (RIPEMD) is a lesser used algorithm with roots in MD4.
Secure Hash Algorithm (SHA) was designed by the U.S. National Security Agency (NSA) and has been adopted by the U.S. Government as a standard. SHA-1, in particular, is a very popular algorithm used in cryptography.

**Algorithm**

The name of the digest algorithm to use. May be one of the following:

- MD2
- MD4
- MD5
- RMD160
- RIPEMD-160
- SHA
- SHA-1
**Example**

```
ALWAYS DIGEST using algorithm "MD5" digest text "Hello World!" encoding as hex store result in @MD5 store status in @Status

IF COMPARE if the value @MD5 is = the value "ed076287532e86365e841e92bfc50d8c"

THEN MESSAGE title $PROJECT text "MD5 checksum is correct" buttons 2 default
```
button 1 icon 1 store result in @NULL

The output of the DIGEST verb using the MD5 algorithm can be verified in a shell using the md5sum package as follows:

$ echo -n 'Hello World!' | md5sum

ed076287532e86365e841e92bfc50d8c -
The DISPLAY-CROSS-REFERENCE verb displays information for record IDs that have been extracted from the cross-reference file.

Properties:

- **Cross-reference ID list**: The name of the list that contains the record IDs based on the cross-reference; can be a field or variable containing the name of the list. The items in the list must be delimited by value marks.

  To create the list, you can use the GET-KEY verb with the Get List command and the Key Type specified as Cross Reference. You can also use a list produced by the AND-LISTS verb.
Notes

This verb searches against regular fields specified in the cross-reference file. The information that is displayed depends on the display fields you have specified for the cross-reference file.

If the list specified for the verb is empty, the cross-reference prompt is displayed and the operator can enter a value for the search.

Example

The following logic flow gets a list of record IDs based on the value T-NAME, which is a value entered by the operator. After the cross-reference records are displayed, the operator’s selection is stored in @REC.ID. Then the record is read using @REC.ID, which allows the record information to be displayed.

```
ALWAYS GET-KEY command get list type cross reference from file CUST.LIST key value NAME store result in @LIST set flag in @ERROR
```
ALWAYS DISPLAY-CROSS-REFERENCE ID list @LIST for data file CUST.LIST store result in @REC.ID

ALWAYS READ-RECORD from file CUST.LIST with ID @REC.ID store status in @FLAG

See Also

For more information on cross-reference search, see the EDGE 7.11 Client Reference Manual.
The DISPLAY-VALUES verb displays the values of a field on a screen.

**Properties:**
- Display the values in

  The values to be displayed; can be a number, a literal, a field, a multivalued field or variable, or one position in a multivalued field or variable.

  If the field or variable is multivalued, the value for each position is displayed on a separate line. The value markers are not displayed.

  If you specify a value position for the multivalued field or variable, only that value is displayed. For example, if you enter PROD.COMM,3, the value of the third position in the PROD.COMM field is displayed.
Notes

When the verb is run, whether in operations or in a task, the current screen is erased and the field is displayed on an area of 80 characters across and 23 lines down.

In operations, if DISPLAY-VALUES is performed from a function key, you should specify the next screen to route to after the display so that the screen is repainted.

Example

At the bottom of the screen, the command line displays the number of characters and number of lines in the display. You can also use commands to scroll when there are more than 23 lines. For a description of the commands, see Table 16-18.
**Table 16-18.** DISPLAY-VALUES Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bottom or B</td>
<td>Displays the last 23 lines.</td>
</tr>
<tr>
<td>Down or D</td>
<td>Displays the next screen down.</td>
</tr>
<tr>
<td>Quit or Q</td>
<td>Ends the display and returns to the logic flow or task.</td>
</tr>
<tr>
<td>Top or T</td>
<td>Displays the first 23 lines.</td>
</tr>
<tr>
<td>Up or U</td>
<td>Displays the next screen up.</td>
</tr>
</tbody>
</table>
The DIVIDE verb divides one number by another number and stores the result in the same or another field.

**Properties:**

- **Divide the value**: The number to be divided (dividend); can be a number, or a field or variable containing a number.
- **By the value**: The number to divide by (divisor); can be a number, or a field or variable containing a number.
- **Store result in**: The location in which to store the result of the division; can be a field or variable.

If 1 is divided by 0, the result is given as 0.
Notes

The number of digits displayed after the decimal point depends on the format specified for the field in which the result is stored. For example, if the field is formatted with two decimal places, the result will be rounded to two decimal places.

If retained in a variable, the result of a division which is not a whole number can contain up to four decimal places.

See Also

The CALCULATE verb.
The EDGE verb allows advanced EDGE developers to build systems that can be used to administer EDGE installations and guides. This is an optional feature used with AdvantEDGE. For details, see the AdvantEDGE Developers Manual.
The EMAIL verb is an optional verb that requires additional licensing. The module can be enabled during the installation or upgrade of EDGE, or anytime thereafter, and must be enabled before any setup can be done. For more information about email, see the Email Adapter documentation, which is available separately:

- Email Adapter Installation Guide
- Email Adapter Implementation and Planning Guide
- Email Adapter Programmers Guide
- Email Adapter Quick-Start Guide
- Email Adapter Release Notes
The ENCRYPT verb is an optional verb that requires additional licensing. The module can be enabled anytime after the installation or upgrade of EDGE.

ENCRYPT performs encryption on a field, variable, or string literal (known as “plaintext” or “cleartext” data) using one of the supported ciphers (listed below) and stores the result (known as “ciphertext”) into a field or variable. To an outsider who does not know the correct passphrase, the ciphertext data looks like unintelligible random data.

Because the output of an encryption can result in binary data, EDGE also performs an ASCII encoding of the ciphertext before storing the result. Users have the choice of using either Base64 or hexadecimal encoding. EDGE supports the following symmetric ciphers:

- DES, DES-X, Triple-DES
- AES (128, 193, or 256 bit)
- Blowfish
- CAST-128
- IDEA
- RC2 and RC5
Data Encryption Standard (DES) encryption uses a 56 bit key to encrypt data communications. This method has been around for a long time and it is considered insecure by today’s standards.

DES-X improves on DES but is not a commonly used cipher.

Triple DES encryption uses the same technique as DES except that it performs the cipher algorithm three times, thus providing 168 bit encryption. This is a very popular encryption algorithm, however it is slower when compared to Blowfish and AES.

Advanced Encryption Standard (AES) - also known as Rijndel – has been adopted by the U.S. Government as an encryption standard and operates with a 128 bit 192 bit, or 256 bit key.

Blowfish is a fast, free encryption cipher using 128 bit key length.

CAST-128 (or CAST-5) has been approved for Canadian Government use by the Communications Security Establishment.

International Data Encryption Algorithm (IDEA) was developed in Switzerland for use in older versions of Pretty Good Privacy (PGP).

RC2 and RC5 stand for the Rivest Cipher (of Rivest-Shamir-Adleman, or RSA).
The image shows a Verb Properties window with the following settings:

- **Cipher**: "DES3"
- **Plain text**: "Hello World!"
- **Passphrase**: "mysecret"
- **Encoding**: Base64

The window also has options to store the result and status:

- **Store result in**: @CipherText
- **Store status in**: @Status

The window is labeled "ENCRIPT."
<table>
<thead>
<tr>
<th>Cipher</th>
<th>The name of the cipher to use. May be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES128</td>
<td>128-bit AES</td>
</tr>
<tr>
<td>AES192</td>
<td>192-bit AES</td>
</tr>
<tr>
<td>AES256</td>
<td>256-bit AES</td>
</tr>
<tr>
<td>BF</td>
<td>Blowfish</td>
</tr>
<tr>
<td>DES</td>
<td>56-bit Data Encryption Standard</td>
</tr>
<tr>
<td>DES-X</td>
<td>Extended Data Encryption Standard</td>
</tr>
<tr>
<td>DES3</td>
<td>Triple-DES</td>
</tr>
<tr>
<td>CAST</td>
<td>CASE-128</td>
</tr>
<tr>
<td>IDEA</td>
<td></td>
</tr>
<tr>
<td>RC2</td>
<td></td>
</tr>
<tr>
<td>RC5</td>
<td></td>
</tr>
<tr>
<td>Plain text</td>
<td>A field, variable, or string literal containing the text to</td>
</tr>
<tr>
<td></td>
<td>passed through the cipher.</td>
</tr>
<tr>
<td>Passphrase</td>
<td>A field, variable, or string literal containing the “key”</td>
</tr>
<tr>
<td></td>
<td>used to protect the data.</td>
</tr>
<tr>
<td>Encoding</td>
<td>The type of binary-to-text encoding to be used on the</td>
</tr>
<tr>
<td></td>
<td>output of the cipher. May be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Base-64</td>
</tr>
<tr>
<td></td>
<td>Hex</td>
</tr>
<tr>
<td>Store result in</td>
<td>A field or variable which will contain the resulting</td>
</tr>
<tr>
<td></td>
<td>ciphertext.</td>
</tr>
</tbody>
</table>
Security Note

Any encrypted data is only as secure as the availability of the pass-phrases used against that data and additional steps should be taken to ensure that the passphrase is not divulged to an unauthorized individual. For example, if the passphrase is to be hard-coded into the application, only authorized individuals should have access to the EDGE source code.

Ideal, passphrases should not be stored on the EDGE Server at all and are entered by the user only when specific access is needed. If method is used, then EDGE Client Encryption is also recommended to prevent passphrases being transmitted in cleartext over the network.

Example

ALWAYS COPY the value “Hello World!” into @PlainText
ALWAYS ENCRYPT using cipher "DES3" plain text @PlainText with passphrase "mysecret" encoding as base64 store result in @CipherText store status in @Status
IF COMPARE if the value @CipherText is = the value
The output of the ENCRYPT verb can be verified in a shell with the OpenSSL package as follows:

```
$ echo -n 'Hello World!' | openssl enc -des3 -a -nosalt -pass
pass:mysecret 9Y0TIyftRt++n1Mmf8Q==
```
The EXECUTE verb executes a specified command, either a user-defined or UNIX command on the server, or a Windows command on the client. Some of the properties vary depending on whether the verb is executed on the server or client.
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute command</td>
<td>The command to be executed; can be a literal, or a field or variable containing the command. The command can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>User-defined command.</td>
</tr>
<tr>
<td></td>
<td>UNIX command.</td>
</tr>
<tr>
<td></td>
<td>Windows command.</td>
</tr>
<tr>
<td></td>
<td>DOS command.</td>
</tr>
<tr>
<td>On</td>
<td>Specifies where to execute the command; can be one of the following:</td>
</tr>
<tr>
<td>Client</td>
<td>The command is interpreted as a Windows or DOS application to be executed on the EDGE client. When executed, the application is launched on the PC and the next command in the logic flow is executed. EDGE does not wait for the application to complete.</td>
</tr>
<tr>
<td></td>
<td>This parameter should be checked only for guides that are executed through EDGE Client. If it is checked and the guide is executed through ASCII, the EXECUTE command is ignored.</td>
</tr>
<tr>
<td>Server</td>
<td>The command is interpreted as a UNIX command and is executed on the server. The task waits until the completion of the command.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

**Clear screen**
Specifies whether the screen is cleared before the command is executed. This is only available when the command is executed from the server; can be one of the following:

- **Yes**
  The screen is cleared before the command is executed. The original screen is saved and when the command is complete, the screen is repainted.

- **No**
  The screen is not cleared before the command is executed and is not repainted after.

**Prompt when done**
Specifies whether the user is prompted when the command is done. This is only available when the command is executed from the server; can be one of the following:

- **Yes**
  The operator is prompted when the command is done.

- **No**
  There is no prompt. If the logic is to be performed from the task scheduler, this value should not be used.

**Store status in**
The location in which to store the standard output of the command being executed; can be stored in a field or variable or left blank to display on the screen or redirect output. For information about redirecting stdout and stderr, see Redirecting Output on the Server.

This property is only available when the command is executed from the server.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimize GEO</td>
<td>Specifies whether to minimize the GEO window while the application is active. This is only available if the command is executed from the client. One of the following can be specified:</td>
</tr>
<tr>
<td>Yes</td>
<td>The GEO window is minimized and the specified Windows application is made active in the foreground.</td>
</tr>
<tr>
<td>No</td>
<td>The GEO window is not minimized.</td>
</tr>
<tr>
<td>Create new instance</td>
<td>Specifies whether EDGE Client is to always launch a new instance of the application. One of the following can be specified:</td>
</tr>
<tr>
<td>Yes</td>
<td>EDGE Client always starts a new instance of the application. If the application does not allow multiple instances, a Windows error is displayed.</td>
</tr>
<tr>
<td>No</td>
<td>EDGE Client only launches the application if necessary. If the application is already running, EDGE Client activates it. If there is more than one instance, EDGE Client activates the next one in the Windows task list (maintained by Windows).</td>
</tr>
</tbody>
</table>
Redirecting Output on the Server

The redirection symbol, that is, the greater-than symbol (>), can be used to redirect the stdout and stderr. For example, EXECUTE "date >/tmp/out" sends the output of the command to /tmp/out. The redirection command must be enclosed by quotation marks and follow the end of the command and all arguments. If redirecting both stdout and stderr, stdout must be redirected first.

- >/tmp/out Redirects stdout to /tmp/out.
- 2>/tmp/err Redirects stderr to /tmp/err.
- >/tmp/file 2>&1 Redirects both stdout and stderr to /tmp/file.

Commands on the Client

When the command is to be executed on the client, you can specify an explicit path to the application or allow Windows to search for the application based on the PATH environment variable. For example, the command, "C:\APPL\NOTEPAD.EXE MYDOC.TXT" contains an explicit path to an application and document. The command "NOTEPAD.EXE" contains an implicit path to the application.

To find the application command, displayed as the Command Line, do either of the following:
EXECUTE

- Select the application icon and select Properties from the File menu.
- Select the application icon and press ALT+ENTER.

To find the path command name, do either of the following:

- Select the directory icon and select Properties from the File menu.
- Select the directory icon and press ALT+ENTER.

If an error is encountered while executing the application on the client, an error code is displayed in a dialog window.

**See Also**

For a list of ACL commands that can be run from the EXECUTE verb, see Chapter 10. For UNIX system commands, see your UNIX documentation. For information on Windows commands, see Windows documentation.
EXECUTE-ADHOC

The EXECUTE-ADHOC verb allows you to execute a single ad hoc report or batch of ad hoc reports from a guide or a task.

Properties:

Execute from

Specifies whether a single or batch of ad hoc reports is to be executed; can be one of the following:
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ad hoc</td>
<td>To execute one report.</td>
</tr>
<tr>
<td>Batch</td>
<td>To execute a batch of reports.</td>
</tr>
<tr>
<td>Project ID</td>
<td>The location of the reports; can be a literal, a field or variable containing the project or account name, or blank. In a logic flow, leaving this blank defaults to the current project. If used in a task, defaults to the current project or EDGE, depending on where the task is executed. For system reports, the project ID is &quot;EDGE&quot;.</td>
</tr>
<tr>
<td>Name of adhoc</td>
<td>The name of the ad hoc report or the batch name; can be a literal, or a field or variable containing the name. (For information about creating batch reports, see Chapter 7.) If a batch name is entered, the output is sent to the printer, regardless of the setting of the Send to printer property.</td>
</tr>
<tr>
<td>Store status in</td>
<td>The location in which to store the result code; can be a field or variable. The result codes are: 0 Successful. 1 Invalid project name. 2 Invalid ad hoc name.</td>
</tr>
<tr>
<td>Send to printer</td>
<td>Specifies whether to send the reports to a printer; can be one of the following:</td>
</tr>
</tbody>
</table>

---

EDGE 7.11 Developer Reference Manual
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Yes</th>
<th>No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sends the report to the printer. If the task is executed from the task scheduler, choose this value for this property.</td>
<td>Displays on the terminal.</td>
<td></td>
</tr>
<tr>
<td>Yes</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>No</td>
<td></td>
<td>Displays on the terminal.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Printer device</th>
<th>The name of the printer to be used; can be a literal, a field or variable containing the name, or blank to use the default printer specified in the ad hoc report. This property is available only when the setting of the Send to printer property is Yes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run in background</td>
<td>Specifies whether to run the report in the background; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Runs the report in the background. If set to this value, the Prompt when done property is not available.</td>
</tr>
<tr>
<td></td>
<td>If the task is executed from the task scheduler, choose this value for this property.</td>
</tr>
<tr>
<td></td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>The user is not prompted.</td>
</tr>
<tr>
<td>Prompt when done</td>
<td>Specifies whether to prompt the user that the report is done; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>The user is prompted when the report is done.</td>
</tr>
<tr>
<td></td>
<td>No</td>
</tr>
</tbody>
</table>
This property is not available if the report is run in the background.
EXECUTE-STORED-PROCEDURE

The EXECUTE-STORED-PROCEDURE verb allows stored procedures to be executed from EDGE. The stored procedure can be one that returns or does not return data.

For more information, see the DataBase Link documentation for the relational database you are using.
The EXECUTE-TASK verb executes a task from a guide or from another task.

Properties:

Project ID

The location of the task; can be a literal, a field or variable containing the project or account name, or blank.

In a logic flow, leaving this blank defaults to the current project. If used in a task, defaults to the current project or EDGE, depending on where the task is executed.

For system tasks, the project ID is "EDGE".
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of task</td>
<td>The name of the task; can be a literal, or a field or variable containing the name.</td>
</tr>
<tr>
<td>Store status in</td>
<td>The location in which to store the result code; can be a field or variable. The result codes are:</td>
</tr>
<tr>
<td></td>
<td>0 Successful.</td>
</tr>
<tr>
<td></td>
<td>1 Invalid project name.</td>
</tr>
<tr>
<td></td>
<td>2 Invalid task name.</td>
</tr>
<tr>
<td>Run in background</td>
<td>Specifies whether to run the task in the background; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes Runs in the background. The Prompt when done property is not available.</td>
</tr>
<tr>
<td></td>
<td>No Runs in the foreground.</td>
</tr>
<tr>
<td>Prompt when done</td>
<td>Specifies whether to prompt the user that the task is done; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes Prompts the user when the task is done.</td>
</tr>
<tr>
<td></td>
<td>No Does not prompt the user.</td>
</tr>
<tr>
<td></td>
<td>This property not available when the task is run in the background.</td>
</tr>
</tbody>
</table>
The EXIT verb quits the logic flow or task.

**Properties:**

None.

**Notes**

When the EXIT statement is reached, any statements following the exit statement are ignored. Frequently, an EXIT statement is used after a conditional IF statement. For example, you might exit if a value is null, or if the record count has reached a specified number.

**Note:** *It is not necessary to use the EXIT verb as the last line in a logic flow or task. The logic is automatically exited if there are no further statements.*
The FIND-VALUE verb locates the value position of a specified value in a multivalued field or variable.

**Properties:**

- **Find the value**: The value to be located; can be a number, a literal, or a field or variable containing the value to be located.
- **In the multi-values**: The location to be searched; can be a multivalued field or variable.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store result in</td>
<td>The location in which to store the number of the value position which contains the specified value; can be a field or variable.</td>
</tr>
<tr>
<td></td>
<td>When the value occurs in more than one value position, only the first occurrence of the value is returned. If the value is not found, 0 (zero) is returned.</td>
</tr>
<tr>
<td>Case sensitive match</td>
<td>Specifies whether the search is to distinguish between uppercase and lowercase letters; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes The search distinguishes between uppercase and lowercase letters. For example, the value &quot;Abc,&quot; is not considered the same as &quot;ABC&quot; or &quot;abc.&quot;</td>
</tr>
<tr>
<td></td>
<td>No or blank The search does not distinguish between uppercase and lowercase letters. For example, the values &quot;ABC&quot; and &quot;abc&quot; are considered the same.</td>
</tr>
<tr>
<td>Ignore blanks</td>
<td>Specifies whether the search is to ignore trailing and leading blanks when determining whether the value matches; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes Leading and trailing blanks for each position are ignored when the search is done.</td>
</tr>
</tbody>
</table>
No or blank

Leading or trailing blanks are considered part of the value when the search is done.

For example, the value “Smith” contains leading blanks and the value “Smith” contains trailing blanks. If you search for “Smith” and select Yes for this prompt, the values “Smith” and “Smith” is found. If you select No or leave the property blank, neither value is found.
The FLAT-FILE-ACCESS verb allows you to access a flat file for reading or writing on the server or the client machine.

Properties:

Mode
The type of file to access; can be one of the following:
- Ascii.
- Binary.
- Default.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command</strong></td>
<td>The type of action to take; can be one of the following:</td>
</tr>
<tr>
<td>Read</td>
<td>Reads a flat file. The entire file is always returned; there is no way to specify a position in the file. Binary Mode only supports the read command and cannot be used in tasks.</td>
</tr>
<tr>
<td>Write</td>
<td>Writes a flat file. If an existing file is specified, it is overwritten.</td>
</tr>
<tr>
<td><strong>From/To</strong></td>
<td>The location of the file to be read or written; can be one of the following:</td>
</tr>
<tr>
<td>Client</td>
<td>Looks for the file on the EDGE client. (This option is not available in a task.)</td>
</tr>
<tr>
<td>Server</td>
<td>Looks for the file on the EDGE server.</td>
</tr>
<tr>
<td><strong>File path</strong></td>
<td>The pathname of the file; can be a literal, or a field or variable containing the UNIX or DOS file path.</td>
</tr>
<tr>
<td><strong>Read/write buffer</strong></td>
<td>The name of the location in which to store the data that was read or that contains the data to be written; can be a field or variable. This property is only enabled when the <strong>Mode</strong> is set to Ascii or Default.</td>
</tr>
</tbody>
</table>
Notes

If the full pathname is not given, the relative path is appended to the current working directory. The current working directory on the server is the project directory. On the client, the working directory is defined within the EDGE Client environment.

The amount of activity on your server may cause the FLAT-FIILE-ACCESS verb to experience problems during runtime. To prevent this, you can increase the value of the edgeo.ini option blockedRetry-Limit.
When transferring data from a client to a server, character filtering is performed as follows:

<table>
<thead>
<tr>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASCII</td>
<td>Each CTRL-M (carriage return) character is removed. The CTRL-Z (end of file marker) character is changed to a CTRL-D.</td>
</tr>
<tr>
<td>Binary</td>
<td>No character filtering occurs.</td>
</tr>
<tr>
<td>Default</td>
<td>No character filtering occurs.</td>
</tr>
</tbody>
</table>

If a file transfer takes a long time, the EDGE client may time out. To avoid this, increase the Heartbeat setting in the edgeo.ini file.

By default, files larger than 64K are truncated. To avoid this, increase the maxFileSize setting in the edgeo.ini file.

Whenever the Write command of the FLAT-FILE-ACCESS verb is invoked, the specified file is written, even if the command was invoked in test mode.

### Table 16-19. FLAT-FILE-ACCESS Error Messages

<table>
<thead>
<tr>
<th>Found On</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEO and server</td>
<td>Permission denied for File <em>filename</em> on client/server.</td>
</tr>
<tr>
<td>GEO and server</td>
<td>File <em>filename</em> does not exist on client/server.</td>
</tr>
<tr>
<td>Server only</td>
<td>Failed to open file <em>filename</em> on client/server.</td>
</tr>
<tr>
<td>Found On</td>
<td>Message</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>Server only</td>
<td>Failed to read file <em>filename</em> on client/server.</td>
</tr>
<tr>
<td>Server only</td>
<td>Failed to write file <em>filename</em> on client/server.</td>
</tr>
</tbody>
</table>
The FORMAT-INPUT verb formats an external formatted date, time, or numeric value to an internal format.

**Properties:**

- **Format the value**: The value to be formatted; can be a literal, or a field or variable containing the value to be formatted.
- **Using format code**: The format code; can be a literal, or a field or variable containing the format code. For a listing of codes, see Appendix AA.
- **Store result in**: The value in internal format; can be stored in a field or variable.
Properties: (continued)

Date and time values are stored in internal format, even if the resulting value is stored to an alphanumeric field.

Numbers are stored without any non-numeric characters and with the decimal character based on the U.S. standard; that is, a period.

See Also

Other verbs that allow you to format data include: CHARACTER-CONVERSION, DATE-FORMAT, MONEY-CONVERSION, PICTURE-FORMAT, and TIME-FORMAT.
The FORMAT-OUTPUT verb formats an internal formatted date, time, or numeric value to an external format value which can be printed or displayed.

**Properties:**

- **Format the value**: The value to be formatted; can be a literal, or a field or variable containing the value. The value must be in internal format. (Values stored in date, time, and numeric type fields are automatically stored in internal format.)

- **Using format code**: The format code; can be a literal, or a field or variable containing the. For a listing of codes, see Appendix AA.
**Properties: (continued)**

Store result in The formatted value; can be stored in a field or variable. To retain the formatting, the value must be stored to an alphanumeric field, rather than a date, time, or numeric field.

**Notes**

The FORMAT-OUTPUT verb does not affect the way the value is stored, only the format of the value when printed or displayed. To store data in a different format, use the FORMAT-INPUT verb.

**See Also**

Other verbs that allow you to format data include: CHARACTER-CONVERSION, DATE-FORMAT, MONEY-CONVERSION, PICTURE-FORMAT, and TIME-FORMAT.
The `FORMAT&CONCAT` verb builds a formatted, multivalued string.

**Properties:**

- **Format multi-valued data**: The data to be formatted; can be a literal, or a field or variable containing the data. The field or variable can be multivalued.

- **Using format code**: The format codes; can be a literal, or a field or variable containing the format code(s). The field or variable can be multivalued. The format code in each value is applied to the corresponding value in the data. If there are more values in the data than there are formatting codes, the extra values are not formatted.
**Properties: (continued)**

- **The resulting formatted string is single-valued, regardless of the number of values in the source.**
- **If the format code is empty (""), value marks, if any, are removed and the data is copied as is to the result.**

**Insertion type**  
Specifies how the formatted string is to be copied to the result; can be one of the following:

- **Append to**  
  Appends as a new value to the end of the result.
- **Insert into**  
  Inserts as a new value at the beginning of the result.
- **Replace**  
  Replaces the previous values in the result.  
  The result is single-valued.

**Store result in**  
The location in which to store the formatted string; can be a field or variable.

**Notes**

Currently, the only format codes that are supported are those listed in the Numeric Codes table in Appendix AA.
Example

The following logic flow retrieves data from a file, formats it, and stores it in a field in the TEMP file. The logic flow can then be attached to a screen at an appropriate point, for example, at the beginning of a call; this populates the RESULT field in the TEMP file with the current contents of the CUST.LIST file. This field is then specified as the content field for a list box.
ALWAYS GET-KEY command get list type primary key from file CUST.LIST store result in @file.list set flag in @result
ALWAYS COPY the value "ML#15" into @FMT,1
ALWAYS COPY the value "ML#15" into @FMT,2
ALWAYS LOOP command for each value of multi-valued field @file.list store result in @ID
ALWAYS READ-RECORD from file CUST.LIST with ID @ID store status in @OK
ALWAYS COPY the value C-FIRST.NAME into @DATA,1
ALWAYS COPY the value C-LAST.NAME into @DATA,2
ALWAYS FORMAT&CONCAT multi-valued field @DATA using format code @FMT insertion type append store result in T-RESULT
ALWAYS LOOP command end

--------
ALWAYS GET KEY command: Get List key type: Primary Key from file: CUST.LIST return in: @file.list setting: @result to Y or N
ALWAYS COPY "ML#15" into @FMT,1
ALWAYS COPY "ML#15" into @FMT,2
ALWAYS LOOP FOR EACH Value in Multi-Valued Field @file.list store result in @ID
ALWAYS READ RECORD from CUST.LIST with ID of @ID and setting @OK to Y or N
ALWAYS COPY C-FIRST.NAME into @DATA,1
ALWAYS COPY C-LAST.NAME into @DATA,2
ALWAYS FORMAT&CONCAT using @DATA with format codes in @FMT appending value to T-RESULT
ALWAYS LOOP END
The GET-CALL-RECORD verb retrieves an existing call record.

Properties:

Call ID: The call ID; can be a number, a literal, or a field or a variable containing the call ID.

Notes

If the GET-CALL-RECORD finds the call ID in the STATUS file, the following are performed:

- The associated records in the HISTORY and RESPONSES files are also read.
- At the end of the call, these files are automatically updated.
There are numerous reasons why a call ID may not be found, including:

- The call ID doesn't exist.
- The call ID is in a protected queue or a queue the operator does not have access to.
- The call ID is locked.

If the call ID is not found, the SYS field $CALLID will be equal to null. If the call ID is null, you can assign an ID using the ASSIGN-NEW-CALL-ID verb.

**See Also**

ASSIGN-NEW-CALL-ID verb.

CALL-EVENT verb.
GET-CROSS-REFERENCE-DATA

The GET-CROSS-REFERENCE-DATA verb retrieves a list of formatted information based on the display fields identified for the cross-reference file.

Properties:

- **Cross reference ID list**: The list of record IDs separated by value marks; can be a multivalued field or variable containing the list.
- **For data file**: The name of the file on which the cross-reference file was built; must be the filename.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store result in</td>
<td>The location in which to store the list of formatted cross-reference display information; can be a multivalued field or variable.</td>
</tr>
<tr>
<td></td>
<td>The display fields that the verb retrieves are the cross-reference fields specified when the cross-reference file is defined.</td>
</tr>
<tr>
<td></td>
<td>The display information for each record is preceded by a sequential number beginning at 1 or at the number specified under the Number property. Each record is separated from the next record by a value mark.</td>
</tr>
<tr>
<td>Store header in</td>
<td>The location in which to store the report header, which contains the name of each display field in the list; can be stored in a field or variable.</td>
</tr>
<tr>
<td></td>
<td>If the specified data file does not have a cross-reference file, the result and header will contain null.</td>
</tr>
<tr>
<td>Starting with number</td>
<td>The number at which to begin sequentially numbering the items in the list; can be blank, a number, or a field or variable containing the number. If left blank, the default of 1 is used.</td>
</tr>
</tbody>
</table>

### Notes

This verb enables you to perform custom cross-referencing in a guide. Unlike the built-in cross-reference search that can be invoked
in operations, this verb gets information, but does not display it. Record selection and display must be controlled entirely through your logic flows.

The following figure displays a listing of cross-reference data saved to a multivalued field.

<table>
<thead>
<tr>
<th>Position</th>
<th>Record ID</th>
<th>Cross-reference fields</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>SUNFLOWER</td>
<td>MARY</td>
</tr>
<tr>
<td>2.2</td>
<td>AGAPANTHUS</td>
<td>GEORGE</td>
</tr>
<tr>
<td>3.3</td>
<td>ROSE LANE</td>
<td>MARTIN</td>
</tr>
</tbody>
</table>

**See Also**

For information on creating a list of record IDs, see the AND-LISTS and GET-KEY verbs.
The GET-KEY verb finds one key or a list of several keys in a file.

**GET-KEY Commands**

- **First Key**  
  Gets the key of the first record as based on the primary key, a secondary key, or a cross-reference value.

- **Get List**  
  Gets the list of keys based on specified criteria.

- **Next Greater**  
  Gets the key whose value is greater than the key value.
Next Key  Gets the key of the record following the key just accessed.
Next Smaller  Gets the key whose value is smaller than the key value.
Previous Key  Gets the key of the record previous to the key just accessed.

Notes

All primary keys are sorted in alphanumeric order. In this order, data is always sorted from left to right. For example, 8 would follow 79, because 8 is greater than 79 in alphanumeric sort order.

The GET-KEY commands work sequentially on the file. This means the result from the Next and Previous commands depend on the current reference or specified value. Since the access is sequential, the next key moves down the file and the previous key moves up the file, assuming the beginning of the file is the top.

When the Next command is used at the end of the file, a null result is returned since there are no more records. Similarly, when the Previous command is used at the top of the file, a null result is returned.

The environment variable, MAX_XREF_MATCH_KEYS, can be set to limit the number of matches returned by the Get List with Cross Ref-
GET-KEY

erence command. For more information about setting an EDGE en-

Table 16-20 summarizes the commands and key types available with

Table 16-20.GET-KEY Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Primary Key</th>
<th>Secondary Key</th>
<th>Cross-reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Key</td>
<td>Retrieves the first record ID in the specified file.</td>
<td>Retrieves the first record ID whose secondary key value is equal to the specified secondary key value.</td>
<td>Retrieves the first record ID whose cross-reference value is equal to the specified cross-reference value.</td>
</tr>
<tr>
<td>Next Key</td>
<td>Retrieves the next record ID in the specified file.</td>
<td>Retrieves the next record ID whose secondary key value is equal to the specified secondary key value.</td>
<td>Retrieves the next record ID whose cross-reference value is equal to the specified cross-reference value.</td>
</tr>
</tbody>
</table>

Assumes an ID is active from using First Key.
Assumes an ID is active from using First Key.
Assumes an ID is active from using First Key.
Assumes an ID is active from using First Key.
<table>
<thead>
<tr>
<th>Command</th>
<th>Primary Key</th>
<th>Secondary Key</th>
<th>Cross-reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Previous Key</td>
<td>Retrieves the previous record ID in the specified file.</td>
<td>Retrieves the previous record ID whose secondary key value is equal to the specified secondary key value.</td>
<td>Retrieves the previous record ID whose cross-reference value is equal to the specified cross-reference value.</td>
</tr>
<tr>
<td>Next Greater Key</td>
<td>Retrieves the record ID in the specified file that is greater than the key value.</td>
<td>Retrieves the record ID which has a secondary key value greater than the specified secondary key value.</td>
<td>Retrieves the record ID when the cross-reference value is greater than the specified cross-reference value.</td>
</tr>
</tbody>
</table>
### Table 16-20. GET-KEY Commands (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Primary Key</th>
<th>Secondary Key</th>
<th>Cross-reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Smaller Key</td>
<td>Retrieves the record ID in the specified file that is smaller than the key value.</td>
<td>Retrieves the record ID which has a secondary key value less than the specified secondary key value.</td>
<td>Retrieves the record ID when the cross-reference value is less than the specified cross-reference value.</td>
</tr>
<tr>
<td>Get List</td>
<td>Retrieves a list of all the IDs in the specified file.</td>
<td>Retrieves a list of all IDs in the specified file where the secondary key matches the specified key value.</td>
<td>Retrieves a list of all IDs in the specified file where the cross-reference value matches the specified key value.</td>
</tr>
</tbody>
</table>
**GET-KEY Verb Example**

Table 16-21 represents a sample file for the record ID, secondary key named STATE, and cross-reference field named REGION.

**Table 16-21. GET-KEY Sample File**

<table>
<thead>
<tr>
<th>Record ID (ACCT.NO)</th>
<th>Secondary Key (STATE)</th>
<th>Cross-reference (REGION)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NY</td>
<td>EAST</td>
</tr>
<tr>
<td>10</td>
<td>AZ</td>
<td>WEST</td>
</tr>
<tr>
<td>25</td>
<td>CA</td>
<td>WEST</td>
</tr>
<tr>
<td>30</td>
<td>NY</td>
<td>EAST</td>
</tr>
<tr>
<td>400</td>
<td>CA</td>
<td>WEST</td>
</tr>
<tr>
<td>45</td>
<td>AZ</td>
<td>WEST</td>
</tr>
<tr>
<td>7</td>
<td>GA</td>
<td>SOUTH</td>
</tr>
</tbody>
</table>
Table 16-22 depicts the results of GET-KEY commands. The results are based on the secondary key value of CA and the cross-referenced value of WEST.

**Table 16-22. Results from GET-KEY Search**

<table>
<thead>
<tr>
<th>Command</th>
<th>Primary Key</th>
<th>Secondary Key</th>
<th>Cross-reference</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Key Value = 30</td>
<td>Key Value = CA</td>
<td>Key Value = WEST</td>
</tr>
<tr>
<td>First Key</td>
<td>1</td>
<td>25</td>
<td>10</td>
</tr>
<tr>
<td>Next Key</td>
<td>10</td>
<td>400 (found another CA)</td>
<td>25</td>
</tr>
<tr>
<td>Used after the first key.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Key</td>
<td>25</td>
<td>null</td>
<td>400</td>
</tr>
<tr>
<td>Used a second time.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Previous Key</td>
<td>10</td>
<td>400</td>
<td>25</td>
</tr>
<tr>
<td>Used after the second Next Key.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Next Greater Key</td>
<td>400</td>
<td>7 (found GA)</td>
<td>null (nothing greater than West)</td>
</tr>
<tr>
<td>Next Smaller Key</td>
<td>25</td>
<td>10 or 45 (found AZ)</td>
<td>7 (found SOUTH)</td>
</tr>
<tr>
<td>Get List</td>
<td>1, 10, 25, 30, 400, 45, 7</td>
<td>25, 400</td>
<td>10, 25, 400, 45</td>
</tr>
</tbody>
</table>
**First Key, Next Key, Previous Key Commands**

**With Primary Key**

The First Key, Next Key, and Previous Key commands combined with the Primary Key type retrieve a record ID based on the primary key of the file.

**Properties:**

- **Get key command**
  - The command selected; can be one of the following:
    - **First Key**
      - Retrieves the first record ID in the file.
    - **Next Key**
      - Retrieves the next record ID based on the current position in the file.
    - **Previous Key**
      - Retrieves the previous record ID based on the current position in the file.

- **Key type**
  - Specifies the key type. When Primary Key is selected, the search is based on the record ID.

- **From file**
  - The file name to be searched; must be the filename.

- **Store result in**
  - The location in which to store the value of the record ID retrieved; can be a field or variable.

- **Set flag in**
  - The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.
**With Secondary Key**

The First Key, Next Key, and Previous Key commands combined with the Secondary Key type retrieves a record ID based on a secondary key value.

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get key command</td>
<td>The command selected; can be one of the following:</td>
</tr>
<tr>
<td>First Key</td>
<td>Retrieves the first record ID whose secondary key value is equal to the specified secondary key value.</td>
</tr>
<tr>
<td>Next Key</td>
<td>Retrieves the next record ID whose secondary key value is equal to the specified secondary key value.</td>
</tr>
<tr>
<td>Previous Key</td>
<td>Retrieves the previous record ID whose secondary key value is equal to the specified secondary key value.</td>
</tr>
<tr>
<td>Key type</td>
<td>Specifies the key type. When Secondary Key is selected, the search is based on the value in the secondary key.</td>
</tr>
<tr>
<td>From file</td>
<td>The file name to be searched; must be the filename.</td>
</tr>
<tr>
<td>Key value</td>
<td>The value to be searched for in the secondary key; can be a literal, or a field or variable containing the value.</td>
</tr>
<tr>
<td>Secondary key field</td>
<td>The name of the secondary key; can be a literal, or a field or variable containing the name of the secondary field.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

- **Store result in**
  - The location in which to store the value of the record ID retrieved; can be a field or variable.

- **Set flag in**
  - The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.

**With Cross-Reference**

The First Key, Next Key, and Previous Key commands combined with the Cross-Reference Type retrieves a record ID based on a cross-reference value.

**Properties:**

- **Get key command**
  - The command selected; can be one of the following:
    - **First Key**
      - Retrieves the first record ID whose cross-reference value is equal to the specified cross-reference value.
    - **Next Key**
      - Retrieves the next record ID whose cross-reference value is equal to the specified cross-reference value.
    - **Previous Key**
      - Retrieves the previous record ID whose cross-reference value is equal to the specified cross-reference value.
**Next Greater and Next Smaller Commands**

**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key type</td>
<td>Specifies the key type. When Secondary Key is selected, the search is based on the value in the secondary key.</td>
</tr>
<tr>
<td>From file</td>
<td>The file name to be searched; must be the filename.</td>
</tr>
<tr>
<td>Cross-reference value</td>
<td>The value to be used in the search; can be a literal, or a field or variable containing the value.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the value of the record ID retrieved; can be a field or variable.</td>
</tr>
<tr>
<td>Set flag in</td>
<td>The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>

**With Primary Key**

The Next Greater and Next Smaller commands combined with the Primary Key Type searches for the value specified for the record ID and, if found, retrieves the value of the next greater or next smaller key.

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get key command</td>
<td>The command selected; can be one of the following:</td>
</tr>
<tr>
<td>Next Greater</td>
<td>Retrieves the record ID in the specified file that is greater than the specified key value.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Smaller</td>
<td>Retrieves the record ID in the specified file that is smaller than the specified key value.</td>
</tr>
<tr>
<td>Key type</td>
<td>Specifies the key type. When Primary Key is selected, the search is based on the value in the record ID.</td>
</tr>
<tr>
<td>From file</td>
<td>The file name to be searched; must be the filename.</td>
</tr>
<tr>
<td>Key value</td>
<td>The value to be searched for; can be a number, a literal, or a field or variable containing the value.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the value of the record ID retrieved; can be a field or variable.</td>
</tr>
<tr>
<td>Set flag in</td>
<td>The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>

**With Secondary Key**

The Next Greater and Next Smaller commands combined with the Secondary Key type searches for the value specified for the secondary key and, if found, gets the value of the next greater or next smaller key.

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get key</td>
<td>The command selected; can be one of the following:</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Next Greater</td>
<td>Retrieves the record ID which has a secondary key value greater than the specified secondary key value.</td>
</tr>
<tr>
<td>Next Smaller</td>
<td>Retrieves the record ID which has a secondary key value less than the specified secondary key value.</td>
</tr>
<tr>
<td>Key type</td>
<td>Specifies the key type. When Secondary Key is selected, the search is based on the value in the specified secondary key.</td>
</tr>
<tr>
<td>From file</td>
<td>The file name to be searched; must be the filename.</td>
</tr>
<tr>
<td>Key value</td>
<td>The value to be searched for; can be a number, a literal, or a field or variable containing the value.</td>
</tr>
<tr>
<td>Secondary key field</td>
<td>The secondary key field to search; can be a literal, or a field or variable containing the name of the secondary field.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the result, which is null if not found or the value of the primary key if found; can be stored in a field or variable.</td>
</tr>
<tr>
<td>Set flag in</td>
<td>The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>
**With Cross-Reference**

The Next Greater and Next Smaller commands combined with the Cross-Reference Type searches for the value specified for the cross-reference file and, if found, gets the value of the next greater or next smaller key.

**Properties:**

- **Get key command**
  
  The command selected; can be one of the following:

  - **Next Greater**
    
    Retrieves the record ID when the cross-reference value is greater than the specified cross-reference value.

  - **Next Smaller**
    
    Retrieves the record ID when the cross-reference value is less than the specified cross-reference value.

- **Key type**

  Specifies the key type. When Cross-reference is selected, the search is based on the value in the cross-reference file.

- **From file**

  The file name to be searched; must be the filename.

- **Cross-reference value**

  The value to be searched for; can be a number, a literal, or a field or variable containing the value.

- **Store result in**

  The location in which to store the result, which is null if not found or the value of the primary key if found; can be stored in a field or variable.
### Get List Command

#### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set flag in</td>
<td>The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>

#### With Primary Key

The Get List command combined with the Primary Key type gets a list of the primary keys in the file.

#### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key type</td>
<td>Specifies the key type. When Primary Key is selected, the search is based on the record IDs in the file.</td>
</tr>
<tr>
<td>From file</td>
<td>The file name to be searched; must be the filename.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the result, which is a list of record IDs separated by value marks or null if no record IDs are found; can be stored in a multivalued field or variable.</td>
</tr>
<tr>
<td>Set flag in</td>
<td>The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>
**With Secondary Key**

The Get List command combined with the Secondary Key type gets a list of primary keys based on a value in the specified secondary key.

**Properties:**

- **Key type**
  Specifies the key type. When Secondary Key is selected, the search is based on the values in the secondary field.

- **From file**
  The file name to be searched; must be the filename.

- **Key value**
  The value to be searched for; can be a number, a literal, or a field or variable containing the value.

- **Secondary key field**
  The secondary key field to search; can be a literal, or a field or variable containing the name of the secondary field.

- **Store result in**
  The location in which to store the result, which is a list of record IDs separated by value marks or a null if no record IDs are found; can be stored in a multivalued field or variable.

  The record IDs are listed by creation date, rather than a sorted order.

- **Set flag in**
  The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.
With Cross-reference

The Get List command combined with the Cross-Reference Type gets a list of primary keys based on a specified cross-reference value.

Properties:

- **Key type**: Specifies the key type. When Cross-reference is selected, the search is based on the cross-reference file.
- **From file**: The file name to be searched; must be the filename.
- **Cross-reference value**: The value to be searched for; can be a number, a literal, a field or variable containing the value, or "*" for all values.
- **Store result in**: The location in which to store the result, which is a list of record IDs separated by value marks or null if no record IDs are found; can be stored in a field or variable.
- **Set flag in**: The location in which to store the result of the command, which is Y if successful or N if not successful; can be stored in a field or variable.
The GET-NEXT-ELEMENT verb retrieves the key of the next value in a selection list or multivalued field and stores the key in a field or variable.

Properties:

- **Store result in**: The location in which to store the key to the record; can be a field or variable.

Notes

You must open the list or multivalued field with the OPEN-LIST verb before you can use this verb to get the next element.

If the selection list or multivalued field is empty, $READ.OK contains N.
See Also

OPEN-LIST verb.
The GET-PROPERTY verb retrieves the properties of an ActiveX object at runtime. This verb can only be used for ActiveX objects.

**Properties:**

- **For object**
  - The name of the ActiveX screen object that you want to retrieve the properties for; must be a literal specifying the name of the ActiveX object. The name must be in the same case and enclosed in quotation marks. For example:
  
  "TreeView1"
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property name</th>
<th>The property name of the ActiveX control; must be a literal specifying the name of the ActiveX property. The name must be in the same case and enclosed in quotation marks. For example:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>&quot;BorderStyle&quot;</td>
</tr>
<tr>
<td></td>
<td>The property name is displayed on the EDGE Configuration page and on the screen editor properties sheet.</td>
</tr>
<tr>
<td>Store value in</td>
<td>The location in which to store the result of the retrieval; can be a field or variable.</td>
</tr>
<tr>
<td>Store status in</td>
<td>The location in which to store the result of the retrieval; can be a field or variable.</td>
</tr>
<tr>
<td></td>
<td>Table 16-23 lists the status codes.</td>
</tr>
</tbody>
</table>
### Table 16-23. GET-PROPERTY Status Codes

<table>
<thead>
<tr>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful.</td>
</tr>
<tr>
<td>1</td>
<td>Invalid object name. No object could be found with the specified name on the current screen.</td>
</tr>
<tr>
<td>4</td>
<td>Object name is blank.</td>
</tr>
<tr>
<td>5</td>
<td>Invalid EDGE property name.</td>
</tr>
<tr>
<td>-1</td>
<td>Verb not implemented in ASCII operations.</td>
</tr>
</tbody>
</table>
The GET-RECORD-TSTAMP verb retrieves the timestamp (date and time last updated) for a record in an EDGE database file.

Properties:

- **Get timestamp from file**: The name of the file that contains the record; must be the filename. Do not enclose in quotation marks.
- **Record ID**: The record ID; can be a literal, or a field or variable containing the record ID.
- **Store result in**: The location in which to store the timestamp; can be a field or variable. The timestamp is returned in internal UNIX format (the number of seconds since January 1, 1970).
Notes

The timestamp can only be retrieved for files for which record tracking is enabled. Record tracking is described in Chapter 13.

If record tracking has not been set for the file, 0 (zero) is returned in the result.
The GOTO verb bypasses lines of logic and goes to a specific LABEL.

Properties:

Goto label  The name of the label; must be the name.

Notes

You can also use the GOTO and LABEL structure to perform looping actions within logic. When using this loop structure be sure to add a way to exit the loop.

See Also

The LABEL statement in Chapter 15, Logic Flows & Tasks and the LOOP verb.
GROUP-EXTRACT

The GROUP-EXTRACT verb copies a group or groups of data to a field or variable.

Properties:

- **Group extract from**: The value containing the data to be extracted; can be a literal, or a field or variable containing the value.
- **Number of groups to skip**: The number of groups to skip before beginning the extract; can be a number, or a field or variable containing a number. The number must be a non-negative integer.
**Properties** (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Separating character</td>
<td>The character that separates the groups; can be a literal, or a field or variable containing the character. The separator character cannot be a number. To specify a space, enter &quot; &quot; by entering a quotation mark, pressing the space bar, then entering another quotation mark.</td>
</tr>
<tr>
<td>Number groups to extract</td>
<td>The number of groups to extract; can be a number, or a field or variable containing a number. The number must be a non-negative integer.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The data extracted; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>

**Notes**

For this verb, a group is defined as data separated by a delimiter. For example, in the string 123/456/789, the slash (/) is a delimiter. When more than one group is extracted, the delimiter between the groups is also extracted. In the previous example, if one group is skipped and two groups are extracted, the result is 456/789.

This verb performs the same function as group extract in relational fields used in ad hoc reporting.
The INSERT-VALUE verb places a value at a specified position in a multivalued field or variable.

Properties:
- **Insert the value**: The value to be inserted in the multivalued field or variable; can be a number, a literal, or a field or variable containing the value.
- **Into the multi-value**: The location in which to insert the value; can be a multivalued field or variable.
- **At value position**: The number of the position in the multivalued field in which to insert the value; can be a number, or a field or variable containing the number. The first position is one (1).
If you specify zero (0), the value to be inserted is put in the first position and all other values in the multi-valued field are removed. (To sort values, see the SORT-VALUES verb.) For other examples, see Table 16-24.

<table>
<thead>
<tr>
<th>Multivalued Field</th>
<th>Insert X at value position 1</th>
<th>Insert X at value position 3</th>
<th>Insert X at value position 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X</td>
<td>A</td>
<td>X</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>B</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>C</td>
<td></td>
</tr>
</tbody>
</table>
The INTEGER verb changes a decimal number to a whole number by truncating any numbers to the right of the decimal place.

Properties:
- Convert value to integer: The number to be computed; can be a number, or a field or variable containing a number.
- Store result in: The location in which to store the integer result; can be a field or variable.

Notes
To retain the conversion, store the result in a field that is defined with zero decimal places. If the result is stored to a field that is defined
with decimal places, the integer will be stored with decimal places. For example, if 1.4 is converted to an integer and stored in a field defined with two decimal places, the number is stored as 1.00.

**See Also**

CALCULATE verb.
The IRA verb calculates the return on an Individual Retirement Account.

**Properties:**

- **Amount per year**: The amount to be invested per year; can be a number, or a field or variable containing the number.
- **Years of investment**: The number of years the IRA is to be invested at this rate; can be a whole number, or a field or variable containing the whole number.
Properties: (continued)

Percentage rate

The interest rate; can be a number with decimal places, or a field or variable containing the number. Do not include the percent symbol (%).

Store result in

The location in which to store the total compound interest based on the Annualized Compound Rate and the investment amount; can be a field or variable.

Notes

The Annualized Compound Rate is computed as follows, where:

- Years: The number of years in IRA.
- Amount: The amount invested into IRA in dollars each year.
- Rate: The fixed annual interest rate in percent.

\[
\text{Amount} \times \sum_{n=1}^{\text{Years}} \left( \frac{\text{Rate}}{100} \right)^n
\]
LOAD-LIST

The LOAD-LIST verb copies a list of record IDs from a record or selection list to a multivalued field or variable.

Properties:

Load list from  The source from which the list is loaded; can be one of the following:

- Record  The list is loaded from a record.
- Selection List  The list is loaded from a selection list.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Name of file</th>
<th>The name of the file containing the record to be loaded; must be the filename entered without quotation marks. Before loading the record, you must read the record you intend to load.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of selection list</td>
<td>The name of the selection list containing the record IDs; can be a literal, or a field or variable containing the name of the selection list.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the list of values; can be a multivalued field or variable. Each field or value in the record or the selection list is stored as a value in the multivalued field or variable. If the record contains multivalued fields, each value position in the multivalued field is stored as a separate value.</td>
</tr>
</tbody>
</table>

**Notes**

This verb should not be used with a record or selection list larger than 64K. Instead, use the OPEN-LIST verb.

For more information about selection lists, see Chapter 7.
See Also

The OPEN-LIST and GET-NEXT-ELEMENT verbs.
The LOCATE-BY-ACP verb locates sites which are closest to a specific telephone area code and prefix. This is an optional feature used with the Locator module. For details, see the Locator optional module documentation.
The LOCATE-BY-ZIP verb locates sites which are closest to a specific zip code. This is an optional feature used with the Locator module. For details, see the Locator optional module documentation.
The LOG verb allows you to log messages to a specified log file.

Properties:

- **Log message**: The message to send to the log file; can be a literal, or a field or variable containing the message.
- **Date/time stamp**: Specifies whether to log the date and time to the log file; otherwise, the date and time are not logged.

The date/time stamp format is:

```
DD/MM HH:MM:SS.nnnn
```
### Properties: (continued)

<table>
<thead>
<tr>
<th>WHERE</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>DD</td>
<td>Day in the range 01-31.</td>
</tr>
<tr>
<td>MM</td>
<td>Month in the range 01-12.</td>
</tr>
<tr>
<td>HH</td>
<td>Hour in the range 00-23.</td>
</tr>
<tr>
<td>MM</td>
<td>Minutes in the range 00-59.</td>
</tr>
<tr>
<td>SS</td>
<td>Seconds in the range 00-59.</td>
</tr>
<tr>
<td>nnnn</td>
<td>Fractions of the second. The number of digits in nnnn depends on the server platform.</td>
</tr>
</tbody>
</table>

**Logic/line stamp**

Specifies whether to log the logic flow name (if in logic) or the task name (if in task), and the LOG verb's line number in the logic flow or task; otherwise, the name and line number are not logged.

The Logic/Line Stamp format is:

```
NAME(LINE)
```

**where**

<table>
<thead>
<tr>
<th>NAME</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAME</td>
<td>The logic flow or task name.</td>
</tr>
<tr>
<td>LINE</td>
<td>The line number of the LOG verb in that logic flow or task.</td>
</tr>
</tbody>
</table>

Logic flow names are logged as:

```
LOGICixPROJ
```
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Where</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGIC</td>
<td>The name of the logic flow.</td>
</tr>
<tr>
<td>PROJ</td>
<td>The name of the project.</td>
</tr>
</tbody>
</table>

Task names are logged without the ix and project name.

**Log to file**

The name of the log file; can be one of the following:

- **ASCII Guide Watcher Log**
  - Logs the message to the current guide’s guide watcher log. Messages are logged only if ASCII Guide Watcher is enabled. If the verb is invoked in a task, and the ASCII guide watcher log is specified, nothing is logged.
  - This option is only valid when using ASCII for operations, not EDGE Client.

- **Standard Error Log**
  - Logs the message to the standard error log of the current guide or task. Messages are logged only if standard error logging is enabled.

- **User-defined server log**
  - Logs the message to a specified file on the server. If this is selected, a prompt is displayed, asking for the filename.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server log filename</td>
<td>The filename on the server to log to; can be a literal, or a field or variable containing the filename. This prompt is displayed only if the User-Defined Server Log is selected as the To file. The filename for the user-defined log file can be any valid UNIX filename. Relative paths use the project directory as the current working directory. If the file does not exist, one will be created. If the file does exist, log messages are appended to the end of the file. If EDGE cannot create or open the file, an error message is logged to the standard error and the LOG verb is ignored.</td>
</tr>
</tbody>
</table>

**Notes**

One major use of the LOG verb is to test guides under development. Therefore, messages are logged by both the development and the operations guide, in test and live mode.

Messages logged to the standard error are also written in the EDGE Client Server Debug log.
The LOOP verb allows you to define a sequence of actions which are repeated until a specified condition is met. You can use the LOOP verb, for example, to retrieve all the record IDs from a file.
**Properties:**

<table>
<thead>
<tr>
<th>Loop command</th>
<th>The command to be executed; can be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>End</strong></td>
<td>Ends a loop sequence.</td>
</tr>
<tr>
<td><strong>Exit</strong></td>
<td>Exits the current loop and goes to the statement following the End command for that loop.</td>
</tr>
<tr>
<td><strong>For</strong></td>
<td>Begins a loop with a specified starting value that is incremented or decremented by a specified value.</td>
</tr>
<tr>
<td><strong>For Each</strong></td>
<td>Begins a loop which repeats the loop based on the loop type specified in the For each type property.</td>
</tr>
<tr>
<td><strong>Repeat</strong></td>
<td>Starts a loop sequence which is repeated until the condition set by the Until command is met.</td>
</tr>
<tr>
<td><strong>Until</strong></td>
<td>Sets the condition under which a repeat command is completed.</td>
</tr>
<tr>
<td><strong>While</strong></td>
<td>Starts a loop sequence which is repeated while the specified condition is true.</td>
</tr>
</tbody>
</table>

**For each type**
The loop type; can be one of the following:
- character in a variable or field
- record ID in a file
Properties: (continued)

- value of a multivalued field
  This property is used with the **For Each** command.

<table>
<thead>
<tr>
<th>Control variable</th>
<th>The value on which to base the loop; can be a literal, a file name, a field or variable, or a multivalued field or variable containing the values.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This property is used with the <strong>For Each</strong> command.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Store result in</th>
<th>The location in which to store the retrieved value; can be a field or variable.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This property is used with the <strong>For Each</strong> command.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value 1</th>
<th>The location of the value to be compared; can be a field or variable. This is the loop variable.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>This property is used with the <strong>Until</strong> or <strong>While</strong> command.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Comparison type</th>
<th>The symbol representing the type of comparison; can be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>= Equal to.</td>
</tr>
<tr>
<td></td>
<td>&lt; Less than.</td>
</tr>
<tr>
<td></td>
<td>&gt; Greater than.</td>
</tr>
<tr>
<td></td>
<td>&lt;= Less than or equal to.</td>
</tr>
<tr>
<td></td>
<td>&gt;= Greater than or equal to.</td>
</tr>
<tr>
<td></td>
<td># Not equal to.</td>
</tr>
<tr>
<td><strong>Properties:</strong> (continued)</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Value 2</strong></td>
<td></td>
</tr>
<tr>
<td>The location of the value which identifies the condition; can be a literal, or a field or variable containing the value. This is the ending value.</td>
<td></td>
</tr>
<tr>
<td>This property is used with the Until or While command.</td>
<td></td>
</tr>
<tr>
<td><strong>For loop variable</strong></td>
<td></td>
</tr>
<tr>
<td>The location in which to store the current loop setting; can be a field or variable.</td>
<td></td>
</tr>
<tr>
<td>This property setting is initialized with the starting value when the loop begins.</td>
<td></td>
</tr>
<tr>
<td>This property is used with the For command.</td>
<td></td>
</tr>
<tr>
<td><strong>Starting value</strong></td>
<td></td>
</tr>
<tr>
<td>The number on which to start the loop; can be a number, or the name of a field or variable containing the number.</td>
<td></td>
</tr>
<tr>
<td>This property is used with the For command.</td>
<td></td>
</tr>
<tr>
<td><strong>Ending value</strong></td>
<td></td>
</tr>
<tr>
<td>The number after which to end the loop; can be a number, or a field or variable containing the number. The ending value is inclusive.</td>
<td></td>
</tr>
<tr>
<td>This property is used with the For command.</td>
<td></td>
</tr>
</tbody>
</table>
The LOOP verb allows you to perform a series of actions multiple times. There are several forms of the LOOP verb. Each loop must start with one of the following commands:

- For command.
- For Each command.
- Repeat command.
- While command.

The number of times the loop is performed depends on values that are specified in the command used to start the LOOP verb. See the description of each command.

Each loop must end with an End command. Statements following the End command are not part of the loop. The start and end of a
A loop must be grouped together within one control structure; for example, if you start a loop with `THEN` in an `IF-THEN-ELSE` statement, you must close the loop within the grouped `THEN` statements. When a loop is completed or exited, the first statement following the `End` command is executed.

You can unconditionally exit the loop by using the `Exit` command. Loops can be nested inside other loops. The only limitation to nesting is the 64K limitation for logic flows and tasks.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>End</em></td>
<td>The End command ends a loop sequence and sends the program control back to the loop statement that started the current loop. Every loop must end with the End command. If a loop includes a nested loop, both loops must include End commands.</td>
</tr>
<tr>
<td><em>Exit</em></td>
<td>The Exit command exits the current loop and goes to the statement following the End command for that loop. If this command is used in an inner loop of a nested loop, the inner loop is exited, but the outer loop is not exited.</td>
</tr>
<tr>
<td><em>For</em></td>
<td>The For command begins a loop with a specified starting value that is incremented or decremented by a specified value.</td>
</tr>
</tbody>
</table>
The statements following the For command are executed each time the loop is performed. When an End command is encountered, the loop variable is adjusted by the step value, then tested against the ending value.

If the loop variable is within the bounds of the ending value, the loop is performed again. If the loop variable is outside the bounds of the ending value (greater than the value for incrementing steps, less than for decrementing steps), the loop is complete and control passes to the statement following the End command.

A For loop has the following format:

```
ALWAYS LOOP command for loop variable n starting value n to ending value n step value n

... body of loop

... ALWAYS LOOP command end
```

**For Each**

The For Each command begins a loop that repeats for one of the following:

- **Character in a Variable/Field.**
- **Record ID in a File.**
- **Value of a Multivalued Field.**
When you select a loop type, prompts for additional properties are displayed.

**Character in a Variable/Field**

The character in a variable/field begins a loop with the first character in the variable and continues the loop for each character in the variable.

The statements following the character in a variable/field are executed each time the loop is performed. When an End command is encountered, the next character in the variable/field is retrieved and the loop is performed again.

When all the characters in the variable/field have been retrieved, the loop is complete and control passes to the statement following the End command.
A character in a variable/field loop has the following format:

   ALWAYS LOOP command for each character in variable n store result in n
   ...
   body of loop
   ...
   ALWAYS LOOP command end

**Record ID in a File**

The record ID in a file begins a loop with the first record ID in the file and continues the loop for each record in the file.

The statements following the record ID in a file are executed each time the loop is performed. When an End command is encountered, the next record ID in the file is retrieved and the loop is performed again.

When all the record IDs in the specified file have been retrieved, the loop is complete and control passes to the statement following the End command.

The record associated with the record ID is not retrieved. If you wish to access the fields in the record, you must use the **READ-RECORD** verb.
A record ID in a file loop has the following format:

    ALWAYS LOOP command for each record ID from file n store result in n
    ...
    body of loop
    ...
    ALWAYS LOOP command end

**Value of a Multivalued Field**

The value of a multivalued field begins a loop with the first value in
the multivalued field and continues the loop for each value in the
field.

The statements following the value in a multivalued field are exe-
cuted each time the loop is performed. When an End command is
encountered, the next value in the specified field is retrieved and the
loop is performed again.

When all the fields in the specified field have been retrieved, the loop
is complete and control passes to the statement following the End
command.

A value in a multivalued field loop has the following format:

    ALWAYS LOOP command for each value of multi-valued field n store
    result in n
    ...
    body of loop


**Repeat**

The Repeat command starts a loop sequence which is repeated until the condition set by the Until command is met.

An Until command must follow the Repeat command and precede the End Loop command. Statements can be placed between the Repeat and Until commands, as well as between the Until and End commands. The End command returns to the Repeat command.

The statements between the Repeat and Until commands are executed each time the loop is performed. The Until command determines when the loop is complete.

A Repeat loop has the following format:

```
ALWAYS LOOP command repeat
...
repeat for all values
...
ALWAYS LOOP command until n {condition} n
...
body of loop
...
ALWAYS LOOP command end
```
**Until**

The Until command sets the condition under which a Repeat command is completed.

The Until command must follow a Repeat command.

The statements following the Until command are executed each time the loop is performed. When an End command is encountered, it returns to the Repeat command. The statements between the Repeat and the Until, if any, are executed.

When the Until statement is encountered, the condition is tested. If the condition is false, the statements following the Until are executed. If the condition is true, the loop is complete and control passes to the statement following the End command.

An Until loop has the following format:

```
ALWAYS LOOP command repeat
  ...
  repeat for all values
  ...
ALWAYS LOOP command until n {condition} n
  ...
  body of loop
  ...
ALWAYS LOOP command end
```
**While**

The While command starts a loop sequence which is repeated while the specified condition is true.

The statements following the While command are executed each time the loop is performed.

When an End command is encountered, the While condition is tested. If the test is true, the loop is performed again. If the test is false, the loop is complete and control passes to the statement following the End command.

The While command does not change the value of the loop variable. The variable must be changed either by statements within the loop, or by some other means; for example, a key pressed by the operator.

A While loop has the following format:

```
ALWAYS LOOP command while n {condition} n
    ... body of loop
    ...
ALWAYS LOOP command end
```
The MENU verb adds or deletes a menu, menu item, or popup menu to EDGE Client. The MENU verb also enables or disables, checks or clears menu items.

Any changes made by the MENU verb are automatically removed at the end of a call.
### Properties:

<table>
<thead>
<tr>
<th>Menu command</th>
<th>The following MENU commands are available for the MENU verb:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add</td>
<td>Adds a new menu item.</td>
</tr>
<tr>
<td>Check</td>
<td>Adds a check mark to the menu item.</td>
</tr>
<tr>
<td>Disable</td>
<td>Disables the menu item. When disabled, the item is grayed.</td>
</tr>
<tr>
<td>Enable</td>
<td>Enables the menu item.</td>
</tr>
<tr>
<td>Query</td>
<td>Queries the status of the menu item.</td>
</tr>
<tr>
<td>Remove</td>
<td>Removes a menu item.</td>
</tr>
<tr>
<td>Uncheck</td>
<td>Clears the check mark.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Menu text</th>
<th>The name of the menu or menu item; can be a literal, or a field or variable containing the text.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tool tip text</td>
<td>Additional help text to be displayed at the bottom of the screen; can be a literal, or a field or variable containing the text.</td>
</tr>
<tr>
<td>Help ID</td>
<td>The ID of the help text for the menu; passed to WinHelp for context sensitive help. (Currently not supported.)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Menu location</th>
<th>The location of the menu item; can be a literal, or a field or variable containing the location. One of the following can be specified:</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Menu bar.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>P</strong></td>
<td>Pop-up menu. (Currently not supported.)</td>
</tr>
<tr>
<td><strong>Menu location reference</strong></td>
<td>The name of the menu on which to place the item. If the item menu is to be on a submenu, separate the names with a pipe (</td>
</tr>
<tr>
<td><strong>Menu position</strong></td>
<td>Specifies where the item is to go on the menu; can be a literal, or a field or variable containing the position; can be one of the following:</td>
</tr>
<tr>
<td><strong>A</strong></td>
<td>After the item specified in the Menu position reference property.</td>
</tr>
<tr>
<td><strong>B</strong></td>
<td>Before the item specified in the Menu position reference property.</td>
</tr>
<tr>
<td><strong>F</strong></td>
<td>Insert first.</td>
</tr>
<tr>
<td><strong>L</strong></td>
<td>Insert last.</td>
</tr>
<tr>
<td><strong>Menu position reference</strong></td>
<td>The name of the item to use as a reference to place the new item; can be a literal, or a field or variable containing the value. This property is required if A or B is specified as the menu position.</td>
</tr>
<tr>
<td><strong>Route to screen</strong></td>
<td>The name of the window, screen, and object to route to when the menu item is selected; can be a literal, or a field or variable containing the name. If blank, no window, screen, or object is routed to.</td>
</tr>
</tbody>
</table>
This property is only enabled if adding a menu item; the Menu command must be A and the Menu text cannot be null.

A Window Helper screen is available to assist with the syntax of this property; to display the Window Helper, select the three-dot button.

Set to result

The name of a result to set when the menu item is selected; can be a literal, or a field or variable containing the name. If blank, no result is set.

Call logic

The name of a logic flow to call when the menu item is selected; can be a literal, or a field or variable containing the name. If blank, no logic flow is called.

This property is only enabled if adding a menu item; the Menu command must be A and the Menu text cannot be null.

Accelerator key

The accelerator key to be used to activate the menu item; can be a literal, or a field or variable containing the value. If blank, no accelerator key is defined. The accelerator key is added to the text for this menu item.

Skip exit screen logic

Specifies whether exit screen logic is performed if the menu item routes to another screen; can be a literal, or a field or variable containing the value. The value can be one of the following:

Y The exit screen logic is skipped.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Before separator</td>
<td>N</td>
<td>The exit screen logic is performed.</td>
</tr>
<tr>
<td></td>
<td>Y</td>
<td>Adds a separator line before the menu item; can be a literal, or a field or variable containing the value. The value can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>No separator line is added.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the item is the first item on the menu, this property has no effect.</td>
</tr>
<tr>
<td>After separator</td>
<td>Y</td>
<td>Specifies whether to add a separator line after the menu item; can be a literal, or a field or variable containing the value. The value can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>No separator line is added.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the item is the last item on the menu, this property has no effect.</td>
</tr>
<tr>
<td>Menu type</td>
<td>Y</td>
<td>Specifies whether this is a new menu or a command on an existing menu; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>Creates a new menu.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Puts a command on an existing menu.</td>
</tr>
<tr>
<td>Disabled</td>
<td>Y</td>
<td>Specifies whether the menu item is disabled by default; can be a literal, or a field or variable containing the value. The value can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>N</td>
<td>No separator line is added.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>If the item is the last item on the menu, this property has no effect.</td>
</tr>
</tbody>
</table>
Properties: (continued)

Y The item is disabled.
N The item is enabled.

Checked

Specifies whether the menu item is checked by default; can be a literal, or a field or variable containing the value. If an item is checked, it is active. The value can be one of the following:

Y The item is checked.
N The item is not checked.

Store status in

The location in which to store the results of the MENU command; can be a field or variable. The following information is returned, separated by value marks:

Status code Code as listed in Table 16-25.
Menu text 1 The name of the first menu item.
Code 1 A string indicating information about the first item; one or more of the following can be returned:

C Menu item is checked.
D Menu item is disabled.
G Menu item cannot be modified.
Properties: (continued)

- P  Menu item is a pop-up menu.

- Menu text n  The name of the nth menu item.

- Code n  A string indicating information about the nth item.

Add

The Add command adds a new menu item. All of the verb properties can be specified.

Check

The Check command places a check next to the menu item. The following properties can be specified with this command:

- Menu text.
- Menu location.
- Menu location reference.
- Store status in.

Disable

The Disable command disables the menu item, which means the item cannot be selected. The following properties can be specified:

- Menu text.
- Menu location.
Enable

The Enable command enables the menu item, which allows the item to be selected. The following properties can be specified:

- Menu text.
- Menu location.
- Menu location reference.
- Store status in.

Query

The Query command returns the status of menu items. The item that is queried depends on the properties filled in. To query the status of the entire menu bar, leave the Menu text property blank and set the location to "M".

The following properties can be specified:

- Menu text.
- Menu location.
- Menu location reference.
- Store status in.
**Remove**

The Remove command removes the menu item. The item is restored at the end of call. The following properties can be specified:

- Menu text.
- Menu location.
- Menu location reference.
- Store status in.

**Uncheck**

The Uncheck command clears the check mark from the menu item. The following properties can be specified:

- Menu text.
- Menu location.
- Menu location reference.
- Store status in.

**Table 16-25. Status Codes for MENU Verb**

<table>
<thead>
<tr>
<th>Status Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful.</td>
</tr>
<tr>
<td>1</td>
<td>Verb command is invalid.</td>
</tr>
<tr>
<td>2</td>
<td>Menu item does not exist or is not applicable.</td>
</tr>
<tr>
<td>3</td>
<td>Windows ran out of handles.</td>
</tr>
<tr>
<td>Status Code</td>
<td>Description</td>
</tr>
<tr>
<td>-------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>4</td>
<td>Invalid parameters.</td>
</tr>
<tr>
<td>5</td>
<td>Internal error.</td>
</tr>
<tr>
<td>6</td>
<td>Windows error.</td>
</tr>
<tr>
<td>7</td>
<td>Dynamic menu limit exceeded.</td>
</tr>
<tr>
<td>8</td>
<td>Not supported.</td>
</tr>
<tr>
<td>9</td>
<td>Out of memory.</td>
</tr>
<tr>
<td>10</td>
<td>Menu already exists.</td>
</tr>
<tr>
<td>11</td>
<td>Menu does not exist.</td>
</tr>
<tr>
<td>12</td>
<td>Item does not exist.</td>
</tr>
<tr>
<td>13</td>
<td>Cannot remove built-in menu item.</td>
</tr>
<tr>
<td>14</td>
<td>Too many items for EDGE Client to support.</td>
</tr>
<tr>
<td>15</td>
<td>Invalid menu path.</td>
</tr>
</tbody>
</table>
The MESSAGE verb displays messages using a standard Windows message box.

Properties:

Message title  The title for the message, which is displayed in the title bar of the message box; can be a literal, or a field or variable containing the message title.
Message text  The message to display; can be a literal, or a field or variable containing the message.

If you want the message text to be displayed on multiple lines, copy each line of the message text into a value in a multi-valued field. Then use the `CONVERT` verb to convert value marks (ASCII 29) to line feeds (ASCII 10). The message box replaces the line feed characters with line breaks.

Buttons  The response buttons to display with the message; can be a number representing the button, or a field or variable containing the number. One of the following numbers can be specified:

- Abort, Retry, Ignore (1).
- OK (2).
- OK, Cancel (5)
- Retry, Cancel (6).
- Yes, No (4).
- Yes, No, Cancel (3).
Properties: (continued)

Default button

Specifies which of the buttons is the default button; can be a number representing the icon, or a field or variable containing the number. When the message is displayed, this default button is highlighted and can be selected by pressing RETURN. One of the following numbers can be specified:

- First button (1).
- Second button (2).
- Third button (3).

Icon

Specifies whether to display an icon with the message; can be a number representing the icon, or a field or variable containing the number. One of the following numbers can be specified:

- Exclamation mark (4).
- Information mark (1).
- No icon (0).
- Question mark (2).
- Stop sign (3).

System modal

Specifies whether or not a response is required while the message box is displayed; can be one of the following:
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store result in</td>
<td>The location in which to store the results of the button pressed by the user; can be a field or variable. One of the following can be returned, depending on the type of buttons that were defined:</td>
</tr>
<tr>
<td></td>
<td>ABORT.</td>
</tr>
<tr>
<td></td>
<td>CANCEL.</td>
</tr>
<tr>
<td></td>
<td>IGNORE.</td>
</tr>
<tr>
<td></td>
<td>NO.</td>
</tr>
<tr>
<td></td>
<td>OK.</td>
</tr>
<tr>
<td></td>
<td>RETRY.</td>
</tr>
<tr>
<td></td>
<td>YES.</td>
</tr>
<tr>
<td></td>
<td>If an unusual error is encountered, the result will be Errornnn.</td>
</tr>
</tbody>
</table>

Y Response is required before user can continue. The box remains in the foreground until the user responds.

N No response is required immediately. The box can be left open in the background while the user does something else.

This property is optional. If not specified, N is assumed.
The MONEY-CONVERSION verb converts an unformatted number to a currency format.

**Properties:**

- **Money convert the value:** The value to be converted to currency format; can be a number, or a field or variable containing the number.
- **Decimal places:** The number of decimal places for the formatted number; must be a number.
When the number of decimal places specified is less than in the input value, the number is rounded. For example, 123.56 formatted with 0 (zero) decimal places is rounded to 124.

When the number of decimal places specified is greater than in the input value, a decimal place and zeroes are added to the number. For example, 123 formatted with 2 decimal places is formatted as 123.00.

**Using format**

The type of format to use with the conversion, which reflects the money sign, comma, and decimal separators specified for the site; can be one of the following:

- Comma 2,500.00
- Money sign $2500.00
- Comma and money sign $2,500.00
- None 2500.00

The money sign, comma, and decimal separator can be specified for non-U.S. formats. For information, see Chapter 3.

**Leading character**

The leading character to precede the monetary value; can be one of the following:

- zero (0)
### Properties: (continued)

- **asterisk (*)**
- **blank**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of leading chars</td>
<td>The number of leading characters to place before the formatted value; must be a number.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the formatted value; can be a field or variable.</td>
</tr>
</tbody>
</table>
The MSGQUEUE verb is used with to send and receive messages from the server. For more information, see Chapter 18, Message Queues.
The MULTIPLY verb multiplies a number by another number.

**Properties:**

- **Multiply the value:** The number to be multiplied; can be a number, or a field or variable containing the number.
- **With the value:** The number to use as the multiplier; can be a number, or a field or variable containing the number.
- **Store result in:** The location in which to store the result of the multiplication; can be a field or variable name.
See Also

CALCULATE verb.
OPEN-LIST

The OPEN-LIST verb positions a pointer at the beginning of a list.

Properties:

- Open list from
  - Field name: The list is a multivalued field or variable name.
  - Selection list: The list is a selection list.
- Field name: The name or location of the field list; can be a multivalued field or variable name.
**Notes**

Only one multivalued field or list can be opened at a time. When you open a second list, the first list is closed. If the selection list exists and has records, the OPEN-LIST verb is successful and the following occurs:

- The system field $READ.OK is set to Y.
- The pointer is positioned to the beginning of the list.

If the selection list does not exist or does not have any records, the OPEN-LIST verb fails, and the following occurs:

- The system field $READ.OK is set to N.
To retrieve a value from either a multivalued field or selection list, you can use:

- **OPEN-LIST verb** To point to the beginning of the list.
- **GET-NEXT-ELEMENT verb** To get each value in the list.
The OPERATOR-ACCESS verb sets properties associated with operator access.
Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project ID</td>
<td>The name of the project; can be a literal, a field or variable containing the project name.</td>
</tr>
<tr>
<td>Command</td>
<td>The following operator access commands are available:</td>
</tr>
<tr>
<td></td>
<td>Add Queue, Get Property, Queue List, Remove Queue, Set Property.</td>
</tr>
<tr>
<td>User ID(s)</td>
<td>The user IDs of the operators to receive additional queues; can be a literal, or the field or variable containing several user IDs.</td>
</tr>
<tr>
<td>Store status in</td>
<td>The location in which to store the status code; can be a field or variable. For a listing of status codes, see Table 16-26.</td>
</tr>
</tbody>
</table>

The remaining properties are specific to one or more commands, and are described in the following sections.

Notes

The OPERATOR-ACCESS verb can be used to make changes to existing operator access definitions. Status codes for OPERATOR-ACCESS
are listed in Table 16-26. When used in operations or in testing in live mode, access is changed for the next call.

The verb has no effect in test mode.

To save user IDs or queue names in a multivalued field or variable, use the APPEND-VALUE verb.

**Table 16-26. OPERATOR-ACCESS Status Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>null</td>
<td>Command was successful.</td>
</tr>
<tr>
<td>1</td>
<td>Project does not exist.</td>
</tr>
<tr>
<td>2</td>
<td>User ID does not exist.</td>
</tr>
<tr>
<td>3</td>
<td>User ID does not have an operator access record.</td>
</tr>
<tr>
<td>4</td>
<td>Command was unsuccessful because the user ID property was not passed a value.</td>
</tr>
</tbody>
</table>

**See Also**

For more information on User Access, see Chapter 2, Setting Up Projects.

For more information on Queues, see Chapter 14, Queues.
**Add Queue**

The Add Queue command adds the specified queues to the specified users.

- **Queue(s)**: The queues to be added; can be a literal, or a field or variable containing the queue names. You can specify /U and /G queues. The queue must already exist to be used.

- **Search queue list**: Specifies whether to add the queues to the search and manual queues; can be one of the following:
  - Yes: Queues are added to the search queues.
  - No: Queues are added to the manual queues.

- **Add at position(s)**: The position in the search queue list at which to add the queues; can be a number, or a field or variable containing the number.

  If 1 is entered, for example, the queue is added to the beginning of the list. This property is only available when the value for the Search Queue List property is Yes.

**Get Property**

The Get Property command gets the value for a property based on specified agents.

- **Get property**: The property for which to retrieve the current setting; can be one of the following:
  - Inbound
  - Manual Access Only
Queue List

The Queue List command retrieves the list of queues assigned to a specific agent.

- **Search queue list**: Specifies which queues are to be retrieved; can be one of the following:
  - Yes: The search queue list is retrieved.
  - No: The manual queue list is retrieved.

- **Store queue list in**: Destination for the multivalued list of queue names; can be stored in a field or variable.

- **Store property in**: The location in which to store the setting of the property; can be a field, multivalued field, or variable. The setting can be one of the following:
  - Y: The property is set for the agent.
  - N: The property is disabled for the agent.
### Remove Queue

The Remove Queue command removes queues from specified operators’ access.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Queue(s)</td>
<td>The names of the queues to be removed from access; can be a literal, or a field or variable containing the queue names.</td>
</tr>
<tr>
<td>Search queue list</td>
<td>Specifies whether to remove queues from the search or manual queue lists; can be one of the following:</td>
</tr>
<tr>
<td>Yes</td>
<td>The queue is removed from the search queue list.</td>
</tr>
<tr>
<td>No</td>
<td>The queue is removed from the manual queue list.</td>
</tr>
</tbody>
</table>

### Set Property

The Set Property command sets a value for a specified property for one or more agents.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set property</td>
<td>Indicates which property to be set; can be one of the following:</td>
</tr>
<tr>
<td>Inbound</td>
<td>Allows the agent to accept inbound calls.</td>
</tr>
<tr>
<td>Manual access only</td>
<td>Restricts the agent to accessing call records manually.</td>
</tr>
<tr>
<td>Outbound</td>
<td>Allows the agent to make outbound calls.</td>
</tr>
<tr>
<td>Project access</td>
<td>Allows operator access to the project.</td>
</tr>
<tr>
<td>Allow access to property</td>
<td>Training</td>
</tr>
<tr>
<td>--------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>View group queues</td>
<td>Allows agent to view group queues.</td>
</tr>
<tr>
<td>View protected queues</td>
<td>Allows agent to view protected queues.</td>
</tr>
<tr>
<td>View shared queues</td>
<td>Allows agent to view shared queues.</td>
</tr>
</tbody>
</table>

Indicates whether to allow or disable the property.; one of the following can be specified:

- Yes: The specified operators are allowed access to the project.
- No: The specified operators are denied access to the project.
The PASSWORD verb allows you to specify a user ID and password to be verified against the PASSWORD field in the USER.ID.FILE for that user and to return the result of the comparison.

**Properties:**
- **Password command:** The command to execute; can be one of the following:
  - **Change:** Changes the password in the USER.ID.FILE to the password in the **Password** property.
  - **Verify:** Compares the password in the USER.ID.FILE to the password in the **Password** property.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ID</td>
<td>The user ID of the password to be verified; can be a literal, or a field or variable containing the user ID.</td>
</tr>
<tr>
<td>Password</td>
<td>The password to be compared; can be a literal, or a field or variable containing the password.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the result; can be a field or variable. The following results are returned:</td>
</tr>
<tr>
<td>Yes</td>
<td>The command was successful.</td>
</tr>
<tr>
<td>No</td>
<td>The meaning depends on the password command as follows:</td>
</tr>
<tr>
<td>Change command</td>
<td>Password was not changed.</td>
</tr>
<tr>
<td>Verify command</td>
<td>The user ID does not exist, or the password does not match.</td>
</tr>
</tbody>
</table>
The PICTURE-FORMAT verb allows you to store a value with a specified pattern.

**Properties:**

- **Format the value**
  - The value to format; can be a number, a literal or a field or variable containing a value.

- **Using format picture**
  - The format shown as a picture (mask) and enclosed in quotation marks. The picture can include the following:
Properties: (continued)

- Symbols that indicate the number of characters to be extracted and the type of fill to be added. The symbols are described below.

- Characters to add to the formatted result.

The following symbols indicate the type of fill to be added when the value to be formatted has fewer characters than the format picture:

#  Filled with blanks.
%
*  Filled with asterisks.

The placement of the filled characters depends on the specified justification. If left-justified, the fill is to the right of the value, for example ABC**. If right-justified, the fill is to the left, for example, ****ABC.

The symbols used to represent characters can be entered as either:

- Multiple symbols that indicate one character per symbol. For example, ###.
- A symbol followed by a number. For example, #3 is the same as ##. 

**Properties: (continued)**

If the format picture has fewer characters than the value to be formatted, the resulting value is truncated from the right or left, depending on the justification. For example, the value ABCDEF with a ### format and left justification results in ABC. The same value formatted with right justification results in DEF.

**Justification**

The field justification; can be one of the following:

- **Left** Justifies the value to the left; generally used with alphanumeric values. If the format picture is longer than the format value, the value is filled to the left of the result.

- **Right** Justifies the value to the right; generally used with numeric values. If the format picture is longer than the format value, the value is filled to the right of the result.

The type of fill depends on the symbol used in the picture format. For more information, see Using format picture.

**Store result in**

The location to in which to store the result; can be a field or variable name. If the result contains fill characters, (blanks, zeroes, or asterisks), the fill characters are stored regardless of the justification specified in the field definition.
Notes

Symbols can be entered between the characters. The entire picture must be enclosed in quotation marks. For example, the pattern for social security numbers could be defined as “###-##-####” or “#3-#2-#4.”
The POWER verb raises a number by a specified power. For example, if 5 is raised to the power of 2, the result is 25.

Properties:
- **Raise the value**: The number to be raised; can be a number, or a field or variable containing the number. The number can be a fractional number, for example, 2.5.
- **To the power**: The number of the power; can be a number, or a field or variable containing the number.
- **Store result in**: The location in which to store the result of the power computation; can be a field or variable name.
See Also

CALCULATE verb.
The PRINT-CORRESPONDENCE verb prints a correspondence report.

Properties:

Correspondence name
The name of the correspondence report; can be a literal, or a field or variable containing the name of the correspondence.

Select the arrow to display a list of currently defined correspondence reports.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Print using</th>
<th>The records to be processed; can be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>All IDs</td>
<td>Uses all the record IDs in the file.</td>
</tr>
<tr>
<td>Selection list</td>
<td>Uses the IDs from a selection list.</td>
</tr>
<tr>
<td>IDs</td>
<td>Uses individual IDs. If selected, can be either of the following:</td>
</tr>
<tr>
<td></td>
<td>$ID Uses the current record ID. In a logic flow, the current record is printed. In a task, the multi-record processing setting determines the records to be processed.</td>
</tr>
<tr>
<td></td>
<td>IDs The IDs must be enclosed in quotation mark and separated by commas; for example, “100, 101, 102”. If used in a task, the multi-record option should be set to process using no record IDs.</td>
</tr>
<tr>
<td>Selection list</td>
<td>The name of the selection list. The list name must be enclosed in quotation marks.</td>
</tr>
</tbody>
</table>

If the multiprocessing option processes individual IDs or IDs
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID list</td>
<td>The name of the ID list.</td>
</tr>
<tr>
<td>Using printer device</td>
<td>The printer device; can be a literal, or a field or variable containing the EDGE name of the printer.</td>
</tr>
<tr>
<td></td>
<td>Select the arrow to display the list of currently defined available printers.</td>
</tr>
<tr>
<td>Run in foreground</td>
<td>Specifies whether to run in the foreground; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes  Reports are processed in the foreground.</td>
</tr>
<tr>
<td></td>
<td>No   Reports are processed as a UNIX background process.</td>
</tr>
<tr>
<td>Prompt on error</td>
<td>Specifies whether to prompt the user when an error occurs. Available only when run in foreground. Can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes  When an error occurs, the processing is halted and a message is displayed.</td>
</tr>
<tr>
<td></td>
<td>No   When an error occurs, the user is not prompted and the report continues to run. This option should be used if running from the task scheduler.</td>
</tr>
</tbody>
</table>
Notes

If the multiprocessing option processes individual IDs or IDs from a selection list, the correspondence is printed for each ID selected.

If the correspondence is to print using data from a new record, that record must first be written to a file.

In a task, the values displayed for RECORD ID and RECORD COUNT reflect the record selection made in the multi-task processing option as displayed in Figure 16-2. The RECORD COUNT displays the total number of records processed. If you are making the record selection through the PRINT-CORRESPONDENCE verb, these fields will not reflect the progress.

The PRINT-CORRESPONDENCE verb does not work from a global level task. If used, EDGE does not generate a task assembly error, but it does generate a run-time error indicating CORRESPOND cannot be opened. This is because the Correspondence module is not available from the global level of EDGE.

If the value of the Prompt on error property is Yes and an error is encountered, an error message similar to that in Figure 16-2 is displayed. To continue, press ENTER.
An alternative method of printing correspondence reports from tasks is through the EXECUTE verb combined with the CORRPRINT command. For details, see the CORRPRINT command in Chapter 10. For more information about correspondence, see Chapter 6.
The PRINT-VALUES verb sends the values of a single or multivalued field to a printer.

**Properties:**

- **Print the values**
  - The values to be printed; can be a number, a literal, or a field or variable containing the values.

- **To printer**
  - The printer to be used. Select the arrow to display the list of currently defined printers.
The QUEUE verb controls the scheduling and status of a queue.

Properties:

Queue command

The command to be executed for the queue; can be one of the following:
Properties:  (continued)

- Close.
- Count.
- Open.
- Open by Schedule.
- Queue List.
- Schedule.
- Status.

**Queues**

The queues to be counted; can be a literal, or a field or variable containing the names of the queues. This property is not available for the Queue list command.

The other properties are specific to one or more commands, and are described in the following sections.

**Notes**

To save queue names in a multivalued field or variable, use the APPEND-VALUE verb. To change a date to internal format, see the DATE-FORMAT verb.

The following commands can also be specified through the Queue Definitions module.

- Close.
• Open.
• Open by Schedule.
• Schedule.

For more information about queues, see Chapter 14.

**Close**

The Close command closes the queues, which prevents call events from being retrieved.

**Properties:**

There are no properties specific to this command.

**Notes**

After this command is invoked in a guide, operator access to the specified queues is denied. The queue access remains closed until it is changed either through the Open for the QUEUE verb or through the Queue Definition screen.

**Count**

The Count command counts the call events in specified queues.

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>From date</td>
<td>The schedule date at which to start the count, which is a required entry; can be one of the following:</td>
</tr>
</tbody>
</table>
Properties: (continued)

- A date in internal format.
- A field with the data type as date, or any field with the date stored in internal format.
- A variable containing the date in internal format.
- Null (""") to start at zero (December 31, 1967). Also used for FIFO queues.

To date

The schedule date at which to end the count, which is a required entry; can be one of the following:

- A date in internal format.
- A field with the data type as date, or any field with the date stored in internal format.
- A variable containing the date in internal format.
- Null (""") to end at today’s date. Also used for FIFO queues.

Store result in

The location in which to store the number of call events counted for each specified queue; can be a field, multivalued field, or variable name.
Open

The Open command opens the specified queues for retrieval of call events.

Properties:

There are no properties specific to this command.

Notes

This command changes the queue access to open, which allows operators to access the queue. If this command is used on a queue that was previously defined as open by schedule, the queue is opened and the schedule is ignored. The queue access remains open until it is changed either through a QUEUE verb command or through the Queue definition screen.

Open by Schedule

The Open by Schedule command opens queues using the currently defined schedule.

Properties:

There are no properties specific to this command.

Notes

This command changes the queue access to open by schedule, which allows agents to access the queue according to the currently
defined schedule. To set a schedule, use the QUEUE Schedule. The queue access remains opened by schedule until it is changed either through a QUEUE verb command or through the Queue definition screen.

The Queue List command retrieves a list of all defined queues in the project.

**Properties:**

- **Store result in**
  The location in which to store the list of defined queues; can be a multivalued field or variable name.

**Notes**

One common application of this command is to get a list of the queues in a new project, then pass this list to the QUEUE Open to open all queues for the agents. It can also be used to close all the queues in a project.

Protected queues are listed as /U or /G.
**Schedule**

The Schedule command adds or modifies a schedule for a queue.

**Properties:**

- **Schedule start time**
  
  The times to open the queue; can contain up to three start times in the following formats:
  
  - A literal in internal format.
  
  - A field with the data type as date, or any field with the dates stored in internal format.
  
  - A variable containing the dates in internal format.

- **Schedule end time**
  
  The times to close the queue; can contain up to three end times in the following formats:
  
  - A literal in internal format.
  
  - A field with the data type as date, or any field with the dates stored in internal format.
  
  - A variable containing the dates in internal format.
### Notes

Queue access must be set to open by schedule before the Schedule command can add or modify the schedule. To add or change a schedule to a queue which has an open or closed access, first use the QUEUE Open by Schedule.

### Status

The Status command retrieves the current status for the queue.

**Properties:**

- **Open flag**
  - The location in which to store the number representing the status of the queue; can be a field or variable; can be one of the following:
  - 1 Open.
  - 2 Closed.
  - 3 Open by schedule.

- **Schedule start time**
  - The location in which to store the start times for the schedule; can be a field or variable name. This is a required entry. A null is returned if no schedule is defined.

- **Schedule end time**
  - The location in which to store the end times for the schedule; can be a field or variable name. This is a required entry. A null is returned if no schedule is defined.
Notes

The start and end times are returned in internal format.
The QUOTES verb encloses a value within quotation marks.

Properties:

- Double-quote the value: The value to be enclosed in quotation marks; can be a literal, or a field or variable name.
- Store result in: The location in which to store the value; can be a field or variable name.

Notes

The QUOTES verb is useful when you want to place text within quotation marks for a Select statement or to create a string.
The code shown below creates a select statement which includes a value inside quotation marks.

- ALWAYS QUOTES surround the value of "CA" with double quotes store result in @STATE
- ALWAYS APPEND to the end of CUST.LIST the value of @STATE store result in @COMMAND
- ALWAYS EXECUTE command @COMMAND on server

**See Also**

For another method of creating a string, see the CALCULATE verb.
The RANDOM verb generates a random number between 1 and a number you specify.

**Properties:**

- **Random number from 1 to**
  The end number; can be a number, or a field or variable containing the number. The maximum number that can be entered is 32766.

- **Store result in**
  The location in which to store the random number; can be a field or variable.
The READ-RECORD verb gets information stored from a specified record in a user-defined file and places it in the record workspace. Updates to the record are stored in the record workspace.

**Properties:**

- **From the file**  The file to be read; must be a filename. The name is not enclosed in quotation marks.
Properties: (continued)

Read record ID  The ID of the record to be read; can be a literal, or a field or variable containing the ID. In a guide, $CALEVENT can be used to get the current call event. In a task, $ID contains the current record ID being processed.

Store status in  The location in which to store the result of the read; can be a field or variable name. The outcome can be one of the following:

   Y  Indicates the read was successful.
   N  Indicates the read failed.

The result of READ-RECORD is also reflected in $READ.OK as Y when successful and N when unsuccessful.

Locking record  Specifies whether to restrict any other operator from reading the record with a lock; can be one of the following:

   Yes  If the record is not locked by another operator, the record is read and a lock is set. If the record is locked by another operator, the action depends on the setting for the Do not wait for lock property. See Table 16-27.

   You should lock the record when you intend to write the record or restrict the call event from being accessed by another operator.
Notes

If the read lock is not otherwise released (for example, when the agent goes to $END or writes the record), it is released when the agent’s operations guide process dies.

The effects of various READ-RECORD settings on the outcome of the read are listed in Table 16-27. In addition to setting read locks, if you are using an optional EDGE DataBase Link module, you can set transaction locks that affect reading, writing, and deleting records. For
information on transaction locks, refer to the EDGE 7.11 DataBase Link Reference Manual.

The UNIX operating system determines the number of records an operator can have locked at a time. The current maximum for EDGE is 205.

**Table 16-27. READ-RECORD Verb Results**

<table>
<thead>
<tr>
<th>Locking Record Setting</th>
<th>Do Not Wait Setting</th>
<th>Record Is Locked by Another User</th>
<th>Results of READ-RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>The record is not read, $LOCKED is set to Y, $READ.OK is set to N, and the operator does not wait for the record to be unlocked.</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>$LOCKED is set to Y and the operator waits until the lock on the record is released. When released, $READ.OK is set to Y and the record is read with a lock.</td>
</tr>
<tr>
<td>Locking Record Setting</td>
<td>Do Not Wait Setting</td>
<td>Record Is Locked by Another User</td>
<td>Results of READ-RECORD</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------</td>
<td>----------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes or No</td>
<td>No</td>
<td>$READ.OK$ is set to Y and the record is read with a lock.</td>
</tr>
<tr>
<td>No</td>
<td>N/A</td>
<td>Yes or No</td>
<td>$READ.OK$ is set to Y and the record is read without a lock.</td>
</tr>
</tbody>
</table>
The RECORDING verb is an optional verb used with the EDGE Comverse optional module. For more information about the RECORDING verb, see the EDGE 7.11 Comverse Reference Manual.
When invoked, the REFRESH verb immediately updates the EDGE client screen with the current values of any updated screen objects. After the EDGE client screen is updated by the REFRESH verb, the rest of the logic flow is completed. If using multiple screens, all screen objects whose values have changed are updated regardless of which window is active.

Properties:

None.

Notes

The REFRESH verb is not necessary for regular guide processing. Screen objects are automatically updated at the end of any logic flow without the use of this verb. The primary reason you would use this verb is if you wanted to update the screen before the logic flow is
completed. For example, you may want your agents to continue the flow of conversation with your customers instead of pausing to wait for the next question to display on the screen. To display the next question quickly, you could use the REFRESH verb. For more information, see the example below.

The REFRESH verb slows the performance of a logic flow because the server must update the client anytime the REFRESH verb is used.

The REFRESH verb cannot be used to refresh an internet web page, which means it has no effect on EDGE Internet.

**Example**

The REFRESH verb is used with the AdvantEDGE scripting feature to quickly refresh the screen with the next question to ask the customer. In Figure 16-3, a question is displayed on the screen along with an input field for the customer’s response. To quickly display the next question before the logic flow is completed, the REFRESH verb is placed in the logic flow between the question and answer steps. See Figure 16-4 for an example of the logic flow.
The REFRESH verb updates the screen with the new question prior to completing the logic flow. Figure 16-5 is an example of a screen that is updated using the REFRESH verb. Only the question has changed; the previous customer response is still displayed.
At the end of the logic flow, the screen is repainted and the customer’s previous response is cleared from the screen. For an example, see Figure 16-6. For more information on scripting, see the AdvantEDGE Maintenance Manual.

The previous answer is cleared automatically after the logic flow is complete.

Figure 16-5. Screen Update after REFRESH Verb

The next question is displayed using the REFRESH verb.

Figure 16-6. Screen after Completion of Entire Logic Flow
The RELEASE-RECORD verb unlocks a record that was previously read with a lock.

Properties:

- **Release lock from file**
  - The file in which the record exists; must be the filename or file abbreviation. Do not enclose in quotation marks.

- **Record ID**
  - The ID of the record to be released; can be a literal, or a field or variable containing the ID. In a guide, $CALLEVENT can be used to get the current call event. In a task, $ID contains the current record ID being processed.
Properties: (continued)

Store status in

The location in which to store the result of the release process, which is Y if successful or N if not successful; can be stored in a field or variable.

Notes

If you use the WRITE-RECORD verb following the READ-RECORD verb, you do not need to use this verb. The WRITE-RECORD verb automatically releases the lock on the record.

If the release process fails, the system field $DBERRNO will contain an error code. For more information, see Chapter 12, EDGE Files and Fields.
The REMOVE verb deletes all the values that match a specified value from a multivalued field.

**Properties:**

- **Remove the value:** The value of the data to be removed; can be a literal, or a field or variable containing the value.
- **From the multi-values:** The location of the data; can be a multivalued field or variable containing the values.
- **Store result in:** The location in which to store the resulting value; can be a field or variable name.
  
  You can store the changes back to the field or variable in which the original data was stored.
Notes

The REMOVE verb deletes values based on the contents, rather than the position, of the value. The value to be deleted must match the data exactly in order to be deleted. The REMOVE verb will not delete a portion of the value. If there are duplicate values, each value that matches the specified value is deleted. The remaining values after the deletion are moved to occupy the deleted positions.

In the following example, YELLOW is the value to be removed from the original values. The value YELLOW GREEN is not deleted because it is not an exact match of the value YELLOW.

<table>
<thead>
<tr>
<th>Original Values</th>
<th>New Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>YELLOW</td>
<td>RED</td>
</tr>
<tr>
<td>RED</td>
<td>BLUE</td>
</tr>
<tr>
<td>YELLOW</td>
<td>YELLOW GREEN</td>
</tr>
<tr>
<td>BLUE</td>
<td></td>
</tr>
<tr>
<td>YELLOW GREEN</td>
<td></td>
</tr>
</tbody>
</table>

See Also

DELETE-VALUE verb, FIND-VALUE verb, INSERT-VALUE verb.
This verb allows a logic flow called by the CALL-LOGIC verb to return a value to the calling logic. Only a logic flows with a Parameter Definition where the property Returns value is checked can use this verb.

**Properties:**

| Value to Return | The value to return to the calling logic flow. This can be a literal, a field or a variable. This value is stored in the variable specified by the Store status code in property of the CALL-LOGIC verb. |
The ROLL verb moves values in a multivalued field to the next value position and adds a value at the first position.

**Properties:**

- **Roll the multivalues**
  
  The location of the values to be rolled; can be a multivalued field or variable name.
  
  You cannot specify a value position with the field or variable.

- **Add the value**
  
  The value to be added to the first position; can be a literal, or a field or variable containing the value.
The ROLL verb can be used, for example, to store a history of information. The first position could contain the most current data and the last position could be removed each time the value was updated.

For examples of ROLL actions, see Table 16-28. The results are based on values in @VALUE. Note that when position 0 (zero) is removed, the entire value is removed and the specified value put in value position 1.

**Table 16-28. ROLL Verb Examples**

<table>
<thead>
<tr>
<th>Value Position</th>
<th>@VALUE</th>
<th>Add &quot;X&quot; Removing Position 0</th>
<th>Add &quot;X&quot; Removing Position 1</th>
<th>Add &quot;X&quot; Removing Position 3</th>
<th>Add &quot;X&quot; Removing Position 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>B</td>
<td>A</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>C</td>
<td>B</td>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>4</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>C</td>
</tr>
</tbody>
</table>
### Table 16-28.ROLL Verb Examples (continued)

<table>
<thead>
<tr>
<th>Value Position</th>
<th>@VALUE</th>
<th>Add &quot;X&quot; Removing Position 0</th>
<th>Add &quot;X&quot; Removing Position 1</th>
<th>Add &quot;X&quot; Removing Position 3</th>
<th>Add &quot;X&quot; Removing Position 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
<td>E</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td>E</td>
<td>E</td>
<td>D</td>
</tr>
</tbody>
</table>

**See Also**

REMOVE, DELETE-VALUE, FIND-VALUE, and INSERT-VALUE verbs.
The ROUTE verb moves the agent’s cursor to an input field on the same or a different screen.

Properties:

Route to screen

The name of the window, screen, and object to be routed to; can be blank, a literal, or a field or variable containing the screen name.

If blank, the current screen is routed to, and after the route is done, the logic flow continues to the next statement. If a screen name is specified, after the route is done, the logic flow is exited.
## Properties: (continued)

<table>
<thead>
<tr>
<th>Field</th>
<th>A Window Helper screen is available to assist with the syntax of this property. Select the three-dot button to display the Window Helper screen. The name of the input field to route to; can be a literal, a field or variable containing the field name, or blank. If blank or an invalid field name is used, the cursor moves to the first field on the specified screen. To route to a multivalued field, include the position number, for example, &quot;COMMENT,4&quot;. This property is the same as the Object name property found on the Window Helper screen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setting result to</td>
<td>The result to be set; can be the result name without quotation marks or blank. This property is optional.</td>
</tr>
</tbody>
</table>
The SCREEN-SAVER verb is no longer required. It is retained for backward compatibility.
The SEND-TO-VOAD verb is no longer supported. It is retained for backward compatibility.
The SET-PRINTER verb changes the printer assignment while logged on.

**Properties:**

- **Set to printer**
  The name of the EDGE printer to be assigned; can be a literal, or a field or variable containing the printer name.
  
  To find the names of printers at your site, select EDGE, select the Server area, then click the Printers tab. For more information, see Chapter 3.

- **Number of copies**
  The number of copies to be printed; can be a number, or a field or variable containing the number.
Notes

In operations, the SET-PRINTER verb can change the setting for the default printer.

In tasks, the SET-PRINTER verb can change the default printer for tasks.

See Also

For more information about EDGE printers, see Chapter 3.
The SET-PROPERTY verb allows you to temporarily set some of the properties for an object on a screen.
Properties:

For object: The name of the object whose properties are to change; can be a literal, or a field of variable containing the object name.

To specify the object name, use the following format:

windowPart : screenName : objectname

The colons are required as placeholders, even if just the objectname is being specified. For example, to select the object OBJ1 on the current screen, enter the following:

"::OBJ1"

To select the specified object on every window or screen, use an asterisk as the window name or screen name. For example to select the object OBJ2 on every window and screen, enter the following:

"*:OBJ2"

To set one value position in a multivalued object, specify the value position, separated by a comma; for example, "::OBJ3,2". If the value position is not entered, all the value positions are set with the new properties.

A Window Helper screen is available to assist with the syntax of this property. To display the Window Helper screen select the three-dot button.
**Properties: (continued)**

If a window or screen is not specified, only the properties of the object on the active window and screen are changed.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Visible</strong></td>
<td>Specifies whether the object is to be made visible, not visible, or is not to be changed; can be a number, or a field or variable containing the indicator. One of the following can be specified:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Blank</td>
<td>No change. This is the default.</td>
</tr>
<tr>
<td>No (0)</td>
<td>The object is not visible.</td>
</tr>
<tr>
<td>Yes (1)</td>
<td>The object is visible on the screen.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enabled</strong></td>
<td>Specifies whether the object is to be enabled, disabled, or not changed; can be a number, or a field or variable containing the value. One of the following can be specified:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Blank</td>
<td>No change. This is the default.</td>
</tr>
<tr>
<td>No (0)</td>
<td>The object is disabled and cannot be accessed. It is displayed as a grayed object.</td>
</tr>
<tr>
<td>Yes (1)</td>
<td>The object is enabled and the operator can move to the object.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tab stop</strong></td>
<td>Specifies whether to set a tab stop for the object; can be a number, or a field or variable containing the indicator; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blank</td>
<td>No change. This is the default.</td>
</tr>
<tr>
<td>No (0)</td>
<td>The tab stop is removed. When the tab key is pressed, it bypasses objects without tab stops. To select the object, the agent must use the mouse.</td>
</tr>
<tr>
<td>Yes (1)</td>
<td>A tab stop is set for the object. When the tab key is pressed, it stops at each object which is set with a tab.</td>
</tr>
<tr>
<td>Left</td>
<td>The first screen column to be used by the object; can be a number, or a field or variable containing the column. To indicate no change, leave the property blank.</td>
</tr>
<tr>
<td>Top</td>
<td>The first screen row to be occupied by the object; can be a number, or a field or variable containing the row. To indicate no change, leave the property blank.</td>
</tr>
<tr>
<td>Width</td>
<td>The number of screen columns used by the object; can be a number, or a field or variable containing the value. To indicate no change, leave the property blank.</td>
</tr>
<tr>
<td>Height</td>
<td>The number of rows to be used by the object; can be a number, or a field or variable containing the number. To indicate no change, leave the property blank.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Style</strong></td>
<td>The style to be applied to the object. For more information on styles, see the section <strong>Styles</strong>.</td>
</tr>
<tr>
<td><strong>Checked</strong></td>
<td>Specifies whether to check the check box when the object is a check box; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Blank No change. This is the default.</td>
</tr>
<tr>
<td></td>
<td>No (0) Does not put a check in the check box.</td>
</tr>
<tr>
<td></td>
<td>Yes (1) Puts a check in the check box.</td>
</tr>
<tr>
<td><strong>Separator before</strong></td>
<td>Specifies whether to place a separator before the menu item; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Blank No change. This is the default.</td>
</tr>
<tr>
<td></td>
<td>No (0) Does not put a separator before the item.</td>
</tr>
<tr>
<td></td>
<td>Yes (1) Puts a separator before the item.</td>
</tr>
<tr>
<td><strong>Separator after</strong></td>
<td>Specifies whether to place a separator after the menu item; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Blank No change. This is the default.</td>
</tr>
<tr>
<td></td>
<td>No (0) Does not put a separator after the item.</td>
</tr>
<tr>
<td></td>
<td>Yes (1) Puts a separator after the item.</td>
</tr>
<tr>
<td><strong>Properties:</strong> (continued)</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Border style</strong></td>
<td></td>
</tr>
<tr>
<td>The border style to be applied to the object; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td>Blank</td>
<td></td>
</tr>
<tr>
<td>No change. This is the default.</td>
<td></td>
</tr>
<tr>
<td>Simple (1)</td>
<td></td>
</tr>
<tr>
<td>The top and left borders are black and the bottom and right borders are white, which gives a 3-dimensional look.</td>
<td></td>
</tr>
<tr>
<td>Standard (0)</td>
<td></td>
</tr>
<tr>
<td>Puts a black border around the box.</td>
<td></td>
</tr>
<tr>
<td><strong>Title text</strong></td>
<td></td>
</tr>
<tr>
<td>The text to be used for the object.</td>
<td></td>
</tr>
<tr>
<td><strong>Title alignment</strong></td>
<td></td>
</tr>
<tr>
<td>The title text alignment; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td>Center</td>
<td></td>
</tr>
<tr>
<td>Centers the text.</td>
<td></td>
</tr>
<tr>
<td>Hidden</td>
<td></td>
</tr>
<tr>
<td>Hides the text.</td>
<td></td>
</tr>
<tr>
<td>Left</td>
<td></td>
</tr>
<tr>
<td>Aligns the text to the left.</td>
<td></td>
</tr>
<tr>
<td>Right</td>
<td></td>
</tr>
<tr>
<td>Aligns the text to the right.</td>
<td></td>
</tr>
<tr>
<td><strong>Border width</strong></td>
<td></td>
</tr>
<tr>
<td>The width of the line used to border a group box; can be a number or a field or variable containing the number.</td>
<td></td>
</tr>
<tr>
<td><strong>ActiveX property name</strong></td>
<td></td>
</tr>
<tr>
<td>The property name of the ActiveX control to be set; must be a literal containing the name of the ActiveX property. The name must be in the same case and enclosed in quotation marks, for example:</td>
<td></td>
</tr>
<tr>
<td>&quot;LineStyle&quot;</td>
<td></td>
</tr>
</tbody>
</table>
**Properties: (continued)**

Property names are specified as part of Configuring ActiveX Controls.

ActiveX property value

The value you want to assign to the property of the ActiveX control; must be a literal containing the value for the ActiveX property. The value must be enclosed in quotation marks.

Insert expressions

Specifies whether expressions can be embedded into the input field; can be one of the following:

- **Blank**
  - No change.
  - This is the default.

- **No (0)**
  - Expressions cannot be embedded into the input field.
  - Disables the Insert Expression command on the menu displayed when you right-click in an input field in EDGE Client. It also disables the Insert In Field command in the Expression Setup screen in EDGE Client. For more information, see the EDGE 7.11 Client Reference Manual.
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expressions can be</td>
<td>Yes (1)</td>
<td>Embedded into the input field. Enables the Insert Expression command on the menu displayed when you right-click in an input field in EDGE Client. It also enables the Insert In Field command in the Expression Setup screen in EDGE Client. For more information, see the EDGE 7.11 Client Reference Manual.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Define expressions</td>
<td></td>
<td>Specifies whether expressions can be defined, that is, created, modified, and deleted in EDGE Client; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Blank</td>
<td></td>
<td>No change. This is the default.</td>
</tr>
<tr>
<td>No (0)</td>
<td></td>
<td>Expressions cannot be defined. Removes the Insert Expression command from the menu displayed when you right-click in an input field in EDGE Client. It also disables the Add, Modify, and Delete commands in the Expression Setup screen in EDGE Client. For more information, see the EDGE 7.11 Client Reference Manual.</td>
</tr>
</tbody>
</table>

**Properties: (continued)**
### Properties: (continued)

<table>
<thead>
<tr>
<th>Spell check</th>
<th>Specifies the mode to set the spelling checker, automatic or manual; can be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Blank</td>
<td>No change. This is the default.</td>
</tr>
<tr>
<td>No (0)</td>
<td>The spelling checker is in manual mode and must be invoked by right clicking anywhere in the input field in EDGE Client and then selecting the Spell check command.</td>
</tr>
<tr>
<td>Yes (1)</td>
<td>The spelling checker is in automatic mode, thus a space invokes it.</td>
</tr>
</tbody>
</table>

For more information, see the EDGE 7.11 Client Reference Manual.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text color</td>
<td>The foreground color for the enabled object text; can be blank, a comma-delimited string, or a field or variable. If you leave this property blank, the value defaults to the text color defined in the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.</td>
</tr>
<tr>
<td>Background color</td>
<td>The background color for the enabled object; can be blank, a comma-delimited string, or a field or variable. If you leave this property blank, the value defaults to the background color defined in the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.</td>
</tr>
<tr>
<td>Disabled text color</td>
<td>The foreground color for the disabled object text; can be blank, a comma-delimited string, or a field or variable. If you leave this property blank, the value defaults to the disabled text color defined in the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.</td>
</tr>
<tr>
<td>Disabled background color</td>
<td>The background color for the disabled object; can be blank, a comma-delimited string, or a field or variable. If you leave this property blank, the value defaults to the disabled background color defined in the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.</td>
</tr>
</tbody>
</table>
Properties: (continued)

If you leave this property blank, the value defaults to the disabled background color defined in the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.

Font
The font name, size, and style for bold, italic, underlined, and strikeout; can be blank, a comma-delimited string, or a field or variable.

If you leave this property blank, the value defaults to the font defined in the current style.

Border color
The border color for the object frame; can be blank, a comma-delimited string, or a field or variable.

If you leave this property blank, the value defaults to the border color defined in the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.

Store status in
The location in which to store the resulting status code; can be a field or variable. The result value is always single-valued, even if multiple objects are specified in the verb. This means that if three objects are named, and one is not found, the result is 1, but there is no indication as to which one is missing. Also, if two are missing, the result would still be 1.

The status codes are listed in Table 16-29.
Notes

To specify that the actions are to be held until the logic flow is exited, the environment variable, bufferSetProperty, must be set to 1 and exported before the server starts; for example, the following can be included in EDGE’s .user.profile:

```
bufferSetProperty=1; export bufferSetProperty
```

You can override the setting of the environment variable by prefixing the object names with one of the following:

- `+` Buffers this SET-PROPERTY action until the next nonbuffered SET-PROPERTY verb or until the logic flow is exited. Even if the bufferSetProperty environment variable is not set, this buffers the actions.

- `!` Sends this and all previously buffered SET-PROPERTY actions to the client immediately. Even if the bufferSetProperty environment variable is set, this sends the actions.

Examples:

In the following examples, bufferSetProperty environment variable is not set. When the second SET-PROPERTY logic flow is executed, all the actions are sent to the client.

```
SET-PROPERTY of object name "+A" visible 0
SET-PROPERTY of object name "C" tab stop 0
```
In the following example, bufferSetProperty environment variable is set. When the second SET-PROPERTY logic flow is executed, all the actions are sent to the client.

```
SET-PROPERTY of object "A" visible 0
SET-PROPERTY of object "!B" tab stop 0
```

The following rules apply to this verb:

- You can set multiple properties for an object at one time. Any property left blank retains the value currently set for that property.

- Modifications to properties made with the SET-PROPERTY verb are temporary and only affect the objects on the windows and screens on which they are set. When the screen is closed, the settings return to the default settings, that is, to the settings specified when the screen was developed.

- The SET-PROPERTY verb can be used to set properties in a logic flow attached at Entry to Screen which is performed before the screen is displayed.

- Modifying the row and column position of an object does not affect its position in the tabbing order.

- If an object with a tab stop is set to an off screen position by altering the row and column position, it will still be tabbed to even though it does not appear on the screen.
- Input entered for an input field object with a width shorter than the number of characters in the associated field will scroll within the object window.

**Table 16-29.** SET-PROPERTY Verb Status Codes

<table>
<thead>
<tr>
<th>Result</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful.</td>
</tr>
<tr>
<td>1</td>
<td>Invalid object name. No object could be found with the specified name on the current screen.</td>
</tr>
<tr>
<td>2</td>
<td>Invalid property type found.</td>
</tr>
<tr>
<td>3</td>
<td>Invalid style type found.</td>
</tr>
<tr>
<td>4</td>
<td>Object name is blank.</td>
</tr>
<tr>
<td>5</td>
<td>Invalid EDGE property name.</td>
</tr>
<tr>
<td>-1</td>
<td>Verb not implemented in ASCII operations.</td>
</tr>
</tbody>
</table>

**See Also**

For more information on using objects and properties, see Chapter 20, Screen Objects.
SHOW-MESSAGE

The SHOW-MESSAGE verb displays a message to an operator.

Properties:

Show the message in

The message to be displayed; can be a literal, or a field or variable containing the message.

Notes

The message can have up to 66 characters and is displayed at the bottom of the screen on the message line. The message can be used, for example, to show a message related to a specific field.

The message is displayed until one of the following occurs:

- The operator routes to another screen.
• The screen is repainted.
• Another message with null (""") is sent to clear the previous message.

See Also

The ASK verb.
SLEEP

The SLEEP verb pauses the logic flow or task by a specified number of seconds.

Properties:

Seconds to sleep  The number of seconds to sleep (pause); can be a number, or a field or variable containing the number.

If at runtime the value in the field or variable is not a number, SLEEP is ignored.
The SORT-VALUES verb sorts the contents of a multivalued field.

**Properties:**

- **Data value field**
  
  The location of the dependent field to be sorted with the sort value field; must be a field or variable.

  If there is no dependent field, enter the same field or variable as for the sort value field. See the section [Dependent Field](#) for more information on dependent fields.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort value field</td>
<td>The location of the values to be sorted; can be a multivalued field or variable.</td>
</tr>
<tr>
<td>Sorting order</td>
<td>The order to sort based on the ASCII value; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>- Ascending: A to Z.</td>
</tr>
<tr>
<td></td>
<td>- Descending: Z to A.</td>
</tr>
<tr>
<td>Data value type</td>
<td>The type of data in the field, which when combined with the sort order specifies how the data will be sorted; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>- Alphanumeric: Sorts letters in alphabetic order by ASCII value. When the order is ascending, numbers appear before letters and uppercase letters come before lowercase letters. Numbers sort from left to right. For example, 9.78 is considered larger than 34.55.</td>
</tr>
<tr>
<td></td>
<td>- Floating point: Sorts numbers based on the decimal number. For example, 5.55 is larger than 5.44.</td>
</tr>
</tbody>
</table>
Properties: (continued)

Integer

Sorts based on the greater integer value (whole number) only. Does not consider decimal value in sort. Example: 5.55 and 5.44 are viewed as 5. Negative numbers are sorted, but decimal places are truncated. For example, -1.34 is considered -1.

Trailing options

The options for trailing values when the dependent and sort fields do not have the same number of values; can be one of the following:

Fill

Add blank trailing values to fields with fewer values.

Ignore

Ignore extra trailing values at the end of the dependent field if it contains more values than the sort field. If the sort field contains more values than the dependent field, the dependent field is filled if required by the sort order.

Remove

Remove extra trailing values at the end of the dependent field if it contains more values than the sort field. If the sort field contains more values than the dependent field, the dependent field is filled if required by the sort order.
Dependent Field

A dependent field is a multivalued field or variable that is associated with the multivalued field or variable being sorted. For example, you may store product numbers in PROD.NUM and product names in PROD.NAME. Each multivalued product name is associated with the product number in the corresponding value position. Since the number and name are related, they must remain in the same position in the multivalued field. If you sort PROD.NUM, you must identify PROD.NAME as the data value field. If you do not, the product number and product name may not match.
The SOUNDEX verb converts a string to its soundex equivalent.

Properties:

- Compute soundex on: The string to convert to its soundex equivalent; can be a literal, or a field or variable containing the string.
- Store result in: The location in which to store the soundex equivalent; can be a field or variable.

Notes

The soundex format enables a search based on phonetic sounds, rather than spelling. Soundex converts letters in a string to a coded equivalent.
The following rules apply to soundex:

- The first alphabetic character in the string becomes the first character of the resulting code.
- Each letter following the first letter is checked to determine if it is a consonant. If it is, the corresponding soundex value is appended to the code.
- A soundex code contains four characters; it is zero-filled on the right if fewer than four characters result from the conversion.
- More than one letter may convert to the same soundex value.
- Consecutive duplicate sounds, such as tt or ss, are converted to a single number.
- Vowels not in the first position and the letters H, W, and Y, and all non-alphabetic characters are ignored.

Table 16-30 lists the values used by EDGE to produce soundex codes.

**Table 16-30. Soundex Values**

<table>
<thead>
<tr>
<th>Letter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, E, I, O, U, H, W, Y</td>
<td>null</td>
</tr>
<tr>
<td>B, F, P, V</td>
<td>1</td>
</tr>
<tr>
<td>C, G, J, K, Q, S, X, Z</td>
<td>2</td>
</tr>
</tbody>
</table>
Examples

The following are examples of soundex conversions:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLEN</td>
<td>A450</td>
</tr>
<tr>
<td>ANDERSON</td>
<td>A536</td>
</tr>
<tr>
<td>ANDREWS</td>
<td>A536</td>
</tr>
<tr>
<td>JAMES</td>
<td>J520</td>
</tr>
<tr>
<td>JONES</td>
<td>J520</td>
</tr>
<tr>
<td>JOHNSON</td>
<td>J525</td>
</tr>
<tr>
<td>JOHNSTON</td>
<td>J523</td>
</tr>
<tr>
<td>SMITH</td>
<td>S530</td>
</tr>
<tr>
<td>SMYTH</td>
<td>S530</td>
</tr>
</tbody>
</table>

See Also

SPECIAL-SOUNDEX verb.
The SPECIAL-SOUNDEX verb composes a soundex-type combination from a field and character that you specify.

**Properties:**

- **Compute soundex on**
  - The name of the field on which to apply soundex; can be the field or variable containing the value to be converted to soundex.

- **Extract first char from**
  - The value containing the character to be added to the soundex value; can be a field or variable containing the character. Only the first character in the field is extracted.
Properties:  (continued)

Store result in  The location in which to store the combined value of the first character and the soundex value; can be a field or variable.

Notes

For example, with the SPECIAL-SOUNDEX verb, you can combine the first letter of a first name with the soundex version of the last name, to search by last name and first initial.

See Also

The SOUNDEX verb.
The SQL-GENERAL verb allows you to execute an SQL statement that is not a SELECT statement. For more information, see the EDGE 7.11 DataBase Link Reference Manual, which is available separately.
The SQL-SELECT (formerly called MULTI-RECORD-READ) verb allows you to specify a select query to read data from relational database tables or c-tree files.
When you select a command, additional properties are activated. For information on these properties, see the description of each command on the following pages.

The SQL-SELECT verb provides an efficient means of retrieving multiple records concurrently. For example, if you are doing scheduling and need to look at a week’s schedule, where each day is a record, you can retrieve all seven days by issuing a single request using SQL-SELECT. Depending on your operation, this could improve performance significantly.
SQL-SELECT statements can access tables without primary keys.

The statement you use in the SQL-SELECT Open Query command to select data can contain any valid SQL report commands if you are using DataBase Link tables, or any valid EQL statement if you are using c-tree files. For example, if you are using a DataBase Link product, you can select fields from more than one table by specifying a join, then sort the selected records, if desired. The resulting records can be retrieved using the Get Records command.

**Note:** All data that matches the specified SQL or EQL statement is retrieved at the time the command is invoked. Depending on the complexity of the statement and the size of your database, this may have an effect on performance and disk space.

**Organization**

The data that is returned by the SQL-SELECT Get Records command can be organized in one of two ways:

- Horizontally.
- Vertically.

This organization is specified when you open the query.
Horizontal organization returns the data as a single string, with each record in the string delimited by a user-defined character. The fields within each record are delimited by an EDGE system delimiter called an attribute mark, which has an ASCII decimal value of 030. Each Get Records command returns as many delimited records as specified in the Return Record Count field in the Open Query command.

When you use the Get Records command to retrieve records organized horizontally, because each record consists of multiple records with multiple fields, it is recommended that you return the records in a variable, rather than a field. You can then use the **GROUP-EXTRACT** verb to get each record and copy it to a file record. For more information, see the section **Moving Records into Fields**.

Vertical organization returns the data as a single record with multivalued fields. Each Get Records command returns one record, with the number of multivalues in each field as specified in the Return Record Count field in the Open Query command.

Vertical organization is intended to be used with fields that are single-valued.

**Example**

The following example demonstrates the data that is returned by each method of organization.
Assume the query results in the following data:

<table>
<thead>
<tr>
<th>Row</th>
<th>Name</th>
<th>City</th>
<th>Weather</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Lee</td>
<td>Boston</td>
<td>Sunny</td>
</tr>
<tr>
<td>2</td>
<td>Jackson</td>
<td>Chicago</td>
<td>Windy</td>
</tr>
<tr>
<td>3</td>
<td>Silva</td>
<td>Los Angeles</td>
<td>Sunny</td>
</tr>
<tr>
<td>4</td>
<td>Jones</td>
<td>New York</td>
<td>Snowy</td>
</tr>
<tr>
<td>5</td>
<td>Smith</td>
<td>New York</td>
<td>Cloudy</td>
</tr>
</tbody>
</table>

In the examples, a caret (^) represents the end of a field. A right bracket (]) represents the end of a value in a multivalued field.

**Case 1**

The Open Query command specifies the data is organized horizontally, with a record count of 3, and a separator of ;.

The first Get Records command returns the following record:

Lee^Boston^Sunny;Jackson^Chicago^Windy;Silva^Los Angeles^Sunny;

The Store result in property contains a 3.

The next Get Records command returns the following:
Jones^New York^Snowy;Smith^New York^Cloudy;

The Store result in property contains a 2.

**Case 2**

Data is organized vertically with a record count of 3.

The first Get Records command returns the following record:


The Store result in property contains a 3.

The next Get Records command returns the following:


The Store result in property contains a 2.

**Moving Records into Fields**

You can move records into fields as follows:

1. Create a file with field definitions that match the fields you are retrieving.

2. Use Open Query to select the records.
3. Use Get Records to retrieve the records. Store them in variable1.

4. If you are using vertical records, skip to step 5. If you are using horizontal organization, use the GROUP-EXTRACT verb to extract a record from the variable in step 3. Store that record in variable2. (If the separator character is unprintable, for example, you specified \010 in the Open Query, you must first use the CONVERT verb to create a variable with the unprintable character.)

5. Using the COPY verb, copy variable1 if vertical or variable2 if horizontal into the fields for the file you defined in step 1 as follows:

   COPY variable INTO =REC(filename)

6. After each copy, write the record either using an ID that was selected as part of the select query statement or one that you specify.

   The syntax =REC(filename) copies the variable into a record in the specified file. For more information, see the COPY verb.

   Alternatively, you can specify the name of the first field as the destination of the Get Records command if you are using vertically organized records, or as the destination of the GROUP-EXTRACT verb if you are using horizontally organized records. If there is more than one field in the returned record, the addi-
tional fields are inserted in the fields following the specified field. You can then reference them using those field names. For example, if your file contains three consecutive fields named Name, City, and Weather, and you put your records into the Name field, the first returned field is placed in Name, the second returned field is placed in City, and the third returned field is placed in Weather. (To ensure there is no data in the destination record, use the `CLEAR-RECORD` verb.)

**Note:** If you insert multiple fields into a field and there is data in the fields beyond the insertion location, the data in those fields is moved down and can no longer be referenced by the original field names.

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Successful operation.</td>
</tr>
<tr>
<td>-101000</td>
<td>Soft connection error. The process can be tried again.</td>
</tr>
<tr>
<td>-101001</td>
<td>Hard connection error. The exact error is in the UNIX stderr file.</td>
</tr>
<tr>
<td>-103002</td>
<td>The query is not a SELECT statement.</td>
</tr>
<tr>
<td>-103003</td>
<td>The specified record format is not horizontal or vertical.</td>
</tr>
<tr>
<td>-103004</td>
<td>The specified record size is less than or equal to zero.</td>
</tr>
<tr>
<td>-103005</td>
<td>The specified separator has a value that is &lt;= 0 or &gt; 255.</td>
</tr>
</tbody>
</table>
### Table 16-31. SQL-SELECT Status Codes (continued)

<table>
<thead>
<tr>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-103006</td>
<td>The specified operation is not: Open Query, Get Records, or Close Query.</td>
</tr>
<tr>
<td>-103007</td>
<td>The query name was not specified.</td>
</tr>
<tr>
<td>-103008</td>
<td>A separator character was not specified.</td>
</tr>
<tr>
<td>-103009</td>
<td>The DB_DEVICE was not specified.</td>
</tr>
<tr>
<td>-103010</td>
<td>The query is not a SELECT statement.</td>
</tr>
<tr>
<td>-103011</td>
<td>The specified record format is not horizontal or vertical.</td>
</tr>
<tr>
<td>-103012</td>
<td>The specified query name is too long. The maximum is 30 characters.</td>
</tr>
<tr>
<td>-103020</td>
<td>There is not enough memory for the specified operation.</td>
</tr>
<tr>
<td>-103021</td>
<td>The specified query is not open.</td>
</tr>
<tr>
<td>-103023</td>
<td>No items were selected.</td>
</tr>
<tr>
<td>-103024</td>
<td>The open was unable to connect to the specified database device ID.</td>
</tr>
<tr>
<td>-103025</td>
<td>The maximum number of queries are already open. The maximum is 10.</td>
</tr>
<tr>
<td>-103026</td>
<td>The specified query is already open.</td>
</tr>
</tbody>
</table>
### Close Query

The Close Query command closes a specified SQL-SELECT Open query, and frees the memory associated with it.

**Properties:**

There are no other properties specific to this command.

### Get Records

The Get Records command retrieves the next set of records selected by the Open Query command. Typically this command is used repeatedly until all records have been returned. See the SQL-SELECT Example Logic below. The number of records retrieved by each invocation and their format depends on properties specified in the Open Query command.

**Properties:**

- **Store result in**: The location in which to store the result of the command; can be a field or variable. The result can be one of the following:
  - 0: All selected records have been processed.
  - <0: An error. For a status codes, see Table 16-31.
  - >0: The number of records the Get Records command returned. If this value is less than the value specified in the Return Record Count property in the Open Query command, there are no more records to be retrieved and no invocations of this command are required.
## Open Query

The Open Query command establishes a database connection with a specified database and selects records that meet the specified criteria. To retrieve the selected records, use the Get Records command.

### Properties:

- **Database device ID**
  - The name of the database device to use; can be a literal, or a field or variable containing the name. The name must refer to a database device; if not, a runtime error occurs. For a c-tree database, specify “Ctree”.
  - For information on specifying a database device ID, see the documentation for appropriate EDGE Database Link, which is available separately.

- **Query criteria**
  - The SQL statement that specifies the selection criteria for the query; can be a literal, or a field or variable containing the statement.

- **Return record count**
  - Specifies the number of records to be retrieved with the Get Records command; can be a number, or a field or variable containing the number. This property is used to set the number of records returned each time the Get Records command is called. For example, if this property is set to 25 and the query returns 105 records, 5 invocations of the Get Records command would be required to retrieve all data. The number of records actually returned by each invocation is stored in the Store Status in property of the Get Records command.
Properties: (continued)

Organization flag

Specifies the organization of the data returned by the Get Records command; can be a literal, or a field or variable containing the indicator. The indicator can be one of the following:

"H" Horizontal organization.

"V" Vertical organization.

Record separator

Specifies the record separator (delimiter) to be used to delimit multiple records, when more than one record meets the query criteria; can be a literal, the decimal ASCII value of a character in the range 1-255, preceded by a backslash (\nnn), or a field or variable containing the delimiter.

The ASCII values 0-31 and 252-254 may cause problems and should be avoided.

This property is used only with horizontal organization.

Notes

Up to ten queries can be opened at one time. Queries remain open until one of the following occurs:

- The Close Query command is executed.
- The agent exits the guide.
• The guide is swapped.
• The task ends.

Queries on c-tree files often include double quotes; use the QUOTES verb to insert the quotation marks. For example, in the query
SELECT
FROM CUST WHERE PHONE = "714\]", three lines would be required, an APPEND command with the query up to the "'s, then a QUOTES command for the "714]" portion, finally a second APPEND command to join the 714] to the rest of the statement.

**SQL-SELECT Example Logic**

The following logic is an example of how the SQL-SELECT Open Query and Get Records commands can be used to retrieve all the records (25 at a time) from the EMPLOYEE file.
The SQUARE-ROOT verb performs the square root function.

**Properties:**

- **Take the square root of**: The number whose square root is to be determined; can be a number, or a field or variable containing the number.

- **Store result in**: The location in which to store the result of the calculation; can be a field or variable.
  
  The number to be calculated must be a positive number. If the value is zero or less, the return value is zero.
See Also

CALCULATE verb.
The SR.TRANS (Sales Reporting Transaction) verb copies data from user-defined fields to related fields in the REPORTING file, when the property Gather sales analysis data is checked in Project Options.

**Properties:**

- **Product ID (or "NA")**
  - The product identification; can be a field or variable containing the product ID, or "NA" (not applicable) when there is no product ID.
  - This value updates the PRODUCTS field in the REPORTING file.
  - If NA is entered as the product ID, the PRODUCTS, QTYS, and AMOUNT fields in the REPORTING file are not updated.
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantity</td>
<td>The value representing the quantity sold; can be a number, or a field or variable containing the quantity. This value updates the QTYS and TOTAL.QTY fields in the REPORTING file.</td>
</tr>
<tr>
<td>Amount</td>
<td>The value representing the amount sold; can be a number, or a field or variable containing the amount. This value updates the AMOUNT and TOTAL.AMT fields in the REPORTING file.</td>
</tr>
</tbody>
</table>

**Notes**

This verb allows you to save product, quantity, and amount information to the REPORTING file. To begin gathering data at the start of the project, check the option to gather sales analysis data before the project goes live. For more information, see Chapter 2. To produce Sales Analysis reports, you must define an ad hoc report.

Logic flows using the SR.TRANS verb are usually attached as end of guide logic when the transaction has ended with a confirmed sale.
The STRING-LENGTH verb determines the number of characters in a string and stores the number.

Notes:
All characters, including, alphanumeric characters, symbols, ASCII characters, and spaces are counted.

Properties:
- Always Copy the value "SELECT NAME, ADDRESS FROM EMPLOYEE" into @QUERY
- Retrieve 25 records at a time until all records are retrieved
- ALWAYS SQL-SELECT query name "Query1" command OPEN QUERY database device ID "Oracle" query criteria @QUERY return record count 25 organization "H" record separator "%" store status in TEMP-OpenStat
- Loop Command Repeat
- ALWAYS SQL-SELECT query name "Query1" command GET RECORDS store result in @RECORDS store status in TEMP-RecCount
- Process partial result in @RECORDS.
The SUBTRACT verb computes the difference between two numbers and stores the result.

**Properties:**

- **Subtract the value:** The number to be subtracted; can be a number, or a field or variable containing the number.
- **From the value:** The number from which the first number is subtracted; can be a number, or a field or variable containing the number.
- **Store result in:** The location in which to store the result of the subtraction; can be a field or variable. The result can be stored back to either of the fields or variables which contained the numbers used in the subtraction.
Notes

If the result is a negative number, it is returned as a negative. If the number includes decimal places, the result includes decimal places.

See Also

CALCULATE verb.
SUM

The SUM verb adds the numbers in a multivalued field and stores the total.

Properties:

- Sum the values in: The values to be added; can be a multivalued field or variable.
- Store result in: The location in which to store the value total; can be stored in a field or variable.

Notes

If the result is a negative number, it is returned as a negative. If the number includes decimal places, the result includes decimal places.
See Also

ADD and CALCULATE verbs.
The TEKNEKRON-COMPLETE verb indicates the call has been completed.

**Properties:**

- **From workstation**  The identifier of the workstation completing the call; can be a number, literal, or a field or variable containing the identifier.
Properties:  (continued)

- **Campaign code**: The Teknekron campaign code; can be the code, or a field or variable containing the code.
- **Completion code**: The completion code; can be the code, or a field or variable containing the code.
- **Quantity value**: The location of the quantity value; can be a field or variable containing the quantity. This property is optional.
- **Amount value**: The location of the amount value; can be a field or variable containing the amount. This property is optional.
- **Store status code in**: The location in which to store the error number, if any; can be stored in a field or variable.
- **Store status text in**: The location in which to store the error text, if any; can be stored in a field or variable.

**Notes**

This verb is optional and only functional with the Teknekron option.

For more information, contact [EDGE Customer Care](#).
The TEKNEKRON-DIAL verb identifies information needed to dial the call.

**Properties:**

- **From workstation:** The identifier of the workstation initiating the call; can be a number, a literal, or a field or variable containing the identifier.
- **Phone number:** The location of the phone number; can be a field or variable.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Campaign code</td>
<td>The campaign code for the call; can be a code, or the field or variable containing the code.</td>
</tr>
<tr>
<td>Store status code in</td>
<td>The error number, if any; can be stored in a field or variable.</td>
</tr>
<tr>
<td>Store status text in</td>
<td>The error message, if any; can be stored in a field or variable.</td>
</tr>
<tr>
<td>Phone length</td>
<td>The number of digits in the phone number; must be a number. The number cannot include symbols such as a dash (-) or slash (/).</td>
</tr>
</tbody>
</table>

### Notes

This verb is optional and only functional with the Tekneekron option.

For more information, contact **EDGE Customer Care**.
The TELEPHONY verb is an optional verb. For more information about Telephony, see the Telephony documentation, which is available separately.
The TERMINATE-TASK verb stops the processing of a task.

Properties:
None.

Notes
When this verb is encountered in a task, the following occurs:

- The task terminates immediately at the line on which the TERMINATE-TASK statement occurs. None of the statements following this line are executed.

- If the task specifies ad hoc reports or selection lists to be run after the task, the reports and selection lists are run.

- If the task specifies a task to chain to, that task is run.
See Also

For more information about tasks, see Chapter 15.
The TEXT-EXTRACT verb extracts a selected number of characters and copies them to a specified location.

**Properties:**

- **Text extract from:** The location from which to extract the characters; can be a field or variable or a multivalued field or variable containing the value.

- **Starting at char position:** The position number of the character at which to begin the extract; can be a number, or a field or variable containing the position number.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num of chars to extract</td>
<td>The number of characters to extract; can be a number, or a field or variable containing the number of characters to extract.</td>
</tr>
<tr>
<td></td>
<td>If you extract from a multivalued field or variable, the value marks are counted as a character. For example, if you extract 3 characters from the value A]B]C, the result is A]B.</td>
</tr>
<tr>
<td>Store result in</td>
<td>The location in which to store the extracted characters; can be stored in a field or variable.</td>
</tr>
</tbody>
</table>

### Notes

The TEXT-EXTRACT verb functions the same as text extract using relational fields for ad hoc reporting. For more information, see Chapter 7.
The TIME-FORMAT verb converts time data to an internal or external format.

**Properties:**
- **Format the time:** The time to be formatted; can be one of the following:
  - **Number** The time in internal format. For example, 23445 is 6:30 AM.
  - **Literal** The time in external format. For example, "12:30AM". The time must be enclosed in quotation marks.
Properties: (continued)

Field or variable

The time can be in internal or external format.

Specify a time in internal format if converting to external; specify a time in external format if converting to internal.

To format

The type of format, which depends on the format of the time input; can be one of the following:

External Converts an internal format time to external format.

Internal Converts an external format time to internal format.

Using format

If External format is selected, the time can be converted to one of the formats displayed in Table 16-32.

Store result in

The location in which to store the formatted time; can be a field or variable.

Table 16-32. External TIME-FORMAT Verb Examples

<table>
<thead>
<tr>
<th>Internal Format</th>
<th>External Formats</th>
<th>External Formats</th>
<th>External Formats</th>
<th>External Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>hh:mm (hh=00-23)</td>
<td>hh:mmPM (hh=00-23)</td>
<td>hh:mm:ss (hh=00-23)</td>
<td>hh:mm:ssPM (hh=00-23)</td>
</tr>
<tr>
<td>636</td>
<td>00:10</td>
<td>12:10AM</td>
<td>00:10:36</td>
<td>12:10:36AM</td>
</tr>
<tr>
<td>23445</td>
<td>06:30</td>
<td>06:30AM</td>
<td>06:30:45</td>
<td>06:30:45AM</td>
</tr>
</tbody>
</table>
Note: If you store the result in a field defined as a time type, the time is automatically stored in internal format when the record is written. To store the time in external format, save it to an alphanumeric type field or a variable.

Notes

The internal EDGE format is the number of seconds after midnight. For example, 12:10 a.m. is the number 600. You may want to use the external format when using a time field for a report or to export to a different system.

When storing or importing a time field, the internal format is more efficient.
The TRANSACTION verb enables transaction mode, which provides database consistency; that is, the transaction mode ties together a series of file updates into a logical unit of work to ensure that either all files in the series are updated, or no files are updated. For more information, see the EDGE 7.11 DataBase Link Reference Manual, which is available separately as an optional module.
The TRANSFER-RECORD verb transfers a copy of a record to another file.

**Properties:**
- Transfer record from file: The name of the file containing the record to be transferred; must be a filename.
- Transfer record to file: The name of the file to transfer the data to; must be a filename.

**Notes**
To copy a record from one file to another using the TRANSFER-RECORD verb, you must do the following:
- Make sure the fields and positions of the file you are transferring to is identical to the file you are transferring from. If not, data in the new file may be unrecognizable.

- In the logic flow or task, read the record you want to transfer.

- After transferring the record, you must write the record to save it to the file.

**To Transfer All Records**

You can transfer all the records in a file by using a task. You must complete the previous steps for transferring a record. Also, you must select all record IDs to execute using multirecord processing.

**See Also**

The `COPY` verb can also be used to copy a record by using a special syntax. For more information, see the `COPY` verb.
The TRIM verb removes spaces from data and stores the result.

Properties:

Trim Specifies where to remove spaces (see Table 16-33 for examples); can be one of the following:

- **Extra** Removes all spaces to the left of the first non-space character, all spaces to the right of the last non-space character, and all but one space between any two non-space characters.
- **Leading** Removes all spaces to the left of the first non-space character.
Properties: (continued)

- **Trailing**: Removes all spaces to the right of the last non-space character.
- **From the value**: The value to be trimmed; can be a literal, or a field or variable containing the value.
- **Store result in**: The location in which to store the trimmed value; can be stored in a field or variable.

**Table 16-33. TRIM Verb Examples**

<table>
<thead>
<tr>
<th>Trim</th>
<th>Value</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extra</td>
<td>&quot;     ABC 1 2 3        &quot;</td>
<td>&quot;ABC 1 2 3&quot;</td>
</tr>
<tr>
<td>Leading</td>
<td>&quot;         92689             &quot;</td>
<td>&quot;92689     &quot;</td>
</tr>
<tr>
<td>Trailing</td>
<td>&quot;CHICAGO       &quot;</td>
<td>&quot;CHICAGO&quot;</td>
</tr>
</tbody>
</table>
The Unique Account Processing (UAP) verb is no longer supported. It is retained for backward compatibility.
The VOAD-DIAL verb is no longer supported. It is retained for backward compatibility.
The VOAD-DIAL-ATT verb is no longer supported. It is retained for backward compatibility.
The VOAD-HANGUP verb is no longer supported. It is retained for backward compatibility.
The WINDOW verb manipulates windows.

**Properties:**

- **Window name:** The name of the window; can be blank, a literal, or a field or variable containing the name of the window. Each window must have a unique user defined name. This property is optional.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screen name</td>
<td>The name of the screen to place in the window or an HTTP address to route to in a browser; can be blank, a literal, or a field or variable containing the name of the screen or HTTP address. To specify an HTTP address, make sure it begins with http://. For example, &quot;<a href="http://www.imaedge.com">http://www.imaedge.com</a>&quot;. This property is optional.</td>
</tr>
<tr>
<td>Object name</td>
<td>The name of the object to give focus; can be blank, a literal, or a field or variable containing the name of the object.</td>
</tr>
<tr>
<td>Activate window</td>
<td>Specifies whether the window should be activated; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>True (&quot;Y&quot;) The window is activated; all other windows are deactivated.</td>
</tr>
<tr>
<td></td>
<td>False (&quot;N&quot;) Allows the modification of a window and its attributes without activating the window.</td>
</tr>
<tr>
<td></td>
<td>The default is True (&quot;Y&quot;).</td>
</tr>
<tr>
<td></td>
<td>If the window is already active, it remains active regardless of the setting; that is, setting this property to false does not deactivate an active window.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close window</td>
<td>Allows you to close an existing window; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>True (“Y”)</strong>  Closes an existing window.</td>
</tr>
<tr>
<td></td>
<td><strong>False (“N”)</strong>  Leaves the existing window open.</td>
</tr>
<tr>
<td></td>
<td>The default is False (“N”).</td>
</tr>
<tr>
<td></td>
<td>If the window does not exist, a status code of 1 is returned. For a list of status codes, see Table 16-34.</td>
</tr>
<tr>
<td>Create dialog box</td>
<td>Creates a new window as a dialog box within the EDGE Client application window; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>True (“Y”)</strong>  Creates a dialog box.</td>
</tr>
<tr>
<td></td>
<td><strong>False (“N”)</strong> Creates a normal window.</td>
</tr>
<tr>
<td></td>
<td>The default is False (“N”).</td>
</tr>
<tr>
<td></td>
<td>A dialog box does not contain maximize, minimize, or close buttons. It requires the user to perform an action and prevents any interaction with user defined toolbars or other windows until the appropriate action is taken. The standard EDGE Client menu and toolbar can still be accessed.</td>
</tr>
<tr>
<td></td>
<td>A screen name must be specified.</td>
</tr>
</tbody>
</table>

**Table 16-34:**

- Status Code 1: Window does not exist.
**Properties: (continued)**

If a window with the specified name is already displayed, a status code of 4 is returned. For a list of status codes, see Table 16-34.

**Do not create**
Prevents a window from being created; can be one of the following:

- True ("Y") Prevents a window from being created.
- False ("N") Allows a window to be created.

The default is False ("N").

If this property is specified as true and the specified window does not currently exist, a status code of 1 is returned. For a list of status codes, see Table 16-34.

**Always on top**
Forces the desktop to display an EDGE client window as the topmost window even when the client window does not have focus. The default is False ("N").

- True ("Y") Force the client window to always be the topmost window.
- False ("N") Do not force the client window to always be the topmost window.

**Maximize window**
Maximizes the window to fill the EDGE Client application window; can be one of the following:

- True ("Y") Maximizes the window.
## Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimize window</td>
<td>Minimizes the window within the EDGE Client application window; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>True (&quot;Y&quot;) Minimizes the window.</td>
</tr>
<tr>
<td></td>
<td>False (&quot;N&quot;) Uses the current window dimensions.</td>
</tr>
<tr>
<td></td>
<td>The default is False (&quot;N&quot;).</td>
</tr>
<tr>
<td>Restore window size</td>
<td>Restores the window to its prior size; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>True (&quot;Y&quot;) Restores the window to the prior size.</td>
</tr>
<tr>
<td></td>
<td>False (&quot;N&quot;) Uses the current window dimensions.</td>
</tr>
<tr>
<td></td>
<td>The default is False (&quot;N&quot;).</td>
</tr>
<tr>
<td>Top</td>
<td>The top position of the window in pixels relative to the client portion of EDGE Client; can be blank, a literal, or a field or variable.</td>
</tr>
<tr>
<td></td>
<td>The default is blank which will cascade the window.</td>
</tr>
<tr>
<td>Height</td>
<td>The height of the window in pixels; can be blank, a literal, or a field or variable.</td>
</tr>
<tr>
<td></td>
<td>The default is blank which will use the current window dimensions.</td>
</tr>
<tr>
<td><strong>Properties:</strong> (continued)</td>
<td></td>
</tr>
<tr>
<td>---------------------------</td>
<td></td>
</tr>
<tr>
<td><strong>Left</strong></td>
<td>The left position of the window in pixels relative to the client portion of EDGE Client; can be blank, a literal, or a field or variable.</td>
</tr>
<tr>
<td></td>
<td>The default is blank which will cascade the window.</td>
</tr>
<tr>
<td><strong>Width</strong></td>
<td>The width of the window in pixels; can be blank, a literal, or a field or variable.</td>
</tr>
<tr>
<td></td>
<td>The default is blank which will use the current window dimensions.</td>
</tr>
<tr>
<td><strong>Color</strong></td>
<td>The window background color; can be blank, a comma-delimited string, or a field or variable.</td>
</tr>
<tr>
<td></td>
<td>The comma-delimited string represents the RGB value for red, green, and blue. Values range from &quot;0,0,0&quot; (black) to &quot;255,255,255&quot; (white).</td>
</tr>
<tr>
<td></td>
<td>Click the button to the right of the entry field to select the window background color from the following Windows dialog. Click the Define Custom Colors button to define a custom color.</td>
</tr>
<tr>
<td></td>
<td>If you leave this field blank, the background color defaults to the background screen color for the current style.</td>
</tr>
</tbody>
</table>
Frame style Defines the window style for a main window or child window according to the Microsoft Document Interface (MDI) standard; can be blank, a comma-delimited string, or a field or variable. Note that the Pop Up option is always disabled.

If you leave this property blank, EDGE selects an appropriate default for the window type.
Properties: (continued)

The following dialog displays when you click the selection button to the right of the Frame entry field. Select window frame style properties from this dialog.
Effect of Border and Edge options on the resulting window:

- Window components are interdependent.

- The resulting window does not include minimize and maximize boxes unless the window includes a system menu.

- The resulting window does not include a system menu unless the window also includes a title bar.

- In this dialog box, the title bar option is automatically checked when you select a normal border. This option is disabled when you select a double border.

- A minimize or maximize box displays in a disabled state when its corresponding option is unchecked in this dialog.
Examples of border and edge window styles:

- Normal border with raised edge
- Normal border with 3D edge
- Double border with raised edge
- Double border with 3D edge
**Properties: (continued)**

Other Windows options:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title Bar</td>
<td>Adds a title bar to the window. Uncheck to omit the title bar.</td>
</tr>
<tr>
<td>Maximize Box</td>
<td>Adds a maximize box to the window. Uncheck to add the maximize box in a disabled state.</td>
</tr>
<tr>
<td>Minimize Box</td>
<td>Adds a minimize box to the window. Uncheck to add the minimize box in a disabled state.</td>
</tr>
<tr>
<td>System Menu</td>
<td>Adds a system menu to the window. Uncheck to omit the system menu.</td>
</tr>
<tr>
<td>Resize</td>
<td>Allows the user to resize the window. Uncheck to disable manual resizing.</td>
</tr>
<tr>
<td>Visible</td>
<td>Creates a visible window. Uncheck to create an invisible window.</td>
</tr>
<tr>
<td>Enabled</td>
<td>Enables user input. Uncheck to disable user input.</td>
</tr>
</tbody>
</table>

**Store status code in**

The location in which to store the success or failure of the command; can be a field or variable. For a list of status codes, see Table 16-34.
The WINDOW verb allows you to do the following:

- Activate a window, screen, and object.
- Create a window.
- Close a window.

### Table 16-34. Window Status Codes

<table>
<thead>
<tr>
<th>Codes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>null (0)</td>
<td>Success.</td>
</tr>
<tr>
<td>1</td>
<td>Named window not found.</td>
</tr>
<tr>
<td>2</td>
<td>Named screen not found.</td>
</tr>
<tr>
<td>3</td>
<td>Dialog window active. Returned when logic called from a window created with the create dialog box property attempts to create a new window.</td>
</tr>
<tr>
<td>4</td>
<td>Window exists. Returned if you try to create a dialog window, with the same name as a window that already exists.</td>
</tr>
<tr>
<td>5</td>
<td>Screen not specified. Returned if you try to create a dialog window without specifying a screen name.</td>
</tr>
<tr>
<td>6</td>
<td>Close failed. Returned if you try to close a window that cannot be closed.</td>
</tr>
<tr>
<td>7</td>
<td>Parse error.</td>
</tr>
</tbody>
</table>
• Maximize a window.
• Minimize a window.
• Open a window.
• Position a window.
• Size a window.

Windows are the means of displaying multiple screens within an application. You can use windows, for example, to present several screens at one time or the same screen in multiple windows. If data is updated for a particular field on any screen, all other screens that contain that exact field are automatically updated even if they are not active. Although multiple windows may be displayed at the same time, only one window is considered active at any one time.

Although the Window name, Screen name, and Object name properties are optional, at least one of these three properties must be specified when using the WINDOW verb.

The WINDOW verb does not terminate logic flows which means that the logic flow continues to the next statement. Be careful about making assumptions regarding which screen is the current screen, since the current screen may change because the active window has changed.
To prevent a window from being closed with the close button, the SYS field `$WINCLOSING` must be set to N.

**See Also**

The **MENU**, **ROUTE**, and **SET-PROPERTY** verbs also allow you to route to a window, screen, and object path.
WRITE-RECORD

The WRITE-RECORD verb creates or updates a specified record in a user defined file from the information in the record workspace.

Properties:

- **To the file**: The file name to write the record to; must be a file-name.
- **Write record ID**: The ID of the record to be written; can be a literal, or a field or variable containing the ID. In a guide, $CALLEVENT contains the current call event. In a task, $ID contains the current record ID being processed.
Properties: (continued)

**Store status in**
The location in which to store the result of the write; can be a field or variable. This field is optional. The result is one of the following:

- **Y**: Indicates the write was successful.
- **N**: Indicates the write failed.

The result of WRITE-RECORD is also reflected in $READ.OK as Y when successful and N when unsuccessful.

**Locking record**
Specifies whether to lock the record if not already locked and to keep the record locked after writing; can be one of the following:

- **Yes**: The record remains locked after it is written.
- **No**: The record is written and unlocked.

**Do not wait for lock**
Specifies whether to wait to write the record if it is locked by another operator. Applies only when the Locking record property is Yes.

- **Yes**: Do not wait to write the record.
- **No**: Wait to write the record.

For the results of various lock settings, see Table 16-35.
**Notes**

To permanently store data in a file, you must use the WRITE-RECORD verb. This verb is frequently used in the end of guide logic so the latest and most correct data is written when the record is filed. For example, EDGE automatically writes data to its own files at the end of the guide. For more information on end of guide logic, see Chapter 15.

The effects of various WRITE-RECORD settings on the outcome of the read are listed in Table 16-35. If you are using an optional EDGE DataBase Link module, you can set transaction locks that affect reading, writing, and deleting records. For information on transaction locks, refer to the EDGE DataBaseLink Reference Manual.

---

**Note:** If the TRANSACTION verb is used, WRITE-RECORD actions are not done until the transaction is committed

For more information on the results of locking, see the READ-RECORD verb, Table 16-27.
### Table 16-35. WRITE-RECORD Verb Results

<table>
<thead>
<tr>
<th>Locking Record Setting</th>
<th>Do Not Wait Setting</th>
<th>Record Is Locked by Another Operator</th>
<th>Result of WRITE-RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>The SYS field $LOCKED is set to Y, the operator does not wait for the record to be unlocked, the record is not written, and the logic flow continues.</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>The operator waits until the record is written and released by the operator with the lock, then the record is locked and written. The record remains locked.</td>
</tr>
<tr>
<td>No</td>
<td>N/A</td>
<td>Yes</td>
<td>The record is written without a lock. Warning: The operator with the lock can overwrite the record which was written without a lock.</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes or No</td>
<td>No</td>
<td>The record is locked and then written. The record remains locked until one of the following occurs: The record is released (See the RELEASE-RECORD verb). The record is written without a lock.</td>
</tr>
</tbody>
</table>
Table 16-35. WRITE-RECORD Verb Results (continued)

<table>
<thead>
<tr>
<th>Locking Record Setting</th>
<th>Do Not Wait Setting</th>
<th>Record Is Locked by Another Operator</th>
<th>Result of WRITE-RECORD</th>
</tr>
</thead>
</table>
| No                     | N/A                 | No                                   | The $END screen is reached.  
The record is written without a lock and released. |

Notes

If the view contains more than one object, the READ-RECORD and WRITE-RECORD verbs cannot be used because the SELECT statement includes columns from more than one object. To create an INSERT or UPDATE SELECT statement that includes fields from one table, use the SQL-GENERAL verb.
The Microsoft Windows operating system provides a method of exchanging data between applications called Dynamic Data Exchange (DDE). This process is available in EDGE through the EDGE Client interface.

The applications that are used in DDE processes are referred to in one of the following ways, depending on their function:

- **Client application**: The application that requests information.
- **Server application**: The application that supplies information.

An application that has both capabilities is referred to as a DDE client/server application. EDGE Client has both capabilities and hence is a DDE client/server application.

- To have EDGE Client operate as a client, use the EDGE verb called DDE to open channels to DDE server applications and...
to retrieve and change data in them. The DDE verb is available for logic flows.

- To access EDGE Client’s server capabilities, use the set of EDGE Client server commands that respond to the Execute and Request commands in the client application.

The following list contains the names of some DDE client/server applications:

<table>
<thead>
<tr>
<th>Access</th>
<th>Object Vision</th>
<th>SuperBase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ami Pro</td>
<td>Paradox</td>
<td>Toolbook</td>
</tr>
<tr>
<td>Crosstalk</td>
<td>Procomm</td>
<td>Visual Basic</td>
</tr>
<tr>
<td>DynaComm</td>
<td>Q+E Database Editor</td>
<td>WinFax</td>
</tr>
<tr>
<td>Excel</td>
<td>Quattro Pro</td>
<td>Word</td>
</tr>
<tr>
<td>Lotus 1-2-3</td>
<td></td>
<td>WordPerfect</td>
</tr>
</tbody>
</table>
**DDE Concepts**

The Microsoft Windows DDE protocol provides a standard interface for applications to use to exchange data. The protocol allows applications to exchange data and carry out remote commands using Windows messages.

The process of two applications exchanging information is referred to as a conversation. A typical DDE conversation consists of the following steps:

1. The client and server applications are started if they are not already running.

2. The client application initiates the conversation and the server acknowledges that it is ready. To initiate the conversation, you need the name of the application you want to converse with (the server) and the subject of the conversation (the topic).

3. Windows assigns a unique channel number to the conversation. When EDGE Client is the client, EDGE handles this automatically.

4. The applications exchange data using any or all of the methods listed in Table 17-1.

5. The conversation is terminated. To accomplish this when EDGE Client is the client, use the DDE verb *Terminate* command.
To send data to an application, you need to specify the following:

- A server, which is the application name.
- A topic, which is the subject of the conversation.
- An item name which marks a specific piece of data. This may not be required, depending on the application.

What an item is varies by application. For example, in Microsoft Word, bookmarks are items. (To create a bookmark in Microsoft Word, select the text that will make up the item, then assign a bookmark name to it.) In Microsoft Excel, row and column positions are items (for example, R1C1). EDGE Client does not use item names, so when you connect to EDGE Client as a server, you do not specify an item name. (EDGE Client uses other means of identifying the data. For more information, see the section EDGE Client as Server.)

This chapter describes the requirements for using EDGE as both a client and a server. For information on requirements for the application EDGE is to be connected to, refer to the documentation for that application.
<table>
<thead>
<tr>
<th>Methods of Exchanging Data</th>
<th>To accomplish this with EDGE Client as the client</th>
<th>To accomplish this with EDGE Client as the server</th>
</tr>
</thead>
<tbody>
<tr>
<td>The client requests inform-</td>
<td>Use the DDE verb Poke command.</td>
<td>Use the EDGE Client server Commands for Client Request Statements.</td>
</tr>
<tr>
<td>ation from the server.</td>
<td>Use the DDE verb Execute command.</td>
<td>Use the EDGE Client server Commands for Client Execute Statements.</td>
</tr>
<tr>
<td>The server returns the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>information.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The client sends a com-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mand to the server, which</td>
<td></td>
<td></td>
</tr>
<tr>
<td>the server executes.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The client sends data to</td>
<td>Use the DDE verb Poke command.</td>
<td>Not available.</td>
</tr>
<tr>
<td>the server.</td>
<td>Use the DDE verb Advise command with the Return new data property set to N.</td>
<td>Not available.</td>
</tr>
<tr>
<td>The client requests the</td>
<td>Use the DDE verb Advise command with the Return new data property set to Y.</td>
<td>Not available.</td>
</tr>
<tr>
<td>server to notify the client whenever a data item changes (this is referred to as a <em>warm link</em>).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The client requests the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>server to notify the client and send the new data whenever a data item changes (this is referred to as a <em>hot link</em>).</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EDGE Client as Client - DDE Verb

You can use EDGE Client as a DDE client to initiate a conversation with any Windows application that can act as a DDE server. For example, you can initiate a conversation with Microsoft Word, specifying WinWord as the server and a document name as the topic. You can then send information, such as the name and address of the current customer to Word, then send a command to print the document. Or you can initiate a conversation with Microsoft Excel, specifying Excel as the server and a spreadsheet name as the topic. You can request data from specific rows and columns in the spreadsheet and use that information in your guide.

To use EDGE Client as a DDE client, the following is required:

- The server application (the application you are requesting information from) must already be running.
- One of the following DDE verb commands, which initiates the conversation, must be invoked:
  - Advise
  - Execute
  - Poke
  - Request
- Additional DDE commands can be invoked as desired.
The DDE conversation must be terminated.

Unlike many applications that have DDE client support, there is no DDE Initiate command under EDGE Client. The initiation of a DDE conversation is automatically performed when you first specify a different Server in one of these DDE verb commands: Advise, Execute, Poke, or Request.

Therefore, EDGE’s DDE verbs do not require you to keep track of DDE channels. Internal to EDGE, a table is kept of the opened channels and their associated Server value. To obtain the channel number, use the DDE Status command.

**Note:** Although DDE allows other data types, EDGE currently supports only the Microsoft Windows CF_TEXT data type. Other data types such as bitmaps or sound wave files are not supported. *(CF_TEXT is the Windows Clipboard text format for ANSI text.)*

**Configuration Limits**

The following configuration limits for the EDGE DDE verb are kept in the EDGEO.INI file under the Guide heading. (For more information on the EDGEO.INI file, see EDGE 7.11 Client Reference Manual.)

- **ddeMaxAdvises**  The maximum number of advisements. The default is 3.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ddeMaxConnections</td>
<td>The maximum number of connections to DDE servers. The default is 3.</td>
</tr>
<tr>
<td>ddeMaxWait</td>
<td>The maximum time in milliseconds to wait for a response from a DDE server. The default is 1000 (that is, 1 second).</td>
</tr>
</tbody>
</table>
**DDE Verb Usage**

The DDE verb allows EDGE to communicate with Windows applications that support DDE.

![DDE Verb Usage Diagram]

**Properties:**

- DDE command: The command to perform; can be one of the following:
  - Advise
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute</td>
</tr>
<tr>
<td>Poke</td>
</tr>
<tr>
<td>Request</td>
</tr>
<tr>
<td>Status</td>
</tr>
<tr>
<td>Terminate</td>
</tr>
<tr>
<td>Unadvise</td>
</tr>
</tbody>
</table>

**Server**

The server name and topic name that identify the DDE conversation; can be a literal, or a field name or variable that contains the names. The names are separated by a slash.

The server name specifies the name of the DDE server application. This name is frequently the same as the application’s executable filename.

The server application must be running.

The topic name is specific to each DDE server and is frequently the pathname to a file the DDE server has open. The values required are specific to each DDE server application. For more information, see the documentation for the specific application.

The maximum number of conversations that can be opened at one time is kept in the EDGE0.INI file in the item ddeMaxConnections.
Each time one of the following DDE commands specifies a new Server, a DDE conversation is initiated:

- Advise
- Execute
- Poke
- Request

Store status in The location in which the status of the command is to be returned; can be a field or variable name.

For a list of status codes that can be returned from EDGE, see Table 17-2. For a list of the status codes that can be returned from Windows, see Table 17-3. (For information on the Windows codes, see the Microsoft Windows DDE documentation.)

If a single server is specified, the status is one of the following:

0 No connection to specified server.

n Channel number.

If ALL is specified as the server, the status is returned in multivalue format, with one value for each connection. Each value contains the following information, separated by a subvalue mark:

Subvalue 1 Server.
Properties: (continued)

Subvalue 2  Item name.

Note: A subvalue mark has an ASCII decimal value of 28. To extract a subvalue, use the CONVERT verb to create a variable with an ASCII value of 28, then use that variable in the GROUP-EXTRACT verb.

Table 17-2. Status Codes from DDE verb

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>OK</td>
<td>Operation was successful.</td>
</tr>
<tr>
<td>-1</td>
<td>DDE_OUT_OF_MEMORY</td>
<td>Unable to allocate memory for a required structure.</td>
</tr>
<tr>
<td>-2</td>
<td>DDE_ADVISE_ACTIVE</td>
<td>Attempt to Advise a DDE item that is already being Advised.</td>
</tr>
<tr>
<td>-3</td>
<td>DDE_ADVISE_NOT_ACTIVE</td>
<td>Attempt to Unadvise a DDE item that is not being Advised.</td>
</tr>
<tr>
<td>-4</td>
<td>DDE_ADVISE_FULL</td>
<td>Too many Advises active. Use Unadvise or increase the maximum advisements allowed (specified in the EDGEO.INI file).</td>
</tr>
</tbody>
</table>
Table 17-2. Status Codes from DDE verb (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-5</td>
<td>DDE_CONNECTIONS_FULL</td>
<td>Too many DDE connections. Use Terminate or increase the maximum DDE connections allowed (specified in the EDGEO.INI file).</td>
</tr>
<tr>
<td>-6</td>
<td>DDE_SYNTAX_ERROR</td>
<td>Server argument missing / separator.</td>
</tr>
<tr>
<td>-7</td>
<td>DDE_NOT_CONNECTED</td>
<td>Attempt to Unadvise or Terminate from a DDE conversation that does not exist.</td>
</tr>
</tbody>
</table>

Table 17-3. Status Codes from Window’s DDE Library

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>16384</td>
<td>DMLERR_ADVACKTIMEOUT</td>
</tr>
<tr>
<td>16385</td>
<td>DMLERR_BUSY</td>
</tr>
<tr>
<td>16386</td>
<td>DMLERR_DATAACKTIMEOUT</td>
</tr>
<tr>
<td>16387</td>
<td>DMLERR_DLL_NOT_INITIALIZED</td>
</tr>
<tr>
<td>16388</td>
<td>DMLERR_DLL_USAGE</td>
</tr>
<tr>
<td>16389</td>
<td>DMLERR_EXEACKTIMEOUT</td>
</tr>
<tr>
<td>16390</td>
<td>DMLERR_INVALIDPARAMETER</td>
</tr>
<tr>
<td>16391</td>
<td>DMLERR_LOW_MEMORY</td>
</tr>
</tbody>
</table>
### Advise

The Advise command tells the DDE server to advise EDGE whenever a specific data item’s value has changed.

<table>
<thead>
<tr>
<th>Code</th>
<th>Error Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>16392</td>
<td>DMLERR_MEMORY_ERROR</td>
</tr>
<tr>
<td>16393</td>
<td>DMLERR_NOTPROCESSED</td>
</tr>
<tr>
<td>16394</td>
<td>DMLERR_NO_CONV_ESTABLISHED</td>
</tr>
<tr>
<td>16395</td>
<td>DMLERR_POKEACKTIMEOUT</td>
</tr>
<tr>
<td>16396</td>
<td>DMLERR_POSTMSG_FAILED</td>
</tr>
<tr>
<td>16397</td>
<td>DMLERR_REENTRANCY</td>
</tr>
<tr>
<td>16398</td>
<td>DMLERR_SERVER_DIED</td>
</tr>
<tr>
<td>16399</td>
<td>DMLERR_SYS_ERROR</td>
</tr>
<tr>
<td>16400</td>
<td>DMLERR_UNADVACKTIMEOUT</td>
</tr>
<tr>
<td>16401</td>
<td>DMLERR_UNFOUND_QUEUE_ID</td>
</tr>
</tbody>
</table>
Properties:

Call logic

The name of the logic flow to call when the item is modified; can be a literal, or a field or variable containing the logic flow name.

This logic flow should contain the statements necessary to handle the change in the value of the DDE item. The logic flow can reference global variables (those that start with %) and system fields; it can also create and access its own local variables (those that start with @).
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Item</strong></td>
<td>The name of a DDE item on the DDE server; can be a literal, or a field or variable containing the item name. DDE items are specific to each DDE Server application. For example, Word for Windows uses bookmarks as items. Lotus 1-2-3 uses cell addresses (in R1C1 format).</td>
</tr>
<tr>
<td><strong>Store notification in</strong></td>
<td>Specifies where to place the information about the changed value; can be a field or a variable. The data is returned in multivalued format where the values contain the following data:</td>
</tr>
<tr>
<td></td>
<td><strong>Value 1</strong> Server.</td>
</tr>
<tr>
<td></td>
<td><strong>Value 2</strong> Item name.</td>
</tr>
<tr>
<td></td>
<td><strong>Value 3</strong> Changed value.</td>
</tr>
<tr>
<td></td>
<td>Value 3 is present only if the Return new data property is Y; otherwise it is null.</td>
</tr>
</tbody>
</table>

If multiple projects are being used, the logic flow should not access call variables (those that start with #) or fields unless the Unadvise at end of call property is Y. Use of fields or call variables when going across projects cannot be trapped as an error and will produce unpredictable results.
Properties: (continued)

Unadvise at end of call

Specifies whether to stop being advised of changes to this DDE item after the call has been completed. One of the following can be specified:

No

Does not stop the advise at the end of the call.

Yes

Stops the advise at the end of the call.

Unless there is a specific need to be advised of changes to the data after a call has been completed, this prompt should be answered as Y. If you answer N to this property, the logic flow continues to be called every time the DDE item is changed until either the EDGE user logs out, or the DDE Unadvise or DDE Terminate command is invoked.

Return new data

Specifies if the new value of a modified DDE item is returned. One of the following can be specified:

No

Does not return the new value. This is the default.

Yes

Returns the new value.

If No is selected, the logic flow is called but the changed data is not returned. In this case, you can get the modified value by using the DDE Request command.
Notes

You can call the DDE Advise command multiple times for different DDE items and you can specify the same or different logic flows. If you specify the same logic flow, you can have the logic flow determine which item has changed by examining the contents of the variable specified in the Store notification in property. However, you cannot call the DDE Advise command for the same DDE item and specify different logic flows. (If you try, the error DDE_ADVERTISE_ACTIVE is returned.)

The maximum number of advisements that can be open at one time is kept in the EDGEO.INI file in the item ddeMaxAdvises.

The Advise command can initiate a DDE conversation.

Example

DDE Advise from "WINWORD/C:\DOCS\LETTER.DOC" item "NAME" by calling "DDE.NEW.VALUE" storing advisement in ADVISEMENT

Execute

The Execute command requests the DDE server to execute a specified command.
**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command to execute</td>
<td>The string to be executed by the DDE server; can be a literal, or a field or variable containing the string.</td>
</tr>
<tr>
<td></td>
<td>The syntax of the string is specific to each DDE server. For information on the syntax, refer to the documentation for the server application.</td>
</tr>
</tbody>
</table>
Notes

The Execute command can initiate a DDE conversation.

Example

The following logic flow demonstrates how to use the EDGE DDE verb with Window’s Program Manager DDE Server to create a program group called NewGroup:

DDE Execute by "PROGMAN/PROGMAN" command
"[CreateGroup(NewGroup )]"

Poke

The Poke command sends a new value to a data item in the DDE server application.
**Properties:**

- **Item**: The name of a DDE item on the DDE server; can be a literal, or a field name or variable that contains the item name.

  DDE items are specific to each DDE Server application. For example, Word for Windows uses bookmarks as items. Lotus 1-2-3 uses cell addresses (in R1C1 format).

- **Value**: The value to be sent; can be a literal, or a field name or variable that contains the value.
Notes

The Poke command can initiate a DDE conversation.

Example

The following logic flow demonstrates how to use EDGE with Microsoft Word to change DDE items in a document to specific values and print the file. (To create a DDE item in Microsoft Word, select the text that will make up the item, then assign a bookmark name to it.)

- Copy "WINWORD/C:\DOCS\LETTER.DOC" to @DDESERVER
- DDE Poke to @DDESERVER item "NAME" with "Jane Doe"
- DDE Poke to @DDESERVER item "PHONE" with "909 555 1212"
- DDE Execute by @DDESERVER command "[PrintFile]"

Request

The Request command requests the DDE server to send a specific data item.
Properties:

**Item**
- The name of a DDE item on the DDE server; can be a literal, or a field name or variable that contains the item name.
- DDE item names are specific to each DDE Server application. For example, Lotus 1-2-3 uses cell addresses (in R1C1 format) as item names, Word for Windows uses bookmarks as item names.

**Store result in**
- Specifies where the value of the item should be stored; can be a field name or variable.
Notes

The Request command can initiate a DDE conversation.

Example

DDE Request from "EXCEL/C:\EXCEL\BUDGET.WKS" item "R1C1" store result in T-GROSS

Status

The Status command indicates whether a specified DDE server connection is open, and if so, the channel on which it is open. To get the status of all open DDE connections, specify ALL as the Server.
The Status command can be used, for example, to see if a connection to a server has already been established before performing other DDE actions. Since the connection to a server may be disrupted by a user exiting the server application, this command can be used to make sure the connection to the server is still active.

**Properties:**

There are no properties specific to this command.

**Notes**

The Status command can be used, for example, to see if a connection to a server has already been established before performing other DDE actions. Since the connection to a server may be disrupted by a user exiting the server application, this command can be used to make sure the connection to the server is still active.


**Example**

DDE Status from "ALL" Store status in @DDESTATUS

The information returned in @DDESTATUS is similar to the following:

```
WINWORD/c:\docs\letter.doc$1234^[]PROGMAN/PROGMANS1235
```

where `^[]

represents a value mark.

`S`

represents a subvalue mark.

**Terminate**

The Terminate command closes a specified DDE connection created by a previous Advise, Execute, Poke, or Request command. To terminate all open DDE connections, specify ALL as the Server.
By default, a DDE connection to a particular Server remains open until the user logs off.

You should use this command only if you need to terminate a DDE connection prior to the user logging off. For example, if DDE
requests are made to multiple DDE servers or topics, there is a DDE connection for each Server used. If the maximum number of connections specified in the EDGEO.INI file is reached, the next DDE command that tries to open a new connection will fail and return the status DDE_CONNECTIONS_FULL. In this case, you can use the Terminate command to close one of the open connections.

**Example**

DDE Terminate connection to "PROGMAN/PROGMAN"

The Unadvise command cancels a specified DDE Advise command.
**Properties:**

- **Item**
  The name of a DDE item on the DDE server; can be a literal, or a fieldname or variable that contains the name.

  DDE items are specific to each DDE server application. For example, Word for Windows uses bookmarks as items. Lotus 1-2-3 uses cell addresses (in R1C1 format).

**Example**

DDE Unadvise from “WINWORD/C:\DOCS\LETTER.DOC” changes to “NAME”
**EDGE Client as Server**

The EDGE Client DDE server support allows other Windows applications to examine the state that EDGE Client is in and to control EDGE Client by forcing keystrokes to be executed. For example, you can use EDGE Client for the following:

- You can use a macro in Word to request and receive data from an EDGE Client screen to insert into a letter.
- You can use an Aspect script in Procomm for Windows to transfer data from an EDGE Client screen to a mainframe session.
- You can use a Visual Basic program to connect with EDGE Client and automate keystrokes (using SendKeys) to add data from a database.

EDGE Client responds to Execute and Request commands from a DDE client application.

DDE server protocol talks about the following types of DDE links:

- **Cold**: Provides information once, at the time the command is invoked.
- **Warm**: Sets up a link and notifies the client each time the data changes.
Establishing a Conversation

The EDGE Client DDE server commands are all cold link types; that is, they return information at the time they are invoked. They do not set up an on-going link.

To establish a conversation with EDGE Client as a DDE server, use the following in the client Initiate command:

- DDE server name: GEO
- DDE topic name: Server

Hot  Sets up a link, notifies the client each time the data changes, and returns the changed values.
**Commands for Client Execute Statements**

This section describes the commands that can be specified in a client Execute statement. The command names are not case-sensitive. However, the brackets shown in the syntax are required.

The commands that can be used in a client Execute statement are listed in Table 17-4.

*Note: When the Execute concludes, the DDE protocol returns a code indicating success or failure. If the command failed, you can determine the reason for the failure by using the Result command in a client Request statement.*

**Table 17-4. EDGE Client Server Commands for Client Execute Statements**

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AppActivate</td>
<td>Activates a specified application.</td>
</tr>
<tr>
<td>Enter</td>
<td>Makes EDGE Client the currently active application and sends an Enter character to it.</td>
</tr>
<tr>
<td>SendKeys</td>
<td>Sends keystrokes.</td>
</tr>
<tr>
<td>SetScreenText</td>
<td>Obsolete. Replaced by the SetTargetContent command.</td>
</tr>
</tbody>
</table>
The AppActivate command activates a specified application.

**Syntax**

```plaintext
[AppActivate(name)]
```

`name` The name of an application or the window title of the specific application to activate. This must be the actual name and is not enclosed in quotation marks.

If no name is specified, EDGE Client is activated. The name must refer to an application that is currently running.

The Enter command makes EDGE Client the currently active application and sends an Enter character to it.

**Syntax**

```plaintext
[Enter]
```
Notes

This command is the same as specifying a SendKeys( “{enter}” ) command. (For information on SendKeys, see the next section.)

SendKeys

The SendKeys command sends keystrokes to EDGE Client or to a specified application as if the keystrokes were typed at the keyboard.

Syntax

\[[\text{SendKeys}(\text{string},\text{option})]\]

\textbf{string}  Keystrokes to send, enclosed in quotation marks. You can specify up to 1024 keystrokes. For more information, see the Notes section.

\textbf{option}  Indicates whether to activate EDGE Client; can be one of the following:

\begin{itemize}
  \item 0 \hspace{1em} Activates EDGE Client before processing keys. This is the default.
  \item 1 \hspace{1em} Does not activate EDGE Client. Sends keys to the current active application.
\end{itemize}

This property is optional.
Notes

The string used in the SendKeys command contains the actual keystrokes for printable characters and codes for the keystrokes for unprintable characters. Each code counts as one keystroke.

To specify a printable character in the string, if it is not a reserved character, use the character itself. For example, to send the letter J, specify “J”.

To specify non-printable characters, use codes set up for that purpose. All the codes except ALT ( % ), CONTROL ( ^ ), and SHIFT ( + ) must be enclosed in braces within the quotation marks. For a list of codes, see Table 17-5.

The following printable characters are reserved; if you need to send one of these characters, enclose it in braces; for example, to send a plus sign, enclose it in braces {+}:

- +   Plus sign.
- %   Percent sign.
- ~   Tilde.
- (   Left parenthesis.
- )   Right parenthesis.
- {   Left brace.
To specify repeating keys, use the form “{character number}” where there is a space between character and number. For example, “{h 10}” means press the “h” key 10 times; “{left 3}” means press the left arrow three times. Note that if the keystroke is a code in braces, the number is also enclosed in braces.

To specify a combination that includes the CONTROL or ALT keys, you can specify one CONTROL or ALT code, followed by the additional characters in parentheses. For example, to send a CONTROL+C followed by a CONTROL+F, specify the following (the ^ represents the CONTROL key):

“^({cf})”

**Table 17-5. Codes to Use for Non-Printable Characters**

<table>
<thead>
<tr>
<th>Non-Printable Characters</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alt</td>
<td>%</td>
</tr>
<tr>
<td>Backspace</td>
<td>{backspace}</td>
</tr>
<tr>
<td>Break</td>
<td>{break}</td>
</tr>
<tr>
<td>Caps Lock</td>
<td>{capslock}</td>
</tr>
<tr>
<td>Clear</td>
<td>{clear}</td>
</tr>
<tr>
<td>Control</td>
<td>^</td>
</tr>
</tbody>
</table>
### Table 17-5. Codes to Use for Non-Printable Characters (continued)

<table>
<thead>
<tr>
<th>Non-Printable Characters</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Del</td>
<td>{delete} or {del}</td>
</tr>
<tr>
<td>Down Arrow</td>
<td>{down}</td>
</tr>
<tr>
<td>End</td>
<td>{end}</td>
</tr>
<tr>
<td>Enter</td>
<td>{enter} or ~</td>
</tr>
<tr>
<td>Esc</td>
<td>{escape} or {esc}</td>
</tr>
<tr>
<td>Help</td>
<td>{help}</td>
</tr>
<tr>
<td>Home</td>
<td>{home}</td>
</tr>
<tr>
<td>Ins</td>
<td>{insert}</td>
</tr>
<tr>
<td>Left Arrow</td>
<td>{left}</td>
</tr>
<tr>
<td>Num Lock</td>
<td>{numlock}</td>
</tr>
<tr>
<td>Page Down</td>
<td>{pgdn}</td>
</tr>
<tr>
<td>Page Up</td>
<td>{pgup}</td>
</tr>
<tr>
<td>Print Screen</td>
<td>{prtsc}</td>
</tr>
<tr>
<td>Right Arrow</td>
<td>{right}</td>
</tr>
<tr>
<td>Scroll Lock</td>
<td>{scrolllock}</td>
</tr>
<tr>
<td>Shift</td>
<td>+</td>
</tr>
<tr>
<td>Tab</td>
<td>{tab}</td>
</tr>
<tr>
<td>Up Arrow</td>
<td>{up}</td>
</tr>
</tbody>
</table>
### Table 17-5. Codes to Use for Non-Printable Characters (continued)

<table>
<thead>
<tr>
<th>Non-Printable Characters</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>{f1}</td>
</tr>
<tr>
<td>F2</td>
<td>{f2}</td>
</tr>
<tr>
<td>F3</td>
<td>{f3}</td>
</tr>
<tr>
<td>F4</td>
<td>{f4}</td>
</tr>
<tr>
<td>F5</td>
<td>{f5}</td>
</tr>
<tr>
<td>F6</td>
<td>{f6}</td>
</tr>
<tr>
<td>F7</td>
<td>{f7}</td>
</tr>
<tr>
<td>F8</td>
<td>{f8}</td>
</tr>
<tr>
<td>F9</td>
<td>{f9}</td>
</tr>
<tr>
<td>F10</td>
<td>{f10}</td>
</tr>
<tr>
<td>F11</td>
<td>{f11}</td>
</tr>
<tr>
<td>F12</td>
<td>{f12}</td>
</tr>
<tr>
<td>F13</td>
<td>{f13}</td>
</tr>
<tr>
<td>F14</td>
<td>{f14}</td>
</tr>
<tr>
<td>F15</td>
<td>{f15}</td>
</tr>
<tr>
<td>F16</td>
<td>{f16}</td>
</tr>
</tbody>
</table>
**SetTargetContent**

The `SetTargetContent` command stores a data string into the current Target object.

**Syntax**

```
[SetTargetContent(data)]
```

**data** The string containing the data to be stored.

**Notes**

This command must be preceded by a `Target` command that defines the target object.

**Tab**

The `Tab` command makes EDGE Client the currently active application and sends a Tab character to it.

**Syntax**

```
[Tab]
```

**Notes**

This command is the same as specifying a `SendKeys` ("{tab}") command.
**Target**

The Target command specifies an object in the EDGE Client window for which information is to be returned. The target can be set in one of the following ways (see the individual syntax definitions):

- **Syntax 0** Based on its tabbing order.
  - [Target(0, *tab*)]
  - 0 Sets the target to an object based on its tabbing order.
  - *tab* The tabbing order of the object.

- **Syntax 1** Based on its name.
  - [Target(1, *name*, *value*))]
  - 1 Sets the target to an object based on its name.
  - *name* The name of the object.
  - *value* The value position of the object.
**Syntax 2**

```
[Target(2)]
```

2  Sets the target to the object that currently has the focus.  
   The 2 is optional.

**Syntax 3**

```
[Target(3,tag)]
```

3  Sets the target to an object based on its tag.  
   tag  The tag of the object.
**Commands for Client Request Statements**

This section describes the commands that can be specified in a client Request statement. The commands are listed in Table 17-6.

*Table 17-6. EDGE Client Server Commands for Client Request Statements*

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CallTimer</td>
<td>The current value of the call timer.</td>
</tr>
<tr>
<td>FocusCaption</td>
<td>The caption of the object with the current input focus.</td>
</tr>
<tr>
<td>FocusContent</td>
<td>The content of the object with the current input focus.</td>
</tr>
<tr>
<td>FocusFormattedValue</td>
<td>The formatted value of the object with the current input focus.</td>
</tr>
<tr>
<td>FocusName</td>
<td>The name of the object with the current input focus.</td>
</tr>
<tr>
<td>FocusPosition</td>
<td>The position of the object with the current input focus.</td>
</tr>
<tr>
<td>FocusType</td>
<td>The type of the object with the current input focus.</td>
</tr>
<tr>
<td>FocusType Name</td>
<td>The text name for the type of object with the current input focus.</td>
</tr>
</tbody>
</table>
### Table 17-6. EDGE Client Server Commands for Client Request Statements (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FocusValue</td>
<td>The value of the object with the current input focus.</td>
</tr>
<tr>
<td>ObjectCount</td>
<td>The number of objects on the current screen.</td>
</tr>
<tr>
<td>Port</td>
<td>The current host port connection.</td>
</tr>
<tr>
<td>ProjectName</td>
<td>The current project name.</td>
</tr>
<tr>
<td>Result</td>
<td>The result code of the last DDE Execute command.</td>
</tr>
<tr>
<td>ScreenName</td>
<td>The name of the current screen.</td>
</tr>
<tr>
<td>ScreenTitle</td>
<td>The title of the current screen.</td>
</tr>
<tr>
<td>TabObjectCount</td>
<td>The number of objects that can be tabbed to.</td>
</tr>
<tr>
<td>TargetCaption</td>
<td>The caption of the current target object.</td>
</tr>
<tr>
<td>TargetContent</td>
<td>The content of the object set as the current input target.</td>
</tr>
<tr>
<td>TargetFormatted-Value</td>
<td>The formatted value of the current target object.</td>
</tr>
<tr>
<td>TargetName</td>
<td>The name of the current target object.</td>
</tr>
<tr>
<td>TargetPosition</td>
<td>Returns the position of the current target object.</td>
</tr>
<tr>
<td>TargetType</td>
<td>Returns the type of the current target object.</td>
</tr>
<tr>
<td>TargetTypeName</td>
<td>Returns the text name for the type of the current target object.</td>
</tr>
</tbody>
</table>
CallTimer

The CallTimer command returns the current value of the call timer, which indicates the amount of time the agent has been on the current call. The value is returned as a string in hh:mm:ss format. If the agent is not in a guide, 0 (zero) is returned.

FocusCaption

The FocusCaption command returns the caption, if any, of the object with the current input focus. If there is no caption, or if the agent is not in a guide, an empty string (""") is returned.

FocusContent

The FocusContent command returns the contents, if any, of the object with the current input focus. If there are no contents, or if the agent is not in a guide, an empty string (""") is returned.

FocusFormattedValue

The FocusFormattedValue command returns the formatted value, that is the value in external format, of the object with the current input focus; for example, if the object is defined as a date, the value is

---

Table 17-6. EDGE Client Server Commands for Client Request Statements (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TargetValue</td>
<td>Returns the value of the current target object.</td>
</tr>
<tr>
<td>UserID</td>
<td>Returns the current user ID.</td>
</tr>
<tr>
<td>UserName</td>
<td>Returns the name of the current user.</td>
</tr>
</tbody>
</table>
displayed in external date format. If the object is multivalued, value marks are converted to tabs.

If the agent is not in a guide, an empty string ("") is returned.

**FocusName**

The FocusName command returns the name of the object with the current input focus. If the agent is not in a guide, an empty string ("") is returned.

**FocusPosition**

The FocusPosition command returns the position of the first character of the object with the current input focus. The unit of measurement is determined by the current screen metric. The values are in row, column order and are separated by a comma.

If the agent is not in a guide, 0 (zero) is returned.

**FocusType**

The FocusType command returns the type of the object with the current input focus. One of the types listed in Table 17-7 can be returned. If the agent is not in a guide, 0 (zero) is returned.

### Table 17-7. Object Types and Names

<table>
<thead>
<tr>
<th>Type</th>
<th>TypeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>none</td>
</tr>
<tr>
<td>1</td>
<td>text</td>
</tr>
</tbody>
</table>
**FocusType Name**

The `FocusTypeName` command returns the text name for the type of object with the current input focus. One of the names listed in Table 17-7 can be returned.

If the agent is not in a guide, 0 (zero) is returned.

**FocusValue**

The `FocusValue` command returns the value in internal format of the object with the current input focus. If the agent is not in a guide, an empty string ("") is returned.

**ObjectCount**

The `ObjectCount` command returns the number of objects on the current screen. If the agent is not in a guide, 0 (zero) is returned.

---

**Table 17-7. Object Types and Names (continued)**

<table>
<thead>
<tr>
<th>Type</th>
<th>TypeName</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>display field</td>
</tr>
<tr>
<td>4</td>
<td>list box</td>
</tr>
<tr>
<td>5</td>
<td>command button</td>
</tr>
<tr>
<td>6</td>
<td>radio button</td>
</tr>
<tr>
<td>7</td>
<td>check box</td>
</tr>
<tr>
<td>8</td>
<td>selection box</td>
</tr>
</tbody>
</table>
### Port
The Port command returns the current host port connection. If the agent is not logged in, 0 (zero) is returned.

### ProjectName
The ProjectName command returns the current project name. If the agent is not in a guide, an empty string (""") is returned.

### Result
The Result command returns the result code of the last DDE Execute statement.

The value returned to the client Execute statement by DDE generally indicates whether the Execute was successful or failed. If the command failed, DDE does not indicate a reason for the failure. In this case, you can determine the reason by using the Result command in a Request statement.

The status codes that can be returned as a result of the Execute statement are listed in Table 17-8.

#### Table 17-8. Result Codes for the Execute Statements

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No error.</td>
</tr>
<tr>
<td>200</td>
<td>Opening bracket missing.</td>
</tr>
<tr>
<td>201</td>
<td>Closing bracket missing.</td>
</tr>
<tr>
<td>202</td>
<td>Closing parenthesis missing.</td>
</tr>
</tbody>
</table>
**ScreenName**

The ScreenName command returns the name of the current screen. If the agent is not in a guide, an empty string (""") is returned.

**ScreenTitle**

The ScreenTitle command returns the title of the current screen. If the agent is not in a guide, an empty string (""") is returned.

---

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>203</td>
<td>Invalid string.</td>
</tr>
<tr>
<td>204</td>
<td>Too many arguments.</td>
</tr>
<tr>
<td>205</td>
<td>Invalid command name.</td>
</tr>
<tr>
<td>206</td>
<td>Field not found (returned by Target(1)).</td>
</tr>
<tr>
<td>207</td>
<td>Tag not found (returned by Target(3)).</td>
</tr>
<tr>
<td>208</td>
<td>Data string is missing in SetTargetContent.</td>
</tr>
<tr>
<td>300</td>
<td>Closing parenthesis missing (returned by SendKeys only).</td>
</tr>
<tr>
<td>301</td>
<td>String too long (returned by SendKeys only).</td>
</tr>
<tr>
<td>302</td>
<td>Closing brace missing (returned by SendKeys only).</td>
</tr>
<tr>
<td>303</td>
<td>Invalid key (returned by SendKeys only).</td>
</tr>
<tr>
<td>304</td>
<td>Invalid repeat count (returned by SendKeys only).</td>
</tr>
<tr>
<td>305</td>
<td>Out of memory (returned by SendKeys only).</td>
</tr>
</tbody>
</table>
**TabObjectCount**
The TabObjectCount command returns the number of objects that can currently be tabbed to; that is, the objects that can get the focus, have tab stops set, and are visible and enabled. If the agent is not in a guide, 0 (zero) is returned.

**TargetCaption**
The TargetCaption command returns the caption, if any, of the current target object. If there is no caption, or if the agent is not in a guide, an empty string ("") is returned.

**TargetContent**
The TargetContent command returns the contents, if any, of the current target object. If there are no contents, or if the agent is not in a guide, an empty string ("") is returned.

**TargetFormatted Value**
The TargetFormattedValue command returns the formatted value, that is the value in external format, of the current target object. If the object is multivalued, value marks are converted to tabs. If the agent is not in a guide, an empty string ("") is returned.

**TargetName**
The TargetName command returns the name of the current target object. If the agent is not in a guide, an empty string ("") is returned.

**TargetPosition**
The TargetPosition command returns the position of the first character of the current target object. The unit of measurement is determined by the current screen metric. The values are in row, column order and are separated by a comma.
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TargetType</strong></td>
<td>If the agent is not in a guide, 0 (zero) is returned. The TargetType command returns the type of the current target object. One of the types listed in Table 17-7 can be returned. If the agent is not in a guide, 0 (zero) is returned.</td>
</tr>
<tr>
<td><strong>TargetTypeName</strong></td>
<td>The TargetTypeName command returns the text name for the type of the current target object. One of the names listed in Table 17-7 can be returned. If the agent is not in a guide, 0 (zero) is returned.</td>
</tr>
<tr>
<td><strong>TargetValue</strong></td>
<td>The TargetValue command returns the value in internal format of the current target object. If the agent is not in a guide, an empty string (&quot;&quot;) is returned.</td>
</tr>
<tr>
<td><strong>UserID</strong></td>
<td>The UserID command returns the current user ID. If the agent is not in a guide, an empty string (&quot;&quot;) is returned.</td>
</tr>
<tr>
<td><strong>UserName</strong></td>
<td>The UserName command returns the name of the current user. If the agent is not in a guide, an empty string (&quot;&quot;) is returned.</td>
</tr>
</tbody>
</table>
Message Queues

Overview

Message queues are used for interprocess communication, that is, to send and receive messages from external servers. The external server can be on the local machine or on a remote system.

Message queues are accessed through EDGE by using the MSGQUEUE Verb. The MSGQUEUE verb can use a variety of underlying transport mechanisms as I/O devices. The following are supported with the MSGQUEUE verb:

- Internet Sockets.
- Message Queues.
- MQLink. This is an optional module which uses an IBM product called MQSeries™ and requires additional licensing. For more information, see the section Enabling MQLink.
- Named Pipes.
- Old IPC Server.
**Transport Mechanisms**

**Internet Sockets**

Internet sockets are transport mechanisms that are used to communicate with servers that generally are on other machines, although Internet sockets can also be used to communicate with the servers on the current machine. However, because the server can be on another machine, EDGE cannot be used to automatically start the server.

Messages up to 64 Kbytes can be sent using Internet sockets.

A disadvantage to using sockets is that they are not always reliable or fast because of factors outside of an application's control, such as network performance and whether the machine being contacted is running. Also, if a socket connection is broken, it can, in some cases, be several minutes before it is reestablished, blocking communications during that time.

**Message Queues**

Message queues are UNIX transport mechanisms that provide a fairly quick and reliable form of interprocess communication. The IPC SEND and RECEIVE verbs in previous versions of EDGE used this transport mechanism.

There are limitations to using message queues. There is a maximum message size (typically, 2048 bytes), a maximum total size of all mes-
messages on a queue (typically, 4096), a maximum number of message queues that can be created at any one time (typically, 50), and a maximum number of messages at any one time per message queue (typically, 40). These limits are system-dependent and on most machines can be changed only by reconfiguring the kernel.

There is also a problem with cleaning up undelivered messages in the message queues. For example, if an instance of EDGE sends a message to a server but dies before reading the server's response, that response will remain on the queue. Eventually, the queue might be filled with undelivered messages to the point where no more messages can be sent. To prevent this, a message queue server could check for undelivered messages and remove them (on start-up, for instance).

Use the Old IPC Server option to support UNIX IPC message queue servers that work with EDGE 3.6 or earlier. Use the Message Queue option with message queue servers that work with EDGE 3.7.1 or later.

MQLink provides a link between EDGE and IBM's MQSeries product. MQLink is only supported by EDGE on UNIX systems.

MQSeries can guarantee reliable delivery of messages, even if there is a system crash, and it supports transactions. Another consideration
of MQSeries is that message sizes are limited, though the maximum size is user-configurable and large (4 megabytes).

EDGE expects MQSeries to be on the same server as it is. However, the MQSeries on the EDGE server can be configured to communicate across servers and that communication is transparent to EDGE.

MQLink requires additional licensing; for more information, see the section, Enabling MQLink.

MQSeries provides the following advantages:

- Messages can be described as persistent, in which case MQSeries guarantees their delivery. If the system crashes before an application has retrieved a persistent message from a queue, that message will still be available when the system comes back up.

- Transactions, that is, units of work similar to database transactions, are supported. Messages can be put or retrieved from queues as part of a unit of work. If there is an error before this unit of work is committed, the queues are rolled back to their states at the beginning of the unit of work. This does not affect any other changes that may have been made by an application, however. For instance, if a database program is being used and database updates have been made, rolling
back the MQSeries unit of work does not roll back those updates.

- MQSeries can automatically provide information about what has happened to a message after it has been sent. For example, a report message can be sent indicating that a message has been lost due to a transport error or that it has been delivered. Usually reports are sent automatically by MQSeries (if the appropriate values are set), though applications can also send them.

- An MQSeries queue can also be set up with a trigger, which is an asynchronous event caused by a message being placed on a queue. When a trigger is activated, a message describing the trigger event is put on a special queue. An application program monitors this queue, and when a message is put on it, it can do whatever it needs to do, such as run a program. Thus triggers can be used, for example, to start up programs or to perform housekeeping chores. For more information, see the section, Asynchronous Execution of Tasks.

A limitation of MQSeries is that an application can connect with only one input queue manager at a time. An attempt to open an MQSeries I/O device will fail if there is another open device using a different manager for input queues. However, multiple MQSeries I/O devices that use the same manager for input queues can be open at one time.
There is no similar limitation for output queues; that is, an application can open a queue owned by another manager to write data. Therefore, it does not matter which managers own the output queues of any open I/O devices, as long as all the input queues are owned by the same queue manager.

Much of the work of creating an MQSeries I/O device must be done outside of EDGE. For example, setting up queue servers and creating queues cannot be done directly from within EDGE. Also, very few attributes are configurable directly from EDGE. However, MQSeries can allow most things, such as creating a queue, to be done via a script.

EDGE cannot automatically start the MQSeries server.

**Enabling MQLink**

The EDGE MQLink is an optional module and requires additional licensing. This module can be enabled during the installation or upgrade of EDGE, or any time thereafter. It must be enabled before any setup can be done. MQLink requires IBM’s MQSeries product, which must be purchased separately. *MQLink is only supported by EDGE on UNIX systems.*

If you have licensed this product, you are required to contact **EDGE Customer Care** for an authorization password to enable it. Dispensa-
tion of an authorization password requires verification of your license, which may take four to six business hours. Before attempting to enable the MQLink, contact EDGE Customer Care to obtain this password. The authorization password, once you receive it, is only valid for a limited time (approximately 15 minutes); therefore, it should be used immediately.

To run successfully, the module_enable procedure requires a minimum of 20MB of disk space in the temp directory.

To enable MQLink, log in to EDGE. Then go to ACL from the Browser and enter the following:

```
module_enable MQSeries
```

At this point, you are prompted for an authorization password. Enter the authorization password received from EDGE Customer Care.

To determine if MQSeries has already been enabled, select Server from the EDGE Developer toolbar, select the I/O Device tab, then the Add command. Type in a device name, then select the Type pull-down menu. A list of enabled modules is displayed; if MQSeries is enabled, there is an MQSeries type displayed.

**Named Pipes**

Named pipes are UNIX transport mechanisms that provide communication between processes running on the same machine. (These are also referred to as named stream pipes.) They can support mes-
<table>
<thead>
<tr>
<th>Old IPC Server</th>
</tr>
</thead>
</table>
| Messages up to 64 Kbytes. There is no limit to the number of pipes that can be created and no limit to the number of messages per pipe, other than available system resources. *Named pipes are only supported by EDGE on UNIX systems.*
| The Old IPC Server option uses the same underlying message transport system as *Message Queues*. That is, both options use UNIX IPC message queues. Use the Old IPC Server option with message queue servers that work with EDGE 3.6 or earlier. Use the *Message Queue* option with message queue servers that work with EDGE 3.7.1 or later. |
Message Properties

The following sections describe general message properties that are available.

Expiration Times

An expiration time is used to indicate a time limit for how long a message is valid. When an application gets a message, it checks if an expiration has been set. If it has and if the indicated time has expired, the message is discarded.

Message Sizes

EDGE does not limit the size of a message sent by the MSGQUEUE Verb. However, each of the transport mechanisms has a maximum message size. This limit is inherent in the mechanism, not EDGE. If a message that is larger than the mechanism’s maximum is sent through EDGE, EDGE handles it in one of two ways, depending on the mechanism. Sample servers that handle large messages can be found in the $EDGESYS/src/msgqueue directory.

Message Queues and Named Pipes

For message queues and named pipes, which communicate only with the local machine, when a large message is sent, the message is placed in a temporary file and the path of that file is transmitted along with a flag to indicate this is the case. The server has the responsibility of reading the message from the file, then deleting the
file. (To do this, the server must have both read and write permissions on the file.)

The maximum message size for message queues and named pipes then is limited only by the available disk space.

**Internet Sockets and MQSeries**

For Internet sockets and MQSeries, which may communicate across machines, large messages are broken into smaller messages and the smaller messages are transmitted individually. The receiving application needs to join these smaller messages. When the first message piece is received, it should be saved. Then, as subsequent pieces are read, they should each be appended to the message. When all the pieces have been read and appended, the message can be acted on as appropriate.

**Message Types**

Message types are values that are placed at the beginning of a message to identify the message to the application. The application then uses the message as needed. The message types can be in the range 0 (no message type) to 99,999,999. For MQSeries, it is recommended that message types be greater than 65,535, because MQSeries reserves values in the range 1 to 65,535.

This feature must be enabled in the I/O device definition.
EDGE itself creates the following message types; these message types have values greater than 99,999,999:

- **MSG_TYPE_ENTER_GUIDE** This is used to send an enter message to the server on entry to the guide, if specified in the Send enter/exit message on guide entry/exit property in the I/O device.

- **MSG_TYPE_EXIT_GUIDE** This is used to send an exit message to the server on exit from the guide, if specified in the Send enter/exit message on guide entry/exit property in the I/O device.

- **MSG_TYPE_HELLO** This is used to send a HELLO message to the server when an IO device is opened. If a reply is not received, for all device types except message queues, the open will fail. For message queues, if a reply is not received, it is assumed the server is an IPC type; for more information, see the section IPC Server Notes in the upgrade procedures that accompanied this release of EDGE.
I/O Devices

I/O devices are used to specify the transport mechanism. To define an I/O device, select the following:

EDGE / Server area / I/O Dev tab

When you select an existing device or add a new one, a page similar to the following is displayed:
**Properties:**

- **Description**
  A description of the device being defined. This property is optional.

- **Type**
  The type of transport mechanism; can be one of the following:
  - Internet Socket.
  - Message Queue.
  - MQSeries.
  - Named Pipe.
  - Old IPC Server.

  These types are described in the following sections.

**Note**

Although File is displayed in the Type list, this option is intended for use with other modules, such as import and export. This option is *not* intended to be used as a transport mechanism, because the MSGQUEUE commands Receive message and Send message will not function.
**Internet Socket**

When Internet socket is selected as the device type, the following properties are available:

![EDGE (I/O Device Setup) - Untitled](image)

- **Description:**
- **Type:** Internet socket
- **Server pathname:**
- **Remote hostname:**
- **Service name/port number:**
- **Default expiration time:**
- **Default timeout:**
- **Open, send, close in one operation**
- **Allow guide execution when server is not running**
- **Enable user-defined message types**
- **Send enter/exit message on guide entry/exit**
- **Close all open instances**
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow guide execution when server is not running</td>
<td>Specifies whether the guide can execute if the server is not running. If checked, a guide executes whether the server is running or not. If cleared, a guide using the device cannot execute unless the server is running and the device can be opened. This condition can only be enforced if the guide explicitly refers to this IO device. See the IO device property of the MSGQUEUE Verb for for details.</td>
</tr>
<tr>
<td>Close all open instances</td>
<td>Specifies whether the device is to be closed immediately. If checked, a close command closes the device even if it has been opened multiple times. If cleared, the number of times a device has been opened is tracked and the device is not closed until the corresponding number of close commands have been issued.</td>
</tr>
<tr>
<td>Default expiration time</td>
<td>The default expiration time for messages being sent. If blank or 0, the default is set to 60 seconds. If a negative value is specified, messages have no expiration. This value can be overridden in the MSGQUEUE Verb for any message.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default timeout</td>
<td>The default number of seconds EDGE waits when trying to communicate with the server before giving up. If blank or a value of 0 or less is specified, the default is set to 60 seconds. This value can be overridden in the MSGQUEUE verb for any message.</td>
</tr>
<tr>
<td>Enable user-defined message types</td>
<td>If checked, a message type can be included in each message sent using the MSGQUEUE Verb. For a description of message types, see the section Message Types.</td>
</tr>
<tr>
<td>Open, send, close in one operation</td>
<td>If checked, each time a message is sent, the device is opened, the message is sent, and the device is closed. If this is checked and the server generates a reply, the reply is never received. Therefore, it is recommended this be used only if message replies are not expected.</td>
</tr>
<tr>
<td>Remote hostname</td>
<td>The name of remote system to connect to.</td>
</tr>
<tr>
<td>Send enter/exit message on guide entry/exit</td>
<td>If checked, a message is sent whenever the guide is entered or exited. This property is available only if the property Allow guide execution when server is not running is not checked.</td>
</tr>
</tbody>
</table>
Message Queue

When Message queue is selected as the I/O device type, the following pages are available:

- **General Properties**, which is used to define the general characteristics of the device.
- **Server Properties**, which is used to define server characteristics for message queues.

**General Properties**

The Message queue General page is similar to the following:

**Properties: (continued)**

- Service name /port number
- The port number on the remote system. For more information, see your UNIX system administrator.
Message Queues

I/O Devices

EDGE (I/O Device Setup) - Untitled

- **I/O device name**: PROD.UPD

**Description**: 

**Type**: Message queue

**Server pathname**: 

**Remote hostname**: 

**Service name/port number**: 

**Default expiration time**: 

**Default timeout**: 

- [ ] Open, send, close in one operation
- [ ] Allow guide execution when server is not running
- [ ] Enable user-defined message types
- [ ] Send enter/exit message on guide entry/exit
- [ ] Close all open instances
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow guide execution when server is not running</td>
<td>Specifies whether the guide can execute if the server is not running. If checked, a guide is executed whether the server is running or not. If cleared, a guide using the device cannot be executed unless the server is running and the device can be opened. This condition can only be enforced if the guide explicitly refers to this IO device. See the IO device property of the MSGQUEUE Verb for details.</td>
</tr>
<tr>
<td>Close all open instances</td>
<td>Specifies whether the device is to be closed as soon as a close command is received. If checked, a close command closes the device even if it has been opened multiple times. If cleared, the number of times a device has been opened is tracked and the device is not closed until the corresponding number of close commands have been received. The default is cleared.</td>
</tr>
<tr>
<td>Default expiration time</td>
<td>The default expiration time for messages being sent. If blank or 0, the default is set to 60 seconds. If a negative value is specified, messages have no expiration. This value can be overridden in the MSGQUEUE Verb for any message.</td>
</tr>
<tr>
<td>Properties: (continued)</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td></td>
</tr>
<tr>
<td>Default timeout</td>
<td>The default number of seconds EDGE waits when trying to communicate with the server before giving up. If blank or a value of 0 or less is specified, the default is set to 60 seconds. This value can be overridden in the MSGQUEUE Verb for any message.</td>
</tr>
<tr>
<td>Enable user-defined message types</td>
<td>If checked, a message type can be included in each message sent using the MSGQUEUE verb. For a description of message types, see the section Message Types.</td>
</tr>
<tr>
<td>Open, send, close in one operation</td>
<td>If checked, each time a message is sent, the device is opened, the message is sent, and the device is closed. If this is checked and the server generates a reply, the device will need to be reopened to read it, which can degrade performance. Therefore, it is recommended this be used only if message replies are not expected.</td>
</tr>
<tr>
<td>Send enter/exit message on guide entry/exit</td>
<td>If checked, a message is sent whenever the guide is entered or exited. This property is available only if the property Allow guide execution when server is not running is not checked.</td>
</tr>
</tbody>
</table>
Properties: (continued)

Server pathname  The full path on the UNIX host to the server, including the server name. This property is required.

Server Properties

The Message Queues Server page is similar to the following:
Properties:

Start server automatically

If checked, the server is started automatically (if it is not already running) when a guide that uses the server is entered. The following properties are available only if this property is checked:

Command line arguments

The list of arguments to be passed to the server; can be a literal, or a field or variable containing the arguments.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log file</td>
<td>The pathname of a log file; this is passed to the server in the following form:</td>
</tr>
<tr>
<td></td>
<td><code>-l logfile</code></td>
</tr>
<tr>
<td>Maximum log file size (in bytes)</td>
<td>The maximum size of log file; this is passed to the server in the following form:</td>
</tr>
<tr>
<td></td>
<td><code>-M size</code></td>
</tr>
<tr>
<td>Minimum log file size (in bytes)</td>
<td>The minimum size of log file; this is passed to the server in the following form:</td>
</tr>
<tr>
<td></td>
<td><code>-m size</code></td>
</tr>
<tr>
<td>Server environment setup file path</td>
<td>The name of a file containing environment variable definitions. EDGE executes this file to set the environment before running the server.</td>
</tr>
<tr>
<td>Standard error file</td>
<td>The pathname to which the UNIX standard error is to be redirected.</td>
</tr>
<tr>
<td>Standard output file</td>
<td>The pathname to which the UNIX standard output is to be redirected.</td>
</tr>
</tbody>
</table>
When MQSeries is selected as the I/O device type, the following pages are available:

- **General Properties**, which is used to define the general characteristics of the device.
- **MQSeries Properties**, which is used to define specific characteristics for MQSeries, such as queues.

**General Properties**

The MQSeries General page is similar to the following:
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow guide execution when server is not running</td>
<td>Specifies whether the guide can execute if the server is not running. If checked, a guide is executed whether the server is running or not. If cleared, a guide using the device cannot be executed unless the server is running and the device can be opened. This condition can only be enforced if the guide explicitly refers to this IO device. See the IO device property of the MSGQUEUE Verb for details.</td>
</tr>
<tr>
<td>Close all open instances</td>
<td>Specifies whether the device is to be closed as soon as a close command is received. If checked, a close command closes the device even if it has been opened multiple times. If cleared, the number of times a device has been opened is tracked and the device is not closed until the corresponding number of close commands have been received.</td>
</tr>
<tr>
<td>Default expiration time</td>
<td>The default expiration time for messages being sent. If blank or 0, the default is set to 60 seconds. If a negative value is specified, messages have no expiration. This value can be overridden in the MSGQUEUE Verb for any message.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default timeout</td>
<td>The default number of seconds EDGE waits when trying to communicate with the server before giving up. If blank or a value of 0 or less is specified, the default is set to 60 seconds. This value can be overridden in the MSGQUEUE verb for any message.</td>
</tr>
<tr>
<td>Enable user-defined message types</td>
<td>If checked, a message type can be included in each message sent using the MSGQUEUE Verb. For a description of message types, see the section Message Types.</td>
</tr>
<tr>
<td>Open, send, close in one operation</td>
<td>If checked, each time a message is sent, the device is opened, the message is sent, and the device is closed. If this is checked and the server generates a reply, the device will need to be reopened to read it, which can degrade performance. Therefore, it is recommended this be used only if message replies are not expected.</td>
</tr>
<tr>
<td>Send enter/exit message on guide entry/exit</td>
<td>If checked, a message is sent whenever the guide is entered or exited. This property is available only if the property Allow guide execution when server is not running is not checked.</td>
</tr>
</tbody>
</table>
MQSeries Properties

The MQSeries page is used to specify properties unique to the MQSeries.

Properties:

- Allow message transactions
  Specifies whether message transactions are used; to include a message in a transaction, use the MSGQUEUE Verb.
## Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow user to set identity context</td>
<td>If checked, the MQSeries identity context field can be set for each message sent using the MSGQUEUE verb. If cleared, the identity context for each message is automatically filled with the EDGE login ID.</td>
</tr>
<tr>
<td>Enable user-defined message IDs</td>
<td>If checked, message IDs can be specified for each message sent using the MSGQUEUE Verb.</td>
</tr>
<tr>
<td>Generate delivered reports</td>
<td>If checked, MQSeries generates a report that indicates a message has been taken off the destination queue</td>
</tr>
<tr>
<td>Generate exception reports</td>
<td>If checked, MQSeries generates a report if a message that was sent cannot be read because of an error.</td>
</tr>
<tr>
<td>Generate expiration reports</td>
<td>If checked, MQSeries generates a report if a message that was sent cannot be read because it has expired.</td>
</tr>
<tr>
<td>Generate queue arrival reports</td>
<td>If checked, MQSeries generates a report that indicates a message has been placed on the proper destination queue.</td>
</tr>
<tr>
<td>Input MQSeries queue name</td>
<td>The name of the MQSeries queue to use for input. This must be created outside of EDGE. To use a dynamic queue, specify the name of the model queue. For more information on dynamic queues, see the MQSeries documentation.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input MQSeries queue manager name</td>
<td>The name of the MQSeries queue manager to use for input queues. This must be created outside of EDGE. This is optional; if not specified, the default queue manager is used.</td>
</tr>
<tr>
<td>Output MQSeries queue name</td>
<td>The name of the MQSeries queue to use for output. This must be created outside of EDGE.</td>
</tr>
<tr>
<td>Output MQSeries queue manager name</td>
<td>The name of the MQSeries queue manager to use for output queues. This must be created outside of EDGE. This is optional; if not specified, the manager specified in the <code>Input MQSeries queue manager</code> property is used.</td>
</tr>
</tbody>
</table>

### Notes

If either Generate delivered reports or Generate queue arrival reports is specified, sending one message results in two reply messages being returned; if both are specified, three replies are returned:

- The delivered and/or queue arrival report messages.
- The actual reply.

Tasks and logic flows that use MQSeries devices with reports must handle this.
When Named pipe is selected as the I/O device type, the following pages are available:

- **General Properties**, which is used to define the general characteristics of the device.
- **Server Page**, which is used to define server characteristics for named pipes.

**General Properties**

The Named Pipe General page is similar to the following:
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow guide execution when server is not running</td>
<td>Specifies whether the guide can execute if the server is not running. If checked, a guide is executed whether the server is running or not. If cleared, a guide using the device cannot be executed unless the server is running and the device can be opened. This condition can only be enforced if the guide explicitly refers to this IO device. See the IO device property of the MSGQUEUE Verb for details.</td>
</tr>
<tr>
<td>Close all open instances</td>
<td>Specifies whether the device is to be closed as soon as a close command is received. If checked, a close command closes the device even if it has been opened multiple times. If cleared, the number of times a device has been opened is tracked and the device is not closed until the corresponding number of close commands have been received.</td>
</tr>
<tr>
<td>Default expiration time</td>
<td>The default expiration time for messages being sent. If blank or 0, the default is set to 60 seconds. If a negative value is specified, messages have no expiration. This value can be overridden in the MSGQUEUE Verb for any message.</td>
</tr>
<tr>
<td>Properties: (continued)</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Default timeout</td>
<td>The default number of seconds EDGE waits when trying to communicate with the server before giving up. If blank or a value of 0 or less is specified, the default is set to 60 seconds. This value can be overridden in the MSGQUEUE Verb for any message.</td>
</tr>
<tr>
<td>Enable user-defined message types</td>
<td>If checked, a message type can be included in each message sent using the MSGQUEUE verb. For a description of message types, see the section Message Types.</td>
</tr>
<tr>
<td>Open, send, close in one operation</td>
<td>If checked, each time a message is sent, the device is opened, the message is sent, and the device is closed. If this is checked and the server generates a reply, the reply is never received. Therefore, it is recommended this be used only if message replies are not expected.</td>
</tr>
<tr>
<td>Send enter/exit message on guide entry/exit</td>
<td>If checked, a message is sent whenever the guide is entered or exited. This property is available only if the property Allow guide execution when server is not running is not checked.</td>
</tr>
<tr>
<td>Server pathame</td>
<td>The full path on the UNIX host to the server, including the server name. This property is required.</td>
</tr>
</tbody>
</table>
Server Page

The Named Pipe Server page is similar to the following:

Properties:

Connect path-name

The connection point to the server. This is a file (actually a pipe) that must be created by the server before the I/O device is opened. The EDGE process must have read/write permissions on this file. This property is required.
**Properties: (continued)**

- **Start server automatically**
  If checked, the server is started automatically (if it is not already running) when a guide that uses the server is entered. The following properties are available only if this property is checked.

- **Command line arguments**
  The list of arguments to be passed to the server; can be a literal, or a field or variable that contains the arguments.

- **Log file**
  The pathname of a log file; this is passed to the server in the following form:

  ```
  -l logfile
  ```

- **Maximum log file size (in bytes)**
  The maximum size of log file; this is passed to the server in the following form:

  ```
  -M size
  ```

- **Minimum log file size (in bytes)**
  The minimum size of log file; this is passed to the server in the following form:

  ```
  -m size
  ```
**Old IPC Server**

The following pages are available when you select the Old IPC Server option as the I/O device type:

- **General Properties**, which is used to define the general characteristics of the device.
- **Server Properties**, which is used to define server characteristics for UNIX IPC message queues.

---

**Properties: (continued)**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server environment setup file path</td>
<td>The name of a file containing environment variable definitions. EDGE uses this file to set the environment before running the server.</td>
</tr>
<tr>
<td>Standard error file</td>
<td>The pathname to which the UNIX standard error is to be redirected.</td>
</tr>
<tr>
<td>Standard output file</td>
<td>The pathname to which the UNIX standard output is to be redirected.</td>
</tr>
</tbody>
</table>
**General Properties**

The Old IPC Server General page is similar to the following:
<table>
<thead>
<tr>
<th>Properties:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allow guide execution when server is not running</td>
</tr>
<tr>
<td>Specifies whether the guide can execute if the server is not running. If checked, a guide is executed whether the server is running or not. If cleared, a guide using the device cannot be executed unless the server is running and the device can be opened. This condition can only be enforced if the guide explicitly refers to this IO device. See the IO device property of the MSGQUEUE Verb for details.</td>
</tr>
<tr>
<td>Enable user-defined message types</td>
</tr>
<tr>
<td>If checked, a message type can be included in each message sent using the MSGQUEUE verb. For a description of message types, see the section Message Types.</td>
</tr>
<tr>
<td>Server pathname</td>
</tr>
<tr>
<td>The full path on the UNIX host to the server, including the server name. This property is required.</td>
</tr>
</tbody>
</table>
Server Properties

The Server properties page for the Old IPC Server option is similar to the following:
Properties:

Start server automatically

If checked, the server is started automatically (if it is not already running) when a guide that uses the server is entered. The following properties are available only if this property is checked:

Command line arguments
The list of arguments to be passed to the server; can be a literal, or a field or variable containing the arguments.

Log file
The pathname of a log file; this is passed to the server in the following form:

-1 logfile

Maximum log file size (in bytes)
The maximum size of log file; this is passed to the server in the following form:

-M size

Minimum log file size (in bytes)
The minimum size of log file; this is passed to the server in the following form:

-m size
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server environment setup file path</td>
<td>The name of a file containing environment variable definitions. EDGE executes this file to set the environment before running the server.</td>
</tr>
<tr>
<td>Standard error file</td>
<td>The pathname to which the UNIX standard error is to be redirected.</td>
</tr>
<tr>
<td>Standard output file</td>
<td>The pathname to which the UNIX standard output is to be redirected.</td>
</tr>
</tbody>
</table>
**MSGQUEUE Verb**

The MSGQUEUE verb can be used for the following in both logic flows and tasks:

- Close I/O Device.
- Commit Transaction.
- Get Attributes.
- Open I/O Device.
- Rollback Transaction.
- Receive Message.
- Send Message.
- Set Attributes.
- Start Transaction.

**Common Properties**

These properties are common to all commands. The properties specific to a command are described in later sections for each subcommand. The MSGQUEUE property page is similar to the following:
### MSGQUEUE Verb

#### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command</td>
<td>The command to invoke; can be one of the following:</td>
</tr>
<tr>
<td>Using I/O device</td>
<td></td>
</tr>
<tr>
<td>Seconds to wait</td>
<td></td>
</tr>
<tr>
<td>Device component</td>
<td></td>
</tr>
<tr>
<td>Get attributes</td>
<td></td>
</tr>
<tr>
<td>Store values in</td>
<td></td>
</tr>
<tr>
<td>Set attributes</td>
<td></td>
</tr>
<tr>
<td>To the values</td>
<td></td>
</tr>
<tr>
<td>Match message ID</td>
<td></td>
</tr>
<tr>
<td>Include in open trans</td>
<td></td>
</tr>
<tr>
<td>Store message in</td>
<td></td>
</tr>
<tr>
<td>Store type of message in</td>
<td></td>
</tr>
<tr>
<td>Store ID of message in</td>
<td></td>
</tr>
<tr>
<td>Store correlation ID in</td>
<td></td>
</tr>
<tr>
<td>Store report reason in</td>
<td></td>
</tr>
<tr>
<td>Store format in</td>
<td></td>
</tr>
<tr>
<td>Store accounting token in</td>
<td></td>
</tr>
<tr>
<td>Store app identity data in</td>
<td></td>
</tr>
<tr>
<td>Store user ID in</td>
<td></td>
</tr>
<tr>
<td>File contains message</td>
<td></td>
</tr>
<tr>
<td>Message</td>
<td></td>
</tr>
<tr>
<td>File pathname</td>
<td></td>
</tr>
<tr>
<td>With type of message</td>
<td></td>
</tr>
<tr>
<td>With expiration time</td>
<td></td>
</tr>
<tr>
<td>With ID of message</td>
<td></td>
</tr>
<tr>
<td>With correlation ID</td>
<td></td>
</tr>
<tr>
<td>With format</td>
<td></td>
</tr>
<tr>
<td>With accounting token</td>
<td></td>
</tr>
<tr>
<td>With app identity data</td>
<td></td>
</tr>
<tr>
<td>With user ID</td>
<td></td>
</tr>
<tr>
<td>As a persistent message</td>
<td></td>
</tr>
<tr>
<td>Include in open trans</td>
<td></td>
</tr>
<tr>
<td>Store status code in</td>
<td></td>
</tr>
<tr>
<td>Store status text in</td>
<td></td>
</tr>
<tr>
<td>Store device status code</td>
<td></td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Close I/O Device</td>
<td></td>
</tr>
<tr>
<td>Commit Transaction</td>
<td></td>
</tr>
<tr>
<td>Get Attributes</td>
<td></td>
</tr>
<tr>
<td>Open I/O Device</td>
<td></td>
</tr>
<tr>
<td>Rollback Transaction</td>
<td></td>
</tr>
<tr>
<td>Receive Message</td>
<td></td>
</tr>
<tr>
<td>Send Message</td>
<td></td>
</tr>
<tr>
<td>Set Attributes</td>
<td></td>
</tr>
<tr>
<td>Store device status code in</td>
<td>The location in which to store the device-specific status code; can be a field name or variable. If there is no device-specific code, a 0 (zero) is returned. This is valid only if the status code from the command indicates an error. For a list of device status codes, see the documentation for the device.</td>
</tr>
<tr>
<td>Store status code in</td>
<td>The location in which to store the results of the command; can be a field name or variable. For a list of status codes, see Table 18-1.</td>
</tr>
<tr>
<td>Store status text in</td>
<td>The location in which to store the text associated with the status code; can be a field name or variable.</td>
</tr>
</tbody>
</table>
Properties: (continued)

Using I/O device The name of the I/O device; can be a literal, or a field or variable containing the name of the IO device. Note: if this value is not a literal references to this IO device cannot be used to enforce the Allow guide execution when server not running property of the IO device definition.

The other properties vary by command and are described with the appropriate command.

Table 18-1. Status Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>The operation functioned properly.</td>
<td>All.</td>
</tr>
<tr>
<td>-9010</td>
<td>Edge could not allocate memory.</td>
<td>All.</td>
</tr>
<tr>
<td>-9011</td>
<td>Device not open.</td>
<td>Close I/O device.</td>
</tr>
<tr>
<td>-9015</td>
<td>Error closing device.</td>
<td>Close I/O device.</td>
</tr>
</tbody>
</table>
### Table 18-1. Status Codes (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>-9018</td>
<td>I/O device not open. (MQN).</td>
<td>Commit transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Get attributes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Rollback transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set attributes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start transaction.</td>
</tr>
<tr>
<td>-9019</td>
<td>More attributes specified than values.</td>
<td>Set attributes.</td>
</tr>
<tr>
<td>-9020</td>
<td>Invalid attribute specified.</td>
<td>Get attributes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set attributes.</td>
</tr>
<tr>
<td>-9021</td>
<td>File to send as message not found.</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9025</td>
<td>Timed out.</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive message.</td>
</tr>
<tr>
<td>-9026</td>
<td>Semaphore locked error. (Named pipe),</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9027</td>
<td>Error starting server. (Named pipe),</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9028</td>
<td>Message too large to be sent.</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9032</td>
<td>Bad device path. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9033</td>
<td>Bad parameters. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9038</td>
<td>Message type out of range.</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9040</td>
<td>Piece of a message arrived out of sequence. (Internet socket or MQSeries).</td>
<td>Receive message.</td>
</tr>
<tr>
<td>-9041</td>
<td>Invalid message piece. (Internet socket or MQSeries).</td>
<td>Receive message.</td>
</tr>
</tbody>
</table>
**Table 18-1. Status Codes (continued)**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>-9042</td>
<td>Size of message piece not valid. (Internet socket or MQSeries).</td>
<td>Receive message.</td>
</tr>
<tr>
<td>-9045</td>
<td>Error reading file to send as message. (Internet socket or MQSeries)</td>
<td>Receive message.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Send message.</td>
</tr>
<tr>
<td>-9046</td>
<td>Error creating temporary message file. (Message queue or named pipe)</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9100</td>
<td>Neither an input or output queue defined. (MQSeries).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Open I/O device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set attributes.</td>
</tr>
<tr>
<td>-9107</td>
<td>Could not open queue for output. (MQSeries).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9108</td>
<td>Could not open queue manager. (MQSeries).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9109</td>
<td>Error sending message. (MQSeries).</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9111</td>
<td>Error getting attributes. (MQSeries).</td>
<td>Get attributes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Set attributes.</td>
</tr>
<tr>
<td>-9112</td>
<td>Attempt to use transactions with a device that does not support them. (MQSeries).</td>
<td>Receive message.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Send message.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Commands</td>
</tr>
<tr>
<td>-------</td>
<td>-----------------------------------------------------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>-9114</td>
<td>Error rolling back transaction. (MQSeries).</td>
<td>Rollback transaction.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Start transaction.</td>
</tr>
<tr>
<td>-9116</td>
<td>Attempt to send a persistent message using a temporary dynamic queue.</td>
<td>Send message.</td>
</tr>
<tr>
<td></td>
<td>(MQSeries).</td>
<td></td>
</tr>
<tr>
<td>-9119</td>
<td>MQSeries not enabled; run module_enable. (MQSeries).</td>
<td>All</td>
</tr>
<tr>
<td>-9200</td>
<td>Bad connect path. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9201</td>
<td>Binding failed. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9202</td>
<td>Change mode failed. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9204</td>
<td>Connect failed. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9206</td>
<td>Socket failed. (Named pipe).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9207</td>
<td>Error sending message. (Message queue or named pipe).</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9209</td>
<td>Error setting attribute. (Named pipe).</td>
<td>Set attributes.</td>
</tr>
<tr>
<td>-9300</td>
<td>Error reading or sending message. (Message queue).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive message.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Send message.</td>
</tr>
</tbody>
</table>
### Table 18-1. Status Codes (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>-9301</td>
<td>Invalid IPC type. (Message queue).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive message.</td>
</tr>
<tr>
<td>-9302</td>
<td>Error in connecting. (Message queue).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9303</td>
<td>Queue size is invalid. (Message queue).</td>
<td>Receive message.</td>
</tr>
<tr>
<td>-9400</td>
<td>Error reading socket address. (Internet socket).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9404</td>
<td>Error in connecting socket. (Internet socket).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9405</td>
<td>Error connecting to other machine. (Internet socket).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9406</td>
<td>Error in opening. (Internet socket).</td>
<td>Open I/O device.</td>
</tr>
<tr>
<td>-9407</td>
<td>Error reading message. (Internet socket).</td>
<td>Receive message.</td>
</tr>
<tr>
<td>-9408</td>
<td>Error sending message. (Internet socket).</td>
<td>Send message.</td>
</tr>
<tr>
<td>-9409</td>
<td>Error getting attribute. (Internet socket).</td>
<td>Get attributes.</td>
</tr>
<tr>
<td>-9410</td>
<td>Error setting attribute. (Internet socket).</td>
<td>Set attributes.</td>
</tr>
</tbody>
</table>
# MSQUEUE Verb Commands

## Close I/O Device

The Close I/O device command closes an I/O device.

**Properties:**

There are no properties specific to this command; see Common Properties.

## Commit Transaction

This command commits a transaction. This command is valid only for an I/O device type of MQSeries. (For information on transactions, see the section MQLink.)

**Properties:**

There are no properties specific to this command; see Common Properties.
**Get Attributes**

The Get attributes command retrieves information about specified attributes for the I/O device.

**Properties:**

- **Device component**
  - The type of component for which attribute values are to be returned; can be a literal, or a field name or variable that contains the component type. This property is valid only with MQSeries and message queues; it is ignored for other device types. One of the following types can be specified:
    - I Input queue.
    - M Queue manager (MQSeries only).
    - O Output queue.

- **Get attributes**
  - The attribute whose value is to be retrieved; can be a literal, or a field name or variable containing the attribute. The attribute must be specified by its numeric value, not as a named constant; for example, the MQSeries default queue persistence attribute is defined as follows:
    
    ```c
    #define MQIA_DEF_PERSISTENCE 5
    ```
  - To retrieve the value of this attribute, specify 5.
Properties: (continued)

For MQSeries devices only, values for multiple attributes can be retrieved with one call by using a multivalued field or variable and specifying each attribute in a value. If an attempt is made to read multiple device attribute values for any other device type, only the first attribute value is retrieved.

The names of the attributes depend on the type of I/O device as follows:

- **Internet sockets**
  - The attributes for sockets are listed in the system include file `sys/socket.h` and detailed in the `getsockopt(2)` manual page.

- **Message queues**
  - The attributes for message queues are listed in Table 18-2.

- **MQSeries**
  - The attributes for MQSeries are listed in the *MQSeries for AIX Application Programming Reference*. They are also defined in the MQSeries include file `cmqc.h`. 
Properties: (continued)

Named pipes

The attributes for Named pipes are the same as those listed for Internet sockets and are listed in the system include file `sys/socket.h` and detailed in the `getsockopt(2)` manual page.

Store value in

The location in which to store the value that was retrieved. If using MQSeries and multiple attributes were specified, the results are returned as multivalues corresponding to the request.

Table 18-2. Message Queue Attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Current number of bytes on the queue.</td>
</tr>
<tr>
<td>11</td>
<td>Current number of messages on the queue.</td>
</tr>
<tr>
<td>12</td>
<td>Maximum number of bytes on the queue.</td>
</tr>
<tr>
<td>13</td>
<td>PID of last process to put a message on the queue.</td>
</tr>
<tr>
<td>14</td>
<td>PID of last process to retrieve a message from the queue.</td>
</tr>
<tr>
<td>15</td>
<td>Time last message put on the queue.</td>
</tr>
<tr>
<td>16</td>
<td>Time last message retrieved from queue.</td>
</tr>
<tr>
<td>17</td>
<td>Time of last change.</td>
</tr>
</tbody>
</table>
Open I/O Device

The Open I/O device commands opens the specified device.

Properties:

Seconds to wait  The time to wait for the connection to open; can be a literal, or a field or variable containing the time. If not specified, the default timeout specified in the I/O device definition is used.

Note

For message queues, to keep EDGE from checking if the server is running when opening the device, set the environment variable MSGQ_SKIP_SRVPID_CHECK to "Y". An error is only returned if there are no message queues associated with the I/O device. The environment variable is typically set in the .user.profile file found in the EDGE home directory, using the following sample syntax:

export MSGQ_SKIP_SRVPID_CHECK; MSGQ_SKIP_SRVPID_CHECK=Y

If it is not set, XGUIDE will send a HELLO message to check if the server is running when opening the I/O device.

AIT recommends setting the MSGQ_SKIP_SRVPID_CHECK variable to Y if the customer’s IPC servers are "transient", i.e., the servers start up,
**Receive Message**  

The Receive message command receives a command.

**Properties:**

- **Include in transaction**
  - Specifies whether to include in the currently open transaction; can be one of the following:
    - **False**  Message is not included in the transaction.
    - **True**   Message is included.
  - This is valid only for MQSeries I/O devices; it is ignored for other device types.

- **Match message ID**
  - The ID of the message to retrieve; can be a literal, or a field or variable containing the ID. This is valid only for MQSeries I/O devices; it is ignored for other device types.

- **Seconds to wait**
  - The time to wait for the message to be received; can be a literal, or a field or variable containing the time. If not specified, the default timeout specified in the I/O device definition is used.

Read/write from queues and go away, then other servers read/write from the same queues and go away, and so on.

AIT recommends not setting the MSGQ_SKIP_SRVPID_CHECK variable if there is one server which creates the queue, services it, and when it is shut down, deletes the queue.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Store accounting token in</td>
<td>The location in which to store the accounting token; can be a field or variable. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>Store app identity data in</td>
<td>The location in which to store the app identity data; can be a field or variable. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>Store correlation ID in</td>
<td>The location in which to store the correlation ID; can be a field or variable. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>Store format in</td>
<td>The location in which to store the format; can be a field or variable. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>Store message in</td>
<td>The location in which to store the message; can be a field or variable.</td>
</tr>
<tr>
<td>Store message type in</td>
<td>The location in which to store the message type, if any; can be a field or variable.</td>
</tr>
<tr>
<td>Store report reason in</td>
<td>The location in which to store the report reason; can be a field or variable. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>Store user ID in</td>
<td>The location in which to store the user ID; can be a field or variable. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
</tbody>
</table>
### Rollback Transaction

The Rollback transaction command rolls back the current transaction for the specified I/O device. This command is valid only for an I/O device type of MQSeries. (For information on transactions, see the section [MQLink](#).)

**Properties:**

There are no properties specific to this command; see [Common Properties](#).

### Send Message

The Send message command sends a message using the specified I/O device.

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>File contains message</td>
<td>Specifies whether the message is in a UNIX file or in EDGE; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>False</td>
</tr>
<tr>
<td></td>
<td>True</td>
</tr>
<tr>
<td>Message</td>
<td>The message to be sent; can be a literal , or a field or variable containing the message.</td>
</tr>
<tr>
<td></td>
<td>This property is enabled only if the <em>File contains message</em> property is False.</td>
</tr>
<tr>
<td>File pathname</td>
<td>The path of the file containing the message to be sent; can be a literal , or a field or variable containing the path. This property is enabled only if the <em>File contains message</em> property is True.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Seconds to wait</td>
<td>The time to wait for the message to be sent; can be a literal, or a field or variable containing the time. If not specified, the default timeout specified in the I/O device definition is used.</td>
</tr>
<tr>
<td>With type of message</td>
<td>The message type; can be a literal, or a field or variable containing the message type.</td>
</tr>
<tr>
<td>With expiration time</td>
<td>The expiration time; can be a literal, or a field or variable containing the time. If a negative value is specified, messages have no expiration.</td>
</tr>
<tr>
<td>With ID of message</td>
<td>The message ID; can be a literal or a field or variable containing the ID. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>With correlation ID</td>
<td>The correlation ID to be sent; can be a literal, or a field or variable containing the ID. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>With format</td>
<td>The message format; can be a literal, or a field or variable containing the format. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
<tr>
<td>With accounting token</td>
<td>The accounting token to be sent; can be a literal, or a field or variable containing the token. This is valid only for MQSeries I/O devices; it is ignored for other device types.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

- **With app identity data**
  The app identity to be sent; can be a literal, or a field or variable containing the identity. This is valid only for MQSeries I/O devices; it is ignored for other device types.

- **With user ID**
  The user ID; can be a literal, selected from the list, or a field or variable containing the user ID. This is valid only for MQSeries I/O devices and only if **Allow user to set identity context** is checked; it is ignored for other device types.

- **As a persistent message**
  Specifies whether the message is persistent; can be one of the following:
  - No
  - Use queue default
  - Yes
  This is valid only for MQSeries I/O devices; it is ignored for other device types. (For information on persistent messages, see the section **MQLink**.)

- **Include in transaction**
  Specifies whether to include the message in the currently open transaction; can be one of the following:
  - False Message is not included in the transaction.
  - True Message is included.
### Set Attributes

**Properties:** (continued)

This is valid only for MQSeries I/O devices; it is ignored for other device types.

The Set attributes command sets values for specified attributes in the I/O device.

**Properties:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device component</td>
<td>The type of component for which attributes are to be set; can be a literal, or a field name or variable that contains the component type. This property is valid only with MQSeries and message queues; it is ignored for other device types. One of the following types can be specified:</td>
</tr>
<tr>
<td></td>
<td>I Input queue.</td>
</tr>
<tr>
<td></td>
<td>M Queue manager (MQSeries only).</td>
</tr>
<tr>
<td></td>
<td>O Output queue.</td>
</tr>
<tr>
<td>Set attributes</td>
<td>The attribute to set; can be a literal, or a field name or variable containing the attribute. The attribute must be specified by its numeric value, not as a named constant; for example, the MQSeries default queue persistence attribute is defined as</td>
</tr>
<tr>
<td></td>
<td>#define MQIA_DEF_PERSISTENCE 5</td>
</tr>
<tr>
<td></td>
<td>To set this attribute, specify 5.</td>
</tr>
</tbody>
</table>
For MQSeries devices only, multiple attributes can be set with one call by using a multivalued field or variable and specifying each attribute in a value. If an attempt is made to set multiple device attribute values for any other device type, only the first attribute value is set.

The names of the attributes depend on the type of I/O device as follows:

- **Internet sockets**
  - The attributes for sockets are listed in the system include file `sys/socket.h` and detailed in the `getsockopt(2)` manual page.

- **Message queues**
  - Attributes cannot be set for message queues.

- **MQSeries**
  - The attributes for MQSeries are listed in the *MQSeries for AIX Application Programming Reference*. They are also defined in the MQSeries include file `cmqc.h`.

- **Named pipes**
  - The attributes for Named pipes are the same as those listed for Internet sockets and are listed in the system include file `sys/socket.h` and detailed in the `getsockopt(2)` manual page.
**Start Transaction**

The Start transaction command specifies the beginning of a unit or work for the specified I/O device. This command is valid only for an I/O device type of MQSeries. (For information on transactions, see the section *MQLink*.)

**Properties:**

There are no properties specific to this command; see Common Properties.
Asynchronous Execution of Tasks

Tasks can be run asynchronously by using the trigger features of the MQSeries type of message queues.

Note: This is an optional feature.

MQSeries Triggers

An MQSeries queue can be set up as a trigger queue. Then, when one of the following events occurs, the trigger is activated:

- A queue receiving its first message.
- A queue receiving any message.
- A queue reaching a certain size.

To use triggers, two queues are needed: the trigger queue and an initiation queue.

When a trigger is activated, a message describing the event is put on the initiation queue. This causes a program, called the monitor program, to be invoked. This program normally just starts a user application which then handles the event that originally caused the trigger. MQSeries provides a monitor program, though a user-defined monitor can also be used. (The monitor program is basically just a server that continually monitors an initiation queue, waiting for a message
to placed on it.) This process is detailed in the MQSeries Message Queue Interface Technical Reference.

Certain information, such as the name of the trigger queue, is included in the message the monitor program reads from the initiation queue (and normally passes to the user application). User data placed in a specified attribute of the trigger queue is also included, as are values specified in certain attributes of the so-called process definition a trigger queue must have. (A definition simply indicates which application program should be executed when a trigger event occurs.)

The concepts discussed in the previous section can be used to provide the ability to run EDGE tasks asynchronously. A sample program, called run_task, is provided in $EDGESYS/src/run_task. This program shows how to set the necessary environment variables and execute a task. You can use this program as a starting point in developing a monitor program and an application.

When a monitor program is activated because of a trigger event, it has no way of knowing which EDGE task should be executed. This information cannot be part of the user data discussed in the previous section because that data is contained in queue and process definition attributes, most of which can not be modified during program execution. With this in mind, there are two methods you can use to run EDGE tasks asynchronously.
The first method is to use a different trigger queue for each task, since the name of the trigger queue is part of the information provided to the monitor program. In this case, you could modify run_task to use the name of the trigger queue to determine which task to run, set the needed environment variables, then execute the task. In fact, the original message that caused the trigger event would not even need to be read. (And since messages can have an Expiry time, there would be no worry about these trigger messages staying on the queues forever.)

The second method is for the message that caused the trigger event to contain the name of the task. In this case, you could modify run_task to get that message from the trigger queue to determine which task to run, set the needed environment variables, then execute the task.

The first method is slightly faster, since reading one message can be avoided, but the second method would be more flexible. Either way, though, if you use EDGE with MQSeries, you have at least two methods to enable asynchronous running of tasks.

Whichever method is used, run_task can be modified to become either the monitor program or the user application. The only difference is that if used as a monitor program, it will need to fork itself before executing a task (so that there is still a process to continue monitoring the initiation queue). Currently, run_task is written as a
user application; that is, it executes a task without forking. (The reason for this is that, as mentioned above, MQSeries provides a monitor program.

When a task is executed, it expects the following environment variables to be set:

- EDGE
- EDGESYS
- HOME
- NAME
- EBF0
- EPB0
- ERS0
- TMPDIR

If the task is run from within EDGE, either directly as a task or using the task scheduler, the environment variables have already been set. However, when a task is executed using asynchronous messages, it runs outside of EDGE. In this case, these environment variables must be manually set first.
Port ID

When a task is ran asynchronously, it must have a unique port associated with it because some commands use it for critical things such as record locking. When a task is run interactively, this is not an issue, as the user is assigned a port automatically when he logs on to EDGE. When a task is ran in the background by the scheduler, EDGE generates a phantom port. A similar method will be used when running a task asynchronously.

Like tasks executed by the task scheduler, asynchronous tasks should not use the ASK verb or any other verb which requires user input because there is no user interacting with the task.
Screens are the interface between the agents and the application. Screens are used, for example, to present data entry forms, to give messages to the agent, to review customer records, or for any other purpose that you need to address.

In addition to the screens you develop, EDGE uses its own set of screens (called EDGE screens) in every project. Although you cannot change the functionality of EDGE screens, you can change the screen title and function key descriptions that are displayed. For more information on EDGE screens, see Chapter 23.

This chapter describes the following:

- Screen Appearance.
- Design Considerations for Screens.
- Creating and Modifying Screens.
- Screen Editor.
• Screen Properties.
• Selection Screen Properties.
• Screen Menu.
• Screen Tools.
Screen Appearance

The appearance of screens in the design phase is different from what is in the operations phase. Also the appearance of screens in the operations phase depends on whether the agent is viewing the screen from a PC using EDGE Client or from an ASCII terminal using ASCII mode.

For information on the appearance of screens during the design phase, see the section Creating and Modifying Screens.

During operations, each screen presented to the agent has the following sections:

- Function Keys.
- Mode information (that is, Live or Test mode).
- Screen Name.
- Screen Title.
- Screen-specific information for the agent.

Figure 19-1 shows a sample screen as viewed in EDGE Client.
Figure 19-1. Freeform Screen as Viewed in EDGE Client
Design Considerations for Screens

The following is a list of things to consider before designing your screen:

- Deciding on Screen Type.
- Deciding on Objects and Fields.
- Determining Files.
- Determining Screen Size and Metric.
- Determining Styles.
- Determining Results.
- Determining Logic Flows for Screen Events.
- Determining Screen Routing and the Window Helper.

Deciding on Screen Type

EDGE provides the following two screen types:

- Freeform Screens.
- Selection Screens.

The freeform screen is generally used for such tasks as data entry, text display, and sales history.
Deciding on Objects and Fields

The selection screen is used to present a list from which the agent makes selections. It is generally used for such tasks as surveys, or to provide a set of menu selections for the agents.

An object is something that can be touched or seen. In EDGE, objects are things that can be placed on screens and hence can be seen. For example, a command button is a screen object. We recommend you do not create screens with more than 200 objects. More than 200 objects on a screen may affect system performance.

Each object has an associated field that supplies the values for the object.

All Screen Objects are functional from EDGE Client, but not all objects are functional from ASCII. Therefore, the hardware agents use will determine the types of objects you should use to design your screens.

The following objects are not functional in the ASCII operations environment:

- ActiveX Controls.
- Check Box.
- Command Button.
- Input Field with combo or spinner box.
User Screens

Design Considerations for Screens

- Graph Object.
- Group Box.
- List Box.
- Radio Button.
- Tab Object.

For more information on Fields, see Chapter 11.

Determining Files

Each of the fields associated with an object on a screen must come from a file that is part of the project. Files can be identified with fields in one of the following ways:

- By expanding the file from the list box and selecting the field. When selected, the filename and fieldname are combined with a dash between the names. For example, the field ACCT in the CUSTOMER file is displayed as CUSTOMER-ACCT.

- By specifying a file with the field. To specify the file, precede the field name with the filename or file abbreviation followed by a dash. For example, if the abbreviation for the HISTORY file is H, you can specify the RESULT field from the HISTORY file as either HISTORY-RESULT or H-RESULT. (For more information on file abbreviations, see the section Guide File Access.)
• By specifying a screen default file. When you use a field from the default file, you do not have to specify the filename. If a field is not preceded by a filename or file abbreviation, EDGE first looks for it in the default file specified for the screen. For example, if you specify CUSTOMERS as your default file, and use a field called RESULT, EDGE first looks for the field in the CUSTOMERS file.

• By specifying fields from the guide assembly default file. If a field is not preceded by a filename or file abbreviation and is not in the screen default file (or there is no screen default file), EDGE looks for it in the guide assembly default file.

Objects can be filled with data from the associated field and file in one of the following ways:

• Reading Files Automatically.

• Reading Files Explicitly.

**Reading Files Automatically**

If an input field is the record ID for a file, when the agent enters data into that field, the corresponding record, if any, is automatically read from the associated file. All fields on the screen from that file are automatically filled in when the record is read. If the agent moves to
another screen that has fields from that file, those fields are also filled in from the last record read.

Note: Using logic to copy data to the field or setting a default value for the field does not cause an automatic read.

**Reading Files Explicitly**

You can explicitly read records from a file by attaching a logic flow to an object. For example, if a customer record is read and it contains a postal code, you can attach a logic flow to an object that is associated with the postal code field. The logic flow can contain statements that read the record in the postal code file for that code. In addition, if there are objects on the screen associated with the city and state fields in the postal code file, EDGE automatically fills them in.

For more information on setting up and using files in a guide, see Chapter 11.

**Determining Screen Size and Metric**

Screen size and metric determine the size of the screen and the unit of measurement to be used. Screen size and metric can be specified from the following areas:

- EDGE level, which sets the default for the entire system. For more information, see the section Set Up Screen Defaults.
In individual screens, which only applies for that screen. For more information, see the section Screen Properties.

**Determining Styles**

Styles determine the way objects are displayed, such as type and size of font. Styles are specified as part of a scheme and a style named *list box*, for example, can have a different look in different schemes.

The result of selecting a style depends on whether the default scheme is specified at the EDGE, project, or operator access levels. For more information, see the section EDGE Client Scheme.

**Determining Results**

Results are used to indicate the outcome of calls. When you design a guide, you must specify a result for every call before the call can be closed. The result, for example, might be the sale was closed, the customer problem was resolved, an incoming call was a wrong number, or there was no answer on an outgoing call.

You do not have to specify a result on every screen, but you must specify a result before the call can be closed.

You can specify results for the following:

- Function Keys.
- Hot Keys.
Determining Logic Flows for Screen Events

Logic flows can be performed based on screen events, such as before a screen is displayed or when a function key is pressed. Logic flows can be specified to performed at the following events:

- During screen display.
- Entry to screen.
- Exit from screen.
- Using function keys.
- Activating screen objects.
- Screen validation.

For information on specifying logic flows for entry, during, exit, or validation of the screen, see the section Logic Properties. For information on specifying logic flows for function keys, see the section Function Keys. For information on specifying logic flows for objects, see the section Logic.
The screens you build must provide the agents with ways to move from one screen to the next. Some screens may also need to provide options for the agents, depending on customer response. For example, if the agent reaches a customer who wants to place an order, the agent needs to be able to get to the order screen. In another situation, if the agent reaches a customer who cannot talk right now, the agent needs to be able to schedule a callback for some later time.

To provide ways to move from one screen to another and to provide options for the agent, you can use the following features:

- Function Keys.
- Hot Keys.
- Logic Flows that are attached to the screen, or to objects on the screen.
- On Selection Screens, as part of the cell selection.

Window Helper

EDGE provides a window helper screen to assist with the syntax when specifying route to screen information. This screen is displayed by pressing the three-dot button on the Route to screen property.

The Route to screen property is used when defining the following:

- Function Keys.
- Hot Keys.
- Logic Flows.
- Selection Screens cells.

The window helper screen is similar to the following:
Properties:

Command  The command to perform; can be a literal or a field or variable containing the command. EDGE will automatically enter data for this property as you complete the window helper properties. The command must be in the following format:
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>{windowPart:}{[screenName]:{objectName}}</td>
<td>The name of the window followed by a list of window properties separated by commas; can be blank, a literal, or a field or variable containing the window name.</td>
</tr>
<tr>
<td>windowPart</td>
<td>The name of the screen to place in the window; can be blank, a literal, or a field or variable containing the screen name.</td>
</tr>
<tr>
<td>screenName</td>
<td>The name of the object to give input focus; can be blank, a literal, or a field or variable containing the object name.</td>
</tr>
</tbody>
</table>

At least one component of the window command must be specified. If colons are not present in the command, then only the screenName is being specified.

**Window name**

The name of the window; can be blank, a literal, or a field or variable containing the name of the window.

Each window must have a unique user defined name.

This property is optional.

**Screen name**

The name of the screen to place in the window; can be blank, a literal, or a field or variable containing the name of the screen.

This property is optional.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object name</strong></td>
<td>The name of the object to give focus; can be blank, a literal, or a field or variable containing the name of the object.</td>
</tr>
<tr>
<td></td>
<td>This property is optional.</td>
</tr>
<tr>
<td><strong>Window options</strong></td>
<td>The properties and actions available when using the <strong>WINDOW</strong> verb; can be one of the following:</td>
</tr>
<tr>
<td>Normal window</td>
<td>Activates a window or creates a secondary window. The following can be specified:</td>
</tr>
<tr>
<td></td>
<td>• Activate.</td>
</tr>
<tr>
<td></td>
<td>• Do not create.</td>
</tr>
<tr>
<td></td>
<td>• Always on top. Forces the desktop to display an EDGE client window as the topmost window even when the client window does not have focus.</td>
</tr>
<tr>
<td></td>
<td>The default is Activate.</td>
</tr>
<tr>
<td>Close window</td>
<td>Closes the existing window.</td>
</tr>
<tr>
<td></td>
<td>This property is not typically used when routing to a screen. Selecting this property disables the other window options in this dialog.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Create dialog box</td>
<td>Allows you to create a secondary window as a dialog box. Selecting this property disables the other window options in this dialog—except window position. A dialog box does not contain maximize, minimize, or close buttons. A dialog box requires you to perform an action and prevents any interaction with menus, toolbars, or other windows until the appropriate action is taken. A screen name must be specified. In addition, the Activate window property should be disabled.</td>
</tr>
<tr>
<td>Color</td>
<td>Sets the window background color; can be blank, a comma-delimited string, or a field or variable.</td>
</tr>
<tr>
<td></td>
<td>If you leave this field blank, the background color defaults to the background screen color for the current style. For a description of the color selection dialog, refer to the WINDOW verb, Color property.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

**Frame Style**
Defines the window style for a main window or child window according to the Microsoft Document Interface (MDI) standard; can be blank, a comma-delimited string, or a field or variable.

If you leave this field blank, the background color defaults to the background screen color for the current style. For a description of the window frame style selection dialog, refer to the WINDOW verb, Frame style property.

**Size window**
The size of the window can be changed; can be one of the following:

- Maximize.
- Minimize.
- Restore window size.
- Do not change size.

The Size window properties are only available when Normal window or Create dialog box is selected. If specified, you cannot specify the Position window properties.

The default is Do not change size.
If you manually enter the screen name to route to rather than use the Window helper, the window name, screen name, and object name must be enclosed in quotation marks and separated by colons as follows:

"windowPart:screenName:objectName"

Properties: (continued)

Position window

The window can be positioned in pixels relative to the client portion of EDGE Client. The following can be specified:

- Height.
- Left.
- Top.
- Width.

The default is to cascade the window.
**Creating and Modifying Screens**

Screens are created and modified from the Screens tab page in the Guide area. To display the Screens tab page, select the following:

Project / Guide Area / Screens

A page similar to the following is displayed:
The following command is available from the Screens tab page:

Verify

The Verify command checks the selected screen for conditions that would generate an assembly error.

For more information, see the section Verify Screen.

**Schemes for Screens**

You *must* select a scheme or scheme named DEFAULT at the EDGE level, otherwise, you will encounter assembly errors. For information about selecting schemes, see the section EDGE Client Scheme.

**Version Control for Screens**

Version Control is used to manage and record changes that are made. For more information, see the section Version Control.

**Screens Menu in Guide Area**

The Screens menu is added to the EDGE Developer menu bar when the Screens tab is selected.

The Screens menu includes the following commands:

- List Screens.
- Print Screens.
- Verify Screen.
**List Screens**

The List Screens command allows you to print information about the screens for the project. You can also print the listing. The drop down box to the right of the Close command button allows you to change the size of the display. The larger the number in this box, the larger the font on the display. You can also print this listing to a client printer.
**Print Screens**

The Print Screens command allows you to print one or more screens from the project. When you select the Print Screens command, a dialog box similar to the following is displayed:

You can print using a selection list which has the names of the screens or by selecting screens from the list box of individual screens. To select nonadjacent screens, press the **CTRL** key while clicking the screen names.
You can also specify the following:

- **Print inactive screens**
  The screens that have the Inactive property set to true are printed.
- **Print object list**
  A list of the objects on the screen is printed.

**Verify Screen**

The Verify command checks the screen for conditions that would generate an assembly error. For example, it checks that all the fields on the screen actually exist in the specified files.

To verify a screen, do any of the following:

- From the Screens tab page, select a screen, then click the Verify command. (To use this command, you must first select a screen).
- From the Screens menu, select Verify. (To use this command, you must first select a screen).
- From the screen editor, click the Verify tool from the toolbar.

If the Verify command finds an error, it displays a message similar to the following:
The message box displays the total number of errors, the name of each object with an error, and the type of error. To return to the screen editor and modify the object, double-click a line with an error. The screen editor is displayed with the object selected.

The verify is based on the last saved version of the screen. If you are verifying the screen while in the screen editor, save the screen (Save Object, from the File menu) each time you make changes.
When you add or modify a screen, the screen editor is invoked and a screen similar to the following is displayed:
Screen Types

There are two types of screens:

- **Freeform.** Freeform screens allow you to use the entire screen area for objects.
- **Selection.** Selection screens use part of the screen for selection information.

To specify the type of screen, use the `Type` property.

**Freeform Screens**

Figure 19-2 is an example of a freeform screen:
Figure 19-2. Freeform Screen Example
Selection Screens

Figure 19-3 is an example of a selection screen. For information on selection screens, see the section Selection Screen Properties.
Figure 19-3. Selection Screen Example
**Function Keys**

You can define up to eight function keys on a screen. Only those you define are displayed on the screen during operations. For example, if you define function keys 1, 3, and 5, then only these three function keys are displayed; function keys 2, 4, 6, and 8 are not displayed.

Function keys are defined for each screen and their definitions apply only to that screen. If a similar operation occurs on several screens, it is recommended you define the same function key for this operation on all screens. For example, if the call can be canceled from several screens and you define function key 8 as the cancel key on one screen, you should define function key 8 as the cancel key on all other screens that provide cancellation. (For information on defining key sequences that are effective throughout the guide, see the section of Hot Keys.)

You can hide the function key area entirely by setting the Display fkeys property to Hide on the Main tab for the screen.

To define a function key for a screen, click the function key on the screen and its properties sheet similar to Figure 19-4 is displayed:
**Figure 19-4. Function Key Properties**

**Properties:**

- **Description:** The text that is displayed on the agent’s screen to describe the function key. Up to 29 characters can be displayed.

- **Exit Logic:** Specifies whether the logic flows attached to Exit from screen should be performed when this function key is pressed. For example, the Exit from screen logic is often used to validate input fields. If the function key you are defining is used to cancel the call, you can bypass the validation by selecting False.

  The default is True.

  For information on attaching logic flows to screen events, see the section Logic Properties.
### Order of Processing of Function Key Options

During operations, when a function key is pressed, the function key options are invoked in the following order:

1. The result, if specified, is set.

2. The logic flow, if specified, is executed. If the logic contains a `ROUTE` verb, that ROUTE supersedes the Route to screen, if specified, and step 3 is skipped.

3. If the Exit Logic property is specified as True, the Exit from screen logic, if any, is executed.

4. The Screen validation logic, if any, is executed.

### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic</td>
<td>The name of the logic flow to be invoked when the function key is pressed. The logic flow is invoked before the Route to Screen, if specified.</td>
</tr>
<tr>
<td>Result</td>
<td>The name of the result to be assigned when the function key is pressed.</td>
</tr>
<tr>
<td>Route to Screen</td>
<td>The name of the screen to be routed to when the function key is pressed.</td>
</tr>
</tbody>
</table>

**Note:** The logic flow attached must not have a Parameter Definition.
5. The current screen is exited.
6. EDGE routes to the specified screen.
Screen Properties

Screen properties allow you to specify properties that affect the entire screen. For example, you can specify commands to be disable when the screen is displayed.

To display the screen properties sheet, do any of the following:

- Click the Properties tool on the toolbar.
- Double-click anywhere on the screen, but not on an object or function key.
- Press F4 anywhere on the screen.
- Right-click anywhere on the screen and select Properties from the shortcut menu.

A screen similar to the following is displayed:
The following can be defined in a screen:

- **Main Properties.**
- **Commands Properties.**
- **Logic Properties.**
- **Web Properties.**
**Main Properties**

Main screen properties allow you to specify properties for the screen such as screen type, width, height, and title. To define main properties for a screen, click the Main tab on the screen properties sheet.

A properties sheet similar to the following is displayed:
Properties: Main Tab

Auto arrange  Specifies whether to automatically rearrange screen objects to fit the screen space or let objects remain in their positions; can be true or false.

The default is False.

The following events can cause the screen to be rearranged:
Properties: Main Tab (continued)

- An object’s property is changed. For example, its height or width is changed.

- The screen metric is changed.

- The function keys are changed to shown or hidden.

- The screen size is changed.

- The screen title is changed to show or hide.

- The selection area is changed to show or hide.

Default file

The name of the default file for the screen. You can change the default file at any time. If you do, check that your fields are associated with the correct file.

For more information, see the section Determining Files.

Display fkeys

The setting for the function keys; can be one of the following:

- Hide Function keys are hidden.
- Show Function keys are displayed.
- Use guide defaults The setting from the Guide Options is used.

The default is Use guide defaults.

This property affects the screen during operations as well as during development.
Hiding function keys does not affect their use in operations. For example, if F8 is the exit key and function keys are hidden, pressing F8 still causes you to exit the screen.

The setting for this property can also be changed by using the Show/Hide Function Keys tool. For example, if you press the Show/Hide Function Key tool to show the keys, this property is set to Show when you save the screen.

You can show or hide the function keys for the entire guide through Guide Options.

**Display title**

Specifies whether the title is displayed; can be one of the following:

- Hide: The title is hidden.
- Show: The title is displayed.
- Use guide default: The setting from the Guide Options is used.

The default is Use guide defaults.

This property affects the screen during operations as well as during development.
**Properties: Main Tab (continued)**

The setting for this property can also be changed with the Show/Hide Title tool. For example, if you use the Show/Hide Title tool to show the title, the Display title property is set to Show when you save the screen.

**Fkeys style**

The Style for the function key.

**Inactive**

Specifies whether the screen is inactive, which means it is not assembled as part of the guide. Can be true or false.

The default is False.

You can use this property to save a screen that you no longer use, but do not want to delete. For example, if the screen is one you built during testing and you might need it later, you can select True. Then, when you need it, change the selection to False.

This setting also allows you to remove a screen from the guide without having to do a full assembly. (If you delete a screen you no longer need, you will be forced to do a full assembly the next time you assemble the guide.)

**Metric**

The unit of measurement for objects on a screen; can be one of the following:

- **Centimeters**
  
  A centimeter is .39 of an inch.
### Properties: Main Tab (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inches</td>
<td>An inch is 1/12 of a foot or 2.54 centimeters.</td>
</tr>
<tr>
<td>Millimeters</td>
<td>A millimeter is .04 of an inch.</td>
</tr>
<tr>
<td>Pixels</td>
<td>Pixel stands for picture element. A pixel is a single dot on the screen. The actual size depends on the physical monitor.</td>
</tr>
<tr>
<td>Points</td>
<td>1/72 of an inch.</td>
</tr>
<tr>
<td>Row/Col</td>
<td>Rows and columns use the font specified in the Set Up Screen Defaults to determine height and width.</td>
</tr>
<tr>
<td>Twips</td>
<td>Twips are a physical measurement, where 20 twips equal a point, and 72 points equal an inch; that is, a twip is 1/1440 of an inch.</td>
</tr>
</tbody>
</table>

The default is Row/Col.

<table>
<thead>
<tr>
<th>Name</th>
<th>The name of the screen. The name can contain up to ten alphanumeric characters and periods, but must start with an alphanumeric character. Must be specified before the screen can be saved. After a screen is saved, the name cannot be changed.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rebound screen</td>
<td>Specifies whether the agent is to return to the screen after being routed to another screen; can be true or false:</td>
</tr>
</tbody>
</table>
Properties: Main Tab (continued)

The default is False.

A rebound screen is used, for example, when you want a common screen that agents can use for comments and that they can get to from anywhere.

If a screen is specified as a rebound screen, whenever that screen is called, the system places the ID of the calling screen in $RET. You can then use $RET in a ROUTE command to exit the rebound screen.

Figure 19-5 is an example of the use of a rebound screen. Screen B is defined as a rebound screen and it can be entered from two screens. When exit paths are defined for screen B, the routes are set to $RET. During operations, this causes the agent to return to the appropriate screen.

If you do not want the agent to be returned to the previous screen, you can attach a logic flow to Entry to screen logic for that screen. For example, if you use a common screen for errors, depending on the error, you may want the agent to continue processing or to go to a special recovery screen. Use the logic flow to determine where to route to.

Note: The EDGE screens $CALLBACK, $CALLHIST, and $ERR are designated as rebound screens.
Properties: Main Tab (continued)

Figure 19-5.  Example of a Rebound Screen

The height of the screen in units determined by the setting of the Metric property, including the area used by the function keys. Typical measurements are:

Row/Col  23 rows.
Twips    4100 twips.

Screen style
The Style to be used for the screen.

Screen width
The width of the screen in units determined by the setting of the Metric property. Typical measurements are:

Row/Col  80 columns.
Twips    8400 twips
Properties: Main Tab (continued)

Title  The title that is to be displayed on the screen title bar. Whether the title is actually displayed in operations depends on the setting of the Display title property.

Type  The type of screen; can be one of the following:

- Freeform  Allows the entire screen to be used.
- Selection  Allows part of the screen to be used as a selection area.

The default is Freeform.

Commands Properties

The commands properties for the screen specify whether the agent can use keyboard commands to control cursor movement and perform actions. Screen commands can be specified at the guide or screen level. If specified at the screen level, the setting overrides the guide setting. (Screen commands for the entire guide can be suppress though Guide Options.)

Screen commands can be specified as follows:

- Allow  The agent is allowed to use the screen command.
- Suppress  The agent is restricted from using the screen command.
- Use guide default  The setting from Guide Options is used.
Screen commands are specified in the Commands tab of the Screen properties sheet. When the Commands tab is selected, a properties sheet similar to the following is displayed:

![Properties: Commands Tab](image)

**Properties: Commands Tab**

- **.B (Backtab)** The cursor is moved to the previous input object.
- **.C (Tab)** The cursor is moved to next input object.
- **.E (Erase entry)** The entry at the current input location is erased.
Logic Properties

Logic properties allow you to specify logic flows to be run when a screen event occurs. For example, logic flows can be run when the screen is exited. Logic flows can also be run when the agent presses a function key or enters data for an object. Specifying a logic flow with an event is referred to as attaching logic flows.

Properties: Commands Tab (continued)

.G (Goto) Goes to another row or input location (The location is specified after the command.)
.J (Route to screen) Routes (jumps) directly to another screen. (The screen name is specified after the command.)
.L (Previous line) The cursor is moved to the previous line.
.N (Next line) The cursor is moved to the next line.
.P (Previous screen) The cursor is moved to the previous screen.
.R (Modify) The text at an input object is modified.
.T (Goto first input) The cursor is moved to the first input object on the screen.
.W (Undo) The change in the input object is undone.
.Z (Print) The screen is printed.
Note: Logic flows that have a Parameter Definition cannot be attached to events using logic properties.

Logic flows can be attached to the following screen events:

- Activation.
- Deactivation.
- During paint.
- Entry to screen.
- Exit from screen.
- Screen validation.

To specify logic flows for screen events, select the Logics tab on the Screen properties sheet.

For information about attaching a logic flow to function keys, see the section Function Keys. For information about attaching a logic flow to an object, see the section Logic.
## User Screens

### Screen Properties

#### Properties Sheet

<table>
<thead>
<tr>
<th>Main</th>
<th>Commands</th>
<th>Logics</th>
<th>Web</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Deactivation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During paint 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During paint 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During paint 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During paint 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>During paint 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry to screen 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry to screen 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry to screen 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry to screen 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Entry to screen 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exit screen 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exit screen 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exit screen 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exit screen 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exit screen 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screen validation 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screen validation 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screen validation 3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screen validation 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screen validation 5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Properties: Logics Tab

Activation  The logic flow to invoke when the window displaying the screen gains the focus. This occurs when an agent selects a different window or when a new window is created. Activation logic is not invoked when you route to a new screen within the same window.

An agent can switch between windows in one of the following ways:

- Click a window to activate it.

Select the window from the Window menu.

The guide developer can use activation logic to perform special processing when an agent switches between windows. For example, a guide developer can use the SET-PROPERTY verb to enable/disable tools on the toolbar that are only applicable in certain windows.

Each window can only have one activation logic.

The three-dot button invokes the logic editor.
**Properties: Logics Tab (continued)**

**Deactivation**
The logic flow to invoke when the window displaying the screen loses the focus. This occurs when an agent selects a different window or when a new window is created. Deactivation logic is not invoked when you route to a new screen within the same window.

An agent can switch between windows in one of the following ways:

- Click a window to activate it.
- Select the window from the Window menu.

The guide developer can use deactivation logic to perform special processing when an agent switches between windows. For example, a guide developer can use the SET-PROPERTY verb to enable/disable tools on the toolbar that are only applicable in certain windows.

Each window can only have one deactivation logic.

The three-dot button invokes the logic editor.
During paint: The logic flows to invoke after the screen is painted but before any keyboard entry is allowed. You can use this, for example, when you want to perform calculations that may take some time. If you perform the calculations using Entry to screen logic, the screen is not displayed until after the calculations are finished. If you use During paint, the screen is displayed before the logic flow is invoked. This allows the agent to view the screen in the meantime, although nothing can be entered on the screen until the logic flow is completed.

A logic flow executed during the paint does not affect data displayed on the screen until the screen is repainted.

Entry to screen: The logic flows to invoke just before the screen is presented to the agent. You can use this, for example, to verify the agent should be given access to the screen.

Exit from screen: The logic flows to invoke when the agent presses a function key or hot key to exit the screen or, if specified on a selection screen, the selection has a route to screen specified. The Route to screen must move to a new screen and must have the Exit logic property set to True. (For information on defining function keys, see the section Function Keys; for information on hot keys, see the section Hot Keys; for information on specifying a route on a selection screen, see the section Cell Properties.)
Properties: Logics Tab (continued)

If a screen is exited in any other way, the Exit from screen logic flow is not invoked. For example, if you route to another screen because of a logic flow that is attached to a field, Exit from screen logic is not invoked. To invoke a logic flow under all conditions when leaving a screen, attach it to Screen validation.

Screen validation

The logic flows to invoke under all conditions when the agent leaves the screen. A logic flow attached at this point is always executed when the screen is exited.

Execution Order of Logics

During operations, when a screen is entered, the logic flows are invoked in the following order:

1. Activation logic (when an agent selects a different window or when a new window is created).
2. Entry to screen logic.
3. During paint logic.
4. Tab page entry logic (if a tab object is on the screen).
5. Tab control entry logic (if a tab object is on the screen).

During operations, when a screen is exited, the logic flows are invoked in the following order:
1. Tab page exit logic (if a tab object is on the screen).
2. Tab control exit logic (if the tab object is on the screen).
3. Exit from screen logic.
4. Screen validation logic.
5. Deactivation logic.

When you switch from one tab page to another, the logic flows are invoked in the following order:

1. The tab page exit logic of the tab page being exited.
2. The tab control exit logic.
3. The tab page entry logic of the tab page being entered.
4. The tab control entry logic.

**Web Properties**

The Web tab is used to specify properties for field help and cross-reference buttons, and function key and hot key layouts.

The web tab properties sheet is similar to the following:
**Properties: Web Tab**

**Display field help btn**
- **Hide**: The ? button is not displayed.
- **Show**: The ? button is displayed.
- **Use web default**: The setting at the project level is used. This is the default.

**Display xref button**
- **Hide**: The Search button is not displayed.
- **Show**: The Search button is displayed.

A ? button is automatically added after every input field; can be one of the following:

A Search button is automatically added after every input field that invokes a cross-reference; can be one of the following:
### Properties: Web Tab (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function key layout</td>
<td>The Function key layout property specifies how the function keys are displayed.</td>
</tr>
<tr>
<td></td>
<td>This option is ignored if the Display fkeys property is set to N. Can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>- Hyper-text links: Function keys are displayed as hypertext links.</td>
</tr>
<tr>
<td></td>
<td>- Links and table: Function keys are displayed as hypertext links in a table.</td>
</tr>
<tr>
<td></td>
<td>- Table and submit buttons: Function keys are displayed in tables of submit buttons.</td>
</tr>
<tr>
<td>Hot key layout</td>
<td>The Hot key layout property specifies how the hot keys are displayed; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>- Do not display: Hot keys are not displayed.</td>
</tr>
<tr>
<td></td>
<td>- Hyper-text links: Hot keys are displayed as hypertext links.</td>
</tr>
</tbody>
</table>

The setting at the project level is used. This is the default.
Properties: Web Tab (continued)

- Links and table: Hot keys are displayed as hypertext links in a table.
- Table and submit buttons: Hot keys are displayed in tables of submit buttons.
- Use web default: The setting at the project level is used. This is the default.

Table 19-1 describes when a Search button or ? button is displayed.

**Table 19-1. Field Help and Xref Button Displays**

<table>
<thead>
<tr>
<th>Display field help button</th>
<th>Display xref button</th>
<th>Button Displayed</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>N</td>
<td>None. Cannot use cross-reference and field help.</td>
</tr>
<tr>
<td>N</td>
<td>Y or blank</td>
<td>Search button (only after cross-reference fields)</td>
</tr>
<tr>
<td>Y or blank</td>
<td>N</td>
<td>? button (only after input fields)</td>
</tr>
<tr>
<td>Y or blank</td>
<td>Y or blank</td>
<td>? button (only after input fields)</td>
</tr>
</tbody>
</table>
Selection screens are used to present a list from which the agent makes selections; for example, a list of sources where your new customer heard about your company. For an example of a selection screen, see Figure 19-3.

The selection area of selection screens have properties not on free-form screens. These properties are divided into the following tabs when the Type property is set to Selection and you click the selection area:

- Main Properties.
- Translation Table Properties.
- Cell Properties.

**Main Properties**

The Main tab includes properties that determine the display of the selection area. A properties sheet similar to the following is displayed:
Properties: Main Tab

Always show  The Number of the cells to always show on the selection screen. This means these selections are always shown, regardless of whether they were selected on a previous screen. For example, you can use this option to include a selection such as Not Interested.
Properties: Main Tab (continued)

Exclude from fields
The selections to exclude based on choices made on a previous selection screen. For example, on the first selection screen the agent could select all the magazines the customer subscribes to. On the next screen, the logic could exclude these selections. The agent could readily see the magazines that were not selected and might be sold to the customer.

You can indicate the selection cells to be excluded by entering any of the following:

- The $SELNO field.
- The fieldname specified in the Selection # field property.

Include from fields
The selections to include based on choices made on a previous selection screen.

You can indicate the selection cells to be included by entering any of the following:

- The $SELNO field.
- The fieldname specified in the Selection # field property.
## Properties: Main Tab (continued)

**Note:** If you are going to use a second screen showing selections based on the first screen, the selection choices on the second screen should be exactly the same as in the first screen. A suggested way to do this is to make a copy of the first screen. Then, in the copy, make changes to the text in the freeform section to reflect the new question.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum # selections</td>
<td>The maximum number of selections an agent can make on the selection screen. The limit is 50. For example, you may show ten choices but allow the agent to choose only four.</td>
</tr>
</tbody>
</table>

**Note:** You can use the selection screen as a menu by setting the maximum number of selections to one.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Only rotate first</td>
<td>The number of selections to be rotated. The limit is 50. For example, you may have ten selections on a screen where the tenth selection is Refused. If you specify 9 as the number to rotate, the first nine selections are rotated; the tenth selection is always displayed in the tenth position. If no number is specified, none of the selections are rotated. This property is only available when the Rotation property is set to random or sequential.</td>
</tr>
</tbody>
</table>
### Properties: Main Tab (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Rotation</strong></td>
<td>The order of the choices are displayed in the order you entered them.</td>
</tr>
<tr>
<td>No rotation</td>
<td>The choices are displayed in the order you entered them.</td>
</tr>
<tr>
<td>Random</td>
<td>The choices are displayed in random order.</td>
</tr>
<tr>
<td>Sequential</td>
<td>The first choice is selected randomly, then the rest of the choices follow sequentially.</td>
</tr>
</tbody>
</table>

This option is intended to reduce bias based on the order in which the selections are shown that customers generally remember the first and last choice from a list of choices. Additionally, the agent may emphasize the same choices when given choices in the same order.

You can restrict the rotation of choices to a specific number. For example, you may have ten choices on a screen where the tenth choice is Refused. If you specify the first nine choices only, the tenth choice would always be displayed in the tenth position.

<table>
<thead>
<tr>
<th>Selection # field</th>
<th>The name of a field in which to store the selection numbers chosen by the agent. This field is used in addition to $SELNO.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selection text field</td>
<td>The name of a field in which to store the text of the selection chosen by the agent. This field is used in addition to $SELTXT.</td>
</tr>
</tbody>
</table>
The Translation table tab contains properties for specifying tables and fields to be associated with selection choices. Each entry in the table corresponds to a cell from the selection screen and can be used to translate the choices to a set of standard codes. For example, you might want the agent to select an industry type which you can translate and store as a SIC code. Or you might use a table for product orders where the agent needs a description and the shipping clerk needs a product number.

You can specify up to three tables for a selection screen. For each table, you can specify a field in one of your files in which to store the translated information. To specify the translation table, set the Type property to Selection, click the selection area in the screen editor, and select the Translation Table tab. A properties sheet similar to the following is displayed:
Creating Tables

Tables allow you to correlate data stored in a table with cells on a selection screen. Tables can be used for many different purposes. For example, if you are selling merchandise from a catalog, your agents may need to identify the product by description, product number, color, and price. The customer may not know the product number,
but the order department will need this information. A selection screen showing the product descriptions could include the tables for product number, color, and price.

The position of a table entry must correspond exactly to the selection Number of its related choice on the selection screen. For example, the first entry in the table must correspond to the choice with selection number 1.

To indicate a table is to be used on a selection screen, use the Translation Table Properties properties. The Translation table properties allow you to specify a field into which the translated information from the table can be saved. You can then display the field on the screen, if desired.

To display the Tables page in the Guide area, select the following:

Project / Guide / Tables

The Tables tab page displayed is similar to the following:
When you create or modify a table, a dialog box similar to the following is displayed:
Cell Properties

The Cell tab specifies the selections from which an agent can choose. The selections are available only with a selection screen.

To specify properties for a cell, click the cell in the selection area to display the properties sheet, then select the Cell tab.

A properties sheet similar to the following is displayed:
**Properties: Cell Tab**

- **Column**: The column number in the selection area. You can have up to five columns.
- **Number**: A selection number for the cell. During operations, the agent chooses a selection by specifying the number for the cell. You must specify a number for each cell. The cell numbers do not have to be in sequence, but must be unique and in the range 0–99.
When the agent specifies a selection, the number is stored in a multivalued field in the SYS file called $SELNO. $SELNO is cleared at the beginning of a call, or whenever a selection screen that is different from the previous selection screen is entered, so if you need to save the information for future use, you should do so before the screen is exited.

If you plan to return to the selection screen before the call is completed, you can use a logic flow to copy an empty value into $SELNO and attach it at Entry to Screen. If you set $SELNO to an empty value, EDGE automatically clears $SELTXT. (For information on attaching logic to screens, see the section Logic Properties.)

A hierarchy for the choices and should be used when you set up different routes for different choices. The hierarchy determines the routing when more than one choice with a route is selected. For example, if choice 6 sets a route to the order screen and choice 7 sets a route to the end screen, you can give choice 6 a higher rank than choice 7. Then, if the agent selects both 6 and 7, the routing goes to the order screen (the higher ranking).

The highest rank is 1; the lowest rank is 99. If empty, the rank is by default 99.

This Rank property is used only when the Route to screen property is specified.
**Properties: Cell Tab (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result</td>
<td>The name of the result to be assigned if the choice is selected during operations. The result is assigned only if the <code>Route to screen</code> property is used to exit the selection screen.</td>
</tr>
<tr>
<td>Route to screen</td>
<td>The name of the screen to be routed to when the cell is selected. Clicking the three-dot button brings up the <code>Window Helper</code>.</td>
</tr>
<tr>
<td>Row</td>
<td>The row number in the selection area. You can have up to ten rows.</td>
</tr>
<tr>
<td>Text</td>
<td>The text that describes the selection and is usually what the agent uses to solicit information from the customer. When the agent enters the selection number, the associated text is stored in the multivalued system field in the SYS file called <code>$SELTXT</code>. The cells are numbered from Cell1 through Cell50. After you specify a description, the description is added to the cell name. For example, if the description is Newspapers, the cell would be identified as Cell1-Newspapers.</td>
</tr>
</tbody>
</table>
**Screen Menu**

When you invoke the screen editor, a menu called Screen is added to the menu bar. The Screen menu includes commands to display the toolbox or properties sheet as well as commands that correspond to the screen tools that are in the toolbox.

The Screen menu has the following commands:

### Screen Menu Commands:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveX toolbox</td>
<td>The ActiveX toolbox displays the icons of the ActiveX controls that have been configured. For more information, see Chapter 21.</td>
</tr>
<tr>
<td>Align</td>
<td>The selected objects are aligned; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Bottom</strong></td>
</tr>
<tr>
<td></td>
<td>The selected objects are aligned by the bottom of the selected objects, with the lowest positioned object as the basis.</td>
</tr>
<tr>
<td></td>
<td>You can also use the Align to Bottom tool.</td>
</tr>
<tr>
<td></td>
<td><strong>Left</strong></td>
</tr>
<tr>
<td></td>
<td>The selected objects are aligned to the left side of the selected objects, with the most left-positioned object as the basis.</td>
</tr>
<tr>
<td></td>
<td>You can also use the Align to Left tool.</td>
</tr>
</tbody>
</table>
### Screen Menu Commands: (continued)

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Right</strong></td>
<td>The selected objects are aligned to the right side of the selected objects, with the most right-positioned object as the basis. You can also use the <strong>Align to Right</strong> tool.</td>
</tr>
<tr>
<td><strong>Top</strong></td>
<td>The selected objects are aligned to the top of the selected objects, with the highest positioned object as the basis. You can also use the <strong>Align to Top</strong> tool. The alignment types are enabled when more than one object is selected. The objects are aligned to the first object selected in a multiple selection.</td>
</tr>
<tr>
<td><strong>Bring to front</strong></td>
<td>The selected object is brought to the front of all other objects, which means the object can cover other objects in the area. The object is sent to the end of the <strong>Tabbing order</strong>. This property is enabled only when a screen object is selected.</td>
</tr>
<tr>
<td><strong>Function keys</strong></td>
<td>The function keys are shown or hidden. You can also use the <strong>Show/Hide Function Keys</strong> tool.</td>
</tr>
<tr>
<td><strong>Grid</strong></td>
<td>The grid is shown or hidden for the current screen.</td>
</tr>
</tbody>
</table>
Screen Menu Commands: (continued)

You can also use the Show/Hide Grid tool.

The dimensions for the grid are based on the Metric property.

Group

The selected objects are combined in a group. This allows you to move the objects as a unit.

You can also use the Group Objects tool.

Grouped objects can be moved or deleted as a single unit. If you select one object in a group, the entire group is selected.

This property is enabled when there are objects selected.

The property page displays the common properties of the objects in the group.

Group outlines

The outlines for groups are shown or hidden.

When group outlines are shown, a box is displayed around the objects in the group.

You can also use the Show/Hide Group Outlines tool.

Object tools

The objects from the toolbox; can be one of the following:
Screen Menu Commands: (continued)

Audio Object.
Check Box.
Command Button.
Display Field.
Graph Object.
GroupBox.
Hyperlink Object.
Image Object.
Input Field.
Label.
List Box.
RadioButton.
Selection (the cursor becomes an arrow to use as a pointer).
Shape.
Tab Object.
Table Object.
Video Object.

Options
A dialog box is displayed to specify the screen settings for all screens. The Options dialog box is similar to the following:
Screen Menu Commands: (continued)

A lasso is used to select objects on your screen. To use, hold your left mouse button down and drag your cursor to select the object; enclose the entire object within the lasso.

If not checked, the objects touched by the lasso become part of the selection.

The grid settings that apply to all the screens. The grid defaults that can be set are as follows:

- The objects on all the screens in the project are aligned to the grid.
- The horizontal grid line increments. This value is based on the Metric property.
Screen Menu Commands: (continued)

- **Vertical**: The vertical grid line increments. This value is based on the Metric property.

- **Properties**: The properties sheet is displayed. For more information, see Screen Properties.

- **Rulers**: A horizontal and vertical ruler is displayed. To reposition the rulers, drag them up and down or from left to right on the screen.

  - You can also use the Show/Hide Rulers tool.

  - The ruler marks are based on the Metric property.

- **Same height**: The selected objects are made the same height.

  - The objects are sized to the first object selected in a multiple selection.

  - If the objects contain the Use Caption Width, Use Field Width, or Use Text Width properties, set these properties to false.

  - You can also use the Same Height tool.

- **Same width**: The selected objects are made the same width.

  - The objects are sized to the first object selected in a multiple selection.

  - If the objects contain the Use Caption Width, Use Field Width, or Use Text Width properties, set these properties to false.
<table>
<thead>
<tr>
<th>Screen Menu Commands: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>You can also use the Same Width tool.</td>
</tr>
<tr>
<td><strong>Send to back</strong> The selected object is moved further back from the agent. The object can therefore be hidden by other objects. This object is moved to the top of the Tabbing order.</td>
</tr>
<tr>
<td>This property is only enabled when an object is selected.</td>
</tr>
<tr>
<td><strong>Snap to grid</strong> The objects in the current screen are aligned to the grid. If enabled, objects are aligned to the nearest grid location when they are created or moved. When objects are created with the Snap to grid enabled, the height and width of the object are in multiples of the grid line increment.</td>
</tr>
<tr>
<td><strong>Tabbing order</strong> The order in which objects are tabbed to when ENTER or TAB is pressed during operations. This property does not change the placement of objects on the screen, only the sequence in which objects are tabbed to.</td>
</tr>
<tr>
<td>A list of all the objects on the screen are displayed. However, to be in the tabbing order, the object must meet the following criteria:</td>
</tr>
<tr>
<td>• It must be visible and enabled.</td>
</tr>
<tr>
<td>• It must be able to get the focus. For information on focus, see the section Object Focus.</td>
</tr>
</tbody>
</table>
Screen Menu Commands: (continued)

- It must have a tab set. (If the object is visible, enabled, and can get the focus, it by default has a tab stop set. The tab stop can be set or cleared using the SET-PROPERTY verb. Setting or clearing the tab stop does not change the object’s tabbing order. For example, if the object has a tabbing order of 2, clearing the tab stop causes the field to be skipped when the agent tabs. If tabbing is reset, the object still has a tabbing order of 2.)

A dialog box similar to the following is displayed:
Screen Menu Commands: (continued)

The default tabbing order is the order in which the objects are defined. To set the tabbing order to the order of the objects on the screen, press the Reset Order button.

To change the order of any object, drag its name before the name of the object which should follow it in the tabbing order.
The Reset button will automatically reorder the objects according to a Left to Right, Top to Bottom algorithm. The object which is topmost and leftmost will be first in the tabbing order.

The exceptions to this algorithm are the Tab Object and the Group Box object. For both of these objects, when the algorithm encounters them, it then sorts all of their children (subobjects which are fully enclosed by the object) first and then continues with the following objects. The children objects will thus be displayed immediately after their parent in the tabbing order. This is the only case where the group box is considered a container; otherwise it is treated just like any other object.

<table>
<thead>
<tr>
<th>Title</th>
<th>The name of the screen. The name is located at the top of the screen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toolbox</td>
<td>The Screen Editor Toolbox similar to Figure 20-1 is displayed.</td>
</tr>
<tr>
<td>Ungroup</td>
<td>The objects in a group are separated, which allows the objects to move independently on the screen. Ungrouping objects does not affect the functionality of the objects.</td>
</tr>
</tbody>
</table>
Screen Tools

Screen tools allow you to do a variety of actions including aligning objects, changing the height and width of objects, and showing and hiding parts of the screen.

To help align objects on the screen, you can display a grid and have objects snapped to the grid. Grid measurements are specified in the Horizontal and Vertical properties.

Screen tools are available from the screen toolbox and Screen menu.

The following are the screen tools:

- Align to Bottom.
- Align to Left.
- Align to Right.
- Align to Top.
- Group Objects.
- Same Height.
- Same Width.
- Show/Hide Function Keys.
- Show/Hide Grid.
• Show/Hide Group Outlines.
• Show/Hide Rulers.

The screen toolbox is shown in Figure 20-1.

**Align to Bottom**

The Align to Bottom tool, 📣, is used to align the bottom of all selected objects, with the lowest positioned object as the basis.

**Align to Left**

The Align to Left tool, 📝, is used to align the left side of all selected objects, with the most left-positioned object as the basis.

**Align to Right**

The Align to Right tool, 📝, is used to align the right side of all selected objects, with the most right-positioned object as the basis.

**Align to Top**

The Align to Top tool, 📝, is used to align the top of all selected objects, with the highest positioned object as the basis.

**Group Objects**

The Group Objects tool, 🕰️, is used to combine selected objects in a group. This allows you to move the objects as a unit. When groups are shown, lines that connect all the objects in the group can be displayed by selecting the Show/Hide Groups tool.

Grouped objects can be moved or deleted as a single object. If you select one object in a group, the entire group is selected. When
objects are grouped, the object properties for the individual objects cannot be changed.

**Same Height**

The Same Height tool, is used to make selected objects the same height, based on the object with the most height.

**Same Width**

The Same Width tool, is used to make selected objects the same width, based on the object with the greatest width.

**Show/Hide Function Keys**

The Show/Hide Function Keys tool, is used to show or hide function keys. The tool is available only when the Display fkeys property is set to Show or Hide.

The tool can be in the following three states:

- Depressed and not gray indicates the function keys are shown.
- Not depressed and not gray indicates the function keys are hidden.
- Not depressed and gray indicates to use guide default.

The current setting of the tool becomes the setting for the Display function keys property. For example, if you press the Show/Hide Function Key tool to Show the keys and save the screen, the Display function keys property is set to Show.
**Show/Hide Grid**

The Show/Hide Grid tool, ![grid icon], is used to show or hide a grid. The dimensions for the grid are based on the settings in the **Horizontal** and **Vertical** properties.

*Figure 19-6* is an example of a screen with the grid and rulers displayed.

**Show/Hide Group Outlines**

The Show/Hide Group Outlines tool, ![group outlines icon], is used to display or hide a border which indicates that objects are grouped. For example, the following display three check boxes that are grouped.

![Check boxes grouped]

**Show Group Outlines**

**Hide Group Outlines**

**Show/Hide Rulers**

The Show/Hide Rulers tool, ![ruler icon], is used to show or hide a ruler. The ruler marks are based on the grid measurements set in the **Horizontal** and **Vertical** properties.

*Figure 19-6* is an example of a screen with the grid and rulers displayed.
Show/Hide Title

The Show/Hide Title tool, , is used to show or hide the title of the screen. The tool is available only when the Display title property for the screen is set to Show or Hide.

The tool can be in the following three states:

- Depressed and not gray indicates the function keys are shown.

Figure 19-6. Example of Grid and Rulers
Not depressed and not gray indicates the function keys are hidden.

Not depressed and gray indicates to use guide default.

**Snap to Grid**

The Snap to Grid tool, is used to align objects to the grid. If selected, objects are aligned to the nearest grid location when they are created or moved. When objects are created with the Snap to Grid tool selected, the height and length of the object is in multiples of the grid increment. The grid increments are modified through the Options command in the Screen menu.

**Ungroup Objects**

The Ungroup Objects tool, is used to separate the objects in the group, which allows the objects to move independently on the screen. Ungrouping objects does not affect the functionality of the objects.
Screen objects are things that can be placed on user screens and hence can be seen. For example, a Display Field object specifies an area of the screen where data can be displayed. A screen can contain up to 400 objects. However, for performance reasons, we recommend that you do not have screens with more than 200 objects.

To select a screen object, click the tool for the object on the toolbox. The toolbox is shown in Figure 20-1.

Note: The toolbox can be dragged to other parts of the screen as well as moved to the top, bottom, or sides of the screen.

After a tool is selected, you can place the object on the screen by dragging the mouse across the area for the object. After you release the mouse, the object is selected and the properties can be specified.
You can move an object by dragging it anywhere on the screen. You can also use the **Left**, **Top**, **Width**, and **Height** properties to change the size and location of objects.

The properties depend on the type of object. When appropriate, list boxes allow you to select files, fields, and logic flow names for properties. For example, if the property requires a fieldname, you can move the cursor to the property, then click the three-dot button, select the file and expand it, and select the field. For more information on properties, see the section **Object Properties**.

---

*Note:* Positioning objects beyond the bounds of the screen may cause undesirable or unpredictable behavior in the guide.
Screen Objects Overview

Figure 20-1. Screen Editor Toolbox

Select
Display Field
List Box
Radio Button
OLE Object
Graph Object
Image Object
Shape
Video Object
Group Objects
Align to Left
Align to Top
Same Height
Show/Hide Function Keys
Show/Hide Group Outlines
Show/Hide Grid
Label
Input Field
Command Button
Check Box
Group Box
Tab Object
Table Object
Audio Object
Hyperlink Object
Ungroup Objects
Align to Right
Align to Bottom
Same Width
Show/Hide Title
Snap to Grid
Show/Hide Rulers
Creating an Object

Objects are created by placing them on the screen. This must be done before the properties can be specified.

You can place objects on the screen by doing the following:

- Click the tool that represents the object from the toolbox. The cursor changes to a cross-bar. Then click once in the screen editor area. The object is created.

- Place the object on the screen by clicking and dragging the cursor from top to bottom or bottom to top of the screen editor area. When the mouse is released, the object is created.

Note: To keep the tool selected and be able to continue placing objects of that type, press the CTRL key while clicking the tool.

After the object is created, it is displayed with a default name representing the type of object. For example, the first display field object is named DisplayField1.

You can copy an object in one of the following ways:

- Select the object and select Copy from the Edit menu. Then select Paste from the Edit menu. A copy of the object is placed on the screen.
Creating an Object

- Press Ctrl and drag the object to the left, right, down, or up. A copy of the object is placed on the screen.

Handles around an object indicate the object is selected. You can change the size of the object by dragging the handles.

You can move the selected object in one of the following ways:

- Click in the center of the object and drag it to another area of the screen.

- Ctrl+↑, Ctrl+↓, Ctrl+←, Ctrl+→ will nudge up, down, left or right the selection by one pixel.

Object Focus

During runtime, mouse or keyboard activity can be directed to only one object at a time. That object is said be the focus object or to have the focus. An object can have the focus only if it can accept input, and it is visible and enabled. You can left-click on an object or tab to it to give it the focus.

Note: Display field and label objects cannot accept input and therefore can never have the focus.

Graph objects cannot have the focus.
Screen Objects

Description of Objects

EDGE contains the following screen objects:

- Audio Object.
- Check Box.
- Command Button.
- Display Field.
- Graph Object.
- Group Box.
- Hyperlink Object.
- Image Object.
- Input Field.
- Label.
- List Box.
- Radio Button.
- Shape.
- Tab Object.
- Table Object.
- Video Object.

Properties, which are used to define the object, are described in the section Object Properties.

ActiveX controls can also be placed on screens as objects. For more information on ActiveX controls, see Chapter 21. For more information on creating screens, see Chapter 19.
Audio Object

The Audio object allows an audio clip to be played. To hear an audio clip, a sound card and output device must be installed in the PC.

The following audio files are supported:

- MPEG-1 (.mpg, .mpeg, .mpv, .mp2, .mpa).
- Wave (.wav).
- AU (.au, .snd).
- AIFF (.aif, .aiff)
Properties:

Auto Start
Properties: (continued)

Background Sound
Continuous Play
Enabled
Field Name
Height
Left
Name
Path
Play/Stop Controls
Playback Track Bar
Repeat Count
Timer Information
Top
Type
Visible
Volume
Width
Check Box

A check box object is used to indicate a true or false condition.

Properties:

Caption
**Properties: (continued)**

- Caption Type
- Enabled
- Height (%)
- Left (%)
- Logic
- Name
- Style
- Top (%)
- Type
- Use Caption Width
- Value Field
- Visible
- Width (%)

**Notes**

The condition is returned in the **Value Field** property; can be one of the following:

- **True**  
  Check box is checked.

- **False**  
  Check box is cleared.
The exact value returned for the true or false condition of the button can be specified as a guide option. The default is 1 for true and 0 for false. (For more information, see the section Guide Options.)
Command Button

A command button object triggers an action or command when it is clicked. The action that is triggered depends on your logic.

Accelerator keys can also be defined on command buttons. They are shortcut keys that can be used in EDGE Client by pressing ALT+key. key is an underlined letter on the command button. For more information, see the section Accelerator Keys.

The command button object’s properties sheet has the following tabs:
- Main: Contains general properties for the command button.
- Icon: Contains properties for an optional icon you can have on a command button.
Properties: Main Tab

Caption
Caption Type
**Properties: Main Tab (continued)**

- Enabled
- Height
- Left
- Logic
- Name
- Style
- Text Left (%)
- Text Top (%)
- Toggle Hilite
- Top
- Type
- Use Caption Width
- Value Field
- Visible
- Width
**Properties: Icon Tab**

- Height (%)
- Image Location
- Index
- Left (%)
- Local Image
- Server Image
- Top (%)
- Width (%)
- Type

![Properties Sheet](image)
Properties: Icon Tab (continued)

Width (%)

Notes

The condition of the button is returned in the Value Field property and can be one of the following:

True  Command button is highlighted.
False  Command button is not highlighted.

The exact value returned for the true or false condition of the button can be specified as a guide option. The default is 1 for true and 0 for false. (For more information, see the section Guide Options.)

The Toggle Hilite property determines how the condition is changed:

- Only while the mouse button is held down.
- Toggled each time it is clicked.

The command button object allows you to display a graphic, such as an icon, on the button. The following is an example of a command button with an icon:
The icon is placed on the command button using the top left corner of the face of the button as the origin (0,0). This means the icon image is always drawn relative to the face of the button and not the button rectangle, which includes a three-dimensional edge. If the button style includes color or the scheme is transparent, the background color for the button will bleed through the icon.

**Accelerator Keys**

Accelerator keys are shortcut keys on command buttons that can be used in EDGE Client by pressing Alt+key where key is an underlined alphanumeric character or punctuation character that does not require the Shift key to access.

The accelerator key is valid only for the currently active window.

To add an accelerator key, in the Caption property, prefix a & (ampersand) to the character you want to be the shortcut key.

For example, the Caption property can contain the following:

```
&Help
```

The command button object in the screen editor displays the following:

```
Help
```
To invoke the logic attached to the command button, the agent can click the command button or press ALT+h. If ALT+h is used as a shortcut by another command button on the same screen, the logic is not invoked. Instead, the object focus is changed based on the tabbing order of the screen. To invoke the logic when using multiple accelerator keys with the same letter, the agent must press ENTER when the applicable command button has the focus.

If ALT+h can also be used to access a menu or a menu command, the accelerator key for the command button takes precedence. For example, if you defined H as the accelerator key for a command button, but you want to access the Help menu on the toolbar, pressing ALT+h invokes the logic attached to the command button. The menu item that shares the same shortcut key must be selected with the mouse.

If you have multiple &’s in the Caption property, the last occurrence of the & is used. For example, the valid accelerator key for &H&elp is e.

Note: To include a & (ampersand) in the Caption, but not have it be an accelerator key, prefix the & with another &. For example, to display One & Two, type One && Two.
Screen Objects Display Field

Display Field

A display field object presents data from a file in the location you specify. Data in display fields cannot be updated by the agent.

Properties:
- Bullet Type
- Display Style
- Embedded Field Style
- Enabled
Properties: (continued)

Field Name
Height
Interpret Expression
Justification
Left
Multiline
Name
Numbering
Scroll Bar
Start Number
Style
Top
Type
Use Field Width
Visible
Width
Word Wrap
Notes

If you specify a highlight or blinking display style, the exact characteristics depend on whether the agent is using EDGE Client or ASCII.
**Graph Object**

A graph object can be used to display information in graph format. The following is an example of a clustered bar graph:

![Clustered Bar Graph Example](image)

The following types of graphs can be displayed:

- 3D-surface.
- Absolute area.
- Clustered bar.
- Line.
- Pie.
- Scatter.
- Simple bar.
- S-percent area.
- S-percent bar.
- Stacked area.
- Stacked bar.
- Z-clustered bar.

The graph object contains the following tabs:

- **Main**: Contains properties for the graph object.
- **Data**: Contains properties for the data of the graph.
**Properties: Main Tab**

- Bottom Title
- Dimensions
- Graph Type
- Height
Properties: Main Tab (continued)

- Left
- Left Title
- Name
- Right Title
- Style
- Top
- Top Title
- Type
- Visible
- Width
Properties: Data Tab

- Amplitude
- Color
- Fill
- Legend
- Line Style
- Point Symbol
Properties: Data Tab (continued)

- X Distance
- X Label
- Y Label
- Z Distance
- Z Label
Group Box

A group box object can be used to visually relate screen objects by enclosing them in an outline. You can also define a title for the group box.

![Properties Sheet](image)

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Groupbox1</td>
<td></td>
</tr>
<tr>
<td>Main</td>
<td></td>
</tr>
<tr>
<td>Border thickness</td>
<td>1</td>
</tr>
<tr>
<td>Border type</td>
<td>Engraved</td>
</tr>
<tr>
<td>Enabled</td>
<td>True</td>
</tr>
<tr>
<td>Height</td>
<td>3.56</td>
</tr>
<tr>
<td>Left</td>
<td>29.43</td>
</tr>
<tr>
<td>Logic</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Groupbox1</td>
</tr>
<tr>
<td>Style</td>
<td>GroupBox</td>
</tr>
<tr>
<td>Title</td>
<td></td>
</tr>
<tr>
<td>Title alignment</td>
<td>Left</td>
</tr>
<tr>
<td>Top</td>
<td>14.50</td>
</tr>
<tr>
<td>Type</td>
<td>Groupbox</td>
</tr>
<tr>
<td>Visible</td>
<td>True</td>
</tr>
<tr>
<td>Width</td>
<td>31.29</td>
</tr>
</tbody>
</table>
Notes

Logic can be attached to the group box and it will be executed whenever any object inside the group box is modified, after any logic attached to that object is executed. An object is considered to be
inside a group box if the center of the object is inside the rectangle of the group box.

The following is an example of border types:

- **Engraved border**
- **Inset border**

Objects are pasted on to a group box using the Paste Into command from the Edit menu.
Hyperlink Object

The hyperlink object is used to attach links to an Image Object or Label.

![Properties Sheet](image)
A hyperlink is a specific location on the image or label object that executes an action when it is clicked. For example, when you click a
hyperlink, a screen or URL is routed to, a result can be set, or a logic is executed.

URL stands for Universal Resource Locator. It is a universal syntax that allows you to specify a protocol and an address. Some protocols are http, mailto, and ftp. The URL for a hyperlink object can be specified using the properties Target URL or Target Field URL. The following is an example of a URL:

http://www imaedge.com

When the cursor is over a hyperlink area, depending on your client the cursor changes, and the client’s status line displays the target of the link. If your hyperlink is attached to an image but the image does not load in your client, the hyperlink can still be used.

The image or label object must be enabled in order for the hyperlink to work. If the hyperlink is placed any object other than an image or label, the link is not saved.
The image object acts as a container for a graphical image. The image can be one of the following types:

- BMP.
- GIF.
- JPG.
- PNG.
- XPM.
### Properties:

- **Alternate Text**
- **Enabled**
**Properties: (continued)**

- Field Name
- Frame Type
- Frame Width
- Height
- Image Path
- Index
- Left

Logic. Currently not used.

- Name
- Sizing
- Style
- Top
- Type
- Visible
- Width
Notes

The image types that you can use depends on what EDGE Client or your web browser supports. Also, some image types may display better than others.

Areas of an image can be specified as hyperlinks. A hyperlink is a specific location on the image that executes an action when it is clicked. To add a hyperlink to an image, use the Hyperlink Object.
**Input Field**

An input field object presents data from a file in the location you specify. Data in an input field can be updated directly by the agent.

**Properties:**
- Content Text
- Database
- Defining Expressions
Properties: (continued)

- Enabled
- Field Name
- Height
- Input Field Type
- Inserting Expressions
- Justification
- Left
- List Source
- Logic
- Multiline
- Name
- Scroll Bar
- Spell Check
- Spinner Increment
- Style
- Top
- Type
- Use Field Width
Properties: (continued)

Visible
Width
Word Wrap
Label

The label object allows you to place a label on the screen. Labels can be used, for example, to display messages or to identify the screen.

Areas of a label can be specified as hyperlinks. To add a hyperlink to a label, use the Hyperlink Object.
Properties:

Bullet Type
Properties: (continued)

Enabled
Height
Justification
Left
Multiline
Name
Numbering
Scroll Bar
Start Number
Style
Text
Top
Type
Use Text Width
Visible
Width
Word Wrap
**List Box**

A list box object allows you to display a list of data from which the agent can select one or more entries.
**Properties:**

- **Content Field** *(When Context type is Field.)*
- **Content Text** *(When Context type is Text.)*
- **Content Type**
- **Enabled**
- **Height**
- **Left**
- **Logic**
- **Logic Run On**
- **Multiple Selection**
- **Name**
- **Save As Text**
- **Sorted**
- **Style**
- **Top**
- **Type**
- **Value Field**
- **Visible**
- **Width**
**Notes**

You can either use a field as the source of the entries, or specify the entries as text on this screen. If a multivalued field is used as the source of the list, each value in the field becomes an entry in the list.

With Windows 95, a list box cannot contain more than 32,767 items. The total size in bytes is limited only by available memory. With Windows 3.1, the maximum number of characters that can be placed in a list box is 64K.

In operations, if there are more entries than can be displayed at one time, a scroll bar is displayed.

In EDGE Client, to select more than one entry, press the [CTRL] key simultaneously while left-clicking the selections. To deselect an entry, press the [CTRL] key while left-clicking the selection.

---

*Note:* If you plan to have agents select multiple entries, choose the Logic Run On property setting carefully. With multiple entries, each time you select an item in the list box, you are performing a selection change or a single click.
OLE Object

The OLE object is no longer supported. It is retained for backward compatibility.
Radio Button

A radio button object is used to indicate a true or false condition. When you click a radio button, the condition is set to true.
Properties:

- Caption
- Caption Type
- Enabled
- Height
- Left
- Logic
- Name
- Style
- Top
- Type
- Use Caption Width
- Value Field
- Visible
- Width

Notes

The state of the button is returned in the Value field property and can be one of the following:

- True  Radio button has been clicked.
The exact value returned for the true or false condition of the button can be specified as a guide option. The default is 1 for true and 0 for false. (For information on specifying Guide Options, see the section Guide Options.)

Radio buttons can be grouped, which allows only one button in the group to have a true condition. When one button in a group is clicked, it is set to true and all other buttons in the group are automatically set to false.

To group radio buttons, specify the Value Field for each button with the same field name, but give each a different value position. For example, to group three radio buttons to a field called PMT.METHOD, assign the first button to PMT.METHOD,1; the second button to PMT.METHOD,2; and the third to PMT.METHOD,3. During operations, if the agent clicks the first button, PMT.METHOD,1 is set to true; PMT.METHOD,2 and PMT.METHOD,3 are set to false. If the agent then clicks the second button, PMT.METHOD,2 is set to true and the other values are set to false.
The shape object allows shapes to be added to a screen. The following shapes can be specified:

- Line (horizontal, vertical, or diagonal).
- Rectangle.
- Ellipse.
- Rounded rectangles.
### Properties:
- Border Style
- Border Thickness
- Corner Radius
- Filled
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
</tr>
<tr>
<td>Left</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Shape</td>
</tr>
<tr>
<td>Style</td>
</tr>
<tr>
<td>Top</td>
</tr>
<tr>
<td>Type</td>
</tr>
<tr>
<td>Visible</td>
</tr>
<tr>
<td>Width</td>
</tr>
</tbody>
</table>
Tab Object

A tab object is used to create tab controls.

The tab object’s properties sheet has the following tabs:

- **Main**: Contains properties for the tab control.
- **Tab**: Contains properties specific to a tab on the control.
### Properties: Main Tab

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active tab data</td>
<td></td>
</tr>
<tr>
<td>Current tab</td>
<td>Tab1</td>
</tr>
<tr>
<td>Enabled</td>
<td>True</td>
</tr>
<tr>
<td>Entry logic</td>
<td></td>
</tr>
<tr>
<td>Exit logic</td>
<td></td>
</tr>
<tr>
<td>Fixed width</td>
<td>False</td>
</tr>
<tr>
<td>Height</td>
<td>2.81</td>
</tr>
<tr>
<td>Left</td>
<td>25.57</td>
</tr>
<tr>
<td>Multiline</td>
<td>False</td>
</tr>
<tr>
<td>Name</td>
<td>Tab Object1</td>
</tr>
<tr>
<td>Number of tabs</td>
<td>1</td>
</tr>
<tr>
<td>Style</td>
<td>Tab Object</td>
</tr>
<tr>
<td>Top</td>
<td>2.81</td>
</tr>
<tr>
<td>Type</td>
<td>Tab Object</td>
</tr>
<tr>
<td>Visible</td>
<td>True</td>
</tr>
<tr>
<td>Width</td>
<td>14.14</td>
</tr>
</tbody>
</table>
Properties: Main Tab (continued)

- Entry Logic
- Exit Logic
- Fixed Width
- Height
- Left
- Multiline
- Name
- Number of Tabs
- Style
- Top
- Type
- Visible
- Width
**Properties: Tab Tab**

- Enabled
- Entry Logic
- Exit Logic
- ID
- Title

**Notes**

A tab control contains a minimum of one or more tabs with objects. The following is an example of a tab control with three tab pages with screen objects on Tab1:
The tab object allows the developer to add tabs to the tab control and then add objects on those tab pages. Clicking the tab title activates the tab in the screen editor.

Currently, nested tab controls are not supported; that is, you cannot have a tab control on top of a tab control.

To add objects to a tab page, create the tab control first, then place screen objects on the tab page as you would place objects on a screen.
Objects are pasted on a tab page using the Paste Into command from the Edit menu.

The SET-PROPERTY verb can be used to enable or disable a tab. For more information, see the ID property. If a user selects a disabled tab, the previously selected tab is displayed. When a tab is enabled, you can use the ROUTE verb to route to an object on the tab page, but not to the tab itself.
**Table Object**

The table object is used to hold screen objects in the cells of the table. Each cell can contain one object.

The table object’s properties sheet is divided into the following two tabs:

- **Main**  
  Contains properties for the whole table.

- **Cell**  
  Contains properties specific for the current cell.
Properties: Main Tab

- Border
- Border Width
- Columns
Properties: Main Tab (continued)

Height
Left
Name
Rows
Style
Table Style
Top
Type
Use Columns Width
Use Rows Height
Visible
Width
**Properties: Cell Tab**

- Adjust Cell
- Adjust Child
- Column
- Height
- Row
- Width

**Notes**

Each table cell can contain only one object.
To add an object to a cell, move the object into the cell or create the object in the cell.

To resize a column or row, use one of the following methods:

- **Height** and **Width** properties on the **Cell** tab.
- Your mouse.

To resize a column, place your cursor on the vertical border of the column you want to resize (the cursor becomes a double arrow). Use your mouse to drag the border to your desired size.

To resize a row, place your cursor on the horizontal border of the row you want to resize (your cursor becomes a double arrow). Use your mouse to drag the border to your desired size.

To insert a column or row, use one of the following methods:

- **Columns** and **Rows** properties on the **Main** tab.
- Your keyboard. A row or column is added above or below the currently selected row or column.

Decide where you want the column or row added. Select the cell above or below it. Press [Insert].

To delete a column or row, use one of the following methods:
- **Columns** and **Rows** properties on the **Main** tab.
- Your keyboard.

Select a cell in the column or row you want to delete. Press [Delete].
Video Object

The video object allows a video clip to be played.

The following video file types are supported:

- MPEG-1 (.mpg, .mpeg, .mpv, .mp2, .mpa).
- Audio-Video Interleaved (.avi).
- Nonproprietary QuickTime files (.mov)
### Properties:

- **Auto Start**
- **Continuous Play**
Notes

To view the video, the client must have the appropriate software installed.
Some of the settings for the video object can be overridden at runtime. For example, at runtime the agent can display or hide the Playback Track Bar.
Object Properties

Each object is defined by a set of properties, such as the type of the object, the location of the object on the screen, and whether the object is visible to the agent. Some properties, such as the type of the object, cannot be changed. Some properties, such as the name of object, can be changed during the design of the screen. Other properties, such as position of the object on the screen, can be changed at runtime. (To change a property during runtime use the SET-PROPERTY verb.)

Object properties are modified through the object properties sheet. An example is displayed in Figure 20-2.

To display an object’s properties sheet, do any of the following:

- Add an object to the screen.
- Click the object on the screen.
- Select the object from the Object selection box.
Table 20-1 lists the object properties and whether they can be modified by the `SET-PROPERTY` verb (a √ next to the property indicates that it can be modified by `SET-PROPERTY`). Also, in each of the object property’s description, the `SET-PROPERTY` verb is specified whether it can be used to modify the property.
<table>
<thead>
<tr>
<th>Object Property</th>
<th>Modified By SET-PROPERTY Verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Tab Data</td>
<td></td>
</tr>
<tr>
<td>Adjust Cell</td>
<td></td>
</tr>
<tr>
<td>Adjust Child</td>
<td></td>
</tr>
<tr>
<td>Alternate Text</td>
<td></td>
</tr>
<tr>
<td>Amplitude</td>
<td></td>
</tr>
<tr>
<td>Auto Start</td>
<td></td>
</tr>
<tr>
<td>Background Sound</td>
<td></td>
</tr>
<tr>
<td>Border</td>
<td></td>
</tr>
<tr>
<td>Border Style</td>
<td></td>
</tr>
<tr>
<td>Border Thickness</td>
<td>✓</td>
</tr>
<tr>
<td>Border Type</td>
<td>✓</td>
</tr>
<tr>
<td>Border Width</td>
<td></td>
</tr>
<tr>
<td>Bottom Title</td>
<td></td>
</tr>
<tr>
<td>Bullet Type</td>
<td></td>
</tr>
<tr>
<td>Caption</td>
<td></td>
</tr>
<tr>
<td>Caption Type</td>
<td></td>
</tr>
<tr>
<td>Color</td>
<td></td>
</tr>
<tr>
<td>Column</td>
<td></td>
</tr>
<tr>
<td>Object Property</td>
<td>Modified By SET-PROPERTY Verb</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------</td>
</tr>
<tr>
<td>Columns</td>
<td></td>
</tr>
<tr>
<td>Content Field</td>
<td></td>
</tr>
<tr>
<td>Content Text</td>
<td></td>
</tr>
<tr>
<td>Content Type</td>
<td></td>
</tr>
<tr>
<td>Continuous Play</td>
<td></td>
</tr>
<tr>
<td>Corner Radius</td>
<td></td>
</tr>
<tr>
<td>Current Tab</td>
<td></td>
</tr>
<tr>
<td>Database</td>
<td></td>
</tr>
<tr>
<td>Defining Expressions</td>
<td>✓</td>
</tr>
<tr>
<td>Descending</td>
<td></td>
</tr>
<tr>
<td>Dimensions</td>
<td></td>
</tr>
<tr>
<td>Display Style</td>
<td></td>
</tr>
<tr>
<td>Embedded Field Style</td>
<td></td>
</tr>
<tr>
<td>Enabled</td>
<td>✓</td>
</tr>
<tr>
<td>Entry Logic</td>
<td></td>
</tr>
<tr>
<td>Exit Logic</td>
<td></td>
</tr>
<tr>
<td>Field Name</td>
<td></td>
</tr>
<tr>
<td>Fill</td>
<td></td>
</tr>
</tbody>
</table>
Table 20-1. Properties Modified by the SET-PROPERTY Verb (continued)

<table>
<thead>
<tr>
<th>Object Property</th>
<th>Modified By SET-PROPERTY Verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filled</td>
<td></td>
</tr>
<tr>
<td>Fixed Width</td>
<td></td>
</tr>
<tr>
<td>Frame Type</td>
<td></td>
</tr>
<tr>
<td>Frame Width</td>
<td></td>
</tr>
<tr>
<td>Graph Type</td>
<td></td>
</tr>
<tr>
<td>Height</td>
<td>√</td>
</tr>
<tr>
<td>Height (%)</td>
<td></td>
</tr>
<tr>
<td>ID</td>
<td>√</td>
</tr>
<tr>
<td>Image Location</td>
<td></td>
</tr>
<tr>
<td>Image Path</td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td></td>
</tr>
<tr>
<td>Input Field Type</td>
<td></td>
</tr>
<tr>
<td>Inserting Expressions</td>
<td>√</td>
</tr>
<tr>
<td>Interpret Expression</td>
<td></td>
</tr>
<tr>
<td>Justification</td>
<td></td>
</tr>
<tr>
<td>Left</td>
<td>√</td>
</tr>
<tr>
<td>Left (%)</td>
<td></td>
</tr>
<tr>
<td>Left Title</td>
<td></td>
</tr>
</tbody>
</table>
### Table 20-1. Properties Modified by the SET-PROPERTY Verb (continued)

<table>
<thead>
<tr>
<th>Object Property</th>
<th>Modified By SET-PROPERTY Verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Legend</td>
<td></td>
</tr>
<tr>
<td>Local Image</td>
<td></td>
</tr>
<tr>
<td>Line Style</td>
<td></td>
</tr>
<tr>
<td>List Source</td>
<td></td>
</tr>
<tr>
<td>Logic</td>
<td></td>
</tr>
<tr>
<td>Logic Run On</td>
<td></td>
</tr>
<tr>
<td>Multiline</td>
<td></td>
</tr>
<tr>
<td>Multiple Selection</td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>✓</td>
</tr>
<tr>
<td>Number of Tabs</td>
<td></td>
</tr>
<tr>
<td>Numbering</td>
<td></td>
</tr>
<tr>
<td>Open New Window</td>
<td></td>
</tr>
<tr>
<td>Path</td>
<td></td>
</tr>
<tr>
<td>Play/Stop Controls</td>
<td></td>
</tr>
<tr>
<td>Playback Track Bar</td>
<td></td>
</tr>
<tr>
<td>Point Symbol</td>
<td></td>
</tr>
<tr>
<td>Repeat Count</td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td></td>
</tr>
</tbody>
</table>
Table 20-1. Properties Modified by the SET-PROPERTY Verb (continued)

<table>
<thead>
<tr>
<th>Object Property</th>
<th>Modified By SET-PROPERTY Verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right Title</td>
<td></td>
</tr>
<tr>
<td>Row</td>
<td></td>
</tr>
<tr>
<td>Rows</td>
<td></td>
</tr>
<tr>
<td>Save As Text</td>
<td></td>
</tr>
<tr>
<td>Server Image</td>
<td></td>
</tr>
<tr>
<td>Scroll Bar</td>
<td></td>
</tr>
<tr>
<td>Sizing</td>
<td></td>
</tr>
<tr>
<td>Shape</td>
<td></td>
</tr>
<tr>
<td>Sorted</td>
<td></td>
</tr>
<tr>
<td>Spell Check</td>
<td>√</td>
</tr>
<tr>
<td>Spinner Increment</td>
<td></td>
</tr>
<tr>
<td>Start Number</td>
<td></td>
</tr>
<tr>
<td>Style</td>
<td>√</td>
</tr>
<tr>
<td>Table Style</td>
<td></td>
</tr>
<tr>
<td>Target Field URL</td>
<td></td>
</tr>
<tr>
<td>Target Screen</td>
<td></td>
</tr>
<tr>
<td>Target Type</td>
<td></td>
</tr>
<tr>
<td>Target URL</td>
<td></td>
</tr>
</tbody>
</table>
### Table 20-1. Properties Modified by the SET-PROPERTY Verb (continued)

<table>
<thead>
<tr>
<th>Object Property</th>
<th>Modified By SET-PROPERTY Verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text</td>
<td></td>
</tr>
<tr>
<td>Text Left (%)</td>
<td></td>
</tr>
<tr>
<td>Text Top (%)</td>
<td></td>
</tr>
<tr>
<td>Timer Information</td>
<td></td>
</tr>
<tr>
<td>Title</td>
<td></td>
</tr>
<tr>
<td>Title Alignment</td>
<td>✓</td>
</tr>
<tr>
<td>Toggle Hilite</td>
<td></td>
</tr>
<tr>
<td>Top</td>
<td>✓</td>
</tr>
<tr>
<td>Top (%)</td>
<td></td>
</tr>
<tr>
<td>Top Title</td>
<td></td>
</tr>
<tr>
<td>Type</td>
<td></td>
</tr>
<tr>
<td>Use Caption Width</td>
<td></td>
</tr>
<tr>
<td>Use Columns Width</td>
<td></td>
</tr>
<tr>
<td>Use Field Width</td>
<td></td>
</tr>
<tr>
<td>Use Rows Height</td>
<td></td>
</tr>
<tr>
<td>Use Text Width</td>
<td></td>
</tr>
<tr>
<td>Value Field</td>
<td></td>
</tr>
<tr>
<td>Visible</td>
<td>✓</td>
</tr>
</tbody>
</table>
### Table 20-1. Properties Modified by the SET-PROPERTY Verb (continued)

<table>
<thead>
<tr>
<th>Object Property</th>
<th>Modified By SET-PROPERTY Verb</th>
</tr>
</thead>
<tbody>
<tr>
<td>Volume</td>
<td></td>
</tr>
<tr>
<td>Width</td>
<td></td>
</tr>
<tr>
<td>Width (%)</td>
<td>√</td>
</tr>
<tr>
<td>Word Wrap</td>
<td></td>
</tr>
<tr>
<td>X Distance</td>
<td></td>
</tr>
<tr>
<td>X Label</td>
<td></td>
</tr>
<tr>
<td>X Left</td>
<td></td>
</tr>
<tr>
<td>X Right</td>
<td></td>
</tr>
<tr>
<td>Y Label</td>
<td></td>
</tr>
<tr>
<td>Y Left</td>
<td></td>
</tr>
<tr>
<td>Y Right</td>
<td></td>
</tr>
<tr>
<td>Z Distance</td>
<td></td>
</tr>
<tr>
<td>Z Label</td>
<td></td>
</tr>
</tbody>
</table>
**Active Tab Data**

The Active tab data property specifies the file where the data for the tab is retrieved.

This property cannot be modified by the SET-PROPERTY verb.

**Adjust Cell**

The Adjust cell property is used to size the cell in EDGE Developer based on the size of the object in the cell; can be one of the following:

- **True**: The cell is resized according to the size of the object. For example, if the object is larger than the cell, the cell adjusts to fit the object.
  
  The column width adjusts to the largest cell in the column.
  The row height adjusts to the tallest cell in the row.

- **False**: The cell is not resized.
  This is the default.

This property is on the **Cell** tab of the table object.

If Adjust cell and Adjust Child are both set to True, Adjust cell takes precedence.
**Adjust Child**

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>

The Adjust child property is used to size the objects being put into a cell in EDGE Developer; can be one of the following:

- **True** The object is resized to fit into the cell.
- **False** The object is not resized.

This is the default.

This property is on the *Cell* tab of the table object. Adjust child is recommended for use with images.

If Adjust child and Adjust Cell are both set to True, Adjust cell takes precedence.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>
**Alternate Text**

The Alternate text property specifies the text displayed in EDGE Client if the image cannot be displayed.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| Image Object |

**Amplitude**

The Amplitude property specifies the name of the field that contains data to be graphed. The data must be delimited by pipe symbols (|), and must be in the following order:

number of data sets
number of data points per data set
data1
...
datan

The following example has two data sets, with four data points per data set:
The Amplitude property is found on the Data tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

| Graph Object |

---

### Auto Start

The Auto start property specifies whether the audio or video clip starts automatically when a screen is routed to; can be one of the following:

- **True**: The audio or video clip is started automatically.
- **False**: The audio or video clip is not started automatically.

This is the default.
Background Sound

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Object</td>
</tr>
<tr>
<td>Video Object</td>
</tr>
</tbody>
</table>

The Background sound property specifies whether the audio clip starts automatically as a background sound when a screen is routed to (the audio object is not displayed in EDGE Client); can be one of the following:

- **True**  The audio clip becomes a background sound and the audio object is not displayed.
- **False** The audio clip is not a background sound; the audio object is displayed.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.
**Border**

The Border property specifies whether the table has a border. The border is applied to the whole table, including the inside borders. Can be one of the following:

- **True**  
  There is a border for the table.  
  This is the default.

- **False**  
  There is no border around the table.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

<table>
<thead>
<tr>
<th>Table Object</th>
</tr>
</thead>
</table>

**Border Style**

The Border style property specifies the style of the border line of the shape; can be one of the following:

- **Dash.**
- **Dot.**
- **None** (cannot be selected if the Shape property is set to Line).
- **Solid.** This is the default.

If Border style is set to Solid, the Border Thickness property can be specified.
Border Thickness

The Border thickness property (previously named Border width) specifies the width of the line in pixels used to border the object.

The default is 1 pixel.

The width thickens towards the inside of the object (the object is not resized). For example:

For the shape object, this property only has effect when the Border Style property is set to Solid. When the Border Style property is set to Dash or Dot, the Border thickness is always set to 1.
This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Box</td>
</tr>
<tr>
<td>Shape</td>
</tr>
</tbody>
</table>

**Border Type**

The Border type property (previously named Border style) specifies the border of the group box; can be one of the following:

- **Engraved**: A black border surrounds the box.
  
  This is the default.

- **Inset**: The top and left borders are black and the bottom and right are white, which gives a 3-dimensional look.

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group Box</td>
</tr>
</tbody>
</table>

**Border Width**

The Border width property specifies the width of the border for the outside edge of the table.

The default is 4.
This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

| Table Object |

**Bottom Title**

The Bottom title property specifies a title for the bottom of the graph, which is useful for describing the x-axis. The font specified by the graphing program is Times New Roman.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

| Graph Object |

**Bullet Type**

The Bullet type property specifies the type of bullet to display; can be one of the following:

- Point
- Square

This property is enabled when **Numbering** is set to Bullets.
This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

**Caption**

The Caption property specifies the text that is displayed on the screen to describe the associated object. The location of the text must be specified on the Caption type property.

Through this property, accelerator keys can be defined for command buttons. To add an accelerator key to a command button object, prefix a & (ampersand) to the letter you want to be the accelerator key. For more information, see the section Accelerator Keys.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box</td>
</tr>
<tr>
<td>Command Button</td>
</tr>
<tr>
<td>Radio Button</td>
</tr>
</tbody>
</table>
**Caption Type**

The Caption type property specifies the location of the caption text; can be one of the following:

- **Field**  
  Indicates the caption text is stored in a field.

- **Text**  
  Indicates the caption text is specified in the Caption property.  
  This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box</td>
</tr>
<tr>
<td>Command Button</td>
</tr>
<tr>
<td>Radio Button</td>
</tr>
</tbody>
</table>

**Color**

The Color property specifies the name of the field that contains color specifiers for the graph. For pie graphs, there must be one color specified for each data point. For all other graph types, there must be one color specified for each data set. The data must be delimited by pipe symbols ( | ), and must be in the following order:

- number of data sets (or data points if pie graph)
- color1
- ...
- colorn

The codes to use for colors are listed in Table 20-2.
The following example has two data sets; one data set is red and the other is blue:

\[ 2 \mid 4 \mid 1 \]

This property is optional. If not specified, default colors are used.

The Color property is found on the *Data* tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graph Object</strong></td>
</tr>
</tbody>
</table>

### Table 20-2. Color Codes for Graphs

<table>
<thead>
<tr>
<th>Code</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>black</td>
</tr>
<tr>
<td>1</td>
<td>blue</td>
</tr>
<tr>
<td>2</td>
<td>green</td>
</tr>
<tr>
<td>3</td>
<td>cyan</td>
</tr>
<tr>
<td>4</td>
<td>red</td>
</tr>
<tr>
<td>5</td>
<td>magenta</td>
</tr>
</tbody>
</table>
The Column property specifies the current column selected.

The Column property is found on the Cell tab on the table object.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| Table Object |  |
**Columns**

The number of columns in the table.

The default is 1.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>

**Content Field**

The Content field property specifies the name of the field that contains the text to be displayed in the list box. This property is enabled when the **Content Type** property is specified as field.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>List Box</td>
</tr>
</tbody>
</table>

**Content Text**

The Content text property specifies the text to be displayed in the list box. You can also specify the separator to be used with the list.

To create or modify a list, click the three-dot button next to the property. A dialog box similar to the following is displayed:
**Properties: Commands and Properties**

- **Add**: Adds an item to the list. Pressing ENTER also adds the item.
- **Apply**: Saves the list and the dialog box remains open.
- **Cancel**: The Cancel button closes the dialog box without saving any changes to the list. If items have changed, no warning is displayed. The Close box has the same effect.
- **Delete**: Deletes the selected item from the list.
- **OK**: Saves changes to the list and closes the dialog box.
- **Replace**: Replaces the selected item with the text entered at the top of the list.
**Properties: Commands and Properties (continued)**

**Separator**

The Separator list box specifies the separator to be used with the list; can be one of the following:

- Attribute marks (ASCII 30).
- Pipes (|).
- Sub-value marks (ASCII 28).
- Value marks (ASCII 29).

This property is enabled when the **Content Type** property is specified as text.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>List Box</td>
</tr>
</tbody>
</table>

**Content Type**

The Content type property specifies whether the contents of a list box are to be obtained from a field or from text entered with the object; can be one of the following:

- **Field** The content is stored in a field. The field is specified on the **Content Field** property.
**Text**

The content is text that is stored in the **Content Text** property. This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| List Box |

**Continuous Play**

The Continuous play property specifies whether an audio or video clip is played repeatedly; can be one of the following:

- True  The audio or video clip is played repeatedly.
- False The audio or video clip is not played repeatedly.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| Audio Object  |
| Video Object  |

**Corner Radius**

The Corner radius property is enabled when the **Shape** property is set to Rounded rectangle. This property is used to specify how round
the corners are; the value is specified in pixels. If the Corner radius property is set to 100, the shape becomes a circle.

The default is 15. The minimum value is 1.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
<th>Shape</th>
</tr>
</thead>
</table>

**Current Tab**

The current tab on the tab object that is active.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
<th>Tab Object</th>
</tr>
</thead>
</table>

**Database**

The Database property specifies the file to use when the List Source property is set to Database. This property is disabled if the Input Field Type property is set to Normal or if the List source property is not set to Database.
Defining Expressions

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Field</td>
<td></td>
</tr>
</tbody>
</table>

The Defining expressions property specifies whether expressions can be defined, that is, created, modified, and deleted in EDGE Client; can be one of the following:

- **True**: Expressions can be defined.
- **False**: Expressions cannot be defined.
  
  This is the default.

This property enables the Insert Expression command on the menu displayed when you right-click in an input field in EDGE Client. It also enables the Add, Modify, and Delete commands in the Expression Setup screen in EDGE Client. For more information, see the EDGE 7.11 Client Reference Manual.

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Field</td>
<td></td>
</tr>
</tbody>
</table>
The Descending property is displayed when the Shape property is set to Line. This property specifies how the line slopes and has no effect when the line is horizontal; can be one of the following:

- True  The line slopes downward, from left to right. For example:

  ![Image of descending line]

  This is the default.

- False The line slopes upward, from left to right. For example:

  ![Image of ascending line]

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

<table>
<thead>
<tr>
<th>Shape</th>
</tr>
</thead>
</table>

### Dimensions

The Dimensions property specifies whether the graph is displayed as a two- or three-dimensional object.

The default is three-dimensional.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

### Display Style

The Display style property specifies the style for the display field object in ASCII operations only. (EDGE Client uses the Style property.) The exact characteristics may depend on the terminal type. For example, on some terminals, the blink type causes the field to be displayed in bold type. (For information about defining styles, see the section **Styles**.)

One of the following can be selected:

- **Blink** The field blinks.
- **Normal** No special characteristics. This is the default.
- **Reverse** Causes the color to reverse.
This property cannot be modified by the SET-PROPERTY verb.

**Embedded Field Style**

The Embedded field style specifies the font style to use to display the embedded expression. *Styles* are defined as part of a scheme.

This property cannot be modified by the SET-PROPERTY verb.

**Enabled**

The Enabled property specifies whether the object is available to the agent. One of the following can be selected:

- **True**  
  The object is enabled, which means the object is displayed using the background color and text color values of the current scheme.  
  This is the default.

- **False**  
  The object and any links are disabled, which means the object uses the disabled background color and disabled text color values of the current scheme.
Note: For an agent to access a field, it must be able to get the focus and it must be both enabled and visible.

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects except for the graph object.</td>
</tr>
</tbody>
</table>

**Entry Logic**

The Entry logic property is available in the following locations:

- The Main tab of the tab object properties sheet. This is the tab control’s entry logic. The logic is invoked when you first enter the screen with the tab control and when you switch from one tab page to another.

- The Tab tab of the tab object properties sheet. This is the tab page’s entry logic. The logic is invoked when you switch to the tab page. If the tab page is the first tab page, this logic is also invoked.

You can select a logic from a list of existing logics or create your own logic by selecting the three-dot button, which invokes the logic editor.
This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

| Tab Object |

Exit Logic

The Exit logic is available in the following locations:

- The Main tab of the tab object property page. This is the tab control exit logic. The logic is invoked whenever you switch from one tab page to another.

- The Tab tab of the tab object property page. This is the tab page exit logic. The logic is invoked when you exit the tab page.

You can select a logic from a list of existing logics or create your own logic by selecting the three-dot button, which invokes the logic editor.

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

| Tab Object |
Field Name

The Field name property specifies the name of the field that contains the source of the data for the screen object.

Screen objects must obtain their values from fields in a file; variables cannot be used.

For an image object, the field name property specifies the image directory path and if specified, takes precedence over the Image Path property. If an image path or field name is not specified, no image is displayed in EDGE Client.

By clicking the three-dot button, the fields dialog box is displayed. Select a field and click OK.

To specify a value in a multivalued field, use the field name, followed by a comma and the value position. For example, to access the third value in the PROD.NO field, reference it as follows:

    PROD.NO,3
This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

<table>
<thead>
<tr>
<th>Audio Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Image Object</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
<tr>
<td>Video Object</td>
</tr>
</tbody>
</table>

**Fill**

The Fill property specifies the name of the field that has fill specified for the graph. For pie graph types, there must be one fill specified for each data point. For all other graph types, there must be one fill specified for each data set. The data must be delimited by pipe symbols (|), and must be in the following order:

- number of data sets (or data points if pie graph)
- fill1
- ...
- filln

The codes to use for Fill are listed in Figure 20-3.

This property is optional and is used only if the Graph Type is pie, or any of the bar or area types.

The following example has two data sets, one data set is solid and the other is downward diagonal lines:
The Fill property is found on the Data tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graph Object</strong></td>
</tr>
</tbody>
</table>

**Table 20-3. Fill Styles for Graphs**

<table>
<thead>
<tr>
<th>Code</th>
<th>Fill Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Solid fill. This is the default.</td>
</tr>
<tr>
<td>1</td>
<td>Null (empty fill).</td>
</tr>
<tr>
<td>2</td>
<td>Horizontal hatch lines.</td>
</tr>
<tr>
<td>3</td>
<td>Vertical hatch lines.</td>
</tr>
<tr>
<td>4</td>
<td>Downward diagonal lines.</td>
</tr>
<tr>
<td>5</td>
<td>Upward diagonal lines.</td>
</tr>
<tr>
<td>6</td>
<td>Vertical and horizontal lines.</td>
</tr>
<tr>
<td>7</td>
<td>Upward and downward diagonal lines.</td>
</tr>
</tbody>
</table>
**Filled**

The Filled property specifies whether the shape is filled in. This property is not displayed when the Shape property is set to Line. Can be one of the following:

- **True**  The shape is filled in.
  - This is the default.
- **False**  The shape is not filled in.

When Filled is set to False, you must place your cursor over the border in order to move the object.

When Filled is set to True and Border Style is set to None, the object can be invisible in EDGE Client. This occurs when the background color is the same color as the object.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

<table>
<thead>
<tr>
<th>Shape</th>
</tr>
</thead>
</table>

**Fixed Width**

The Fixed width property specifies how to determine the width of the tabs.
Screen Objects

Can be one of the following:

True  The tabs are all the same width, based on the width required for the longest tab Title.

False The width of each tab is based on the length of the title for that tab.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Frame Type**

The Frame type property specifies the type of frame; can be one of the following:

None  A frame is not added around the image.

Simple  A frame is added around the image.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

<table>
<thead>
<tr>
<th>Tab Object</th>
</tr>
</thead>
</table>

Objects With This Property

<table>
<thead>
<tr>
<th>Image Object</th>
</tr>
</thead>
</table>
**Frame Width**

The Frame Width property specifies the width of the frame. The default is 1.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Image Object</strong></td>
</tr>
</tbody>
</table>

---

**Graph Type**

The Graph type property specifies type of graph which can be one of the following:

- 3D-surface.
- Absolute area.
- Clustered bar.
- Line.
- Pie.
- Scatter.
- Simple bar.
- S-percent area.
- S-percent bar.
- Stacked area.
- Stacked bar.
- Z-clustered bar.

To view a sample graph, specify the type. The graph object displays a representative sample of the type of graph selected.
**Height**

The **Height** property specifies the height of the object, in the metric defined for the screen. The following are special considerations:

- For the shape object, this property is not displayed when the **Shape** property is set to Line.
- For the **Cell** tab of the table object, this property specifies the height of the cell. This affects the whole row. The default is the height of the table divided by the number of rows.
- For the video object, the object must be sized correctly so the clip and the controls can be displayed in EDGE Client. Currently, you cannot determine the exact height of video objects in EDGE Developer.

This property can be modified by the **SET-PROPERTY** verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graph Object</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>All objects.</strong></td>
</tr>
</tbody>
</table>
**Height (%)**

The Height (%) property specifies the percentage of the graphic height on the command button to display.

The Height (%) property is found on the *Icon* tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Button</td>
</tr>
</tbody>
</table>

**ID**

The ID property is used to specify the name of the tab. The SET-PROPERTY verb uses this ID to enable or disable the tab.

The ID property is found on the *Tab* tab on the tab object.

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab Object</td>
</tr>
</tbody>
</table>

**Image Location**

The Image Location property specifies whether the graphic is located on the client or the EDGE server. This property is available only if a graphic type is specified in the *Type* property.
One of the following can be selected:

- **Local**: The graphic file is saved on the client in the location specified in the **Local Image** property.
- **Server**: The graphic file is saved on the server. The name of the graphic is specified in the **Server Image** property.

The Image Location property is found on the *Icon* tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command Button</strong></td>
</tr>
</tbody>
</table>

**Image Path**

The Image path property specifies the location of the image. This property is used when the **Field Name** property is empty.

If you do not specify an image path or **Field Name**, no image is displayed in EDGE Client.

If both the **Field Name** property and Image path property have data specified, the Field name property takes precedence.
**Index**

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Image Object</td>
</tr>
</tbody>
</table>

The Index property specifies the location of the graphic in the library file; this property is available only if icon Type is ICL, ICL DLL, or BMP DLL.

The default is location 1.

The Index property is found on the Icon tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Button</td>
</tr>
</tbody>
</table>
**Input Field Type**

The input field type specifies the display format of the input field; can be one of the following:

- **Combo box**
  The agent can enter data in the edit portion of the box or click the down arrow to display the list to select from. The values in the list can come from the specified **Database** or from the **Content Text** property.
  The list can display up to ten items. A scroll bar is added to a list of eleven or more items.

- **Drop-down**
  The agent clicks the down arrow to display the list to select from. The values in the list can come from the specified database or from the **Content Text** property.
  The agent cannot enter a value that is not in the list.
  The list can display up to ten items. A scroll bar is added to a list of eleven or more items.

- **Normal**
  The data from the specified field is listed in the list box.
  This is the default.

- **Spinner**
  Spinner displays a spin box. A spin box is a standard Windows property that can be used to display predetermined values.
  The spin box is always displayed to the immediate right of the input field. The height of the spin box matches the height of the input box; the width of the spin box is one half of its height. The space required for the spin box is not included in the dimensions specified for the input box.
Inserting Expressions

The agent clicks the up or down arrows to make a selection.

If the input field type is changed from Normal or Spinner to Combo box or Drop-down, the object’s height is automatically adjusted.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

<table>
<thead>
<tr>
<th>Input Field</th>
</tr>
</thead>
</table>

The Inserting expressions property specifies whether expressions can be embedded into the input field in EDGE Client; can be one of the following:

- **True**: Expressions can be embedded.
- **False**: Expressions cannot be embedded.

This is the default.

This property enables the Insert Expression command on the menu displayed when you right-click in an input field in EDGE Client. It also enables the Insert in Field command in the Expression Setup screen in EDGE Client. For more information, see the EDGE 7.11 Client Reference Manual.
This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Interpret Expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects With This Property</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
</tbody>
</table>

The Interpret expression property specifies whether the embedded expression is interpreted; can be one of the following:

- **True**: The embedded expression is interpreted and displayed.
- **False**: The embedded expression is not interpreted and displayed in the embedded expression format.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Justification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects With This Property</td>
</tr>
<tr>
<td>Display Field</td>
</tr>
</tbody>
</table>

The Justification property specifies the alignment of the text displayed; can be one of the following:

- **Center**: Only supported with input field types of Normal and Spinner.
- **Left**: This is the default.
Screen Objects

Object Properties

- Right. Only supported with input field types of Normal and Spinner.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

Left

The Left property specifies the first screen column occupied by the object. When the metric is Row/Col, the first left column is 1. For all other metrics, it is 0 (zero).

For the shape object, this property is not displayed when the Shape property is set to Line.

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects.</td>
</tr>
</tbody>
</table>
**Left (%)**

The Left (%) property specifies how far from the left side of the command button the graphic is to start. 0 (zero) specifies it starts at the left side.

The Left (%) property is found on the *Icon* tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Button</td>
</tr>
</tbody>
</table>

**Left Title**

The Left title property specifies a title for the left side of the graph, which is useful for describing the y-axis. The font is specified by the graphing program as Times New Roman.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

**Legend**

The Legend property specifies the name of the field that has legend text for the graph. Legend text describes the data in the graph. For pie graph types, there must be one legend specified for each data
point. For all other graph types, there must be one legend specified for each data set. The legend text must be delimited by pipe symbols (|), and must be in the following order:

- number of data sets (or data points if pie graph)
- legend1
- ...
- legendn

The following example has two data sets, one is Apples and the other is Oranges:

2 | Apples | Oranges

This property is optional. If left blank, no legends are displayed. The font is specified by the graphing program as Arial.

The Legend property is found on the Data tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

**Line Style**

The Line style property specifies the name of the field containing the line style for the graph. This property is optional and is used only if the Graph Type is two-dimensional line or scatter. If not specified,
default line styles are used. If not specified, default line styles are used. The available styles are listed in Table 20-4.

The data must be delimited by pipe symbols (|) in the following order:

- number of data sets (or data points if pie graph)
- line style1
- ...
- line stylen

The Line style property is found on the Data tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Graph Object</strong></td>
</tr>
</tbody>
</table>

**Table 20-4. Line Styles for Graphs**

<table>
<thead>
<tr>
<th>Code</th>
<th>Line style</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>solid line</td>
</tr>
<tr>
<td>1</td>
<td>dashed line (----)</td>
</tr>
<tr>
<td>2</td>
<td>dotted line (....)</td>
</tr>
<tr>
<td>3</td>
<td>alternating dashes and dots (-.-.-)</td>
</tr>
</tbody>
</table>
List Source

The list source property specifies where the contents of the list are to be obtained; can be one of the following:

- **Database**: The contents are taken from the database file specified in the Database property.
- **Member of set**: The contents are taken from the Field Name property. This is the default.
- **Text**: The contents are taken from the Content Text property.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Code</th>
<th>Line style</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>dash - dot - dot (-..-..-..)</td>
</tr>
<tr>
<td>5</td>
<td>null (invisible) line</td>
</tr>
</tbody>
</table>

Local Image

The Local Image property specifies where the command button graphic is stored on the client. This property is available only if local is specified in the Image Location property. The location identifies the full client path and name of the graphic file.

Objects With This Property

Input Field
**Note:** The graphic must exist in the same path on each agent’s system.

The Local Image property is found on the *Icon* tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

### Logic

The Logic property specifies the name of the logic flow to execute for the object; can be specified in either of the following ways:

- By selecting an existing logic flow from the list box.
- By clicking the three-dot button, which opens the Logic editor and allows you to create a new logic flow.

Logic flows can be used, for example, to format a telephone number after the agent has entered it as an unformatted number. Or you can use a logic flow to verify an account number or to look up a product description based on a product number.
Note: Logic flows that have a Parameter Definition cannot be used in a Logic property.

After the logic flow has finished, the screen is automatically updated and any modifications to objects on the screen are displayed.

The action that invokes the logic flow depends on the object, as shown on Table 20-5.

Before performing the logic, EDGE stores information in the following fields in the SYS file for the duration of the logic flow:

<table>
<thead>
<tr>
<th>$ENTRY</th>
<th>A copy of the data the agent just entered for the object. $ENTRY is set only if the logic is invoked from an input field object.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If the agent presses RETURN without entering any new data, the contents of $ENTRY depend on the settings of the SET_DENTRY_NULL environment variable as follows:</td>
</tr>
<tr>
<td></td>
<td>SET_DENTRY_NULL $ENTRY value.</td>
</tr>
<tr>
<td>0 (zero)</td>
<td>$ENTRY contains the contents of the object.</td>
</tr>
<tr>
<td>1 (one)</td>
<td>$ENTRY is empty (&quot;&quot;).</td>
</tr>
<tr>
<td></td>
<td>SET_DENTRY_NULL is in the .user.profile file.</td>
</tr>
</tbody>
</table>
If you need to save the data from any of these fields, copy it to a field or variable that you have defined.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box</td>
</tr>
<tr>
<td>Command Button</td>
</tr>
<tr>
<td>Group Box</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
<tr>
<td>List Box</td>
</tr>
<tr>
<td>Radio Button</td>
</tr>
</tbody>
</table>
**Logic Run On**

The Logic run on property specifies when the logic flow is called for a list box object. If a single action causes multiple events (such as a single click and a selection change), the logic will only be called once. The setting can be one of the following:

- **all**
  The logic flow is called when any click or selection change occurs.

---

**Table 20-5. Actions That Invoke Logic Flows**

<table>
<thead>
<tr>
<th>Object</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box</td>
<td>Clicking the check box. The logic flow is invoked both when the check box is checked and when it is cleared.</td>
</tr>
<tr>
<td>Command Button</td>
<td>Clicking the command button.</td>
</tr>
<tr>
<td>Group Box</td>
<td>When any object inside the group box is modified after any logic attached to that object is performed.</td>
</tr>
<tr>
<td>Input Field</td>
<td>If data in the input field object has changed, changing the focus; for example, if you change data in the input field, then click another object, the logic flow is invoked. If data for the object has not changed, pressing ENTER or TAB when the input field object has the focus.</td>
</tr>
<tr>
<td>List Box</td>
<td>The logic flow is invoked based on the setting specified in the Logic Run On property.</td>
</tr>
<tr>
<td>Radio Button</td>
<td>Clicking the radio button.</td>
</tr>
<tr>
<td><strong>Action</strong></td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>double click</td>
<td>The logic flow is called when the user performs a double click. This is the default.</td>
</tr>
<tr>
<td>double click or selchange</td>
<td>The logic flow is called when the user double clicks or a selection change occurs.</td>
</tr>
<tr>
<td>selchange</td>
<td>The logic flow is called when a selection change occurs. This includes keyboard directional keys in addition to mouse clicks.</td>
</tr>
<tr>
<td>single click</td>
<td>The logic flow is called when the user performs a single click. This includes when the user clicks on the currently selected item.</td>
</tr>
<tr>
<td>single click or selchange</td>
<td>The logic flow is called when the user performs a single click or a selection change occurs.</td>
</tr>
<tr>
<td>single or double click</td>
<td>The logic flow is called when the user performs a single or double click.</td>
</tr>
</tbody>
</table>

Before performing the logic, EDGE stores information in the following fields in the SYS file for the duration of the logic flow:

<table>
<thead>
<tr>
<th><strong>$LB.CLICK</strong></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No click.</td>
</tr>
<tr>
<td>1</td>
<td>Single click.</td>
</tr>
<tr>
<td>2</td>
<td>Double click.</td>
</tr>
</tbody>
</table>
$LB.SELCHANGE  The $LB.SELCHANGE indicates if the selection has changed in the list box object; can be one of the following:

- null  Initial value.
- 0  No change has occurred.
- 1  Yes, the selection has changed.

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

| List Box |

**Multiline**

The multiline property allows text to be on one or more lines; can be one of the following:

- **True**  The multiline functionality is enabled. To enable the Multiline property, set the Use Field Width property to False. To complete the multiline functionality you must also set the Scroll Bar and Word Wrap properties.
  
  This is the default for the Label object.

- **False**  The multiline functionality is disabled.
  
  This is the default for the Display Field and Input Field objects.
If the multiline feature is enabled, a RETURN is converted to a multi-value mark ( ^\] ).

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

The Multiple selection property specifies whether more than one item can be selected from the associated list box; can be one of the following:

- **True** The agent can select multiple items.
- **False** The agent can select only one item.

This is the default.

When the agent makes a selection, the items are stored in the field specified on the **Value Field** property.

In operations, multiple items are selected as follows:

- **All items** Click on any item, hold down the control key and press slash (/).
Contiguous items  Click the first item, hold down the shift key and click the last item.
Non-contiguous items  Hold down the control key and click each item.

For more information on selecting from list boxes, see Chapter 1.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>List Box</td>
</tr>
</tbody>
</table>

**Name**

The Name property specifies a unique name for the object. The default name is the type of object followed by a unique identifier. For example, Label1 or DisplayField3.

This name is used to identify the object in the screen editor and can be changed by entering a new name in the Name property.
This property is used by the SET-PROPERTY verb to refer to the screen object.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects</td>
</tr>
</tbody>
</table>

**Number of Tabs**

The number of tabs on the tab object. If the number of tabs cannot be shown in the tab object width area, left and right arrows are displayed for scrolling during runtime in EDGE Client.

There is no limit on the number of tabs you can have on a tab object. The default is 1.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tab Object</td>
</tr>
</tbody>
</table>

**Numbering**

The Numbering property is used to create a bulleted list, definition list, or a numbered list; can be one of the following:

- None. This is the default.
- Bullets.
• Definition list.
• Numbers.

This property is enabled when the Multiline property is set to True.

A definition list is used, for example, to create a glossary of terms where every other value would be indented. For example:

```
Item1
    This is the definition for item1.
Item2
    This is the definition for item2.
Item3
    This is the definition for item3.
```

To create a definition list, use a multivalued field. Use value marks to separate the multiple values.

To create a numbered list, use \n to start a new line. For example, 1\n2\n3 displays as follows:

```
1
2
3
```
Open New Window

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

The Open new window object specifies whether a new window is opened when the link is clicked; can be one of the following:

- **True**: A new window is opened.
- **False**: A new window is not opened.
  This is the default.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hyperlink Object</td>
</tr>
</tbody>
</table>

Path

The Path property specifies the directory path of the audio or video clip. This path is used when the Field Name property is empty.
**Play/Stop Controls**

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Object</td>
</tr>
<tr>
<td>Video Object</td>
</tr>
</tbody>
</table>

The Play/stop controls property specifies whether the play and stop controls are shown; can be one of the following:

- **True**: The play and stop controls are shown.
- **False**: The play and stop controls are not shown.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Object</td>
</tr>
<tr>
<td>Video Object</td>
</tr>
</tbody>
</table>

**Playback Track Bar**

The Playback track bar property specifies whether a window with time elapsed information is shown; can be one of the following:

- **True**: The time elapsed information is shown.
False  The time elapsed information is not shown.
This is the default.

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

| Audio Object |
| Video Object |

Point Symbol

The Point symbol property specifies the name of the field that contains data point specifiers for the graph. The data must be delimited by pipe symbols ( | ), and must be in the following order:

   number of data sets
   point symbol1
   ...
   point symboln

The codes to use for point symbols are listed in Table 20-6.

This property is optional and is used only if the Graph Type is line or scatter. If not specified, representative point symbols are used.

The Point symbol property is found on the Data tab on the graph object.
**Repeat Count**

The Repeat count property specifies the number of times to repeat an audio or video clip. This property is ignored if the Continuous Play property is True.

The default is 0.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Object</td>
</tr>
<tr>
<td>Video Object</td>
</tr>
</tbody>
</table>

**Table 20-6. Point Styles for Graphs**

<table>
<thead>
<tr>
<th>Code</th>
<th>Print style</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Plus (+).</td>
</tr>
<tr>
<td>1</td>
<td>Cross (x).</td>
</tr>
</tbody>
</table>
The Result property identifies the outcome of a link. The three-dot box can used to create a result.

For more information, see the section Results in Chapter 24, Finishing Touches.
### Right Title

The Right title property specifies text to be used as a title for the right side of the graph. The font is specified by the graphing program as Times New Roman.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hyperlink Object</td>
</tr>
</tbody>
</table>

### Row

The Row property specifies the current row selected.

The Row property is found on the Cell tab on the table object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>
**Rows**

The Rows property specifies the number of rows in the table.

The default is 1.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>

**Save As Text**

The Save as text property specifies whether EDGE Client stores the text or the index of the selection in the Value Field property; can be one of the following:

- **True**
  - The text is saved. The text is the text displayed in the list box.
  - This is the default.

- **False**
  - The index is saved. The index is the location of the selection in the list box. For example, 3 indicates the third selection in the list.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>List Box</td>
</tr>
</tbody>
</table>
**Scroll Bar**

The Scroll bar property specifies whether scroll bars are displayed and, if so, how they are displayed. This property is enabled when the **Multiline** property is set to True. Can be one of the following:

- **Both**: Both horizontal and vertical scroll bars are displayed on the object, if necessary.
- **Horizontal**: A horizontal scroll bar is displayed to allow the agent to scroll the text left and right.
- **None**: Scroll bars are not displayed. This is the default.
- **Vertical**: A vertical scroll bar is displayed to allow the agent to scroll the text up and down.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

**Server Image**

The Server Image property specifies the name of the graphic file stored on the server. This property is available only if server is specified in the **Image Location** property. The graphic can be specified in either of the following ways:
By selecting an existing graphic from the drop-down list box.
(The existing graphic files were saved to the server using the
Server Images tab.)

By clicking the three-dot button, which allows you to select
and save a new graphic from your PC to the server.

The Server Image property is found on the Icon tab on the command
button object.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Button</td>
</tr>
</tbody>
</table>

### Sizing

The Sizing property specifies how the image is sized; can be one of
the following:

- **Crop**
  - The image is cropped to fit the image object.
  - This is the default.
- **Resize to fit**
  - Resize the image object to fit the image.
- **Stretch to fit**
  - Resize the image to fit in the image object.

Depending on the setting of this property, the location of hyperlinks
on an image may change in operations. This is because hyperlinks
are attached to the *image object* and not the *image*. AIT recommends you set Sizing to Resize to fit and that your *image object* is not much larger than the *image*.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| Image Object |

---

**Shape**

The Shape property specifies the shape of the object; can be one of the following:

- Ellipse.
- Line.
- Rectangle. This is the default.
- RoundedRectangle.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| Shape |

---
**Sorted**

The Sorted property specifies whether the items specified in the Content Text property are sorted in ascending order; can be one of the following:

- **True**  The list is sorted.
- **False** The list is not sorted.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

| List Box |

**Spell Check**

The Spell check property specifies the mode to set the spelling checker in EDGE Client; can be one of the following:

- **True**  The spelling checker is in automatic mode, a space invokes it.
- **False** The spelling checker is in manual mode and must be invoked by right clicking anywhere in the input field and then selecting the Spell check command.

This is the default.
This property can be modified by the SET-PROPERTY verb.

### Spinner Increment

The Spinner increment property specifies the amount of increment for the spin box. For example, if the increment is 1, the number displayed in the spinner is incremented or decremented by 1 each time the up or down arrow is pressed.

When a selection is made from the spin box, it is stored under the fieldname for the object. You can use the fieldname in a logic flow to determine whether the value is appropriate.

The default is 1.

This property is enabled only if the Input Field property is set to Spinner.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Field</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Field</td>
</tr>
</tbody>
</table>
**Start Number**

The Start number property specifies the value to start the numbered list.

This property is enabled when Numbering is set to Numbers.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

**Style**

The Style property specifies a style for the object. Each object has a default style, which is the same as the object type. For example, the default style for a groupbox object is `groupbox`.

*Styles* are defined as part of a scheme. A default scheme can be set at the EDGE, project, and operator access levels. Currently, the tab control color scheme is controlled by your settings in Microsoft Windows. By default, the color scheme for the foreground text is black and disabled text is gray.
**Table Style**

This property can be modified by the `SET-PROPERTY` verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects.</td>
</tr>
</tbody>
</table>

The Table style property specifies the visual or dimensional effect of the table object; can be one of the following:

- 2D
- 3D

The default is 3D. However, if the `Border Width` property is set to less than 4, the table is displayed in 2D.

When the Table style is set to 2D, a 3D object such as a Display field added to the table is not affected.

This property cannot be modified by the `SET-PROPERTY` verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>

**Target Field URL**

The Target field URL property specifies a field that contains a valid URL to route to when the link is clicked. For example, a URL can begin...
with http, mailto, or ftp. For a complete list of URL types, refer to your System Administrator.

The URL is passed to the default browser. If the URL is invalid, the browser displays an error message.

For example, this property can be used to specify a field that contains an email address. An agent can send an email to a customer by clicking the hyperlink to bring up the default browser. The email screen is displayed with the customer’s email address, similar to the following:
If you have a specific URL you want to specify instead of using the Target field URL property you can use the Target URL property.

The Target field property is enabled when Target Type is URL Field.

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

| Hyperlink Object |

**Target Screen**

The Target screen property specifies the screen to route to when the link is clicked. This property is enabled when Target Type is Screen.

This property cannot be modified by the SET-PROPERTY verb.

Objects With This Property

| Hyperlink Object |

**Target Type**

The Target type property specifies the target of the link when it is clicked; can be one of the following:

- **None** When the link is clicked, routing does not occur; instead a logic can be executed or a result can be set.
Target URL

The Target URL property specifies a valid URL to route to when the link is clicked. For example, a URL can begin with http, mailto, or ftp. For a complete list of URL types, refer to your System Administrator.

The URL is passed to the default browser. If the URL is invalid, the browser displays an error message.

This property is enabled when Target Type is URL.

To specify a field that contains a URL instead of specifying one directly, use the Target Field URL property.
**Text**

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hyperlink Object</td>
</tr>
</tbody>
</table>

The Text property specifies the text to be displayed for the label object. For example, if you enter *Customer Name* in this box, this text is displayed as the label. By default, the Name of the label is displayed.

To start a new line, separate the items with \
. For example, Item1
\nItem2
\nItem3 displays as follows:

```
  Item1
  Item2
  Item3
```

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

**Text Left (%)**

The Text left property specifies a percentage value on how far from the left side of the command button the caption is to start. Zero (0)
specifies it starts farthest from the left. Do not enter the % (percent) sign.

The drop down box allows you to select Center which will center the text on the command button. All other entries must be entered as a value.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Text Top (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Text top property specifies a percentage value on how far from the top of the command button the caption is to start. Zero (0) specifies it starts at the very top.</td>
</tr>
<tr>
<td>The drop down box allows you to select Center which will center text on the command button. All other entries must be entered as a value.</td>
</tr>
<tr>
<td>This property cannot be modified by the SET-PROPERTY verb.</td>
</tr>
<tr>
<td>Objects With This Property</td>
</tr>
<tr>
<td>Command Button</td>
</tr>
</tbody>
</table>

Objects With This Property
**Timer Information**

The Timer information property specifies whether elapsed time information is shown; can be one of the following:

- **True**  The timer information is shown.
- **False** The timer information is not shown.

This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

- Audio Object
- Video Object

**Title**

The Title property specifies the title to be displayed on the group box or tab sheet. If not specified, the Name property is used.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

- Group Box
- Tab Object

**Title Alignment**

The Title alignment property specifies alignment for the title on the group box; can be one of the following:
**Toggle Hilite**

The Toggle hilite property indicates whether the highlight condition of the command button is to remain after the mouse is released. When a command button is clicked, it is said to be highlighted. The condition of the button is considered to be true if the button is highlighted and false otherwise.

By default, the command button is highlighted only while the mouse is held down. As soon as the mouse is released, the command button

This property can be modified by the SET-PROPERTY verb.

---

**Objects With This Property**

<table>
<thead>
<tr>
<th>Group Box</th>
</tr>
</thead>
</table>

- Centered.
- Hidden (title is not displayed).
- Left. This is the default.
- Right.
is no longer highlighted. This property can change the default; can be one of the following:

- **True**: The highlight condition remains until the command button is clicked again. That is, the condition of the command button is changed (toggled) only when the button is clicked. Releasing the mouse has no effect.

- **False**: The condition of the command button is true only when the mouse is clicked. When the mouse is released, the condition reverts to false.

  This is the default.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Command Button</strong></td>
</tr>
</tbody>
</table>

**Top**

The Top property specifies the top position occupied by the object. When the metric is Row/Col, the first top position is 1. For all other metrics, it is 0 (zero).

The bottom row depends on the Show/Hide Function Keys. For more information, see the section Screen Tools.
### Top (%)

This property specifies how far from the top of the command button the graphic is to start. 0 (zero) specifies it starts at the top.

The Top (%) property is found on the Icon tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects.</td>
</tr>
</tbody>
</table>

### Top Title

The Top title property specifies text to use as a title for the top of the graph, which is useful in naming the graph. The font is specified by the graphing program as Times New Roman.
This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

**Type**

The Type property is used in one of the following ways:

- To display the type of object. This is display only.
- On the *Icon* tab of the command button object, type specifies the type of icon used; can be one of the following:
  
  (None) This is the default.
  BMP
  BMP DLL
  ICL
  ICO
  ICO DLL
  Others (jpeg, jpg, gif, png)

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects.</td>
</tr>
</tbody>
</table>
**Use Caption Width**

The Use caption width property indicates whether the width of the object is to be based on the Width property or the Caption property.

One of the following can be selected:

- **True**: The object width is set by the length of the text in the Caption property.
  
  This is the default.

- **False**: The object width is set by the Width property.

This property cannot be modified by the SET-PROPERTY verb.

---

**Use Columns Width**

The Use columns width property specifies whether the width of all the columns is the width of the table; can be one of the following:

- **True**: The width of the table is the width of all the columns.

- **False**: The width of the table does not use the width of all the columns.

  This is the default.
**Use Field Width**

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table Object</td>
</tr>
</tbody>
</table>

The Use field width property indicates whether the width of the object is to be based on the **Width** property or the **Field Name** property.

One of the following can be selected:

- **True**
  - The object width is set by the length of the characters in the **Field Name** property.
  - This is the default.

- **False**
  - The object width is set by the **Width** property.

If an input field object is multilined, the Use field width property is not applicable. The **Storage width** property determines the number of characters for multiline input field objects. If the Storage width property is null or zero, there is no limit.

If an input field object is not multilined, and Use field width is set to **True**, the Use field width property allows the field to contain or display up to the length of characters in the **Field Name** property.
**Use Rows Height**

The Use rows height property specifies whether the height of all the rows is the height of the table; can be one of the following:

- **True**  The height of the table is the height of all the rows.
- **False** The height of the table does not use the height of all the rows. This is the default.

This property cannot be modified by the SET-PROPERTY verb.

**Objects With This Property**

<table>
<thead>
<tr>
<th>Display Field</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Field</td>
</tr>
</tbody>
</table>

**Use Text Width**

The Use text width property specifies how to set the width of the object; one of the following can be selected:

- **True**  The width of the label is calculated by the length of the text in the **Text** property and the font defined by the **Style** property.
False  The width of the label is set by the Width property.
This is the default.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

**Value Field**

The Value field property specifies the name of the field or variable that is to receive values from the object. The values that it receives depend on the object. For example, if the object is a list box, the indices or text of the selected items are returned to the field or variable specified here. If the object is a check box, command button, or radio button, the current condition (that is, true or false) is returned. (The exact values returned for the true or false condition of the button can be specified in Guide Options. The default is 1 for true and 0 for false.)
Visible

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check Box</td>
</tr>
<tr>
<td>Command Button</td>
</tr>
<tr>
<td>List Box</td>
</tr>
<tr>
<td>Radio Button</td>
</tr>
</tbody>
</table>

The Visible property specifies whether the object is displayed in EDGE Client. When the property is not visible, it is not displayed on the agent’s screen. In the screen editor, the object is visible, but displayed as grayed.

One of the following can be selected:

- **True**: The object is visible.
  - This is the default.
- **False**: The object is not visible.

The Visible property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects.</td>
</tr>
</tbody>
</table>
**Volume**

The setting for how loud your audio file is played at; can be from 1 - 100 where 100 is the loudest.

This property cannot be modified by the SET-PROPERTY verb.

**Width**

The Width property specifies the number of horizontal screen columns used by the object. The width can be set by dragging the object the number of desired characters, by explicitly entering the number of characters in the Width property, or by setting the Use ... width property. If the Use ... width property is checked, you cannot explicitly set the Width property.

For the shape object, this property is not displayed when the Shape property is set to Line.

For the table object, the table is always as wide as the maximum width given by the total width of the cells on any given row. This is specified in the table object’s Main tab.

*Note: For an agent to access a field, the object must be able to get the focus and be both enabled and visible.*
For the table object, the width of the cell affects the whole column. The default is the width of the table divided by the number of columns. This is specified in the table object’s Cell tab.

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>All objects.</td>
</tr>
</tbody>
</table>

**Width (%)**

The Width (%) property specifies the percentage of the graphic width on the command button to display.

The Width (%) property is found on the Icon tab on the command button object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command Button</td>
</tr>
</tbody>
</table>
**Word Wrap**

The Word wrap property specifies whether the text is to wrap from one line to the next or continue to scroll to the right; can be one of the following:

- **True**  The automatic word wrap feature is enabled.
  This is the default for the **Label** object.
- **False**  The automatic word wrap feature is disabled.
  This is the default for the **Display Field** and **Input Field** objects.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Field</td>
</tr>
<tr>
<td>Input Field</td>
</tr>
<tr>
<td>Label</td>
</tr>
</tbody>
</table>

**X Distance**

The X distance property specifies the name of the field that contains values which specify the horizontal position of data points on x-axis style graphs. There must be one value specified for each data set or one value for each data point.

If there is one value per data set, the same horizontal positioning is used for the points in every data set.
The data must be delimited by pipe symbols ( | ), and must be in the following order:

   number of data sets or data points.
   position1
   ...
   position nn

This property is optional and is not used if the graph is a pie type.

The X distance property is found on the Data tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

### Objects With This Property

| Graph Object |

---

**X Label**

The X label property specifies the name of the field that contains text strings for data points on x-axis style graphs. There must be one value specified for each data set or data point.

The data must be delimited by pipe symbols ( | ), and must be in the following order:

   number of data sets or data points.
   position1
... positionn

The X label property is optional and is not used if the graph is a pie type. If not specified, the x-axis is automatically labeled with numbers. The font is specified by the graphing program as Courier New.

The X label property is found on the Data tab on the graph object.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

**X Left**

The x left property specifies the x coordinate of the left end of the line.

This property is displayed when the **Shape** property is set to Line.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape</td>
</tr>
</tbody>
</table>
**X Right**

The x right property specifies the x coordinate of the right end of the line.

This property is displayed when the *Shape* property is set to Line.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape</td>
</tr>
</tbody>
</table>

**Y Label**

The Y label property specifies the name of the field that contains values which specify text strings for data points on the y-axis of an axis-style graph. There must be one value specified for each data set or data point. The data must be delimited by pipe symbols ( | ), and must be in the following order:

- number of data sets or data points.
- position1
- ...
- positionn

This property is optional and is not used if the graph is a pie type. If not specified, the y-axis is automatically labeled with numbers. The font is specified by the graphing program as Courier New.

The Y label property is found on the *Data* tab on the graph object.
This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

**Y Left**

The y left property specifies the y coordinate of the left end of the line.

This property is displayed when the Shape property is set to Line.

This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape</td>
</tr>
</tbody>
</table>

**Y Right**

The y right property specifies the y coordinate of the right end of the line.

This property is displayed when the Shape property is set to Line.
This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shape</td>
</tr>
</tbody>
</table>

**Z Distance**

The Z distance property specifies the name of the field that contains values which specify the vertical position of data points on the z-axis of a three-dimensional axis-style graph. There must be one value specified for each data set or one value for each data point.

If there is one value per data set, the same vertical positioning is used for the points in every data set.

The data must be delimited by pipe symbols ( | ), and must be in the following order:

- number of data sets or data points.
- position1
- ...
- positionn

This property is optional and is not used if the graph is a pie type.

The Z distance property is found on the Data tab on the graph object.
This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>

**Z Label**

The Z label property specifies the name of the field that contains values which specify text strings for data points on the z-axis of a three-dimensional axis-style graph. There must be one value specified for each data set or data point.

The data must be delimited by pipe symbols (|), and must be in the following order:

- number of data sets or data points.
  - position1
  - ...
  - positionn

This property is optional and is not used if the graph is a pie type. If not specified, the z-axis is automatically labeled with numbers. The font is specified by the graphing program as Courier New.

The Z label property is found on the Data tab on the graph object.
This property cannot be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Objects With This Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Graph Object</td>
</tr>
</tbody>
</table>
Overview

EDGE uses ActiveX controls to extend the functionalities of EDGE screen objects and to provide custom services. ActiveX controls are software components produced by third party vendors and are available from many sources. The following are examples of functions that can be performed from EDGE using ActiveX controls:

- Sending an email.
- Adding pop-up menus to an application.
- Providing a graphical calendar for date display and user entry.
- Providing the ability to execute commands at a specified time interval.

This chapter discusses the following:

- Installing and Registering ActiveX Controls.
- Configuring ActiveX Controls.
ActiveX Controls Overview

- Wrappers.
- Creating Wrappers.
- Adding ActiveX Objects to Screens.
- Specifying ActiveX Property Values.
- Using with EDGE Client.

For general information on using ActiveX controls on user screens, see Chapter 20. For information on using controls in logic flows, see the discussion of CALL-EXTERNAL-FUNCTION.
Installing and Registering ActiveX Controls

Before you can begin to work with ActiveX controls in EDGE, you must take care of the following:

- Downloading ActiveX Controls.
- Registering ActiveX Controls.

Downloading ActiveX Controls

When you install EDGE Developer, you have the option of installing the ActiveX controls that are provided with EDGE. If other ActiveX controls are desired, they must be downloaded to your PC.

Before you can use ActiveX controls in your development environment, they must exist on your client PC and be configured in EDGE. For more information, see the section Configuring ActiveX Controls.

In EDGE Client, if an ActiveX control is used in a screen of a guide, EDGE Client first searches for the control on the local PC. If the ActiveX control is not on the PC, EDGE Client attempts to download the control from the web server via the Code location (URL), which is specified when the control is configured. The Code location can also be used to install a control on multiple PCs. If this is not successful, EDGE Client ignores the ActiveX control.

ActiveX controls are not available in ASCII operations.
Registering ActiveX Controls

An ActiveX control must be installed on the client and registered before it can be configured in EDGE. Typically an ActiveX control is registered automatically when it is installed. If it is not registered, use the Windows utility REGSVR32.EXE to register the control. If you do not have this program, you can download it from Microsoft at the following website:

http://www.microsoft.com

A registered control may require licensing. It is important to license an ActiveX control if required; otherwise, the ActiveX control may not function correctly. For example, the ActiveX control may load in EDGE Client with some functions disabled or not load at all.

Once a control is registered, you must configure it in EDGE using the ActiveX Configuration page before it can be added to your screens. For more information, see the section Configuring ActiveX Controls.

Upgrading ActiveX Controls

When you install an updated version of an ActiveX control that was configured in EDGE, you must reconfigure the control. To do so, select the control to be modified and select the ActiveX control to be updated. After you select the control and press OK, the control is updated. This means any property settings you had made are lost and must be reconfigured.
<table>
<thead>
<tr>
<th>Deleting <strong>ActiveX Controls</strong></th>
<th>When you install an updated version of a control after creating logic flows and screens using the control, it is possible that your logic flows and screens can have problems.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If you delete the configuration for an ActiveX control that is used on a screen, an error message is displayed in the screen editor and EDGE Client indicating the control is not available. Deleting an ActiveX control does not delete objects from the screens.</td>
</tr>
</tbody>
</table>
ActiveX controls have properties, events, and methods that can be configured and then used in the screen editor. To display the ActiveX controls configured on the client and server, select the following:

EDGE / Guide / ActiveX tab

A tabbed page similar to the following is displayed:
The ActiveX Controls list box displays the names of the controls configured for EDGE. A command button is used to toggle the list between the following listings:

- ActiveX Controls on Client; displays controls that are installed on your PC and configured on the EDGE server. Only these controls can be used in the screen editor.

- ActiveX Controls on Server; displays all of the controls that are configured on the EDGE server.

When you add or modify an ActiveX control, a page similar to the following is displayed:
**Properties:**

- **General properties**
  The general configuration properties.

- **License Information**
  Information about licensing.

- **Configuration properties**
  Tabs used to configure the properties, events, and methods for each ActiveX control.
Properties: (continued)

ActiveX control The control you are configuring.

The three-dot button is used to display a list of controls installed on the client. When clicked, a dialog box similar to the following is displayed:

Properties: General
### Properties: General (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveX name</td>
<td>The EDGE name for the control. A name must be entered before the other properties are enabled. Once this page is saved, this name cannot be modified.</td>
</tr>
<tr>
<td>CLSID</td>
<td>CLSID stands for class identifier. It is a unique identifier for an ActiveX control. This is display only.</td>
</tr>
<tr>
<td>Code location (URL)</td>
<td>The location from which the ActiveX control can be downloaded. The location is a URL (Universal Resource Locator); such as <a href="http://www.xyz.com/calendar.exe">www.xyz.com/calendar.exe</a>. The ActiveX control can be an .OCX, .DLL, .EXE, .CAB, or an .INF file. At runtime, if an ActiveX control is not found on the client, its executable code is retrieved from this location. This property is optional. However, if the location of a control’s executable code is not specified, and the agent’s PC does not have the control installed, the control will not load.</td>
</tr>
</tbody>
</table>
### Properties: General (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A description entered by the guide developer about the ActiveX control.</td>
</tr>
<tr>
<td></td>
<td>This is optional.</td>
</tr>
</tbody>
</table>

### Properties: License Information

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Key available</td>
<td>The Key available check box indicates whether a license key for the control is stored on the server. If checked, a key is stored and thus a license is not required.</td>
</tr>
<tr>
<td></td>
<td>This is display only.</td>
</tr>
<tr>
<td>Licensed</td>
<td>The Licensed check box indicates whether the ActiveX control requires a license at runtime. If checked and the Key available property is not checked, you must use the Update License command to retrieve and store the license.</td>
</tr>
<tr>
<td></td>
<td>This is display only.</td>
</tr>
<tr>
<td>Update License</td>
<td>The command button is used to retrieve license information.</td>
</tr>
<tr>
<td></td>
<td>If the license is obtained after configuring a control, you need to install the .LIC file for the control in the same folder where the control resides, then press the Update License command button.</td>
</tr>
</tbody>
</table>
Properties: Configuration

The configuration properties are on the following tabs:

- Properties Tab.
- Events Tab.
- Methods Tab.

Properties Tab

The Properties tab on the ActiveX Configuration page allows you to determine whether a property is to be displayed on the screen editor properties sheet and how a property is specified in the screen editor.

The order in which the properties are saved affects the configuration of the control. This is because one property may need the value of another property.

Typically you should not need to change the order of the properties. However, if you need to change the order, select the property and use the spinner control arrows to move the property up or down in the list.

If the property names in the list box are longer than the width of the box, double-click the name and the whole property name is displayed in a separate message box.

When you select a property from the list, its default property information is displayed.
The Properties tab is similar to the following:

![Properties Tab Diagram]

**Properties: Properties Tab**

- **ActiveX data type**: The ActiveX control’s data type.
  - This is display only.

- **EDGE data type**: Determines how values for the property can be specified when creating a screen object. For example, if *Yes/No list* is selected, *Yes* or *No* can be selected from a list.
  - The EDGE data types available for selection depend on what the ActiveX data type is. Table 21-1 describes the EDGE data types.
  - If the EDGE data type for the property is an EDGE field or variable, the value is assigned at runtime.
If, after assembling the guide, you modify the configuration of an ActiveX control by changing the EDGE data type, you must update all the screens that use the control and reassemble the guide.

Enumeration items

Enumeration items provide the values that can be assigned to the property. For example, the SortOrder property can be assigned the values 0-Ascending or 1-Descending.

If an EDGE field or variable is selected as the data type instead of the enumeration type, the field must contain an integer value representing the enumeration item. For example, the SortOrder property has the following enumeration items: 0-Ascending and 1-Descending. To specify descending sort order, the EDGE field or variable should contain the number 1.

This is display only.

Show on properties sheet

This check box specifies whether the selected ActiveX property is displayed in the properties sheet for the ActiveX screen object.

If checked, the property is available in the ActiveX tab of the screen editor’s properties sheet.

If not checked, the property is not available in the screen editor’s properties sheet. However, the property can be specified through the SET-PROPERTY verb.
Properties: Properties Tab  (continued)

If one or more of the ActiveX properties have this check box checked, then an ActiveX tab is added to the screen editor’s property sheet.

The default is checked.

Note:  Before saving the configuration, you should determine whether the setting of the EDGE data type is appropriate for your needs. For example, if you want to set a property through a field or variable, you should specify the data types as EDGE field or variable.

Table 21-1 provides descriptions of the EDGE data types which determine how the property can be specified on the screen editor. For example, if you configure a control property as a Yes or No list, the screen object property displays a list from which you can select either Yes or No.

Table 21-1. Description of EDGE Data Types

<table>
<thead>
<tr>
<th>EDGE Data Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Color</td>
<td>A number derived from the BGR (blue, green, red) values for the color. For more information, see the section Specifying Color.</td>
</tr>
<tr>
<td>EDGE field or variable</td>
<td>An EDGE field or variable can be specified.</td>
</tr>
</tbody>
</table>
Events Tab

An event is a message sent by Windows when you cause something to happen such as clicking an object. A guide developer can attach logic flows to execute when an event occurs.

The Events tab displays the ActiveX control’s events. For more information about the arguments, see your ActiveX control documentation.
ActiveX Controls Configuring ActiveX Controls

**Methods Tab**

A method is a function that performs a specific action such as printing and is executed using the `CALL-EXTERNAL-FUNCTION` verb.

**Properties: Events Tab**

Show on properties sheet

Specifies whether the selected ActiveX event is displayed in the screen editor’s logics tab on the properties sheet.

- If checked, the event is displayed in the logics tab.
- If not checked, the event is not displayed in the logics tab, thus a guide developer cannot attach logic flows to this event.

The default is checked.

Show on properties sheet

Specifies whether the selected ActiveX event is displayed in the screen editor’s logics tab on the properties sheet.

- If checked, the event is displayed in the logics tab.
- If not checked, the event is not displayed in the logics tab, thus a guide developer cannot attach logic flows to this event.

The default is checked.
The Methods tab displays the ActiveX control’s methods. For more information about the arguments, see your ActiveX control documentation.

**Properties: Methods Tab**

**EDGE name**

This property is used to rename a method and is used by the `CALL-EXTERNAL-FUNCTION` verb. By default, the name is the same as the ActiveX method selected.

The name can contain alphanumeric characters, periods, and underscores, but must start with an alphabetical character.
An ActiveX control can have methods with the same name but with a different parameter list. When EDGE reads in the control’s information into this page, the methods with the same names will be suffixed with a number. (The first method in the list will not be suffixed with a number). For example, if there are three methods called Print, the EDGE names by default will be Print, Print1, and Print2. Because these names may not provide enough description, the guide developer has been given the option to rename the methods.

Show in verb properties sheet

Specifies whether the ActiveX method is available in the properties sheet for the CALL-EXTERNAL-FUNCTION verb.

If checked, the method is displayed in the properties sheet.

The default is checked.
Wrappers

Wrappers are ActiveX controls that call other ActiveX controls. EDGE provides wrappers for several ActiveX controls to integrate them into EDGE. The following wrappers are provided:

- EDGE ListView Control.
- EDGE TreeView Control.

To be used in EDGE, the wrappers must be configured like any other ActiveX control as described in the section Configuring ActiveX Controls.

In addition to the wrappers EDGE provides, you can write your own wrapper for any control. For information, see the section Creating Wrappers.

This section describes brief descriptions of the properties, events, and methods in the wrappers.
Syntax Convention

Many of the control properties require you to specify items separated by value marks and subvalue marks. The following notation is used to represent these marks:

[VM] Stands for the value mark which is represented by the ASCII 29 character.

[subVM] Stands for the subvalue mark which is represented by the ASCII 28 character.

EDGE ListView Control

The EDGE ListView control displays items using one of the following views:

- Large or standard icons.
- Small icons.
- List.
- Report.

You can arrange items into columns with or without column headings as well as display icons and text. The following is an example of a list view in a report format:
The EDGE ListView control exposes the following properties:

**Properties: EDGE ListView**

- **Arrange**
  Arrangement of icons when the View property is Icon or SmallIcon; can be one of the following:
  - 0-NoArrange: Items are not arranged.
  - 1-AutoLeft: Items are aligned along the left side of the control.
  - 2-AutoTop: Items are aligned along the top of the control.

The EDGE data type can be one of the following:

- Enumeration with list (default)
- Integer
- EDGE field or variable
Properties: EDGE ListView (continued)

BackColor  The background color for the cells; can be selected from the three- button or specified as a hexadecimal number. For more information, see the section Specifying Color.

The EDGE data type can be one of the following:

- Color (default)
- EDGE field or variable
- Integer

BorderStyle  The border style; can be one of the following:

- 0-NoBorder  No border.
- 1-FixedSingle  A single line border around the control.

The default is 1-FixedSingle.

The EDGE data type can be one of the following:

- Enumeration with list (default)
- Integer
- EDGE field or variable

Destination Field  Returns the value of the currently selected item.

The EDGE data type can be one of the following:
Properties: EDGE ListView (continued)

Text (default)
Text with formatting dialog
EDGE field or variable

Caution! To store the value of the item, you must specify the data type as EDGE field or variable.

Enabled
Specifies whether the object can respond to user-generated events or is only displayed.

The EDGE data type can be one of the following:

Yes or No list (default)
True or False list
EDGE field or variable

ForeColor
The foreground color used to display text and graphics in an object; can be specified from the three-button or entered as a hexadecimal number. For more information, see the section Specifying Color.

The EDGE data type can be one of the following:

Color (default)
EDGE field or variable
Integer
Properties: EDGE ListView  (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hide-Column-Headers</td>
<td>Specifies whether to hide the column headers specified in the ListCols property when in Report View.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>IconPaths</td>
<td>The paths to the icons, using the following format:</td>
</tr>
<tr>
<td></td>
<td>path1[VM]path2[VM]path3...</td>
</tr>
<tr>
<td></td>
<td>The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes or No list (default)</td>
</tr>
<tr>
<td></td>
<td>True or False list</td>
</tr>
<tr>
<td></td>
<td>EDGE field or variable</td>
</tr>
<tr>
<td>LabelEdit</td>
<td>Specifies whether a user can edit the label column, which is the first column to the right of the icon; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>0-Automatic  Editing is initiated when a selected object is clicked. The BeforeLabelEdit event is generated when the user clicks the label of a selected node.</td>
</tr>
<tr>
<td></td>
<td>1-Manual  Does not allow editing</td>
</tr>
<tr>
<td></td>
<td>The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Text (default)</td>
</tr>
<tr>
<td></td>
<td>Text with formatting dialog</td>
</tr>
<tr>
<td></td>
<td>EDGE field or variable</td>
</tr>
</tbody>
</table>
### Properties: EDGE ListView (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LabelWrap</td>
<td>Specifies whether labels are wrapped when the View is in icon view. The length of the label is determined by setting the icon spacing in the Control Panel, Desktop option, in Windows NT 4.0. In Windows 95/98, use the Appearance tab in the Display control panel. The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Yes or No list (default)</td>
</tr>
<tr>
<td></td>
<td>True or False list</td>
</tr>
<tr>
<td></td>
<td>EDGE field or variable</td>
</tr>
<tr>
<td>ListCols</td>
<td>The names of column headers; used only if View is set to 3-Reports; using the following format: column1[VM]column2[VM] ...</td>
</tr>
<tr>
<td>ListData</td>
<td>The items (list entries) to be included in the ListView control; using the following format:</td>
</tr>
</tbody>
</table>
**Properties: EDGE ListView (continued)**

item1[subVM]subitem1[subVM]subitemn2[subVM] ... subitemn-n

item2[subVM]subitem1[subVM]subitemn2[subVM] ... subitemn-n

...

itemn[subVM]subitem1[subVM]subitem2[subVM] ... subitemn-n

The items are also referred to as labels. The subitems are only displayed when the View property is set to Report.

The EDGE data type can be one of the following:

- **Text** (default)
- Text with formatting dialog
- EDGE field or variable

**MultiSelect**

Specifies whether a user can select multiple items in the control; can be one of the following:

The EDGE data type can be one of the following:

- **Yes or No list** (default)
- True or False list
- EDGE field or variable

**SortKey**

Determines how the items are sorted; can be one of the following:

- 0 Sort using the object’s Text property.
Properties: EDGE ListView (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sort using the subitem whose collection is specified here. The Sorted property must be set to true before the items are sorted. The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Integer (default)</td>
</tr>
<tr>
<td></td>
<td>EDGE field or variable</td>
</tr>
<tr>
<td>SortOrder</td>
<td>The order in which to sort items; can be one of the following: 0-Ascending. 1-Descending. The Sorted property must be set to True before a list will be sorted. The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Enumeration with list (default)</td>
</tr>
<tr>
<td></td>
<td>Integer</td>
</tr>
<tr>
<td></td>
<td>EDGE field or variable</td>
</tr>
<tr>
<td>Sorted</td>
<td>Specifies whether the objects are sorted. The EDGE data type can be one of the following:</td>
</tr>
</tbody>
</table>
**Properties: EDGE ListView (continued)**

- **Yes or No list** (default)
- **True or False list**
- **EDGE field or variable**

**View**

The type of listview object to be created; can be one of the following:

- **0-Icon**
  
  Displays a full-sized icon and a text label below the icon. The `Arrange` property affects the display of the icons. In EDGE Client, the icons can be moved by the user.

- **1-SmallIcon**
  
  Displays a small icon and a text label to the right of the icon. The `Arrange` property affects the display of the icons. In EDGE Client, the icons can be moved by the user. The items are displayed horizontally. The following is an example:

- **2-List**
  
  Displays a small icon and a text label to the right of the icon. The items are displayed vertically.

- **3-Report**
  
  Displays the icons, text labels, and information in columns with the leftmost column containing the small icon, followed by the text label, followed by additional columns with subitems.

The default is 0-Icon.
Properties: EDGE ListView (continued)

The EDGE data type can be one of the following:

- Enumeration with list (default).
- Integer.
- EDGE field or variable.

EDGE ListView Control Events

The ListView control displays the following events which are inherited from the Microsoft ListView control. It adds no additional events.

Properties: EDGE ListView Control Events

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AfterLabelEdit</td>
<td>Occurs after a user edits, makes a change to a label, then either clicks another node or ListItem object or presses the ENTER key.</td>
</tr>
<tr>
<td>BeforeLabelEdit</td>
<td>Occurs after a user clicks the label to select it, and clicks a second time to begin editing. This event occurs after the second click.</td>
</tr>
<tr>
<td>Click</td>
<td>Occurs when a user presses then releases a mouse button over an object.</td>
</tr>
<tr>
<td>ColumnClick</td>
<td>Occurs when a ColumnHeader is clicked. Only available in Report view.</td>
</tr>
</tbody>
</table>
**Properties: EDGE ListView Control Events (continued)**

- **Db1Click**: Occurs when a user presses and releases a mouse button and then presses and releases it again over an object, quickly.
  
  If there is code in the `Click` event, the `Db1Click` event will never occur.

- **ItemClick**: Occurs when the mouse is clicked on the text or image of a ListItem object.

- **KeyDown**: Occurs when a user presses a key while an object has the focus.

- **KeyPress**: Occurs when a user releases a key.

- **KeyUp**: Occurs when a user releases a key while an object has the focus.

- **MouseDown**: Occurs when a user presses a mouse button.

- **MouseUp**: Occurs when a user releases a mouse button.

**EDGE ListView Control Methods**

The EDGE ListView control has the following method:

- **GetSelectedItems**: Returns the index of the selected item, which is the number of the row selected. The first row is row 1.
**EDGE TreeView Control**

The EDGE TreeView control displays a hierarchical list of node objects, each of which consists of a label and an optional bitmap. A treeview is typically used to display the hierarchical information such as in Windows Explorer.

The following is an example of a tree view:

![Tree View Example](image)

The TreeView control exposes the following properties:

**Properties: EDGE TreeView**

- **BorderStyle**
  
  The border style; can be one of the following:
  
  0-NoBorder  
  No border.
  
  1-FixedSingle  
  A single line border around the control.

  The default is 0-NoBorder.

  The EDGE data type can be one of the following:
### Properties: EDGE TreeView (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>Data for the control with the following syntax:</td>
</tr>
<tr>
<td></td>
<td>- root[subVM]</td>
</tr>
<tr>
<td></td>
<td>- parent[subVM]</td>
</tr>
<tr>
<td></td>
<td>- child[subVM] ... [VM]</td>
</tr>
<tr>
<td></td>
<td>- root[subVM]</td>
</tr>
<tr>
<td></td>
<td>- parent[subVM]</td>
</tr>
<tr>
<td></td>
<td>- child[subVM] ... [VM]</td>
</tr>
<tr>
<td></td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>- Text (default)</td>
</tr>
<tr>
<td></td>
<td>- Text with formatting dialog</td>
</tr>
<tr>
<td></td>
<td>- EDGE field or variable</td>
</tr>
<tr>
<td>Destination-Field</td>
<td>Returns the value of the current selected item which includes the values of the root and parent nodes. Uses the following syntax:</td>
</tr>
<tr>
<td></td>
<td>- rootValue[subVM] parentValue [subVM] childValue [subVM] ...</td>
</tr>
<tr>
<td></td>
<td>The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>- Text (default)</td>
</tr>
<tr>
<td></td>
<td>- Text with formatting dialog</td>
</tr>
<tr>
<td></td>
<td>- EDGE field or variable</td>
</tr>
</tbody>
</table>
### Properties: EDGE TreeView (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Text</strong></td>
<td>(default)</td>
</tr>
<tr>
<td><strong>Text with formatting dialog</strong></td>
<td></td>
</tr>
<tr>
<td><strong>EDGE field or variable</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Caution!** To store the value of the item, you must specify the data type as **EDGE field or variable**.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Enabled</strong></td>
<td>Specifies whether the object can respond to user-generated events or is only displayed.</td>
</tr>
<tr>
<td></td>
<td>The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Yes or No list</strong> (default)</td>
</tr>
<tr>
<td></td>
<td><strong>True or False list</strong></td>
</tr>
<tr>
<td></td>
<td><strong>EDGE field or variable</strong></td>
</tr>
<tr>
<td><strong>Indentation</strong></td>
<td>The width of the indentation in twips, which means each new child node is indented by this amount.</td>
</tr>
<tr>
<td></td>
<td>The EDGE data type can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Real</strong> (default)</td>
</tr>
<tr>
<td></td>
<td><strong>EDGE field or variable</strong></td>
</tr>
<tr>
<td><strong>LabelEdit</strong></td>
<td>Specifies whether a user can edit labels; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>0-Automatic</strong> Editing is initiated when a selected object is clicked.</td>
</tr>
<tr>
<td></td>
<td><strong>1-Manual</strong> Cannot be edited.</td>
</tr>
</tbody>
</table>
**Properties: EDGE TreeView (continued)**

- **LineStyle**
  - Specifies the style of lines used between node objects; can be one of the following:
    - **0-TreeLines**
      - Displays lines between node siblings and their parent node.
    - **1-RootLines**
      - Displays lines between node siblings and their parent node, and also displays lines between the root nodes.
  - The default is 0-TreeLines.

- **Sorted**
  - Specifies whether the objects are sorted.
  - The default is 0-Automatic.

The EDGE data type can be one of the following:

- **Enumeration with list** (default).
- **Integer**.
- **EDGE field or variable**.

The EDGE data type can be one of the following:

- **Enumeration with list** (default).
- **Integer**.
- **EDGE field or variable**.
### Properties: EDGE TreeView (continued)

<table>
<thead>
<tr>
<th>Style</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes or No list</td>
<td>(default).</td>
</tr>
<tr>
<td>True or False list.</td>
<td></td>
</tr>
<tr>
<td>EDGE field or variable.</td>
<td></td>
</tr>
<tr>
<td><strong>Style</strong></td>
<td>The type of graphics (plus/minus signs and lines) and text that appear for each node; can be one of the following:</td>
</tr>
<tr>
<td>0-TextOnly</td>
<td>Displays only text.</td>
</tr>
<tr>
<td>2-PlusMinusText</td>
<td>Displays plus/minus signs and text.</td>
</tr>
<tr>
<td>4-TreeLinesText;</td>
<td>Displays lines and text.</td>
</tr>
<tr>
<td>6-TreeLinesPlusMinusText</td>
<td>Displays lines, plus/minus signs, and text.</td>
</tr>
</tbody>
</table>

The style types for 1, 3, 5, and 7 include images and are not supported. If one of these styles is specified, the style defaults to a similar style that does not include images. For example, 1-ImageText defaults to 0-TextOnly.
### EDGE TreeView Control Events

The EDGE TreeView control has the following events:

**Properties: EDGE TreeView Control Events**

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AfterLabelEdit</td>
<td>Occurs after a user makes a change to a label, then either clicks another node or ListItem object or presses the ENTER key.</td>
</tr>
<tr>
<td>BeforeLabelEdit</td>
<td>Occurs after a user clicks the object to select it, and clicks a second time to begin editing. This event occurs after the second click.</td>
</tr>
<tr>
<td>Click</td>
<td>Occurs when a user presses then releases a mouse button over an object.</td>
</tr>
<tr>
<td>Collapse</td>
<td>Occurs when a node object is collapsed by pressing the minus sign.</td>
</tr>
<tr>
<td>DblClick</td>
<td>Occurs when a user presses and releases a mouse button then quickly presses and releases it again. If there is code in the Click event, the DblClick event will never occur.</td>
</tr>
<tr>
<td>Expand</td>
<td>Occurs when a node object is expanded by pressing the plus sign. Expanding a node object makes its child nodes visible.</td>
</tr>
<tr>
<td>KeyDown</td>
<td>Occurs when a user presses or releases a key while an object has the focus.</td>
</tr>
<tr>
<td>KeyPress</td>
<td>Occurs when a user presses a key.</td>
</tr>
</tbody>
</table>
### Properties: EDGE TreeView Control Events (continued)

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KeyUp</td>
<td>Occurs when a user releases a key while an object has the focus.</td>
</tr>
<tr>
<td>MouseDown</td>
<td>Occurs when a user presses a mouse button.</td>
</tr>
<tr>
<td>MouseUp</td>
<td>Occurs when a user releases a mouse button.</td>
</tr>
<tr>
<td>NodeClick</td>
<td>Occurs when a node object is clicked. The NodeClick event occurs before the Click event.</td>
</tr>
</tbody>
</table>

### EDGE TreeView Control Methods

The EDGE TreeView control does not expose any methods and there are no additional methods added.
Creating Wrappers

Many programming languages and applications can be used to create ActiveX wrappers. The EDGE wrappers discussed in this chapter were written using Microsoft Visual Basic 5. The source code for these wrappers is provided with EDGE Developer. Sample VB5 source code is included with the installation and is installed in the subdirectory Wrappers.

To create a wrapper in Visual Basic 5, do the following:

1. Download Visual Basic 5 Control Creation Edition from the Microsoft site by selecting the following:
   www.microsoft.com/vbasic/controls
   You can also download help files and documentation from this site.

2. Open the MS Visual Basic program and double-click the ActiveX Control icon to open the design program. This automatically opens a new project and adds a UserControl designer to the project.

3. To open the ActiveX Control Interface Wizard feature, select the following:
   Add-Ins menu / Add-In Manager

4. Create the control.
5. If the control is to be used for EDGE Client, you may want to create a cabinet (.cab) file for the activeX control. A .cab file contains all of the information necessary to download, install, and register the components required to run the control.

To create a .cab file from the Application Setup Wizard, select the following:

Start / Programs / Visual Basic n.n CCE /

6. To compile a control into an .ocx file, do the following:

a. On the File menu, click Make Project.ocx and click OK to build the .ocx file.

b. On the File menu, click Remove Project to remove the Project from the project group, so that Visual Basic will use the compiled component (.ocx file) instead of the project.

c. Save the file.
Adding ActiveX Objects to Screens

ActiveX objects are added to screens through the ActiveX toolbox, which contains the icons of all the configured controls. ActiveX objects are added to a screen like any other screen object. For more information on creating screen objects, see Chapter 20.

When you add an ActiveX object and the object does not have a user interface to a screen (that is, it only provides a service), the control is shown as an outline; otherwise, the ActiveX control’s user interface is displayed. Examples of ActiveX objects without a user interface to a screen are a timer or email service.

ActiveX Toolbox

The ActiveX toolbox displays the icons for the ActiveX controls that have been configured and exist on your machine. Up to 50 configured ActiveX controls can be displayed. If more are configured, they are not displayed.

To display the ActiveX toolbox, select ActiveX Toolbox from the Screen menu. A toolbox similar to the following is displayed:
If you move your cursor over an icon, the EDGE name of the object is displayed in a tooltip.

**Properties Sheet**

The properties sheet for an ActiveX control has the following tabs:

- Main Tab.
- ActiveX Tab.
- Logics Tab.

**Main Tab**

The Main tab displays the properties that are common among the ActiveX objects. A properties sheet similar to the following is displayed:
ActiveX Controls

Adding ActiveX Objects to Screens

Properties:

**ActiveX name**  The EDGE name of the ActiveX control. This is the same name you specified in the ActiveX configuration tab page. For more information, see Configuring ActiveX Controls.

This is display only.

**(Custom)**  The Custom property has a Modify command button that displays a page with additional properties that are not listed in the Main tab. These properties are specific to the ActiveX control.
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Height</td>
<td>The height of the ActiveX control. This property can be modified by the SET-PROPERTY verb. If the ActiveX object has a predetermined height, your setting for this property is ignored.</td>
</tr>
<tr>
<td>Left</td>
<td>The first screen column location occupied by the object on the screen. When the metric is Row/Col, the first left column is 1. For all other metrics, it is 0 (zero). This property can be modified by the SET-PROPERTY verb.</td>
</tr>
<tr>
<td>Name</td>
<td>The object name in the screen editor. By default, the name is ActiveXn. For example, ActiveX1. This property cannot be modified by the SET-PROPERTY verb.</td>
</tr>
<tr>
<td>Top</td>
<td>The top position of the object on the screen. When the metric is Row/Col, the first top position is 1. For all other metrics, it is 0 (zero). This property can be modified by the SET-PROPERTY verb.</td>
</tr>
<tr>
<td>Type</td>
<td>The object’s type in the screen editor. For ActiveX objects, the type is ActiveX. This is display only.</td>
</tr>
</tbody>
</table>
**ActiveX Controls**

**ActiveX Tab**

The ActiveX tab displays the properties that were specified to be displayed when the control was configured. (For more information, see **Properties Tab** in the Configuring ActiveX Controls section.)

Each ActiveX object has a different set of properties. The ActiveX tab for a calendar control is similar to the following:

<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Visible</strong></td>
</tr>
<tr>
<td><strong>Width</strong></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
If the value specified for a property is an EDGE field or variable, you see the settings reflected at runtime; otherwise, the setting is reflected immediately on the ActiveX object.

**Logics Tab**

The Logics tab is used to specify the following:

- A logic flow to be executed when an event occurs.
• An EDGE multivalued field to store an ActiveX control’s event parameter list. For more information about an event’s parameter list, see your ActiveX control documentation.

The Logics tab for a calendar control is similar to the following:
Each event has an associated event parameter used to store the parameter list based on an event.
Each time an event occurs, all of its event parameters are stored in this multivalued field and then the logic is called. You can use the multivalued field in your logic flows. See your ActiveX control documentation for the description and order of the arguments in the list.
Specifying ActiveX Property Values

The EDGE data type specified for the ActiveX property determines the way the property value can be specified. For example, if *Integer* is specified, you must specify an integer value for the property.

In logic flows, you can determine the current value of ActiveX properties by using the *GET-PROPERTY* verb. You can set the properties using the *SET-PROPERTY* verb.

The following is discussed:

- Specifying Color.
- Specifying Enumeration Items.
- Specifying Yes/No or True/False.
- Specifying Data.

**Specifying Color**

An ActiveX property that expects a color value can be defined as one of the following data types:

- Color.
- Integer.
- Field or Variable.
**Color**

When the data type for the property is specified as Color, the color is specified as a padded hexadecimal value, which represents the color based on the BGR (blue, green, red) values for the color. For example, when adding to a screen, the property displays a sample of the selected color, a hexadecimal number representing the color, and a three-dot button, similar to the following:

![Color property example](image)

When the three-dot button is clicked, a color chart similar to the following is displayed:
You can select the color directly from this chart, or define a custom color. If you already know the decimal values of the color you can enter the values in hexadecimal format directly into the property. For information on determining the hexadecimal value, see the next section.

**Determining the Hexadecimal Value**

The BGR value is a set of numbers based on the values of blue, green, and red, where each value is a number between 0 and 255. For exam-
ple, the color blue in BGR is 255, 0,0. To determine the values of a color, you can use the color chart. (If the values are in RGB (red, green, blue) format, they must be transposed to BGR. For example, blue in RGB is 0, 0, 255; in BGR, blue is 255, 0, 0.)

The hexadecimal number is derived by changing the decimal value for each color to a hexadecimal value. For example, Figure 21-1 shows how the color blue is converted. (To change a decimal value to a hexadecimal value, use the Windows Calculator feature.)

\[
\begin{array}{ccc}
\text{blue} & \text{green} & \text{red} \\
255 & 0 & 0 \quad \text{Decimal values} \\
\text{ff} & 00 & 00 \quad \text{Hexadecimal values} \\
0x00ff0000 \quad \text{Padded hexadecimal value}
\end{array}
\]

\textit{Figure 21-1. Deriving the Hexadecimal Value}

\textbf{Integer}

If the EDGE data type for the ActiveX object is specified as \textit{Integer}, the screen property is similar to the following:
When the type is Integer, the color must be entered as a decimal value. The decimal number is based on the BGR (blue, green, red) values for the color converted from hexadecimal to decimal.

To derive a decimal value, do one of the following:

- Use the Calculator program to change the hexadecimal value to decimal. For example, the color blue in hexadecimal is ff0000; which converts to the decimal number 16711680.

- Use the following formula with the numbers for the BGR values:

  \[(\text{blue} \times 65536) + (\text{green} \times 256) + \text{red}\].

  For example, the color blue is 255,0,0, which converts to 16711680.

**Field or Variable**

If the EDGE data type for the ActiveX object is specified as *Field or variable*, the screen property is similar to the following:

<table>
<thead>
<tr>
<th>Property name</th>
<th>Decimal value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ForeColor</td>
<td>16711680</td>
</tr>
</tbody>
</table>
To specify a color, the field or variable specified for this property must contain the decimal value for the color. (To derive a decimal value, see the section **Integer**.)

If the EDGE data type for the ActiveX object is specified as **Enumeration with list**, the screen property is similar to the following:
To specify an item from the screen property, you select the item from the list. If the EDGE data type is specified as Field or Variable, the number of the item must be contained in the field or variable. For example, for BorderStyle the number 0 specifies no border.

If the EDGE data type for the ActiveX object is specified as Yes/No list or True/False list the screen property is similar to the following:

```
Property name   True/False list
```

To specify from the screen property, you select either yes/true or no/false by pressing the down arrow. If the EDGE data type is specified as Field or Variable, one of the following values must be contained in the field or variable:

- "Y" or "T"; for yes or true.
- "N" or "F"; for no or false.

The values are literals that must be enclosed in quotation marks.
**Specifying Data**

An ActiveX property that expects data can be specified in the following ways, depending on the EDGE data type used when the control is configured:

- **Text.**
- **Text with Formatting Dialog.**
- **Field or Variable.**

**Text**

If the EDGE data type for the ActiveX object is specified as **Text**, the screen property is similar to the following:

<table>
<thead>
<tr>
<th>Property name</th>
<th>Data can be entered</th>
</tr>
</thead>
<tbody>
<tr>
<td>IconPaths</td>
<td></td>
</tr>
</tbody>
</table>

The text data type expects a literal which is enclosed in quotation marks.

**Text with Formatting Dialog**

If the EDGE data type for the ActiveX object is specified as **Text with formatting dialog**, the screen property is similar to the following:
When you click the three- button, a dialog box similar to the following is displayed:

![Dialog Box Image]

When you add items to the list, each item is separated by a value mark. You cannot use this dialog box when the data requires sub-value marks.
**Using with EDGE Client**

EDGE Client supports any ActiveX control that has been configured and added to a screen from EDGE Developer. When a screen with an ActiveX control is displayed in EDGE Client, EDGE Client verifies the control exists on the client.

If the control is not found on the client, EDGE Client attempts to download the control from the web server. The location of the control is specified in the Code location (URL). If this location is specified, EDGE Client will download the control, install it on to the client, and then execute it. If the location is not specified, the agent will not be able to use the control.
Overview

Toolbars consist of icons that provide a shortcut to execute an action. For example, to open a file you typically select Open from the File menu. However, if there is an icon on a toolbar to open a file you can click the icon instead of selecting from the menu.

EDGE Client provides a standard toolbar, but you can also define your own toolbars. There is no limit on the number of toolbars you can add, nor the number of icons on a toolbar.

Toolbars can be positioned below each other, next to each other, along the sides, at the bottom of the EDGE Client window, or they can be made to float, in which case they may even be outside the EDGE Client window. Toolbars can be undocked and resized.

These user-definable toolbars are created in EDGE Developer at the project level and can be previewed in an EDGE Client preview window.
For an example of an EDGE Client preview window with user-defined toolbars, see Figure 22-1.

**Figure 22-1. Toolbars in EDGE Client**

More than one toolbar can be active at a time. EDGE Client users can control which toolbars are displayed during the current call.
The user-defined toolbars are not displayed between calls; that is, when the agent is in the Call Center, Project Selection, Auto-Receive, or Auto-Dial screen the toolbars are not visible. When a project is started, the toolbars are restored to their defined states, not the state in which user may have put them.
Creating and Modifying Toolbars

Toolbars are created and modified from the Toolbars tabbed page in the Guide area. To display the Toolbars page, select the following:

Project / Guide / Toolbars

A page similar to the following is displayed:

![Toolbar Editor](image)

When you add or modify a toolbar, the toolbar editor is invoked and a page similar to the following is displayed:
Figure 22-2. Toolbar Editor Area

The Toolbar property displays the name of the toolbar. The name can contain any alphanumeric character as well as periods, but must start with an alphabetical character and cannot exceed ten characters. You must specify a name before you can add any icons to the toolbar. (For information about adding icons to the toolbar, see the section Creating and Modifying Icons.)

When your cursor is in an empty area of the toolbar bitmap area, a tooltip is displayed providing information about adding and deleting toolbar icons.

When your cursor is over a toolbar icon, a tooltip is displayed providing information about adding and removing separators.
If this is a new toolbar, that is, it has not been saved yet, select the toolbar name or characters in the name, then click the right mouse button in the toolbar name area and the following shortcut menu is displayed:

```
Undo
Cut
Copy
Paste
Delete
Select All
```

**Disable/Enable Toolbar**

The toolbar itself cannot be disabled by the guide developer, although you can hide it. You can disable individual toolbar icons using the icon property `Enabled`.

When an active window is created with the WINDOW verb using the Create dialog box property, toolbars are disabled.
Toolbar Properties

To display the toolbar property page, move the cursor to an empty area in the bitmap area, then do one of the following:

- Select Show Properties in the Toolbars menu.
- Click the right mouse button and select Show Properties from the shortcut menu.
- Press F4.

A page similar to the following is displayed:
Properties:

Allow docking: Specifies whether the toolbar is initially docked. To specify the toolbar’s initial location, enter pixel values for the Left and Top properties.

When a toolbar is docked, it becomes attached to the top (that is, the Standard toolbar area), sides, or bottom of the EDGE Client window at run-time; can be one of the following:
**Properties:**

- **False**  
  The toolbar is not initially docked. The toolbar is floating and can be moved anywhere.  
  Selecting false also disables the Location and Order properties.

- **True**  
  The toolbar is initially docked.  
  This is the default.

- **Allow resize**  
  The toolbar can be resized when it is floating; can be one of the following:

  - **False**  
    The toolbar cannot be resized.

  - **True**  
    The toolbar can be resized. This is the default.

  To resize a toolbar, move your cursor to any side of the toolbar until your cursor becomes a double arrow. Then click the left mouse button to resize it. Resizing a toolbar may also change its orientation.

- **Left**  
  The initial location of a floating toolbar. This value is in pixels; and is relative to the upper left corner of the EDGE Client screen. This property is available only if the Location property is Floating.

  This property can be modified by the SET-PROPERTY verb.

- **Location**  
  The toolbar location on the EDGE Client window. For an example, see the section Example of Positions.

  The following are the Location values:
Properties:

**Bottom**  The toolbar is docked horizontally at the bottom of the EDGE Client window.

**Floating**  The toolbar can be positioned anywhere on the screen; therefore, it is not docked. For example, it can be in the middle of the EDGE Client window.

When a toolbar is floating, a title bar is displayed at the top of the toolbar.

You must also specify the Left and Top properties.

**Left**  The toolbar is docked vertically on the left side of the EDGE Client window.

**Right**  The toolbar is docked vertically on the right side of the EDGE Client window.

**Top**  The toolbar is docked horizontally on the top of the EDGE Client window.

Depending on where you place your toolbars, they may lay on top of some of your EDGE Client windows.

**Order**  The order in which to display the docked toolbar when there is more than one toolbar. This property is used when toolbars have the same Location and Rank.

For toolbars docked at the top or bottom, the smaller the Order, the closer the toolbar is to the left edge of EDGE Client.
**Properties:**

For toolbars docked at the right or left, the smaller the Order, the closer the toolbar is to the top of EDGE Client.

For an example, see the section Example of Positions.

**Rank**

The distance between the toolbar and the edge of the window on which it is docked. The smaller the rank, the closer the toolbar is to the edge.

For toolbars docked to the top, a rank of zero indicates that the toolbar should be positioned on the same line as EDGE Client’s standard toolbar.

If the rank is greater than zero, the toolbar is docked below EDGE Client’s main toolbar.

This property is not available when the Location property is Floating.

For an example, see the section Example of Positions.

**Save location**

Specifies whether the toolbar bitmap is saved on the client or on the EDGE server; can be one of the following:

- **Local**
  - The toolbar bitmap is saved on the client in the location specified in the Toolbar path property.

- **Server**
  - The toolbar bitmap is saved in a file on the server.
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title</strong></td>
<td>The title of the toolbar window; this is used when the toolbar is floating. If the title is longer than the toolbar, the title is truncated.</td>
</tr>
<tr>
<td></td>
<td>The title is also used to represent the toolbar in EDGE Client’s list of toolbars. For more information, see your EDGE 7.11 Client Reference Manual.</td>
</tr>
<tr>
<td><strong>Toolbar path</strong></td>
<td>The path and filename of the toolbar bitmap image where it is to be saved. This path is required before the toolbar is saved.</td>
</tr>
<tr>
<td></td>
<td>If EDGE Developer or EDGE Client fails to load the bitmap image of a toolbar icon, the toolbar is not created.</td>
</tr>
<tr>
<td></td>
<td>This property is only available if the Save location property is set to Local.</td>
</tr>
<tr>
<td></td>
<td>If you enter a filename that already exists, a message is displayed that the file will be overwritten. To overwrite the file, select OK.</td>
</tr>
<tr>
<td><strong>Top</strong></td>
<td>The initial location of a floating toolbar. This value is in pixels; and is relative to the upper left corner of the EDGE Client window.</td>
</tr>
<tr>
<td></td>
<td>This property is available when the Location property is Floating.</td>
</tr>
<tr>
<td></td>
<td>This property can be modified by the <code>SET-PROPERTY</code> verb.</td>
</tr>
</tbody>
</table>
**Properties:**

Visible

- **False** Toolbar is hidden initially. The agent can make any toolbar visible in EDGE Client. Thus the developer cannot prevent an agent from accessing a toolbar. For more information, see your EDGE 7.11 Client Reference Manual.

- **True** Toolbar is visible. This is the default.

This property can be modified by the `SET-PROPERTY` verb.

---

**Example of Positions**

*Figure 22-3* shows examples of toolbar positions. Assume four toolbars are defined in EDGE Developer with the following positions:

<table>
<thead>
<tr>
<th>Toolbar</th>
<th>Location Property</th>
<th>Order Property</th>
<th>Rank Property</th>
<th>Left Property</th>
<th>Top Property</th>
</tr>
</thead>
<tbody>
<tr>
<td>Toolbar A</td>
<td>Top</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Toolbar B</td>
<td>Top</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Toolbar C</td>
<td>Top</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Toolbar D</td>
<td>Floating</td>
<td>100</td>
<td>150</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**Toolbars Menu**

The Toolbars menu is similar to the following:

**Figure 22-3. Multiple Toolbars Display in Docked and Floating Positions**
The menu contains the following commands:

- Add button. See the section Add a New Icon.
- Delete button. See the section Delete an Icon.
- Insert separator. See the section Add a Separator.
- Remove separator. See the section Remove a Separator.
- Show properties. See the section Show Properties.
- Toolbar preview. For more information, see the section Preview Toolbars.
Creating and Modifying Icons

You can add, delete, or modify a toolbar icon through the Toolbar Editor. A toolbar must contain icons before it can be saved.

Commands can be selected in the following ways:

- Select the command in the Toolbars menu.
- Move your cursor into the toolbar bitmap area. Click the right mouse button and select the command from the shortcut menu.
- Select the appropriate tool from the toolbox.

For information on icon properties, see the section Toolbar Icon Properties.

Add a New Icon

When you select the Add command, a browse window is opened, listing .bmp and .ico files. Change folders as necessary and select the filename with the icon to be added.

You can add up to 16 icons without separators on a toolbar before the scroll bar appears.
### Add a Separator
A separator is eight pixels of space. Before you can select the command, you must select an icon. The separator is added to the left of the selected icon.

### Delete an Icon
Before you can select the delete command, you must select an icon. After an icon is deleted, you cannot use Undo to bring it back.

### Remove a Separator
To delete a separator, select the icon after the separator to be removed, then select the command.

### Show Properties
To show the properties for an icon on the toolbar, select an icon before selecting the command.
**Toolbar Icon Properties**

The toolbar icons property page is similar to the following:

**Properties:**

- **Bitmap path**: The location of the bitmap or icon to use as the image on the icon. This path is required.
Properties:

A library of Toolbar icons is included with EDGE Developer. You can also create your own toolbar bitmap image file, and specify its location here.

The dimensions of the image for the icon that can be displayed cannot be greater than 16 x 15 pixels. If you specify an icon larger than these dimensions, the icon is truncated.

If you delete the bitmap after the toolbar image is created, the icon image is saved, but you will not be able to modify it.

<table>
<thead>
<tr>
<th>Enabled</th>
<th>Specifies whether the icon is enabled when the toolbar is loaded in EDGE Client.</th>
</tr>
</thead>
<tbody>
<tr>
<td>False</td>
<td>The icon is disabled. When an icon is disabled, it is still visible on the EDGE Client screen, but it is grayed and cannot be selected.</td>
</tr>
<tr>
<td>True</td>
<td>The icon is enabled. This is the default.</td>
</tr>
</tbody>
</table>

This property can be modified by the SET-PROPERTY verb.

<table>
<thead>
<tr>
<th>Help ID</th>
<th>The context sensitive help identifier.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Key-strokes</th>
<th>The keystrokes to simulate on the client when this icon is clicked. The syntax of the keystrokes are the same as EDGE Client’s DDE SendKeys server command.</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Logic</th>
<th>The logic on the server to execute when the icon is clicked.</th>
</tr>
</thead>
</table>
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Name</strong></td>
<td>The name of the icon. By default, it is Button(n); where (n) is a value automatically assigned. For example, the first icon is called Button1 by default.</td>
</tr>
<tr>
<td></td>
<td>This property is used the SET-PROPERTY verb to identify the object. To use the SET-PROPERTY verb with toolbar icons, you should uniquely identify the icons, even across toolbars, because the SET-PROPERTY verb operates on all icons of a given name.</td>
</tr>
<tr>
<td><strong>Result</strong></td>
<td>The EDGE result that should be set when the icon is clicked.</td>
</tr>
<tr>
<td><strong>Route to screen</strong></td>
<td>The screen to route to. When the three-dot button is selected, the Window Helper function is displayed in a second dialog box.</td>
</tr>
<tr>
<td><strong>Status bar</strong></td>
<td>The text displayed in the status bar when the cursor is moved over the toolbar icon.</td>
</tr>
<tr>
<td><strong>Tooltip</strong></td>
<td>The text displayed when the cursor is moved over the toolbar icon. The tooltip appears in a small text box next to the icon.</td>
</tr>
</tbody>
</table>
Preview Toolbars

Once a toolbar is created, you can preview it in the EDGE Client preview window. In the EDGE Client preview window, by default, all the toolbars that are visible are shown. The EDGE Client preview window only allows you to view toolbars; you cannot move or select any of them and you cannot execute any other EDGE Client functions.

The preview feature displays the saved versions of your toolbars. If you make changes to a toolbar, save the changes before previewing the toolbar.

To preview the toolbars, do one of the following:

- Select Toolbar Preview in the Toolbars menu.
- Move your cursor into the toolbar bitmap area. Click the right mouse button and select Toolbar Preview from the shortcut menu.
- Select the preview tool.

When you are in the sample EDGE Client window, and you click the right mouse button a shortcut menu similar to the following is displayed:
A sample of an EDGE Client preview window is displayed in Figure 22-1.
Overview

The EDGE Guide area is used to define processes that can be used for project level guide development. The EDGE Guide area has the following tabs:

- ActiveX Controls.
- General.
- Libraries/Functions.
- Multiple Project Assembly and Swap.
- Schemes.
- Styles.
- System Screens.
The General tab is similar to the following:

The following commands are available:

- **CBR Options.** This is used with the optional module CBR Link. For more information, see the CBR Link Reference Manual.

- **Project Operations.**
Project Selection Menu

The Project Operations command invokes EDGE Client, using the path set up in the Client Settings page.

The Project Selection Menu command allows you to specify the projects that are to be displayed in operations and the order in which they are displayed.

When you create a project, the project ID is added to the bottom of the list of projects in the system. If a project is not in this list, it cannot be displayed on any operator’s screen. (You can also restrict the projects that are displayed to an individual operator by not giving Operator Access to the project.)

When you select this command, a page similar to the following is displayed:
Add projects in the desired order.

In operations, the Project Selection screen shows the short project description from Project Options, if any; otherwise the project name is displayed.

**Separate XGUIDE Setup**

XGUIDE is the name of the EDGE executable program that resides in memory on the server whenever an EDGE guide is running. Generally all guides use the same XGUIDE. This provides very good performance—as agents switch between guides, the programs are already in memory and do not need to be loaded. However, having all guides use the same XGUIDE also requires more memory than having each guide use its own XGUIDE. On the other hand, having each guide use its own XGUIDE requires less memory, but more time for agents to go between guides.
Separate XGUIDES are most useful in situations where agents rarely switch guides and where memory on the server is limited. It is possible to configure separate XGUIDEs for some guides and to have all others use the general XGUIDE. For more help, contact EDGE Customer Care.

To set up separate XGUIDEs, be sure no agents are using the projects to be set up, then select the following:

EDGE / Guide area / General tab / Separate XGUIDE Setup

From the list of projects that are displayed, select those that are to have separate XGUIDEs. Each project that is highlighted is flagged and will have a separate XGUIDE created the next time the project is assembled and swapped. Once the flag is set or cleared for a project, agents will not be able to log into the project until it is assembled and swapped. From the agent’s point of view, once the project is swapped, there should be no difference between using a separate XGUIDE or the combined XGUIDE, except for the performance when first entering the guide.

To determine if a project has been assembled and swapped with a separate XGUIDE, go to the project being checked, then enter the following at ACL:

[INS,101]–> !ls bin

If the following files are displayed, the project has a separate XGUIDE.
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XGUIDE</td>
<td></td>
</tr>
<tr>
<td>XGUIDE.OP</td>
<td></td>
</tr>
<tr>
<td>XGUIDE1</td>
<td></td>
</tr>
<tr>
<td>XGUIDE2</td>
<td></td>
</tr>
</tbody>
</table>

**Version Bump**

This is used to bump the version. For more information, see the section [Version Control](#).

**Web Setup**

This is used to set up CyberEDGE projects. For more information, see the CyberEDGE documentation, which is available separately.
The Libraries/Functions tab is used to define the libraries and functions to be used by the CALL-EXTERNAL-FUNCTION verb. Before that verb can be used, the library filename, the function name, and the parameter return argument types must be specified. The library can be a shared library on the server or in a DLL on the client workstation.

The following can be defined as libraries:

- ActiveX Controls.
- Server Libraries.
- Client DLLs.

ActiveX Controls

ActiveX controls are software components produced by third party vendors and are available from many sources. Before they can be defined as a library, the controls must exist on the client PC and they must be configured in EDGE. All the methods for the control are available in the CALL-EXTERNAL-FUNCTION verb. For information on configuring these controls, see Chapter 21, ActiveX Controls.

Up to 16 parameters can be defined for each ActiveX control method.
Server Libraries

Libraries on the server can be either static (compiled with the program) or dynamic (linked at runtime).

The name of the library should begin with the letters lib. The library either should be placed in a directory where the assembly process will look for it (by default, EDGE looks first in /usr/lib, then in $EDGE-SYS), or the path to it must be specified as the External library path name for the library.

If a function in a server library is used in a CALL-EXTERNAL-FUNCTION verb, it must exist at the time the logic that calls it is assembled.

The functions in a server library can be written in C or C++, or any other language that has C-equivalent parameter and return types and its compiled code can be placed in a static or dynamic library.

Up to 50 parameters can be defined for server functions.

Client DLLs

Libraries on the client are called DLLs (Dynamic Link Libraries) and are composed of one or more exported functions; that is, a function that has been identified as exported so it can be linked from an external source at run time (in other words, dynamically linked).
The functions in the DLL can be written in C or C++, or any other language that supports the creation of Windows DLL files and do not have to exist until the time the logic that calls them is executed.

Up to sixteen parameters can be defined for client functions.

**OLE Dispatch Interfaces**

Reserved for future development.

To define a library, select the following:

EDGE / Guide / Libraries / Functions tab

Then select modify or add. A property sheet similar to the following is displayed:
Properties:

ActiveX name  The ActiveX control name, which is the name of a control that has been configured on the server. Must be selected from the list of existing configured ActiveX controls.

Available only if the Library type is ActiveX control.

For more information, see the section Configuring ActiveX Controls.

Debug mode  If checked and a function in this library is specified in the CALL-EXTERNAL-FUNCTION verb, the function is not actually invoked. Instead, previously specified values are returned.

Not available with ActiveX controls.

For more information, see the section Notes, which follows.
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>External library name</td>
<td>The name used in EDGE to identify the library.</td>
</tr>
<tr>
<td>External library path name</td>
<td>The full path to the library. If there are functions in the library that call functions in other libraries, the pathnames to each of the called libraries must be supplied in the order required by the functions. Each pathname must be separated by one or more spaces.</td>
</tr>
</tbody>
</table>
| Library type            | The library used for addressing; can be one of the following:  
  - 16-bit Client DLLs.  
  - 32-bit Client DLLs.  
  - ActiveX Controls.  
  - OLE Dispatch Interfaces.  
  - Server Libraries. |
Notes

Debug and trace mode can be set without having to reassemble logic flows and tasks, as the mode is determined dynamically at runtime.

If debug or trace mode is set for a library, the setting affects all the logics and tasks that call functions in that library. It is not possible to set an external function to debug mode for one task or logic only.

To define a user external function, select the following:

EDGE / Guide / Libraries / Functions tab

Select the library that contains the function, then select modify or add. A page similar to the following is displayed:

Properties: (continued)

Trace mode  If checked and a function in this library is specified in the CALL-EXTERNAL-FUNCTION verb, the current values of its parameters are written to stderr each time the verb is called.

For more information, see the section Notes, which follows.
**Properties: Functions**

- **Call Context**: Reserved for future development.
- **Calling convention**: The calling convention for this function; can be one of the following:
  - Pushed onto stack left to right
  - The Pascal calling convention; used for 16-bit DLLs.
Properties: Functions (continued)

Pushed onto stack  This is used for 32-bit DLLs. right to left.
The default is Pushed onto stack right to left.
In both cases, the called function cleans up the stack.
This property is used only with DLLs.

Debug mode  If checked and this function is specified in the CALL-EXTERNAL-FUNCTION verb, the function is not actually invoked. Instead, the values specified as the debug values are returned.

Debug value  The value to use as the function's return value when the library or function is in debug mode.

External function ID name  The name used in EDGE to identify the external function. This name is used in the CALL-EXTERNAL-FUNCTION verb.

External function name  The name of the function as it is defined in the library.

Return type  The type of the return value; can be one of the following:
**Properties: Functions (continued)**

- 8-bit quantity unsigned.
- 8-bit quantity signed.
- 16-bit quantity unsigned.
- 16-bit quantity signed.
- 32-bit quantity unsigned.
- 32-bit quantity signed.
- 32-bit floating point value.
- 64-bit floating point value.
- Null terminated ASCII string.
- Pointer to a 64-bit floating point value.
- Pointer to a string.
- Pointer to a single character.
- Single character.
- Void.

**Trace mode**

If checked and this function is specified in the **CALL-EXTERNAL-FUNCTION** verb, the current values of its parameters are written to stderr each time the verb is called.

**Properties: Library Information**

- **External library name**
  
  The **External library name**, as specified in EDGE, that contains this function. Display only.

- **External library type**
  
  The **Library type**, as specified in EDGE, that contains this function. Display only.
**Properties: Library Information (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Library Reference</td>
<td>The <strong>External library path name</strong> for this function. Available only if the library type is server library or client DLL. Display only.</td>
</tr>
</tbody>
</table>

**Properties: Parameters**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug value</td>
<td>The value to use as the parameter’s value when the library or function is in debug mode.</td>
</tr>
<tr>
<td>Parameter usage</td>
<td>Indicates how this parameter is to be used; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>• Input only.</td>
</tr>
<tr>
<td></td>
<td>• Input and output.</td>
</tr>
<tr>
<td></td>
<td>• Output only.</td>
</tr>
<tr>
<td></td>
<td>This is used internally in EDGE to optimize the call.</td>
</tr>
<tr>
<td>Position</td>
<td>The order of this parameter in the function; this property is display only.</td>
</tr>
<tr>
<td>Prompt</td>
<td>The text to be displayed for this parameter in the <strong>CALL-EXTERNAL-FUNCTION</strong>. If not specified, the parameter is displayed as Parameter n, where n is the position.</td>
</tr>
<tr>
<td>Type</td>
<td>The parameter type; for a list of types, see the description of <strong>Return type</strong>.</td>
</tr>
</tbody>
</table>
Multiple Project Assembly and Swap

The Multiple Project Assembly and Swap module is used to assemble and swap multiple projects in a single command. When you create an assembly instruction, you specify the projects to be assembled and swapped. (Assembling and swapping are described in Chapter 24, Finishing Touches.)

There is no limitation on the number of projects you can have in an instruction. Once the instruction is created, it can be used to assemble and swap the projects whenever needed.

To define an assembly instruction, select the following:

EDGE / Guide / Multiple Project A/S tab

Then click the name of an existing instruction, or click Add to add an instruction. A page similar to the following is displayed:
**Properties:**

- **Instruction ID**: The identifier assigned to the instruction.
- **Max Errors**: The maximum number of errors to accept before swapping the guide. If the number is reached, the guide is not swapped. This is done on a guide-by-guide basis.
- **Project**: The project name; can be selected from all projects on the system.
- **Swap Project**: If checked, the guide is swapped after the assembly, assuming the maximum number of errors has not been reached.
Caution! If a full assembly is required, but a partial is specified, only a partial assembly is performed. This may impact the execution of the guide.

**Run the Multiple Assembly/Swap**

To run the assemble/swap instruction, select the instruction from the Multiple Project A/S tab sheet, then select the Run Assemble/Swap command from either the Guide menu or from the tab sheet.

To begin the process, select the instruction, then select Run Assemble/Swap command. When all the assemblies are completed the instruction screen is displayed.

As the assembly progresses, information about the assembly is displayed similar to the following:

**Properties: (continued)**

Type of Assembly The type of the assembly; can be one of the following:

- Full.
- Partial.
- None.
For information on performing multiple assemblies from ACL, see the command `MULTI.ASSEM`.

**Listing Assembly Results**

The following reports are available from the Multiple Assemble/Swap menu:

- **List All Assembly Results**
  - Provides a summary report of all the multiple assemblies that have been done.

- **List Current Assembly Results**
  - Lists the results for the current assembly.
Each project is shown only once. This means that if a project is listed more than once in an instruction, only the results of the last entry are shown.

The reports are similar to the following:

```
MULTIPLE ASSEMBLE/SWAP REPORT

<table>
<thead>
<tr>
<th>PROJECT</th>
<th>NAME</th>
<th>START</th>
<th>ASSEM</th>
<th>ASSEM</th>
<th>ASSEM</th>
<th>SWAP</th>
<th>SWAP</th>
<th>SWAP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SOFTSELL</td>
<td>06/27/96</td>
<td>04:52PM</td>
<td>04:52PM</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
**Display Fields**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSEM ERRORS</td>
<td>The number of assembly errors. Errors are stored in /usr/include/sys/error.h.</td>
</tr>
<tr>
<td>ASSEM START</td>
<td>The time the multiple assembly process started.</td>
</tr>
<tr>
<td>ASSEM STOP</td>
<td>The time the multiple assembly process stopped.</td>
</tr>
<tr>
<td>PROJECT NAME</td>
<td>The name of the project.</td>
</tr>
<tr>
<td>START DATE</td>
<td>The date the multiple assembly process started.</td>
</tr>
<tr>
<td>SWAP ERRORS</td>
<td>The number of swap errors.</td>
</tr>
<tr>
<td>SWAP START</td>
<td>The time the swap process started.</td>
</tr>
<tr>
<td>SWAP STOP</td>
<td>The time the swap process stopped.</td>
</tr>
</tbody>
</table>
Schemes

Schemes are a collection of styles to be used to change the appearance of your screens in EDGE Developer and EDGE Client. The styles in each scheme are different combinations of colors and fonts.

Schemes can be set at the EDGE level, project level, and operator access level. If no scheme is selected at a level other than the EDGE level, the option set for Scheme on the GEO Scheme page is used in the following order of precedence:

1. Operator access level. The default is None.
2. Project level. The default is None.
3. EDGE level.

For more information, see the section EDGE Client Options, Labels, and Schemes.

At the EDGE level, you must select a scheme or the scheme named DEFAULT, otherwise you will encounter assembly errors.

The DEFAULT scheme is predefined in EDGE. If you want to modify how the DEFAULT scheme is configured, see the section Defining Schemes.

Characteristics defined in one scheme do not carry over to another scheme. For example, you can name a style Text box and define it
with one set of characteristics for your administrative project and another set in your outbound project. If you copy screens from one project to another, the characteristics of the style are determined by the current project.

**Defining Schemes**

Schemes are defined or modified through the Scheme editor. To display the Scheme editor page, select the following:

- EDGE / Guide / Schemes tab

When you add or modify a scheme, a page similar to the following is displayed:
Note: To define style names, use the Styles page. For more information, see the section Styles.
**Properties:**

**Colors**

The colors used for this style. To change a color, select the three-dot button next to the appropriate color. The standard Windows 95/98 color palette is displayed. Alternatively, if you know the RGB Colors, you can enter them directly. When a color is changed, the sample text is also changed.

Colors can be specified for the following:

- Text color.
- Background color.
- Border color.
- Disabled text color.
- Disabled background color.
- Non-visited links color. (CyberEDGE only)
- Active links color. (CyberEDGE only)
- Visited links color. (CyberEDGE only)
- URL of background image. (CyberEDGE only)

**Description**

A description of the scheme.

This property is optional.
RGB Colors

An RGB code consists of values for each of the components of the color, with 0 representing none of the color and 255 representing...
the highest amount of the color. For example, the RGB code for red is 255,0,0. The RGB code for white is 255,255,255. Sample color codes are displayed on the following table:

**Table 23-1. Sample RGB Codes**

<table>
<thead>
<tr>
<th>Color</th>
<th>RGB Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>black</td>
<td>0,0,0</td>
</tr>
<tr>
<td>blue</td>
<td>0,0,255</td>
</tr>
<tr>
<td>green</td>
<td>0,255,0</td>
</tr>
<tr>
<td>gray</td>
<td>128,128,128</td>
</tr>
<tr>
<td>red</td>
<td>255,0,0</td>
</tr>
<tr>
<td>white</td>
<td>255,255,255</td>
</tr>
</tbody>
</table>

Schemes can be applied at the following levels:

- Globally.
- Project-wide
- by Agent.

Styles in a scheme set for an agent take precedence over styles in a scheme set at the project level, which in turn take precedence over styles in the system-wide scheme.
To apply a scheme globally, select the following:
   EDGE / Server / GEO Setup / GEO Scheme

To apply a scheme to a project, select the following:
   Project / Setup / GEO Setup / GEO Scheme

To apply a scheme for an agent, select the following:
   Project / Setup / Operator Access

Select a user ID to modify, then select the GEO Scheme tab.

To allow any agent to apply a different scheme to their workstation from within EDGE Client, select the following:
   EDGE / Server / GEO Setup

Then check the boxes, Allow scheme modifications and Allow setup modifications. The following shows the check boxes needed to be checked:
To allow agents to change schemes from within EDGE Client, check these boxes.
Styles

Styles are used to define combinations of colors and fonts. The Styles tab is used only to create or modify style names.

To define a new style name or modify an existing one, select the following:

EDGE / Guide / Styles tab

Then select a style to modify, or click Add to add a new one. A dialog box similar to the following is displayed:

To create a new style, enter a name. To modify an existing style name, enter a new name over the existing one. At this time, you cannot delete style names.

The characteristics for a style are defined in the scheme that is to use the style. For more information, see the section Schemes.
System Screens

EDGE system screens are standard elements that are used for a variety of functions such as scheduling callbacks, ending a call, or presenting options to the operator.

Note: The EDGE Client screens that agents use to log in, to start operations, and to process information between calls are described in the EDGE 7.11 Client Reference Manual.

These functions set internal EDGE variables, write information to EDGE files, and route to EDGE screens. For example, the $END function writes information to files, sets internal variables, then routes to the Call Center screen.

The EDGE functions can be executed in the following ways:

- From a logic flow using the ROUTE verb.
- From a user screen.
- Using the jump command from the menu bar in EDGE Client.

For information on routing, see Chapter 19. For information on the jump command, see the EDGE 7.11 Client Reference Manual.
These functions have names that begin with a dollar sign ($); for example, \$END. This name is also used as the screen name displayed in ASCII mode during operations. (The term \textit{function} is used to describe the method of accessing the EDGE screens from the guide. The term \textit{screen} is used to describe the actual screen displayed as part of the function.)

The name of the screen displayed in EDGE Client is a descriptive screen identifier, not the $ function name. If you are developing a guide for EDGE Client, use the EDGE $ function names; EDGE Client automatically presents the correct screen.

\textit{Note: Unlike ASCII, EDGE Client screen names do not correspond to function names.}

For a list of the EDGE functions, corresponding screen and a brief description of each function, see Table 23-2.

\textbf{Table 23-2. EDGE Screen Names in EDGE Client}

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Screen Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$CALLBACK</td>
<td>Allows operator to schedule callbacks.</td>
<td>Schedule Callback</td>
</tr>
<tr>
<td>$CALLHIST</td>
<td>Displays historical data for the current call record.</td>
<td>Call History</td>
</tr>
</tbody>
</table>
Modifying EDGE Screens for EDGE Client

The text displayed on the command buttons on the Call Center screen can be modified using the EDGE Client Labels page at the EDGE or project level. The text on the other EDGE Client command buttons cannot be changed. For more information, see the EDGE 7.11 Client Reference Manual.

EDGE is shipped with a set of default screens used in ASCII. When you create a project, a list of the EDGE screens is made available to you. You can use this list if you need to copy and modify a screen for your project. You can modify only a copy of an EDGE screen and then only certain features.

To modify one of the EDGE screens, first copy the screen, then modify the copy.

At the EDGE level, you can copy the following:

- Default screens.

### Table 23-2. EDGE Screen Names in EDGE Client (continued)

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Description</th>
<th>Screen Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$END</td>
<td>Ends the current call and invokes end of guide processing.</td>
<td>Call Center</td>
</tr>
<tr>
<td>$RET</td>
<td>Routes to previous screen.</td>
<td>Not applicable</td>
</tr>
</tbody>
</table>
• Copies of EDGE screens previously modified at the project level.

At the project level, you can copy the following:

• Default screens.
• Copies of EDGE screens previously modified at the project level.
• Copies of EDGE screens previously modified at the EDGE level.

At the EDGE level, you have access to all default screens. At the project level, you have access to a subset of the default screens. The description of each screen notes which levels have access to the screen.

You can modify the following elements of EDGE screens:

• Change or hide the screen title.
• Change the function key description that is displayed on the screen.
• Disable a function key that is defined for the screen.
• Hide all function keys for the screen.
You cannot change the information displayed on the screen, change the routing, nor add function keys.

**Copying an EDGE Screen**

To copy an EDGE screen at the EDGE level, select the following:

EDGE / Guide area / System Screens

A page similar to the following is displayed:

![EDGE Screen Copy](image)

To copy an EDGE screen at the project level, select the following:
project / Guide area / System Screens

A page similar to the following is displayed:

**Properties:**

- **Default System Screens**: Lists the default system screens available to be copied. Default screens are the unmodified or template version of the screens.

- **System screens**: The system screens that have been copied.
To copy these screens to a project or to the EDGE level, use the Drag and Drop method.

**Modifying an EDGE Screen**

To modify a copy of an EDGE screen, select System Screens from the Guide area at the EDGE or project level, as desired. The Select System Screen tab is displayed, listing the EDGE screens that you have previously copied at that level. Select the desired screen.

A screen similar to the following is displayed:
To modify the screen title or a function key description, type in the appropriate text.

To disable a function key, delete its description. You cannot enable a key that is not enabled by default. For example, adding text to a function key that does not have it by default has no effect.

To hide the function keys or title, select the appropriate option. If you hide function keys, the function keys can still be selected unless you have also disabled them. The guide defaults are set as part of Guide Options.

Note: The modifications apply only to screens displayed on ASCII terminals. To change EDGE screens that are displayed by EDGE Client, use the EDGE Client setup features, which are described in the EDGE 7.11 Client Reference Manual.
Alphabetical Listing of EDGE Functions

This section lists EDGE functions in alphabetical order and provides a detailed description of each function, and the screen displayed where applicable.

$AUTOEND

The $AUTOEND function displays the Call Center screen at the end of a call. This screen is displayed in place of $END when the $AUTOFKEY field is set. At the end of the delay set in $AUTODELAY, or following the default time-out of four seconds, the action associated with the function key specified in the field $AUTOFKEY is automatically performed. If $AUTOFKEY is not set, this screen is not displayed.
All processing performed by $END is also performed by $AUTOEND.

$AUTOEND can be modified at both the EDGE and the project level.

Note: This screen is only displayed for the length of time specified by $AUTODELAY; all commands are disabled so no selections can be made from this screen.

$CALLBACK

The $CALLBACK function routes agents to the Set Callback screen to allow agents to schedule callbacks. This function is available from the Call Center screen using the View Callbacks command, or from the guide.
The following commands are defined for the Set Callback screen:

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cancel</td>
<td>Clears the SCHED.DATE and SCHED.TIME fields in the STATUS file, then returns to the screen that invoked this screen.</td>
</tr>
<tr>
<td>Set</td>
<td>Saves the callback date and time in SCHED.DATE and SCHED.TIME in the STATUS file, then returns to the screen that invoked this screen.</td>
</tr>
<tr>
<td>View Callback</td>
<td>Displays the View Callback screen. For information, see the EDGE 7.11 Client Reference Manual.</td>
</tr>
<tr>
<td>Queues</td>
<td></td>
</tr>
</tbody>
</table>

The following information is displayed:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calendar</td>
<td>A calendar with the current month is displayed. The callback date can be specified by clicking the date on the calendar. The left and right arrow button can be clicked to change the displayed month; the calendar displays one month at a time.</td>
</tr>
<tr>
<td>Call event</td>
<td>The ID of the current call record. This property is display only.</td>
</tr>
<tr>
<td>Caller’s time</td>
<td>The current time and date at the customer’s location. The time zone information for the customer is derived from the customer’s telephone number in the STATUS file. This property is display only.</td>
</tr>
<tr>
<td>Function</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Last date to schedule</td>
<td>The last date to schedule callbacks for the current queue. For information on specifying the last date, see the section <a href="#">Setting the Last Day for Callbacks</a>. This property is display only.</td>
</tr>
<tr>
<td>Local time</td>
<td>The current time and date at the agent’s location. The time zone information for the agent is derived from the site setup information. This is display only.</td>
</tr>
<tr>
<td>Set callback date</td>
<td>The date to schedule the callback. This is the customer’s date. A date can be selected from the displayed calendar.</td>
</tr>
<tr>
<td>Set callback time</td>
<td>The time to schedule the callback. Enter the time in the customer’s time zone; EDGE automatically adjusts the callback time to the agent’s time zone so the call can be pulled from the queue at the scheduled time.</td>
</tr>
</tbody>
</table>

The Set Callback screen is a [Rebound screen](#).

$CALLBACK$ can be modified at both the EDGE and the project level.

$CALLHIST$  
The $CALLHIST$ function displays the Call History screen which shows the HISTORY file data for the current call event.
The following command is defined for the Call History screen:

**OK** Returns to the screen that invoked this screen.

The following historical data is displayed:

- **CB date** The dates of callbacks set for previous calls.
- **CB time** The times of callbacks set for previous calls.
- **Call event** The call record whose history is being displayed.
- **Called** The dates of the previous calls.
- **Queue** The queues assigned for the previous calls.
- **Result** The results set for the previous calls.
- **Today** The current date and time.
### $CALLHIST

$CALLHIST is a **Rebound screen**.

$CALLHIST can be modified at both the EDGE and the project level.

### $END

The $END function performs the actions described below, and routes the Call Center screen displaying the call event of the call just completed, the queue the call event has been placed in, and the result assigned to the call event.

<table>
<thead>
<tr>
<th>Total calls</th>
<th>The total number of calls in the HISTORY file for this call record. The maximum number of calls saved in the HISTORY file is specified in Guide Options.</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>The operator IDs associated with the previous calls.</td>
</tr>
</tbody>
</table>
The Call Center screen is described in the EDGE 7.11 Client Reference Manual.

Each path through the guide must end by routing to $END. If you design a guide with a path that does not end with a route to $END, the agent will not be able to exit the guide.

The following information is displayed:

- **Call Event**: The identifier for the call record.
- **Queue**: The queue into which this call record has been placed. The queue is specified as part of the result definition.
- **Result**: The result assigned to this call record. If a result has not been specified when this screen is routed, the $SETRESULT screen is automatically invoked.

The following actions are performed when $END is routed:

- The records for the current call event are written to the STATUS, HISTORY, REPORTING, and RESPONSES files.
- The project monitor is updated.
- The fields in the SYS file are cleared.
- Call variables (variables that start with a #) are cleared.

$END can be modified at both the EDGE and the project level.
The $INEND function displays the Call Center screen at the end of an inbound telephony-initiated call; the $PDINEND function displays the Call Center screen at the end of an outbound telephony-initiated call. At the end of the delay set in $AUTODELAY, or following the default time-out of four seconds, the user is routed to Auto-Receive (if inbound), or Auto-Dial (if outbound). For more information on Auto-Receive and Auto-Dial, see the EDGE 7.11 Client Reference Manual.

This function is used in place of $END when the telephony feature is used to initiate the call. All the processing that is performed by $END is performed by $INEND and $PDINEND. (For more information, see the description of $END in this chapter.)
$INEND and $PDINEND can be modified at both the EDGE and the project level.

Note: This screen is only displayed for the length of time specified by $AUTODELAY; all commands are disabled so no selections can be made from this screen.
Alphabetical Listing of ASCII Screens

This section lists ASCII screens in alphabetical order and provides a detailed description of each screen.

$AUTOEND

The $AUTOEND screen is displayed at the end of a call.

This screen is displayed in place of $END when both $AUTODELAY and $AUTOFKEY fields are set. At the end of the delay set in $AUTODELAY, the action associated with the function key specified in the field $AUTOFKEY is automatically performed. If $AUTOFKEY is not set, this screen is not displayed.
All the processing that is performed by $END is also performed by $AUTOEND.

$AUTOEND can be modified at both the EDGE and the project level.

The $CALLBACK screen allows the operators to schedule callbacks. You can route to this screen.

The following function keys are defined for the $CALLBACK screen:

- **F1** Saves the callback date and time in SCHED.DATE and SCHED.TIME in the STATUS file, then returns to the screen that invoked this screen.
F2  Clears the SCHED.DATE and SCHED.TIME fields in the STATUS file, then returns to the screen that invoked this screen.

F3  If a call was previously scheduled, returns to the screen that invoked this screen. No changes are copied to the STATUS file. If no call was previously scheduled, the following message is displayed:

    CALLBACK DATE MISSING

The following information is displayed:

CALL ID  The ID of the current call record.

DATE     The date to schedule the callback. This is the customer’s date.

LAST DATE TO SCHEDULE CALLBACKS  The last date to schedule callbacks for the current queue. (For information on specifying the last date, see the section Setting the Last Day for Callbacks.)

OUR TIME and DATE  The current time and date at the operator’s location.

THEIR TIME and THEIR DATE  The current time and date at the customer’s location. This information is derived from the customer’s telephone number.

TIME  The time to schedule the callback. This is the customer’s time.
$CALLBACK is a rebound screen.

$CALLBACK can be modified at both the EDGE and the project level.

$CALLHIST

The $CALLHIST screen displays the HISTORY file data for the current call event.

The following function key is defined for the $CALLHIST screen:

F1        Returns to the screen that invoked this screen.
The following historical data is displayed:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BY USER</td>
<td>The operator IDs associated with the previous calls.</td>
</tr>
<tr>
<td>CALL ID</td>
<td>The call record whose history is being displayed.</td>
</tr>
<tr>
<td>PREVIOUS</td>
<td>The dates of the previous calls.</td>
</tr>
<tr>
<td>QUEUE</td>
<td>The queue assigned for the previous calls.</td>
</tr>
<tr>
<td>RESULT</td>
<td>The results set for the previous calls.</td>
</tr>
<tr>
<td>SCHED.CALLBACK</td>
<td>The times and dates of callbacks set for previous calls.</td>
</tr>
<tr>
<td>TOTAL CALLS</td>
<td>The total number of calls in the HISTORY file for this call record.</td>
</tr>
<tr>
<td></td>
<td>The maximum number of calls saved in the HISTORY file is specified in Guide Options.</td>
</tr>
</tbody>
</table>

$CALLHIST is a rebound screen.

$CALLHIST can be modified at both the EDGE and the project level.

$END

The $END screen displays the call event of the call just completed, the queue the call event has been placed in, and the result assigned to the call event. This is the last screen of a call.
The following function keys are defined for the $END screen:

- **F1** Routes the operator to the First inbound screen as specified in Guide Options. This is the same screen that F1 on the $OP screen routes to.

- **F2** Routes the operator to the $GETID screen.

- **F3** Routes the operator to the First outbound screen, or if there are no calls, to $GETNEXT screen.

- **F4** Routes the operator to the $QVSELL screen.

- **F8** Routes the operator to the $OP screen.
Each path through the guide must end at this screen. If you design a guide with a path that does not end at this screen, the operator will not be able to exit the guide.

The following information is displayed:

- **CALL EVENT** The identifier for the call record.
- **QUEUE** The queue this call record has been placed in. The queue is specified as part of the result definition.
- **RESULT** The result assigned to this call record. If a result has not been specified when this screen is routed to, the $SETRESULT screen is automatically invoked.

The following actions are performed when $END is routed to:

- The records for the current call event are written to the STATUS, HISTORY, REPORTING, and RESPONSES files.
- The Project monitor is updated.
- The fields in the SYS file are cleared.
- Call variables (variables that start with a #) are cleared.

$END can be modified at both the EDGE and the project level.

**$ERR**

The $ERR screen displays an error message on the operator’s screen.
The following function key is defined for the $ERR screen:

F1 Returns to the screen that invoked this screen.

$ERR is an example of a rebound screen.

$ERR can be modified at both the EDGE and the project level.

$GETID

The $GETID screen provides the operator with options to manually access a call event.
The following function keys are defined for the $GETID screen:

- **F1** Retrieves the call record for the specified call ID, then Routes the operator to the First outbound screen.
- **F2** Creates a new call ID using the value entered at the prompt, then routes to the first outbound call.
- **F3** Has EDGE create a sequential call ID, starting at 1, then routes to the first outbound screen. For information on changing the starting value for creating sequential call IDs, contact EDGE Customer Care.
- **F8** Routes to the $OP screen.

$GETID can be modified at both the EDGE and the project level.
$GETNEXT

The $GETNEXT screen is displayed when the operator requests the next available call ID by pressing either F1 on the $END screen or F3 on the $OP screen and there are no calls available to the operator.

The following function key is defined for the $GETNEXT screen:

- **F1** Routes the operator to the $OP screen.

$GETNEXT can be modified at both the EDGE and the project level.

$INEND

The $INEND screen is displayed at the end of a telephony-initiated call. The user is then routed to $INWAIT.
This screen is displayed in place of $END when the telephony feature is used to initiate the call. All the processing that is performed by $END is performed by $INEND.

$INEND can be modified at both the EDGE and the project level.

$INERR

The $INERR screen is displayed when the telephony feature is used to initiate a call and a conflict occurs; for example, the number does not exist.
The following function keys are defined for the $INERR screen:

- **F1** Routes the operator to the $MAIN screen.
- **F2** Routes the operator to the $INWAIT screen.

For more information, see the documentation for telephony.

$INERR can be modified at the EDGE level only.

$INWAIT

The $INWAIT screen is displayed while the system waits for the next telephony call.
This screen is displayed until a call for the current operator is found. The operator is then automatically routed to the first inbound screen.

For more information, see the telephony documentation.

$INWAIT can be modified at the EDGE level only.

$MAIN

The $MAIN screen lists the projects available to the operator and prompts the operator for a selection.
The following function keys are defined for the $MAIN screen:

- **F1** Routes the operator to the project menu.
- **F2** Routes the operator to $INWAIT. This is available only when telephony is used and nothing has been typed at the ENTER prompt.
- **F3** Allows the operator to select a project based on a user-defined code.
- **F6** Displays the next page of available projects. This prompt is displayed only if the operator has more than 50 available projects.
- **F6** Displays the previous page of available projects. This prompt is displayed only if the operator has more than 50 available projects.
This screen is presented when the operator logs on, only if there is more than one project available to the operator. The operator can enter the number of the project to access.

For information on predictive dialing and telephony, see the documentation for telephony.

$MAIN can be modified at the EDGE level only.

$NONE

The $NONE screen is displayed when an operator enters operations and has not been assigned Operator Access to any projects.
The following function key is defined for the $NONE screen:

F8 Exit operations and returns to the login prompt.

$NONE can be modified at the EDGE level only.

$OP

The $OP screen provides the operator with functions to receive inbound calls, make calls manually, or request the next available call from a queue.
The following function keys are defined for the $OP screen:

- **F1** Routes the operator to the first inbound call screen. This is the same screen that F1 on the $END screen routes to. To specify the first inbound screen, use Guide Options.

- **F2** Routes the operator to the $GETID screen.

- **F3** Routes the operator to the first outbound screen, or if there are no calls, to $GETNEXT screen. To specify the first outbound screen, use Guide Options.

- **F4** Routes the operator to the $QVSEL screen.

- **F6** Routes the operator to the $PM screen.

- **F8** Routes to the $MAIN screen.
This screen is presented when the operator first enters the guide, or has pressed F8 at $END.

$OP can be modified at both the EDGE and the project level.

The $PDINWAIT screen is displayed while the system is waiting for the predictive dialer to get a call.

There are no function keys defined for this screen. However, the operator can exit by pressing either the BREAK or DELETE key. This routes the operator back to the $MAIN screen.
When a call comes in, EDGE determines the project and the operator is automatically routed to the first outbound screen. To specify the first outbound screen, use Guide Options.

For more information on predictive dialing and telephony, see the telephony documentation.

$PDINWAIT can be modified at the EDGE level only.

The $PM screen displays project monitor statistics for the operator.
The following function keys are defined for the $PM screen:

- **F1** Routes the operator to the $PMRG screen.
- **F8** Routes the operator to the $OP screen.

The following statistics for the operator are displayed:

- **AVERAGE IS** The average of all users on the project for this category.
- **CALLS PER HOUR** The calls per hour for the current project. This value is the result of dividing the total calls for this project by the hours worked for this project.
- **HOURS WORKED** The total hours worked for the current project, as well as all projects.
- **REV PER HOUR** The revenue per hour for the current project. This value is the result of dividing the total revenue for this project by the hours worked.
- **TOTAL CALLS** The total number of calls made and received for the current project, as well as all projects.
- **TOTAL REVENUE** The total of all sales for the current project.
- **TOTAL UNITS** The total of all units sold for the current project.
- **UNITS PER HOUR** The units per hour for the current project. This value is the result of dividing the total units for this project by the hours worked.
The Project Monitor is updated whenever an operator ends a call in that project; the operator’s statistics on this screen are updated whenever the operator views it. However, the averages are updated not more than once every ten minutes.

$PM$ can be modified at both the EDGE and the project level.

The $PMRG$ screen displays Project Monitor statistics for result groups.

The following function keys are defined for the $PMRG$ screen:

- **F1** Routes the operator to the $PM$ screen.
F8 Routes the operator to the $OP screen.

The following statistics are displayed for the result groups:

- **CALLS PER HOUR**: The calls per hour that have been assigned to this result. This value is the result of dividing the total calls for this result by the total hours worked with calls that were assigned to this result.
- **YOU**: The statistics for the operator who requested the report.
- **AVG**: The averages based on all operators.
- **RESULT**: The names of the results whose statistics are being displayed.
- **TOTAL CALLS**: The total number of calls made and received that have been assigned to this result.

The statistics are updated every ten minutes if activity is taking place. If no activity is taking place, the statistics reflect the last time they were updated.

$PMRG can be modified at both the EDGE and the project level.

$QVIEW

The $QVIEW screen displays scheduled callbacks by dates and time.
The following function keys are defined for the $QVIEW screen:

**F1** Displays the next page of callbacks. If there are no more pages, EDGE displays the callbacks starting at the beginning of the queue.

**F2** Displays the callbacks, starting at the date and time specified in the DATE and TIME prompts at the top of the screen. If no DATE or TIME is specified, the display starts with the first callback in the queue.

**F3** Selects the callback on the line specified in the SELECT # prompt at the top of the screen, then routes to the first outbound screen. To specify the first outbound screen, use Guide Options.

**F7** Routes the operator to the $QVSEL screen.
F8  Routes the operator to the $OP screen.

To specify DATE, the operator can use any standard format, such as 3/15/95 or 15 MAR 95. In addition, the following can be specified:

D  Current date.

Dn  Current date plus the specified number of days.

month day  The specified day in the current year.

To specify TIME, the operator can use any standard format, such as 1:00PM or 13:00. In addition, the following can be specified:

hA  The AM time starting at the hour specified by h, for example, 8A.

hP  The PM time, starting at the hour specified by h, for example, 8P.

T  The current time.

If there are no callbacks in the specified queue, the following message is displayed. The queue is specified in the $QVSEL screen.

THERE ARE NO CALLS IN THIS QUEUE.

The following fields are displayed by default:

- Call ID.
- Scheduled time and date of next callback.
You can specify other fields using the Callback queue display fields property in Guide Options.

$QVIEW can be modified at both the EDGE and the project level.

The $QVSEL screen displays the list of time-ordered queues that currently contain callbacks and to which the operator has access.

The following function keys are defined for the $QVSEL screen:

- **F1** Routes the operator to $QVIEW. To specify the queue to view, the operator must enter the line number at the ENTER prompt.
- **F8** Routes the operator to $OP.
For information on specifying queues for operators, see the section Operator Access.

This screen is displayed by pressing F4 from the $OP screen.

$QVSEL can be modified at both the EDGE and the project level.

The $SETRESULT screen is displayed if the operator is routed to $END and a result has not been set for the call.
The following function keys are defined for the $SETRESULT screen:

F1 Routes the operator to $END. A valid result must have been entered.

Note: This screen should never be displayed. All paths to $END should include routines for setting results.

$SETRESULT can be modified at both the EDGE and project level.

The $SYSERR screen displays a message indicating a system error has occurred.
The following function key is defined for the $SYSERR screen:

   F1       Exits operations and returns to the login prompt.

System errors are errors that occur in areas that are required for EDGE to work. For example, if EDGE cannot open the user ID file, a system error occurs.

$SYSERR can be modified at the EDGE level only.
Overview

As described in the sections Understanding Guides and Projects and Project Design, there are many things you need to do to create a project. Then, after you have created all the components of the project, you must get it ready for operations. This chapter discusses creating the following components:

- Bulletins.
- Guide Options.
- Hot Keys.
- Results.

It also describes the following steps to Getting the Guide Ready for operations:

- Assembling the Guide.
- Testing the Guide.
- Swapping.

This chapter also describes Periodic Maintenance.
Bulletins

The Bulletin module allows you to create a message that can be displayed to selected users. The message is displayed as the first screen after the user logs in. If more than one bulletin exists, they are displayed one after another. You can create as many bulletins as desired.

When a user logs in to EDGE Client or EDGE Developer, all the bulletins specified for that user are displayed, similar to the following:
The bulletin is displayed to the specified users each time they log in until the bulletin is deleted or the user is deleted from the list of users to be sent the bulletin. The command buttons are enabled when there is more than one bulletin.

For information on a similar feature that can be displayed before a project is opened, see the section Intro Screen.

To create or modify a bulletin, select the following:
Users area / Bulletins tab

The property page is similar to the following:

Special notice!

The regular backup will be done early this week because of the Friday holiday. Make sure you log off Thursday evening. Thanks.

System Administrator
### Properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Message</strong></td>
<td>The message to be displayed to users; enter without quotation marks unless you want them to be displayed. You must press ENTER to go to a new line. If the message has more than 10 lines, the scroll bar becomes active. Only 20 lines are displayed to the user.</td>
</tr>
<tr>
<td><strong>Send to the following users</strong></td>
<td>The users to send the message to; can be individual user IDs or all users. To specify individual IDs, click inside the box and select an ID from the list. To specify all users, click (All).</td>
</tr>
</tbody>
</table>
Guide File Access

Guide file access makes files available to the guide. You can also designate the primary file for the guide and specify abbreviations for the files.

The primary file is the file used by the Look Up screen when an agent needs to find a call ID. It is also the file that contains the fields displayed on the View Callbacks screen. (For more information on these screens, see the EDGE 7.11 Client Reference Manual.)

File abbreviations are used with field names to indicate the file that contains the field. For example, if the abbreviation for the INV.LIST file is I, you can specify the PROD.NO field as either INV.LIST-PROD.NO or I-PROD.NO.

You should provide guide file access before you design your screens or create your logic. In any case, you must provide file access before you can assemble your guide.

Note: Any saved change to guide file access requires that the next assembly performed be a full assembly.

To display the guide file access page, select the following:

Project / Guide area / Options tab / Guide File Access command
Properties:

File abbreviation  The abbreviation for the file. It must start with a letter. By default, the first two letters of the filename are used. Change the abbreviation as desired. If you use the same abbreviation for more than one file, EDGE automatically appends a sequential number.

You can specify more than one abbreviation for a file. You can then read more than one record from the file; for example, you can add CUST.LIST twice, identifying them as CU1 and CU2. You can then read a record from CU1 and a second record from CU2.
<table>
<thead>
<tr>
<th><strong>Properties:</strong> (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Guide files</strong></td>
</tr>
<tr>
<td><strong>Primary</strong></td>
</tr>
<tr>
<td><strong>System reserved</strong></td>
</tr>
</tbody>
</table>
Guide Options

Guide options allow you to define options related to the guide, such as the guide assembly default file. It also contains the definitions for hot keys.

To display the guide options page, select the following:

   Project / Guide area / Options tab / Guide Options command

The following tabs can be selected:

   • Guide Options 1.
   • Guide Options 2.
   • Hot Keys.

Guide Options 1

The Guide Options 1 tab specifies some guide defaults. It is recommended you first create your project files, queues, screens, logic flows, and results, and define access to files before setting these properties. The properties on this screen should be set before you assemble your guide, or you may get errors due, for example, to an incorrect guide assembly default file.

When the Guide Options 1 tab is selected, a page similar to the following is displayed:
Properties:

- **Allow routing**: If checked, specifies routing can be specified in the end of guide logic.
- **Callback queue display field(s)**: The fields from the primary file that are to be displayed in the View Callbacks (SQVIEW) screen.

Up to 80 characters can be displayed. By default, EDGE displays the Call ID and callback time and date, which are included in the 80 character limit. If 80 characters is exceeded, none of the fields display.
The method of saving alphabetic characters to the database; can be one of the following:

- **As displayed**: Uses the capitalization specified in the Formatting property in the field definition. For information, see the section General Tab in Chapter 11, User Files and Fields.
- **Capitalize**: The first character of each word is capitalized. All other letters are lowercase.
- **Lowercase**: All letters are converted to lowercase.
- **None**: All fields are saved as entered by the agent. This is the default.
- **Uppercase**: All letters are converted to uppercase.

Note that the Formatting property in the field definition indicates how the field is to be displayed; this property indicates how the data is to be saved.

**Note**: The Data entry text conversion property on the EDGE Client Options tab has precedence over this property if set at the operator access, project, or EDGE level.
**Properties: (continued)**

**End-of-guide logic**

The names of logic flows that are performed at the end of the guide. The logic flows are executed sequentially in the order specified. There is no default.

**Note:** Logic flows with a Parameter Definition cannot be used here.

The logic flows are performed just before the functions performed by $END and do not impair the agent’s productivity or force the customer to wait for processing. For this reason, the end of the guide is a good place to attach logic flows that write data to files.

**First inbound screen**

The first user-defined screen to route to if the agent selects Receive Call on the Call Center screen. This is also the screen routed to when in telephony mode after receiving an inbound call on the Auto-Receive screen. The default is FIRST.

When the three-dot button is selected, a Window Helper page is displayed that allows the window as well as the screen to be specified.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>First outbound screen</td>
<td>The first user-defined screen to route to if the agent selects Next Call or Make Call on the Call Center screen. This is also the screen routed to when a callback is selected from the View Callbacks screen, or when a predictive dialer call is passed to the agent on the Auto-Dial screen. The default is FIRST. When the three-dot button is selected, a Window Helper page is displayed that allows the window as well as the screen to be specified.</td>
</tr>
<tr>
<td>Guide assembly default file</td>
<td>The name of the default file for the entire guide. Any field used on a screen or in a logic flow that does not reference a specific file is assumed to be in this file. Generally, this file is the primary file for the project. The default is the RESPONSES file.</td>
</tr>
<tr>
<td>Maximum HISTORY calls saved</td>
<td>The total number of call records to be saved in the HISTORY file for each call event. The default is 10.</td>
</tr>
<tr>
<td>New call ID default queue</td>
<td>The queue in which new call events are placed, unless otherwise assigned. The default is the ERROR queue. When a call event is created, it is not in any queue. The queue is typically set using an EDGE result, but if no queue is assigned, newly created call events are placed in this queue.</td>
</tr>
</tbody>
</table>
The commands that are to be suppressed throughout the guide. If a box is checked, the command represented by that code is unavailable to agents. For a list of codes and the command each represents, see Table 24-1.

Most of these commands are only available in ASCII operations; the EDGE Client equivalents are indicated in the table where appropriate. For information on using commands in ASCII operations, see Appendix AC, Keyboard Commands in ASCII Operations.

Table 24-1. Screen Commands

<table>
<thead>
<tr>
<th>ASCII Command</th>
<th>EDGE Client Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.B</td>
<td>No equivalent.</td>
<td>Moves the cursor to the previous input object.</td>
</tr>
<tr>
<td>.C</td>
<td>No equivalent.</td>
<td>Moves the cursor right to the next input object.</td>
</tr>
<tr>
<td>.E</td>
<td>Clear Field (CTRL+E)</td>
<td>Erases the entry at the current input location.</td>
</tr>
<tr>
<td>.G</td>
<td>No equivalent.</td>
<td>Moves the cursor to another row or input location.</td>
</tr>
<tr>
<td>.J</td>
<td>CTRL+N</td>
<td>Jumps (routes directly) to another screen.</td>
</tr>
</tbody>
</table>
Guide Options 2

The Guide options 2 tab is used to set the values for screen objects, such as the value to use for True and False. When the Guide options 2 tab is selected, a page similar to the following is displayed:

Table 24-1.  Screen Commands (continued)

<table>
<thead>
<tr>
<th>ASCII Command</th>
<th>EDGE Client Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.L</td>
<td>No equivalent.</td>
<td>Moves the cursor up to the previous line.</td>
</tr>
<tr>
<td>.N</td>
<td>No equivalent.</td>
<td>Moves the cursor down to the next line.</td>
</tr>
<tr>
<td>.P</td>
<td>Previous (CTRL+P)</td>
<td>Moves the cursor to the previous screen.</td>
</tr>
<tr>
<td>.R</td>
<td>No equivalent.</td>
<td>Modifies text at an input object.</td>
</tr>
<tr>
<td>.T</td>
<td>Top (CTRL+T)</td>
<td>Moves the cursor to the first input object on the screen.</td>
</tr>
<tr>
<td>.W</td>
<td>No equivalent.</td>
<td>Undoes the change in the input object.</td>
</tr>
<tr>
<td>.Z</td>
<td>No equivalent.</td>
<td>Prints the screen.</td>
</tr>
</tbody>
</table>
Properties:

Close windows at end of call

Specifies whether Windows are closed during end-of-guide processing. The default is checked.

- **Checked** - Windows are closed during end-of-guide processing.
- **Cleared** - Windows remain open during end-of-guide processing. The data is cleared from the window.
Enable EDGE system transaction

Specifies whether EDGE system transactions are used. An EDGE system transaction guarantees that all EDGE files, such as STATUS, are updated and is generally specified when your EDGE files are relational database tables. For information on transactions, refer to the EDGE DBL documentation.

The default is cleared.

Enhanced range validation

Specifies whether range validation on input fields is to be based on the data type. The default is cleared.

- **Checked**
  Range validation is based on data type. For example, in a date field with a range of 4/1/97 to 4/30/97, enhanced validation would determine that 3/1/97 is an invalid value.

- **Cleared**
  Range validation is based on internal date formats. (Internal format for Date type fields is the number of days since December 31, 1967. Internal format for Time type fields is the number of seconds since midnight.)
For example, in a date field with a range of 10,000 to 10,120, an internal date of 10,500 is an invalid value. Dates entered in external format would also be invalid values.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match case in member-of-set validation</td>
<td>Specifies whether validation on member-of-set fields is case-sensitive. This indicates whether an agent entry of yes matches a member-of-set validation of YES. The default is cleared.</td>
</tr>
<tr>
<td>OLE object file path name</td>
<td>OLE is no longer supported. This property is retained for backward compatibility.</td>
</tr>
<tr>
<td>Show function key labels</td>
<td>Specifies whether function keys are displayed in the guide. A change to this property requires that the next assembly be a full assembly.</td>
</tr>
<tr>
<td></td>
<td><strong>Checked</strong> Function keys are displayed on screen.</td>
</tr>
<tr>
<td></td>
<td><strong>Cleared</strong> Function keys are hidden. Function keys are still active and will execute if selected, however they do not show on screen. The function keys can still be seen on the EDGE Client Go menu.</td>
</tr>
<tr>
<td></td>
<td>The default is checked. You can override the guide options setting in individual screens, if desired.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Show row labels</td>
<td>Specifies row labels to be displayed on screens. This is the default. You can override the guide options setting in individual screens, if desired.</td>
</tr>
<tr>
<td>Show title</td>
<td>Specifies whether the screen title is displayed on screens. A change to this property requires that the next assembly be a full assembly.</td>
</tr>
<tr>
<td></td>
<td>The default is checked. You can override the guide options setting in individual screens, if desired.</td>
</tr>
<tr>
<td>True and False values for button objects</td>
<td>The value to be returned by EDGE to indicate the true and false state of button objects used on screens. You can specify up to five characters for each value.</td>
</tr>
<tr>
<td></td>
<td>This value is placed in the Value field associated with the button object. Used with check boxes, command buttons, and radio buttons.</td>
</tr>
<tr>
<td></td>
<td>EDGE uses a case-insensitive search; for example, if the True value is specified as OK, any of the following is considered true:</td>
</tr>
<tr>
<td></td>
<td>OK</td>
</tr>
<tr>
<td></td>
<td>Ok</td>
</tr>
<tr>
<td></td>
<td>ok</td>
</tr>
<tr>
<td></td>
<td>okK.</td>
</tr>
<tr>
<td></td>
<td>Any value that does not exactly match the true value (except for case) is considered false.</td>
</tr>
</tbody>
</table>
## Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Warn ASCII users when routing to a window</td>
<td>Specifies whether to warn agents using ASCII operations when a route is executed to a destination that includes a window name. Windows are only available in EDGE Client. The default is checked.</td>
</tr>
<tr>
<td></td>
<td><strong>Checked</strong> A warning is displayed to an agent in ASCII operations when a logic flow including a route to a window is executed. EDGE then attempts to process the route to screen and field. The Window route will not occur.</td>
</tr>
<tr>
<td></td>
<td><strong>Cleared</strong> No warning is displayed to an agent in ASCII operations when a logic flow including a route to a window is executed. EDGE attempts to process the route to screen and field. The Window route does not occur.</td>
</tr>
<tr>
<td>Windows Help file pathname</td>
<td>Name of a user-defined Windows help file to be used with EDGE Client.</td>
</tr>
</tbody>
</table>
Hot Keys

A hot key is a sequence of key strokes that provides a shortcut to a feature. In EDGE, hot keys give the ability to define project-wide key-strokes for operations. A hot key can be defined to perform any of the following:

- Route to a screen.
- Execute a logic flow.
- Set a result.

Hot keys are valid for the entire project. They cannot be assigned at the system level nor for an individual screen. Also, note the following:

- The maximum number of key strokes in a hot key sequence is 8.
- The maximum number of hot keys per project is 225.
- In EDGE Client, hot key sequences take precedence over menu shortcuts with the same key sequence. To change the precedence, you can use the edgeo.ini option hotkeyPriority in the [guide] section. For information on edgeo.ini options, see the EDGE 7.11 Client Reference Manual.
- Hot key sequences that are not available on a terminal are ignored. For example, if there is no Alt key, a hot key based on
an Alt key sequence does not work. If you are using ASCII terminals in operations, the codes returned may vary among different terminal types. On PCs, some key strokes, such as numlock, print screen, and caps lock, are intercepted by the underlying operating system and are never returned to EDGE.

- Hot keys are assembled and swapped as a guide option.

When you select the Hot Keys tab, a page similar to the following is displayed:
**Properties:**

**Description**
Describes the hot key. Optional.

**Exit LF?**
Specifies whether the exit from screen logic is to be performed.

**Hot key sequence**
The keystroke sequence being defined. To specify a nonprintable character such as the `ESCAPE` key, use the appropriate code listed in Table 24-2. To specify a printable character, type the character.
Properties: (continued)

It is recommended that key strokes begin with a non-printable character. The codes for all nonprintable characters except ALT (%), CONTROL (^), and SHIFT (+) must be enclosed in braces. For example, to indicate ESCAPE+8, define the keystrokes as follows:

{esc}8

To indicate ALT+8, define the keystrokes as follows:

%8

To specify a percent sign, caret, or plus sign as a key-stroke, enclose the key in braces. For example, to indicate % 8, define the keystrokes as follows:

{%}8

To indicate keys that are to be repeated, enclose the keys and the number of repetitions in braces. For example, to indicate that the ESCAPE key is to be pressed 5 times, define the keystrokes as follows:

{esc 5}

Logic

The logic to perform when the hot key sequence is pressed.

Note: A logic flow with a Parameter Definition cannot be used here.
Properties: (continued)

Result  The result to set when the hot key sequence is pressed.
Route to screen  The screen to route to when the hot key sequence is pressed.

Table 24-2. Codes for Hot Keys

<table>
<thead>
<tr>
<th>Key</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALT</td>
<td>%</td>
</tr>
<tr>
<td>BACKSPACE</td>
<td>{backspace} or {bs} or {bksp}</td>
</tr>
<tr>
<td>BREAK</td>
<td>{break}</td>
</tr>
<tr>
<td>CAPS LOCK</td>
<td>{capslock}</td>
</tr>
<tr>
<td>CLEAR</td>
<td>{clear}</td>
</tr>
<tr>
<td>CONTROL</td>
<td>^</td>
</tr>
<tr>
<td>DEL</td>
<td>{delete} or {del}</td>
</tr>
<tr>
<td>DOWN ARROW</td>
<td>{down}</td>
</tr>
<tr>
<td>END</td>
<td>{end}</td>
</tr>
<tr>
<td>ENTER</td>
<td>{enter} or ~</td>
</tr>
<tr>
<td>ESC</td>
<td>{escape} or {esc}</td>
</tr>
<tr>
<td>F1</td>
<td>{f1}</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>F16</td>
<td>{f16}</td>
</tr>
</tbody>
</table>
### Table 24-2. Codes for Hot Keys (continued)

<table>
<thead>
<tr>
<th>Key</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>HELP</td>
<td>{help}</td>
</tr>
<tr>
<td>HOME</td>
<td>{home}</td>
</tr>
<tr>
<td>INS</td>
<td>{insert}</td>
</tr>
<tr>
<td>LEFT ARROW</td>
<td>{left}</td>
</tr>
<tr>
<td>NUM LOCK</td>
<td>{numlock}</td>
</tr>
<tr>
<td>PAGE DOWN</td>
<td>{pgdn}</td>
</tr>
<tr>
<td>PAGE UP</td>
<td>{pgup}</td>
</tr>
<tr>
<td>PRINT SCREEN</td>
<td>{prtsc}</td>
</tr>
<tr>
<td>RIGHT ARROW</td>
<td>{right}</td>
</tr>
<tr>
<td>SCROLL LOCK</td>
<td>{scrolllock}</td>
</tr>
<tr>
<td>SHIFT</td>
<td>+</td>
</tr>
<tr>
<td>TAB</td>
<td>{tab}</td>
</tr>
<tr>
<td>UP ARROW</td>
<td>{up}</td>
</tr>
</tbody>
</table>
Results

The result identifies the outcome or disposition of a call event. For example, a call to a wrong number might be identified with a result called WRONG.NO. A call ending with a sale might be given a result called SALE.

Generally, you assign each result to an appropriate queue. For example, a result called WRONG.NO might be assigned to a queue called PURGE. (The PURGE queue containing the wrong numbers would not be placed in any agent’s operator access; therefore, you could ensure that wrong numbers are not called again.)

If you do not assign a result to a queue, the call event returns to the queue from which it originated, as identified in $OLDQUEUE, unless $QUEUE is set to another queue through logic.

Note: Currently, the result is set for the call ID. You cannot set a result for each call event.

To specify results, select the following:

Project / Guide area / Results tab

Result Definitions

When you add or modify a result, a page similar to the following is displayed:
Properties:

Callback arranged by

The method by which the callback is arranged; can be one of the following:

- None
  - No callback is set for this result.
- Operator
  - The agent is responsible for scheduling the callback using the Set Callback screen.
<table>
<thead>
<tr>
<th>Properties: (continued)</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
</tr>
<tr>
<td>Description</td>
</tr>
<tr>
<td>Do NOT update</td>
</tr>
<tr>
<td>Reset recall count</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>System</td>
<td>The system assigns the callback based on the setting in the System arrangement group box on this page. The default is system.</td>
</tr>
<tr>
<td>Description</td>
<td>The description of the result. This field is optional.</td>
</tr>
<tr>
<td>Do NOT update</td>
<td>Specifies whether these files should be updated when this result is set; one or more of the following can be checked:</td>
</tr>
<tr>
<td></td>
<td>HISTORY file.</td>
</tr>
<tr>
<td></td>
<td>REPORTING file.</td>
</tr>
<tr>
<td></td>
<td>Global and project monitors.</td>
</tr>
<tr>
<td></td>
<td>By default, these files are updated at the end of each call. If checked, EDGE does not write these records for the current call event.</td>
</tr>
<tr>
<td>Reset recall count</td>
<td>Specifies whether the number-of-times-called field (RCNT field in the STATUS file) is incremented each time this result is set for a call event. For example, if recording the number of contacts for a call event, you might want to reset the count each time a sale it made.</td>
</tr>
<tr>
<td>Checked</td>
<td>The RCNT field in the STATUS file is reset to zero each time this result is set.</td>
</tr>
</tbody>
</table>
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Cleared</strong></td>
<td>The RCNT field in the STATUS file is incremented by one each time this result is set. The default is cleared.</td>
</tr>
<tr>
<td><strong>Send to queue</strong></td>
<td>The queue in which the call event is placed when this result is set. If no queue is specified, the call event is returned to the queue from which it originated. If the queue is a protected queue, as in the example, EDGE sends the call to the queue for the current user (/U user-protected) or group (/G group-protected). The default is no queue is specified.</td>
</tr>
</tbody>
</table>
| **System arrangement**    | The time and day offset for system arranged callbacks. The fields in the group box are used together to determine the correct date and time. The schedule used to determine the callback date; can be one of the following:  
  - Calendar days which indicates the calculated time and date is not to be adjusted, and calls may be scheduled 24 hours a day, 7 days a week. |
**Properties: (continued)**

- **Work Schedule** which indicates the calculated time and date is adjusted to fall within the work schedule.

  The default is Calendar days.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Days</td>
<td>The number of days to add to the current day to determine the callback date. The default is 1.</td>
</tr>
<tr>
<td>Hours</td>
<td>The number of hours to add to the current time to determine the callback time. The default is 1.</td>
</tr>
<tr>
<td>Minutes</td>
<td>The number of minutes to add to the current time to determine the callback time. The default is 0.</td>
</tr>
<tr>
<td>Version</td>
<td>The current version of the record. A value is displayed only if Version Control is enabled.</td>
</tr>
</tbody>
</table>

**List Results**

The List Results command is located on the Results menu, and displays the properties for the currently defined results. The listing is similar to the following:
The following fields are displayed:

- **Result ID**: The name of the result definition.
- **Description**: The description of the result.
- **Callback by**: The method of assigning callbacks.
- **Days**: The number of days to add when assigning the callback date.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours</td>
<td>The number of hours to add when assigning the call-back time.</td>
</tr>
<tr>
<td>Min</td>
<td>The number of minutes to add when assigning the call-back time.</td>
</tr>
<tr>
<td>Reset recall</td>
<td>Whether the recall count is reset to zero.</td>
</tr>
<tr>
<td>Send to new queue</td>
<td>The queue to which the call event is sent.</td>
</tr>
</tbody>
</table>
Getting the Guide Ready

After you have created all the components of your guide, you must follow these steps to prepare the guide for operations:

1. Assemble the guide.
2. Resolve assembly errors.
3. If needed, reassemble the guide until the assembly is error free.
4. Test the guide to see that it works as you intended. If not, make changes and start with step 1.
5. Clear test data; for example, if you have added customer records or call events, they should be cleared. If you have used the reporting feature, the REPORTING File should be cleared.
6. Swap the guide to operations.
7. Test the operations guide to see that it continues to work as you intended. If it does not, swap the guide again and resolve problems.
8. If this is a new guide or if you have made significant changes to the guide, we recommend you swap the guide when no agents are using the guide and test the guide before the agents begin to use it.
9. After the guide works from the Test Guide option as you intended, set the project options to give operators access to
the project and run the guide from operations as an agent. Again, be sure no agents are using the guide. During the testing phase from operations, make sure that you cannot access records in someone else’s protected queue and that you cannot access a queue to which you have not been assigned.

10. Before the agents access the project, you may want to change the order projects are listed in the Project Selection Menu, or specify a different Short description for agents.
Assembling the Guide

After you have completed the guide, you can assemble it. The assembly process checks and links all the components you have created producing an executable program called XGUIDE.

You can assemble a development guide while the operations guide is in use. However, no one can be using the development guide. Although most assemblies have little effect on system performance, a full assembly of a large guide may affect performance. For more information, see the section Technical Notes.

Note: If you are using Version Control, ensure the correct versions of all areas are checked in.

Types of Assembly

There are two types of assembly:

- Full.
- Partial.

A full assembly checks the entire guide: all files, guide options, logic flows, results, screens, and tables. A partial assembly checks only those changes you have made since the last assembly.
A full assembly is required whenever you have a new guide that has not been assembled, or whenever you make a change in key components of the guide. Once you do a full assembly, if you need to reassemble, you may be able to do a partial assembly depending on the changes made to the guide. For example, a full assembly is required when you delete results, logic flows, screens, or tables, or after you have changed guide file access or guide options. A partial assembly can be performed when you have changed existing logic flows or screens, or added new logic flows, screens, or results.

When you select the assemble option, the system indicates whether a partial assembly can be performed.

For information on assembling more than one project at a time, see the section Multiple Project Assembly and Swap.

**Placeholders**

When doing guide development, you can reference objects that have not yet been created. Then, during the assembly process, you can specify placeholders are to be generated for objects that are referenced but have not yet been created. For example, if you are designing a screen and want to attach a logic flow to Entry to screen logic, the logic flow does not have to currently exist. If you assemble the guide before creating the logic flow, you can direct the EDGE assembly process to create the object using a predefined placeholder.
The following areas are scanned for non-existent objects:

- Guide Options.
- Hot Keys.
- Logic Flows.
- User Screens.

EDGE can currently create any of the following placeholder objects, if necessary:

- Logic Flows.
- Results.
- Screens.

The object must be specified as a literal; the scan cannot detect missing objects in variables.

If any other object type is missing, an assembly error is generated, or if an object is missing from any area other than one of the listed areas, an assembly error is generated.

The placeholders are predefined and cannot be modified to consist of other values or properties. They are not available for use with the Multiple Project Assembly and Swap.
**Logic Flows**

The logic flow placeholder has the following lines:

-- placeholder Logic Flow Added by EDGE Assembly

ALWAYS ASK with prompt "Placeholder Logic Flow LOGIC.ID Called!"
into @PHOLDER

**Results**

The results placeholder has the following properties:

- The description is Placeholder Result ‘result.ID’.
- No callback arrangements are specified.

**Screens**

The screens placeholder has the following properties:

- The screen type is freeform.
- The screen title is Placeholder Screen screen.ID.
- The last updated date and user are the date of the assembly and the user doing the assembly.
- The screen is defined as a rebound screen.
- Function key 1 has the following text and routes to $RET:
Starting the Assembly

To start an assembly, you can select either the assembly tool on the EDGE Developer toolbar, or the assembly command on the Guide menu. A page similar to the following is displayed:

![Assemble Guide dialog box](image)

**Properties:**

- **Begin full guide assembly**
  - Assembles all the elements of the guide. If you select this command, you are asked to confirm that you want a full assembly:
### Properties: (continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Begin partial guide assembly</td>
<td>Assembles only those elements of the guide that have changed since the last assembly.</td>
</tr>
<tr>
<td>Display assembly errors</td>
<td>Displays a listing of assembly errors. For more information, see the section Display Assembly Errors.</td>
</tr>
<tr>
<td>Force link</td>
<td>Specifies whether the object code should be linked; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td><strong>Checked</strong> The object code is linked. Automatically checked on a full assembly. This is commonly used to force linking when a patch has been applied to your system.</td>
</tr>
<tr>
<td></td>
<td><strong>Cleared</strong> The object code is not linked unless there has been a guide change that requires it (for example, a logic flow has been added or modified). The default is cleared unless a full assembly is required.</td>
</tr>
<tr>
<td>Last guide assembly</td>
<td>The date of the last assembly. If blank, the guide has not been assembled. This property is display only.</td>
</tr>
</tbody>
</table>
**Properties: (continued)**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last guide swap</td>
<td>The date of the last swap. If blank, the guide has not been swapped.</td>
</tr>
<tr>
<td></td>
<td>This property is display only.</td>
</tr>
<tr>
<td>Placeholders</td>
<td>The method of handling non-existent objects; can be one of the following:</td>
</tr>
<tr>
<td></td>
<td>Automatically create placeholders  Non-existent objects are automatically created.</td>
</tr>
<tr>
<td></td>
<td>Do not create placeholders  Non-existent objects are not created and an assembly error is generated.</td>
</tr>
<tr>
<td></td>
<td>Prompt to create placeholders  Each time a non-existent object is encountered, a dialog box similar to the following is displayed:</td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="ETW for Windows" /></td>
</tr>
<tr>
<td>Total errors</td>
<td>The number of errors encountered in the last assembly.</td>
</tr>
<tr>
<td></td>
<td>This property is display only.</td>
</tr>
</tbody>
</table>
Finishing the Assembly

Properties: (continued)

- **Type**: The type of assembly last performed.
  
  This property is display only.

When the assembly has finished, statistics similar to the following are displayed:

If any errors are reported, select the command **Display Assembly Errors** from the Assemble page to display the error listing.
**Display Assembly Errors**

Note: *Errors assembling toolbars are grouped with Options errors.*

To test the guide, select the Testing the Guide command. To display the fields, screens, logics, and results used in the guide, select the List Guide Flow command from the Guide menu.

The Display Assembly Errors command allows you to view or print errors that occurred during assembly.

If you receive errors when you assemble the guide, you should correct them and reassemble the guide before releasing the guide to operations. However, you can test a guide that has assembly errors; EDGE ignores the object with errors.

The report is similar to the following:
The following information is displayed:

**Type**  The component containing the error; can be one of the following:

- **L** Logic flow.
- **O** Guide options.
- **R** Result.
- **S** Screen.
- **T** Table.

**Item**  The name of the item containing the error.

<table>
<thead>
<tr>
<th>TYPE</th>
<th>ITEM</th>
<th>LOCATION</th>
<th>PROBLEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>CALLBACK.Set</td>
<td>LINE 1</td>
<td>Field ID: SCHED.DATE ... not found in guide file: CUST.LIST</td>
</tr>
<tr>
<td>O</td>
<td>OPTIONS</td>
<td>FIRST INB. SCREEN</td>
<td>SCREEN NOT SPECIFIED FOR ROUTE</td>
</tr>
<tr>
<td>R</td>
<td>NO.INT</td>
<td>NEW QUEUE</td>
<td>QUEUE ID: NOT.INT ... DOES NOT EXIST</td>
</tr>
<tr>
<td>S</td>
<td>CUST.INFO</td>
<td>Input Object #5 R 2 Logic Flow: CORR.ACCT ... does not exist. C 70</td>
<td></td>
</tr>
</tbody>
</table>
| S    | LOOKUP     | Command Button #27 Field ID: HEIGHT ... not found in guide file: CUST.LIST R 15 C 44
### List Guide Flow

<table>
<thead>
<tr>
<th>Location</th>
<th>The line number, function key number, or other identifier for the location of the error.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem</td>
<td>A brief description of the error.</td>
</tr>
</tbody>
</table>

The List Guide Flow command displays the names of fields, screens, logics, and results used in the guide. The List Guide Flow command is available from the Guide menu.

The report is similar to the following:
The following information is displayed:

- **T**: The type of component; can be one of the following:
  - **L**: Logic flow.
  - **S**: Screen.

- **Source ID**: The name of the logic flow or screen.

- **Field IDs**: The field IDs used in the logic flow or screen.
<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screen IDs</td>
<td>The screen IDs used in the logic flow or screen.</td>
</tr>
<tr>
<td>Logic IDs</td>
<td>The logic IDs used in the logic flow or screen.</td>
</tr>
<tr>
<td>Result IDs</td>
<td>The result IDs used in the logic flow or screen.</td>
</tr>
<tr>
<td>From Screen</td>
<td>The screen ID of the screen that precedes the component.</td>
</tr>
</tbody>
</table>
Testing the Guide

You should always test the guide before releasing it to operations. You should try to make sure that all the paths through the guide conclude without error. You can test both the development and operations guide in live mode and in test mode. Generally, you should test in the following sequence:

- Development guide Test mode. (No data is written.)
- Development guide Live mode. (Data is written to files.)
- Operations guide Test mode. (No data is written.)
- Operations guide Live mode. (Data is written to files.)

To aid in testing, the following tools are provided:

- **DEBUG verb.**
- Debug mode. This mode is entered by pressing `CTRL+K` from within the guide. Debug mode is described as part of the DEBUG verb.
- **Guide Watcher.**

Before you can test the guide, be sure the following have been set up:
Set your user ID to allow multiple sessions using the Allow multiple sessions. (When you test the guide, EDGE Developer logs into EDGE Client, using your current user ID.)

Specify the path to EDGE Client using the Client Settings property sheet.

To test a guide, you can select either the test tool on the EDGE Developer toolbar or the Test Guide command on the Guide menu. When testing the guide from EDGE Developer, you can specify whether to automatically invoke EDGE Client or to use ASCII mode. You should test using the mode your agents will use. You can also specify Live / Test mode and Development or Operations guide.
Properties:

Guide  The guide to be tested; can be one of the following:

- Development  The guide on which you are currently working.
- Operations  The guide the agents use.

Mode  The test mode to be used; can be one of the following:

- Test  No data is written to files.
- Live  Records are updated at end of guide, and when the verb WRITE-RECORD is used in logic. This includes updates to monitor reports and performance statistics.
### Properties: (continued)

For more information, see the section [Live or Test Mode](#).

<table>
<thead>
<tr>
<th><strong>Client</strong></th>
<th>The EDGE client that will be used for testing; can be one of the following:</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEO</td>
<td>An EDGE Client session is launched for testing. The EDGE Client application used is specified in the EDGE Developer Client Settings.</td>
</tr>
<tr>
<td>ASCII</td>
<td>An ASCII session is used for testing.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Login</strong></th>
<th>The login user ID and password utilized if EDGE Client is the client. If ASCII is the client, the current EDGE Developer user ID is used.</th>
</tr>
</thead>
<tbody>
<tr>
<td>User ID</td>
<td>The user ID to use for EDGE Client testing.</td>
</tr>
<tr>
<td>Password</td>
<td>The password for the user ID used for EDGE Client testing.</td>
</tr>
<tr>
<td>Workstation ID</td>
<td>The workstation ID information for the user ID used for EDGE Client testing.</td>
</tr>
</tbody>
</table>

The default is the current EDGE Developer user ID, password, and workstation ID.

| **Guide watcher** | Specifies whether the guide watcher is used when testing the guide. If ASCII is used, the guide watcher results are logged into a flat file. If EDGE Client is used, the guide watcher results are displayed in a dialog box on the screen. |

Live or Test Mode

You should start your testing in test mode, then finish in live mode. When you test the guide in test mode, no data is written to files. This mode allows you to check field validation settings and other criteria without having to later delete records.

In test mode, check the following:

- Is the first inbound screen displayed when you take an inbound call?
- Is the first outbound screen displayed when you make an outbound call?
- Can you make a call using the command Next Call? Is the call pulled from the appropriate queue based on your callback setting and operator access record?
- Do all function keys route to the proper screens or windows?
- Are call events filed properly on the Call Center screen ($END)?

**Properties: (continued)**

- Guidewatcher trace file
  - The name of the flat file to which to write the ASCII guide watcher trace information. The default is blank. If no filename is specified, the guide watcher trace is not recorded. This property is only enabled if ASCII testing is specified.
• Can you schedule a callback using the Set Callback screen ($CALLBACK)?

• Test all the paths that lead to the filing of a call.

• If you use the View Call History ($CALLHIST) process, is the screen data displayed correctly and does it return you to the original screen?

When you test the guide in live mode, data is gathered and records are updated. When you test the operations guide in live mode, you should use test records; after the testing is complete, clear the test records from the files.

In live mode, check the following:

• If you are updating records, check to make sure the data is written properly to the file. Check the data from ACL.

• Try to manually retrieve records.

• Check file access. Is data being properly written to the assigned files?

• Generate correspondence and project monitor reports. Are the figures accurate?

Caution! When you test the guide using the Test Guide screen, the operations environment is not exactly duplicated, even when
using the operations guide and live mode. From the Test Guide screen, you have access to all queues, and so can access protected queues and queues you have not been given access to. For this reason, it is recommended, especially for new guides, that you test the guide from operations before releasing the guide to your agents.

Guide Watcher

The guide watcher is a debugging tool that allows you to trace the flow of a guide during guide development as well as operations. You can use the guide watcher, for example, if the guide unexpectedly routes to a screen or executes a logic flow in an unanticipated manner.

The guide watcher traces the flow of a guide’s execution and writes the information to a UNIX file if using ASCII operations, or displays the information in a dialog box in EDGE Client. The following information is traced:

- Automatic reads of records.
- Cross-referencing.
- Field validation.
- Function keys.
- Hot keys.
• Logic flows.
• Routing.
• Screens.
• Windows.
• Swapping.

Note: Currently, logic flows called from other logic flows do not display in the guide watcher.

The guide watcher also provides time and date stamps, and identifies the following system information:

• Port.
• Process ID.
• Project name.
• Telephony setup information.
• User ID.

Note: To display the names of fields, screens, logics, and results used in the guide, see the section List Guide Flow. To trace an individual logic flow, use the DEBUG verb.
To activate the guide watcher, check the box. A dialog similar to the following is displayed:

![Guide Watcher Dialog](image)

If using ASCII, you will need to enter a UNIX pathname. Once you check the guide watcher box and specify the guide watcher filename, this becomes the default. You can also specify the filename using the environment variable GUIDE_WATCHER.

**Note:** If ASCII is used, the guide watcher trace log is independent of the standard error log. When the guide watcher is invoked, a message is written to standard error on whether it started successfully or failed to open the guide watcher trace log. With EDGE Client, the data is not written to a file; therefore, this does not apply.
Swapping

EDGE keeps track of two assembled versions of each guide: the version in development and the version in operations. When you assemble the guide, you assemble the development version. When you swap the guide, the version that was the development version becomes the operations version and vice versa. If there are agents on the system during the swap, the operations version becomes available to an agent whenever that agent takes the next call. This means, in fact, there may be two versions of the guide running at the same time.

The source for the guide remains in the development area of EDGE. When you swap, the source matches the operations guide; it does not match the version in development. To get the development guide to match the source, you must reassemble the guide. However, before reassembling, you should test that the operations guide is operating as expected. If it is not, and if you have not assembled since the last time you swapped, you can swap again. This has the effect of reverting to the conditions that existed before the first swap. The old operations guide is again the operations guide and the development guide matches the source.
Caution! If you swap while agents are using the guide, you cannot reassemble the development version until all agents have finished the calls they were on when the guide was swapped and either logged off or taken a new call.

For an illustration of how swapping works, see Figure 24-1. For more information on how EDGE manages the two versions, see the section Technical Notes.

To swap a guide, you can select either the swap tool on the EDGE Developer toolbar or the swap command on the Guide menu.

The following prompt is displayed:

![ETW for Windows](image)

Release the Guide to operators. Are you sure?

OK  Cancel

When the guide has been swapped, the following message is displayed:
After the guide is swapped, the development guide no longer matches your code.
Figure 24-1. Assembly and Swapping
Technical Notes

This section describes how EDGE manages the two versions of the guide and keeps track of components for partial assemblies. It is intended for those who want a more detailed understanding of the assembly and swapping process.

EDGE assembles components in the following areas:

- Files.
- Guide options, including toolbars.
- Logic flows.
- Results.
- Screens.
- Tables.

When you first create a project, EDGE creates an internal file called CHANGED, which it uses to determine the components that need to be assembled. Each time you create or modify a component of the guide, a record is written in the CHANGED file. When you assemble the guide, each record in the guide is marked as being assembled for that version of the guide. For example, when you start development, a record is written in the CHANGED file for each component you create. When you assemble, each of the records is marked with a Y in attribute 1, indicating that it was assembled for version 1 of the guide. If you make a changed to a component, a new record is writ-
When you swap, the operations guide becomes guide 1 and the guide for development is now guide 2. The next time you perform a partial assembly, EDGE checks the CHANGED file and assembles every component that is not marked as having been assembled for version 2. It puts a Y in attribute 2 in the corresponding record in the CHANGED file. After this assembly, the two guides match.

The guide number in each agent’s PORTISON record (attribute 7) is updated with the new guide number (1 or 2) when they take a new call and are switched to the swapped version of the guide. EDGE checks this field in the PORTISON record when each assembly occurs, and prevents an assembly if there are agents still working on the guide number you are assembling.

Until you reassemble after a swap, the development guide does not match the source code; it is still the old operations version. This allows you, for example, to test the new guide in live operations, and if it is not working as intended, to swap the guide again, which puts the old operations guide back in place. Your agents can continue to use the old operations guide while you work on the development guide.

An EDGE assembly works as follows:
1. If a partial assembly is being performed, a record is written in the CHANGED file for each error in the ERRORS file. This ensures that components with errors are reassembled.

2. If a full assembly is being performed, all the necessary object files are cleared.

3. Guide files are validated and processed to identify any field changes. All the screens are checked for these fields. A record is written in the CHANGED file for each screen with one of the changed fields to ensure the screen is reassembled.

4. Guide options are validated.

5. Results, tables, and logic flows are validated and assembled.

6. The guide is linked to form a new executable. (The EDGE executable is called XGUIDE.)

7. Screens are validated and assembled.

8. During each step, a record is written to the ERRORS file for any error found.

9. At the completion of the assembly process, EDGE marks the CHANGED records associated with the assembled components to show that the component has been assembled for the current version (1 or 2) of the guide. If the CHANGED record shows that the component has been assembled for both versions of the guide, the record is deleted.
Periodic Maintenance

The following is a list of maintenance tasks that are described in other chapters.

**General Cleanup**
- Clean EDGE/tmp.
- Clear Processed ACTZs.
- Compact File.
- Database Maintenance.
- Delete Call IDs (from queues).
- Delete Records (clear files).
- DELETELIST (ACL command to delete selection lists).

**EDGE Logs**
- Clear Ad Hoc Reporting Log.
- Clear the Task Scheduler Log.
- Clear Queue Problem Log.

**Reporting Data**
- Initialize the Global Monitor.
- Clear Project to Date Monitor.
This appendix lists the ASCII Codes 1-127.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>&lt;Ctrl-@&gt;</td>
</tr>
<tr>
<td>1</td>
<td>&lt;Ctrl-A&gt;</td>
</tr>
<tr>
<td>2</td>
<td>&lt;Ctrl-B&gt;</td>
</tr>
<tr>
<td>3</td>
<td>&lt;Ctrl-C&gt;</td>
</tr>
<tr>
<td>4</td>
<td>&lt;Ctrl-D&gt;</td>
</tr>
<tr>
<td>5</td>
<td>&lt;Ctrl-E&gt;</td>
</tr>
<tr>
<td>6</td>
<td>&lt;Ctrl-F&gt;</td>
</tr>
<tr>
<td>7</td>
<td>&lt;Ctrl-G&gt;</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>8</td>
<td>&lt;Ctrl-H&gt;</td>
</tr>
<tr>
<td>9</td>
<td>&lt;Ctrl-I&gt;</td>
</tr>
<tr>
<td>10</td>
<td>&lt;Ctrl-J&gt;</td>
</tr>
<tr>
<td>11</td>
<td>&lt;Ctrl-K&gt;</td>
</tr>
<tr>
<td>12</td>
<td>&lt;Ctrl-L&gt;</td>
</tr>
<tr>
<td>13</td>
<td>&lt;Ctrl-M&gt;</td>
</tr>
<tr>
<td>14</td>
<td>&lt;Ctrl-N&gt;</td>
</tr>
<tr>
<td>15</td>
<td>&lt;Ctrl-O&gt;</td>
</tr>
<tr>
<td>16</td>
<td>&lt;Ctrl-P&gt;</td>
</tr>
<tr>
<td>17</td>
<td>&lt;Ctrl-Q&gt;</td>
</tr>
<tr>
<td>18</td>
<td>&lt;Ctrl-R&gt;</td>
</tr>
<tr>
<td>19</td>
<td>&lt;Ctrl-S&gt;</td>
</tr>
<tr>
<td>20</td>
<td>&lt;Ctrl-T&gt;</td>
</tr>
<tr>
<td>21</td>
<td>&lt;Ctrl-U&gt;</td>
</tr>
<tr>
<td>22</td>
<td>&lt;Ctrl-V&gt;</td>
</tr>
<tr>
<td>23</td>
<td>&lt;Ctrl-W&gt;</td>
</tr>
<tr>
<td>24</td>
<td>&lt;Ctrl-X&gt;</td>
</tr>
<tr>
<td>25</td>
<td>&lt;Ctrl-Y&gt;</td>
</tr>
<tr>
<td>26</td>
<td>&lt;Ctrl-Z&gt;</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>27</td>
<td><code>&lt;ESC&gt;</code></td>
</tr>
<tr>
<td>28</td>
<td><code>&lt;Ctrl-\&gt;</code></td>
</tr>
<tr>
<td>29</td>
<td><code>&lt;Ctrl-[]&gt;</code></td>
</tr>
<tr>
<td>30</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>2</td>
</tr>
<tr>
<td>32</td>
<td>(space)</td>
</tr>
<tr>
<td>33</td>
<td><code>!</code></td>
</tr>
<tr>
<td>34</td>
<td><code>&quot;</code></td>
</tr>
<tr>
<td>35</td>
<td><code>#</code></td>
</tr>
<tr>
<td>36</td>
<td><code>$</code></td>
</tr>
<tr>
<td>37</td>
<td><code>%</code></td>
</tr>
<tr>
<td>38</td>
<td><code>&amp;</code></td>
</tr>
<tr>
<td>39</td>
<td><code>'</code></td>
</tr>
<tr>
<td>40</td>
<td>`(</td>
</tr>
<tr>
<td>41</td>
<td><code>)</code></td>
</tr>
<tr>
<td>42</td>
<td><code>*</code></td>
</tr>
<tr>
<td>43</td>
<td><code>+</code></td>
</tr>
<tr>
<td>44</td>
<td><code>,</code></td>
</tr>
<tr>
<td>45</td>
<td><code>-</code></td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>46</td>
<td>.</td>
</tr>
<tr>
<td>47</td>
<td>/</td>
</tr>
<tr>
<td>48</td>
<td>0</td>
</tr>
<tr>
<td>49</td>
<td>1</td>
</tr>
<tr>
<td>50</td>
<td>2</td>
</tr>
<tr>
<td>51</td>
<td>3</td>
</tr>
<tr>
<td>52</td>
<td>4</td>
</tr>
<tr>
<td>53</td>
<td>5</td>
</tr>
<tr>
<td>54</td>
<td>6</td>
</tr>
<tr>
<td>55</td>
<td>7</td>
</tr>
<tr>
<td>56</td>
<td>8</td>
</tr>
<tr>
<td>57</td>
<td>9</td>
</tr>
<tr>
<td>58</td>
<td>:</td>
</tr>
<tr>
<td>59</td>
<td>;</td>
</tr>
<tr>
<td>60</td>
<td>&lt;</td>
</tr>
<tr>
<td>61</td>
<td>=</td>
</tr>
<tr>
<td>62</td>
<td>&gt;</td>
</tr>
<tr>
<td>63</td>
<td>?</td>
</tr>
<tr>
<td>64</td>
<td>@</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>65</td>
<td>A</td>
</tr>
<tr>
<td>66</td>
<td>B</td>
</tr>
<tr>
<td>67</td>
<td>C</td>
</tr>
<tr>
<td>68</td>
<td>D</td>
</tr>
<tr>
<td>69</td>
<td>E</td>
</tr>
<tr>
<td>70</td>
<td>F</td>
</tr>
<tr>
<td>71</td>
<td>G</td>
</tr>
<tr>
<td>72</td>
<td>H</td>
</tr>
<tr>
<td>73</td>
<td>I</td>
</tr>
<tr>
<td>74</td>
<td>J</td>
</tr>
<tr>
<td>75</td>
<td>K</td>
</tr>
<tr>
<td>76</td>
<td>L</td>
</tr>
<tr>
<td>77</td>
<td>M</td>
</tr>
<tr>
<td>78</td>
<td>N</td>
</tr>
<tr>
<td>79</td>
<td>O</td>
</tr>
<tr>
<td>80</td>
<td>P</td>
</tr>
<tr>
<td>81</td>
<td>Q</td>
</tr>
<tr>
<td>82</td>
<td>R</td>
</tr>
<tr>
<td>83</td>
<td>S</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>84</td>
<td>T</td>
</tr>
<tr>
<td>85</td>
<td>U</td>
</tr>
<tr>
<td>86</td>
<td>V</td>
</tr>
<tr>
<td>87</td>
<td>W</td>
</tr>
<tr>
<td>88</td>
<td>X</td>
</tr>
<tr>
<td>89</td>
<td>Y</td>
</tr>
<tr>
<td>90</td>
<td>Z</td>
</tr>
<tr>
<td>91</td>
<td>[</td>
</tr>
<tr>
<td>92</td>
<td>\</td>
</tr>
<tr>
<td>93</td>
<td>]</td>
</tr>
<tr>
<td>94</td>
<td>^</td>
</tr>
<tr>
<td>95</td>
<td>_</td>
</tr>
<tr>
<td>96</td>
<td>'</td>
</tr>
<tr>
<td>97</td>
<td>a</td>
</tr>
<tr>
<td>98</td>
<td>b</td>
</tr>
<tr>
<td>99</td>
<td>c</td>
</tr>
<tr>
<td>100</td>
<td>d</td>
</tr>
<tr>
<td>101</td>
<td>e</td>
</tr>
<tr>
<td>102</td>
<td>f</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>103</td>
<td>g</td>
</tr>
<tr>
<td>104</td>
<td>h</td>
</tr>
<tr>
<td>105</td>
<td>i</td>
</tr>
<tr>
<td>106</td>
<td>j</td>
</tr>
<tr>
<td>107</td>
<td>k</td>
</tr>
<tr>
<td>108</td>
<td>l</td>
</tr>
<tr>
<td>109</td>
<td>m</td>
</tr>
<tr>
<td>110</td>
<td>n</td>
</tr>
<tr>
<td>111</td>
<td>o</td>
</tr>
<tr>
<td>112</td>
<td>p</td>
</tr>
<tr>
<td>113</td>
<td>q</td>
</tr>
<tr>
<td>114</td>
<td>r</td>
</tr>
<tr>
<td>115</td>
<td>s</td>
</tr>
<tr>
<td>116</td>
<td>t</td>
</tr>
<tr>
<td>117</td>
<td>u</td>
</tr>
<tr>
<td>118</td>
<td>v</td>
</tr>
<tr>
<td>119</td>
<td>w</td>
</tr>
<tr>
<td>120</td>
<td>x</td>
</tr>
<tr>
<td>121</td>
<td>y</td>
</tr>
<tr>
<td>Value</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>-------------</td>
</tr>
<tr>
<td>122</td>
<td>z</td>
</tr>
<tr>
<td>123</td>
<td>{</td>
</tr>
<tr>
<td>124</td>
<td></td>
</tr>
<tr>
<td>125</td>
<td>}</td>
</tr>
<tr>
<td>126</td>
<td>~</td>
</tr>
<tr>
<td>127</td>
<td>(del)</td>
</tr>
</tbody>
</table>
Overview

The `CALCULATE` verb can be used to create the following types of complex functions:

- Arithmetic Functions.
- Array Functions.
- Bessel Functions.
- Financial Functions.
- Logarithmic Functions.
- Statistical Functions.
- String Functions.
- Trigonometric Functions.

This appendix describes each function, shows the syntax, and provides examples.
The EDGE Developer interface includes a screen which resembles a calculator. The calculator allows you to create an expression by selecting functions, fields, numbers, and mathematical operators. The EDGE Developer interface also includes editing and testing features. You can use the **CALCULATE** functions in an expression by selecting the function from the applicable drop down box on the keypad or by typing the abbreviation of the function in the display area.
Arithmetic Functions

The following arithmetic functions are available:

- ABS (Absolute Value)
- DIST (Euclidean Distance)
- EXP (Exponential)
- INT (Integer Portion)
- MOD (Remainder)
- POW (Power)
- ROUND (Rounded Value)
- SQRT (Square Root)

**ABS (Absolute Value)**

ABS (x)

The Absolute Value function returns the absolute value of its argument. That is, if x is greater than or equal to zero, the function returns x. Otherwise, it returns x multiplied by negative one.

If the function is passed an array, it returns an array with each element replaced by its absolute value.
## CALCULATE Functions

### Arithmetic Functions

#### Examples

- \( \text{ABS} \left( -5 \right) = 5 \)
- \( \text{ABS} \left( -12.3 \right) = 12.3 \)
- \( \text{ABS} \left( 7 \right) = 7 \)
- \( \text{ABS} \left( \left[ -5, 4.3, -12, 19 \right] \right) = \left[ 5, 4.3, 12, 19 \right] \)

#### See Also

- \text{INT (Integer Portion)} and \text{ROUND (Rounded Value)}.

### DIST (Euclidean Distance)

**DISTANCE \( (x, y) \)**

The DIST function calculates the Euclidean distance from the origin to the point at coordinates \( x, y \). The distance is equivalent to the following:

- \( \text{SQRT}(x * x + y * y) \).

**Examples**

- \( \text{DISTANCE( 3, 4 )} = 5.0 \)
- \( \text{DISTANCE( 0, 0 )} = 0.0 \)
- \( \text{DISTANCE( [3,4,5], 4 )} = [5.0,5.65685,6.40312] \)

### EXP (Exponential)

**EXP \( (x) \)**

- \( \text{EXP} \left( x \right) \)
The EXP function computes the value of e (approximately 2.7182828) raised to the power of $x$.

**Examples**

- $\text{EXP(0)} = 1.0$
- $\text{EXP(1)} = 2.7182828...$

**See Also**

POW (Power), LN (Natural Logarithm), and LOG (Logarithm).

---

**INT (Integer Portion)**

The INT function returns the integer portion of $x$.

**Examples**

- $\text{INT(2.4)} = 2.0$
- $\text{INT(-14.5)} = -14.0$
- $\text{INT(3.8)} = 3.0$
- $\text{INT([2.4, -14.5, 3.8 ])} = [2.0, -14.0, 3.0]$

**See Also**

ABS (Absolute Value) and ROUND (Rounded Value).
MOD (Remainder)

MOD (x, y)

The MOD function returns the remainder of dividing x by y. The remainder is also referred to as the modulus. The value for y must be greater than or equal to zero.

Examples

MOD( 7, 3 ) = 1.0
MOD( 10, 2 ) = 0.0
MOD( 29, 5 ) = 4.0
MOD( [ 7, 10, 29 ], 5 ) = [ 2.0, 0.0, 4.0 ]
MOD( 29, [ 3, 2, 5 ] ) = [ 2.0, 1.0, 4.0 ]
MOD( [7,10,29], [3,2,5] ) = [ 1.0, 0.0, 4.0 ]

POW (Power)

POW (x, y)

The POW function raises x to the power of y.

Examples

POW( 5, 2 ) = 25.0
POW( 5, [0, 1, 2, 3] ) = [ 1.0, 5.0, 25.0, 125.0 ]
ROUND (Rounded Value)

The ROUND function rounds `num` to the specified number of decimal places. If the digit to the right of the desired significant digit is five or greater, the number is rounded up, otherwise the number is rounded down. The value for `num` must be greater than or equal to zero.

**Examples**

```
ROUND( 1.23456, 2 ) = 1.23
ROUND( 2.34567, 2 ) = 2.35
ROUND([ 1.2345, 2.3456 ], 2 ) = [ 1.23, 2.34 ]
```

**See Also**

`INT (Integer Portion)` and `ABS (Absolute Value)`.

SQRT (Square Root)

The SQRT function returns the square root of its argument.

**See Also**

`SQRT (Square Root)` and `EXP (Exponential)`.
Examples

\[
\text{SQRT( 4 ) = 2.0}
\]
\[
\text{SQRT( 25 ) = 5.0}
\]
\[
\text{SQRT( [4, 25, 81] ) = [ 2.0, 5.0, 9.0 ]}
\]

See Also

POW (Power).
Array Functions

The Array functions provide a means for manipulating arrays. The following array functions are available:

- ARRAY(Conversion)
- ELEMENT (Array Element)
- LOOKUP (Lookup Table Value)
- STRING (Conversion)
- SUBARRAY (Extract Arrays)

ARRAY(Conversion)

The ARRAY function converts a string to an array. Conversely, the STRING function can be used to convert an array to a string. By default, any expression that results in an array is stored as a multivalued field.

Examples

ARRAY( "[1,2,3]") = 1\[2\]3

See Also

STRING (Conversion) and SUBARRAY (Extract Arrays).
**ELEMENT (Array Element)**

ELEMENT (array, number)

The Array Element function returns a single element of an array. The first argument must be an array. The second argument must be a number between 1 and the number of elements in the array.

**Examples**

- ELEMENT([ 1, 7, 19, 20, 13, 5 ], 3) = 19.0
- ELEMENT([ [ 1, 2, 3], [4, 5, 6], 7 ], 2) = [4, 5, 6]

**See Also**

LOOKUP (Lookup Table Value).

**LOOKUP (Lookup Table Value)**

LOOKUP (table, mode, value)

The Lookup Table Value function returns the row whose index value is greater than or equal to the lookup value. If the lookup value is greater than all index values, the last row is returned.

**Mode**

Specifies how the lookup values for the row are compared against the value passed; can be one of the following:

- 0  A numeric comparison is made.
- 1  A case-sensitive string comparison is made.
- 2  A non case-sensitive string comparison is made.
Examples

Suppose table CODES contains the following:

<table>
<thead>
<tr>
<th>Index</th>
<th>Value</th>
<th>Row</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>[ 2, 4,18, 23 ]</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>[ 13, 29,18, 7 ]</td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>[ 89, 47,23,100 ]</td>
<td></td>
</tr>
<tr>
<td>9999</td>
<td>[ 23, 99, 0,19 ]</td>
<td></td>
</tr>
</tbody>
</table>

```
LOOKUP( "CODES", 0, "35" ) = [ 13, 29, 18, 7 ]
LOOKUP( "CODES", 0, "1000") = [ 23, 99, 0, 19 ]
```

See Also

ELEMENT (Array Element).

STRING (Conversion)

STRING ( array )
The STRING function converts an array into a string. All references to multivalued fields are first converted into an array. The STRING function allows you to convert multivalued fields into a string value.

**Examples**

```
STRING([ 1, 2, 3, 4 ] ) = "[1,2,3,4]"
STRING([101,102,111] ) = "[101,102,111]"
```

**See Also**

ARRAY(Conversion).

---

**SUBARRAY**

(Extract Arrays)

The SUBARRAY function returns a portion of `array`. The new array should contain `count` elements. The first element in the new array will correspond to the `start` element in the original array.

**Examples**

```
SUBARRAY( [ 1, 2, 3, 4, 5 ], 2, 3 ) = [ 2, 3, 4 ]
SUBARRAY( [ 1, 2, 3 ], 2, 3 ) = [ 2, 3 ]
```
See Also

ARRAY(Conversion).
**Bessel Functions**

Bessel functions are used to compute wave variables, primarily in the field of communications. The following Bessel functions are available:

- **Bessel Functions of First Kind**
  - BESSELj0 \( (x) \)
  - BESSELj1 \( (x) \)
  - BESSELjn \( (x) \)

  The Bessel functions of the first kind compute the value of the Bessel functions of the first kind, or order 0, 1, and \( n \), respectively.

- **Bessel Functions of Second Kind**
  - BESSELY0 \( (x) \)
  - BESSELY1 \( (x) \)
  - BESSELYn \( (x) \)

  The Bessel functions of the second kind compute the value of the Bessel functions of the second kind, or order 0, 1, and \( n \), respectively.
Financial Functions

Financial functions provide various financial calculations. The following financial functions are available:

- ECIR (Effective Continuous Interest Rate)
- FV (Future Value)
- IP (Interest Paid)
- PAYMENT (Loan Payment)
- PP (Principal Paid)
- PV (Present Value)
- RB (Remaining Balance)

ECIR (nominal_rate)

The Effective Continuous Interest Rate function calculates the effective continuous interest rate. This function is equivalent to the expression:

\[(\text{EXP}(\text{nom}_\text{rate} / 100) - 1) \times 100\]
**Example**

If you borrowed $1000.00 against a credit card that had an 18% nominal interest rate compounded continuously, and did not make any payments for one year, then your balance after one year would be:

\[ 1000.00 \times (1 + \frac{\text{ECIR}(18.0)}{100}) = 1197.22 \]

**See Also**

FV (Future Value), PV (Present Value), and PAYMENT (Loan Payment).

**FV (Future Value)**

FV ( \text{present\_value}, \text{interest\_per\_period}, \text{num\_periods} )

The Future Value function calculates the future value of a loan given its present value, interest rate per period, and the number of periods. This function is equivalent to the expression:

\[ \text{present\_value} \times \text{POW}(1 + \text{interest\_per\_period}, \text{num\_periods}) \]

**Example**

The following calculation could be used to calculate the future value of placing $100 in a savings account with an annual interest rate of 5% compounded monthly for 8 years.

\[ \text{FV}(100.00, 0.05 / 12, 12 \times 8) \]
**IP (Interest Paid)**

**See Also**

PV (Present Value), ECIR (Effective Continuous Interest Rate), and PAYMENT (Loan Payment).

IP (principle, interest_rate_per_period, num_periods)

The Interest Paid function calculates the total interest paid over the entire life of a loan. This function is equivalent to the expression:

\[
\text{PAYMENT} (\text{principle}, \text{interest}_\text{rate}_\text{per}_\text{period}, \text{num}_\text{periods}) \\
\times \text{num}_\text{periods} - \text{principle}
\]

**Example**

If you took out a four year loan of 10,000.00 with an annual interest rate of 8.0% requiring monthly payments then the total interest that you pay would be:

\[
\text{IP}(10000.00, 0.08 / 12, 4 \times 12)
\]

**See Also**

PAYMENT (Loan Payment), PP (Principal Paid), and RB (Remaining Balance).
**PAYMENT (Loan Payment)**

PAYMENT (principle, interest_per_period, num_periods)

The Payment function calculates the payment required for a loan given the principle, interest per period, and the number of periods. This function is equivalent to the expression:

```
principle / ( ( 1 - POW( 1+interest_per_period, -num_periods ) ) / interest_per_period )
```

**Example**

The following calculation could be used to determine what the monthly payment would be on a four year simple interest loan of $5000.00 which has an annual interest rate of 8.5%.

```
PAYMENT( 5000.00, 0.085 / 12, 4 * 12 )
```

**See Also**

FV (Future Value), PV (Present Value), and ECIR (Effective Continuous Interest Rate).

**PP (Principal Paid)**

PP (current_balance, interest_rate_per_period, payment)

The Principal Paid function calculates the amount of a loan payment that is paid towards the principal. This function is equivalent to the expression:
payment - current_balance * interest_rate_per_period

**Example**

If you have an outstanding loan of $1100.00 with an annual interest rate of 9% and you make a monthly payment of $100.00 then the amount of your payment that gets applied toward your outstanding principal would be:

\[ PP(1100.00, 0.09 / 12, 100.00) = 91.75 \]

**See Also**

IP (Interest Paid), PAYMENT (Loan Payment), and RB (Remaining Balance).

**PV (Present Value)**

\[ PV\ (future\_value,\ interest\_per\_period,\ num\_periods) \]

The Present Value function calculates the present value required for a loan given the desired future value, interest rate per period, and the number of periods. This function is equivalent to the expression:

\[ future\_value / POW(1 + interest\_per\_period, num\_periods) \]
**Example**

The following calculation could be used to calculate the amount required to place in a savings account that has an annual interest rate of 5% compounded monthly to obtain a future value of $5000.00 in five years.

\[
PV( 5000.00, 0.05 / 12, 12 \times 5 )
\]

**See Also**

FV (Future Value), ECIR (Effective Continuous Interest Rate), and PAYMENT (Loan Payment).

**RB (Remaining Balance)**

RB ( current_balance, interest_rate_per_period, payment )

The Remaining Balance function calculates the remaining balance on a loan after making a regular payment. This function is equivalent to the expression:

\[
\text{current_balance} - \text{PP( current_balance, interest_rate_per_period, payment })
\]
**Example**

If you have an outstanding loan of $1100 with an annual interest rate of 9% and you make a regular monthly payment of $100.00 for one year, the remaining loan balance would be:

\[ \text{RB(1100.00, 0.09 / 12, 100.00) = 1008.25} \]

**See Also**

PP (Principal Paid), IP (Interest Paid), and PAYMENT (Loan Payment).
Logarithmic Functions

The following logarithmic functions are available:

- `GAMMA (Log Gamma)`
- `LN (Natural Logarithm)`
- `LOG (Logarithm)`

**GAMMA (Log Gamma)**

`GAMMA (x)`

The Gamma function returns the log gamma of its argument. The log gamma function is equivalent to the natural log of the absolute value of the gamma function evaluated at `x`. The gamma function is defined as follows:

\[
\int_0^\infty e^{-t}t^{x-1}dt
\]

**See Also**

`LN (Natural Logarithm)` and `ABS (Absolute Value)`. 

**LN (Natural Logarithm)**

`LN (x)`
The Natural Logarithm function returns the natural log (log base $e$) of the argument passed. The natural log is undefined for arguments less than or equal to zero.

**Examples**

- $\text{LN}(2.7182828)=1.0$
- $\text{LN}(7.389056099)=2.0$
- $\text{LN}([2.7182828,7.389056099])=[1.0,2.0]$

**See Also**

- LOG (Logarithm) and EXP (Exponential).

**LOG (Logarithm)**

The Logarithm function returns the common log (log base 10) of the argument passed. The log function is undefined for arguments less than or equal to zero.

**Examples**

- $\text{LOG}(100)=2.0$
- $\text{LOG}(10)=1.0$
- $\text{LOG}([100,10])=[2.0,1.0]$
See Also

LN (Natural Logarithm) and EXP (Exponential).
Statistical Functions

The following statistical functions are available:

- AVG (Average Arguments)
- COUNT (Count Arguments)
- STD (Standard Deviation)
- SUM (Sum Arguments)

**AVG (Average Arguments)**

AVG (n1, n2, n3, ...)

The Average Arguments function returns the arithmetic mean of all of its arguments. Array arguments are not averaged as one argument. This function is equivalent to:

\[
\text{SUM}(n1, n2, n3, ...) / \text{COUNT}(n1, n2, n3, ...).
\]

**Examples**

AVG(1, 2, 3, 4, 5) = 3.0
AVG(3, 4, [5, 6, 7]) = 5.0
AVG([3, 4], [5, 6, 7]) = 5.0

**See Also**

SUM (Sum Arguments) and COUNT (Count Arguments).
**COUNT (Count Arguments)**

COUNT (n1, n2, n3, ...)

The Count Arguments function returns a count of all of its arguments. Array arguments do not count as one argument. Instead, an array argument is counted as the number of elements in the array.

**Examples**

COUNT(3,4,5,6,7) = 5.0  
COUNT([3,4,5,6,7]) = 5.0  
COUNT(3,4,[5,6,7]) = 5.0

**See Also**

AVG (Average Arguments) and SUM (Sum Arguments).

**STD (Standard Deviation)**

STD (n1, n2, n3, ...)

The Standard Deviation function returns the standard deviation of all the arguments, which is equal to the square root of the arithmetic mean of the squares of the deviations from the arithmetic mean.
Figure Z-1.  **Standard Deviation Formula**

Figure Z-1 displays the standard deviation formula based on the following:

\[
\text{result} = \sqrt{\frac{\sum_{n=1}^{\text{count}} x_n^2 - \left(\frac{\sum_{n=1}^{\text{count}} x_n}{\text{count}}\right)^2}{\text{count} - 1}}
\]

- **Count** = Number of elements in array \( x \).
- **\( x_n \)** = Element number \( n \) in array \( x \).
- **Result** = Sample data standard deviation.

To return the population data standard deviation use the standard deviation formula and adjust by the formula shown in Figure Z-2.

\[
\text{result}_p = \text{result} \times \sqrt{\frac{\text{count} - 1}{\text{count}}}
\]

**Figure Z-2.  Population Adjustment Formula**

- **Count** = Number of elements in array \( x \).
CALCULATE Functions

**RESULT** = Sample data standard deviation.
**Resultp** = Population data standard deviation.

**Examples**

- STD(2,4,6,8) = 2.5819...
- STD(3,4,5,6) = 1.03279...
- STD(100,150,200,250) = 64.5497...

**SUM (Sum Arguments)**

The Sum Arguments function returns the sum of all of its arguments. The sum of an array is the sum of its elements.

**Examples**

- SUM( 1,2,3,4,5 ) = 15.0
- SUM( 1,2,[3,4,5] ) = 15.0
- SUM( 1,2,SUM( 3,4,5) ) = 15.0

**See Also**

AVG (Average Arguments) and COUNT (Count Arguments).
String Functions

The following string functions are available:

- CONCAT (Concatenate)
- STRLEN (String Length)
- SUBSTR (Substring)

Note: String literals passed to the CALCULATE verb are limited to 2047 characters. String literals returned from the CALCULATE verb are not limited in length.

**CONCAT** (Concatenate)

CONCAT( str1, str2, ... )

The CONCAT function concatenates two or more strings into a single string. This function always returns a string.

**Examples**

CONCAT( "one", "two", "three" ) = "onetwothree"

CONCAT( [ "a", "b", "c" ], "d" ) = "abcd"

CONCAT( "THIS VALUE IS ", 'T-TBL,VAL' ) = "THIS VALUE IS RED" (where the value in T-TBL at the position number in @VAL = RED)
**STRLEN (String Length)**

The STRLEN function returns the length of the passed string.

**Examples**

```
STRLEN( "test" ) = 4
STRLEN( "" ) = 0
```

**See Also**

SUBSTR (Substring) and CONCAT (Concatenate).

---

**SUBSTR (Substring)**

The SUBSTR function returns a substring of the passed string. The offset parameter specifies where in the string the substring should begin. The length parameter specifies the length of the resulting string. If offset is longer than the length of the string, a null string is returned. If length + offset is passed the end of the string, the resulting string is truncated.

**See Also**

SUBSTR (Substring) and STRLEN (String Length).
Examples

SUBSTR("This is a test", 6, 4) = "is a"

SUBSTR("This is a test", 11, 6) = "test"

See Also

CONCAT (Concatenate) and STRLEN (String Length).
Trigonometric Functions

The following trigonometric functions are available:

- **ARCCOS (Arc Cosine)**
- **ARCSIN (Arc Sine)**
- **ARCTAN (Arc Tangent)**
- **COS (Cosine)**
- **COSH (Hyperbolic Cosine)**
- **SIN (Sine)**
- **SINH (Hyperbolic Sine)**
- **TAN (Tangent)**
- **TANH (Hyperbolic Tangent)**

**ARCCOS (Arc Cosine)**

ARCCOS (x)

The Arc Cosine function returns the arc cosine of its radian argument. The arc cosine function returns a number whose cosine is x.

This function is only defined for values between negative one (-1) and one (1).
**ARCSIN (Arc Sine)**

The Arc Sine function returns the arc sine of its radian argument. The arc sine function returns a number whose sine is $x$. This function is only defined for values between negative one (-1) and one (1).

**Examples**

- \( \text{ARCSIN}(1.0) = 1.570796327 \)
- \( \text{ARCSIN}(0.0) = 0.0 \)
- \( \text{ARCSIN}([-1.0]) = [1.570796327, 0.0, 3.141592654] \)

**ARCTAN (Arc Tangent)**

The Arc Tangent function returns the arc tangent of its radian argument, which is a number whose tangent is $x$.

**Examples**

- \( \text{ARCTAN}(1.0) = 1.570796327 \)
- \( \text{ARCTAN}(0.0) = 0.0 \)
- \( \text{ARCTAN}([1, 0]) = [1.570796327, 0.0] \)
**CALCULATE Functions**

**Examples**

ARCTAN( 0.0 ) = 0.0
ARCTAN( 10000.0 ) = 1.570696
ARCTAN( -10000 ) = -1.570696
ARCTAN( [ -10000, 0, 10000 ] ) = [ -1.570696, 0.0, 1.570696 ]

**COS (Cosine)**

The Cosine function returns the trigonometric cosine of its radian argument.

**Examples**

COS( 3.141592654 ) = -1.0 /* cos ( PI ) */
COS( 1.570796327 ) = 0.0 /* cos ( PI/2 ) */
COS( 0 ) = 1.0
COS( [ 0, 3.141592654 ] ) = [ 1.0, 1.0 ]

**COSH (Hyperbolic Cosine)**

The Hyperbolic Cosine function returns the hyperbolic cosine of its radian argument. The hyperbolic cosine function is defined as

\[ \cosh(x) = \frac{e^x + e^{-x}}{2}. \]
**CALCULATE Functions**

**Examples**

COSH(-1.0) = 1.543081
COSH(0.0) = 1.0
COSH(1.0) = 1.543081
COSH([0, 1]) = [1.0, 1.543081]

**SIN (Sine)**

SIN(x)

The Sine function returns the trigonometric sine of its radian argument.

**Examples**

SIN(3.141592654) = 0.0 /* sin(PI) */
SIN(1.570796327) = 1.0 /* sin(PI/2) */
SIN(0) = 0.0
SIN([0, 0, 1.570796327]) = [0.0, 0.0, 1.0]

**SINH (Hyperbolic Sine)**

SINH(x)

The Hyperbolic Sine function returns the hyperbolic sine of its radian argument. The hyperbolic sine function is defined as

\[(e^x - e^{-x}) / 2.\]
**TAN (Tangent)**

The Tangent function returns the trigonometric tangent of its radian argument.

**Examples**

\[
\begin{align*}
\text{TAN}(-1.0) &= -1.175201 \\
\text{TAN}(0.0) &= 0.0 \\
\text{TAN}(1.0) &= 1.175201 \\
\text{TAN}([0, 1]) &= [0.0, 1.175201]
\end{align*}
\]

**TANH (Hyperbolic Tangent)**

The Hyperbolic Tangent function returns the hyperbolic tangent of its radian argument. The hyperbolic tangent function is defined as \[ \text{SINH}(x) / \text{COSH}(x). \]
Examples

TANH(-1.0) = -0.761594
TANH(0.0) = 0.0
TANH(1.0) = 0.761594
Format Codes

Overview

Format codes are used with the FORMAT-INPUT and FORMAT-OUTPUT verbs to format date, time, and numeric values from internal to external format and from external to internal format.
Format Codes

Format codes are organized by the following categories and described in Table AA-1 through Table AA-6.

- Date Codes.
- Group Extract Codes.
- Numeric Codes.
- Port/Project Code.
- Text Codes.
Time Codes.

**Table AA-1. Date Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>FORMAT-INPUT</td>
<td>Converts date from external format to internal format. Any legal external format may be</td>
<td>April 1 1996 Ô 10319</td>
</tr>
<tr>
<td></td>
<td></td>
<td>input.</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to a date with an abbreviated month name. Using the D code</td>
<td>10319 Ô Apr 01 1996</td>
</tr>
<tr>
<td></td>
<td></td>
<td>with the FORMAT-OUTPUT verb has the same effect as the D4 code, which is described below</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>under Dn.</td>
<td></td>
</tr>
<tr>
<td>DD</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the numeric day of the month.</td>
<td>10319 Ô 01</td>
</tr>
<tr>
<td>DM</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the numeric month.</td>
<td>10319 Ô 04</td>
</tr>
<tr>
<td>DMA</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the alphabetic month name.</td>
<td>10319 Ô April</td>
</tr>
</tbody>
</table>
### Table AA-1. Date Codes (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dn</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to a date with an abbreviated month name and a year of ( n ) (1-4) digits.</td>
<td>For code D2: 10319 Ô Apr 01 96</td>
</tr>
<tr>
<td>Dns</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to a date with a numeric month, a year of ( n ) (1-4) digits, and a separator ( s ).</td>
<td>For code D2/: 10319 Ô 04/01/96</td>
</tr>
<tr>
<td>DnY</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the numeric year of ( n ) (1-4) digits.</td>
<td>For code D2Y: 10319 Ô 96</td>
</tr>
<tr>
<td>DQ</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the quarter number (1-4).</td>
<td>10319 Ô2</td>
</tr>
<tr>
<td>DW</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the numeric day of the week (1=Mon, 7=Sun, etc.).</td>
<td>10319 Ô 1</td>
</tr>
<tr>
<td>DWA</td>
<td>FORMAT-OUTPUT</td>
<td>Converts date from internal format to the full alphabetic day of the week.</td>
<td>10319 Ô Monday</td>
</tr>
<tr>
<td>Code</td>
<td>Verb</td>
<td>Description</td>
<td>Example</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------------</td>
</tr>
<tr>
<td>G^n</td>
<td>FORMAT-OUTPUT</td>
<td>n fields are extracted starting at the first field that is separated by the ^ separator. The separator may be any character.</td>
<td>For code G/2: 1/2/3/4 Ô 1/2</td>
</tr>
<tr>
<td>Gm^n</td>
<td>FORMAT-OUTPUT</td>
<td>n fields are extracted, skipping m fields that are separated by the ^ separator. The separator may be any character.</td>
<td>For code G1/2: 1/2/3/4 Ô 2/3</td>
</tr>
<tr>
<td>MC/A</td>
<td>FORMAT-OUTPUT</td>
<td>Extracts nonalphabetic characters.</td>
<td>R.3#4T Ô 3#4</td>
</tr>
<tr>
<td>MC/N</td>
<td>FORMAT-OUTPUT</td>
<td>Extracts nonnumeric characters.</td>
<td>R.3#4T Ô R#T</td>
</tr>
<tr>
<td>MCA</td>
<td>FORMAT-OUTPUT</td>
<td>Extracts alphabetic characters.</td>
<td>R.3#4T Ô RT</td>
</tr>
<tr>
<td>MCN</td>
<td>FORMAT-OUTPUT</td>
<td>Extracts numeric digits.</td>
<td>R.3#4T Ô 34</td>
</tr>
</tbody>
</table>
### Table AA-3. Numeric Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLp</td>
<td>FORMAT-INPUT</td>
<td>Converts a number from external format to internal format with precision $p$. Precision is the number of decimal places. The result is placed left (ML) or right (MR) justified in the output field. Rounding is performed if necessary.</td>
<td>For code MR3: 1234.5678 õ 1234.568</td>
</tr>
<tr>
<td>MRp</td>
<td>FORMAT-INPUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLps</td>
<td>FORMAT-INPUT</td>
<td>Converts a number from external format to internal format with precision $p$ and scaling $s$. Precision is the number of decimal places. Scaling is the scaling factor ($10^5$). The result is placed left (ML) or right (MR) justified in the output field. Rounding is performed if necessary.</td>
<td>For code MR21: 1234.5678 õ 12345.68</td>
</tr>
<tr>
<td>MRps</td>
<td>FORMAT-INPUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MLp-</td>
<td>OUTPUT</td>
<td>Converts a number from internal format to external format with precision $p$, descaling $s$, options $o$, and picture format $f$. The result is placed left justified (ML) or right justified (MR) in the output field.</td>
<td>For code MR20E(*12) -12.34 õ *****&lt;12.34&gt;</td>
</tr>
<tr>
<td>sof</td>
<td></td>
<td></td>
<td>For code MR31: 54.32 õ 5.432</td>
</tr>
<tr>
<td>MRp-</td>
<td>OUTPUT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sof</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table AA-3. Numeric Codes (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Precision is the number of decimal places. Descaling is the descaling factor (10^{-5}). Options can be one of the following:</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>Adds a CR (credit) suffix if the result is negative.</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td>Adds a DB (debit) suffix if the result is positive.</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>Places &lt;&gt; around the result if the result is negative.</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td></td>
<td>Adds a - suffix if the result is negative.</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td></td>
<td>Suppresses the leading - sign if the result is negative.</td>
<td></td>
</tr>
<tr>
<td>Z</td>
<td></td>
<td>Suppresses leading zeroes.</td>
<td></td>
</tr>
</tbody>
</table>
### Table AA-3. Numeric Codes (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>,</code></td>
<td></td>
<td>Thousands separator. Inserts a , every third digit to the left of the decimal point.</td>
<td></td>
</tr>
<tr>
<td><code>$</code></td>
<td></td>
<td>Adds a $ prefix.</td>
<td></td>
</tr>
</tbody>
</table>

The picture format is optional and is performed after any o option; can be one of the following:

* `#n` Fills the result with the necessary spaces from the left to reach a field width of n. |

* `*n` Fills the result with the necessary asterisks from the left to reach a field width of n. |
Table AA-3. Numeric Codes (continued)

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>%n</td>
<td>Fills the result with the necessary zeroes from the left to reach a field width of ( n ).</td>
<td></td>
</tr>
</tbody>
</table>

The picture format may also include other literal characters that will be inserted into the output. For clarity, the picture format may be enclosed in parentheses ( ). If literal characters are part of the picture format, the use of parentheses is advised to prevent conflict between the options and the picture format.

Table AA-4. Port/Project Code

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>U50BB</td>
<td>FORMAT-OUTPUT</td>
<td>Returns the port number and project name.</td>
<td>.32 SOFTSELL</td>
</tr>
</tbody>
</table>
### Table AA-5. Text Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCL</td>
<td>FORMAT-INPUT FORMAT-OUTPUT</td>
<td>Converts alphabetic characters to lowercase.</td>
<td>ANY text Ô any text</td>
</tr>
<tr>
<td>MCT</td>
<td>FORMAT-INPUT FORMAT-OUTPUT</td>
<td>Converts alphabetic characters to initial capitalization.</td>
<td>ANY text Ô Any Text</td>
</tr>
<tr>
<td>MCU</td>
<td>FORMAT-INPUT FORMAT-OUTPUT</td>
<td>Converts alphabetic characters to uppercase.</td>
<td>ANY text ÔANY TEXT</td>
</tr>
<tr>
<td>Tx</td>
<td>FORMAT-INPUT FORMAT-OUTPUT</td>
<td>Returns $x$ characters of an alphanumeric field, beginning at column 1.</td>
<td>For code T5: ANY text ÔANY t</td>
</tr>
<tr>
<td>Tx, y</td>
<td>FORMAT-INPUT FORMAT-OUTPUT</td>
<td>Returns $x$ characters of an alphanumeric field, beginning at column $y$.</td>
<td>For code T6,3: ANY text ÔY text</td>
</tr>
</tbody>
</table>
## Table AA-6. Time Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Verb</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>MT</td>
<td>FORMAT-INPUT</td>
<td>Converts external time to internal format. The external time may be in 24-hour or AM/PM format.</td>
<td>20:00:01 Ô 72001</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>08:00:01PM Ô 72001</td>
</tr>
<tr>
<td>MT</td>
<td>FORMAT-OUTPUT</td>
<td>Converts internal time to 24-hour format.</td>
<td>72001 Ô 20:00</td>
</tr>
<tr>
<td>MTH</td>
<td>FORMAT-OUTPUT</td>
<td>Converts internal time to AM/PM format.</td>
<td>72001 Ô 08:00PM</td>
</tr>
<tr>
<td>MTHS</td>
<td>FORMAT-OUTPUT</td>
<td>Converts internal time to AM/PM format, including seconds.</td>
<td>72001 Ô 08:00:01PM</td>
</tr>
<tr>
<td>MTS</td>
<td>FORMAT-OUTPUT</td>
<td>Converts internal time to 24-hour format, including seconds.</td>
<td>72001 Ô 20:00:01</td>
</tr>
</tbody>
</table>
EDGE Environment Variables

Overview

The following environment variables are used to modify EDGE’s behavior. They should be set in the EDGE file .user.profile, using the following syntax:

```
ENV_VAR=value;export ENV_VAR
```

where

- **ENV_VAR** The environment variable to set.
- **value** The value of the environment variable.
The available environment variables are listed alphabetically in Table AB-1.

### Table AB-1. EDGE Environment Variables

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>BCKGRNDOUT=var</td>
<td>Specifies the directory for the standard error and standard out files to use for tasks running in the background. The paths are:</td>
<td>$BCKGRNDOUT</td>
</tr>
<tr>
<td></td>
<td>$EDGE/var/stderr.pid</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$EDGE/var/stdout.pid</td>
<td></td>
</tr>
<tr>
<td>BPRINTLEN</td>
<td>Printer page length.</td>
<td>66</td>
</tr>
<tr>
<td>BS_KEY</td>
<td>Controls the value of the erase character. By default, the backspace key (ASCII 8) is the erase character. If the terminal does not support a backspace key, this variable can be used to assign an alternative key.</td>
<td></td>
</tr>
<tr>
<td>bufferSetProperty</td>
<td>If set to 1, specifies actions are to be placed in a buffer until the logic flow is exited. For more information, see the SET-PROPERTY verb.</td>
<td></td>
</tr>
<tr>
<td>CALC_LOG_PATH</td>
<td>The path for the CALCULATE verb log file. If not set, $TMPDIR is used for the path.</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>CEF_SERVER_ENCODE</td>
<td>Supports embedded system delimiters when using the CALL-EXTERNAL-FUNCTION verb. If set to N, the Server Library Encoding Mode is disabled and as a result, the encoding and decoding of embedded system delimiters is bypassed. Do not disable this feature if any external function parameters contain embedded system delimiters.</td>
<td>Y</td>
</tr>
<tr>
<td>CONSOLE</td>
<td>The path to which fatal error messages are written. To disable writing fatal errors specify /dev/null.</td>
<td>System console</td>
</tr>
<tr>
<td>CORRPRINT_TRUNC_AS_EQL</td>
<td>If set to 1, causes Correspondence to display numbers truncated, which is the way numbers are displayed by EQL. By default, Correspondence rounds numbers.</td>
<td>0</td>
</tr>
<tr>
<td>CSIZE</td>
<td>Cache size for Zip/ACP Locator.</td>
<td></td>
</tr>
<tr>
<td>EDGE_EMAIL_CHARSET</td>
<td>Specified character set used by the (optional) EMAIL verb to send outgoing email messages. Example value: ISO8859-1.</td>
<td></td>
</tr>
<tr>
<td>EDGEDTABLE</td>
<td>(Used by the Sequent only). Increases the file table size to the value set by the variable. The size can only be increased, never decreased, that is, if the value specified is smaller than the current table size, the current table size is not decreased to the value set.</td>
<td></td>
</tr>
</tbody>
</table>
### Table AB-1. EDGE Environment Variables (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>EDGEULIMIT</code></td>
<td>Specifies the ulimit (UNIX’s maximum file size variable) for EDGE files.</td>
<td>2000000</td>
</tr>
<tr>
<td><code>EMPTY_AS_ZERO</code></td>
<td>If set to 1, displays a zero in empty numeric fields.</td>
<td></td>
</tr>
<tr>
<td><code>ENABLE_DB_DEVICE_DELETION</code></td>
<td>If set to 1, allows database devices to be deleted from EDGE Developer.</td>
<td>null</td>
</tr>
<tr>
<td><code>ENABLE_RDB_TRACE</code></td>
<td>If set to 1, enables the DDE verb. The default is null, which disables the verb.</td>
<td>null</td>
</tr>
<tr>
<td><code>ETL1</code></td>
<td>Sets the number of lines printed per page.</td>
<td>60</td>
</tr>
<tr>
<td><code>EQL_CASE_SENSITIVE</code></td>
<td>If set to non-zero, EQL keywords are case-sensitive.</td>
<td></td>
</tr>
<tr>
<td><code>F_AUTO_RETURN</code></td>
<td>If set to 1, an automatic ENTER is performed when the end of an input field in the guide is reached; the focus continues to the next screen object. If not set, the cursor stays on the current object. If set, overrides the default autoTab option in the EDGE client .ini file. If autoTab option is explicity set by the user the client setting will override the setting for this environment variable.</td>
<td>1</td>
</tr>
<tr>
<td><code>F_EXEC_LOGIC</code></td>
<td>If set to 1, logic flows attached to a field are executed when using tab and arrow keys, even if the field has not been modified.</td>
<td></td>
</tr>
<tr>
<td><code>GEOHEARTBEAT=first, addl, limit</code></td>
<td>Specifies the EDGE Client heartbeat parameters:</td>
<td>60,5,5</td>
</tr>
</tbody>
</table>
**Table AB-1. EDGE Environment Variables (continued)**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>first</td>
<td>The time, in seconds, between heartbeats. The default is 60.</td>
<td></td>
</tr>
<tr>
<td>addl</td>
<td>The time, in seconds, to send additional heartbeats, if the first heartbeat has not been answered. The default is 5.</td>
<td></td>
</tr>
<tr>
<td>limit</td>
<td>The number of unanswered heartbeats that cause the host server to terminate. The default is 5.</td>
<td></td>
</tr>
<tr>
<td>GUIDE_WATCHER</td>
<td>The path for the guide watcher log file.</td>
<td></td>
</tr>
<tr>
<td>IGNORE_OVERLAPPING</td>
<td>If set to non-zero, screen assemblies will report overlapping and overrun fields as errors.</td>
<td></td>
</tr>
<tr>
<td>IGNORE_SPECIAL_CHARS</td>
<td>If set to 1, keys such as cursor up are ignored on input in the EDGE menus and screens. This variable should not be changed.</td>
<td></td>
</tr>
<tr>
<td>INTERACTIVE</td>
<td>If set to Y, the user is prompted when chaining from one task to the next.</td>
<td>N</td>
</tr>
<tr>
<td>IO_LOG_FILE=var</td>
<td>The name of the log file for all I/O errors. The path is: $EDGE/tmp/var</td>
<td>io_error</td>
</tr>
<tr>
<td>LANG</td>
<td>The language code for the system. The default specifies English. Null is equivalent to the default.</td>
<td></td>
</tr>
</tbody>
</table>
Table AB-1. EDGE Environment Variables (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOGON_NO_VALIDATION</td>
<td>Determines whether the Logon program validates the system login and password against the operating system.</td>
<td>blank</td>
</tr>
<tr>
<td></td>
<td>Y The Logon program does not validate the system login and password. However, Logon will still prompt for system login and password.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>blank The Logon program validates the system login and password.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This is the default.</td>
<td></td>
</tr>
<tr>
<td>MAX_LOGFILE_SIZE</td>
<td>Specifies the maximum size of the telephony log.</td>
<td></td>
</tr>
<tr>
<td>MAX_XREF_MATCH_KEYS</td>
<td>Limits the number of matches returned by Cross-Reference Files search (CTRL+X) and the GET-KEY verb, Get List, Cross Reference command. This variable is honored only if partial cross-reference search is used.</td>
<td></td>
</tr>
<tr>
<td>MAXVFIL</td>
<td>Determines the size of the c-tree library &quot;virtual file&quot; table. Large values will help performance, however, performance issues can arise if the value is set too high that the system is unable to open file handles for other operations.</td>
<td>40</td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>MQSERIES_SKIP_SRV_CHECK</td>
<td>If set to Y, a HELLO message is not sent to the MQSeries I/O device output queue. If set to N, the HELLO message is sent and must be responded to by the server.</td>
<td>N</td>
</tr>
<tr>
<td>MSGQ_SKIP_SRVPID_CHECK</td>
<td>Keeps EDGE from checking if the server is running when opening a message queue I/O device.</td>
<td>N</td>
</tr>
<tr>
<td>NO_BEEPS PLEASE</td>
<td>If set, EDGE will not beep when it tries to lock a record that has already been locked, and is waiting for the lock to clear.</td>
<td></td>
</tr>
<tr>
<td>PRINTERCMD</td>
<td>The command for system printing.</td>
<td></td>
</tr>
<tr>
<td>PRINTERCMD1</td>
<td>Only used by the Dr. tools spool utility to redirect held files, to a 'real' spooler such as lp.</td>
<td></td>
</tr>
<tr>
<td>PRINTERNAME</td>
<td>The EDGE system printer name.</td>
<td></td>
</tr>
<tr>
<td>PRINTCOPIES</td>
<td>The number of print copies.</td>
<td></td>
</tr>
<tr>
<td>QSEED_STATUS_NOWAIT</td>
<td>If set to Y, Qseed skips locked status records.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>If set to N, Qseed will wait if it encounters a locked status record.</td>
<td></td>
</tr>
<tr>
<td>SET_DENTRY_NULL</td>
<td>If set, the system field $ENTRY returns “” if only ENTER is pressed at an input field. If not set, $ENTRY returns the value in the field.</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>STAT_INTERVAL</td>
<td>The interval after which summary statistics for each type of command executed are written to the stat log file.</td>
<td>3600 seconds</td>
</tr>
<tr>
<td>STRICT_DATE_CHECK</td>
<td>Specifies the action to take when invalid numeric dates are entered (invalid dates are dates such as 23/2/99 or dates with no year specified); can be one of the following:</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Y                                                                                                           If a month greater than 12 is encountered, the month and day are swapped. If the swap still results in an invalid date, an error message is displayed.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>If no year is specified, the current system year is used.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>In the example above, the date is swapped as 2-23-99.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N                                                                                                           If an invalid date is encountered, an error message is displayed indicating a valid date must be entered.</td>
<td></td>
</tr>
<tr>
<td>SUPERSRVR_LOG_DEBUG</td>
<td>Determines whether debug information is logged.</td>
<td>blank</td>
</tr>
</tbody>
</table>

This option has effect only in ASCII operations. For information on similar options for EDGE Client, see the EDGE 7.11 Client Reference Manual.
### Table AB-1. EDGE Environment Variables (continued)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUPERSRVR_LOG_TRACE</td>
<td>Y The SuperServer logs debug information. If a log file is not specified in the EDGE Developer SuperServer Setup screen, debug information is not logged.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>blank The SuperServer does not log debug information.</td>
<td>blank</td>
</tr>
<tr>
<td></td>
<td>This is the default.</td>
<td></td>
</tr>
<tr>
<td>TANDEM_TOGGLE</td>
<td>Determines whether trace information is logged.</td>
<td>blank</td>
</tr>
<tr>
<td></td>
<td>Y The Superserver logs trace information in the specified log file. If a log file is not specified in the EDGE Developer SuperServer Setup screen, trace information is not logged.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>blank The Superserver does not log trace information.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>This is the default.</td>
<td></td>
</tr>
<tr>
<td>TPHNY_CONSULT_PATH</td>
<td>The key to toggle between modes in User Monitor. If not specified, CTRL+T is used.</td>
<td>$TMPDIR</td>
</tr>
<tr>
<td></td>
<td>The path to which data that is send with the miscellaneous data parameters is written.</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>TPHNY_LOG_PATH</td>
<td>The directory in which to store the telephony log files.</td>
<td>$TMPDIR</td>
</tr>
<tr>
<td>UPPER</td>
<td>If set to 1, all lowercase input is automatically converted to uppercase.</td>
<td>1</td>
</tr>
<tr>
<td>USE_KEYPAD</td>
<td>If set to 1 (the default), the numeric keypad functions in application keypad mode. If set to zero, the numeric keypad functions as a numeric keypad. Both the host and the terminal must support this capability.</td>
<td>50</td>
</tr>
<tr>
<td>V_CALC_STACK_SIZE</td>
<td>A memory stack that is used throughout EDGE for processing. Check server logs for error messages or warnings relating to insufficient stack size depending on usage. A typical message that will appear in the server logs are OUT_OF_MEMORY or guide assembly errors.</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>VERCONT_PATH</td>
<td>The path to the RCS executable files. For example, if you copied them to the D drive in a folder called gnu, you would set this variable as follows: VERCONT_PATH=&quot;D:/gnu&quot;;export VERCONT_PATH  For information about version control, see the section Version Control.</td>
<td></td>
</tr>
<tr>
<td>VERCONT_PGM</td>
<td>The underlying utility to use with EDGE Version Control. If set to &quot;RCS&quot;, the RCS version control utility is used; otherwise, EDGE assumes SCCS is to be used.</td>
<td></td>
</tr>
</tbody>
</table>
Overview

While in EDGE in ASCII operations, you can use keyboard commands to perform actions in a guide. (Depending on your keyboard type, some of the keys may or may not be available.) In addition, some commands are available by typing a period (dot) and a letter; these commands are referred to as dot commands. The commands are listed in Table AC-1.

You can restrict the use of many of the commands on a guide-by-guide basis. When a dot command is suppressed from the Guide Options screen, the keys associated with that command are also restricted.
Alphabetical Listing

The commands available in ASCII are listed alphabetically in Table AC-1.

*Note: In the table, N/A indicates no key is available.*

### Table AC-1. Keyboard Commands in ASCII Operations

<table>
<thead>
<tr>
<th>Description</th>
<th>Keys</th>
<th>Dot Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Getting Information</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Invokes a cross-reference search.</td>
<td>CTRL+X</td>
<td>N/A</td>
</tr>
<tr>
<td>Displays field help.</td>
<td>CTRL+V</td>
<td>N/A</td>
</tr>
<tr>
<td>Prints the guide screen in test.</td>
<td>CTRL+y</td>
<td>.z</td>
</tr>
<tr>
<td><strong>Moving Around the Screen</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Moves to the next entry field.</td>
<td>CTRL+m</td>
<td>N/A</td>
</tr>
<tr>
<td>Moves right to the next object.</td>
<td>CTRL+i</td>
<td>.c</td>
</tr>
<tr>
<td>Moves to the previous input object.</td>
<td>SHIFT+tab</td>
<td>.b</td>
</tr>
<tr>
<td>Description</td>
<td>Keys</td>
<td>Dot Commands</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>-----------------</td>
<td>--------------</td>
</tr>
<tr>
<td>Goes to another row or input location.</td>
<td>CTRL+g f15</td>
<td>.g</td>
</tr>
<tr>
<td>Goes to the first entry field on screen.</td>
<td>CTRL+t home f10</td>
<td>.t</td>
</tr>
<tr>
<td>Moves to the beginning of the entry field.</td>
<td>CTRL+a</td>
<td>N/A</td>
</tr>
<tr>
<td>Moves to the end of the entry field.</td>
<td>CTRL+d END</td>
<td>N/A</td>
</tr>
<tr>
<td>Moves down to the next line.</td>
<td>CTRL+j down arrow</td>
<td>.n</td>
</tr>
<tr>
<td>Moves up to the previous line.</td>
<td>CTRL+z up arrow</td>
<td>.l</td>
</tr>
<tr>
<td>Moves the cursor left in the field.</td>
<td>CTRL+u left arrow</td>
<td>N/A</td>
</tr>
<tr>
<td>Moves the cursor right in the field.</td>
<td>CTRL+f right arrow</td>
<td>N/A</td>
</tr>
<tr>
<td><strong>Editing Data</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Erases the input character to the left of the cursor.</td>
<td>CTRL+h BACKSPACE</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Table AC-1. Keyboard Commands in ASCII Operations (continued)

<table>
<thead>
<tr>
<th>Description</th>
<th>Keys</th>
<th>Dot Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>Erases the entry at the current input location.</td>
<td>CTRL+e f9</td>
<td>.e</td>
</tr>
<tr>
<td>Erases the input character at the cursor.</td>
<td>DELETE</td>
<td>N/A</td>
</tr>
<tr>
<td>Modifies the text at an input prompt.</td>
<td>CTRL+r INSERT</td>
<td>.r N/A</td>
</tr>
<tr>
<td>Toggles between overwrite and insert mode.</td>
<td>CTRL+o INSERT</td>
<td>N/A</td>
</tr>
<tr>
<td>Undoes the change in the entry field.</td>
<td>CTRL+w</td>
<td>.w</td>
</tr>
</tbody>
</table>

Moving From Screen to Screen

| Jumps (routes directly) to another screen.                                | CTRL+n PAGE DOWN f14 | .j          |
| Moves to the previous screen.                                            | CTRL+p PAGE UP f12   | .p           |
| Invokes function key n.                                                  | fn              | .fn          |
EDGE Directory Structure

Overview

EDGE uses a directory structure with the following directories created under the edge directory:

- EDGE Directory.
- EDGESYS Directory.
- TEMPLATE Directory.
- STEMPLEASE Directory.

For example, if EDGE is installed on the /usr partition, the following directories are created:

- /usr/edge/EDGE
- /usr/edge/EDGESYS
- /usr/edge/TEMPLATE
- /usr/edge/STEMPLE.
Figure AD-1 depicts this structure.

![Figure AD-1. EDGE Directory Structure](image)

The following briefly describes the directories under the EDGE directory. The subdirectories and files contained in these directories are described in more detail later in this appendix.

- **EDGE Directory**
  Stores all system files.

- **EDGESYS Directory**
  Stores the executable programs including, EDGE libraries, system screens, tables, and error messages.

- **TEMPLATE Directory**
  Stores the template files for creating a project.

- **STEMPLE Directory**
  Stores the template files for creating the EDGE system level files.
The following table lists the directories and files contained in the EDGE directories.

**Table AD-1. EDGE Directories and Files**

<table>
<thead>
<tr>
<th>Directory or File</th>
<th>EDGE</th>
<th>EDGESYS</th>
<th>TEMPLATE</th>
<th>STEMPLATE</th>
</tr>
</thead>
<tbody>
<tr>
<td>backgrnd</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>backup</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>bbin</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bin/</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>cbinn</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CUSTOM-SUBS</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>data_d/</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>data_x/</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>dict_d/</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>dict_x/</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>edge_country</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>etc</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>include</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lib</td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Directory or File</td>
<td>EDGE</td>
<td>EDGESYS</td>
<td>TEMPLATE</td>
<td>STEMPLATE</td>
</tr>
<tr>
<td>------------------</td>
<td>------</td>
<td>---------</td>
<td>----------</td>
<td>-----------</td>
</tr>
<tr>
<td>.object</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.object1</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>.object2</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>odbc</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>odbclog</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>POINTER-FILE</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>.profile</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>src</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>terminfo</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>tmp</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Structure for c-tree Files

Currently, all files that are created with the default database device use a c-tree file structure. A c-tree structure contains four directories which store the parts of the file. The directories are:

- **data_d**: The data portion of the files, including the field data and disk usage for the file.
- **data_x**: The index to the data in the data_d file, including the record key, secondary key, and address of data.
- **dict_d**: The dictionary portion of the fields in the file. (This is used only for ad hoc reporting and ACL.)
- **dict_x**: The index to the data in the dict_d (dictionary) file.

The following files are created when an EDGE file is created:

- **F_filename**: The F_filename file contains field definitions for the file. The dict_d and dict_x files of the data file are built from this file when field positions are assigned.

(An F_filename file is created when a new file is created, but data is not stored until fields are created.)
<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MD file</td>
<td>The MD file contains information on where the file is located. If the file is a relational database type, it also includes database mapping information. The information is stored in a record with the filename name used as the record name. For example, the CUSTOMER record in the MD file contains information about the CUSTOMER file.</td>
</tr>
<tr>
<td>PFILES file</td>
<td>The PFILES file contains file the cross-reference definition for the file, stored in a record with the filename used as the record name.</td>
</tr>
<tr>
<td>PROJ.PARAM</td>
<td>The PROJ.PARAM file contains records with project information, such as project options and the names of the files in the project.</td>
</tr>
</tbody>
</table>

If a cross-reference file is built, the following file is created:

- **X_filename**
  - The cross-reference data for the file; which consists of the data and record ID of the data.
The POINTER-FILE directory contains saved selection lists, which are generated lists of record IDs from a file. (For more information on selection lists, see the section Selection Lists.)

The POINTER-FILE directory is found in three directories. The EDGE directory stores the selection lists that were saved on the system level. The TEMPLATE directory stores an empty directory, which is copied when a project is created. The project directory stores selection lists that were saved for the project.
The EDGE directory stores system files, including the user IDs, port, and site configuration information.

In addition to the data_d/, data_x/, dict_d/, and dict_x/ directories and POINTER-FILE, the EDGE directory contains the following directories and files:

- **bin**: Stores the system-level tasks that have been assembled. Also stores the getacct program, which is used to determine where EDGE is currently located on the system.
  
  Task files are stored under the task name suffixed with an _T. For example, the UPDATE task would be stored as UPDATE_T.
  
  If you are using a data server or server based router with telephony, information might be stored in here or at the project level bin directory.

- **backgrnd**: Stores the stderr and stdout files for each task which is run by the Task Scheduler.

- **backup**: Stores backup copies of deleted database devices to permit restoration by AIT Customer Care. (Refer to the EDGE 6.0.1 DataBase Link Reference Manual).

- **CUSTOM-SUBS**: Used for special situations, such as custom verbs.
<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>edge_country</td>
<td>Stores the LANG environment variable, which determines the language used for the site.</td>
</tr>
<tr>
<td>include</td>
<td>Stores include files, which contain definitions of functions.</td>
</tr>
<tr>
<td>lib</td>
<td>Stores libraries used at the system level.</td>
</tr>
<tr>
<td>.object</td>
<td>Stores the .objects, which is the object code that is created when a guide is assembled. Each time a guide is swapped, the assembled logics are copied to this directory.</td>
</tr>
<tr>
<td>odbc</td>
<td>Stores information for the ODBC optional module.</td>
</tr>
<tr>
<td>.profile</td>
<td>Used to initialize the EDGE environment. Is executed before the user Login screen is displayed.</td>
</tr>
<tr>
<td>tmp</td>
<td>A scratch directory used for the following:</td>
</tr>
<tr>
<td>-----</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>• Temporarily stores print jobs while they are being created (before they are spooled).</td>
</tr>
<tr>
<td></td>
<td>• Stores default selection lists, for example, the DLIST, which allows users to temporarily (while in ACL) store data by directing it to DLIST. For example:</td>
</tr>
<tr>
<td></td>
<td>SELECT FROM CUST USING DLIST</td>
</tr>
<tr>
<td></td>
<td>• Acts as a buffer for the transfer of data between EDGE processes. These files are created during the logon process.</td>
</tr>
<tr>
<td></td>
<td>• Stores the log files for telephony, copy.proj, and the compact utility.</td>
</tr>
</tbody>
</table>
## EDGESYS Directory

The EDGESYS directory stores the executable programs, error messages, and libraries used by EDGE.

The EDGESYS directory is replaced when EDGE is upgraded.

In addition to the data_d/, data_x/, dict_d/, and dict_x/ directories and POINTER-FILE, the EDGESYS directory contains the following directories and files:

<table>
<thead>
<tr>
<th>Directory</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bbin</td>
<td>Stores translated BASIC programs.</td>
</tr>
<tr>
<td>bin</td>
<td>Stores programs which are linked from a combination of programs contained in bbin/, cbin/ and etc/ directories. (By storing them in one directory and linking to others, the search path is shortened and performance improved.) The XGUIDE##, which is the executable program consisting of all the swapped guides, is also stored in this directory.</td>
</tr>
<tr>
<td>cbin</td>
<td>Stores C programs and utilities.</td>
</tr>
<tr>
<td>etc</td>
<td>Stores EDGE shell scripts.</td>
</tr>
<tr>
<td>include</td>
<td>Used during the assembly process of tasks or guide for error messages, and definition of constants.</td>
</tr>
<tr>
<td>lib</td>
<td>Stores runtime, support, and interface libraries. The interface libraries are used with telephony, relational databases, and other optional modules.</td>
</tr>
<tr>
<td>Directory</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>src</td>
<td>Contains ipc server examples.</td>
</tr>
<tr>
<td>terminfo</td>
<td>Contains information about the capabilities of the terminal types that can be used with EDGE. Also includes keyboard mapping by terminal type. (This is used by ASCII terminals only.)</td>
</tr>
</tbody>
</table>
**TEMPLATE Directory**

The TEMPLATE directory is used every time you create a new project. The act of creating a new project, copies the TEMPLATE directory into the new directory under the project name.

*Note: This directory is replaced during an EDGE Upgrade. It is recommended that you do not create a project called TEMPLATE.*

In addition to the data_d/, data_x/, dict_d/, and dict_x/ directories and POINTER-FILE, the TEMPLATE directory contains the following directories and files:

- **bin**: Stores the XGUIDE that is used as a template when creating a project. (The XGUIDE contains the EDGE executable objects which are run when a guide is opened.)
- **.object1**: This directory is empty. In the project directory, after the first assembly, this contains the compiled logic flows.
- **.object2**: This directory is empty. In the project directory, after the first assembly, this contains the compiled logic flows.
The STEMPLATE directory stores all system level template files. This directory is used only during the upgrade of EDGE software. This directory can also be used to copy files to EDGE if a file gets corrupted. For example, if your STATUSID file becomes corrupted or needs compacting, it may be easier to copy it from the STEMPLATE directory. (You should do so when agents are not using EDGE.)

This directory contains the data_d/, data_x/, dict_d/, and dict_x/ directories.

Note: This directory is replaced during an EDGE upgrade.
### Overview

An .ini file is a Windows initialization file that contains configuration information. It is a text file that can be viewed and modified with any text editor.

The `etww.ini` file is a text file located in the WINDOWS folder that contains settings for EDGE Developer.
**etww.ini Sections and Options**

The etww.ini file has the following sections:

- **[3270]** Defines the key map for 3270.
- **[Communications]** The name of the connection setup file.
- **[Debug]** Defines features for debugging.
- **[Directories]** The path and directories for the connection setup files.
- **[General Setup]** Defines options used to display and refresh ETWW screens.
- **[GEO]** The login name and password for EDGE Client.
- **[Settings]** Defines login settings for the EDGE server.
- **[Screen Editor]** The conditions under which ETWW screens can be set up.
- **[Splash]** Defines the conditions under which the splash screen can be set up.

To set an option, use the following syntax:

```
optionName=setting
```

For example, **AutoLogon**=Y. There should be no spaces around the equal (=) sign. Make sure the option is in the correct section.
names and option names are case sensitive. The setting, if it is Y or N, is case-insensitive. The case for other settings depends on the target and should generally be entered as shown in the tables.

The following tables list the sections and options of the etww.ini file. The brackets ([ ]) enclosed around the section names are required. The sections can be in any order in the file; the options must be in the appropriate section, but can be in any order within that section.

**Table AE-1. [3270] in the etww.ini file**

<table>
<thead>
<tr>
<th>[3270]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>3270KeyMap</td>
<td>Key map for 3270.</td>
<td></td>
</tr>
</tbody>
</table>

**Table AE-2. [Communications] in the etww.ini File**

<table>
<thead>
<tr>
<th>[Communications]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConnSetUpFile</td>
<td>Name of the default connection setup file.</td>
<td></td>
</tr>
</tbody>
</table>

### Table AE-3. [Debug] in the etww.ini File

<table>
<thead>
<tr>
<th>[Debug]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>HostDebugFile</td>
<td>Name of the host trace file.</td>
<td>/dev/null</td>
</tr>
<tr>
<td>HostTrace</td>
<td>Determines whether host trace is needed; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Turn on host tracing.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>N - Do not use host tracing.</td>
<td></td>
</tr>
<tr>
<td>Trace</td>
<td>Determines whether client trace is needed; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Turn on client tracing.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>N - Do not use client tracing.</td>
<td></td>
</tr>
<tr>
<td>TraceAppend</td>
<td>Determines how to handle the existing trace file; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Append to existing file.</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>N - Overwrite existing file.</td>
<td></td>
</tr>
<tr>
<td>TraceFileName</td>
<td>Name of the client trace file.</td>
<td>ETWWTrace.TXT</td>
</tr>
<tr>
<td>TraceLevel</td>
<td>Client trace level.</td>
<td>1</td>
</tr>
</tbody>
</table>
### Table AE-4. [Directories] in the ETWW.INI File

<table>
<thead>
<tr>
<th>[Directories]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEOPath</td>
<td>Program path for EDGE Client.</td>
<td></td>
</tr>
<tr>
<td>ProgDir</td>
<td>This is no longer used.</td>
<td></td>
</tr>
</tbody>
</table>

### Table AE-5. [General Setup] in the etww.ini File

<table>
<thead>
<tr>
<th>[General Setup]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlwaysOnTop</td>
<td>Keep ETWW screen always on top; can be one of the following:</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Y - ETWW screen always on top of other screen.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Not so.</td>
<td></td>
</tr>
<tr>
<td>AutoLogon</td>
<td>Automatic login to server; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Server login is attempted.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Server login is not attempted.</td>
<td></td>
</tr>
<tr>
<td>AutomaticRefresh</td>
<td>Automatic refresh; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table AE-5. **[General Setup] in the etww.ini File** (continued)

<table>
<thead>
<tr>
<th>[General Setup]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>BrowserOnTop</td>
<td>Y - Enable automatic refresh.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Disable automatic refresh.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Keep browser screen always on top; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Browser screen always on top of other screen.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Not so.</td>
<td></td>
</tr>
<tr>
<td>Largelcons</td>
<td>Determines whether large icons should be used in tool bar; can be one of</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Use large icons.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Use small icons.</td>
<td></td>
</tr>
</tbody>
</table>

Table AE-6. **[GEO] in the etww.ini File**

<table>
<thead>
<tr>
<th>[GEO]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEOPasswr</td>
<td>EDGE Client server password.</td>
<td></td>
</tr>
</tbody>
</table>
**Table AE-6. [GEO] in the etww.ini File**

<table>
<thead>
<tr>
<th>[GEO]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEOConnPort</td>
<td>The port number used to launch EDGE Client from EDGE Developer using a socket connection.</td>
<td>23</td>
</tr>
<tr>
<td>GEOServerID</td>
<td>EDGE Client server login name.</td>
<td></td>
</tr>
</tbody>
</table>

**Table AE-7. [Settings] in the etww.ini File**

<table>
<thead>
<tr>
<th>[Settings]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>loginMatch</td>
<td>EDGE server login telnet prompt string to look for when attempting server login.</td>
<td>in:</td>
</tr>
<tr>
<td>passwordMatch</td>
<td>EDGE server password telnet prompt string to look for when attempting server login.</td>
<td>ord:</td>
</tr>
</tbody>
</table>
### Table AE-8. [Screen Editor] in the etww.ini File

<table>
<thead>
<tr>
<th>[Screen Editor]</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>HorizontalGridIncrements</td>
<td>Number of measurement units in between grid lines in the horizontal direction.</td>
<td></td>
</tr>
<tr>
<td>Metric</td>
<td>Unit of measurement; can be one of the following:</td>
<td>Rol/Col</td>
</tr>
<tr>
<td></td>
<td>Row/Col.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Centimeters.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Inches</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Millimeters.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Pixels.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Points.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Twips.</td>
<td></td>
</tr>
<tr>
<td>NewVersion</td>
<td>Determines which version of the screen editor should be used; can be one of the following:</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Y - Use new version.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Use old version.</td>
<td></td>
</tr>
<tr>
<td>ObjectPropertyRows</td>
<td>Number of rows for the object properties.</td>
<td></td>
</tr>
<tr>
<td>[Screen Editor]</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>-------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>SnapToGrid</td>
<td>Determines whether the size of object should adjust to nearest grid line; can be one of the following:</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Y - Adjust to grid line.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Do not adjust.</td>
<td></td>
</tr>
<tr>
<td>UseMaxCharWidth</td>
<td>Determines whether the maximum character width should be used; can be one of the following:</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td>Y - Use maximum character width.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Do not use.</td>
<td></td>
</tr>
<tr>
<td>VerticalGrid Increments</td>
<td>Number of measurement units in between grid lines in the vertical direction.</td>
<td></td>
</tr>
<tr>
<td>[Splash]</td>
<td>Description</td>
<td>Default</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>RemoveSplash</td>
<td>Removes the splash screen upon mouse click; can be one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Enable removal upon mouse click.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Do not remove.</td>
<td></td>
</tr>
<tr>
<td>ShowSplash</td>
<td>Show the splash screen; can be one of the following:</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>Y - Display splash screen.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Do not display.</td>
<td></td>
</tr>
<tr>
<td>SplashAut</td>
<td>Determines whether the timer for the splash screen should be used; can be</td>
<td>Y</td>
</tr>
<tr>
<td></td>
<td>one of the following:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Y - Enable the splash timer.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>N - Disable the splash timer.</td>
<td></td>
</tr>
<tr>
<td>SplashTimer</td>
<td>Timer for removal of the splash screen in seconds, applicable only when</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>SplashAut is set to Y.</td>
<td></td>
</tr>
</tbody>
</table>
Symbols

? button, 1620
.object file, 2124
.profile file, 2124
\* (correspondence), 329
\ (correspondence), 330
\AD (correspondence), 325, 330
\ADJUST (correspondence), 325, 330
\BM (correspondence), 324, 331
\BMARGIN (correspondence), 324, 331
\BOT (correspondence), 324, 331
\BP (correspondence), 325, 333
\BR (correspondence), 329
\COMMAND (correspondence), 323
\DATE backslash command, 340, 344
\ELSE (correspondence), 327
\ELSEIF (correspondence), 327
\ENDIF (correspondence), 327
\ENDDOCUMENT (correspondence), 328
<table>
<thead>
<tr>
<th>Command</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>\ENTRY1-9</td>
<td>340, 366</td>
</tr>
<tr>
<td>\FF (correspondence)</td>
<td>335</td>
</tr>
<tr>
<td>\FI (correspondence)</td>
<td>329</td>
</tr>
<tr>
<td>\FILE (correspondence)</td>
<td>336</td>
</tr>
<tr>
<td>\FILE backslash command</td>
<td>340</td>
</tr>
<tr>
<td>\FILL (correspondence)</td>
<td>324, 329</td>
</tr>
<tr>
<td>\FMT (correspondence)</td>
<td>324, 326</td>
</tr>
<tr>
<td>\FORMFEED (correspondence)</td>
<td>335</td>
</tr>
<tr>
<td>\F (correspondence)</td>
<td>327</td>
</tr>
<tr>
<td>\IN (correspondence)</td>
<td>331</td>
</tr>
<tr>
<td>\INDENT (correspondence)</td>
<td>331</td>
</tr>
<tr>
<td>\JUSTIFY (correspondence)</td>
<td>325, 331</td>
</tr>
<tr>
<td>\LABEL (correspondence)</td>
<td>328</td>
</tr>
<tr>
<td>\LENGTH (correspondence)</td>
<td>324, 333, 335</td>
</tr>
<tr>
<td>\LINELENGTH (correspondence)</td>
<td>329</td>
</tr>
<tr>
<td>\LL (correspondence)</td>
<td>329</td>
</tr>
<tr>
<td>\LM (correspondence)</td>
<td>324, 331</td>
</tr>
<tr>
<td>\LMARGIN (correspondence)</td>
<td>324, 331</td>
</tr>
<tr>
<td>\NA (correspondence)</td>
<td>325, 331</td>
</tr>
<tr>
<td>\NFF (correspondence)</td>
<td>335</td>
</tr>
<tr>
<td>\NFMT (correspondence, 326</td>
<td></td>
</tr>
<tr>
<td>\NFMT (correspondence)</td>
<td>324</td>
</tr>
<tr>
<td>\NOADJUST (correspondence)</td>
<td>325</td>
</tr>
<tr>
<td>\NOFILL (correspondence)</td>
<td>324, 329</td>
</tr>
<tr>
<td>\NOFORMAT (correspondence)</td>
<td>324, 326</td>
</tr>
<tr>
<td>\NOFORMFEED (correspondence)</td>
<td>335</td>
</tr>
<tr>
<td>\NOJUSTIFY (correspondence)</td>
<td>325, 331</td>
</tr>
<tr>
<td>\NOVERT (correspondence)</td>
<td>325, 329, 331</td>
</tr>
<tr>
<td>\NV (correspondence)</td>
<td>325, 331</td>
</tr>
<tr>
<td>\PAGE (correspondence)</td>
<td>325, 333</td>
</tr>
<tr>
<td>\PL (correspondence)</td>
<td>324, 333</td>
</tr>
<tr>
<td>Index</td>
<td></td>
</tr>
<tr>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>$CALLHIST function, 1935, 1945</td>
<td></td>
</tr>
<tr>
<td>$CALLHIST screen, 1954</td>
<td></td>
</tr>
<tr>
<td>$CALLID field, 1072</td>
<td></td>
</tr>
<tr>
<td>$CDATE field, 320, 985</td>
<td></td>
</tr>
<tr>
<td>$CMONTH field, 320, 985</td>
<td></td>
</tr>
<tr>
<td>$CORRID field, 320</td>
<td></td>
</tr>
<tr>
<td>$DATE field, 320, 321, 985</td>
<td></td>
</tr>
<tr>
<td>$DAY field, 320, 321, 985</td>
<td></td>
</tr>
<tr>
<td>$DBERRNO field, 732, 748, 1346</td>
<td></td>
</tr>
<tr>
<td>$DOW field, 320, 321, 985</td>
<td></td>
</tr>
<tr>
<td>$END function, 709, 1934, 1936, 1942, 1947, 1949</td>
<td></td>
</tr>
<tr>
<td>$END screen, 1955, 1991</td>
<td></td>
</tr>
<tr>
<td>$ENTRY field, 1774</td>
<td></td>
</tr>
<tr>
<td>$ERR screen, 1957</td>
<td></td>
</tr>
<tr>
<td>$ERROR.MSG, 734</td>
<td></td>
</tr>
<tr>
<td>$EVENTINDEX field, 1072</td>
<td></td>
</tr>
<tr>
<td>$FIELDNAME field, 1774</td>
<td></td>
</tr>
<tr>
<td>$FIRSTNAME field, 230</td>
<td></td>
</tr>
<tr>
<td>$GETID screen, 1958</td>
<td></td>
</tr>
<tr>
<td>$GETNEXT screen, 1960</td>
<td></td>
</tr>
<tr>
<td>$INDEX field, 321</td>
<td></td>
</tr>
<tr>
<td>$INEND function, 1949</td>
<td></td>
</tr>
<tr>
<td>$INEND screen, 1960</td>
<td></td>
</tr>
<tr>
<td>$INERR screen, 1961</td>
<td></td>
</tr>
<tr>
<td>$INPCH, 76</td>
<td></td>
</tr>
<tr>
<td>$INWAIT screen, 1962</td>
<td></td>
</tr>
<tr>
<td>$LASTNAME field, 230</td>
<td></td>
</tr>
<tr>
<td>$LCALLIDS field, 1072</td>
<td></td>
</tr>
<tr>
<td>$LEVENTS field, 1072</td>
<td></td>
</tr>
<tr>
<td>$LOGICERR field, 1040</td>
<td></td>
</tr>
<tr>
<td>$LOGICLINE field, 1041</td>
<td></td>
</tr>
<tr>
<td>$LOGICNAME field, 1041</td>
<td></td>
</tr>
</tbody>
</table>
Index

<table>
<thead>
<tr>
<th>Field/Screen</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>$MAIN screen</td>
<td>1963</td>
</tr>
<tr>
<td>$MONTH field</td>
<td>320, 321, 985</td>
</tr>
<tr>
<td>$NONE screen</td>
<td>1965</td>
</tr>
<tr>
<td>$OLDQUEUE field</td>
<td>1072</td>
</tr>
<tr>
<td>$OLDRESULT field</td>
<td>1072</td>
</tr>
<tr>
<td>$OP screen</td>
<td>1966</td>
</tr>
<tr>
<td>$OPNAME field</td>
<td>230</td>
</tr>
<tr>
<td>$PAGE field</td>
<td>321</td>
</tr>
<tr>
<td>$PCBDATE field</td>
<td>1072</td>
</tr>
<tr>
<td>$PCBTIME field</td>
<td>1072</td>
</tr>
<tr>
<td>$PDINEND function</td>
<td>1949</td>
</tr>
<tr>
<td>$PDINWAIT screen</td>
<td>1968</td>
</tr>
<tr>
<td>$PM screen</td>
<td>1969</td>
</tr>
<tr>
<td>$PMRG screen</td>
<td>1971</td>
</tr>
<tr>
<td>$PREV.ENTRY field</td>
<td>1775</td>
</tr>
<tr>
<td>$PROJECT field</td>
<td>321</td>
</tr>
<tr>
<td>$QVIEW screen</td>
<td>1972, 1989</td>
</tr>
<tr>
<td>$READ.OK field</td>
<td>748, 1444</td>
</tr>
<tr>
<td>$RET function</td>
<td>1608, 1936</td>
</tr>
<tr>
<td>$RPAGE field</td>
<td>321</td>
</tr>
<tr>
<td>$SELNO field</td>
<td>750, 1634</td>
</tr>
<tr>
<td>$SELTXT field</td>
<td>750, 1634</td>
</tr>
<tr>
<td>$TIME field</td>
<td>321</td>
</tr>
<tr>
<td>$VAL field</td>
<td>1775</td>
</tr>
<tr>
<td>$YEAR field</td>
<td>320, 321, 985</td>
</tr>
</tbody>
</table>

Numerics

- 24-hour clock, 154
- 3270-PLAY verb, 1023
A
A (EQL reserved word), 394
Absolute value function (CALCULATE verb), 2055
Accelerator key, 1665, 1670
ACL
  accessing, 459
  command line editing, 462
  commands, 459
  commands to invoke help, 469
  documentation conventions, 473
  edit mode, 463
  editing, 463, 464
  EQL, 381
  exiting, 459, 545
  history stack, 462
  history stack commands, 464
  insert mode, 463
  line edit commands, 464
  online help, 469
  security features, 471
  task scheduler log, 995–996
  wildcards, 461
ACL commands
  alphabetical listing, 473
  ANALYZELOGIC, 476
  ARCHIVE, 479
  CHANGEPTR, 481
  CHECKFILE, 483
  chkdb2xref, 485
chkxref2db, 487
CLEANTMP, 489
CLEAR-FILE, 491
CLEARUSER, 492
compact, 494
COPY, 498
copy.proj, 503
COPYLIST, 501
CORRPRINT, 506
COUNT, 508
crestore, 510
ctreecp, 512
ctrlchar, 514
DELETELIST, 516
DELHELP, 517, 564
DELREC, 518
DELRECS, 520
DUMP, 521
dump_shadow, 523
edgeio, 524
EDIT, 527
EDITHELP, 469, 532, 564
EDITLIST, 533
EXECTASK, 534
filecheck, 536
FILEINFO, 541
FIXFILE, 543
GOTO, 545
INITIALMON, 546
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST</td>
<td>548</td>
</tr>
<tr>
<td>LISTFILES</td>
<td>550</td>
</tr>
<tr>
<td>LISTLISTS</td>
<td>552</td>
</tr>
<tr>
<td>LOOKUP</td>
<td>554</td>
</tr>
<tr>
<td>LSREC</td>
<td>557</td>
</tr>
<tr>
<td>LSTAPE</td>
<td>558</td>
</tr>
<tr>
<td>LSTAPELOCKS</td>
<td>559, 590</td>
</tr>
<tr>
<td>MOVE</td>
<td>560</td>
</tr>
<tr>
<td>MULTI.ASSEM</td>
<td>562</td>
</tr>
<tr>
<td>PASSWD</td>
<td>563</td>
</tr>
<tr>
<td>PRINTMON</td>
<td>565</td>
</tr>
<tr>
<td>procllook</td>
<td>569</td>
</tr>
<tr>
<td>Qclear</td>
<td>572</td>
</tr>
<tr>
<td>Qdelete</td>
<td>574</td>
</tr>
<tr>
<td>Qrebuild</td>
<td>576</td>
</tr>
<tr>
<td>Qseed</td>
<td>581</td>
</tr>
<tr>
<td>Qtrans</td>
<td>584</td>
</tr>
<tr>
<td>RMREC</td>
<td>589</td>
</tr>
<tr>
<td>RMTAPELOCKS</td>
<td>590</td>
</tr>
<tr>
<td>SECURE</td>
<td>591</td>
</tr>
<tr>
<td>SELECT</td>
<td>593</td>
</tr>
<tr>
<td>set_tracking</td>
<td>595</td>
</tr>
<tr>
<td>SETSTDERR</td>
<td>594</td>
</tr>
<tr>
<td>SHOWSECURITY</td>
<td>597</td>
</tr>
<tr>
<td>smart_merge</td>
<td>599</td>
</tr>
<tr>
<td>TAPEFWD</td>
<td>601</td>
</tr>
<tr>
<td>TAPERead</td>
<td>602</td>
</tr>
<tr>
<td>TAPERew</td>
<td>603</td>
</tr>
<tr>
<td>UNARCHIVE</td>
<td>604</td>
</tr>
</tbody>
</table>
vercont, 607
vfiles, 611
XQUERY, 613
xrefall, 615
Activation logic, 1615
Active tab data, 1731
ActiveX controls
  archiving, 443
  CALL-EXTERNAL-FUNCTION, 1096–1099
  colors, 1871–1876
  configuring, 1827
  events, 746, 1837
  GEO, 1880
  libraries, 1909, 1912
  methods, 1096, 1838
  object properties sheet, 1863
  properties, 1243, 1364
  properties tab, 1833–1837
  registering, 1825
  wrappers, 1841–1861
ACTZ file
  add update definition to ACTZ file, 162
  adding records, 166
  clearing error logs, 165
  clearing records, 164
  time zones, 191
  updating single entry, 157
  updating with file, 160
  validating telephone numbers, 157
Ad hoc batch reports
creating, 373
executing from ACL, 613

Ad hoc reporting
archiving, 443
backslash commands, 339
batch reports, 373–376
column display, 349–350
column headings, 640
columnar reports, 338, 349
creating, 345
date offset, 984
display options, 350–353
EQL, 381
executing from ACL, 613
form report, 338
freeform report, 338
heading/footing, 344
headings/footings, 355
justification of fields, 641
label report, 338
label setup, 356–357
listing, 372
log, 183
modifying, 345
output device setup, 357–359
record selection, 359–364
report display, 364–365
report name, 347
report type, 347
running, 368
selection lists, 338, 377
setting defaults, 341
setting report properties, 348
source file, 347
specifying site, 194
title/prompts setup, 365–366
types, 338
ADD verb, 1024
AdvantEDGE Command Language, see ACL.
Aligning objects, 1647
Alphanumeric fields
defining, 634
formatting options, 636
ANALYZELOGIC command, 476
AND (EQL reserved word), 394
AND-LISTS verb, 1026
AppActivate command, 1480
APPEND verb, 1028
APPEND-VALUE verb, 1030
Arc cosine function (CALCULATE verb), 2084
Arc sine function (CALCULATE verb), 2085
Arc tangent function (CALCULATE verb), 2085
Archive
begin archive, 445
defining an instruction, 435–444
from ACL, 479
what can be archived, 443–444
see also Unarchive.
ARCHIVE command, 479
Index

Areas
  quick reference, 63
  version control, 103
  window in browser, 37
Arithmetic expression relationship, 662
Array function (CALCULATE verb), 2061
ASCII
  keyboard commands, 2112–2115
  screens, 1936–1941
ASCII screens
  $AUTOEND, 1951
  $CALLBACK, 1952
  $CALLHIST, 1954
  $END, 1955
  $ERR, 1957
  $GETID, 1958
  $GETNEXT, 1960
  $INEND, 1960
  $INERR, 1961
  $INWAIT, 1962
  $MAIN, 1963
  $NONE, 1965
  $OP, 1966
  $PDINWAIT, 1968
  $PM, 1969
  $PMRG, 1971
  $QVIEW, 1972
  alphabetical listing, 1951–1978
ASCIIFILE (EQL reserved word), 388, 394
ASK verb, 1032
Assemblies
- default file, 1573, 1992
- full, 2015
- guide watcher, 2034–2036
- listing errors, 2023
- multiple projects, 1919–1924
- multiproject assemble, 562
- tasks, 971–972
- with version control, 106

Assigning password to command, 563
Assigning security levels, 471, 591
ASSIGN-NEW-CALL-ID verb, 707, 1035
AT&T 6500 cluster controller, 204
Audio object, description, 1659
Auto arrange of screen objects, 1603
AVERAGE (EQL reserved word), 394
Average argument (CALCULATE verb), 2077

B
B (EQL reserved word), 394
backgrnd directory, 2123
Backing up EDGE components, *see Archive.*
Backslash commands
- ad hoc reporting, 339
- correspondence, 323–336
Base year, 193
Batch reports, 373–376
bbin directory, 2126
BCKGRNDOUT environment variable, 2102
Bessel functions (CALCULATE verb), 2066
BETWEEN (EQL reserved word), 395
bin directory, 2123, 2126, 2128
Block size
   archiving, 436
   exporting, 798
   importing, 774
Browser
   icon view, 35
   name view, 36
BS_KEY environment variable, 1370, 2102
bufferSetProperty environment variable, 2102
Building cross-reference file, 615
Bulletins, 1981–1984
BY (EQL reserved word), 395
Bypass EDGE user login, 147

C
C (EQL reserved word), 395
Cache setting, 138
Caching (counters), 69
CALC_LOG_PATH environment variable, 2102
CALCULATE verb
   functions, 2053–2089
   keypad, 1042
   operators, 1057
   order of operations, 1051
   precision, 1040
   status codes, 1041
   testing values, 1059
   verb, 1037
Calendar
creating, 81
using, 74
with system arranged results, 78
see also Work schedules.

Call Center screen, 1942

Call events
archiving, 444
assigning with counters, 68
call history, 1946, 1954
clearing, 572
components, 706
count, 903
deleting, 572, 885–889
deleting from queues, 584
generating, 707–708
in queues, see Queues.
removing from queues, 842–847, 855–856
seeding, 581
transferring from queues, 584, 895–898
unarchiving, 448

Call IDs
change value, 1101
locked, 727
system assigned, 1036

CALL verb, 1061, 1063

Callbacks
$CALLBACK function, 1943
$CALLBACK screen, 1952
$QVIEW screen, 1972
$QVSEL screen, 1975
in HISTORY file, 712
last day to schedule, 86
last scheduled date (SYS file), 738
scheduling, 1945
setting up preferred calling hours, 74–78
system arrangement, 2009
view callbacks, 1943

**CALL-EVENT verb**
cancel scheduled events command, 1074
check for available call command, 1075
create call events command, 1076
create call IDs command, 1078
delete call events command, 1079
delete call IDs command, 1080
deselect call event command, 1081
get next call event command, 1084, 1093, 1094, 1095
locate call events command, 1085
lock call events command, 1086
lock call IDs command, 1087
schedule call events command, 1089
select call events command, 1093
status codes, 1073
unlock call events command, 1094
unlock call IDs command, 1095
verb, 707, 841, 1067

**CALL-EXTERNAL-FUNCTION verb**, 1096–1099, 1916

Calling codes, see Country codes.
CallTimer command, 1491

Capitalization rules
data entry, 1990
for data entry in EDGE Client, 172
Case sensitivity, 32
Case-sensitive member-of-set, 171
cbin directory, 2126
CF_TEXT, 1454
CHANGE-CALL-ID verb, 1101
CHANGED file, 2041
CHANGEPTR command, 481
Changing printer
  server default printer, 200
  server printer for current login session, 58, 481
Characters
  converting, 1103
  counting, 1403
  extracting, 1415
  removing spaces, 1423
  searching for, 554
Check box
  description, 1662
  invoking logic flows, 1776
CHECKFILE command, 483
Checking file, 483, 536
Checking for control characters, 514
chkdb2xref command, 485
chkxref2db command, 487
Clauses (EQL)
  form options clause, 408
  general options clause, 408–410
  grand total clause, 429–431
  group clause, 411–413
Index

headings and footings clause, 413–415
label options clause, 415–416
noformat options clause, 416–418
output clause, 418–419
selection clause, 419–423
sort clause, 423–425
source clause, 406–408
target clause, 406, 425–428
total clause, 428
Clean EDGE/tmp directory, 185
CLEANTMP command, 489
CLEAR-FILE command, 491
Clearing
call events, 572
files, 491
files (importing), 774
processing errors, 793, 812
records, 1106
user IDs, 492
CLEARUSER command, 492
Client setup, 136
Client/server applications, 1449
Code location (URL), 1824
Coded project selection, 128–132
Codes (correspondence), 310–319
Cold link (DDE), 1477
Color and fonts, 1752
Colors
ActiveX controls, 1871–1876
BGR (ActiveX controls), 1872

EDGE 7.11 Developer Reference Manual

2157


Index

ETW_W toolbar, 40
graph objects, 1741
logic editor, 941
RGB, 1929
schemes, 1928, 1929
Colors in EDGE Client, 176
Columnar reports
  ad hoc reporting, 338, 349
  EQL, 383
COMBINE verb, 1108
Command button labels in EDGE Client, 173
Command buttons
  accelerator key, 1665, 1670
  description, 1665
  invoking logic flows, 1776
Command line editing, 462
COMMUNICATE-(RECEIVE) verb, 1110
Communication setup, 204
compact command
  condensing files, 494
  restoring backups, 510
Compact file, 697–700
Comparison rules, 927
Comparisons, correspondence, 311
Compiled logic flows, 2128
COMPOUND-INTEREST verb, 1112
CONCAT function (CALCULATE verb), 2081
Condensing files, 494
Condition terms
  COMPARE, 929
  COMPARE-MULTI-VALUES, 931
comparison rules, 927
MEMBER-OF-SET, 934
PATTERN, 935
RANGE, 938
Configuring ActiveX controls, 1827
Conjunction terms
   AND, 924
   OR, 926
Connecting to EDGE Developer–W, 27
Connection setup
   new connection, 146
   server login, 144
   setup files, 144
   socket setup, 206
   telnet setup, 144, 145
CONSOLE environment variable, 2103
Constructing statements
   condition terms, 927
   conjunction terms, 924
   EDGE statements, 915
Containers, see Wrappers.
Control characters
   checking for, 514, 696
   removing, 697
CONVERT verb, 1114
COOKIE verb, 1117
COPY command, 498
COPY verb, 1118
copy.proj command, 503
Copying
Index

fields, 674–676
lists, 501
project components, 503
records, 498, 512
COPYLIST command, 501
Correspondence
accessing, 305
archiving, 444
creating, 308–309
date format codes, 311
date offset, 984
default line command settings, 324
field format codes, 310–319
field formatting commands, 326
fields, 308
form feed commands, 335
group extract codes, 313
IF-ELSE commands, 326
justification and length codes, 315
label formatting commands, 328
line commands, 322–336
line formatting commands, 329
margin setup commands, 330
mask character codes, 316
multivalue fields, 308, 332
overview, 303
page setup commands, 332
printer escape commands, 333
printer operation commands, 333–335
printing, 307, 506, 1317
read file commands, 336
system fields, 320–321
text extract codes, 318
CORRPRINT command, 506
CORRPRINT_TRUNC_AS_EQL environment variable, 304, 2103
Cosine function (CALCULATE verb), 2086
COUNT (EQL reserved word), 395
Count arguments (CALCULATE verb), 2078
COUNT command, 508
Count report (EQL), 385–386
COUNT verb, 1122
COUNTER verb
  create counter command, 1128
  current counter command, 1131
  decrement counter command, 1133
  delete counter command, 1134
  increment counter command, 1134
  platform restrictions, 1130
  status codes, 1126
Counters
  archiving, 443
  caching, 69
  creating, 71
  creating a project counter, 68
  cycling, 70
  DBL type counters, 73
decrementing, 70
deleting, 68
direction, 73
Index

EDGE_COUNTERS file, 67
file for counter names, 67
in relational field definition, 71
incrementing, 73
limits, 67
offsets, 70
project, 68, 180
system-assigned call events, 68
using, 67
Counting records, 508
Country calling codes, see Country codes.
Country codes
   add update definition to ACTZ file, 162
current (SYS file), 732
define new code, 150–157
updating ACTZ file, 157
updating with a file, 160
COUNT-VALUES verb, 1124
Creating an object, 1655
Creating online help, 532
crestore command, 510
Cross-reference button, 1620
Cross-reference files
   building, 615, 623
   checking, 485, 487
   creating or modifying, 683
deleting, 624
deleting records, 518
displaying record ID information, 1173
querying, 654
soundex codes, 683, 684
using to validate data, 654
X_filename, 2121
CSIZE environment variable, 2103
c-tree files
default database device, 2120
structure, 2120
using in SQL-SELECT verb, 1398
creecp command, 512
ctrlchar command, 514
cursor movement, 642
customer care, 8
customer preferred calling hours, see Preferred calling hours.
custom-subdirectory, 2123

D
D (EQL reserved word), 395
Data
exporting, 524
importing, 524
receiving, 599
required entry validation, 657
required value for write, 658
sending, 599
validating, 647–658
DATA (EQL reserved word), 396
Data entry case conversion, 1990
Data entry text conversion, 172
Data format
exporting, 798
<table>
<thead>
<tr>
<th>Database device name</th>
<th>181</th>
</tr>
</thead>
<tbody>
<tr>
<td>Database maintenance</td>
<td></td>
</tr>
<tr>
<td>checking for file corruption</td>
<td>700–704</td>
</tr>
<tr>
<td>checking or removing control characters</td>
<td>696–697</td>
</tr>
<tr>
<td>compacting a file</td>
<td>697–700</td>
</tr>
<tr>
<td>Database-created objects</td>
<td></td>
</tr>
<tr>
<td>views</td>
<td>1443–1447</td>
</tr>
<tr>
<td>Databases</td>
<td></td>
</tr>
<tr>
<td>archiving</td>
<td>443</td>
</tr>
<tr>
<td>available</td>
<td>620</td>
</tr>
<tr>
<td>default</td>
<td>620</td>
</tr>
<tr>
<td>DATA-CONFERENCE verb</td>
<td>1135</td>
</tr>
<tr>
<td>DATA-TRANSFER verb</td>
<td>1140</td>
</tr>
<tr>
<td>DATE command</td>
<td>985</td>
</tr>
<tr>
<td>Date fields</td>
<td></td>
</tr>
<tr>
<td>current (SYS file)</td>
<td>732</td>
</tr>
<tr>
<td>current date at customer site</td>
<td>751</td>
</tr>
<tr>
<td>defining</td>
<td>634</td>
</tr>
<tr>
<td>formatting options</td>
<td>636</td>
</tr>
<tr>
<td>minimum values</td>
<td>634</td>
</tr>
<tr>
<td>range validation</td>
<td>656</td>
</tr>
<tr>
<td>Date formats</td>
<td></td>
</tr>
<tr>
<td>correspondence</td>
<td>311</td>
</tr>
<tr>
<td>country codes</td>
<td>152</td>
</tr>
<tr>
<td>internal</td>
<td>634</td>
</tr>
<tr>
<td>DATE-FORMAT verb</td>
<td>1145</td>
</tr>
<tr>
<td>Dates</td>
<td></td>
</tr>
</tbody>
</table>
base year, 193
date offset (correspondence), 320
date offset (task scheduler), 984
daylight savings time, 154
invalid, 2108
order to display, 152
separator, 152
Daylight savings time, 154
DDE
  client/server applications, 1449
  concepts, 1450
  data types, 1454
  GEO as server, 1477–1497
  GEO capabilities, 1448
  links, 1452
DDE server
  AppActivate command, 1480
  CallTimer command, 1491
  commands for execute statements, 1479
  commands for request statements, 1489
  Enter command, 1480
  FocusCaption command, 1491
  FocusContent command, 1491
  FocusFormattedValue command, 1491
  FocusName command, 1492
  FocusPosition command, 1492
 _FOCUSType command, 1492
  FocusValue command, 1493
  initiating a conversation, 1478
  ObjectCount command, 1493
Port command, 1494
ProjectName command, 1494
result codes for Execute statement, 1494
Result command, 1494
ScreenName command, 1495
ScreenTitle command, 1495
SendKeys command, 1481–1485
SetTargetContent command, 1486
Tab command, 1486
TabObjectCount command, 1496
Target command, 1487
TargetCaption command, 1496
TargetContent command, 1496
TargetFormattedValue command, 1496
TargetName command, 1496
TargetPosition command, 1496
TargetType command, 1497
TargetTypeName command, 1497
TargetValue command, 1497
types of links, 1477
UserID command, 1497
UserName command, 1497
DDE verb
Advise command, 1461
configuration limits, 1454
definition, 1456
Execute command, 1465
initiating a conversation, 1458
list of commands, 1456
maximums, 1454
Poke command, 1467
Request command, 1469
Server property, 1457
Status command, 1471
store status in property, 1458
Terminate command, 1473
Unadvise command, 1475
dde_server, 1477–1497
Deactivation logic, 1616
Debug mode, 1153
DEBUG verb, 1152–1160
Decimal separator, 155
Decrementing counters, 70
DECRYPT verb, 1161
Default ad hoc report settings, 341
Default file
guide assembly, 1573
in logic flows, 951
screen, 1573, 1604
Default line command settings (correspondence), 324
Default project counter, 68
DEFAULT scheme, 1586, 1925
Default server printer setup, 202
Defining an I/O device, 195
Defining expressions
   EDGE Developer setup, 1749
   using SET-PROPERTY verb, 1366
Delete column or row in table object, 1716
DELETELIST command, 516
DELETE-RECORD verb, 1165
DELETE-VALUE verb, 1168
Deleting
   ActiveX controls, 1826
call events, 572, 584
counters, 68
field definitions, 518, 676
online help, 517
projects, 58
queue definition, 574
records, 518, 1165
selection list, 516
   value from multivalued field, 1168
DELHELP command, 517, 564
DELREC command, 518
DELECS command, 520
DESC (EQL reserved word), 396
Description of screen objects, 1657–1720
DET-SUP (EQL reserved word), 396
Devices
   listing locks for I/O, 559
   removing tape locks, 590
DICT (EQL reserved word), 396
dict_d directory, 2120
dict_x directory, 2120
DIGEST verb, 1169
Directing output to another destination, 594
DISPLAY (EQL reserved word), 396
Display field object, description, 1672
DISPLAY-CROSS-REFERENCE verb, 1173
Displaying
   device records, exporting, 812
device records, importing, 794
file information, 541
files in current account, 550
files in specified directory, 550
I/O devices, 558
options (ad hoc reporting), 350
record IDs, 548
records on screen, 521
security level, 597
UNIX processes, 569
DISPLAY-VALUES verb, 1176
DIVIDE verb, 1179
DLIST (EQL reserved word), 396
Docking a toolbar, 1888
Dot commands, see Screen commands.
Downloading ActiveX controls, 1824
DSPLST (EQL reserved word), 396
DUMP (EQL reserved word), 397
DUMP command, 521
Dump report (EQL), 386–387
dump_shadow command, 523
During paint logic, 1617
Dynamic data exchange, see DDE.

E
E (EQL reserved word), 397
EDGE
    guide area, 1903–1924
temporary work directory, 489
EDGE ACL commands, see ACL commands.
EDGE base year, 193
EDGE Client labels, 173
EDGE Client options
  allow access modifications, 170
  allow label modifications, 171
  allow new window creation, 171
  allow scheme modifications, 171
  allow setup modifications, 171
  case-sensitive member-of-set, 171
  data entry text conversion, 172
  menu items access, 173
EDGE Client scheme, 176
EDGE Customer Care, 8
EDGE Developer
  areas and modules, 63
  Browser, 34
  connect screen, 28
  connection, 27
  log off, 29
  log on, 28
  menus, 40
  requirements, 27
  terminology, 30
  toolbar, 39
  user interface, 34
EDGE directories and files, 2118
EDGE directory structure
  EDGE, 2117, 2123
  EDGESYS, 2117, 2126
  STEMPLATE, 2117, 2129
  TEMPLATE, 2117, 2128
Index

EDGE files
   F_filename, 2120
   HISTORY file, 710–712
   MD, 2121
   modifying, 708
   PFILES, 2121
   PROJ.PARAM, 2121
   record IDs, 706–708
   REPORTING file, 708, 713–716
   RESPONSES file, 717–718
   STATUS file, 719–726
   STATUSID file, 727
   SYS file, 728–767
   TEMP file, 768

EDGE functions
   $AUTOEND, 1942
   $CALLBACK, 1943
   $CALLHIST, 1945
   $END, 1934, 1947
   $INEND, 1949
   $PDINEND, 1949
   defined, 1935
   list, 1935

EDGE ListView control, 1842–1852
EDGE login, bypass, 147
EDGE objects See Screen objects.
EDGE Product Support, see Portrait Software Customer Care.
EDGE Query Language see EQL.
EDGE screens
   archiving, 443
defined, 1935
list, 1935
modifying for ASCII, 1936–1941
modifying for GEO, 1936
see also EDGE functions.
EDGE setup, 135
EDGE statements
  ALWAYS, 915
  COMMENT, 916
  ELSE, 923
  IF, 920
  LABEL, 918
  THEN, 922
EDGE tmp directory, 2125
EDGE TreeView control, 1853–1859
EDGE_COUNTERS file, 67
edge_country file, 2124
EDGEDTABLE environment variable, 2103
dgeo command, 524
EDGESYS directory
  bbin directory, 2126
  bin directory, 2126
  cbin directory, 2126
  etc directory, 2126
  include directory, 2126
  src directory, 2127
  terminfo directory, 2127
EDGEULIMIT environment variable, 2104
EDIT command, 527
Edit menu, 55
EDITHELP command, 469, 532, 564
Editing
  command line, 464
  freeform assisted mode, 945
  full-screen commands, 528
  property assisted mode, 942
  quick edit mode, 947
  records, 527
  selection list, 533
EDITLIST command, 533
Effective continuous interest rate (CALCULATE verb), 2067
Element function (CALCULATE verb), 2062
EMAIL verb, 1182
Embedded expressions
  archiving, 443
EMPTY_AS_ZERO environment variable, 2104
ENABLE_RDB_TRACE environment variable, 2104
Enabling record tracking, 595
ENCRIPT verb, 1183
End of call processing, 1947, 1955
Enter command, 1480
Entry logic execution order, 1618
Entry to screen logic, 1617
Entry to screen logic flow, 1634
Environment variables
  BCKGRNDOUT, 2102
  BS_KEY, 1370, 2102
  bufferSetProperty, 2102
  CALC_LOG_PATH, 2102
  CONSOLE, 2103
  CORRPRINT_TRUNC_AS_EQQ, 304, 2103
Index

- CSIZE, 2103
- EDGEDTABLE, 2103
- EDGEULIMIT, 2104
- EMPTY_AS_ZERO, 2104
- ENABLE_DBDEVICEDELETION, 2104
- ENABLE_RDB_TRACE, 2104
- EQL_CASE_SENSITIVE, 2104
- ETL1, 2104
- F_AUTO_RETURN, 642, 2104
- F_EXEC_LOGIC, 2104
- GEOHEARTBEAT, 2104
- GUIDE_WATCHER, 2105
- IGNORE_OVERLAPPING, 2105
- IGNORE_SPECIAL_CHARS, 2105
- in EQL, 394
- INTERACTIVE, 2105
- IO_LOG_FILE, 2105
- LANG, 2105
- LOGON_NO_VALIDATION, 2106
- MAX_LOGFILE_SIZE, 2106
- MAX_XREF_MATCH_KEYS, 2106
- MSGQ_SKIP_SRVID_CHECK, 1552, 2107
- NO_BEEPS_PLEASE, 2107
- QSEED_STATUS_NOWAIT, 2107
- SET_DENTRY_NULL, 2107
- STAT_INTERVAL, 2108
- SUPERSRVR_LOG_DEBUG, 2108
- SUPERSRVR_LOG_TRACE, 2109
- TANDEM_TOGGLE, 2109
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPHNYCONSULT_PATH, 2109</td>
</tr>
<tr>
<td>TPHNYLOG_PATH, 2110</td>
</tr>
<tr>
<td>UPPER, 2110</td>
</tr>
<tr>
<td>USEKEYPAD, 2110</td>
</tr>
<tr>
<td>V_CALC_STACK_SIZE, 2110</td>
</tr>
<tr>
<td>VERCONT_PATH, 2111</td>
</tr>
<tr>
<td>VERCONT_PGM, 2111</td>
</tr>
<tr>
<td>EQ (EQL reserved word), 397</td>
</tr>
<tr>
<td>EQL</td>
</tr>
<tr>
<td>ACL, 381</td>
</tr>
<tr>
<td>ad hoc reporting, 381</td>
</tr>
<tr>
<td>columnar reports, 383</td>
</tr>
<tr>
<td>count report, 385–386</td>
</tr>
<tr>
<td>defined, 381</td>
</tr>
<tr>
<td>dump report, 386–387</td>
</tr>
<tr>
<td>environment variables, 394</td>
</tr>
<tr>
<td>EQL_CASESENSITIVE, 394</td>
</tr>
<tr>
<td>form options clause, 408</td>
</tr>
<tr>
<td>form report, 387–389</td>
</tr>
<tr>
<td>general options clause, 408–410</td>
</tr>
<tr>
<td>grand total clause, 429–431</td>
</tr>
<tr>
<td>group clause, 411–413</td>
</tr>
<tr>
<td>headings and footings clause, 413–415</td>
</tr>
<tr>
<td>identifiers, 392</td>
</tr>
<tr>
<td>label options clause, 415–416</td>
</tr>
<tr>
<td>label report, 389–390</td>
</tr>
<tr>
<td>noformat options clause, 416–418</td>
</tr>
<tr>
<td>noformat report, 391</td>
</tr>
<tr>
<td>notation, 382</td>
</tr>
</tbody>
</table>
output clause, 418–419
relational operators, 420
report types, 383–391
reserved words, 394–404
selection clause, 419–423
sort clause, 423–425
sorting, 423–425
source clause, 406–408
target clause, 406, 425–428
total clause, 428
EQL statement
producing, 593
EQL_CASE_SENSITIVE environment variable, 394, 2104
Error limit
exporting, 799
importing, 775
Error logs
clearing, 165
exporting, 799
importing, 776
SmartMerge, 824, 829
Error messages
system (SYS file), 750
user defined, 734
user-defined on screen, 652
verifying logic flows, 958–959
etc directory, 2126
ETL1 environment variable, 2104
ETWGE0.INI file, 146
ETWW.INI file, 2130
<table>
<thead>
<tr>
<th>Event</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euclidean distance (CALCULATE verb)</td>
<td>2056</td>
</tr>
<tr>
<td>Events</td>
<td></td>
</tr>
<tr>
<td>ActiveX</td>
<td>1837</td>
</tr>
<tr>
<td>EDGE ListView control</td>
<td>1851</td>
</tr>
<tr>
<td>EDGE TreeView control</td>
<td>1858</td>
</tr>
<tr>
<td>screen</td>
<td>1576, 1612</td>
</tr>
<tr>
<td>EXECTASK command</td>
<td>534</td>
</tr>
<tr>
<td>EXECUTE verb</td>
<td>486, 488, 1189</td>
</tr>
<tr>
<td>EXECUTE-ADHOC verb</td>
<td>1195</td>
</tr>
<tr>
<td>EXECUTE-TASK verb</td>
<td>1200</td>
</tr>
<tr>
<td>Executing</td>
<td></td>
</tr>
<tr>
<td>ad hoc reports</td>
<td>613</td>
</tr>
<tr>
<td>task</td>
<td>534</td>
</tr>
<tr>
<td>Execution order of function keys</td>
<td>1599</td>
</tr>
<tr>
<td>Execution order of logics</td>
<td>1618</td>
</tr>
<tr>
<td>Exit from screen logic</td>
<td>1617</td>
</tr>
<tr>
<td>Exit from screen logic flow</td>
<td>1598</td>
</tr>
<tr>
<td>Exit logic execution order</td>
<td>1618</td>
</tr>
<tr>
<td>EXIT verb</td>
<td>1202</td>
</tr>
<tr>
<td>Exponential function (CALCULATE verb)</td>
<td>2056</td>
</tr>
<tr>
<td>Exporting</td>
<td></td>
</tr>
<tr>
<td>archiving instructions</td>
<td>443</td>
</tr>
<tr>
<td>block size</td>
<td>798</td>
</tr>
<tr>
<td>data</td>
<td>599</td>
</tr>
<tr>
<td>data format</td>
<td>798</td>
</tr>
<tr>
<td>displaying device records</td>
<td>812</td>
</tr>
<tr>
<td>error limit</td>
<td>799</td>
</tr>
<tr>
<td>field mark</td>
<td>799</td>
</tr>
<tr>
<td>instructions</td>
<td>797</td>
</tr>
<tr>
<td>listing log files</td>
<td>812</td>
</tr>
</tbody>
</table>
log files, 799
menu commands, 812
overview, 769
preparation, 770
record format, 800
record mark, 800
record size, 800
SmartMerge, 826–833
to device, 801
using edgeio, 524
Expressions
defining, 1749
interpreting, 1767
External DB override (SmartMerge), 834–835
External server communication, 1498
Extracting, GROUP-EXTRACT verb, 1249

F
F (EQL reserved word), 397
F_AUTO_RETURN environment variable, 642, 2104
F_EXEC_LOGIC environment variable, 2104
F_filename, 2120
FIELD (EQL reserved word), 397
Field format codes (correspondence), 310–319
Field formatting commands (correspondence), 326
Field help button, 1620
Field mark
eventing, 799
importing, 775
Fields
<table>
<thead>
<tr>
<th>Data Type</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advanced user-defined</td>
<td>661</td>
</tr>
<tr>
<td>Alphanumeric</td>
<td>634</td>
</tr>
<tr>
<td>Arithmetic expression</td>
<td>662</td>
</tr>
<tr>
<td>Assigning positions</td>
<td>670</td>
</tr>
<tr>
<td>Copying</td>
<td>674–676</td>
</tr>
<tr>
<td>Data types</td>
<td>633</td>
</tr>
<tr>
<td>Date</td>
<td>634</td>
</tr>
<tr>
<td>Default values</td>
<td>651</td>
</tr>
<tr>
<td>Defining</td>
<td>631–647</td>
</tr>
<tr>
<td>Deleting</td>
<td>518, 676</td>
</tr>
<tr>
<td>Determining</td>
<td>620</td>
</tr>
<tr>
<td>File translate</td>
<td>664</td>
</tr>
<tr>
<td>Formatting</td>
<td>635–640</td>
</tr>
<tr>
<td>Group extract</td>
<td>666</td>
</tr>
<tr>
<td>Length</td>
<td>642</td>
</tr>
<tr>
<td>Listing definitions</td>
<td>678</td>
</tr>
<tr>
<td>Maximum in a file</td>
<td>631</td>
</tr>
<tr>
<td>Member of set</td>
<td>655</td>
</tr>
<tr>
<td>Multivalues</td>
<td>643</td>
</tr>
<tr>
<td>Numeric</td>
<td>634</td>
</tr>
<tr>
<td>Position</td>
<td>635</td>
</tr>
<tr>
<td>Primary</td>
<td>646</td>
</tr>
<tr>
<td>Protected</td>
<td>644</td>
</tr>
<tr>
<td>Range validation</td>
<td>656</td>
</tr>
<tr>
<td>Record ID</td>
<td>646</td>
</tr>
<tr>
<td>Relational</td>
<td>659–669</td>
</tr>
<tr>
<td>Required entry validation</td>
<td>657</td>
</tr>
<tr>
<td>Required value for write</td>
<td>658</td>
</tr>
<tr>
<td>Secondary keys</td>
<td>644</td>
</tr>
<tr>
<td>Specifying Format</td>
<td>172</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----</td>
</tr>
<tr>
<td>Specifying Multivalued Fields</td>
<td>643</td>
</tr>
<tr>
<td>Synonyms</td>
<td>647</td>
</tr>
<tr>
<td>Time</td>
<td>634</td>
</tr>
<tr>
<td>Types</td>
<td>646</td>
</tr>
<tr>
<td>Using in Correspondence</td>
<td>308</td>
</tr>
<tr>
<td>Validating</td>
<td>634, 653</td>
</tr>
<tr>
<td>FIFO Queues</td>
<td>840, 856</td>
</tr>
<tr>
<td>File Checking Multiple Files</td>
<td>611</td>
</tr>
<tr>
<td>File Menu</td>
<td>57</td>
</tr>
<tr>
<td>File Translate Relationship</td>
<td>664</td>
</tr>
<tr>
<td>Filecheck Command</td>
<td>536</td>
</tr>
<tr>
<td>FILEINFO Command</td>
<td>541</td>
</tr>
<tr>
<td>Files</td>
<td></td>
</tr>
<tr>
<td>Abbreviations</td>
<td>966, 1985, 1986</td>
</tr>
<tr>
<td>Archiving</td>
<td>443</td>
</tr>
<tr>
<td>Building Cross-reference</td>
<td>615</td>
</tr>
<tr>
<td>CHANGED</td>
<td>2041</td>
</tr>
<tr>
<td>Checking</td>
<td>536</td>
</tr>
<tr>
<td>Checking for Control Characters</td>
<td>696</td>
</tr>
<tr>
<td>Checking for File Corruption</td>
<td>700</td>
</tr>
<tr>
<td>Checking Integrity (C-Tree Files Only)</td>
<td>483</td>
</tr>
<tr>
<td>Checking Multiple</td>
<td>611</td>
</tr>
<tr>
<td>Clearing</td>
<td>491</td>
</tr>
<tr>
<td>Compacting</td>
<td>494, 697</td>
</tr>
<tr>
<td>Compacting (C-Tree Files Only)</td>
<td>494</td>
</tr>
<tr>
<td>Copying Records</td>
<td>498, 512</td>
</tr>
<tr>
<td>Counting Records</td>
<td>508</td>
</tr>
<tr>
<td>Cross-referencing</td>
<td>485, 487, 1027</td>
</tr>
<tr>
<td>Default Assembly File</td>
<td>1573</td>
</tr>
</tbody>
</table>
default file for screens, 1573
default for tasks, 966
defining, 625–630
defining cross-reference, 683
deleting, 630
deleting records, 518, 520, 692
design considerations, 619
diagnostic information, 536
displaying information, 541
editing records, 527
exporting, 524
fixing corrupted (c-tree files only), 543
fragmentation, 697, 700–704
global files, 619, 626, 627
guide access, 1985
importing, 524
information, 541
listing, 550
listing record IDs, 548, 557
maximum per guide, 21, 1987
moving records, 560
name considerations, 32
pathname, 541
primary file, 1985, 1987
project, 619, 628
project access for global files, 627
reading automatically, 1574
rebuilding corrupted (c-tree files only), 543
records, 617
removing control characters, 697
searching for characters, 554
specifying for screens, 1572
stderr, 141, 594, 995, 1193
using to validate data, 653
FIND-VALUE verb, 1203
FIXFILE command, 543
FL (EQL reserved word), 397
FLAT-FILE-ACCESS verb, 1206
Focus, objects, 1656
FocusCaption command, 1491
FocusContent command, 1491
FocusFormattedValue command, 1491
FocusName command, 1492
FocusPosition command, 1492
FocusType command, 1492
FocusValue command, 1493
Fonts and color, 1752
Footing
  icons (ad hoc reporting), 344
FOOTING (EQL reserved word), 397
Footing, EQL clause, 413
FOPTIONS (EQL reserved word), 398
Form feed commands (correspondence), 335
Form options clause (EQL), 408
Form report, 338
Form report (EQL), 387–389
FORMAT (EQL reserved word), 398
FORMAT (EQL reserved words), 416
Format codes, 2090
FORMAT&CONCAT verb, 1215
FORMAT-INPUT verb, 1211
FORMAT-OUTPUT verb, 1213
Formatting
  export record, 800
  import records, 776
  numbers to currency, 1293
  with FORMAT&CONCAT verb, 1215
  with FORMAT-INPUT verb, 1211
  with FORMAT-OUTPUT verb, 1213
Forwarding tape, 601
FPP (EQL reserved word), 398
Freeform assisted mode, 945
Freeform report, 338
Freeform screens
  example, 1569
  overview, 1593
  sample screen, 1593
  uses for, 1570
FROM (EQL reserved word), 398
Full-screen editing commands, 528
Function key
  execution order, 1599
Function key layout, 1621
Function keys
  attaching logic flows to, 1599
  last used by operator, 735
  user screens, 1597
Future value (CALCULATE verb), 2068
Index

G
GE (EQL reserved word), 398
General options clause (EQL), 408–410
GEO
as DDE client, 1453–1476
as DDE server, 1477–1497
colors, 1928
DDE capabilities, 1448
EDGE screens, 1935
executing Windows application on client, 1190
invoking from EDGE Developer, 1905
GEO labels, see EDGE Client labels
GEO options, see EDGE Client options
GEO scheme, see EDGE Client scheme
GEO setup
font, 1929
styles, 1929
GEO.INI file, 1454, 1457
GEOHEARTBEAT environment variable, 2104
GET-CALL-RECORD verb, 1219
GET-CROSS-REFERENCE-DATA verb, 1221
GET-KEY verb
First, Next, and Previous Key commands with Cross-reference, 1233
First, Next, and Previous Key commands with Primary Key, 1231
First, Next, and Previous Key commands with Secondary Key, 1232
Get List with Cross-reference, 1240
Get List with Primary Key, 1238
Get List with Secondary Key, 1239
Next Greater and Next Smaller commands with Cross-reference, 1237
Next Greater and Next Smaller commands with Secondary Key, 1235
Next Greater Key and Next Smaller commands with Primary Key, 1234
verb, 1224
GET-NEXT-ELEMENT verb, 1241
GET-RECORD-TSTAMP verb, 1246
Global files, 627
Global monitor
  archiving, 443
  defining reports, 256–259
  initializing, 256, 266
  overview, 253, 255, 275
  report headings, 258–259
  sample reports, 260–263
Going from ACL to EDGE, 545
GOTO command, 545
GOTO verb, 1248
Grand total clause (EQL), 429–431
GRAND-TOTAL (EQL reserved word), 398
Graph object, description, 1675
Greenwich Mean Time, 163, 722
Grids, using, 53
Group, 216
GROUP (EQL reserved word), 398
Group box, description, 1680
Group clause (EQL), 411–413
Group extract codes (correspondence), 313
Group extract relationship, 666
Group IDs, 215
GROUP-EXTRACT verb, 1249, 1392
Grouping objects, 1647, 1651
GT (EQL reserved word), 398
Guide performance reports, 184
Guide watcher, 2034–2036
GUIDE_WATCHER environment variable, 2105
Guides
  archiving, 443
  assembling, 2013, 2040
  assembly default file, 1573
  checklist, 23
  creating, 23
  creen menus, 1586–1590
  definition, 11
  designing, 21
  EDGE, 1903–1924
  file access, 1985
  list guide flow, 2025
  maximum files per project, 21, 1987
  options, 1988
  preparing for operations, 2013
  primary file, 1985, 1987
  releasing to operators, 2037
  separate XGUIDEs, 1906
  showing a message, 1373
  swapping, 1132, 2037
  testing, 141, 2028
  testing with guide watcher, 2034–2036
  versions, 2041

H
H (EQL reserved word), 398
HEADING (EQL reserved word), 399
Heading icons (ad hoc reporting), 344
Heading, EQL clause, 413
Headings/footings
  EQL clause, 413–415
  on reports (ad hoc reporting), 355
Help
  adding commands to ACL list, 564
  creating online, 532
  deleting online, 517
  invoking, 469
  modifying online, 532
  online for guides, 643
Help menu, 59
HISTORY file
  archiving, 444
  field descriptions, 710–712
  fields copied from SYS file, 728
  record ID, 706–708
History stack, 462
History stack commands (ACL), 464
Hot key layout, 1621
Hot keys, 2000
Hot link (DDE), 1452, 1477
Hyperbolic cosine function (CALCULATE verb), 2086
Hyperbolic sine function (CALCULATE verb), 2087
Hyperbolic tangent function (CALCULATE verb), 2088
Hyperlink object, description, 1683
I
I (EQL reserved word), 399
I/O devices
  archiving, 443
Index

File option, 197, 1510
internet sockets, 1511–1514
listing, 558
listing locks, 559
message queues, 1514–1520
MQLink, 1521–1527
named pipes, 1528–1534
Old IPC Server, 1534–1539
removing tape locks, 590
setting up, 195
setting up for message queues, 1509–1539
transport mechanisms, 1498–1505

Icons
  ad hoc reporting, 985
  command button objects, 133
  ETW–W toolbar, 39
  footing (ad hoc reporting), 344
  toolbar icons., 1896

Identifiers (EQL), 392
ID-SUP (EQL reserved word), 399
IF-ELSE commands (correspondence), 326
IGNORE_OVERLAPPING environment variable, 2105
IGNORE_SPECIAL_CHARS environment variable, 2105
IMA user ID, 227

Image object
  description, 1686
  hyperlink, 1686

Importing
  add/update, 773
  archiving instructions, 443
Index

- block size, 774
- clear file, 774
- data, 599
- data format, 774
- displaying device records, 794
- error limit, 775
- field mark, 775
- import records, 776
- instructions, 772
- listing log files, 793
- log files, 776
- menu commands, 793
- overview, 769
- preparation, 770
- processing errors, 793, 812
- record size, 777
- selection criteria, 777
- SmartMerge, 819–825
- using edgeio, 524
- IN (EQL reserved word), 399
- include directory, 2124, 2126
- Incrementing counters, 73
- Initializing
  - global monitor, 256, 266
  - project monitor, 281, 546
  - system monitor, 546
- INITIALMON command, 256, 266, 268, 281, 546
- Input field
  - cross-reference or search button, 1620
  - current value position, 765
<table>
<thead>
<tr>
<th>Inserting expressions</th>
</tr>
</thead>
<tbody>
<tr>
<td>EDGE Developer setup, 1766</td>
</tr>
<tr>
<td>using SET-PROPERTY verb, 1365</td>
</tr>
<tr>
<td>INSERT-VALUE verb, 1251</td>
</tr>
<tr>
<td>Integer portion function (CALCULATE verb), 2057</td>
</tr>
<tr>
<td>INTEGER verb, 1253</td>
</tr>
<tr>
<td>INTERACTIVE environment variable, 2105</td>
</tr>
<tr>
<td>Interest paid function (CALCULATE verb), 2069</td>
</tr>
<tr>
<td>International ACTZ update, 160</td>
</tr>
<tr>
<td>Internationalization, 1116</td>
</tr>
<tr>
<td>Internet sockets</td>
</tr>
<tr>
<td>I/O devices, 1511–1514</td>
</tr>
<tr>
<td>transport mechanisms, 1499</td>
</tr>
<tr>
<td>Interpreting expressions</td>
</tr>
<tr>
<td>EDGE Developer setup, 1767</td>
</tr>
<tr>
<td>Interprocess communication, see Message queues.</td>
</tr>
<tr>
<td>INTO (EQL reserved word), 399</td>
</tr>
<tr>
<td>Intro screen, 88</td>
</tr>
<tr>
<td>Invoking multiproject assemble instruction, 562</td>
</tr>
<tr>
<td>IO_LOG_FILE environment variable, 2105</td>
</tr>
<tr>
<td>IP address, 144, 145</td>
</tr>
<tr>
<td>IRA verb, 1255</td>
</tr>
<tr>
<td>J</td>
</tr>
<tr>
<td>Justification and length codes (correspondence), 315</td>
</tr>
<tr>
<td>Justification of fields, 641</td>
</tr>
<tr>
<td>Page</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td><strong>K</strong></td>
</tr>
<tr>
<td>KEY (EQL reserved word), 399</td>
</tr>
<tr>
<td>Keyboard commands in ASCII operations, 2112–2115</td>
</tr>
<tr>
<td><strong>L</strong></td>
</tr>
<tr>
<td>L (EQL reserved word), 399</td>
</tr>
<tr>
<td>LABEL (EQL reserved word), 400</td>
</tr>
<tr>
<td>Label formatting commands (correspondence), 328</td>
</tr>
<tr>
<td>Label object, 1693</td>
</tr>
<tr>
<td>Label options clause (EQL), 415–416</td>
</tr>
<tr>
<td>Label report, 338</td>
</tr>
<tr>
<td>Label report (EQL), 389–390, 415</td>
</tr>
<tr>
<td>Label setup (ad hoc reporting), 356</td>
</tr>
<tr>
<td>Labels on command buttons in EDGE Client, 173</td>
</tr>
<tr>
<td>LANG environment variable, 2105</td>
</tr>
<tr>
<td>Lasso, 1640</td>
</tr>
<tr>
<td>Last day to schedule callbacks, 86</td>
</tr>
<tr>
<td>LE (EQL reserved word), 400</td>
</tr>
<tr>
<td>Length of fields, 642</td>
</tr>
<tr>
<td>lib directory, 2124</td>
</tr>
<tr>
<td>Libraries, 1909–1914</td>
</tr>
<tr>
<td>License information, 1832</td>
</tr>
<tr>
<td>Limits</td>
</tr>
<tr>
<td>DDE configuration, 1454</td>
</tr>
<tr>
<td>function keys, 1597</td>
</tr>
<tr>
<td>list boxes, 1698</td>
</tr>
<tr>
<td>screen objects, 1571, 1652</td>
</tr>
<tr>
<td>Line commands (correspondence)</td>
</tr>
<tr>
<td>default settings, 324</td>
</tr>
<tr>
<td>field formatting commands, 326</td>
</tr>
<tr>
<td>Command Type</td>
</tr>
<tr>
<td>------------------------------------</td>
</tr>
<tr>
<td>form feed commands</td>
</tr>
<tr>
<td>IF-ELSE commands</td>
</tr>
<tr>
<td>label formatting commands</td>
</tr>
<tr>
<td>line formatting commands</td>
</tr>
<tr>
<td>margin setup commands</td>
</tr>
<tr>
<td>page setup commands</td>
</tr>
<tr>
<td>printer escape commands</td>
</tr>
<tr>
<td>printer operation commands</td>
</tr>
<tr>
<td>read file commands with numeric arguments</td>
</tr>
<tr>
<td>Linking</td>
</tr>
<tr>
<td>views</td>
</tr>
<tr>
<td>Links (DDE)</td>
</tr>
<tr>
<td>LIST (EQL reserved word)</td>
</tr>
<tr>
<td>List box</td>
</tr>
<tr>
<td>description</td>
</tr>
<tr>
<td>multiple selects</td>
</tr>
<tr>
<td>shortcut menu</td>
</tr>
<tr>
<td>LIST command</td>
</tr>
<tr>
<td>LISTFILES command</td>
</tr>
<tr>
<td>Listing</td>
</tr>
<tr>
<td>ad hoc reports</td>
</tr>
<tr>
<td>archive instructions</td>
</tr>
<tr>
<td>guide flow</td>
</tr>
<tr>
<td>I/O device locks</td>
</tr>
<tr>
<td>import devices</td>
</tr>
<tr>
<td>operator access</td>
</tr>
<tr>
<td>record IDs</td>
</tr>
<tr>
<td>screens</td>
</tr>
<tr>
<td>selection lists</td>
</tr>
</tbody>
</table>
Index

- tasks, 973
- user IDs, 225
- LISTLISTS command, 552
- ListView control wrapper properties, 1843
- ListView control. see EDGE ListView control.
- LOAD-LIST verb, 1257
- Loan payment function (CALCULATE verb), 2070
- Localizing EDGE, 26, 150, 1260
- LOCATE-BY-ACP verb, see Locator documentation.
- LOCATE-BY-ZIP verb, see Locator documentation.
- Locked call IDs, 727
- Locked user IDs, 224
- Log files
  - exporting, 799, 812
  - importing, 776
  - listing, 793
  - Smartmerge, 824, 829
  - SuperServer, 210, 212, 2108–2109
  - task scheduler, 995
- Log gamma function (CALCULATE verb), 2074
- LOG verb, 1262
- Logarithm function (CALCULATE verb), 2075
- Logic editor, 941–948
- Logic flows
  - activation, 1615
  - analyzing, 476
  - archiving, 443
  - attaching to function keys, 1599
  - attaching to hot keys, 2003
Index

- attaching to screen events, 1612–1619
- attaching to screen objects, 1773–1776
- commands, 954
- deactivation, 1616
- development steps, 949
- during paint, 1617
- during screen display, 1617
- editing, 951
- entry to screen, 1617, 1634
- error messages, 958–959
- execution order, 1618
- exit from screen, 1598, 1617
- list logic code command, 956
- list logics command, 955
- placeholder, 2018
- routing, 1353
- screen validation, 1618
- text search, 56, 959
- using EXIT verb, 1202
- verify logic command, 957–959

Login
- disabling for port, 199
- password display, 194
- setting terminal type, 200

Login validation, 2106

Logon monitor
- defining reports, 269–271
- overview, 253, 264
- report headings, 271
- setting up, 264
LOGON_NO_VALIDATION environment variable, 2106
Look up screen, 1985
LOOKUP command, 554
Lookup table value (CALCULATE verb), 2062
LOOP verb
   End command, 1271
   For command, 1271
   Repeat command, 1276
   Until command, 1277
   verb, 1266
   While command, 1278
LOPTIONS (EQL reserved word), 400, 416
LP (EQL reserved word), 400
LS command, 556
   see also LISTFILES command.
LSREC command, 557
LSTAPE command, 558
LSTAPELOCKS command, 559, 590
LT (EQL reserved word), 400

M
M (EQL reserved word), 400
Maintenance
   checklist, 2044
      see also Database Maintenance.
Margin setup commands (correspondence), 330
Masks
   correspondence, 316
   PICTURE-FORMAT verb, 1311
      see also Appendix C.
MAX_LOGFILE_SIZE environment variable, 2106
Index

MAX_XREF_MATCH_KEYS environment variable, 2106
Maximum number of objects, 1652
MD file, 2121
Member of set, 655
MENU verb, 1279
Menus
  Edit, 55
  export, 812
File, 57
  grid, 53
Help, 59
  import, 793
  international ACTZ, 164
  list box, 44
  logic editor, 943, 954
  monitor, 255, 264, 276
  operator access, 95
  project selection, 1905
  queue definitions, 858
  queue seeding, 876
  queue utilities, 884
Screen editor, 1636
  screens tab, 1586
server area, 183
  server options, 185
task definitions, 970
task scheduler, 991
titlebar, 39
toolbars, 1894
users, 223
Index

View, 59
Window, 61
Message queues
   enabling MQLink, 1503
   expiration time, 1506
   internet sockets as transport mechanisms, 1499
   message size, 1506
   message types, 1507–1508
   MQLink as transport mechanisms, 1500–1504
   MSGQUEUE verb, 1540–1547
   named pipes as transport mechanisms, 1504
   overview, 1498
   transport mechanisms, 1498–1505
   UNIX transport mechanisms, 1499
Message types, 1507–1508
MESSAGE verb, 1289
Methods
   exposing, 1838
   using in CALL-EXTERNAL-FUNCTION, 1096
Modifying online help, 532
Modules
   quick reference, 63
Modules, transferring archived modules, 604
Monetary symbols, 155
MONEY-CONVERSION verb, 1293
Monitors
   archiving, 443
   modules, 253
MOVE command, 560
Moving records, 560
| MQLink         | enabling, 1503 |
|               | I/O devices, 1521–1527 |
|               | transport mechanisms, 1500–1504 |
| MSGQ_SKIP_SRVID_CHECK environment variable | 1552, 2107 |
| MSGQUEUE verb | 1540–1547 |
| MULTI.ASSEM command | 562 |
| Multilined objects | 1778 |
| Multiple files checking | 611 |
| Multiple project assembly and swap | |
| creating an instruction | 1919–1920 |
| invoking existing instruction | 562 |
| reports | 1922 |
| running | 1921 |
| Multiple values | |
| entering | 50 |
| selecting | 48 |
| MULTIPLY verb | 1297 |
| MULTI-RECORD-READ verb, see SQL-SELECT verb. |
| Multivalues | |
| default values | 651 |
| extracting subvalues | 1459 |
| find the position of a value | 1203 |
| get key from list or field | 1241 |
| in correspondence | 308, 332 |
| referencing | 643 |
| sorting | 1376 |
| specifying fields | 643 |
| specifying value position | 1000 |
N

N (EQL reserved word), 400

Named pipes

   I/O devices, 1528–1532
   transport mechanisms, 1504

Natural logarithm (CALCULATE verb), 2074

NE (EQL reserved word), 400

New connection setup, 146

NFOPTIONS (EQL reserved word), 400, 417

NI (EQL reserved word), 401

NO_BEEPS_PLEASE environment variable, 2107

NOFORMAT (EQL reserved word), 401

Noformat options clause (EQL), 416–418

Noformat report (EQL), 391, 416

NOPAGE (EQL reserved word), 401

NOT (EQL reserved word), 401

Notation (EQL), 382

NOTIN (EQL reserved word), 401

Nudge objects, 1656

Numbers

   adding, 1024
   adding value in a multivalued field, 1406
   creating complex expressions, 1037–1060
   deleting values in a multivalued field, 1168
   generating random numbers, 1334
   subtracting, 1404

Numeric fields

   defining, 634
   formatting options, 638
   truncating/rounding, 304
O

Object properties
  Active tab data, 1731
  ActiveX, 1243
  ActiveX controls, 1364, 1863
  Adjust cell, 1731
  Adjust child, 1732
  alphabetical listing, 1724–1730
  Alternate text, 1733
  Amplitude, 1733
  Auto start, 1734
  Background sound, 1735
  Border, 1736
  Border style, 1736
  Border thickness, 1737
  Border type, 1738
  Border width, 1738
  Bottom title, 1739
  Caption, 1740
  Caption type, 1741
  Color (graph object), 1741
  Column, 1743
  Columns, 1744
  Content field, 1744
  Content text, 1744
  Content type, 1746
  Continuous play, 1747
  Corner radius, 1747
  Current tab, 1748
Database, 1748
Defining expressions, 1749
Descending, 1750
Dimensions, 1751
Display style, 1751
Embedded field style, 1752
Enabled, 1752
Entry logic, 1753
Exit logic, 1754
Field name, 1755
Fill, 1756
Filled, 1758
Fixed width, 1758
Frame type, 1759
Frame width, 1760
Graph type, 1760
Height, 1761
Height (%), 1762
ID, 1762
Image location, 1762
Image path, 1763
Index, 1764
Input field type, 1765
Inserting expressions, 1766
Interpret expression, 1767
Justification, 1767
Left, 1768
Left (%), 1769
Left title, 1769
Legend data, 1769
Line style, 1770
List source, 1772
Local image, 1772
Logic, 1773
Logic run on, 1776
Multiline, 1778
Multiple selection, 1779
Name, 1780
Number of tabs, 1781
Numbering, 1781
Open new window, 1783
Path, 1783
Play/stop controls, 1784
Playback track bar, 1784
Point symbol, 1785
Result, 1787
Right title, 1786, 1788
Row, 1788
Rows, 1789
Save as text, 1789
Scroll bar, 1790
Server image, 1790
Shape, 1792
Sizing, 1791
Sorted, 1793
Spell check, 1793
Spinner increment, 1794
Start number, 1795
<table>
<thead>
<tr>
<th><strong>Index</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Style, 1795</td>
</tr>
<tr>
<td>Table style, 1796</td>
</tr>
<tr>
<td>Target field URL, 1796</td>
</tr>
<tr>
<td>Target screen, 1798</td>
</tr>
<tr>
<td>Target type, 1798</td>
</tr>
<tr>
<td>Target URL, 1799</td>
</tr>
<tr>
<td>Text, 1800</td>
</tr>
<tr>
<td>Text left (%), 1800</td>
</tr>
<tr>
<td>Text top (%), 1801</td>
</tr>
<tr>
<td>Timer information, 1802</td>
</tr>
<tr>
<td>Title, 1802</td>
</tr>
<tr>
<td>Title alignment, 1802</td>
</tr>
<tr>
<td>Toggle hilite, 1803</td>
</tr>
<tr>
<td>Top, 1804</td>
</tr>
<tr>
<td>Top (%), 1805</td>
</tr>
<tr>
<td>Top title, 1805</td>
</tr>
<tr>
<td>Type, 1806</td>
</tr>
<tr>
<td>Use caption width, 1807</td>
</tr>
<tr>
<td>Use columns width, 1807</td>
</tr>
<tr>
<td>Use field width, 1808</td>
</tr>
<tr>
<td>Use rows height, 1809</td>
</tr>
<tr>
<td>Use text width, 1809</td>
</tr>
<tr>
<td>Value field, 1810</td>
</tr>
<tr>
<td>Visible, 1811</td>
</tr>
<tr>
<td>Volume, 1812</td>
</tr>
<tr>
<td>Width, 1812</td>
</tr>
<tr>
<td>Width (%), 1813</td>
</tr>
<tr>
<td>Word wrap, 1814</td>
</tr>
<tr>
<td>X distance, 1814</td>
</tr>
<tr>
<td>Index</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>X label, 1815</td>
</tr>
<tr>
<td>X left, 1816</td>
</tr>
<tr>
<td>X right, 1817</td>
</tr>
<tr>
<td>Y label, 1817</td>
</tr>
<tr>
<td>Y left, 1818</td>
</tr>
<tr>
<td>Y right, 1818</td>
</tr>
<tr>
<td>Z distance, 1819</td>
</tr>
<tr>
<td>Z label, 1820</td>
</tr>
<tr>
<td>ObjectCount command, 1493</td>
</tr>
<tr>
<td>Objects</td>
</tr>
<tr>
<td>creating, 1655</td>
</tr>
<tr>
<td>current data, 733</td>
</tr>
<tr>
<td>default values, 651</td>
</tr>
<tr>
<td>filling with data from files, 1572</td>
</tr>
<tr>
<td>focus, 1656</td>
</tr>
<tr>
<td>maximum number, 1571, 1652</td>
</tr>
<tr>
<td>screen objects, 1657–1720</td>
</tr>
<tr>
<td>true/false values, 1998</td>
</tr>
<tr>
<td>using for screen display, 1571, 1652</td>
</tr>
<tr>
<td>odbc directory, 2124</td>
</tr>
<tr>
<td>OF (EQL reserved word), 401</td>
</tr>
<tr>
<td>Offsets, 70</td>
</tr>
<tr>
<td>ON (EQL reserved word), 401</td>
</tr>
<tr>
<td>OPEN-LIST verb, 1299</td>
</tr>
<tr>
<td>Operational mode</td>
</tr>
<tr>
<td>current, 746</td>
</tr>
<tr>
<td>setting, 2028</td>
</tr>
<tr>
<td>Operator access</td>
</tr>
<tr>
<td>add queue access, 1305</td>
</tr>
<tr>
<td>archiving, 444</td>
</tr>
</tbody>
</table>
### Index

<table>
<thead>
<tr>
<th>Task/Keyword</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>assigning search queues</td>
<td>93</td>
</tr>
<tr>
<td>defining</td>
<td>90</td>
</tr>
<tr>
<td>get list of assigned queues</td>
<td>1306</td>
</tr>
<tr>
<td>get property value</td>
<td>1305</td>
</tr>
<tr>
<td>listing</td>
<td>95</td>
</tr>
<tr>
<td>project selection</td>
<td>1905</td>
</tr>
<tr>
<td>remove queue access</td>
<td>1307</td>
</tr>
<tr>
<td>report</td>
<td>95, 282</td>
</tr>
<tr>
<td>set properties</td>
<td>1307</td>
</tr>
<tr>
<td>training</td>
<td>93</td>
</tr>
<tr>
<td>Operator monitor</td>
<td>279</td>
</tr>
<tr>
<td>OPERATOR-ACCESS verb</td>
<td>1302–1308</td>
</tr>
<tr>
<td>OPTIONS (EQL reserved word)</td>
<td>401</td>
</tr>
<tr>
<td>OR (EQL reserved word)</td>
<td>401</td>
</tr>
<tr>
<td>ORDER (EQL reserved word)</td>
<td>401</td>
</tr>
<tr>
<td>Order of precedence for EDGE Client environment settings</td>
<td>168</td>
</tr>
<tr>
<td>Order, execution of function keys</td>
<td>1599</td>
</tr>
<tr>
<td>Order, execution of logics</td>
<td>1618</td>
</tr>
<tr>
<td>Outbound call warning message</td>
<td>76</td>
</tr>
<tr>
<td>Output clause (EQL)</td>
<td>418–419</td>
</tr>
<tr>
<td>Output device setup (ad hoc reporting)</td>
<td>357</td>
</tr>
<tr>
<td>Output, directing to another destination</td>
<td>594</td>
</tr>
<tr>
<td><strong>P</strong></td>
<td></td>
</tr>
<tr>
<td>P (EQL reserved word)</td>
<td>344, 401</td>
</tr>
<tr>
<td>Page setup commands (correspondence)</td>
<td>332</td>
</tr>
<tr>
<td>PASSWD command</td>
<td>563</td>
</tr>
<tr>
<td>Password</td>
<td></td>
</tr>
<tr>
<td>Recording changes</td>
<td>227</td>
</tr>
<tr>
<td>Password Change Recording</td>
<td>227</td>
</tr>
<tr>
<td>Password change recording</td>
<td>227</td>
</tr>
</tbody>
</table>
Passwords
   assigning, 231
   attaching to commands, 563
   deleting, 231
   displaying at login, 194
   for ACL commands, 563
   PASSWORD verb, 1309
   recording changes, 227
Paste into
   group box, 1682
   tab object, 1711
Patch installation, 59
Patterns for data validation, 655
Pausing a process, 1375
PFILES file, 2121
PICTURE-FORMAT verb, 1311
Placeholders, 2016–2019
POINTER-FILE directory, 2122
Port command, 1494
Port number for EDGE Client, 141
Port setup, 198
Portrait Software Customer Care, 8
Positions for fields, 635
Power function (CALCULATE verb), 2058
POWER verb, 1315
Precedence order for EDGE Client environment settings, 168
Preferred calling hours
   for outbound calls, 76
   restricting seeding of queues, 77
   specifying on work schedule, 78–80
Present value function (CALCULATE verb), 2071
Preview toolbars, 1901
Primary fields, 646
Primary file, 1985, 1987
Primary key, see Record IDs.
Principal paid function (CALCULATE verb), 2070
Print screens, 1588
PRINT-CORRESPONDENCE verb, 1317
Printer escape commands (correspondence), 333
Printer operation commands (correspondence)
  form feed commands, 335
  printer escape commands, 333
Printing
  changing default for current session, 58, 203, 481
  changing server default, 200
  changing the printer assignment, 1357
  correspondence reports, 506
  from EDGE on NT, 200
  order of printers used, 203
  PRINTMON command, 565
  PRINT-VALUES verb, 1322
  project monitor report, 565
  setting up, 200
  system monitor report, 565
Printing correspondence reports, 506
Priority groups (queues), 850–854
Processing codes, 661, 2090–2092
procllook command, 569
PROJ.PARAM file, 2121
Project codes, 128
Project counter, See Counters.
Project monitor
$PM screen, 1969
$PMRG screen, 1971
archiving, 444
clear project to date monitor, 277
commands, 290
defining operator monitor, 279–281
defining reports, 283–286
initializing, 281, 546
operator monitor, 279
overview, 254, 276
printing report, 565
report headings, 286–290
result groups, 278, 282
sample reports, 291–294
view, 276

Project options
allow operator access, 99
display preferred calling hour warning message, 100
gather guide performance, 100
manual calling only, 101
short description for agents, 101

Project Selection Menu, 1905
ProjectName command, 1494
Projects
access for global files, 627
archiving, 445–446
copying, 503
creating, 58
definition, 11
deleting, 58
design, 14, 20
designing queues, 838–847
displayed in operations, 1905
files, 20
guide, 21
listing, 58
modifying, 58
operator selection, 1905
planning, 15
reports, 22
requirements, 16
unarchiving, 447–452

Properties
ActiveX controls, 1833–1837
alphabetical listing, 1724–1730
definition, 31
screen objects, 1722–1821
specifying, 47

Property assisted mode
edit tools, 943
lower list box, 944
upper list box, 944

Protected queues, 839, 850
Public key authentication, 241

Q
Qclear command, 572
Qdelete command, 574
Qrebuild command, 576
Qresched command, 579–580
Qseed command, 581
QSEED_STATUS_NOWAIT environment variable, 2107
Qtrans command, 584
QUEUE verb
  Close command, 1325
  Count command, 1325
  introduction, 1323–1325
  Open by Schedule command, 1327
  Open command, 1327
  Queue List command, 1328
  Schedule command, 1329
  Status command, 1330
Queues
  archiving seeding instructions, 444
  assigning operator access, 90, 92, 93
  assigning results, 2006
  CALL-EVENT verb, 841
  clearing call events, 572
  close queues, 1325, 1327
  count call events, 903
  default, 1992
  definitions, 850–861
  deleting call events, 572, 584, 885–889
  deleting definitions, 574, 857
  deleting protected, 857
  designing, 838–847
  detail reports, 901–903
  distribution, 868
  errors, 906–912
  FIFO, 840–841, 853, 856
get a listing of queues, 858–861
list seeding errors, 877
ordering, 836, 840–841, 853
priority groups, 850–854
protected, 839, 850
QUEUE verb, 1323–1331
randomly ordered, 840–841, 853
rebuilding, 576, 890
removing call events, 842–847, 855–856
rescheduling, 579–580
schedules, 851
seeding, 581, 862–880
seeding instructions, 863–867, 871–876
time-ordered, 840–841, 853, 871–876
transferring call events, 584, 895–898
utilities, 884–912
validating telephone numbers, 863
Quick Edit Mode, 947
Quick reference of EDGE areas and modules, 63
QUOTES verb, 1332

R
R (EQL reserved word), 402
Radio button
description, 1700
invoking logic flows, 1776
RANDOM verb, 1334
Randomly ordered queues, 840–841, 853
Range validation, 656
Rank (selection screens), 1634
Read file commands (correspondence), 336
Reading
  flat files, 1206
  records, 1335
  tape, 602
READ-RECORD verb, 1335
Rebound screen, 1607
Rebound screens, 1607
Rebuilding
  corrupted files, 543
  file dictionaries, 673
  queues, 576
Receiving data, 524, 599
Record IDs
  building, 1029
  fields, 646
  GET-KEY verb, 1225
  listing, 548, 557
Record selection (ad hoc reporting), 359
Record tracking
  description, 817–818
  enabling, 595
Record workspace, 748
Recording password changes, 227
RECORDING verb, 1340
Records
  adding, 621
  copying, 498, 512
  counting, 508
  defined, 617
DELETE-RECORD verb, 1165
  deleting, 518, 520, 692
  displaying, 521, 523
  editing, 527
  export record, 800
  get an existing call record, 1219
  import records, 776
  listing record IDs, 548, 557
  load from a record or list, 1257
  moving, 560
  retrieving multiple, 1387
  transferring, 1421
  writing, 1443

REFRESH verb, 1341

Registering ActiveX controls, 1825

Relational fields
  defining, 659–669
  with counters, 71

Relational operators (EQL), 420

Release locked user IDs, 224

RELEASE-RECORD verb, 1345

Releasing records, 1345

Remainder function (CALCULATE verb), 2058

Remaining balance function (CALCULATE verb), 2072

REMOVE verb, 1347

Removing tape lock from I/O device, 590

Report display (ad hoc reporting), 364

Report properties (ad hoc reporting), 348

Report types (EQL), 383–391

REPORTING file, 708, 713–716
Reporting, see Ad hoc reporting.

Reports
- columnar (EQL), 383
- columnar reports (ad hoc reporting), 338, 349
- count (EQL), 385–386
- designing, 22
- dump (EQL), 386–387
- form (EQL), 387–389
- label (EQL), 389–390, 415
- listing (ad hoc reporting), 372
- noformat (EQL), 391

Requirements for EDGE Developer, 27

Reserved words, 181

Reserved words (EQL), 394–404

Resize a column or row in table object, 1716

RESPONSES file, 717–718

Result command (DDE), 1494

Results
- $SETRESULT screen, 1976
- currently set, 749
- defining, 2006
- for function keys, 1599
- in STATUS file, 724
- on selection screens, 1635
- placeholder, 2018
- result groups (Project monitor), 278
- using with screens, 1575

Rewinding tape, 603

RGB codes, 1929

RMREC command, 589
RMREC command see also DELREC command.
RMTAPELOCKS command, 590
ROLL verb, 1350
Round function (CALCULATE verb), 2059
Rounding numeric fields, 304
ROUTE verb, 1353–1354
Routing
determining, 1577–1584
for function keys, 1599
on selection screens, 1635
ROUTE verb, 1353–1354
RSAMPLE (EQL reserved word), 402
Running ad hoc reports, 368
Running ETW–W, 27

S
S (EQL reserved word), 388, 402
SAMPLE (EQL reserved word), 403
Schedules, 74
Scheduling call events
last day to schedule, 86
restricting preferred calling hours, 77
restricting work schedule, 78
Schemes
applying, 1930–1932
archiving, 444
colors, 1929–1930
DEFAULT, 178, 1586, 1925
defining, 1925–1929
selecting, 178
Screen appearance
  during development, 1591
  during operations, 1568

Screen commands (operations), 1993

Screen defaults
  ASCII conversions, 188
  default measurement, 189
  default metric, 189
  font, 189
  screen height, 190
  screen width, 190

Screen development
  attaching logic flows, 1612–1619
  default file, 1573
  design considerations, 1570
  displaying function keys, 1605, 1997
  displaying titles, 1998
  function key area, 1597
  in operations, 1568
  Intro screen, 88
  object focus, 1656
  placeholders, 2018
  properties, 1722–1821
  routing, 1577
  schemes, 1586
  screen objects, 1657–1720
  selection screens options, 1625
  showing row labels, 1998
  tabbing order, 1642
  types of screens, 1570, 1593
Index

using objects, 1571
Screen editor
toolbox, 1654
Screen events, 1576, 1612
Screen menus
inside editor, 1636–1645
outside editor, 1586–1590
Screen objects
ActiveX controls, 1862–1870
audio object, 1659
check box, 1662
command buttons, 1665
display field, 1672
graph object, 1675
group box, 1680
hyperlink object, 1683
image object, 1686
input field, 1690
labels, 1693
list box, 1696
listing, 1587
names, 1774
radio button, 1700
SET-PROPERTY verb, 1359
shape, 1703
tab object, 1706
table object, 1712
video object, 1718
Screen properties
Activation, 1615
Auto arrange, 1603
Deactivation, 1616
Default file, 1604
Description, 1598
Display fkeys, 1604
Display title, 1605
During paint, 1617
Entry to screen, 1617
Exit from screen, 1617
Exit logic, 1598
Fkeys style, 1606
Inactive, 1606
Logic, 1599, 1612–1619
Metric, 1606
Name, 1607
Rebound screen, 1607
Result, 1599
Route to screen, 1599
Screen height, 1609
Screen style, 1609
Screen validation, 1618
Screen width, 1609
Title, 1610
Type, 1610
Screen tools, 1646–1651
Screen validation logic, 1618
ScreenName command, 1495
Screens
archiving, 443
Screens development
using objects, 1652
SCREEN-SAVER verb, 1355
ScreenTitle command, 1495
Search button, 1620
Searching for characters, 554
Secondary keys
defining, 644
SECURE command, 591
Security levels
assigning to ACL commands, 471, 591
displaying, 597
status level, 219, 234
Seeding
archiving instruction, 444
call events, 581
errors, 880
queues, 77, 862, 871–876
SELECT (EQL reserved word), 403
SELECT command, 593
Selecting
fields or variables, 51
multiple values, 48
values, 48
Selection clause (EQL), 419–423
Selection lists
ad hoc reporting, 338
copying, 501
deleting, 516
editing, 533
listing, 377, 552
Selection screens
  overview, 1593
  properties, 1623
  sample screens, 1595
  uses for, 1571
Sending data, 524, 599
SendKeys command, 1481–1485
SEND-TO-VOAD verb, 1356
Server
  communication setup, 204
  DDE applications, 1448
  login, 144
  patch history list, 59
  patch installation, 59
  printers see Printing.
  setup, 182–205
  trace files, 141
Server area menu, 183
Server images, 133
Server property (DDE verb), 1457
Set Callback screen, 1943
SET_DENTRY_NULL environment variable, 2107
set_tracking command, 595
SET-PRINTER verb, 1357
SET-PROPERTY verb, 1359
SET-PROPERTY verb, objects, 1722
SETSTDERR command, 594
SetTargetContent command, 1486
Setting defaults (ad hoc reporting), 341
Setting report properties (ad hoc reporting), 348
Setup menu in EDGE Client
Access command, 170
Appearance command, 171
enabling, 171
Labels command, 171
Shadow file (SmartMerge), 818
Shape object, description, 1703
Shortcut keys, 1665, 1670
Shortcut menu, 39
Shortcut menus
grid, 53
listbox, 44
SHOW-MESSAGE verb, 1373
SHOWSECURITY command, 597
Sine function (CALCULATE verb), 2087
Site setup, 191–194
SLEEP verb, 1375
smart_merge command, 599
SmartMerge
displaying records on screen, 523
exporting, 826–833
external DB override, 834–835
importing, 819–825
log files, 824, 829
record tracking, 817–818
shadow file, 818
Socket connection setup, 206, 209
SORT (EQL reserved word), 403
Sort clause (EQL), 423–425
Sorting in EQL, 423–425
SORT-VALUES verb, 1376
Soundex
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>codes, 684</td>
</tr>
<tr>
<td>conversion, 1380</td>
</tr>
<tr>
<td>SOUNDEX verb, 1380</td>
</tr>
<tr>
<td>Source clause (EQL), 406–408</td>
</tr>
<tr>
<td>SPECIAL-SOUNDEX verb, 1383</td>
</tr>
<tr>
<td>Spelling checker</td>
</tr>
<tr>
<td>automatic mode setup, 1793</td>
</tr>
<tr>
<td>manual mode setup, 1793</td>
</tr>
<tr>
<td>SET-PROPERTY verb, 1367</td>
</tr>
<tr>
<td>Spinner, 1765</td>
</tr>
<tr>
<td>Splash screen, 138</td>
</tr>
<tr>
<td>SQL-GENERAL verb, 1385</td>
</tr>
<tr>
<td>SQL-SELECT verb</td>
</tr>
<tr>
<td>close query command, 1395</td>
</tr>
<tr>
<td>get records command, 1395</td>
</tr>
<tr>
<td>introduction, 1386</td>
</tr>
<tr>
<td>moving records into fields, 1391</td>
</tr>
<tr>
<td>open query command, 1396</td>
</tr>
<tr>
<td>query name limitation, 1387</td>
</tr>
<tr>
<td>status codes, 1393–1394</td>
</tr>
<tr>
<td>using, 1387–1393</td>
</tr>
<tr>
<td>Square root (CALCULATE verb), 2059</td>
</tr>
<tr>
<td>SQUARE-ROOT verb, 1399</td>
</tr>
<tr>
<td>SR.TRANS verb, 1401</td>
</tr>
<tr>
<td>src directory, 2127</td>
</tr>
<tr>
<td>Standard deviation formula (CALCULATE verb), 2078</td>
</tr>
<tr>
<td>STAT_INTERVAL environment variable, 2108</td>
</tr>
<tr>
<td>Status codes, 1245</td>
</tr>
<tr>
<td>STATUS file</td>
</tr>
<tr>
<td>archiving, 444</td>
</tr>
</tbody>
</table>
callback date and time, 1944
field descriptions, 719–726
fields copied from SYS file, 728
fields updated by queue seeding, 862
record ID, 706–708
unarchiving, 448

Status levels
  group, 219
  user ID, 234
STATUSID file, 727
Stderr file, 141, 594, 995, 1193
STEMPLET directory
  data_d/, 2129
  data_x/, 2129
  dict_d/, 2129
  dict_x/, 2129
Storage width, 645, 1808
Storing EDGE components, see Archive.
String function (CALCULATE verb), 2063
String length (CALCULATE verb), 2082
STRING-LENGTH verb, 1403
STRLEN function (CALCULATE verb), 2082
Styles
  archiving, 444
  changing in logic, 1363
  defining, 1933
  selecting, 178
  setting in screen, 1795
Subarray function (CALCULATE verb), 2064
SUBSTR function (CALCULATE verb), 2082
Index

SUBTRACT verb, 1404
Subvalues, 1459
SUM (EQL reserved word), 403
Sum arguments (CALCULATE verb), 2080
SUM verb, 1406
SuperServer
   base port specification, 209
   definition, 206
   log file path, 210
   setup, 206
   start, 210
   status, 210
   stop, 210
   supersrvr utility, 210
SuperServer environment variables
   LOGON_NO_VALIDATION, 2106
   SUPERSRVR_LOG_DEBUG, 2108
   SUPERSRVR_LOG_TRACE, 2109
SuperServer log file, 210, 212, 2108–2109
SuperServer messages, 212
supersrvr utility, 210
SUPERSRVR_LOG_DEBUG environment variable, 2108
SUPERSRVR_LOG_TRACE environment variable, 2109
Swapping
   description, 2037
   multiple projects, 1919–1924
   technical description, 2041
   with version control, 106
Switching back from ACL to EDGE, 545
Synonym fields, 647
<table>
<thead>
<tr>
<th>Index</th>
<th>SYS fields (correspondence), 320–321</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SYS file, 728–767</td>
</tr>
<tr>
<td></td>
<td>System date, 732</td>
</tr>
<tr>
<td></td>
<td>System error messages, 750</td>
</tr>
<tr>
<td></td>
<td>System files, see EDGE files.</td>
</tr>
<tr>
<td></td>
<td>System login validation, 2106</td>
</tr>
<tr>
<td></td>
<td>System monitor</td>
</tr>
<tr>
<td></td>
<td>initializing, 546</td>
</tr>
<tr>
<td></td>
<td>printing report, 565</td>
</tr>
<tr>
<td></td>
<td>System screens, see EDGE screens.</td>
</tr>
<tr>
<td></td>
<td>System time, 732</td>
</tr>
<tr>
<td></td>
<td>System transactions, 1996</td>
</tr>
<tr>
<td></td>
<td><strong>T</strong></td>
</tr>
<tr>
<td></td>
<td>T (EQL reserved word), 403</td>
</tr>
<tr>
<td></td>
<td>Tab command, 1486</td>
</tr>
<tr>
<td></td>
<td>Tab control, 1706</td>
</tr>
<tr>
<td></td>
<td>Tab object, description, 1706</td>
</tr>
<tr>
<td></td>
<td>Tabbing order, 1642</td>
</tr>
<tr>
<td></td>
<td>Table object</td>
</tr>
<tr>
<td></td>
<td>delete column or row, 1716</td>
</tr>
<tr>
<td></td>
<td>description, 1712</td>
</tr>
<tr>
<td></td>
<td>insert column or row, 1716</td>
</tr>
<tr>
<td></td>
<td>resize column or row, 1716</td>
</tr>
<tr>
<td></td>
<td>Tables</td>
</tr>
<tr>
<td></td>
<td>defining for selection screens, 1629</td>
</tr>
<tr>
<td></td>
<td>using with selection screens, 1628, 1630</td>
</tr>
<tr>
<td></td>
<td>TabObjectCount command, 1496</td>
</tr>
<tr>
<td></td>
<td>TANDEM_TOGGLE environment variable, 2109</td>
</tr>
<tr>
<td></td>
<td>Tangent function (CALCULATE verb), 2088</td>
</tr>
<tr>
<td></td>
<td>Tape</td>
</tr>
</tbody>
</table>
Index

- forwarding, 601
- reading, 602
- removing locks, 590
- rewinding, 603
- TAPE.ERRORS log file, 776, 793, 799, 812, 824, 829
- TAPEFWD command, 601
- TAPERead command, 602
- TAPEREW command, 603
- Target clause (EQL), 406, 425–428
- Target command, 1487
- TargetCaption command, 1496
- TargetContent command, 1496
- TargetFormattedValue command, 1496
- TargetName command, 1496
- TargetPosition command, 1496
- TargetType command, 1497
- TargetTypeName command, 1497
- TargetValue command, 1497
- Task definition, 961
- Task editor, 941–948
- Task scheduler
  - assign printer, 991
  - commands, 991
  - date offset, 320, 984
  - defining a task, 980
  - deleting, 991
  - logs, 995–996
  - removing a task, 991
  - restrictions, 981
  - specifying days and times, 986
  - start scheduler, 993
status, 992
stop scheduler, 993–995
unexpected shutdown, 994–995

TASKCRON, 993, 994

Tasks
archiving, 444
assemble, 971–972
chain to next task, 963, 969, 985
commands, 970
creating, 960
DEBUG verb, 1156
default file, 963, 966, 976
development steps, 960
display tasks, 973
EXECUTE-TASK verb, 1200
executing, 534, 974
file access, 964
file limitation, 966
list assembly errors, 972
list tasks, 973
multirecord processing, 967
print task code, 973
printing code, 973
printing to host, 970
run, 974–976
running asynchronously, 1561–1565
running outside of EDGE, 1564–1565
TERMINATE-TASK verb, 1413
text search, 56, 977
using debug statements, 1156
verifying, 977
Technical support, see Portrait Software Customer Care.
TEKNEKRON-COMPLETE verb, 1408
TEKNEKRON-DIAL verb, 1410
Telephone numbers
  maximum digits, 156
  minimum digits, 156
  validating during queue seeding, 863
  validating with ACTZ file, 157
TELEPHONY verb, see Telephony documentation.
Telnet (TCP/IP) connection setup, 144, 145
TEMP file
  uses, 768
TEMPLATE directory
  .object1 directory, 2128
  .object2 directory, 2128
  bin directory, 2128
Terminal connections, 198
TERMINATE-TASK verb, 1413
terminfo directory, 2127
Terms
  Browser, 34
  drag and drop, 30
  objects, 30
  progressive disclosure, 31
  properties, 31
  property pages, 45
  tabbed page, 42
  tearoffs, 31
three-dot buttons, 51
Text color and font, 1752
Text extract codes (correspondence), 318
Text extract relationship, 668
TEXT-EXTRACT verb, 1415
THE (EQL reserved word), 403
Thousands separator, 155
Three-dot button, 51

Time
  country codes formats, 152
  current (SYS file), 732
  current time at customer site, 751
  current time zone, 750
daylight savings, 154
formatting options, 638
internal format, 634
range validation, 656
range validation rules, 1996
separator format, 154
TIME FORMAT verb, 1417
time zones, 191, 722

TIME-FORMAT verb, 1417

Time-ordered queues, 840–841, 853, 856, 871–876

Title/prompts setup (ad hoc reporting), 365
Titlebar menu, 39
tmp directory, 2125
  cleaning, 489
copy.proj log, 504
TO (EQL reserved word), 388, 403

Toolbars
adding, 1884
    docking, 1888
EDGE Developer, 39
    editor, 1884
icon separators, 1897
icons, 1896–1900
    menu, 1894
    modifying, 1884
preview, 1901
properties, 1887–1893
Toolbox
    activeX, 1636, 1862
    screen editor, 1645, 1652
TOTAL (EQL reserved word), 403
    Total clause (EQL), 428
TPHNY_CONSULT_PATH environment variable, 2109
TPHNY_LOG_PATH environment variable, 2110
Tracking records, enabling, 595
Training, (setting operator access), 93
Transactions, MQLink, 1501, 1525, 1548, 1555, 1560
Transactions, system, 1996
TRANSFER-RECORD verb, 1421
Transferring
    archived modules, 604
    call events, 584
TreeView control wrapper
    events, 1858
    properties, 1853
TreeView control, see EDGE TreeView control.
TRIM verb, 1423
<table>
<thead>
<tr>
<th>Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Truncating numbers, 1253</td>
</tr>
<tr>
<td>Truncating numeric fields, 304</td>
</tr>
<tr>
<td><strong>U</strong></td>
</tr>
<tr>
<td>U (EQL reserved word), 403</td>
</tr>
<tr>
<td>UAP verb, 1425</td>
</tr>
<tr>
<td>Unarchive</td>
</tr>
<tr>
<td>begin unarchive, 447</td>
</tr>
<tr>
<td>options (add, replace, overwrite), 439</td>
</tr>
<tr>
<td>transferring archived modules, 604</td>
</tr>
<tr>
<td>UNARCHIVE command, 604</td>
</tr>
<tr>
<td>Ungrouping objects, 1651</td>
</tr>
<tr>
<td>Universal Time Zone Queue</td>
</tr>
<tr>
<td>assigning work schedules, 84</td>
</tr>
<tr>
<td>Calendar, 81, 83</td>
</tr>
<tr>
<td>EDGE interfaces, 85</td>
</tr>
<tr>
<td>Group setup, 215–219</td>
</tr>
<tr>
<td>Qresched command, 579–580</td>
</tr>
<tr>
<td>Qseed command, 581–583</td>
</tr>
<tr>
<td>queue seeding, 862</td>
</tr>
<tr>
<td>rescheduling queues, 579–580, 885, 893–895</td>
</tr>
<tr>
<td>Schedules, 74–75</td>
</tr>
<tr>
<td>time-ordered queue properties, 871</td>
</tr>
<tr>
<td>User setup, 228–232</td>
</tr>
<tr>
<td>UNIX</td>
</tr>
<tr>
<td>bypass EDGE user login, 147</td>
</tr>
<tr>
<td>processes, 569</td>
</tr>
<tr>
<td>Upgrading ActiveX controls, 1825</td>
</tr>
<tr>
<td>UPPER environment variable, 2110</td>
</tr>
<tr>
<td>USE_KEYPAD environment variable, 2110</td>
</tr>
</tbody>
</table>
Index

User access setup, 232
User IDs
  access to ETW-W, 236
  archiving, 444
  clearing, 492
  clearing locked, 491, 492
  defining group IDs, 215
  defining user access, 232
  defining user IDs, 226
  listing, 225
  Password change recording, 227
User IDs, locked, 224
User login, bypass EDGE login, 147
User monitor
  access requirements, 297
  commands, 299
  list of users, 298
  modes, 295–296
  overview, 254, 295
  restrictions, 297
  start monitoring, 297
User screens, 1566
User-defined error messages, 734
User-defined relationship, 661
UserID command (DDE server), 1497
UserName command (DDE server), 1497
Users, 223
USING (EQL reserved word), 404
Index

V
V (EQL reserved word), 404
V_CALC_STACK_SIZE variable, 2110
Validate system login, 2106
Values
displaying with DISPLAY-VALUES verb, 1176
inserting a value, 1251
moving values in a multivalued field, 1350
printing, 1322
selecting, 48
Variables
@variables, 1001
#variables, 1001
%variables, 1001
call variables, 1001
global variables, 1001
local variables, 1001
specifying value position, 1000
using with verbs, 1000
with CALCULATE verb, 1051
VER.CONT file, 126
Verbs
alphabetical list, 1003–1016
by function, 1017–1021
property syntax rules, 999
restrictions in task scheduler, 981
specifying literals, 1000
specifying multivalued fields, 999
specifying value position, 1000
specifying variables, 1000
summary, 1003–1016
using numbers, 1000
vercont command, 607
VERCONT_PATH environment variable, 2111
VERCONT_PGM environment variable, 2111
Verify screen, 1589
Version control
description, 13
disabling, 104
from ACL, 607
from EDGE Developer, 102–126
vfiles command, 611
Video object, description, 1718
View callbacks screen, 1985
View menu, 59
Views, 1443–1447
VOAD auto-dialer, 192
VOAD-DIAL verb, 191, 199, 1426
VOAD-DIAL-ATT verb, 1427
VOAD-HANGUP verb, 1428

W
Warm link (DDE), 1452, 1477
Web tab, 1619
WHERE (EQL reserved word), 404
Wildcards, 461
Window helper, 1577
Window menu, 61
Window menu in EDGE Client
  New Window command, 171
window properties
- always on top attribute
  - Window Helper, 1581
  - WINDOW verb, 1432
- background color, disabled object, 1368
- background color, enabled object, 1368
- border color, object frame, 1369
- font, 1369
- foreground color, disabled object text, 1368
- foreground color, enabled object text, 1368
- SET-PROPERTY verb, 1359–1372
- window background color
  - Window Helper, 1582
  - WINDOW verb, 1434–1435
- window frame style
  - Window Helper, 1583
  - WINDOW verb, 1435–1438
  - Window Helper, 1577–1584
  - WINDOW verb, 1429–1442
- WINDOW verb, 1429
Work schedule, 74, 78
Wrappers
- creating, 1860
- description, 1841
- EDGE ListView control, 1842–1852
- EDGE TreeView control, 1853–1859
- WRITE-RECORD verb, 1443
Writing flat files, 1206
Index

X
X_filename, 2121
XGUIDE
  creating, 2015
  separate, 1906
  storage location, 2126
  template, 2128
XQUERY command, 613
xrefall command, 615

Y
Y (EQL reserved word), 404
Year 2000, 154, 193