The ECL IDE and HPCC Systems® Client Tools

Boca Raton Documentation Team

The ECL IDE and HPCC Systems® Client Tools

Boca Raton Documentation Team
Copyright © 2026 HPCC Systems®. All rights reserved

We welcome your comments and feedback about this document via email to <docfeedback@hpccsystems.com>

Please include **Documentation Feedback** in the subject line and reference the document name, page numbers, and current Version Number in the text of the message.

LexisNexis and the Knowledge Burst logo are registered trademarks of Reed Elsevier Properties Inc., used under license.

HPCC Systems® is a registered trademark of LexisNexis Risk Data Management Inc.

Other products, logos, and services may be trademarks or registered trademarks of their respective companies.

All names and example data used in this manual are fictitious. Any similarity to actual persons, living or dead, is purely coincidental.

2026 Version 10.4.8-1

Overview 4
Documentation Conventions 5
The ECL IDE 8
Introduction 8
ECL Command Line Interface 53
The ECL Command Syntax 53
ECL Compiler 118
Using the ECL Compiler as a Stand Alone option 119
Compiled Options: 123
Examples 124
Command Line DFU 126
Command Line Interface 126
ESDL Command Line Interface 139
The ESDL Command Syntax 139
JSON and XML to ECL Tools 159
json2ecl 159
xml2ecl 163

© 2026 HPCC Systems®. All rights reserved

Overview

This manual contains documentation for the set of Client Tools for use with the HPCC Systems platform. These tools include:

ECL IDE	An integrated development environment for ECL programmers to create, edit, and execute ECL code.
ECL	Command line ECL tool.
ECL Compiler	Command line ECL Compiler
DFUPlus	Command line Distributed File Utility management tool, facilitate automation of data file spray, despray, and other common file handling tasks.
ESDL	Command line ESDL management tool.

© 2026 HPCC Systems®. All rights reserved

Documentation Conventions

ECL Language

Although ECL is not case-sensitive, ECL reserved keywords and built-in functions in this document are always shown in ALL CAPS to make them stand out for easy identification.

Example Code

All example code in this document appears in the following font:

```
MyECLFileName := COUNT(Person);
// MyECLFileName is a user-defined ECL file
// COUNT is a built-in ECL function
// Person is the name of a dataset
```

ECL file names and record set names are always shown in example code as mixed-case. Run-on words may be used to explicitly identify purpose in examples.

Actions

In step-by-step sections, there will be explicit actions to perform. These are all shown with a bullet or a numbered step to differentiate action steps from explanatory text, as shown here:

- Keyboard and mouse actions are shown in small caps, such as: DOUBLE-CLICK, or press the ENTER key. word.
- On-screen items to select are shown in boldface, such as: press the **OK** button.

© 2026 HPCC Systems®. All rights reserved

Installation

The installation program installs all client tools, including the ECL IDE, DFUPlus, and the ECL Command line tool.

1. From the HPCC Systems® download page, https://hpccsystems.com/download/

   Download the appropriate Client Tools for your Operating System. (available for CentOS, Ubuntu, Mac OSX, or Windows)

   Note: The ECL IDE only runs on Windows operating systems.

2. Install the client tools software to your machine.

Windows:

Run the executable file, for example: hpccsystems-clienttools_community-7.X.X-XWindows-i386.exe on your machine. Follow the prompts to complete the installation.

RPM-Based Systems (CentOS/RedHat):

An RPM installation package is provided.

```
sudo yum install <rpm file name>
```

Debian-Based Systems (Ubuntu):

For Ubuntu installations a Debian package is provided. To install the package, use:

```
sudo dpkg -i <deb filename>
```

After installing the package, run the following to "fix" any missing dependencies:

```
sudo apt-get install -f
```

Mac OSX:

Run the installation file, for example: hpccsystems-clienttools_community-7.X.X-XDarwin-x86_64.dmg. Follow the prompts to complete the installation.

© 2026 HPCC Systems®. All rights reserved

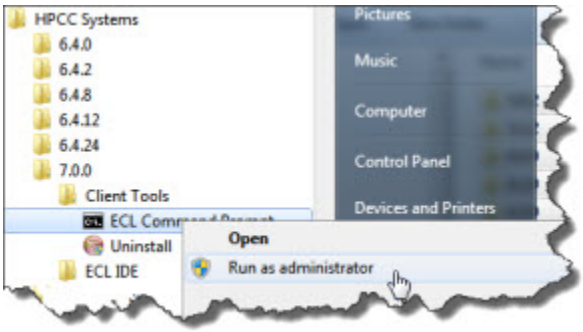
Client Tools and the Windows Path

The HPCC Systems Client tools are designed to allow you to have more than one version installed. For that reason, the choice to add the folder of executable files to your system path is optional during installation.

If you plan to use only one version, it may be easier to add the folder to your system or user path. If you want multiple versions installed, you probably want to manage the path yourself.

To make this easier, the installer creates a Start menu shortcut for an ECL Command Prompt. This command prompt adds the executable folder to the path for the session of that command window.

Some command line tool functions may require elevated permissions. You can **right-click** > **Run As Administrator** for the ECL Command Prompt for this purpose.



© 2026 HPCC Systems®. All rights reserved

The ECL IDE

Introduction

ECL IDE is the simple and easy way to create Queries into your data, and ECL files with which to build your queries. You build your Queries and files (see the *ECL Language Reference* and the *ECL Programmer's Guide)* using Enterprise Control Language (ECL). ECL has been designed specifically for use in huge projects where many things can be similar.

ECL's extreme scalability comes from a design that allows you to leverage every query you create for re-use in subsequent queries as needed. To do this, ECL takes a *dictionary* approach to building queries wherein each ECL definition defines an ECL file. Each previously defined ECL file can then be used in succeeding ECL File definitions--*the language extends itself as you use it*.

ECL IDE is an ECL programmer's tool. Its main use is to create the ECL files and is designed to make ECL coding as easy as possible.

ECL files

These are the basic building blocks from which you create queries into your data. ECL files are extensively defined in the *ECL Language Reference.*

You create an ECL file by coding an expression that defines how a calculation or record set derivation is to be done. Once an ECL file is defined you may use it in succeeding ECL file definitions, making each succeeding ECL file more and more highly leveraged upon the work you have done before. This results in extremely efficient query construction.

Built-in Functions

ECL IDE also has all the ECL built-in functions available to you for simple point-and-click use in your query construction. For example, the Standard String Library (Std.Str) contains common functions to operate on STRING fields such as the **ToUpperCase** function which converts characters in a string to uppercase.

These are installed as plugins and are in the ecllib folder shown in the Repository window.

How are all these used to build Queries?

ECL IDE allows you to mix-and-match your data with any of the ECL built-in functions and/or ECL files you have defined to create Queries. Because ECL files build upon each other, the resulting queries can be as complex as needed to obtain the result.

Once the Query is built, you send it to a High Performance Computing Cluster (HPCC) which processes the query and returns the result--extremely quickly.

Complex Queries that may have taken weeks to format, program, and execute using old-style mainframe data-mining tools can literally execute and return the result in seconds.

Configuration Files

ECL IDE uses configuration files (.CFG) to store the information for environments to which it will connect. Create multiple configuration files to connect to different environments. Use the **Save As** option on the **Preferences** window to create a new configuration file.

© 2026 HPCC Systems®. All rights reserved

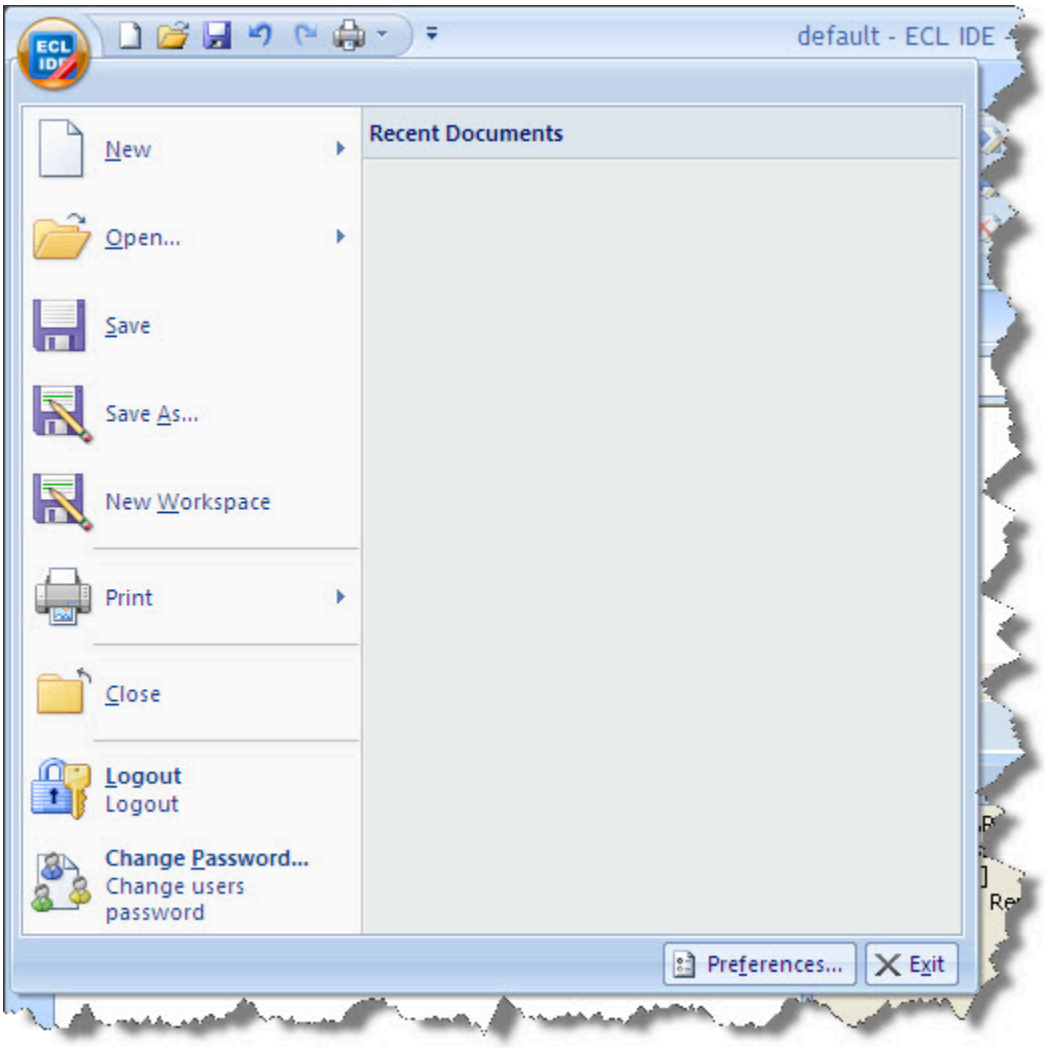
To create a configuration, you need to know the following:

- The IP address of the ESP Server.
- If services are not using the default ports, you will also need to know the port bindings for the following ESP Web services:
  - WsTopology (default port is 8010)
  - WsWorkunits (default port is 8010)
  - WsAttributes (default port is 8145) **Note:** Not part of Community Edition. For legacy use only.
  - WsAccount (default port is 8010)
  - WsSMC (default port is 8010)
  - WsFileSpray(default port is 8010)
  - WsDfu(default port is 8010)

You can find the ESP Server's IP address and the bindings for service ports using the System Servers section in ECL Watch.

To create or edit configuration files:

© 2026 HPCC Systems®. All rights reserved



Press the **Preferences** button on the Orb button drop menu.

or

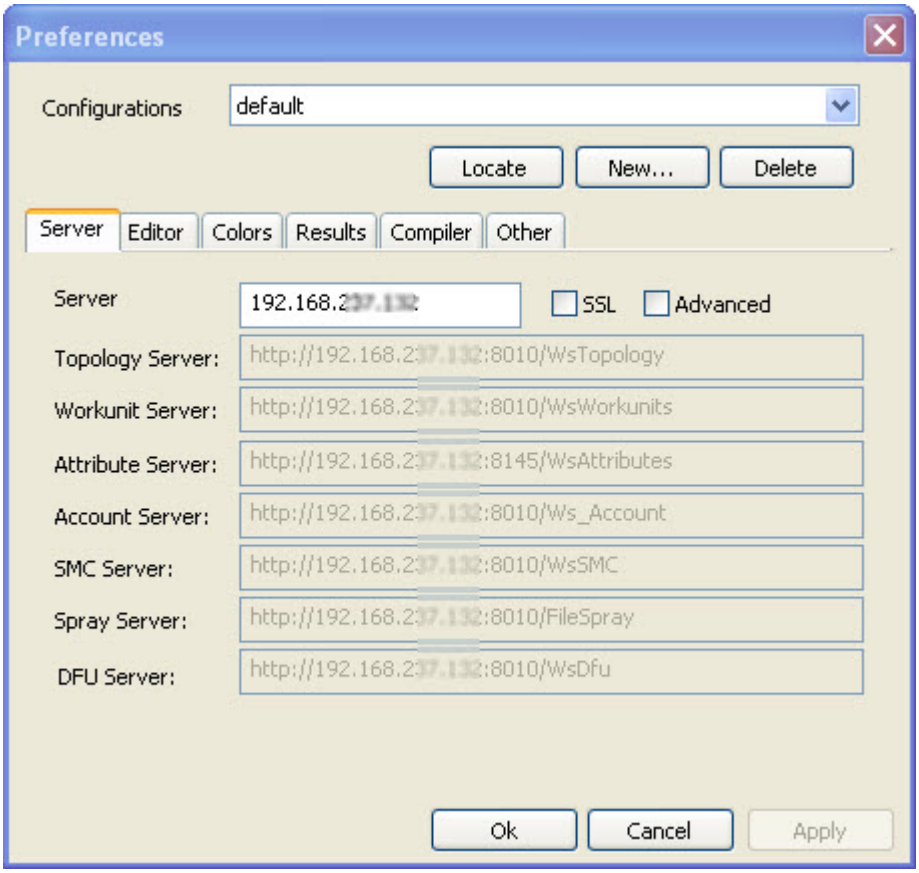
- Press the **Preferences** button on the Login window.

  Note: Some options, such as the Server settings, can only be changed by accessing the preferences through the Login window.

The Preferences window displays:

© 2026 HPCC Systems®. All rights reserved

- Select the **Server** tab and enter the IP address of the ESP in the Server field, but do not include the port details:



© 2026 HPCC Systems®. All rights reserved

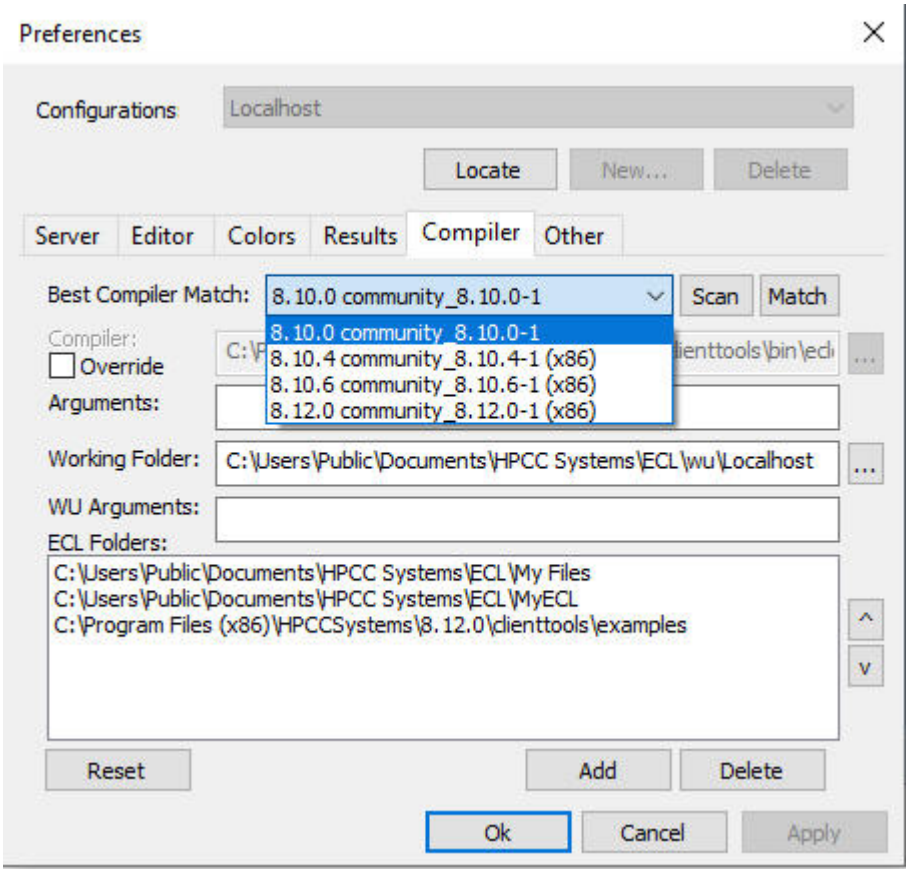
- Select the **Compiler** tab.

  The compiler details are automatically specified. If you want to override the defaults, check the **Override Automatic Compiler Selection** checkbox.

  Note: You can have multiple versions of the compiler and client tools installed. This allows you to have multiple configurations where the compiler matches each server you need to access. Check the **Override Automatic Compiler Selection** checkbox and specify the Complier you want to use.

  For **Compiler**, it is usually best to let the system select the best match for you. You can see the available compilers in the **Best Compiler Match** drop list.

  Use the **Scan** button to update the list of available compilers. Use the **Match** button to find the best match.



- Specify your Working Folder.

  When you submit to a Local target, your queries are compiled and stored in this location.

  To compile a query locally rather than on a cluster in your environment, select **Local** as the **Target** before pressing **Submit** on the **Builder** window.

  If you are running under Windows and want to compile locally, install the Microsoft VS 2008 C++ compiler (either Express or Professional edition). Linux users need GCC.

- Specify your ECL Folders.

© 2026 HPCC Systems®. All rights reserved

When creating queries in ECL IDE, you must save the ECL file before you can run the query. Specify the locations on your computer where you want the ECL files to be saved. All folders specified here are listed in the **Repository** window in ECL IDE where you can get easy access to all your files.

Sample ECL files which are supplied with ECL IDE are automatically installed onto your computer when you download ECL IDE. They are downloaded to the My Documents folder on your computer and the location of this Working Folder is automatically added into the configuration.

- If you are creating a new configuration, click **New...** and enter a name without spaces.
- If you are editing an existing configuration, click **Apply** and then **OK** to save your changes.

When you have completed your configuration changes, you are prompted to relogin.

Using the Preferences Window

Server Tab

The **Server tab** has two modes for setting the server information: Normal and Advanced. Use the normal mode when all ESP web services are deployed using default settings.

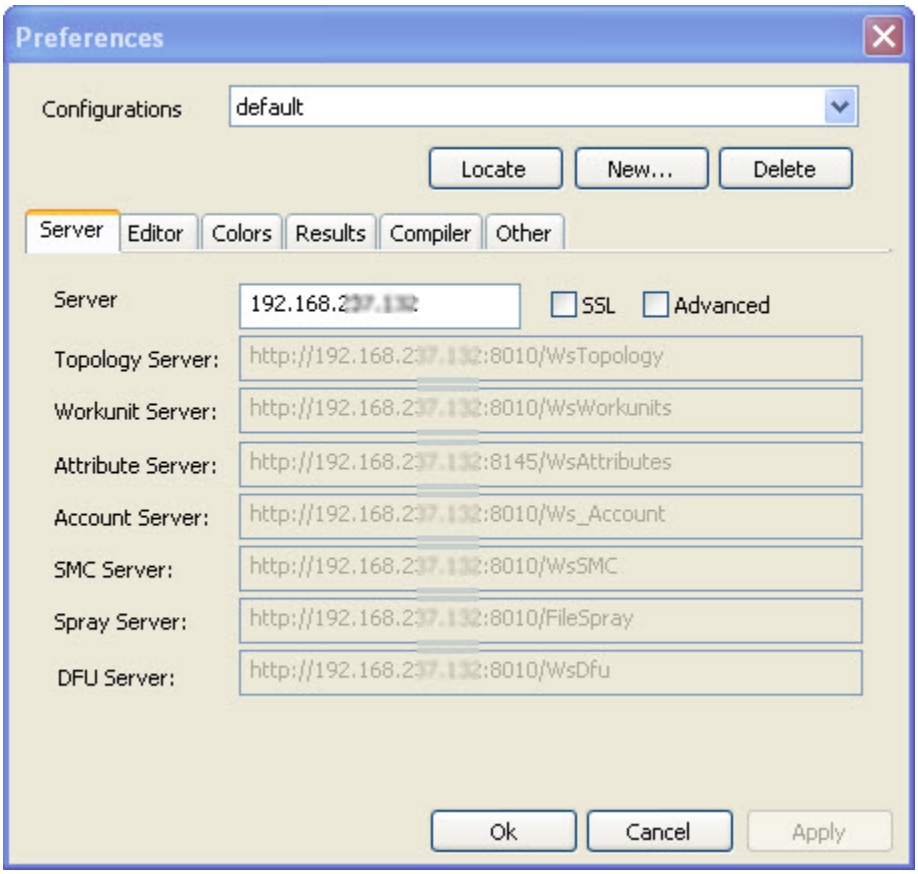
In normal mode, you need only supply the ESP server's IP address or DNS machine name. The remaining values are automatically completed.

If you are creating a connection using Secure Socket Layer (SSL) to connect, check the **SSL** box. See the *Using SSL* section in this manual for more information.

Use Advanced Mode to edit individual values for the ESP Server connections.

Check the **Advanced** box to use Advanced Mode.

© 2026 HPCC Systems®. All rights reserved



The Server settings, can only be changed by accessing the preferences through the Login window.

Press **Apply** and then **OK** to save the current information to a new configuration file. All available configuration files are shown on the **Configurations** drop list when you log in and you simply select the one you want ECL IDE to use to connect to an HPCC Systems environment.

	Press the **Locate** button to find the folder containing a configuration file which you have previously created and saved onto your computer.

Editor tab

The **Editor tab** allows you to configure the editor to your preference.

Tab Width (chars)	Specify the number of character spaces to use for a tab. The default is 2.
Use Spaces	Specify to use spaces for a tab.
Auto Save (secs)	Specify automatic save interval in seconds. The default is 10.
Maintain Indent	Check this box to maintain the level of indention when creating a new line in the editor.
Line Numbers	Check this box to display line numbers in the editor.
Tree	Check this box to display code structures in a collapsible tree at the side of the editor.

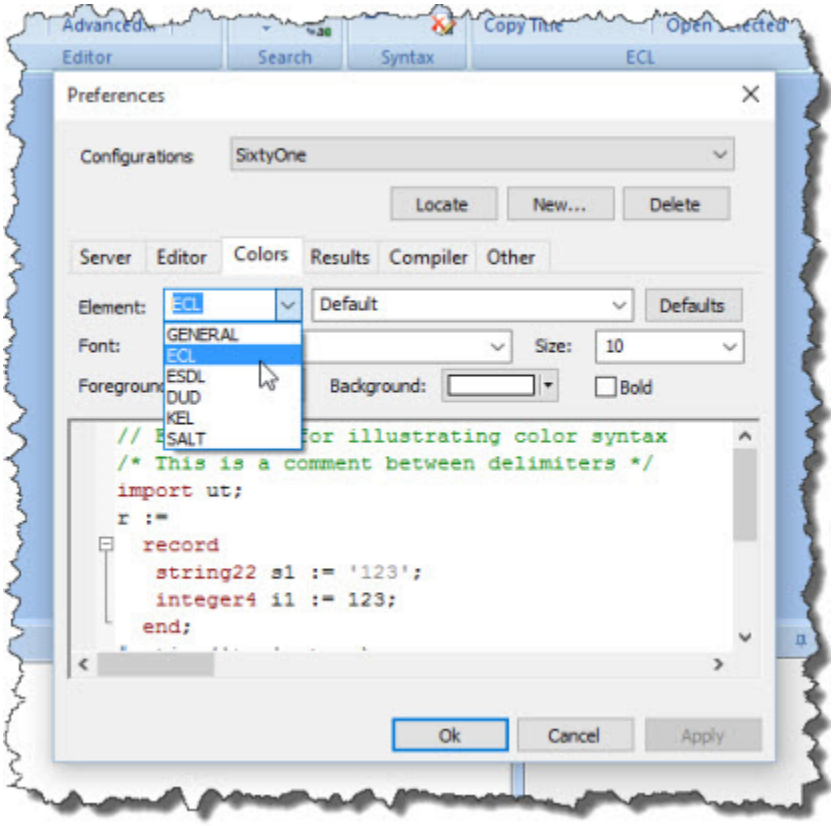
© 2026 HPCC Systems®. All rights reserved

Open MDI children Maximized	Check this box if you prefer child windows to open maximized within the application frame.
Show Whitespace Characters	Check this box to display whitespace characters in the editor.
Target Background Colors	Check this box to change the editor background color depending on the selected Target. The colors are defined in the **Colors** tab.
Keep Repository Synchronised	Check this box if you want to ensure that when an ECL file window is selected, the ECL file is auto selected in the repository.
Auto Complete on Period	Check this box to show a list of available ECL files whenever an existing folder name is typed.
Double Click Selects Qualified Label	Check this box to ensure that on a double click, the entire folder and ECL file name is selected. When unchecked, only the folder name is selected on a double click.

© 2026 HPCC Systems®. All rights reserved

Colors tab

The **Colors tab** allows you to configure the text, foreground and background colors to your preference. There are several predefined color schemes or you can also choose to customize the color scheme to your preference.



Element	There are two Element fields. The one on the left allows you to choose one of the supported file types, such as ECL, ESDL, SALT, KEL, etc. To the right, you can choose the element to customize in the selected file type, such as comment, string, operator, etc.
Default	Press this button to reset all values to the default color settings.
Font	Specify the font to use in the Results window. The default is Arial.
Size	Specify the font size to use in the Results window. The default is 10.
Foreground	Select a color for the foreground from the drop list provided.
Background	Select a color for the background from the drop list provided.
Bold	Check this box to embolden the color of the text.

Results tab

The **Results tab** allows you to configure how results display.

Result Font	Specify the fonts to use in the Results window. The default is Arial.
Size	Specify the font size to use in the Results window. The default is 10.

© 2026 HPCC Systems®. All rights reserved

Limit Result (rows)	Specify the default result limit you want to use. The system default is 100. You can override this setting by pressing the More button on the builder window and changing the value in the Limit field.

Compiler tab

The **Compliler tab** allows you to configure the local compiler (eclcc).

If you are running under Windows and want to compile locally, you need to install the Microsoft VS 2008 C++ compiler. (either Express or Professional edition) **. Linux users** need GCC.

An **Override Automatic Compiler Selection** checkbox is available should you want to override the default compiler selection, or location. Use this option if you are using multiple versions of the client tools or if you installed the compiler in a different location.

Compiler	Specify the compiler's full path and executable name.
Arguments	Optional compiler arguments. See the Command Line ECL section for more info.
Working Folder	Specify the full path to the folder you want to use when compiling locally.
WU Arguments	Optional command line parameters to pass to a locally compiling workunit.
ECL Folder	Specify the full path to the folder you will use to store your ECL files. Some folders are automatically predefined when ECL IDE is installed. They are used to store ECL sample and example code.
	The Arrows next to the ECL Folder allow you to move folders up and down in appearance in the workspace
Add	Press the Add button to browse and select folders to add as ECL folders
Delete	Press the Delete button to remove selected ECL folders
Reset	Resets the values of all fields to the default settings.

Other tab

The **Other tab** allows you to configure other settings.

Active Workunits Refresh Interval (secs)	Specify the refresh interval (in seconds). This determines how often the Active Workunit Toolbox refreshes. The default is 10.
Test GPF	For internal use only.
Disable Invoke	Check this box to disable Invoke.
Ignore Whitespace on Compare	Check this box to ignore whitespace (ii.e. spaces and blank lines) when comparing ECL files.
Show CR/LF on Compare	Check this box to show the type of line endings in the compare window when comparing two ECL files (for example, on checkin or history).
Ignore Server Version	For internal use only.
WU Fetch Limit	The maximum number of WUs that can be fetched, for example, when looking for workunits sent during a specific year/month etc.
WU Persist Limit	The maximum number of days WUs are allowed to persist.

© 2026 HPCC Systems®. All rights reserved

Disable Update Checking	Check this box to disable checking for available updates.
Disable Meta Processing	Check this box to disable metadata processing.

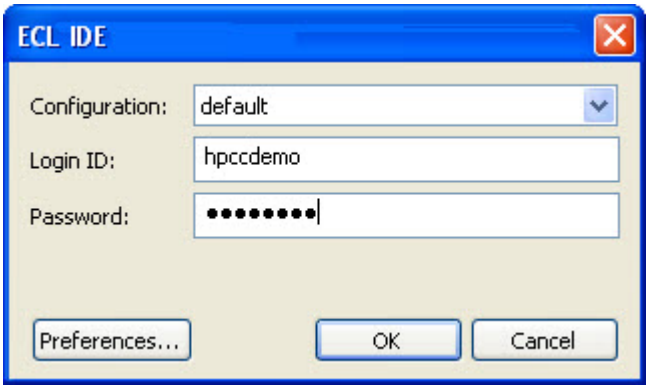
© 2026 HPCC Systems®. All rights reserved

Login

When you start the ECL IDE, the Login window displays.

- If you have created a configuration file to connect to the desired HPCC Systems environment, select it from the **Configurations** drop list. If not, press the **Preferences**button to create one.
- Enter your credentials (Login ID and Password) and press the **OK**button.

  The Login window displays errors (e.g., invalid login information) if it cannot connect properly as shown below.



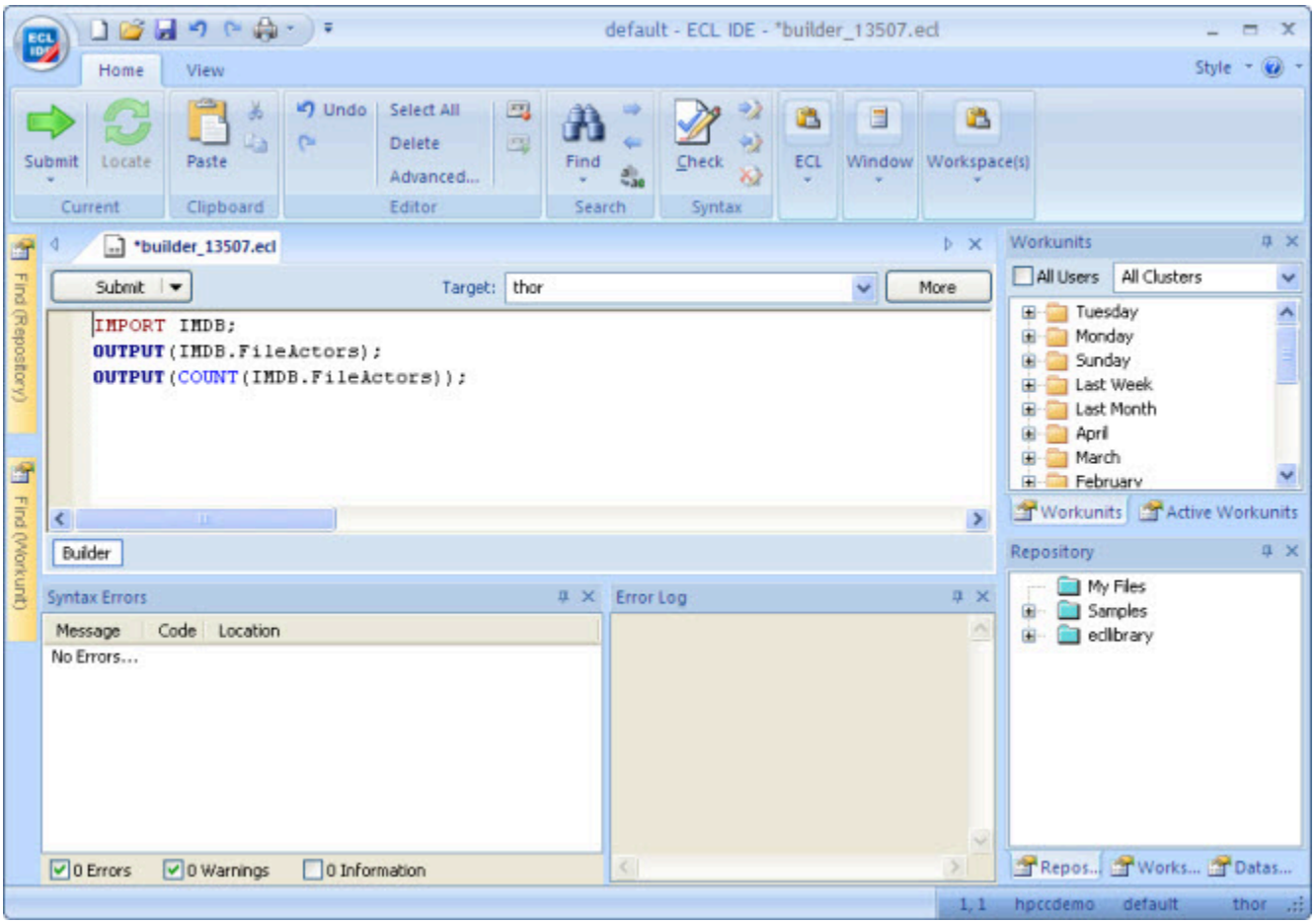
	The title bar of the login window also displays the version of the ECL IDE.

© 2026 HPCC Systems®. All rights reserved

The ECL IDE Workspace

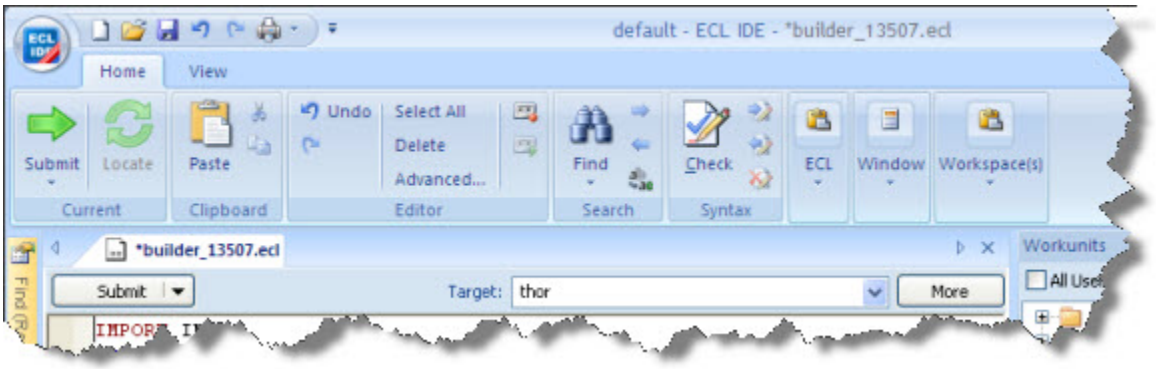
The ECL IDE workspace is customizable. You can choose which toolboxes to display and the location to dock them. You can also move toolbars to the position you choose. In addition, you can open child windows in normal mode or maximized or minimize the Ribbon.

In this manual, you will see screen representations using the locations of toolboxes pinned right.



Using the Ribbon

The Ribbon makes the commands that you need to use to complete a task easy to find.



© 2026 HPCC Systems®. All rights reserved

Commands are organized in similar groups contained in a tab. Each tab contains commands that relates to a type of activity, such as Syntax Checking or Search.

The Ribbon reduces clutter by showing some tabs only when needed. For example, the Browser Navigation tab is shown only when ECL Watch is active.

	After you have learned the locations of commonly used commands, you can gain more available space by minimizing the Ribbon.

To minimize the Ribbon:

- Press the button on the right side of the **Quick Access Toolbar.**

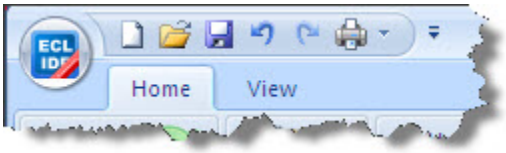
The menu displays,

- Select **Minimize the Ribbon**.

To use the Ribbon while minimized, CLICK on a tab, then CLICK the option or command you want.

The Quick Access Toolbar

Quick Access toolbar provides shortcut buttons for commonly used tasks, such as New Builder Window, Save, or Print. You can customize this button to add other tasks or remove buttons you don't want on the toolbar. You can also choose to display it beneath the Ribbon.



To customize the Quick Access Toolbar:

- Press the button on the right side of the **Quick Access Toolbar.**
- Use menu commands to customize.

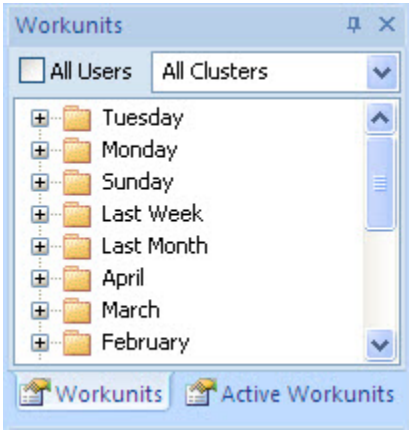
© 2026 HPCC Systems®. All rights reserved

Working with Toolboxes

Toolboxes can be docked, pinned, unpinned, grouped, or hidden. In the graphic shown above, all toolboxes are shown pinned to the right (a pre-configured options on the View ribbon tab).

Grouping, Docking, and AutoHide

Once grouped, the toolboxes open and close as a group until ungrouped. AutoHide mode (unpinned state) also acts upon the group.



In the upper right corner, two toolboxes (Workunits and Active Workunits) are shown grouped. Navigate between grouped toolboxes by clicking on the tab for the toolbox you want to display.

To group toolboxes, drag a toolbox and drop it on another toolbox. To ungroup, drag a toolboxes tab and drop it in another area. To move a group, drag it by the title bar.

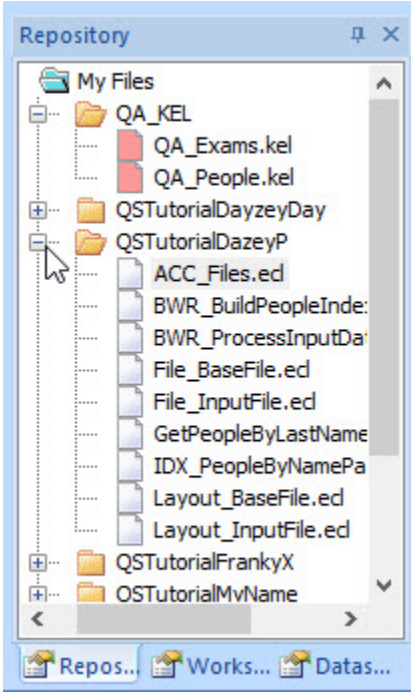
To autohide (unpin) a toolbox or toolbox group, press the pushpin at the top of the toolbox. In autohide mode, the toolbox is hidden until you place the mouse cursor over its tab.

To reset toolbox locations, select the desired option from the **Reset Docked Windows** sub-menu on the View menu.

© 2026 HPCC Systems®. All rights reserved

Repository Toolbox

This toolbox displays the Files and Folders for the current configuration. The files are organized by folders displayed in a tree format. Expand any folder branch to see the ECL files within it. The IDE supports other languages in addition to ECL, namely ESDL, KEL, SALT, and others. To help differentiate the files are color coded according to the type of file they are.



In addition to folders, plugin service libraries are also displayed at the bottom of the repository. These special folders are pre-compiled libraries of functions that you can use in your ECL code. For example, the Standard Library (stringlib) contains common functions to operate on STRING fields such as the **ToUpperCase** function which converts characters in a string to uppercase.

Plugin service libraries are added to a system in one of three ways. Several are provided as standard. Others are available as optional addons. And, you can create your own and add them to a system.

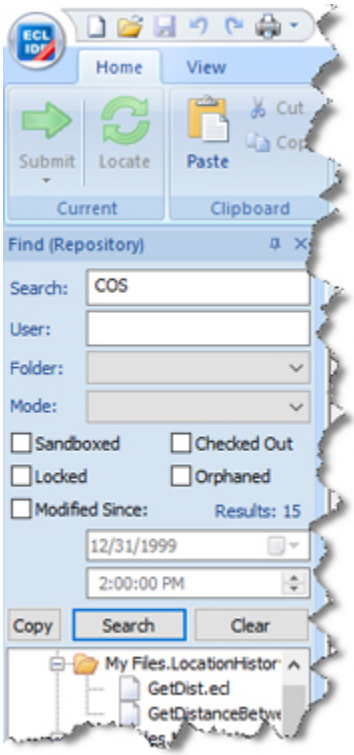
A right-click on the folders in the Repository Toolbox gives you an option of **Locate File in Explorer** which will then open that folder in a new explorer window.

You can rearrange the order of the folders in the Repository Toolbox by modifying the order they appear in the compiler tab of the preferences window.

© 2026 HPCC Systems®. All rights reserved

Find (Repository) Toolbox

This toolbox allows you to search the active Repository. Results are organized by folder and displayed in a tree format. Expand any folder to see the ECL files within it. The number of results is displayed in blue.



Search	Type your search phrase or regular expression (depending on the mode selected).
User	Limits search to a single user.
Folder	Limits search to a single folder.
Mode	Select the search mode from the drop list. There are several options. **Wildcard match:** allows a standard search and supports the use of ? to represent any single character or * to represent any number of characters. **Regular Expression:** allows a regular expression search.
Sandboxed	Check this box to only locate ECL files which are in your sandbox.
Checked out	Check this box to only find ECL files that have been checked out.
Locked	Check this box to only find ECL files that have been locked.
Orphaned	Check this box to only find ECL files that have been orphaned.
Modified since	If checked, this option enables the date and time fields. Enter the time and/or date limit (or press the lookup button to select the date from a calendar). This filters ECL files based on the date and time supplied.
Copy	Press this button after the search to copy the list of found ECL files.
Search	Press this button to initiate the search. When completed, search results are displayed at the bottom of the toolbox.

© 2026 HPCC Systems®. All rights reserved

Clear	Press this button to clear the search results.

WorkunitsToolbox

This toolbox displays Workunits for the environment to which you are connected. Workunits are organized into "folders" by date and displayed in a tree format. Expand any branch to see the Workunits within it.

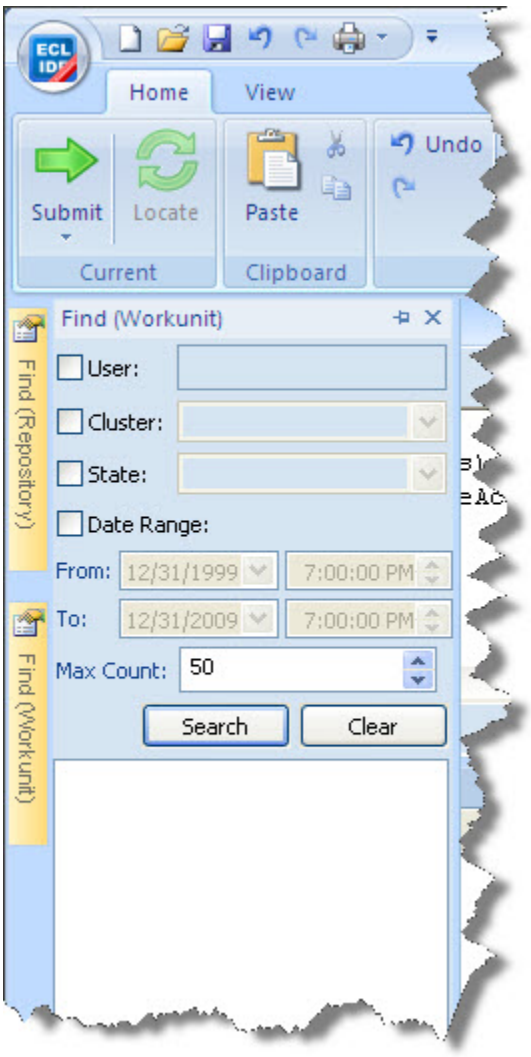
Active Workunits

This toolbox displays active Workunits for the environment to which you are connected. Active Workunits are those which are in one of the following states: submitted, compiled, running, blocked, or aborting. The display is automatically refreshed based on the interval specified in the **Preferences** window.

© 2026 HPCC Systems®. All rights reserved

Find (Workunits) Toolbox

This toolbox allows you to search online Workunits in the environment to which you are connected.



User	If enabled, this filters on the username. Enter a user's name to see only that user's Workunits.
Cluster	If enabled, select a cluster from the drop list to see only that cluster's Workunits.
State	If enabled, select a State from the drop list to see Workunits with that state.
Date Range	If enabled, this allows you to specify a range of dates/times to limit the display results
Max Count	Limits the number of Workunits to display.
Search	Press this button to initiate the search. When completed, search results are displayed at the bottom of the toolbox.
Clear	Press this button to clear the search results.

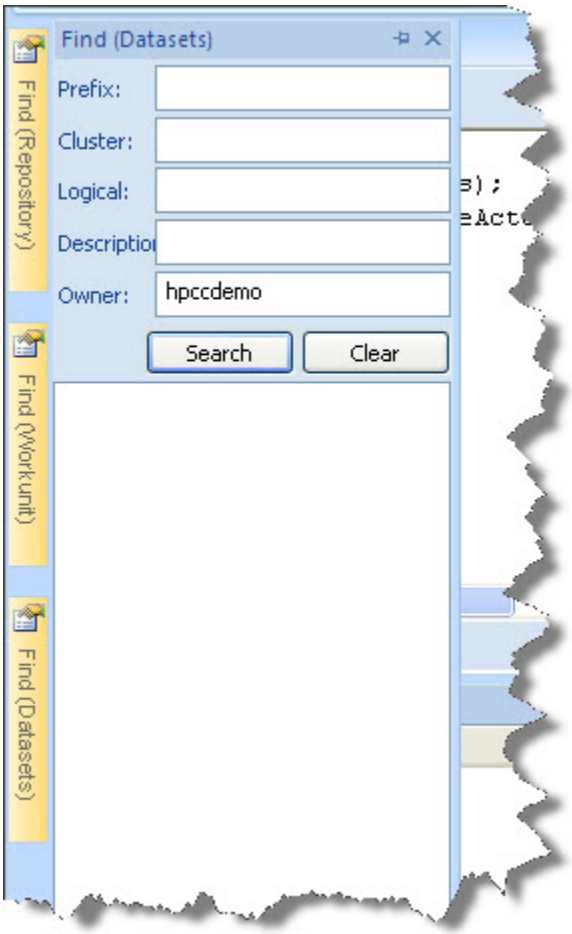
© 2026 HPCC Systems®. All rights reserved

Datasets Toolbox

This toolbox displays Logical files for the environment to which you are connected. Logical files are are organized into "folders" by scope in which they were created and displayed in a tree format. Expand any branch to see the files within it. Right-click to copy the label of the file to paste into your code.

Find (Datasets) Toolbox

This toolbox allows you to search for Logical files in the environment to which you are connected.



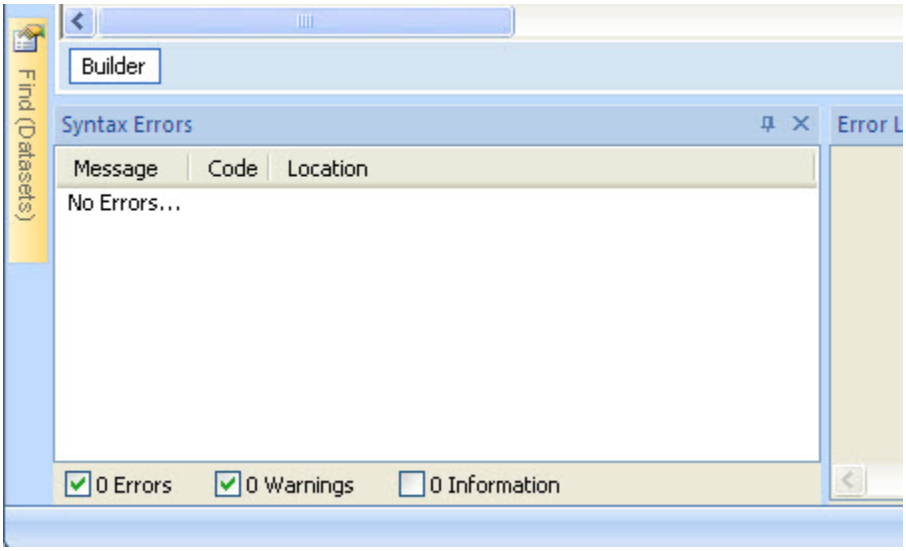
Prefix	This filters on the logical file's prefix.
Cluster	This filters on a cluster. Specify a cluster name from the environment.
Logical	Specify all or part of a file's logical name. Use an asterisk to wildcard a name (for example, *person* find any file containing person in its logical name.
Description	Specify all or part of a file's description. Use an asterisk to wildcard a description (for example, *person* find any file containing person in its description.
Owner	Specify an Owner name to find only files created by a specific user.

© 2026 HPCC Systems®. All rights reserved

Syntax Errors Toolbox

The Syntax Errors toolbox displays syntax errors and warnings if any are found when conducting a Syntax Check. A syntax check is also performed when submitting a job (pressing Submit or Go button).

For example, referencing an ECL file which has not yet been defined will display a syntax error as shown below.

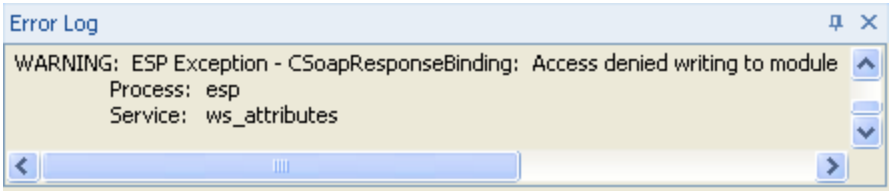


DOUBLE-CLICK on any error displayed to take you to the line containing the error. If the error is in a different attribute, a DOUBLE-CLICK on the error opens that attribute in an editor window.

Error Log Toolbox

The Error Log toolbox displays errors and warnings from the HPCC Systems platform.

For example, attempting to create an ECL file in a folder without WRITE permission will display an access denied error as shown below:

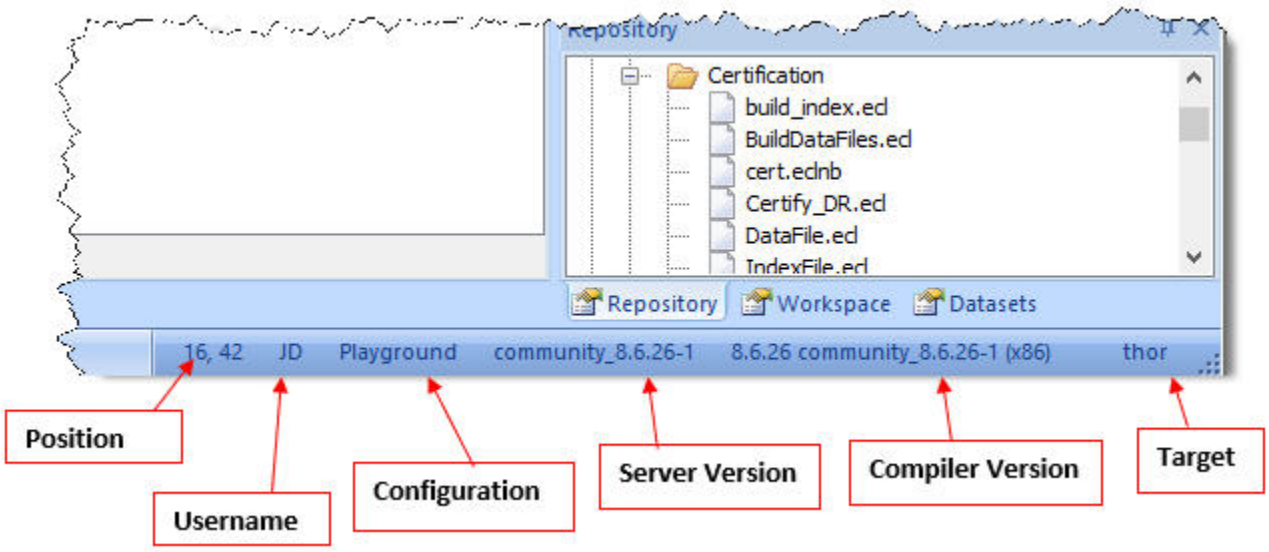


	You can copy the error message text using the right-click pop up menu option.

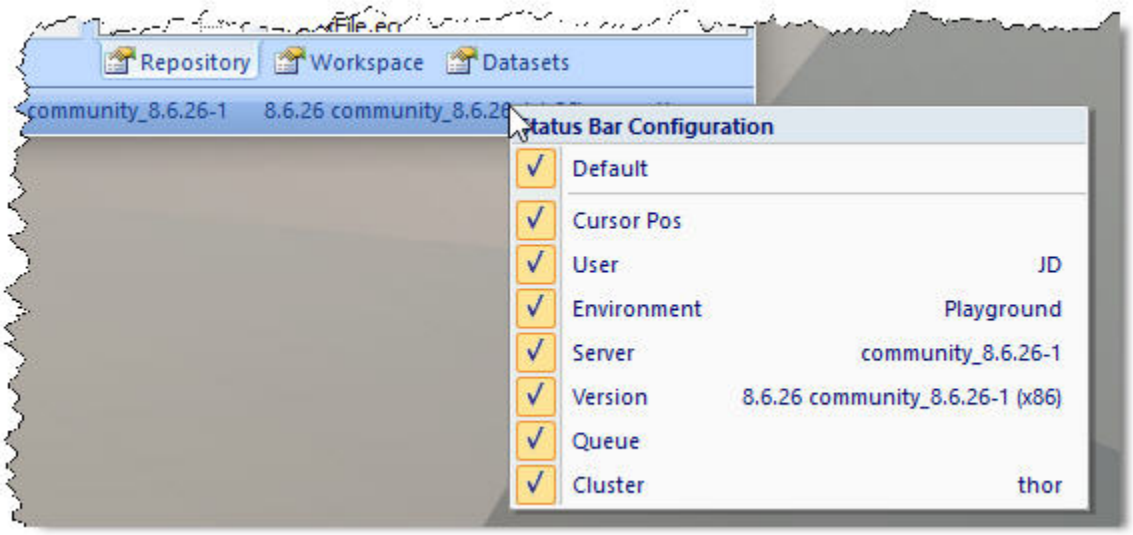
© 2026 HPCC Systems®. All rights reserved

Status Bar

The Status Bar displays useful information about the current file, current server, compiler, and target.



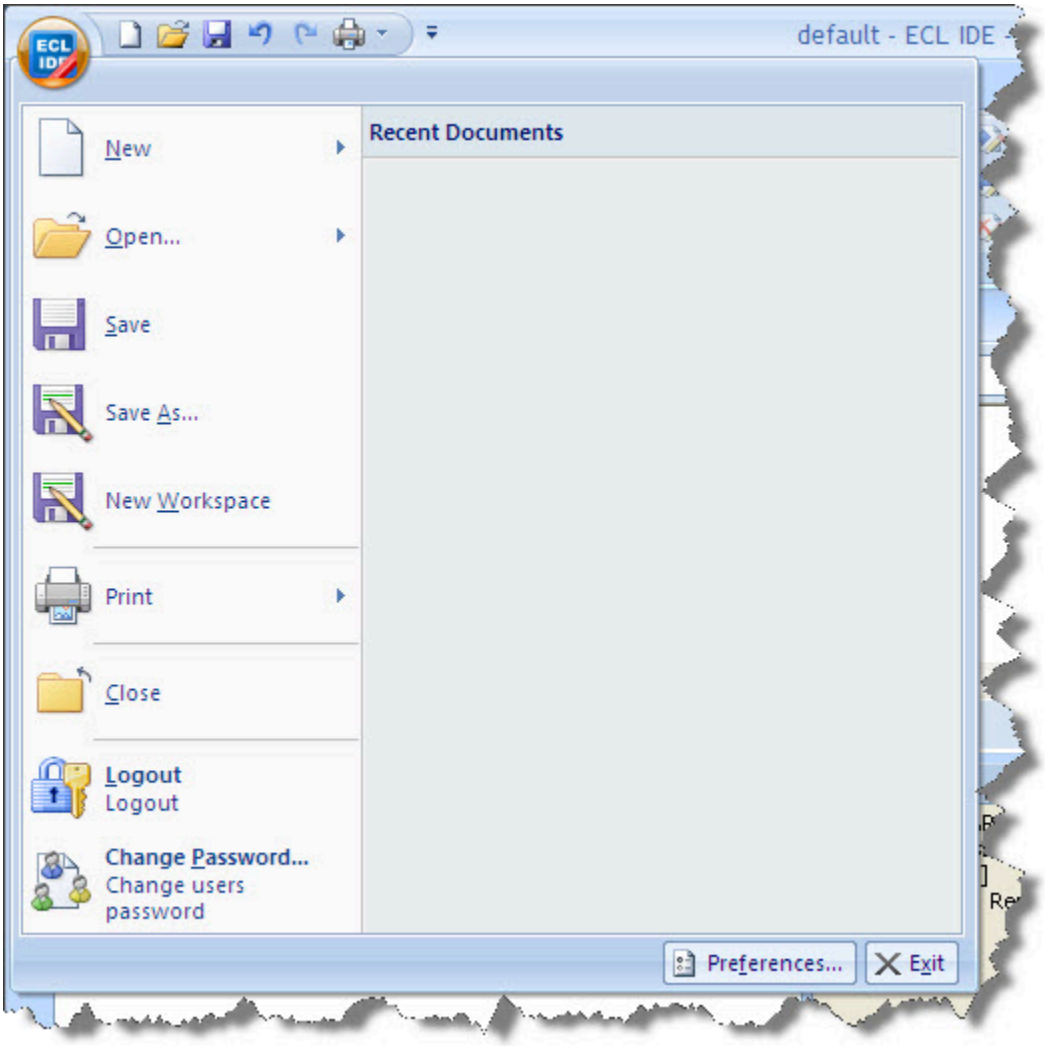
You can select the items you want displayed by right-clicking on the status bar. Check the boxes of the items you want displayed.



© 2026 HPCC Systems®. All rights reserved

Orb Menu

Orb Menu



The Orb button menu provides a list of basic actions enabling you to create and gain access to files and workspaces. It also supplies some basic administrative functions which can be used to control and specify access details for the environment(s) you use. It contains the following selections:

New	Opens a new ECL Builder window. CTRL-N
Open	Provides access to the following Open submenus: **Open Builder:** Opens a saved query (.ECL file) in a new ECL Builder window. CTRL+O **Open Workunit:** opens a dialog to name the Workunit to open. Enter the Workunit ID (WUID), then press the **OK** button.
Save	Saves the contents of the window with input focus. An ECL Builder window query is saved to an .ECL file. An Editor window saves the ECL file to the Repository. CTRL+S
Save As	Saves the contents of the window with input focus to a new filename.

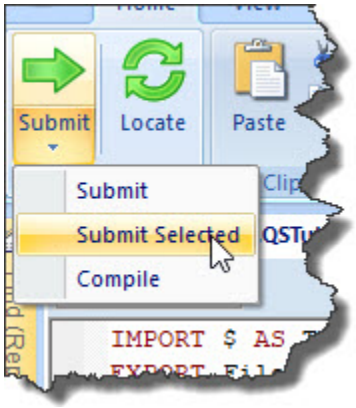
© 2026 HPCC Systems®. All rights reserved

New Workspace	Opens a new blank workspace.
Print	Provides access to the following Print submenus: **Print Setup:** opens a page setup dialog where you can set margins, paper size, and orientation, etc. **Print:** prints the contents of the active window. CTRL+P
Close	Closes the active window. CTRL+F4
Logout	Closes the session and allows you to login again.
Change Password	Allows you to change your password.
Exit	Closes the ECL IDE.
Preferences	Opens the Preferences window to allow you to configure ECL IDE options.

Ribbon Tabs

Current Tab

The **Current Tab** contains the following selections:

	Submit	Submits the ECL code to be compiled and executed on the target cluster.
	**Submit drop list**	Choose from Submit, Submit Selected, or Compile • **Submit** - submits the ECL code to be compiled and executed on the target cluster. • **Submit Selected** - submits the selected code segment to be compiled and executed on the target cluster. • **Compile** - compiles but does not execute the ECL. Allows the workunit to be published for later execution.
	Locate	Locates the ECL file selected in a builder or editor window. F9

Clipboard Tab

The **Clipboard Tab** contains the following selections:

	Paste	Standard Windows text paste operation. CTRL-V

© 2026 HPCC Systems®. All rights reserved

	Cut	Standard Windows text cut operation. CTRL-X
	Copy	Standard Windows text copy operation. CTRL-C

Editor Tab

The Editor Tab contains the following selections:

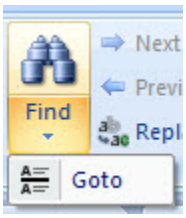
	Undo	Restores the text to state it was in before the last change (multiple Undo calls will restore each successive change back until the text is in its original format). CTRL-Z
	Redo	Restores the text to state it was in before the last Undo (multiple Redo calls will restore each successive change back until the text is in its final changed format). CTRL-Y
	Select All	Highlights all text in the currently active window. CTRL-A
	Delete	Removes selected text from the currently active window delete.
	Record\Stop	Record macros for playback later. CTRL-SHIFT-R
	Play	Playback recorded macro**.** CTRL-SHIFT-P
	Advanced...	The Advanced editor options are as follows:
	Complete Word	A list of keywords and folders from which you can select. This allows you to complete a word by selecting from the list. CTRL-Space
	Match Brace	When the insertion point is adjacent to a grouping character (parenthesis, square bracket, or curly brace) moves the insertion point to the matching grouping character, beginning or end. CTRL-E
	Select to Brace	When the insertion point is adjacent to a grouping character (parenthesis, square bracket, or curly brace) selects to the matching grouping character. CTRL-SHIFT-E
	Toggle Comment	Toggles comment characters before selected lines. CTRL-Q
	Box Comment	Encloses selected text in a comment structure. CTRL-SHIFT-Q
	Uppercase	Changes case of selected text to uppercase. CTRL-SHIFT-U

© 2026 HPCC Systems®. All rights reserved

	Lowercase	Changes case of selected text to lowercase. CTRL-U
	Line Numbers	Toggles line numbers in the editor. CTRL-L
	Word Wrap	Toggles word wrapping in the editor. CTRL-W
	Set Tab Size	Select to set a tab width of between 1-16.
	Toggle Whitespace Chars	Toggles between displaying whitespace characters or not.

Search Tab

The **SearchTab** contains the following selections:

	Find	Locates specific text in the currently active window. CTRL-F
	**Goto**	Press the lower half of the Find icon to access the Go To Line dialog. Enter the line number to move your cursor to.
	Find Next	Locates the next instance of the same specific text in the currently active window. F3
	Find Previous	Locates the previous instance of the same specific text in the currently active window. CTRL-F3
	Replace	Locates specific text in the currently active window and replaces it with new text. CTRL-H

Syntax Tab

The **SyntaxTab** contains the following selections:

	Check Syntax	Checks the ECL code in the window with input focus for syntax errors.
	Next Error	Moves the insertion point to the next error and changes the error message displayed.
	Previous Error	Moves the insertion point to the previous error and changes the error message displayed.
	Clear Errors	Clears the errors displayed in the Syntax Errors toolbox.

ECL Tab

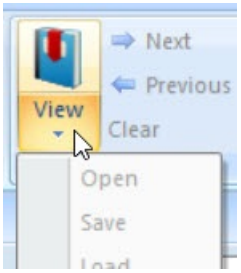
The **ECL Tab** contains the following selections:

© 2026 HPCC Systems®. All rights reserved

Show Complexity...	displays a complexity score. Use this option to evaluate and optimize queries.
Show Dependency...	displays a list of ECL files upon which the current ECL file depends.
Copy Title	copies the title of the active tab to the clipboard.
Locate Current	highlights the current ECL file in the tree
Locate Selected	highlights an item in the Repository tree when its name is highlighted in a text control.
Open Selected	opens an item in the Builder window when its name is highlighted in a text control.

Bookmarks Tab

The **Bookmarks Tab** contains the following selections:

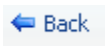
	Press the **Bookmarks** icon to access the Bookmarks panel.
	Press the lower half of the Bookmarks icon to access the submenu: **Open** - Opens the selected bookmark. **Save** - Saves the current bookmarks. **Load** - Loads the last saved bookmarks. **Load Merge** - Merges the last saved bookmarks with the current bookmarks. **Delete** - Deletes the selected bookmark.
	Next bookmark
	Previous bookmark
Clear	Clears bookmarks

Browser Tab

The **Browser** tab is shown only when the ECL Watch page is displayed for a workunit. To view the ECL Watch page from within ECL IDE, simply click on the workunit ID during or after it has completed running.

© 2026 HPCC Systems®. All rights reserved

ECL Watch is one of the pages that can be viewed at this point. The Browser tab is shown at the top of the Builder window and contains the following selections:

	Stop	Stops the web page from further processing.
	Refresh	Reloads the current page to make sure you have the latest version.
	Launch New Window	Opens the current page in a new browser window.
	**Back**	Returns to the last page viewed.
	Forward	Moves forward to the next page (only available after having moved back).

Window Tab

The **Window** Tab contains general actions for controlling open windows on the desktop.

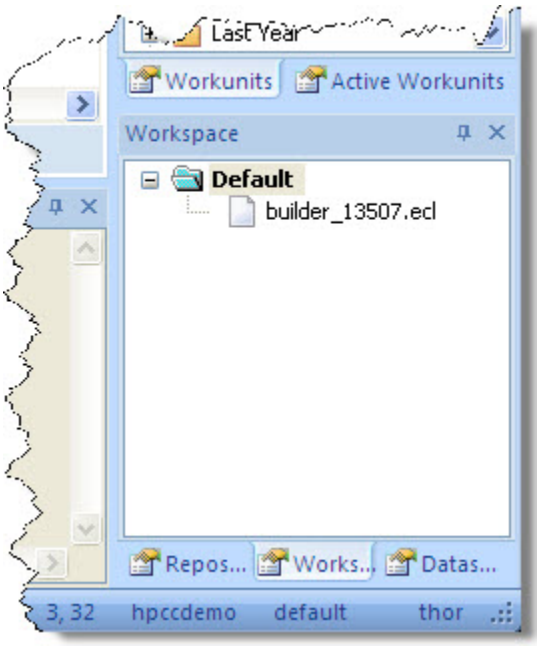
Cascade	displays all open windows so that their top left corner is visible from beneath the window on top.
Tile	displays all open windows one above the other
Arrange Icons	lines up icons for all minimized windows
Sort	sorts open windows
Group	groups similar windows together
Degroup	de-groups windows previously grouped

As well as these options, there is also a Windows drop menu on this tab. These are provided to help you control specific windows which are open on the desktop. All currently open windows are listed on this menu and you can switch to any window listed by simply clicking on the one you want to view. Other available selections are:

New Builder	Opens a new, empty builder window.
Close All	Closes all open windows.
Windows	Displays a dialog listing all currently open windows. Simply click on one or more windows to highlight them in the list and press the required button. Activate, displays the window(s) you selected, OK takes you back to where you were, Save, enables you to save the selected window(s), and Close Window(s) closes only those windows highlighted in the list.

© 2026 HPCC Systems®. All rights reserved

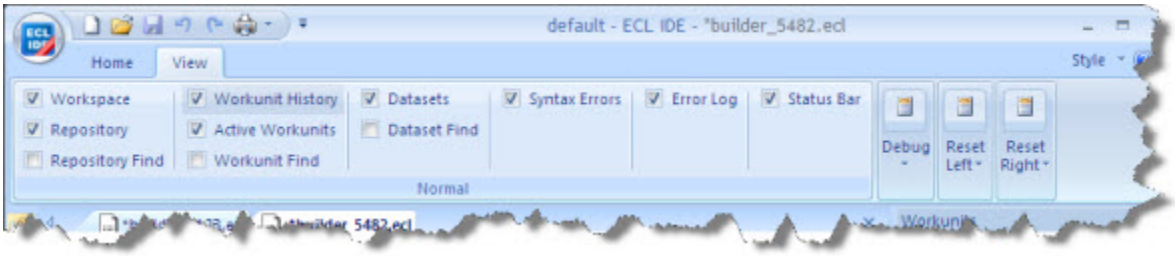
Workspace(s) Tab



The **Workspace Tab** tab allows you to preserve the workspace you are using for future use or remove an existing workspace that you no longer need. Once you have organized a workspace, use **Add** to save it. Press **Remove** to delete the workspace you are currently using.

Your saved workspace(s) are shown in the list under the **Default** option. Selecting a workspace as the default will ensure that every time ECL IDE is started, the selected workspace is automatically loaded ready for you to use.

View Tab



The **View** tab allows you to choose which toolboxes to view. In addition, it has options to reset the positions of these toolboxes to any of the predefined locations.

Help Menu

The **Help** menu contains the following selections:

Download Resources	opens the Download Resources page of ECL Watch in a browser
About	displays information about the program and the server

© 2026 HPCC Systems®. All rights reserved

Pop-up Menus

Workunit Pop-up Menu

Right-click on any Workunit in the **Workunits** panel for a Pop-up menu containing:

Open Results	opens an ECL Builder window displaying the results tab
Show Graph	displays the graph for the selected workunit
Browse ECL Watch	opens a web page of the ECL Watch tool
View Exceptions	displays the contents of the Exceptions tab for the selected workunit.
Open ECL	opens an ECL Builder window displaying the ECL code for the Workunit
Save as Excel Workbook	saves workunit results to an Excel workbook. Each result set is saved as a sheet in the workbook.
Copy WUID	copies the Workunit ID to the clipboard
Abort	stops an executing Workunit
Resubmit	reruns the Workunit
Delete	removes the Workunit from the system data store
Protect	protects the Workunit (prevents deletion and archiving)
Unprotect	removes protection from the Workunit
Follow Up	enables you to mark the selected workunit so you can come back to it later. The **Flag** option sets the marker and **Clear** removes a flag that was previously set.
SetPriority	allows you to set the priority in the queue for a running Workunit. You can move a workunit up or down one place in the queue, or send a workunit to the top or bottom of the queue.
SetState	allows you to set the state of a Workunit.
Refresh	refreshes the display

Repository Pop-up Menu

Right-click in the **Repository Window** for a Pop-up menu containing options which are available depending on whether you have a folder or ECL file selected. Note, if you have TortoiseGit installed, the menu is extended. See Version Control for details

Open in Builder Window	Opens the selected ECL file in an ECL Builder window
Locate File in Explorer	Only valid in locally compiled workunits
Copy "<attribute_name>"	Copies the fully qualified ECL file name to the clipboard
Paste Selection	Pastes the ECL file name you copied where the cursor is currently positioned
Check Syntax	Displays a sub-menu of all available INVOKE structures for the selected ECL file
Invoke	Displays a sub-menu of all available INVOKE structures for the selected ECL file
Insert Folder	Adds a new 'nested' folder within the currently selected folder
Rename Folder	Renames the selected folder

© 2026 HPCC Systems®. All rights reserved

Delete Folder	Deletes the selected folder
Insert File	Adds a new ECL file inside the selected folder
Rename File	Renames the selected ECL file
Change Type	Displays a sub-menu of available file types you can choose to change
Copy File	Enables you to copy the selected ECL file so it can be pasted into a different folder
Move File	Moves the selected ECL file to specified folder
Delete File	Moves the selected ECL file to the Trash folder on your desktop. You can also press the **Delete** key to access this functionality
Create Mod File	Combines the selected ECL files into a MOD file
Refresh	Refreshes the repository folder display
Collapse All	Collapses all open repository files and folders

Results Pop-up Menu

Right-click on Results Pop-up menu containing:

Save	saves the results to a file
SaveAs	saves the results to a new file
Save As XML	saves the results to a file in XML format
Cut	removes (from display) and copies the selected results to the clipboard
Copy	copies the selected results to the clipboard
Paste	pastes contents of clipboard into result.
Copy As ECL	copies the selected results to the clipboard in ECL format
Send to ""	sends the selected results to the application associated with CSV files (for example, Microsoft Excel)
Select All	selects all results
Select Row	selects the current row
Select Column	selects the current column
Delete Selection	deletes the selected results from displayed results
Show Column in Hex	displays hex value for the current column
Format	displays a submenu with options to format selected cells.
Auto Size Column	adjusts the current column width to allow all characters to display
Auto Size All Columns	adjusts the all column widths to allow all characters to display
Find	locates specific text in the results
Find Next	locates the next instance of the same specific text in the results
Find Previous	locates the previous instance of the same specific text in the results
GoTo	positions the cursor at a specified row number in the results
Drilldown	sends the selected result rows to a previously defined Drilldown ECL file. This method sends the selected rows to an ECL file as a set.
Drilldown (1 per row)	sends the selected result rows to a previously defined Drilldown ECL file. This method sends each row to the Drilldown separately.

© 2026 HPCC Systems®. All rights reserved

Drilldown II	generates ECL code that ends with a call to the selected MACRO ECL file.
Copy WUID	copies the Workunit ID to the clipboard
Save As Excel Workbook	sends the result set to a Excel Workbook with each result set on a sheet.
Sync Workunit	locates the current Workunit in the Workunit History tree
Locate Workunit in Explorer	Opens a File Explorer window to the location of the workunit file.
Close Workunit	closes the current Workunit
Delete Workunit	deletes the current Workunit
Close All Workunits	Closes all open workunits

Using Drilldown

There are three different versions of drilldown: Drilldown, Drilldown (1 per row), and Drilldown II.

Drilldown sends the selected result row(s) to the selected MACRO ECL file as a single string of XML data.

Each selected row is contained within <row></row> XML tags. The MACRO must use LOADXML to instantiate an XML scope and should use #FOR(row) to process each separate record. Each field in the record is treated as a template symbol and may be referenced in the MACRO as %fieldname%.

Example Drilldown MACRO:

```
export TestDrilldown(xmlRow) := MACRO
LOADXML(xmlRow);
#DECLARE(Ctr)
#SET(Ctr, 0 )
#DECLARE(OutputStr)
#SET(OutputStr, 'OUTPUT(ProgGuide.Person.file(PersonID IN [')
         )
#FOR(row)
#IF(%Ctr% = 0)
#APPEND(OutputStr, %'personid'% )
#SET(Ctr, 1 )
#ELSE
#APPEND(OutputStr,',' + %'personid'% )
#END
#END
#APPEND(OutputStr,']));\n')
%OutputStr%
ENDMACRO;
```

Drilldown (1 per row) sends the selected result row(s) to the selected MACRO ECL file as XML data. If multiple rows are selected, each is sent to a separate MACRO call as a single string of XML data. The MACRO must use LOADXML to instantiate an XML scope. Each field in the record is treated as a template symbol and may be referenced in the MACRO as %fieldname%.

Example Drilldown (1 Per Row) MACRO:

```
export TestDrilldownOnePerRow(xmlRow) := MACRO
LOADXML(xmlRow);
OUTPUT(ProgGuide.Person.file(PersonID = %personid%));
ENDMACRO;
```

Drilldown II generates ECL code that ends with a call to the selected MACRO ECL file. Each field in the result is defined as a separate set ECL file and as a field in an inline DATASET. The MACRO should be written to make use of whichever of the ECL files best allows it to accomplish its task (typically, to show all base data from which the result set was drawn).

© 2026 HPCC Systems®. All rights reserved

Example Drilldown II MACRO:

```
EXPORT TestDrilldownII := MACRO
OUTPUT(ProgGuide.Person.file(personid IN
          Set_personID));
ENDMACRO;
```

Keyboard Shortcuts

The following table lists all the keyboard shortcuts available in the IDE.

Keyboard Shortcut	Action
F1	Online Help (Syntax)
Ctrl+F	Find
F3	Find Next
Shift+F3	Find Previous
Ctrl+H	Replace
F4	Next Error
Shift+F4	Previous Error
F5	Refresh Repository
F7	Check Syntax
F8	Locate File in Explorer
F9	Locate Current FIle
F11	Locate Selected File
F12	Open Selected File
Ctrl+Enter	Submit ECL
Ctrl+Shift+Enter	Submit Selected ECL
Ctrl+X / Shift+Delete	Cut
Ctrl+C / Ctrl+Insert	Copy
Ctrl+V / Shift+Insert	Paste
Ctrl+Z / Alt+backspace	Undo
Ctrl+Y	Redo
Ctrl+F4	Close File/Builder Window
Alt+F4	Close all and exit
Ctrl+Q	Toggle Comment with //
Ctrl+Shift+Q	Toggle Block Comment /*
Ctrl+L	Toggle Line Numbers
Ctrl+S	Save
Ctrl+O	Open
Ctrl+P	Print
Ctrl+N	New Builder Window
Ctrl+G	Go to Line

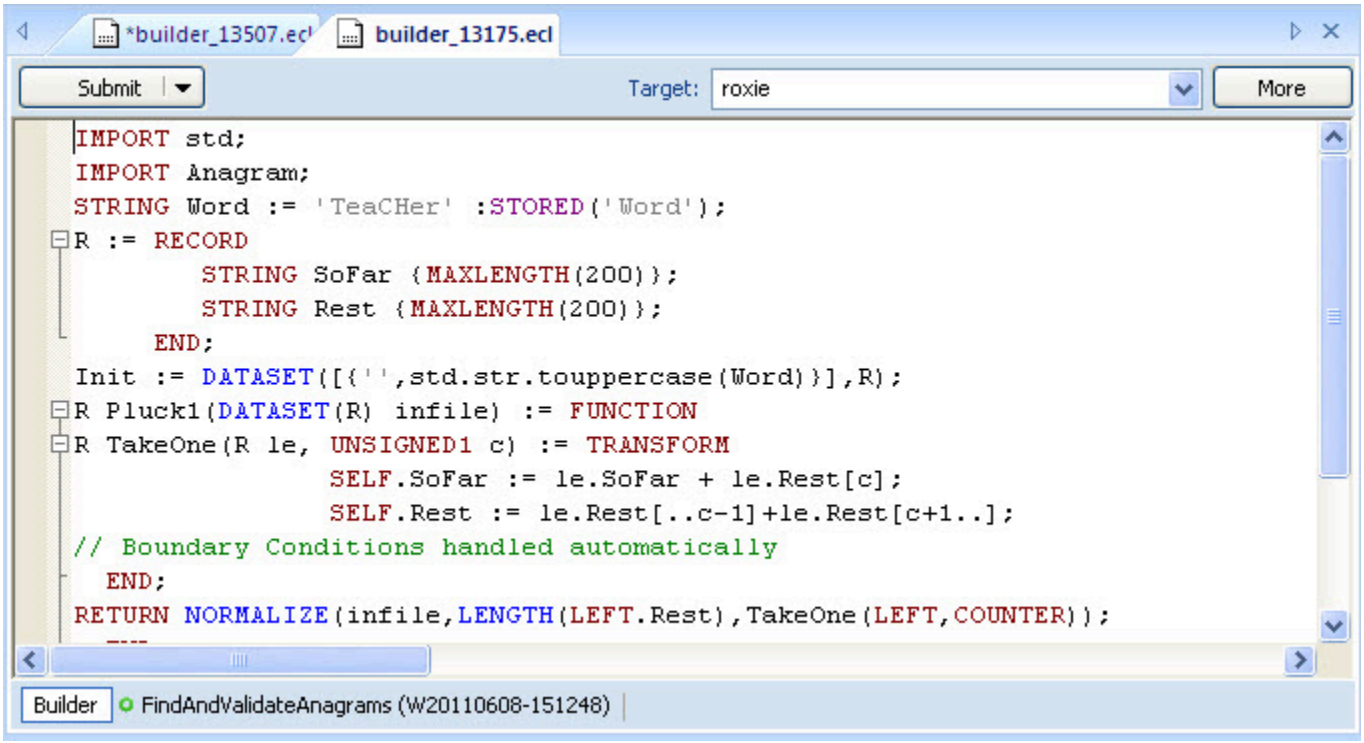
© 2026 HPCC Systems®. All rights reserved

Keyboard Shortcut	Action
Alt+0	Fold All - collapses all elements that can be collapsed
Ctrl+Alt+0	Unfold All - expands all collapsed elements
Ctrl+A	Select All
Ctrl+D	Duplicate Line
Ctrl+T	Switch the current line position with the previous line position
Ctrl+Space	Complete Word
Ctrl+E	Match Brace
Ctrl+Shift+E	Select to Brace
Ctrl+W	Toggle Word Wrap
Ctrl+Alt+C	Copy Filename
Ctrl+Shift+U	Uppercase
Ctrl+U	Lowercase
Ctrl+Shift+R	Record/Stop Macro
Ctrl+Shift+P	Play Macro
Ctrl+Tab	Next Tab
Ctrl+Shift+Tab	Previous Tab
Alt+Shift+Arrow keys	Column Mode Select
Alt + Left Mouse Drag	Column Mode Select
Ctrl+BackSpace	Delete to start of word
Ctrl+Delete	Delete to end of word
Ctrl+Shift+BackSpace	Delete to start of line
Ctrl+Shift+Delete	Delete to end of line

© 2026 HPCC Systems®. All rights reserved

ECL Builder Window

This window allows you to edit, build, and run ECL Queries. It also shows results and Workunit information for each Workunit created by submitting the ECL code.



- **Submit** submits the ECL code to be compiled and executed on the target cluster. Press the arrow on the submit button to display other options.
- **Submit Selected** submits only the selected ECL code for execution.
- **Compile** compiles but does not execute the ECL. This allows the workunit to be published for later execution.

The **More** button displays advanced options. See the Advanced Builder Window Options section below for details. An asterisk on the button indicates that advanced options have been set.

The tabs beneath the text area represent Workunits created by submitting the ECL code. Select a Workunit tab to see results, graphs, and the ECL watch page for the Workunit.

You can save your queries to an ECL file by pressing the **Save** button on the toolbar, or selecting the **File > Save** menu selection.

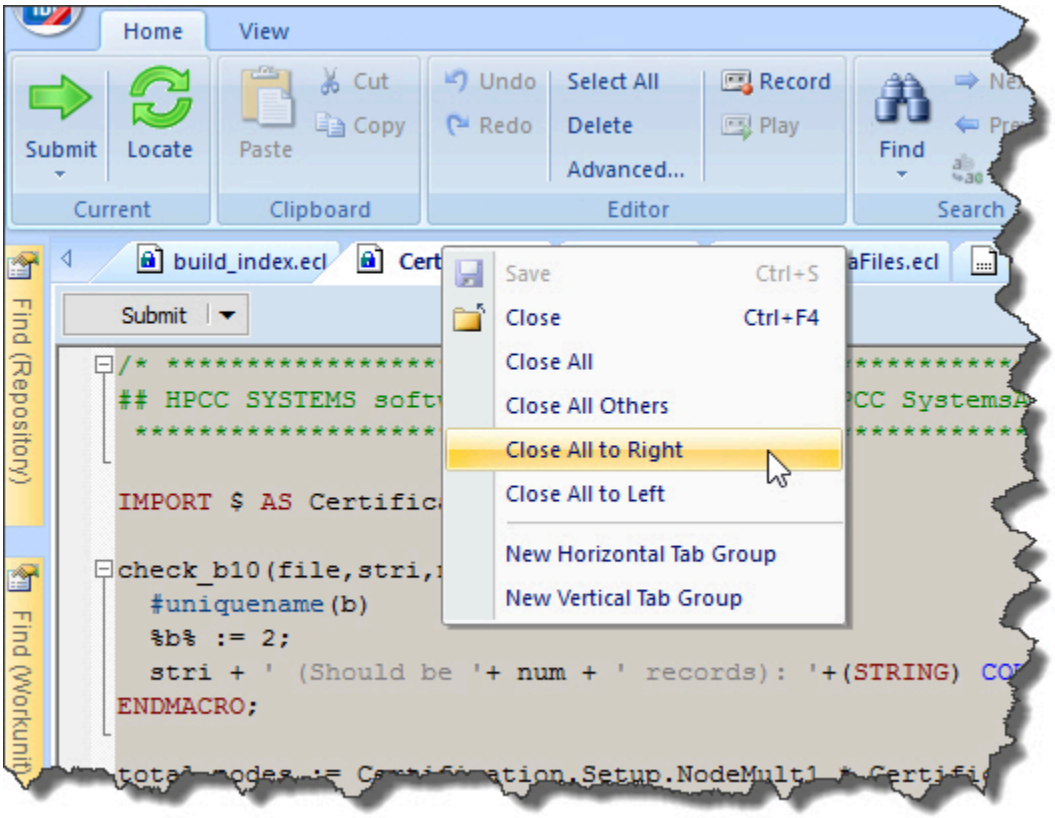
Using CTRL+MOUSEWHEEL will grow or shrink the font size used in the window.

There is also a right-click pop-up menu with the all the options found on the **Edit** menu.

© 2026 HPCC Systems®. All rights reserved

Builder Tab Options

When working in the IDE builder window you may find that you have numerous tabs open. Right-click on any of the open tabs to display tab options.



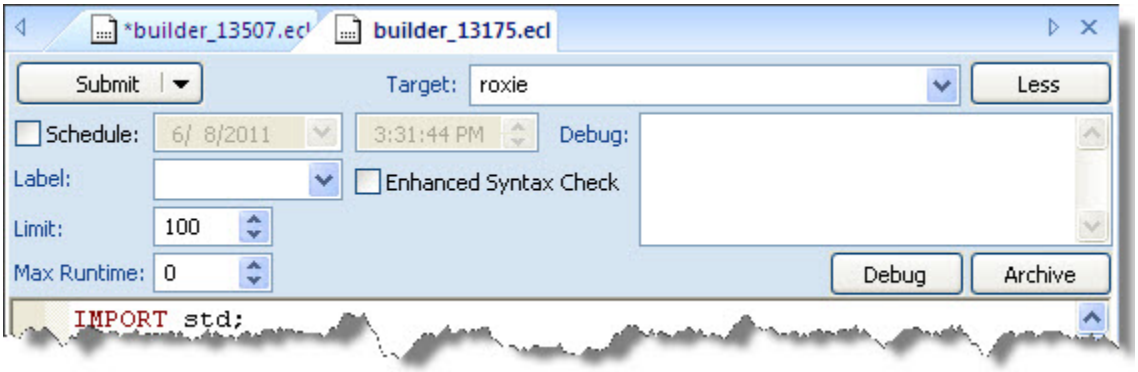
Select the desired action to perform on the tabs.

- **Save** Saves the selected tab.
- **Close** Closes the selected tab.
- **Close All** Closes all open tabs.
- **Close All Others** Closes all open tabs except the selected tab.
- **Close All to Right** Closes all open tabs to the right of the selected tab.
- **Close All to Left** Closes all open tabs to the left of the selected tab.

© 2026 HPCC Systems®. All rights reserved

Advanced Builder Window Options

Press the **More** button to display the advanced options. Press the **Less** button to close this area.



The **Submit** button submits the workunit. Select the options you want to use when running the workunit as follows:

The **Target** drop list shows the cluster where the query will be sent. Select Local, if you want to run the query locally on your computer.

The **Schedule** checkbox allows you to schedule queries to run at a specific time. When enabled, the **Submit** button becomes a **Schedule** button. The **Schedule** button sends a scheduled query to the HPCC.

When the job is submitted it is assigned a workunit ID and appears in the workunit list as *scheduled* (with a clock icon). You can monitor the workunit in the same manner as a submitted workunit.

The **Label** drop list allows you to select a Repository Label which specifies a previous version of ECL files.

The **Limit** spinbox allows you to override the default setting for result limits.

The **Max Runtime** spinbox allows you to specify a timeout that will abort the job if exceeded.

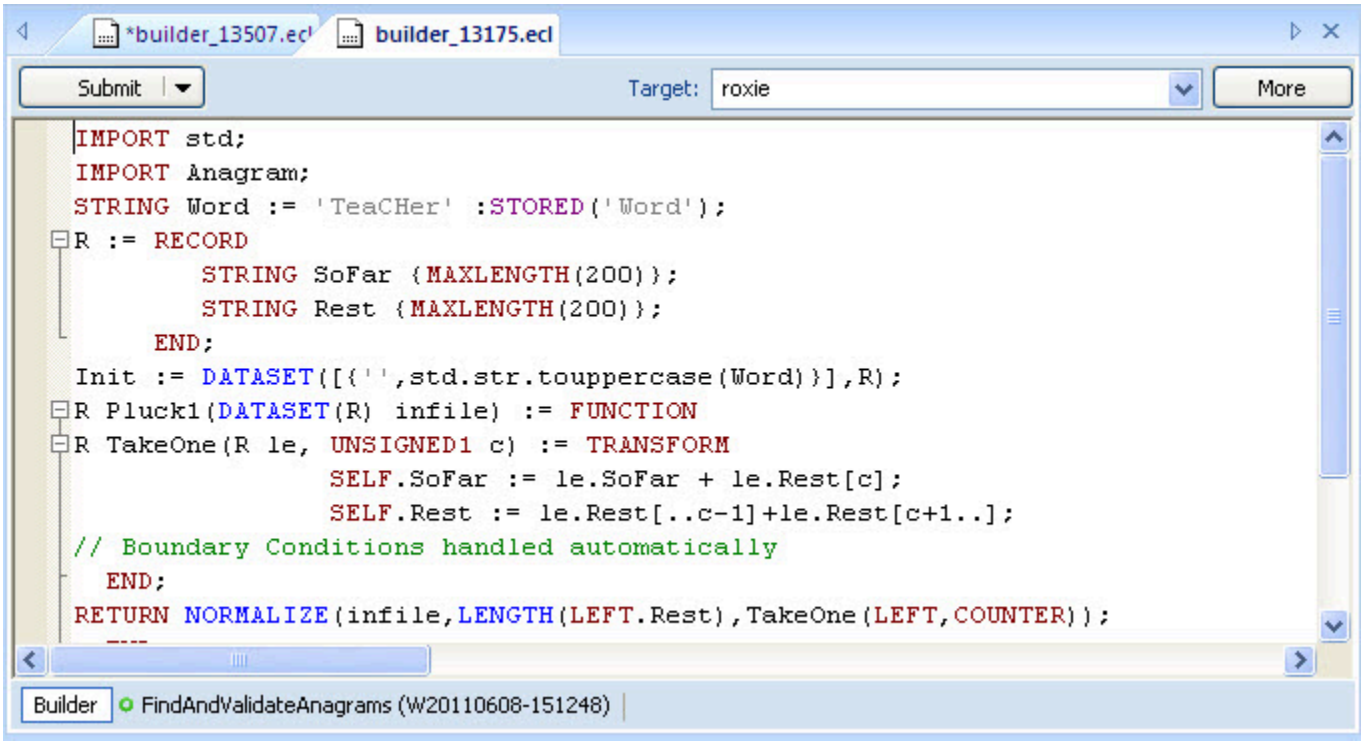
The **Archive** button is for debugging purposes and archives all ECL (including dependencies) into the workunit for examination without executing the code.

The **Enhanced Syntax Check** checkbox allows you to enable enhanced syntax checking.

© 2026 HPCC Systems®. All rights reserved

Editing

The Builder Window also supports editing ECL files.



Using CTRL+MOUSEWHEEL will grow or shrink the font size used in the window.

There is also a right-click pop-up menu with the all the options found on the **Edit** menu.

Support for Other Languages

In addition to providing native support for ECL Files, the ECL IDE also provides support for working with other languages. The IDE provides colored file type differentiation, where different file types all display in a different color. The IDE also provides colorized keywords in the supported languages. The colorization is set up as a default, but can be customized. Other languages that the IDE supports

Language	Functionality
ECL	• Repository icon colorized to identify the file type • Syntax Highlighting • Syntax Check • Submit Selected • Compile
ESDL	• Repository icon colorized to identify the file type • Syntax Highlighting • Syntax Check

© 2026 HPCC Systems®. All rights reserved

	• Generates ECL • Publish
KEL	• Repository icon colorized to identify the file type • Syntax Highlighting • Syntax Check • Generates ECL and DUD files • Submit
SPC (SALT)	• Repository icon colorized to identify the file type • Syntax Highlighting • Syntax Check • Generates ECL • Submit
DUD	• Repository icon colorized to identify the file type • Syntax Highlighting • Syntax Check

Bookmarks

The ECL IDE supports internal bookmarks. Bookmarks provide a way to mark a place that you can easily return to any time in the future.

To create a **bookmark**, a **todo**, or **hack**, simply enter the appropriate code into the place you want to mark, and save it. The syntax is in the form of an ECL comment. Note that bookmarks ARE case-sensitive.

Bookmark syntax. Note the square brackets and that it is all upper case.

```
//[BM]
```

To do syntax. Note that it is all upper case and terminated with a colon.

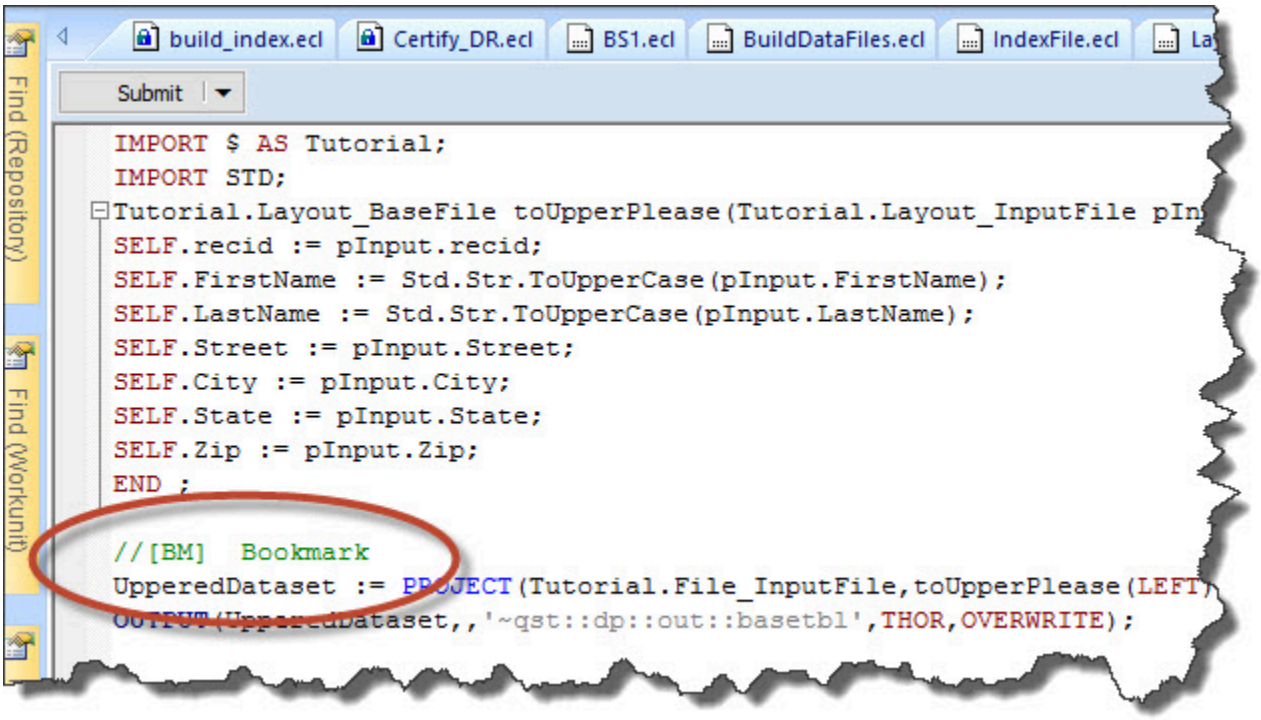
```
//TODO:
```

Hack syntax. Note that it is all upper case and terminated with a colon.

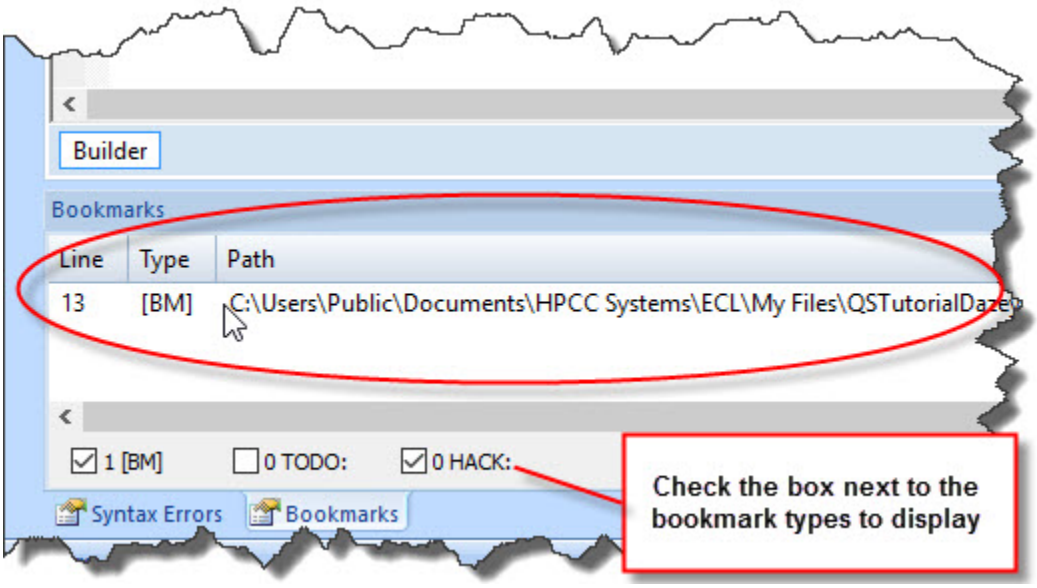
```
//HACK:
```

Enter the code into your workspace in the builder window then save.

© 2026 HPCC Systems®. All rights reserved



Your bookmark appears in the bookmarks window right below your builder window.



To return to your bookmarked place, double-click on the corresponding bookmark in the bookmarks panel.

Results Viewer

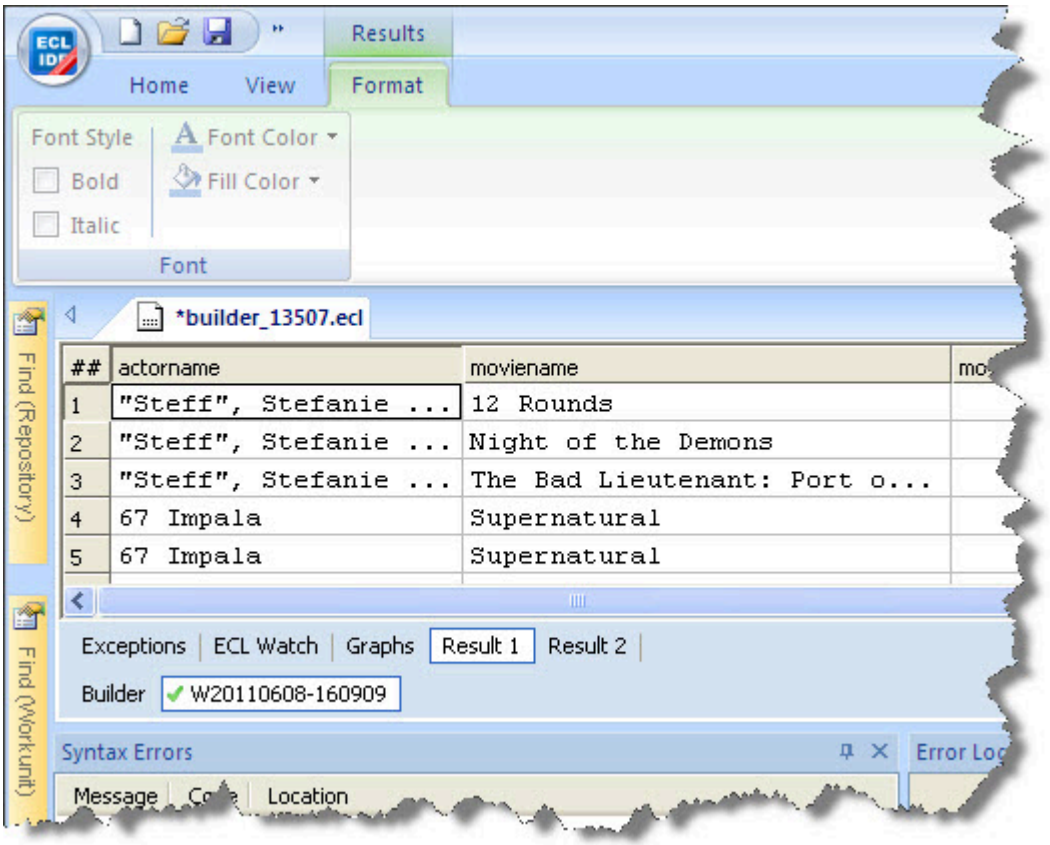
Each result is displayed on a results tab on the workunit tab.

The results window display can be configured in Preferences. Additionally, you can modify a result list display in the following ways:

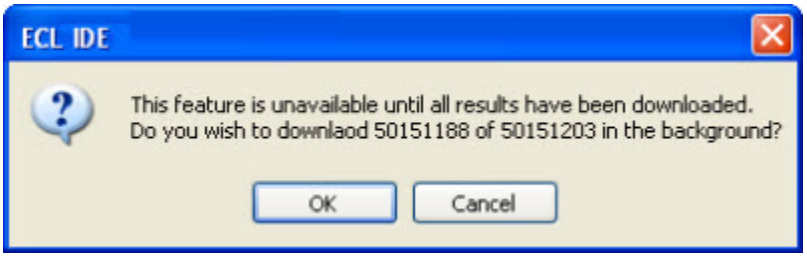
- CLICK on a column header to sort by that column.

© 2026 HPCC Systems®. All rights reserved

- Use the Format menu or toolbar to modify selected cell font or background colors.
- Use Ribbon bar buttons or right-click the popup menu. Cut, Copy, Paste, or Delete to remove or add cells, rows, columns to what is displayed. This does not change the actual result, only the display.
- Autosize one column or all columns.



If you attempt an action that requires more data, such as sorting, you are asked to confirm before the download begins. This is particularly useful when large record sets are returned.



Version Control

The ECL IDE does not directly provide version control functionality, but it does support integration with the TortoiseGit Window Shell Interface to Git version control. This open-source toolset provides easy access to Git commands from Windows Explorer. The IDE's integration extends that to the context menu of the Repository toolbox in the IDE.

© 2026 HPCC Systems®. All rights reserved

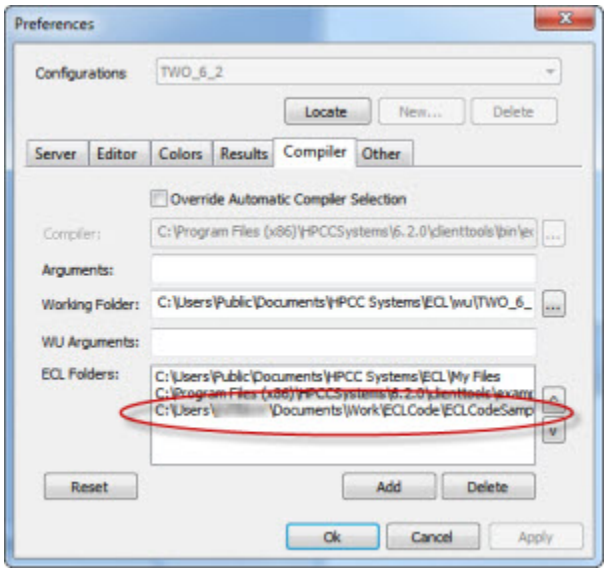
1. Install and configure TortoiseGit (www.tortoisegit.org)

   Follow the instructions on their site.

2. Clone a Git repository to a folder on your local machine.

   Follow the instructions in your Git repository's site.

3. Add that folder to the list of ECL Folders on the **Compiler** tab of the **Preferences** dialog.

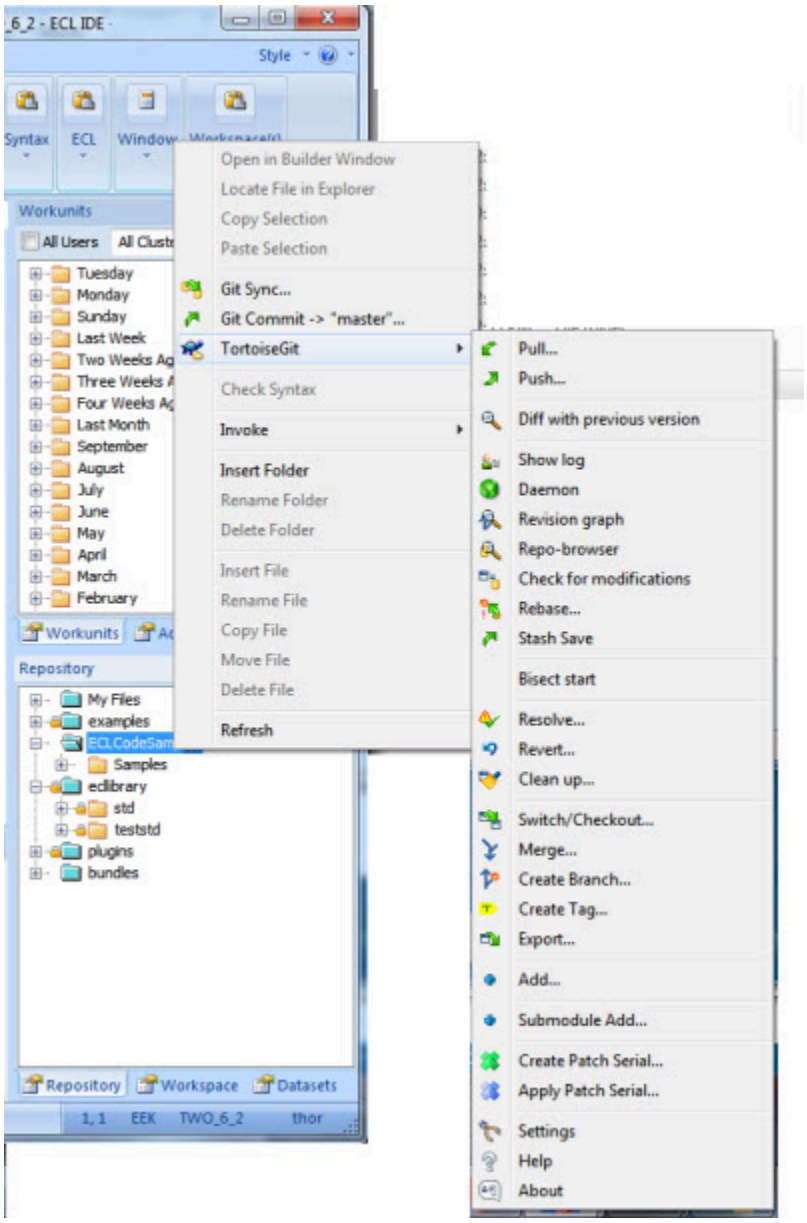


4. Press the **Ok** button.

   The folder now appears in your Repository toolbox.

5. Right-click on the folder in the Repository toolbox, and choose your Git command from the TortoiseGit menu.

© 2026 HPCC Systems®. All rights reserved



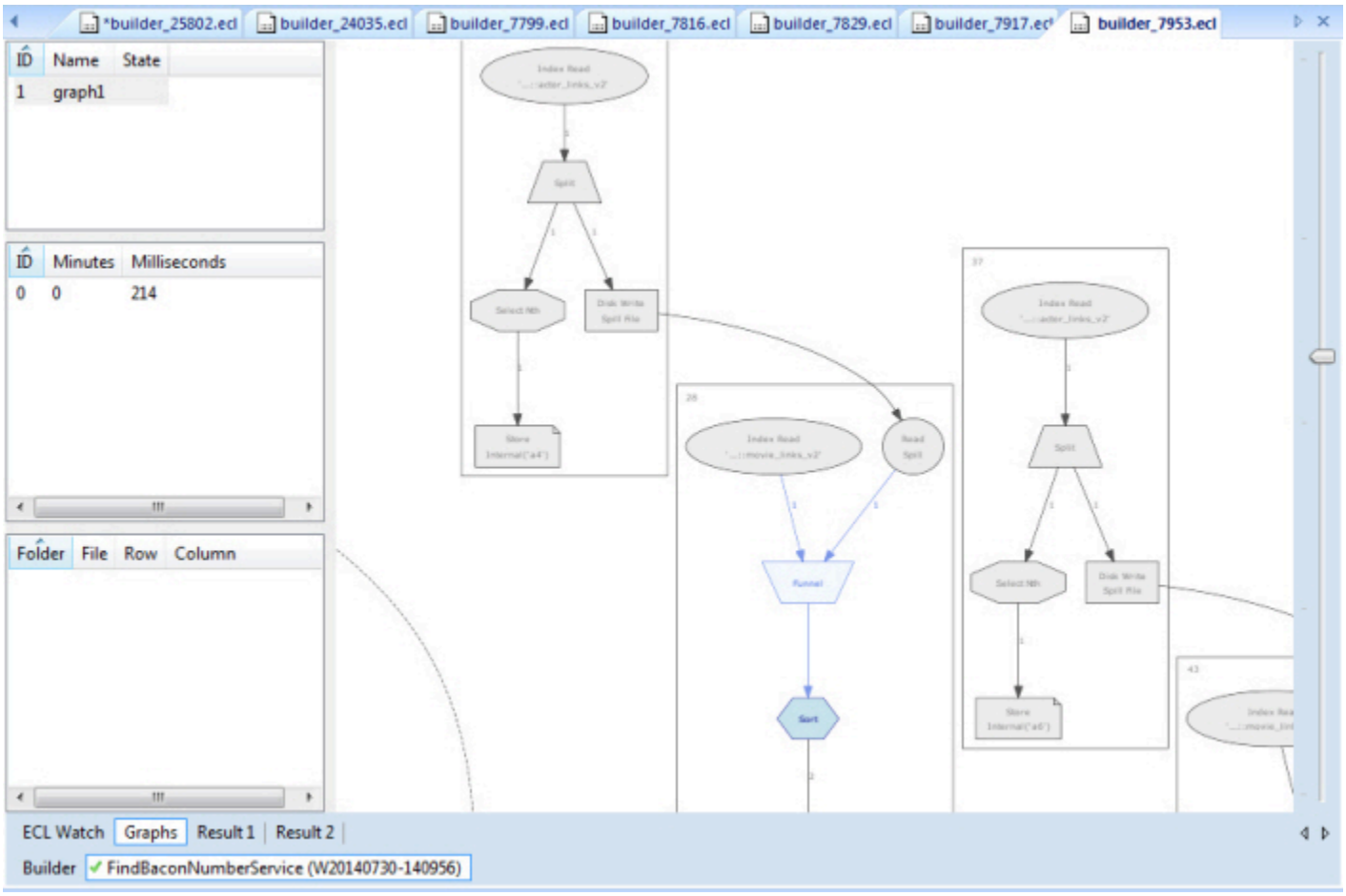
© 2026 HPCC Systems®. All rights reserved

Graph Viewer

A workunit's execution graph is displayed in the Graph Tab.

Modify the display in the following ways:

- Use the Ribbon bar Navigation buttons to move around a graph.
- Use the Ribbon bar Zoom options To Fit /To Width to adjust how the graph is displayed.
- Use the Ribbon bar Running buttons to watch the progress of a running workunit, refreshing the graph as it runs. Options are provided so you can Follow Active or Minimise Inactive areas of the graph according to your preference .



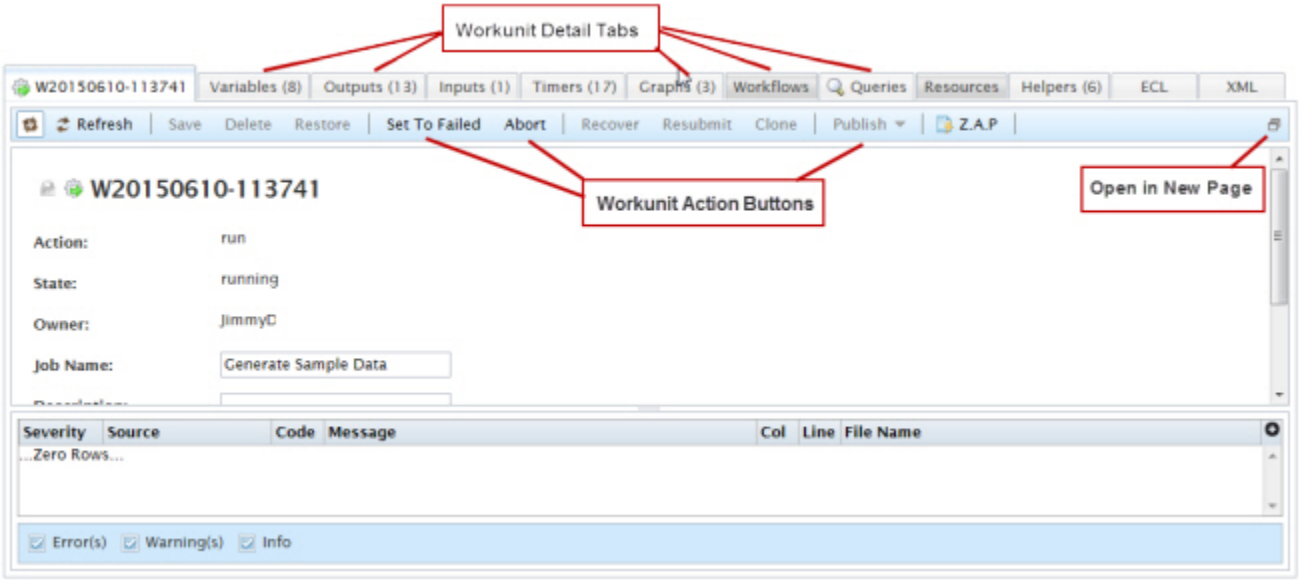
© 2026 HPCC Systems®. All rights reserved

ECL Watch View

A workunit's Workunit Details page view is displayed in the ECL Watch Tab.

The Workunit Details page provides more information about a workunit.

You can see specific information about the workunit by selecting the various Workunit Detail tabs.



You can also perform actions on the workunit using the Workunit Action buttons.

	You can copy a Workunit's ECL Watch URL by rt-clicking on the **Open in New Page** icon and selecting **Copy Shortcut**.

- Press the **Refresh** button to refresh the workunit details.
- Press the **Save** button to save any changes to the workunit.
- Press the **Delete** button to delete the workunit.
- Press the **Restore** button to restore a workunit that has been archived by Sasha.
- Press the **Set To Failed** button to change the workunit state to failed.
- Press the **Abort** button to stop a running workunit and abort the job.
- Press the **Recover** button to restart the workunit.
- Press the **Resubmit** button to resubmit the job.
- Press the **Clone** button to clone the workunit.
- Press the **Publish** button to publish the workunit.
- Press the **Z.A.P.** button to package up system information for analysis. This is useful for bug reporting or troubleshooting.

© 2026 HPCC Systems®. All rights reserved

ECL Command Line Interface

The ECL Command Syntax

ecl [--version] <command> [<options>]

--version	displays version info.
Arguments	
deploy	Create a workunit from an ecl file, archive, or dll
publish	Add a workunit to a query set
unpublish	Remove a query from a query set
run	Run the given ecl file, archive, dll, wuid, or query
results	returns the full results of a given WUID in XML format.
activate	Activate a published query
deactivate	Deactivate the given query alias name
queries	List or manipulate queries and querysets
packagemap	execute packagemap commands (for Roxie)
bundle	manage ECL bundles
roxie	execute commands for Roxie
abort	aborts one or more workunits from the given WUID or job name
status	returns the status of a given workunit or job name. If more than one is found, a list returns.
getname	returns the workunit name from the given WUID.
getwuid	returns the WUID(s) of the given workunit job name.
zapgen	generate and download the zap file for the given WUID.
sign	adds a digital signature to an ecl or text file.
listkeyuid	returns a list of all the key user IDs that can be used by the sign command.
url-secret-name	generate a secret name from a URL for automatic URL mapping

© 2026 HPCC Systems®. All rights reserved

ecl.ini

Many options can be placed in a file called **ecl.ini** in the same directory as the executable. Options that do not change very often should be put in the ini file. For example:

```
;The values below are examples, you should change them to match your platform deployment
eclWatchIP=10.150.50.12
eclWatchPort=28010
eclUserName=emilykate

eclPassword=
resultLimit=200
```

In some examples below, we'll assume ecl.ini has the above content.

We do not recommend storing your password in the INI file or environment variable (which is clear text). The password is included in the INI file for these examples to simplify the example code.

The following options can be provided in an ini file: eclWatchIP, eclWatchPort, eclUserName, eclPassword, activateDefault, waitTimeout, resultLimit.

Evaluation of options follows this order of precedence:

- command line
- ini file
- environment variable
- default value

If a username is supplied by any method, and a password is not, you are prompted for the password.

Environment Variables

Some options can be stored in Environment Variables on your machine. The following options are supported:

```
ECL_WATCH_IP
ECL_WATCH_PORT
ECL_USER_NAME
ECL_PASSWORD
ECL_WAIT_TIMEOUT
ECL_RESULT_LIMIT
ECLCC_PATH
```

We do not recommend storing your password in an Environment Variable.

© 2026 HPCC Systems®. All rights reserved

Git Support

HPCC Systems integrates native support for leveraging Git.

The *--main* option has been extended to support compiling a query directly from a Git repository. When invoked, it retrieves the ECL code specified from the Git repository, compiles the code, and runs the query. The checkout is done on the remote ECLCCServer rather than on the client machine.

Syntax

The syntax to reference a repository location:

```
<protocol:>//<urn>/<user>/<repository>#version
```

The *#version* component can be either the name of a *branch*, *tag*, or the SHA (Secure Hash Algorithm) of a *commit*.

For example:

```
ecl run thor --main demo.main@gituser/gch-ecldemo-d#version1 --server=...
```

This example runs the *demo.main* ECL code in the *version1* branch of the *gituser/gch-ecldemo-d* repository.

This feature is supported by both the ecl and eclcc command line tools.

Additional Examples

The following examples all compile the attribute *demo.main* from *version3* branch of the GitHub repository gituser/gch-ecldemo-d:

```
eclcc --main demo.main@gituser/gch-ecldemo-d#version3
eclcc --main demo.main@gituser/gch-ecldemo-d --mainrepoversion=version3
eclcc --main demo.main@gituser/gch-ecldemo-d --defaultrepoversion=version3
eclcc --main demo.main#version3 --defaultrepo=gituser/gch-ecldemo-d
eclcc --main demo.main --defaultrepo=gituser/gch-ecldemo-d --defaultrepoversion=version3
```

If you are submitting a source file query.ecl, and you want it to replace the definition of demo.main with the same repo, the following will work:

```
eclcc query.ecl --main demo.main@gituser/gch-ecldemo-d#version3
eclcc query.ecl --main demo.main@gituser/gch-ecldemo-d --mainrepoversion=version3
eclcc query.ecl --main "demo.main#version3" --mainrepo=gituser/gch-ecldemo-d
eclcc query.ecl --main demo.main --defaultrepo=gituser/gch-ecldemo-d --defaultrepoversion=version3
```

Multiple Repository Support

Queries can be compiled from multiple Git repositories. Each Git repository is treated as a separate independent package. Dependencies between the repositories are specified in a package file which is checked into the repository and versioned along with the ECL code. The package file indicates what the dependencies are and which versions should be used.

```
package.json:
{
 "name": "demoRepoC",
 "version": "1.0.0",
 "dependencies": {
   "demoRepoD": "gituser/gch-ecldemo-d#version1"
  }
}
```

© 2026 HPCC Systems®. All rights reserved

In the above example, the package file gives a *name* to the package and defines the *dependencies*.

The dependencies property is a list of key-value pairs. The key provides the name of the ECL module that is used to access the external repository. The value is a repository reference using the same syntax described in the previous section.

External Repositories in ECL

To use definitions from an external repository you would add an import definition to your ECL code.

For example consider the following ECL:

```
IMPORT layout;
IMPORT demoRepoD AS demoD;
EXPORT personAsText(layout.person input) :=
 input.name + ': ' + demoD.format.maskPassword(input.password);
```

The name demoRepoD in the second import matches the key value in the package.json file.

This code uses the attribute *format.maskPassword* from the repository gituser/gch-ecldemo-d.

Each package is processed independently of any others. The only connection is through explicit imports of the external packages. This means that packages can have modules or attributes with the same name and they will not clash.

The NPM

You can use the NPM (node package manager) to ensure that labels or branches are tied down to a specific SHA.

For example :

```
npm install --package-lock-only
```

This command creates a *package-lock.json* file in the same location as package.json. The npm program then resolves the references to branches and resolves them to the corresponding SHAs. Using the --package-lock-only option indicates that the npm should not clone the associated versions of the code to node_modules directories.

The generated package-lock.json file will contain something similar to the following:

```
{
 "packages": {
   "node_modules/demoRepoD": {
     "resolved": "https://github.com/gituser/gch-ecldemo-d.git#644c1f4dd80ca1e8f05974983455a244e5",
   }
  }
}
```

If a package-lock.json file is present it will take precedence over the package.json file. The ECL is compiled in the same way, with ECLCC Server automatically downloading the dependencies.

The advantage of using package-lock.json over package.json is that it allows you to use npm's semantic versioning syntax (#semver). It also allows you to use a branch in your package.json file as a logical dependency, but resolved to an actual dependency or specific SHA. That way if the branch is updated the query will not change.

You can also use npm without the --package-lock-only option. This will check out the appropriate version of the code into the node_modules subdirectory of the current project. The ECLCC Server supports the

© 2026 HPCC Systems®. All rights reserved

node_module structure as a way of providing the source for external packages. This is an alternative way to compile the code using eclcc completely independently from the source control system.

© 2026 HPCC Systems®. All rights reserved

ecl deploy

ecl deploy <target> <file> [--job-name=<value>]

ecl deploy <target> <archive> [--job-name=<value>]

ecl deploy <target> <so | dll > [--job-name=<value>]

ecl deploy <target> - [--job-name=<val>]

Examples:

```
ecl deploy roxie findperson.ecl --job-name=FindPersonService
ecl deploy roxie ArchiveQuery.xml --job-name=FindPersonService
ecl deploy roxie libW20150914-125557.so --job-name=FindPersonService
ecl deploy roxie - --job-name=FindPersonService
```

A hyphen (-) specifies that the object should be read from stdin.

ecl deploy	Creates a workunit on the HPCC Systems platform from the given ECL text, file, archive, shared object, or dll. The workunit is created in the *compiled* state.
Arguments	
target	The target cluster to which to deploy
file	The ECL text file to deploy
archive	The ECL archive to deploy
so | dll	The workunit dynamic linked library or shared object to deploy
-	Specifies object should be read from stdin
Options	
--job-name	The published query name
--protect	Protect the workunit from deletion
ECL Options	
--main=<definition>	This option allows you to compile queries from existing source trees or repositories. It can be used to specify default repositories and versions to use. See the "Git Support" section for specific usage.
--snapshot, -sn=<label>	Snapshot label to use from a legacy ECL repository
--ecl-only	Send ECL query to HPCC Systems cluster as text rather than gathering dependencies and building an archive
--limit=<limit>	Sets the result limit for the query
-f<option>[=value]	Set an ECL option (equivalent to #option)
-f-<option>[=value]	Set an eclcc command line option with the single dash ('-')
-f--<option>[=value]	Set an eclcc command line option with the double dash ('--')
-Dname=value	Override the definition of a global attribute 'name'
eclcc Options	
-I <path>	Add path to locations to search for ecl imports
-L <path>	Add path to locations to search for system libraries
--manifest	Specify path to the manifest file
-g, --debug	Enable debug symbols in generated code

© 2026 HPCC Systems®. All rights reserved

-checkDirty	This option reports any modified attributes using git status
--fetchrepos	This option automatically downloads any missing repositories associated with dependencies
-R<repo>[#version]=<path>	This option resolves repository references specified in directory <path>
--updaterepos	Setting this option automatically updates repositories associated with dependencies
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server=<ip>	The IP Address or hostname of ESP server running ECL services
--port=<port>	The ECL services port (Default is 8010)
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username=<name>	The username for accessing ECL services
-pw, --password	The password for accessing ECL services. If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl publish

ecl publish <target> <file> [--query-name=<val>]

ecl publish <target> <file> [--query-name=<val>] [--job-name=<val>]

ecl publish <target> <wuid> [--query-name=<val>]

ecl publish <target> <so | dll> [--query-name=<val>]

ecl publish <target> <archive> [--query-name=<val>]

ecl publish <target> - [--query-name=<val>]

Examples:

```
ecl publish roxie findperson.ecl --query-name=FindPersonService -A
ecl publish roxie W20150914-125557 --query-name=FindPersonService -A
ecl publish roxie libW20150914-125557.so --query-name=FindPersonService -A
ecl publish roxie ArchiveQuery.xml --query-name=FindPersonService -A
ecl publish roxie - --query-name=FindPersonService --activate
ecl publish roxie findperson.ecl --query-name=FindPersonService --no-activate
ecl publish roxie ArchiveQuery.xml --query-name=FindPersonService --no-activate
```

A hyphen (-) specifies that the object should be read from stdin.

ecl publish	Publishes a query into a queryset. The query is created by adding a workunit to a queryset and assigning it a query name.
Arguments	
target	The target cluster to which to publish
wuid	The workunit id to publish (WUID is case-sensitive)
file	The ECL text file to publish
archive	The ECL archive to publish
so | dll	The workunit dynamic linked library or shared object to publish
-	Specifies object should be read from stdin
Options	
--query-name	The query name to use for published workunit
--job-name	The job name to use for the workunit
-A, --activate	Activates query when published (default)
-A-, --no-activate	Does not activate query when published
-sp, --suspend-prev	Suspend previously active query
-dp, --delete-prev	Delete previously active query
--protect	Protects the workunit specified from deletion.
--no-files	Specifies to not copy DFS file information for files referenced by the query
--no-reload	Specifies to not request a reload of the Roxie cluster
--allow-foreign	Specifies to allow the use of foreign files in a Roxie query. If a Roxie query references foreign files and this is not enabled, publish will fail.
--daliip=<IP>	IP address or hostname of the remote Dali to use for remote logical file lookups.

© 2026 HPCC Systems®. All rights reserved

-O, --overwrite	Completely replaces the existing DFS file information. This option is dangerous - only use as a last resort.
--update-super-files	Update local DFS super-files if remote DALI has changed
--update-clone-from	Update local clone from location if remote DALI has changed
--dont-append-cluster	Issue this option only to avoid locking issues due to adding a cluster to the file
---source-process	Process cluster from which to copy files
--timeLimit=<sec>	Value to set for query timeLimit configuration
--warnTimeLimit=<sec>	Value to set for query warnTimeLimit configuration
--memoryLimit=<mem>	Value to set for query memoryLimit configuration. Format <mem> as 500000B, 550K, 100M, 10G, or 1T, etc.
--priority=<val>	The priority for this query. Value can be LOW, HIGH, SLA, NONE. NONE will clear current setting.
--comment=<string>	A comment associated with this query
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
ECL Options	
--main=<definition>	This option allows you to compile queries from existing source trees or repositories. It can be used to specify default repositories and versions to use. See the "Git Support" section for specific usage.
--snapshot, -sn=<label>	Snapshot label to use from a legacy ECL repository
--ecl-only	Send ECL query to HPCC Systems cluster as text rather than as a generated archive
--limit=<limit>	Sets the result limit for the query
-f<option>[=value]	Set an ECL option (equivalent to #option)
-f-<option>[=value]	Set an eclcc command line option with the single dash ('-')
-f--<option>[=value]	Set an eclcc command line option with the double dash ('--')
-Dname=value	Override the definition of a global attribute 'name'
eclcc Options	
-I <path>	Add path to locations to search for ecl imports

© 2026 HPCC Systems®. All rights reserved

-L <path>	Add path to locations to search for system libraries
--manifest	Specify path to the manifest file
-g, --debug	Enable debug symbols in generated code
-checkDirty	This option reports any modified attributes using git status
--fetchrepos	This option automatically downloads any missing repositories associated with dependencies
-R<repo>[#version]=<path>	This option resolves repository references specified in directory <path>
--updaterepos	Setting this option automatically updates repositories associated with dependencies
--nostdinc	Do not include the current directory in -I
--fastsyntax	Delay expanding functions when parsing. May speed up processing for some queries
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username=<name>	The username for accessing ECL services
-pw, --password	The password for accessing ECL services. If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl unpublish

ecl unpublish <target> <query_id>

Example:

```
ecl unpublish roxie FindpersonService.1
ecl unpublish roxie "FindpersonService*"
```

ecl unpublish	executes the supplied ecl unpublish command
Options	
<target>	The name of target queryset containing the query to remove
<query_id>	The query to remove from query set. Wildcards allowed, but must be in quotes (e.g., "MyQuery*").
--activated=yes|no	Specify yes to unpublish activated queries or no to unpublish deactivated queries. When the <query_id> is '*'. Default to all queries.
--suspended-by-user	Unpublish the queries suspended by user when <query_id> is '*'.
--delete-workunit	Delete the workunit for the <query_id>. Default to No.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl run

ecl run <target> <file> [--job-name=<val>] [--input=<file|xml>] [--wait=<i>]

ecl run <target> <wuid> [--input=<file|xml>] [--wait=<ms>]

ecl run <target> <query> [--input=<file|xml>][--wait=<ms>]

ecl run <target> <so | dll> [--job-name=<val>][--input=<file|xml>][--wait=<i>]

ecl run <target> <archive> --job-name=<val> [--input=<file|xml>][--wait=<i>]

ecl run <target> - --job-name=<val> [--input=<file|xml>][--wait=<i>]

Examples:

```
ecl run thor findperson.ecl --job-name=findperson --input=data.xml --wait=1000
ecl run thor W20150914-125557 --input=data.xml --wait=1000
ecl run thor findperson --input=data.xml --wait=1000
ecl run thor libW20150914-125557.so --input=data.xml --wait=1000
ecl run thor - --input=data.xml --poll --wait=1000
ecl run thor findperson.ecl --input="<request><LName>JONES</LName></request>"
ecl run thor findperson.ecl -I C:\MyECL\
```

A hyphen (-) specifies that the object should be read from stdin.

ecl run	executes the supplied ecl run command
Arguments	
target	The target cluster to which to publish
wuid	The workunit id to run (WUID is case-sensitive)
file	The ECL text file to run
archive	The ECL archive to run
so | dll	The workunit dynamic linked library or shared object to run
-	Specifies object should be read from stdin
Options	
--job-name	The job name to use for the workunit
-in,--input=<file|xml>	The file or xml content to use as query input
-X<name>=<value>	Sets the stored input value (stored('name'))
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
--poll	Submits a job asynchronously and polls the server until the state of the workunit changes to completed. It then retrieves the results. Combine with the --wait option to limit the time that it polls.
--exception-level	Sets the minimum severity for reporting exceptions. Possible severity levels are **info**, **warning**, or **error**. The default is info which returns all exceptions.
--protect	This option prevents the workunit from deletion
ECL Options	
--main=<definition>	This option allows you to compile queries from existing source trees or repositories. It can be used to specify default repositories and versions to use. See the "Git Support" section for specific usage.

© 2026 HPCC Systems®. All rights reserved

--snapshot, -sn=<label>	Snapshot label to use from a legacy ECL repository
--ecl-only	Send ECL query to HPCC Systems cluster as text rather than as a generated archive
--limit=<limit>	Sets the result limit for the query
-f<option>[=value]	Set an ECL option (equivalent to #option)
-f-<option>[=value]	Set an eclcc command line option with the single dash ('-')
-f--<option>[=value]	Set an eclcc command line option with the double dash ('--')
-Dname=value	Override the definition of a global attribute 'name'
eclcc Options	
-I <path>	Add path to locations to search for ecl imports
-L <path>	Add path to locations to search for system libraries
--manifest	Specify path to the manifest file
-g, --debug	Enable debug symbols in generated code
-checkDirty	This option reports any modified attributes using git status
--fetchrepos	This option automatically downloads any missing repositories associated with dependencies
-R<repo>[#version]=<path>	This option resolves repository references specified in directory <path>
--updaterepos	Setting this option automatically updates repositories associated with dependencies
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl results

ecl results <wuid> [--noroot] [--exception-level=<value>]

Examples:

```
ecl results W20170519-142920
ecl results W20170519-142920 --noroot --exception-level=error
```

ecl results	returns the full results of a given WUID in XML format.
Arguments	
wuid	The workunit from which to return results. (WUID is case-sensitive)
Options	
--noroot	Suppresses the <Result> root tag in the XML returned.
--exception-level	Sets the minimum severity for reporting exceptions. Possible severity levels are **info**, **warning**, or **error**. The default is info which returns all exceptions.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl activate

ecl activate <queryset> <query_id>

Example:

```
ecl activate roxie FindpersonService.4
```

ecl activate	Activates a published query. This assigns a query to the active alias with the same name as the query.
Arguments	
target	The name of target queryset containing query to activate
query_id	The query to activate
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl deactivate

ecl deactivate <queryset> <active_alias>

Example:

```
ecl deactivate roxie FindpersonService
```

ecl deactivate	Deactivates a published query by removing an active query alias from the given queryset.
Arguments	
target	The name of the target queryset containing the alias to deactivate
active_alias	The active alias to be removed from the queryset
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries list

ecl queries list [<queryset>][--target=<cluster>][--show=<flags>]

Examples:

```
ecl queries list roxie
ecl queries list roxie --target=roxie --show=A
```

ecl queries list	Displays a list of the queries in one or more querysets. If a cluster is provided the querysets associated with that cluster will be shown. If no queryset or cluster is specified all querysets are shown.
Actions	
list	List queries in queryset(s)
Options	
<target>	The name of queryset from which to list queries
-t, --target	The name of the target cluster associated with the queries to list
--inactivate	Show only the queries that do not have an active alias
--show=<flags>	Show only queries with matching flags
Flags	
A	Active
S	Suspended
U	No Flags
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries files

ecl queries files <target> [<query>]

The queries files command displays a list of the files currently in use by the given query. If the *query* option is omitted, it returns a list of files for all queries on the specified target.

Examples:

```
ecl queries files roxie myquery
```

Example result:

```
 > ecl queries files roxie myquery
 ------------------
Query: myquery
Files used:
  jd::subfile1, 100 bytes, 2 part(s)
  jd::subfile2, 100 bytes, 2 part(s)

SuperFiles used:
  jd::mysuperfile
  > jd::subfile2
  > jd::subfile1
```

Options	
target	Name of target cluster on which the query is published
query	Optional. Name of the query for which to get a list of files it uses. If omitted, a list of files for each query on the specified target is returned.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries copy

ecl queries copy <source_query_path> <target_queryset> [--activate]

Examples:

```
ecl queries copy thor/findperson thor2 --activate
ecl queries copy //192.168.1.10:8010/thor/findperson thor
```

ecl queries copy	Copies a query from one queryset to another. A query can be copied from one HPCC Systems environment to another by using a path which begins with '//' followed by the IP or hostname and Port of the source ECL Watch and then followed by the source queryset and query.
Actions	
copy	Copy a query from one queryset to another
Options	
source_query_path	The path of the query to copy using the format: [//ip:port/]queryset/query or queryset/query.
<target>	The name of the target cluster to copy the query to
--source-ssl	Use SSL when connecting to the source (default if --ssl is enabled)
--source-no-ssl	Do not use SSL when connecting to source (default if --ssl is NOT enabled)
--no-files	Specifies to not copy DFS file information for files referenced by the query
--daliip=<IP>	IP address or hostname of the remote Dali DFS to use for copying file information. Only required when remote environment is less than version 3.8
--source-process	Process cluster to copy files from
-A, --activate	Activates the query when copied
-sp, --suspend-prev	Suspend previously active query
-dp, --delete-prev	Delete previously active query
--no-reload	Specifies to not request a reload of the Roxie cluster
-O, --overwrite	Whether to overwrite existing information - true if present
--update-super-files	Update local DFS super-files if remote DALI has changed
--update-clone-from	Update local clone from location if remote DALI has changed
--dont-append-cluster	Only use to avoid locking issues due to adding cluster to file
--allow-foreign	Specifies to allow the use of foreign files in a Roxie query. If a Roxie query references foreign files and this is not enabled, copy will fail. Prevents failure if using foreign files in Roxie query.
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
--timeLimit=<sec>	Value to set for query timeLimit configuration
--warnTimeLimit=<sec>	Value to set for query warnTimeLimit configuration
--memoryLimit=<mem>	Value to set for query memoryLimit configuration. Format <mem> as 500000B, 550K, 100M, 10G, or 1T, etc.
--priority=<val>	Set the priority for the query. Values can be LOW, HIGH, SLA, NONE. NONE will clear the current setting.

© 2026 HPCC Systems®. All rights reserved

--comment=<string>	Set a comment associated with the query
-n, --name=<value>	The destination name for the copied query
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries copy-set

ecl queries copy-set <source_target> <destination_target> [--all] [--clone-active-state]

Examples:

```
ecl queries copy-set roxie1 roxie2
ecl queries copy-set roxie1 roxie2 --all
ecl queries copy-set roxie1 roxie2 --clone-active-state
```

ecl queries copy-set	Copies a set of queries from one target to another.
Actions	
copy-set	Copy a set of queries from one target to another.
Options	
source_target	Target cluster from which to copy queries.
destination_target	Target cluster to copy queries to.
--source-ssl	Use SSL when connecting to the source (default if --ssl is enabled)
--source-no-ssl	Do not use SSL when connecting to source (default if --ssl is NOT enabled)
--all	Specifies to copy both active and inactive queries. If omitted, only active are copied.
--no-files	Specifies to not copy DFS file information for files referenced by the query
--daliip=	IP address or hostname of the remote Dali to use for logical file lookups.
--source-process	Process cluster from which to copy files.
--clone-active-state	Make copied queries active on target if they are active on the source.
-O, --overwrite	Whether to overwrite existing DFS information - true if present
--update-super-files	Update local DFS super-files if remote DALI has changed
--update-clone-from	Update local clone from location if remote DALI has changed
--dont-append-cluster	Only use to avoid locking issues due to adding cluster to file
--allow-foreign	Specifies to allow the use of foreign files in a Roxie query. If a Roxie query references foreign files and this is not enabled, copy will fail.
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	

© 2026 HPCC Systems®. All rights reserved

--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries config

ecl queries config <target> <queryid> [options]

Examples:

```
ecl queries config thor findperson --wait=1000
```

ecl queries config	Updates query configuration values
Actions	
config	Set or update query configuration values
Options	
target	The name of the target queryset
queryid	The name of the query
--no-reload	Specifies to not request a reload of the Roxie cluster
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
--timeLimit=<sec>	Value to set for query timeLimit configuration
--warnTimeLimit=<sec>	Value to set for query warnTimeLimit configuration
--memoryLimit=<mem>	Value to set for query memoryLimit configuration. Format <mem> as 500000B, 550K, 100M, 10G, or 1T, etc.
--priority=<val>	Set the priority for the query. Values can be LOW, HIGH, SLA, NONE. NONE will clear the current setting.
--comment=<string>	Set a comment associated with the query
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries recreate

ecl queries recreate <target> <query> [<destination-target>] [options]

Examples:

```
ecl queries recreate roxie findpeople
ecl queries recreate roxie findpeople roxie2
```

ecl queries recreate	Recompiles a query into a new workunit and republishes the new workunit. This is useful when upgrading to a new ECL compiler and you want to recompile a query from the exact same source. The ECL archive must be available within the workunit of the query.
Actions	
recreate	Recompiles a query into a new workunit and republishes the new workunit.
Arguments	
target	The target the query you wish to recreate is in
query	The query ID of the query you wish to recreate
destination-target	Optional: The target you want to move the new query to (if different from the source target)
Options	
-A, --activate	Activates query when published
--limit=<limit>	Sets the result limit for the query, defaults to 100
-sp, --suspend-prev	Suspends previously active query
-dp, --delete-prev	Deletes previously active query
-A-, --no-activate	Does not activate query when published
--no-publish	Creates a recompiled workunit, but does not publish it
--no-reload	Specifies to not request a reload of the Roxie cluster
--no-files	Specifies to not copy DFS file information for files referenced by the query
--allow-foreign	Specifies to allow the use of foreign files in a Roxie query. If a Roxie query references foreign files and this is not enabled, publish will fail.
--daliip=<IP>	IP address or hostname of the remote Dali to use for remote logical file lookups.
--update-super-files	Update local DFS superfiles if remote DALI has changed
--update-clone-from	Update local clone from location if remote DALI has changed
--dont-append-cluster	Use only to avoid locking issues due to adding cluster to file
--source-process=<value>	Process cluster to copy files from
--timeLimit=<sec>	Value to set for query timeLimit configuration
--warnTimeLimit=<sec>	Value to set for query warnTimeLimit configuration
--memoryLimit=<mem>	Value to set for query memoryLimit configuration. Format <mem> as 500000B, 550K, 100M, 10G, or 1T, etc.
--priority=<val>	The priority for this query. Value can be LOW, HIGH, SLA, NONE. NONE will clear current setting.
--comment=<string>	A comment associated with this query

© 2026 HPCC Systems®. All rights reserved

--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries import

ecl queries import <target> <file> [--clone-active-state] [--replace] [options]

Example:

```
ecl queries import roxie1 myqueryset.xml
```

ecl queries import	Imports the contents of a queryset previously exported to disk
Actions	
import	Imports a queryset from a file
Arguments	
target	The target cluster to import queries to
Options	
--all	Copy both active and inactive queries. If omitted, only active queries are imported.
--replace	Replace entire existing queryset
--queries	Filter query ids to select for import
--no-files	Specifies to not copy DFS file information for files referenced by the query
--daliip=<IP>	Remote Dali DFS to use for copying file information
--source-process=<value>	Process cluster to copy files from
--clone-active-state	Make copied queries active if active on source
-O, --overwrite	Completely replace existing DFS file information (dangerous)
--update-super-files	Update local DFS superfiles if remote DALI has changed
--update-clone-from	Update local clone from location if remote DALI has changed
--dont-append-cluster	Use only to avoid locking issues due to adding cluster to file
--allow-foreign	Specifies to allow the use of foreign files in a Roxie query. If a Roxie query references foreign files and this is not enabled, publish will fail.
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	
--help	Displays usage information for the given command

© 2026 HPCC Systems®. All rights reserved

-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl queries export

ecl queries export <target> [options]

Example:

```
ecl queries export roxie1 --output=myqueryset.xml
```

ecl queries export	saves backup information about a given queryset to a file.
Actions	
export	Exports queryset information to a file
Arguments	
target	Name of target cluster to export from
-O, --output=<filename>	Filename to save exported backup information to (optional)
Options	
--active-only	Only include active queries in the exported queryset.
--protect	Protect the workunits for the included queries
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap add

ecl packagemap add [--daliip][options] <target> <filename>

Examples:

```
ecl packagemap add  -s=192.168.1.10 roxie mypackagemap.pkg
ecl packagemap add roxie mypackagemap.pkg --overwrite
ecl packagemap add roxie mypackagemap.pkg --daliip=192.168.11.11
```

ecl packagemap add	Calls the packagemap add command
Actions	
add	Adds a package map to the target cluster
Arguments	
target	The target to associate the package map with
filename	The name of the file containing package map information.
--daliip=	IP address or hostname of the remote Dali to use for logical file lookups
Options	
-O, --overwrite	Whether to overwrite existing information - true if present
-A, --activate	Activates package map
--allow-foreign	Specifies to allow the use of foreign files. If a package map references foreign files and this is not enabled, package map add will fail.
--pmid=<packagemapid>	id of package map - defaults to filename if not specified
--global-scope	The packagemap specific with this option can be shared across multiple targets
--preload-all	Sets the preload files option for all packages
--replace	Replace the existing packagemap
--update-superfiles	Updates the local DFS super-files if the remote DALI has changed
--update-clone-from	Update the local clone from location if the remote DALI has changed
--dont-append-cluster	Only use to prevent locking issues due to adding cluster to file
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	

© 2026 HPCC Systems®. All rights reserved

--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap delete

ecl packagemap delete [options] <target><packagemap>

Examples:

```
ecl packagemap delete roxie mypackagemap
```

ecl packagemap delete	Calls the packagemap delete command
Actions	
delete	Deletes a package map
Options	
<target>	The name of the target to use when deleting the packagemap
<filename>	The name of the packagemap to delete
--global-scope	Use when the specified packagemap is shared across multiple targets
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap activate

ecl packagemap activate <target> <packagemap>

Example:

```
ecl packagemap activate roxie mypackagemap.pkg
```

ecl packagemap activate	The activate command will deactivate the currently active package map and make the specified package map active.
Arguments	
target	The target containing the package map to activate
packagemap	name of package map to update
--global-scope	The specified packagemap can shared across multiple targets
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap deactivate

ecl packagemap deactivate <target> <packagemap>

Example:

```
ecl packagemap deactivate roxie mypackagemap.pkg
```

ecl packagemap deactivate	The deactivate command will deactivate the currently active package map.
Arguments	
target	The target containing the package map to deactivate
packagemap	Name of package map to deactivate
--global-scope	The specified packagemap can shared across multiple targets
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap list

ecl packagemap list <target>

Examples:

```
ecl packagemap list roxie
```

ecl packagemap list	Calls the packagemap list command
Actions	
list	Lists loaded package map names
Arguments	
target	The target containing the package map to list
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap info

ecl packagemap info [options] <target>

Examples:

```
ecl packagemap info roxie
```

ecl packagemap info	Calls the packagemap info command
Actions	
info	returns package map info
Arguments	
target	The target containing the package map to retrieve
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap add-part

ecl packagemap add-part <target> <pmid> <filename>

Examples:

```
ecl packagemap add-part roxie multipart.pkg addresses.pkg
```

The packagemap add-part command adds additional package map content to an existing package map

ecl packagemap add-part	Calls the packagemap add-part command.
Actions	
add-part	Adds additional package map content to an existing package map
Arguments	
target	Name of target to use when adding package map part
pmid	Identifier of package map to add the part to
filename	one or more part files
Options	
--part-name	Name of part being added (defaults to filename)
--delete-prev	Replace an existing part with matching name
--daliip=<ip>	IP of the remote Dali to use for logical file lookups
--global-scope	The specified package map is shared across multiple targets
--source-process=<value>	Process cluster to copy files from
--allow-foreign	Do not fail if foreign files are used in package map
--preload-all	Set preload files option for all packages
--update-super-files	Update local DFS superfiles if remote DALI has changed
--update-clone-from	Update local clone from location if remote DALI has changed
--dont-append-cluster	Use only to avoid locking issues due to adding cluster to file
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	
--help	Displays usage information for the given command

© 2026 HPCC Systems®. All rights reserved

-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap get-part

ecl packagemap get-part <target> <packagemap> <partname>

Examples:

```
ecl packagemap get-part roxie multipart.pkg contacts
```

The get-part command fetches the given part from the given package map

ecl packagemap get-part	Calls the packagemap get-part command.
Actions	
get-part	Fetches the given part from the given package map
Arguments	
target	Name of target to use when adding package map part
packagemap	Name of the package map containing the part
partname	Name of the part to retrieve
Options	
--global-scope	The specified package map is sharable across multiple targets
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap remove-part

ecl packagemap remove-part <target> <pmid> <partname>

Examples:

```
ecl packagemap remove-part roxie multipart.pkg contacts
```

The remove-part command will remove the given part from the given package map

ecl packagemap remove-part	Calls the packagemap remove-part command.
Actions	
remove-part	Removes the given part from the given package map
Arguments	
target	Name of target to use
packagemap	Name of the package map containing the part
partname	Name of the part to remove
Options	
--global-scope	The specified package map is sharable across multiple targets
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap validate

ecl packagemap validate <target> [<filename>]

Examples:

```
ecl packagemap validate roxie mypackagemap.pkg
ecl packagemap validate roxie --active
ecl packagemap validate roxie mypackagemap.pkg --ignore-queries='findPerson*, findByZip'
```

The packagemap validate command verifies that :

- Referenced superkeys have subfiles defined (warns if no subfiles exist)
- All referenced queries exist in the current Roxie queryset
- All Roxie queries are defined in the package

The result will also list any files that are used by queries but not mapped in the package map.

Filename, --active, and --pmid are mutually exclusive. The --active or --pmid options validate a package map that has already been added instead of a local file.

The --queryid option checks the files in a query instead of all the queries in the target queryset. This is quicker when you only need to validate the files for a single query.

ecl packagemap validate	Calls the packagemap validate command.
Actions	
validate	Validates package map info
Arguments	
filename	The filename containing the package map info to validate
target	The target containing the package map to validate
--active	Validates the package map that is active for the given target
--check-dfs	This option verifies that subfiles exist in the DFS
-pm, --pmid	Validates the specified packagemap
--queryid	Validate the files for the given queryid if they are mapped in the package map
--ignore-optional	Will not report optional files that are not defined in the packagemap
--ignore-warnings	Will not report general packagemap warnings
--ignore-queries=<setOfQueries>	Will not report on the queries which match the expression. The set of queries can be a comma-separated list or you can use the option once per entry. Wildcards are supported.
--global-scope	The specified package map is sharable across multiple targets
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.

© 2026 HPCC Systems®. All rights reserved

--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl packagemap copy

ecl packagemap copy <path> <target>

Copies a package map from one target to another.

Examples:

```
ecl packagemap copy roxie/MyPkg roxie2
ecl packagemap copy //192.168.0.100:8010/roxie/MyPkg roxie2
```

ecl packagemap copy	Calls the packagemap copy command
Actions	
copy	Copies a package map from one target to another
Arguments	
path	Path to the source package map to copy.
target	The target to copy the package map to
Options	
-A, --activate	Activates package map
--daliip=	IP address or hostname of the remote Dali to use for logical file lookups
--pmid=<packagemapid>	id of package map - defaults to filename if not specified
--source-process	Process cluster to copy files from
--preload-all	Set preload files option for all packages
--replace	Replace existing packagmap
--update-super-files	Update local DFS superfiles if remote Dali has changed
--update-clone-from	Update local clone from location if remote Dali has changed
--dont-append-cluster	Only use to avoid locking issues due to adding cluster to file
DFU Options	
--dfu-copy	Use DFU to copy files during deployment, not on Roxie in the background
--init-publisher-wuid	Preallocates the publisher workunit at the beginning of the command and displays the WUID on the command line
--dfu-queue	The DFU Queue to use when doing a DFU copy
--dfu-wait	The amount of time in seconds to wait for the DFU copy to complete (if neither --only-copy-files or --stop-if-files-copied are specified). Default is 1800 (30 minutes)
--dfu-overwrite	Specifies that DFU copy command should overwrite physical files that are already on disk.
--only-copy-files	Copy the files needed for the query, but don't publish the query
--stop-if-files-copied	If all files already exist, publish the query. Otherwise, copy the files needed for the query, but don't publish the query
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services

© 2026 HPCC Systems®. All rights reserved

-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

For the path, the following formats are supported:

- remote packagemap: //IP:PORT/Target/PackageMapId
- local packagemap: target/PackageMapId

© 2026 HPCC Systems®. All rights reserved

ecl roxie attach

ecl roxie attach <processName>

Examples:

```
ecl roxie attach myroxie
```

ecl roxie attach	Attach the roxie to Dali
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl roxie detach

ecl roxie detach <processName>

Examples:

```
ecl roxie detach myroxie
```

ecl roxie detach	Detach the roxie from Dali
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl roxie reload

ecl roxie reload <processName>

Examples:

```
ecl roxie reload myroxie
```

ecl roxie reload	Reloads the roxie info from Dali
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl roxie check

ecl roxie check <processName>

Examples:

```
ecl roxie check myroxie
```

ecl roxie check	Checks the state of the roxie process
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl roxie xref

ecl roxie xref <cluster>

The **roxie xref** command returns file information for the specified queries on the specified cluster. If the *queryids* option is omitted, file information about all queries is returned. The result is in XML format.

Examples:

```
ecl roxie xref myroxie
ecl roxie xref myroxie --queryids=myquery.1,myotherquery.1
```

Example result:

```
<QueryXrefInfo>
 <Endpoint ep="192.168.56.1:9876">
  <Queries reporting="1">
   <Query id="myquery.1"/>
     <SuperFile name="jd::mysuperfile">
       <File name="jd::subfile1"/>
       <File name="jd::subfile2"/>
   <Query id="myotherquery.1"/>
     <SuperFile name="jd::myothersuperfile">
       <File name="jd::subfile1"/>
       <File name="jd::subfile2"/>
       <File name="jd::subfile3"/>
  </Queries>
  <Status>ok</Status>
 </Endpoint>
</QueryXrefInfo>
```

ecl roxie xref	Returns file information for the selected queries in XML format.
Options	
--check-all-nodes	Gets query file information from all nodes. This can be slow.
--queryids=<csv list>	The queries for which to get file information (default is all queries)
--wait=<ms>	Maximum time (in milliseconds) to wait for cluster finish updating. Any value lower than the minimum of 1000 is ignored.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)

© 2026 HPCC Systems®. All rights reserved

--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl bundle depends

ecl bundle depends <bundleName> [--version <versionnumber>]

Examples:

```
ecl bundle depends mybundle
ecl bundle depends mybundle --version=2
```

ecl bundle depends	Shows the dependencies of a bundle
Options	
<bundleName>	The name of a bundle file or installed bundle
--recurse	Displays indirect dependencies
--version	Specify a version of the bundle
--help	Displays usage information about the given command
-v, --verbose	Output additional tracing information

© 2026 HPCC Systems®. All rights reserved

ecl bundle info

ecl bundle info <bundleName> [--version <versionnumber>]

Examples:

```
ecl bundle info mybundle
ecl bundle info https://github.com/hpcc-systems/ecl-bundles.git
ecl bundle info mybundle --version=2
```

ecl bundle info	Lists information about a bundle
Options	
<bundle>	A bundle filename, a bundle folder, a bundle name, or a URL.
--branch	Names a branch to install when bundle references a git repository
--remote	Interprets the bundle name as a remote name from catalog
--version	Specify a version of the bundle
--help	Displays usage information about the given command
-v, --verbose	Output additional tracing information

If a URL ends in .git, it is assumed to be a git repository (fetched using git clone) otherwise it is assumed to be the URL of a file that can be retrieved. In either case, it is fetched to a temporary local location, processed as a local file/directory and then removed.

© 2026 HPCC Systems®. All rights reserved

ecl bundle install

ecl bundle install <bundleName>

Examples:

```
ecl bundle install mybundle
ecl bundle install https://github.com/hpcc-systems/ecl-bundles.git
ecl bundle install mybundle --dryrun
ecl bundle install mybundle --update
ecl bundle install mybundle --keepprior
```

ecl bundle install	Installs a bundle
Options	
<bundleName>	The name or URL of a bundle file, folder, or installed bundle.
--dryrun	List what would be installed, but do not copy
--force	Install even if required dependencies missing
--keepprior	Do not remove any previous versions of the bundle
--remote	Interprets the bundle name as a remote name from catalog
--update	Update an existing installed bundle
--help	Displays usage information about the given command
-v, --verbose	Output additional tracing information

If a URL ends in .git, it is assumed to be a git repository (fetched using git clone) otherwise it is assumed to be the URL of a file that can be retrieved. In either case, it is fetched to a temporary local location, processed as a local file/directory and then removed.

To use the "ecl bundle install <git url>" command, you must have git installed and configured on your system. Git must be accessible to the user (in the path).

Server-side Installation

A system using ECL Server and a remote MySQL repository or ECLCC Server with git hooks can only access bundles if they are installed on the server. Using a bundle in the ECL Playground also requires installation on the ECLCC Server node.

1. In a terminal window on your ECLCC Server (or ECL Server), use this command:

   ```
   sudo su hpcc
   ```

   This switches the user to **hpcc**.

2. Next install the bundle using this command:

   ```
   ecl bundle install <bundle URL>.git
   ```

   For example:

   ```
   ecl bundle install https://github.com/hpcc-systems/Visualizer.git
   ```

3. Close the terminal window or use the **exit** command to return to acting as the original user.

© 2026 HPCC Systems®. All rights reserved

ecl bundle uninstall

ecl bundle uninstall <bundleName>

Examples:

```
ecl bundle uninstall mybundle
ecl bundle install mybundle --dryrun
ecl bundle install mybundle --update
ecl bundle install mybundle --keepprior
```

ecl bundle install	Installs a bundle
Options	
<bundleName>	The name of an installed bundle
--dryrun	List what would be removed, but do not remove them
--force	Uninstall even if other bundles are dependent on this
--version	Specify a version of the bundle
--help	Displays usage information about the given command
-v, --verbose	Output additional tracing information

© 2026 HPCC Systems®. All rights reserved

ecl bundle list

ecl bundle list <pattern>

Examples:

```
ecl bundle list
ecl bundle list myb*
```

ecl bundle list	Lists bundles matching specified pattern
Options	
<pattern>	A pattern specifying bundles to list. If omitted, all bundles are listed
--details	Report details of each installed bundle
--help	Displays usage information about the given command
-v, --verbose	Output additional tracing information

© 2026 HPCC Systems®. All rights reserved

ecl bundle use

ecl bundle use <bundleName> [--version <version>]

Example:

```
ecl bundle use myBundle --version 2
```

ecl bundle use	Makes a specified version of a bundle active
Options	
<bundle>	The name of a bundle file
--version	The version of the bundle to make active, or "none"
--help	Displays usage information about the given command
-v, --verbose	Output additional tracing information

© 2026 HPCC Systems®. All rights reserved

ecl roxie unused-files

ecl roxie unused-files <processName>

Examples:

```
ecl roxie unused-files myroxie
```

ecl roxie unused-files	Finds files in the DFS for the given roxie process that are not currently used by queries on that roxie.process_cluster
process_cluster	The roxie process cluster to check, or
--roxies <list>	A comma-separated list of roxies to check
--planes <list>	A comma-separated list of data planes to search for files
--check-packagemaps	Exclude files referenced in active package maps
--delete	Deletes unused files from DFS
--delete-subfiles	Deletes unused files from DFS and removes them from superfiles.
--delete-recursive	Deletes unused files from DFS and removes them from superfiles recursively.
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl abort

ecl abort -wu <WUID> | -n <jobName>

Examples:

```
ecl abort -wu W20150516-111213
ecl abort -n MyJob
```

ecl abort	aborts one or more Workunits from the given WUID or job name
Options	
-wu	The WUID (Workunit ID) (WUID is case-sensitive)
-n	The job name
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl status

ecl status -wu <WUID> | -n <jobName>

Examples:

```
ecl status -wu W20150516-111213
ecl status -n MyJob
```

ecl status	returns the status of a given workunit or job name. If more than one is found, a CSV list returns.
Options	
-wu	The WUID (Workunit ID) (WUID is case-sensitive)
-n	The job name
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl getwuid

ecl getwuid -n <jobName> [--limit=<limitCount>]

Examples:

```
ecl getwuid -n MyJobName
ecl getwuid -n MyCommonJobName --limit=100
```

ecl getwuid	returns the WUID(s) for a given job name. If more than one is found, a list returns.
Options	
-n	The job name
--limit=*nn*	Integer to set result limit, default is 100
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl getname

ecl getname -wu <WUID>

Examples:

```
ecl getname -wu W20140516-111213
ecl getname -wu W201407*
```

ecl getname	returns the job name for a given workunit.
--wuid	The WUID (Workunit ID) (WUID is case-sensitive)
Options	
--limit=<limit>	This sets the result limit. This is useful when using wildcards in a request. (Default is 100)
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl zapgen

ecl zapgen <WUID> --path <zap_file_path> [options]

Use the zapgen command to create a Zipped Analysis Package (Z.A.P.) containing collecting system information about a workunit and encapsulating it into a shareable package. It is a useful tool for reporting errors, inconsistencies, or other unexpected behavior.

Examples:

```
ecl zapgen W20171017-091320 --path ~/reports
ecl zapgen W20171018-091399 --path ~ --inc-thor-slave-logs --description "Unexpected result from JOIN"
```

ecl zapgen	Generates and downloads a ZAP file for the given workunit in the specified path.
WUID	The WUID (Workunit ID) (WUID is case-sensitive)
--path	The path to store the ZAP file
Options	
--inc-thor-slave-logs	Includes Thor slave(s) log into the ZAP file
--description	Description of the issue
--create-dirs	Creates the necessary directory tree
Common Options	
--help	Displays usage information for the given command
-v, --verbose	Output additional tracing information
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
-ssl, --ssl	Use SSL to secure the connection to the server.
--accept-self-signed	Allows SSL servers to use self signed certificates
--cert	Path to file containing SSL client certificate
--key	Path to file containing SSL client certificate private key
--cacert	Path to file containing SSL CA certificate
--port	The ECL services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary). If you supply a username and do not supply a password, you will be promoted for a password.
--wait-connect=<Ms>	Timeout while connecting to server (in milliseconds)
--wait-read=<Secs>	Timeout while reading from socket (in seconds)
--noKeepAlive	Disable TCP keepalives with the server. If omitted, the default is FALSE, meaning connections to the server remain open.

© 2026 HPCC Systems®. All rights reserved

ecl url-secret-name

ecl url-secret-name url [*--username*=<username>]

ecl url-secret-name	Generates a secret name from a URL for automatic URL mapping.
url	The URL to convert into a secret name.
--username	Optional. The username to associate with the URL. This will override any username embedded in the URL.

The *ecl url-secret-name* command generates a secret name from a URL that can be used to support ECL SOAPCALL/HTTPCALL automated URL to Secret mapping.

A username can either be embedded in the URL, such as https://username@example.com, or passed in as a parameter using the --username=username option. If a username is passed in as a parameter it overrides a username in the URL.

Passwords embedded in the URL are not needed and will be ignored.

When ECL SOAPCALL URL secret mapping is enabled SOAPCALL will convert the URL provided into a name of this format. ECL will then attempt to lookup the secret, and if found, will use the contents of the secret, rather then the original URL.

Examples:

```
ecl url-secret-name https://example.com --username jimi
ecl url-secret-name http://example.com --username jimi
```

© 2026 HPCC Systems®. All rights reserved

ecl roxie memlock

ecl roxie memlock <process_cluster>

Options	
process_cluster	The Roxie process cluster to memlock

The *ecl roxie memlock* command locks Roxie heap memory. This helps to prevent intermittent slow queries by ensuring that the operating system does not swap the memory out.

Examples:

```
ecl roxie memlock mycluster
```

© 2026 HPCC Systems®. All rights reserved

ecl roxie memunlock

ecl roxie memunlock <process_cluster>

Options	
process_cluster	The Roxie process cluster to memunlock

The *ecl roxie memunlock* command unlocks Roxie heap memory.

Examples:

```
ecl roxie memunlock mycluster
```

© 2026 HPCC Systems®. All rights reserved

ecl roxie getmemlocked

ecl roxie getmemlocked <process_cluster>

Options	
process_cluster	The Roxie process cluster to getmemlocked status

The *ecl roxie getmemlocked* command retrieves the Roxie heap memory lock status.

Examples:

```
ecl roxie getmemlocked mycluster
```

© 2026 HPCC Systems®. All rights reserved

ECL Compiler

The ECL Compiler is the compiler component of the High Performance Computing Cluster (HPCC) Systems platform. It is embedded and included when you install the HPCC Systems platform. The compiler is the component that actually compiles the ECL code.

The syntax and many of the compiler options implemented are similar to the gcc compiler. You can execute either the Linux or Windows version of eclcc, which, when run, load several of our shared objects (SO files, on Linux) or DLLs (on Windows). The ECL Compiler can process hThor, Thor, or Roxie targeted ECL code.

To compile and run ECL code locally on your Windows machine, you will need the Microsoft Visual Studio 2008 C++ compiler (either Express or Professional edition). This is available from http://www.microsoft.com/express/Downloads/#2008-Visual-CPP

© 2026 HPCC Systems®. All rights reserved

Using the ECL Compiler as a Stand Alone option

The ECL Compiler is normally used through the ECL IDE, however, you can use the ECL Compiler in a stand alone manner, to create stand alone programs, or workunits. The ECL Compiler can read ECL code from standard input, or can read it from a specified input file. It compiles the code into an executable program (Such as an 'EXE' file in Windows). The resulting program, when executed, runs the job, writing any output to standard output. Alternatively, you could redirect the output to a file or pipe into another process. With the ECL Compiler, you do not need a supercomputer cluster to develop and run ECL code.

Running the ECL Compiler without any options (or specifying --help) will display the syntax.

```
C:\eclcc>eclcc -help
```

Usage: eclcc <options> ECL_file.ecl

General options:

-I <*path*>	Add path to locations to search for ecl imports
-L <*path*>	Add path to locations to search for system libraries
-o <*file*>	Specify name of output file (default a.out if linking to executable, or stdout)
-manifest	Specify path to manifest file listing resources to add
-foption[=value]	Set an ecl option. See #OPTION in the *ECL Language Reference* for details.
-main <*ref*>	Compile definition <ref> from the specified source - note extended Git repository support (see "Git Support" in the previous section)
-syntax	Perform a syntax check of the ECL
-platform=hthor	Generate code for hthor cluster
-platform=roxie	Generate code for roxie cluster (default)
-platform=thor	Generate code for thor cluster

NOTE: If there are spaces in the path you specify, put it in quotes. For example: -L"C:\Program Files"

Output control options:

-E	Output preprocessed ECL in xml archive form
-M	Output meta information for the ecl files
-Md	Output dependency information
-Me	eclcc should evaluate supplied ecl code rather than generating a workunit
-q	Save ECL query text as part of workunit
-qa	Save ECL query archive as part of workunit
-wu	Only generate workunit information as xml file

© 2026 HPCC Systems®. All rights reserved

C++ options:

-S	Generate C++ output, but don't compile
-c	Compile only (don't link)
-g, --debug	Enable debug symbols in generated code
-Wc,xx	Pass option xx to the C++ compiler
-Wl,xx	Pass option xx to the linker
-Wa,xx	Pass straight through to C++ compiler
-Wp,xx	Pass straight through to C++ compiler
-save-cpps	Do not delete generated C++ files (implied if -g)
-save-temps	Do not delete intermediate files
-shared	Generate workunit shared object instead of a stand-alone executable

File resolution options:

-dfs=ip	Use specified ip for DFS filename resolution
-scope=prefix	Use specified scope prefix in DFS filename resolution
-user=id	Use specified username in DFS filename resolution
-password=xxx	Use specified password in DFS filename resolution (blank to prompt)

© 2026 HPCC Systems®. All rights reserved

Other options (list is available using eclcc -help -v):

-aoption[=value]	Set an application option
--allow=str	Allow use of named feature. (e.g., cpp, pipe, all) **cpp**: Allow embedded code within ECL (e.g., C++, JAVA, Python, etc.) **pipe**: Allow the PIPE command to send data to an external program. **userECL**: Allow code that is not found via the ECL include paths **datafile**: Allow access to datafiles from ECL. **extern**: Allow access to an external service function **all**: Allow all features
-allowsigned	Only allows access to a feature from signed code
-fisComplexCompile	Prevents attempts to compile as a child process when a query is complex.
-b	Batch mode. Each source file is processed in turn. Output name depends on the input filename
-checkVersion	Enable/disable ecl version checking from archives
-checkDirty	Causes eclcc to generate a warning for any attribute that has been modified (according to the output of git status). Use of this function requires that git be installed and available on the path.
--component	Set the name of the component this is executing on behalf of
-D*name*=*value*	Override the definition of a global attribute 'name'
--deny=all	Disallow use of all named features not specifically allowed using --allow
--deny=str	Disallow use of named feature **cpp**: Disallow embedded code within ECL (e.g., C++, JAVA, Python, etc.) **pipe**: Disallow the PIPE command to send data to an external program.
--expand <path>	Expand the contents of an archive to the specified directory. The contents of the submitted query will output to stdout.
--fastsyntax	Delay expanding functions when parsing. May speed up processing for some queries
-help, --help	Display help message
--help -v	Display verbose help message
--internal	Run internal tests
--legacy	Use legacy import semantics (deprecated)

© 2026 HPCC Systems®. All rights reserved

Other options (continued):

--leakcheck	Clean up memory since checking for memory leaks
--keywords	Outputs the lists of ECL reserved words to stdout (XML format)
-legacyimport	Use legacy import semantics (deprecated)
-legacywhen	Use legacy when/side-effects semantics (deprecated)
--logfile <*file*>	Write log to specified file
--logdetail=*n*	Set the level of detail in the log file
--maxErrors=<n>	Limit the number of errors, aborting on the nth (default = 5)
--metacache=x	Specify directory to store distributed meta information from the eclcc indexer. To disable the indexer, set to an empty value using "--metacache=". If omitted, the default location is .eclcc/metacache.
--nologfile	Do not write any log file
--nogpg	Do not run gpg to check signatures on signed code
--nosourcepath	Compile as if the source came from stdin
--nostdinc	Do not include the current directory in -I
-pch	Generate precompiled header for eclinclude4.hpp
-P <path>	Specify the path of the output files (only with -b option)
-showpaths	Print information about the search paths eclcc is using
-specs <*file*>	Read eclcc configuration from specified file
-split *m:n*	Process a subset m of n input files (only with -b option)
-v --verbose	Output additional tracing information while compiling
-wxxxx=level	Set the severity for a particular warning code or category. Valid options for level are: all | ignore | log | warning | error | fail **-wall** sets default severity for all warnings
--version	Output version information
--timings	Output additional timing information

© 2026 HPCC Systems®. All rights reserved

Compiled Options:

After you have successfully compiled the code, it produces an executable file. There are a few additional options that can be used when running that executable.

Usage: a.out <options>

-wu=<file>	Write XML formatted workunit to given filespec and exit
-xml	Display output as XML
-raw	Display output as binary
-limit=x	Limit number of output rows
--help	Display help text

© 2026 HPCC Systems®. All rights reserved

Examples

The following example demonstrates what you can do once the ECL Compiler is installed and operational.

Running a basic ECL program using the command line compiler

Once the ECL Compiler is installed, you can use the ECL Compiler to run an ECL program.

- Create a file called hello.ecl, and type in the text

  ```
  Output('Hello world');
  ```

  (including the quotes) into the file.

  You can either use your favorite editor, or you can use the command line by typing the following (for Windows systems):

  ```
  echo Output('Hello world'); > hello.ecl
  ```

  on a Linux system you would need to escape some characters as follows:

  ```
  echo "Output('Hello world');" > hello.ecl
  ```

- Compile your program using the ECL Compiler by issuing the following command:

  ```
  eclcc hello.ecl
  ```

- An executable file is created which you can run by typing the following:

  on Linux systems:

  ```
     ./a.out
  ```

  on Windows systems:

  ```
     a.out
  ```

  This will generate the output "Hello world" (excluding quotes), to the std output, your terminal window in this example. You can redirect or pipe the output to a file or program if you choose. This simple example will verify the compiler is working properly.

Compile with Options

Once verified that the ECL Compiler is working correctly, you can try using some of the options. One such variation might be to specify the -o option which allows us to input more meaningful output filename of Hello.

```
eclcc -oHello hello.ecl
```

This produces a file called "Hello", which can now be run from the command line.

on Linux systems:

```
   ./Hello
```

on Windows systems:

© 2026 HPCC Systems®. All rights reserved

```
   Hello
```

This will result in the output of the following.

```
Hello world
```

There are additional options that can be used when running the executable. Using our Hello program, as an example, we can execute it with an option to generate different output. One such option is the -xml option which generates the output in an XML format.

on Linux systems:

```
   ./Hello -xml
```

on Windows systems:

```
   Hello -xml
```

This would result in the output of the following:

```
<Dataset name="Result 1"><Row><Result_1>Hello world</Result_1></Row></Dataset>
```

The following example provides a defined value passed to the compiler:

```
//file named hello2.ecl
IMPORT ^ as repo;
OUTPUT(repo.optionXX);
```

```
eclcc -Doptionxx='HELLO' hello2.ecl
```

This would result in the output of the following:

```
<Dataset name="Result 1"><Row><Result_1>HELLO</Result_1></Row></Dataset>
```

© 2026 HPCC Systems®. All rights reserved

Command Line DFU

Command Line Interface

dfuplus [--version] action=*operation* [@*filename*|*options*]

--version	displays version info
operation	One of the following actions: spray, despray, copy, remove, rename, list, add, addsuper, copysuper, removesuper, listsuper, savexml, status, abort, resubmit, monitor, listhistory, and erasehistory
@filename	Optional. The name of a file containing necessary *options*. If omitted and no command line *options* are specified, the appropriate *options* must be in the dfuplus.ini file in the same directory as the executable.
options	Optional. A space-delimited list of optional items (listed below) appropriate to the *operation* being executed. If omitted and no *@filename* is specified, the appropriate *options* must be in the dfuplus.ini file in the same directory as the executable.

The **dfuplus** executable accepts command line parameters to send to the Distributed File Utility (DFU) engine via the ESP server. These *options* can be specified on the command line, in the *@filename*, in the dfuplus.ini file in the same directory as the executable, or any combination.

Evaluation of options follows this order of precedence:

- command line
- @filename file
- ini file
- default value

The dfuplus utility does not upload files to a landing zone. You must first upload any file(s) to your landing zone using either ECL Watch or a tool that supports a secure copy protocol, such as SCP or SFTP.

General Options:

The following *options* are common to every *operation*:

server	The URL (http:// or https://) and/or IP address of the ESP server. The port may also be included.
username	A userid with authorized access to the *server*.
password	The password authorizing access for the *username*.
overwrite	Optional. A boolean flag (0 | 1) indicating whether to overwrite any existing file of the same name. If omitted, the default is 0.
replicate	Optional. A boolean flag (1 | 0) indicating whether to replicate the file. If omitted, the default is 1.

© 2026 HPCC Systems®. All rights reserved

	This option is only available on systems where replication has been enabled.
autorecover	Optional. The number of times to attempt recovery of a failed *operation*. If omitted, the default is 0.
nowait	Optional. A boolean flag (0 | 1) indicating whether to return immediately without waiting for completion of the *operation*. If omitted, the default is 0.
connect	Optional. The number of simultaneous connections to limit the *operation* to. If omitted, the default is 25.
nocommon	Optional. A boolean flag (0 | 1, default=1). Set to 0 to enable "commoning up" of pull or push processes on same host.
throttle	Optional. The transfer speed (in Mbits/second) to restrict the *operation* to. If omitted, the default is the best system speed in Linux and multiple-destination Windows, or the NIC speed of a single-destination Windows box.
norecover	Optional. A boolean flag (0 | 1) indicating whether to create or recover the *operation* from recovery information. If omitted, the default is 0.
nosplit	Optional. A boolean flag (0 | 1) indicating whether to split file parts to multiple target parts. If omitted, the default is 0.
compress	Optional. A boolean flag (0 | 1) indicating whether to compress the target file.
push	Optional. A boolean flag (0 | 1) indicating whether to override push/pull default.
encrypt=<password>	Optional. Specifies to encrypt the target filename using the supplied password.
decrypt=<password>	Optional. Specifies to decrypt the source filename using the supplied password.
jobname=<jobname>	Specify a jobname for the DFU operation's workunit.
transferbuffersize=nnn	Optional. Overrides the DFU Server's buffer size value (default is 64k)

© 2026 HPCC Systems®. All rights reserved

dfuplus.ini

Any *options* can be specified in a file called dfuplus.ini in the same directory as the executable. If your operating system is case-sensitive, make sure the filename is in lowercase. Options that rarely change can be put in the dfuplus.ini file. For example:

```
;The values below are examples, you should change them to match your platform deployment
server=http://10.150.50.12:8010
username=rlor

password=password
overwrite=1
replicate=1
```

In all the examples below, we'll assume dfuplus.ini has the above content.

We do not recommend storing your password in the ini file (which is clear text). The password is included in the ini file for these examples to simplify the example code.

Spray Operations:

The **spray** *operation* copies a file from the landing zone, distributing it across all the nodes of the destination HPCC Systems cluster.

These *options* are used by the **spray** *operation*:

srcip	Optional. The IP address of the source machine. If omitted, the information must be supplied by the *srcxml* or *srcplane* parameter.
srcplane	Optional. The source storage plane containing the source file. Note: *srcplane* should not be used at the same time as *srcip*.
srcfile	Optional. The path to the source file. This may contain wildcard characters (* and ?) to include multiple source files in the spray to a single *dstname*. If omitted, the information must be supplied by the *srcxml* parameter.
srcxml	The name of the XML file containing the information required for the *srcip* and *srcfile* parameters. This file may have been obtained by previous use of the savexml *operation*. This option provides the feature of combining multiple source files into a single resulting logical file in the HPCC Systems cluster.
dstname	The logical name of the destination file.
dstcluster	The name of the destination cluster. For a containerized deployment, set this to the target data plane (For example, dstcluster=data)
prefix	Optional. Both of the following (separated by a comma):
filename{:*length***}**	Prepends the filename (optionally limited to *length* characters) to the data.
filesize{:[B|L][1-8]}	Prepends the size of the file to the data. Optionally, you can specify the format of that integer (**B** specifies big endian, **L** specifies little endian) and the size of integer to contain it (**1** to **8** bytes). If format and size are omitted, the default is L4. When using wildcard characters (* and ?) to spray multiple source files (srcfile) to a single dstname, you MUST use both the filename and filesize options if you need to be able to despray the contents of each record in the dstname back to the multiple source files they originally came from. If you never need to do that, then the filesize option may be omitted.

© 2026 HPCC Systems®. All rights reserved

expireDays	Optional. Specifies the file is a temporary file to be automatically deleted after the specified number of days since the file was read. If omitted, the default is -1 (never expires). If set to 0, the file is automatically deleted when it reaches the threshold set in Sasha Server's **expiryDefault** setting.
format	Optional. One of the following values: **fixed csv delimited xml recfmv recfmb** If omitted, the default is fixed.
dstNumParts	Optional. Optional. Set to a positive integer to override the number of parts created when spraying. If omitted (or set to 0), the spray uses the target cluster's default part count.
fixed format options:	
recordsize	The fixed size of each record, in bytes.
csv/delimited options:	
encoding	Optional. One of the following: ascii, utf8, utf8n, utf16, utf16le, utf16be, utf32, utf32le, utf32be ; If omitted, the default is ascii.
maxrecordsize	Optional. The maximum size of each record, in bytes. If omitted, the default is 8192.
separator	Optional. The field delimiter. If omitted, the default is a comma (\,).
terminator	Optional. The record delimiter. If omitted, the default is line feed or carriage return line feed (\r,\r\n).
quote	Optional. The string quote character. If omitted, the default is single quote (').
xml format options:	
rowtag	The XML tag identifying each record. Required.
encoding	Optional. One of the following: utf8 utf8n utf16 utf16le utf16be utf32 utf32le utf32beIf omitted, the default is utf8.
maxrecordsize	Optional. The maximum size of each record, in bytes. If omitted, the default is 8192.

Examples:

```
//fixed spray example:
dfuplus action=spray srcip=10.150.50.14
          srcfile=/var/lib/HPCCSystems/mydropzone/timezones.txt dstname=RTTEMP::timezones.txt
          dstcluster=mythor format=fixed recordsize=155

//fixed spray example using a srcxml file:
dfuplus action=spray srcxml=/var/lib/HPCCSystems/mydropzone/flattimezones.xml
          dstname=RTTEMP::timezones.txt dstcluster=mythor recordsize=155

//csv spray example:
dfuplus action=spray srcip=10.150.50.14
          srcfile=/var/lib/HPCCSystems/mydropzone/timezones.csv dstname=RTTEMP::timezones.csv
          dstcluster=mythor format=csv

//the spray.xml file contains:
<File directory="/var/lib/HPCCSystems/mydropzone/"
   group="thor"
   modified="2004-04-27T14:58:38"
   name="zip"
   numparts="2"
   partmask="zip._$P$_of_$N$">
<Attr job="zip1"
   owner="rtaylor"
   recordSize="5"
```

© 2026 HPCC Systems®. All rights reserved

```
   replicated="1"
   workunit="D20040427-111857"/>
<Part modified="2004-04-27T14:58:40"
   node="10.150.51.29"
   num="1"
   size="165"/>
<Part modified="2004-04-27T14:58:40"
   node="10.150.51.29"
   num="2"
   size="165"/>
</File>

//fixed spray example using the above spray.xml file to combine
// multiple source files into a single logical file
// in this case, zip._1_of_3, zip._2_of_3, and zip._3_of_3 into zip1:
dfuplus action=spray srcxml=spray.xml
          dstcluster=mythor dstname=RTTEMP::myzip1 recordsize=5

//xml spray example:
dfuplus action=spray srcip=10.150.50.14
          srcfile=/var/lib/HPCCSystems/mydropzone/timezones.xml dstname=RTTEMP::timezones.xml
          dstcluster=mythor format=xml rowtag=area

//Multiple spray all .JPG and .BMP files under
// /var/lib/HPCCSystems/mydropzone/ on 10.150.51.26 to single logical file LE::imagedb

dfuplus action=spray srcip=10.150.51.26
          srcfile=/var/lib/HPCCSystems/mydropzone/*.jpg,/var/lib/HPCCSystems/mydropzone/*.bmp
          dstcluster=mythor
          dstname=LE::imagedb
          overwrite=1
          prefix=FILENAME,FILESIZE nosplit=1
//this would result in a RECORD structure like this:
imageRecord := RECORD
STRING filename;
DATA image; //first 4 bytes contain the length of the image data
END;
// using srcplane
dfuplus action=spray srcplane=lzstorageplane
        srcfile=/var/lib/HPCCSystems/dropzone/largedata1
        dstname=mytest::test:spraytest
        dstcluster=mydatastorageplane recordsize=58 overwrite=1 server=127.0.0.1:8010
```

Despray Operations:

The **despray** *operation* combines file parts from all the nodes of the cluster into a single file on the landing zone.

These *options* are used by the **despray** operation:

srcname	The logical name of the source file. This may contain wildcard characters (* and ?) to include multiple source files in the despray to a single *dstfile*.
dstip	Optional. The IP address of the destination machine. If omitted, the information must be supplied by the *dstxml* or *dstplane* parameter. Deprecated, you should use *dstplane* instead.
dstplane	Optional. The destination storage plane. Note: *dstplane* should not be used at the same time as *dstip*. In a containerized deployment, *dstplane* is required if you have more than one Landing Zone.

© 2026 HPCC Systems®. All rights reserved

dstfile	Optional. The path to the destination file. This may contain wildcard characters (* and ?) to despray a single *srcname*to multiple *dstfiles*. If omitted, the information must be supplied by the *dstxml* parameter.
dstxml	The name of the XML file containing the information required for the *dstip* and *dstfile* parameters. This file may have been obtained by previous use of the savexml *operation*. This option provides the feature of splitting a single resulting logical file in the cluster into multiple destination files.
splitprefix	Optional. Both of the following (separated by a comma):
filename{:*length***}**	Uses the prepended filename (see the *prefix* option to the spray *operation*) to split out the data into separate files.
filesize{:[B|L][1-8]}	Uses the prepended size of the file (see the *prefix* option to the spray *operation*) to split out the data into separate files. When using wildcard characters (* and ?) to spray multiple source files (srcfile) to a single dstname, you MUST use both the filename and filesize options if you need to be able to despray the contents of each record in the dstname back to the multiple source files they originally came from. If you never need to do that, then the filesize option may be omitted.
wrap= 0 | 1	Optional. If set to 1, desprays as multiple files on the landing zone. Default is 0.
multicopy= 0 | 1	Optional. If set to 1, each destination part gets the whole file. Default is 0.

Examples:

```
dfuplus action=despray dstip=10.150.50.14
   dstfile=/var/lib/HPCCSystems/mydropzone/timezones.txt srcname=RTTEMP::timezones.txt
//the spray.xml file contains:
<File directory="/var/lib/HPCCSystems/mydropzone/"
   group="thor"
   modified="2004-04-27T14:58:38"
   name="zip"
   numparts="2"
   partmask="zip._$P$_of_$N$">
<Attr job="zip1"
   owner="rtaylor"
   recordSize="5"
   replicated="1"
   workunit="D20040427-111857"/>
<Part modified="2004-04-27T14:58:40"
   node="10.150.51.29"
   num="1"
   size="165"/>
<Part modified="2004-04-27T14:58:40"
   node="10.150.51.29"
   num="2"
size="165"/>
</File>
//despray example using the above spray.xml file to split a single
// logical file into multiple destination files
// in this case, zip._1_of_3, zip._2_of_3, and zip._3_of_3 from zip1:
dfuplus action=despray dstxml=spray.xml dstcluster=mythor
          srcname=RTTEMP::myzip1


//from a RECORD structure that looks like this:
imageRecord := RECORD
STRING filename;
DATA image; //first 4 bytes contain the length of the image data
          END;
```

© 2026 HPCC Systems®. All rights reserved

```
//you can despray into its component files like this:
dfuplus action=dspray srcname=le::imagedb
         dstip=10.150.51.26 dstfile=/var/lib/HPCCSystems/mydropzone/
         splitprefix=FILENAME,FILESIZE

// using dstplane
dfuplus action=despray srcname=mytest::test:spraytest
        dstplane=lzstorageplane
        dstfile=mydespraytest server=127.0.0.1:8010
```

Copy Operations:

The **copy** *operation* copies a logical file (all file parts from all the nodes of the cluster), typically from one cluster to another. It appropriately handles re-distributing the file parts if the source and destination clusters do not have the same number of nodes.

The copy operation can also be used to copy files from other HPCC Systems environments (using the *srcdali* option). This is also known as a remote copy. For a remote copy of a file that contains a variable length field, you must include the **nosplit** option.

To copy a superfile and retain its structure, use the copysuper operation. If you use the copy operation for a superfile, it consolidates all the superfile content into a single logical file on the target, not a superfile. This is not valid for a superfile containing INDEXes and will produce an error.

These *options* are used by the **copy** *operation*:

srcname	The logical name of the source file.
dstname	The logical name of the destination file.
dstcluster	The name of the destination cluster. For a containerized deployment, set this to the target data plane (For example, dstcluster=data)
srcdali	Optional. The IP address of the source Dali server, if different from the destination Dali (associated with the ESP Server specified in the *server* option).
srcusername	Optional. The username to use to access the *srcdali*. If omitted, the General Options *username* is used.
srcpassword	Optional. The password to use to access the *srcdali*. If omitted, the General Options *password* is used.
preservecompression	Optional. A boolean flag (0 | 1) indicating whether to preserve the compression of the source file. If omitted, the default is 1.
ensure	Optional. Copies logical file, but does not copy file parts if they already exist. Default is FALSE.

Example:

```
dfuplus action=copy srcname=RTTEMP::timezones.txt
                    dstname=RTTEMP::COPY::timezones.txt
                    dstcluster=mythor
```

Remove Operations:

The **remove** operation deletes a logical file from the system data store, optionally leaving the physical files in place.

These *options* are used by the **remove** *operation*:

© 2026 HPCC Systems®. All rights reserved

name	The logical name of the file to remove.

Example:

```
dfuplus action=remove name=RTTEMP::timezones.txt
```

Rename Operations:

The **rename** operation renames a logical file in the system data store.

These *options* are used by the **rename** *operation*:

srcname	The logical name of the source file.
dstname	The logical name of the destination file.

Example:

```
dfuplus action=rename srcname=RTTEMP::timezones.txt dstname=RTTEMP::NewTimezones.txt
```

List Operations:

The **list** operation produces a list of logical files in the system data store.

These *options* are used by the **list** *operation*:

name	The mask defining the logical file names to list.

Example:

```
dfuplus action=list name=*
```

Add Operations:

The **add** operation adds a new logical file to the system data store.

This also allows you to restore a superfile whose information was previously exported using the savexml action. This is especially useful in a Cloud implementation where files are stored in a bucket until a new instance is started.

These *options* are used by the **add** *operation*:

srcxml	The path and name of the source XML file containing exported logical or superfile information (typically from a previous savexml operation).
dstname	The logical name of the destination file.

These *options* are used by the **add** *operation* to add files from a remote Dali:

dstname	The logical name of the destination file.
srcname	The logical name of the source file.
srcdali	The IP address of the source Dali server.
srcusername	Optional. The username to use to access the *srcdali*. If omitted, the General Options *username* is used.

© 2026 HPCC Systems®. All rights reserved

srcpassword	Optional. The password to use to access the *srcdali*. If omitted, the General Options *password* is used.

Example:

```
dfuplus action=add srcxml=flattimezones.xml dstname=flattimezones.txt
dfuplus action=add srcxml=exportedMysuper.xml dstname=Mysuper
```

Addsuper Operations:

The **addsuper** operation adds subfiles to an existing superfile (see the *SuperFile Management* section of the *Service Library Reference*).

These *options* are used by the **addsuper** *operation*:

superfile	The logical name of the superfile.
subfiles	A comma-delimited list of the logical names of files to add to the superfile. There must be no spaces between the names.
before	Optional. The logical name of the subfile to follow the added *subfiles*. If omitted, the *subfiles* are added to the end.

Example:

```
dfuplus action=addsuper superfile=mysuper subfiles=file1,file2
```

Removesuper Operations:

The **removesuper** operation removes subfiles to an existing superfile (see the *SuperFile Management* section of the *Service Library Reference*).

These *options* are used by the **removesuper** *operation*:

superfile	The logical name of the superfile.
subfiles	Optional. A comma-delimited list of the logical names of files to remove from the superfile. There must be no spaces between the names. If omitted, all files are removed from the superfile.
delete	Optional. A boolean flag (1 | 0) indicating whether to physically delete the *subfiles* in addition to removing them from the superfile. If omitted, the default is 1--physically delete.

Example:

```
dfuplus action=removesuper superfile=mysuper subfiles=file1,file2
```

Copysuper Operations:

The **copysuper** operation copies a superfile from one cluster to another.

These *options* are used by the **copysuper** *operation*:

srcname=<source-super-name>	The logical name of the source superfile.

© 2026 HPCC Systems®. All rights reserved

dstname=<destination-super-name>	The logical name of the destination superfile.
dstcluster=<cluster-name>	The name of the destination cluster. For a containerized deployment, set this to the target data plane (For example, dstcluster=data)
srcdali	The hostname or IP of the source Dali server.
srcusername	Optional, The username to use for the source environment.
srcpassword	Optional, The password to use for the source environment.

Example:

```
dfuplus action=copysuper srcname=jd::super1 dstname=jd::super2 dstcluster=mythor srcdali=.
```

Listsuper Operations:

The **listsuper** operation lists the subfiles in an existing superfile (see the *SuperFile Management* section of the *Service Library Reference*).

These *options* are used by the **listsuper** *operation*:

superfile	The logical name of the superfile.

Example:

```
dfuplus action=listsuper superfile=mysuper
```

Status Operations:

The **status** operation returns the current operational status of a workunit.

These *options* are used by the **status** *operation*:

wuid	The workunit identifier of the workunit.
limit	For publisher workunits, this limits the number of child workunits to list. If set to 0, then only the parent status is returned. Default is 1000.

Examples:

```
dfuplus action=status wuid=D20220816-135243
dfuplus action=status limit=300 wuid=P20230301-123456
dfuplus action=status limit=0 wuid=P20230301-987654
```

Abort Operations:

The **abort** operation aborts execution of a workunit.

These *options* are used by the **abort** *operation*:

wuid	The workunit identifier of the workunit.

Example:

```
dfuplus action=abort wuid=W20050309-093020
```

© 2026 HPCC Systems®. All rights reserved

Resubmit Operations:

The **resubmit** operation re-submits a workunit.

These *options* are used by the **resubmit** *operation*:

wuid	The workunit identifier of the workunit.

Example:

```
dfuplus action=resubmit wuid=W20050309-093020
```

Savexml Operations:

The **savexml** operation saves the logical file map to an XML file.

This feature also allows you to export the metadata from a superfile and then use it later to restore a superfile. This is especially useful in an Cloud implementation where files are stored in a bucket until a new instance is started.

These *options* are used by the **savexml** *operation*:

srcname The logical name of the source file.

srcname	The logical name of the source file. This can be the logical name of a superfile.
dstxml	Optional. The logical name of the destination XML file. If omitted, the XML result is sent to stdout.

Example:

```
dfuplus action=savexml srcname=rttemp::timezones.txt
          dstxml=flattimezones.xml
    // this results in the following XML file:
    <File directory="/var/lib/HPCCSystems/hpcc-data/thor/rttemp"
          group="thor"
          modified="2004-06-18T14:17:16"
          name="timezones.txt"
          numparts="3"
          partmask="timezones.txt._$P$_of_$N$">
    <Attr job="timezones.txt"
          owner="rtaylor"
          recordSize="155"
          replicated="1"
          size="51305"
          workunit="D20040618-101716"/>
    <OrigName>rttemp::timezones.txt</OrigName>
    <Part modified="2004-06-18T14:17:18"
          node="10.150.50.15"
          num="1"
          size="17050"/>
    <Part modified="2004-06-18T14:17:17"
          node="10.150.50.18"
          num="2"
          size="17050"/>
    <Part modified="2004-06-18T14:17:17"
          node="10.150.50.16"
          num="3"
          size="17205"/>
```

© 2026 HPCC Systems®. All rights reserved

```
    </File>
```

Monitor Operations:

The **monitor** operation initiates a DFU workunit to monitor the appearance of a physical or logical file and trigger an event when that file appears.

These *options* are used by the **monitor** *operation*:

event	The name of the user-defined event to trigger. This is used as the first parameter of the ECL EVENT function.
lfn	Optional. The name of the logical file in the DFU to look for. Using this option precludes using the *ip*, *file*, and *sub* options.
ip	Optional. The IP address or name of the server on which the physical file will reside. This may be omitted if the *file* option contains a full URL.
file	Optional. The fully qualified path of the physical file to look for. This may contain wildcard characters (* and ?).
sub	Optional. Specifies searching subdirectories for the physical file if the *file* option contains wildcard characters (* and ?).
shotlimit	Optional. The number of arrival events to generate before marking the DFU workunit as complete. A value of negative one (-1) indicates continuing until the workunit is manually aborted. If omitted, the default value is one (1).

Note the following caveats and restrictions:

1) If a matching file already exists when the DFU Monitoring job is started, that file will not generate an event. It will only generate an event once the file has been deleted and recreated.

2) If a file is created and then deleted (or deleted then re-created) between polling intervals, it will not be seen by the monitor and will not trigger an event.

3) Events are only generated on the polling interval.

4) Note that the *event* is generated if the physical file has been created since the last polling interval. Therefore, the *event* may occur before the file is closed and the data all written. To ensure the file is not subsequently read before it is complete you should use a technique that will preclude this possibillity, such as using a separate 'flag' file instead of the file, itself or renaming the file once it has been created and completely written.

5) The EVENT function's subtype parameter (its 2nd parameter) when monitoring physical files is the full URL of the file, with an absolute IP rather than DNS/netbios name of the file. This parameter cannot be retrieved but can only be used for matching a particular value in this.

Example:

```
dfuplus action=monitor event=MyEvent ip=edata10 file=/var/lib/HPCCSystems/mydropzone/arr.txt
dfuplus action=monitor event=MyEvent ip=10.150.10.75
            file=/var/lib/HPCCSystems/mydropzone/* shotlimit=-1 sub=1
dfuplus action=monitor event=MyEvent file=//10.15.13.21/var/lib/HPCCSystems/mydropzone/*.txt
dfuplus action=monitor event=MyEvent lfn=RTTEMP::OUT::MyFile
```

Listhistory Operations:

The listhistory operation returns the history metadata in a logical file.

© 2026 HPCC Systems®. All rights reserved

History metadata is created from a copy, remote copy, or spray operation.

lfn	The logical file name of the source file
outformat	Optional. The format of the result. Valid options are: csv, xml, json, or ascii. The default is xml.
csvheader	Optional. A boolean flag (0 | 1) indicating whether to include header information in the first row when outputting in csv format.

Example:

```
dfuplus action=listhistory lfn=progguide::exampledata::accounts
  // this results in the following XML file:
  <History>
    <Origin ip="127.0.0.1"
            name="accounts"
            operation="DFUcopy"
            owner="EmilyKate"
            path="/var/lib/HPCCSystems/hpcc-data/thor/progguide/exampledata/"
            timestamp="2017-05-11T16:47:32"
            workunit="W20170503-143100"/>
   </History>
```

Erasehistory Operations:

The erasehistory operation removes the history metadata from a logical file.

History metadata is created from a copy, remote copy, or spray operation.

Note: If LDAP authentication is enabled on the system, you must have FULL permission for DFUAccess in order to erase history. See the HPCC Systems Administrator's Guide for details.

lfn	The logical file name of the source file
backup	Optional. A boolean flag (0 | 1) indicating whether to write the history to a file before erasing it. Default is 1 (enable).
dstxml	The logical name of the destination XML file. Required if backup is set to 1,

Example:

```
dfuplus action=erasehistory lfn=progguide::exampledata::accounts_copy dstxml=c:\temp\jim.xml

  // this removes the history metadata from the file and writes an XML file containing the following:
  <History>
    <Origin ip="127.0.0.1"
            name="accounts"
            operation="DFUcopy"
            owner="EmilyKate"
            path="/var/lib/HPCCSystems/hpcc-data/thor/progguide/exampledata/"
            timestamp="2017-05-11T16:47:32"
            workunit="W20170503-143100"/>
   </History>
```

© 2026 HPCC Systems®. All rights reserved

ESDL Command Line Interface

The ESDL Command Syntax

The **esdl** utility tool aids with creating and managing ESDL-based and Dynamic ESDL services on an HPCC cluster. It provides commands for generating artifacts (such as XML, ECL, XSD, WSDL, Java, C++), managing and publishing ESDL definitions and bindings, and advanced configuration using manifest files and log transforms. Each command is documented in detail in the sections below.

esdl [--version] <command> [<options>]

Output Generating ESDL Definitions

xml	Generate XML from ESDL definition.
ecl	Generate ECL from ESDL definition.
xsd	Generate XSD from ESDL definition.
wsdl	Generate WSDL from ESDL definition.

ESDL and DESDL Service Managing

publish	Publish ESDL Definition for ESP use.
list-definitions	List all ESDL definitions.
delete	Delete ESDL Definition.
bind-service	Configure ESDL based service on target ESP (with existing ESP Binding).
list-bindings	List all ESDL bindings.
unbind-service	Remove ESDL based service binding on target ESP.
bind-method	Configure method associated with existing ESDL binding.
unbind-method	Remove method from an ESDL binding on a target ESP.
get-binding	Get ESDL binding.
get	Get ESDL definition.

© 2026 HPCC Systems®. All rights reserved

esdl xml

esdl xml [options] filename.ecm [<outdir>]

filename.ecm	The file containing the ESDL definitions
-r|--recursive	process all includes
-I, --include-path	Locations to look for included ESDL files. They can be absolute or relative paths. If you need to specify multiple directories, you can use multiple -I options or use a single entry with the directories separated with the environment separator character. For Linux, use a colon (:) and for Windows, use a semi-colon (;). The paths can also be set using an environment variable--ESDL_INCLUDE_PATH.
-v|--verbose	display verbose information
-?/-h/--help	show usage page
Output	(srcdir|<outdir>)/filename.xml

This generates XML from the ESDL definition. This XML is an intermediate entity used by the ESDL Engine to create the runtime service definitions. This command is rarely used by itself.

Examples:

```
esdl xml MathService.ecm .
```

© 2026 HPCC Systems®. All rights reserved

esdl ecl

esdl ecl sourcePath outputPath [options].

sourcePath	The absolute path to the ESDL Definition file containing the EsdlService definition for the service.
outputPath	The absolute path to the location where ECL output is to be written.
-x, --expandedxml	Output expanded XML files.
--includes	If present, process all included files.
--rollup	If present, rollup all processed includes to a single ECL output file.
-cde	Specifies the HPCC Systems Component files directory (location of xslt files).
--ecl-imports	Comma-delimited import list to be attached to the output ECL. Each entry generates a corresponding IMPORT statement.
--ecl-header	Text to include in header or target (generated) file (must be valid ECL).
-I, --include-path	Locations to look for included ESDL files. They can be absolute or relative paths. If you need to specify multiple directories, you can use multiple -I options or use a single entry with the directories separated with the environment separator character. For Linux, use a colon (:) and for Windows, use a semi-colon (;). The paths can also be set using an environment variable--ESDL_INCLUDE_PATH.
Output	(sourcePath|outputPath>)/filename.ecl

This generates ECL structures from ESDL definition. These structures create the interface (entry and exit points) to the Roxie query.

Examples:

```
esdl ecl MathService.ecm .
```

© 2026 HPCC Systems®. All rights reserved

esdl xsd

esdl xsd sourcePath serviceName [options]

sourcePath	The absolute path to the ESDL Definition file containing the EsdlService definition for the service.
serviceName	Name of ESDL Service defined in the given ESDL file.
--version <version number>	Constrain to interface version
*--method <method name>[;<method name>]**	Constrain to list of specific method(s)
--xslt <xslt file path>	Path to '/xslt/esxdl2xsd.xslt' file to transform EsdlDef to XSD
--preprocess-output <raw output directory> :	Output preprocessed XML file to specified directory before applying XSLT transform
--annotate <all | none>	Flag turning on either all annotations or none. By default, annotations are generated for Enumerations. Setting the flag to 'none' will disable those as well. Setting it to al' enables additional annotations such as collapsed, cols, form_ui, html_head and rows.
--noopt	Turns off the enforcement of 'optional' attributes on elements. If no -noopt is specified then all elements with an 'optional' are included in the output. By default 'optional' filtering is enforced.
-opt,--optional <param value>	Value to use for optional tag filter when gathering dependencies. For example, passing 'internal' when some ESDL definition objects have the attribute optional("internal") ensures they appear in the XSD, otherwise they'd be filtered out
-tns,--target-namespace <target namespace>	The target namespace passed to the transform via the parameter 'tnsParam' used for the final output of the XSD.
-n <int> .	Number of times to run transform after loading XSLT. Defaults to 1
--show-inheritance	Turns off the collapse feature. Collapsing optimizes the XML output to strip out structures only used for inheritance, and collapses their elements into their child. That simplifies the stylesheet. By default this option is on
--no-arrayof	Supresses the use of the arrrayof element. arrayof optimizes the XML output to include 'ArrayOf...' structure definitions for those EsdlArray elements with no item_tag attribute. Works in conjunction with an optimized stylesheet that doesn't generate these itself. This defaults to on.
-v|--verbose	display verbose information
-?/-h/--help	show usage page
Output	(srcdir|<outdir>)/filename.ecl

This generates XSD from the ESDL definition.

Examples:

```
esdl xsd MathService.ecm MathService
```

© 2026 HPCC Systems®. All rights reserved

esdl wsdl

esdl wsdl sourcePath serviceName [options]

sourcePath	The absolute path to the ESDL Definition file containing the EsdlService definition for the service.
serviceName	Name of ESDL Service defined in the given ESDL file.
--version <version number>	Constrain to interface version
*--method <method name>[;<method name>]**	Constrain to list of specific method(s)
--xslt <xslt file path>	Path to '/xslt/esxdl2xsd.xslt' file to transform EsdlDef to XSD
--preprocess-output <raw output directory> :	Output preprocessed XML file to specified directory before applying XSLT transform
--annotate <all | none>	Flag turning on either all annotations or none. By default, annotations are generated for Enumerations. Setting the flag to 'none' will disable those as well. Setting it to al' enables additional annotations such as collapsed, cols, form_ui, html_head and rows.
--noopt	Turns off the enforcement of 'optional' attributes on elements. If no -noopt is specified then all elements with an 'optional' are included in the output. By default 'optional' filtering is enforced.
-opt,--optional <param value>	Value to use for optional tag filter when gathering dependencies. For example, passing 'internal' when some ESDL definition objects have the attribute optional("internal") ensures they appear in the XSD, otherwise they'd be filtered out
-tns,--target-namespace <target namespace>	The target namespace passed to the transform via the parameter 'tnsParam' used for the final output of the XSD.
-n <int> .	Number of times to run transform after loading XSLT. Defaults to 1
--show-inheritance	Turns off the collapse feature. Collapsing optimizes the XML output to strip out structures only used for inheritance, and collapses their elements into their child. That simplifies the stylesheet. By default this option is on
--no-arrayof	Supresses the use of the arrrayof element. arrayof optimizes the XML output to include 'ArrayOf...' structure definitions for those EsdlArray elements with no item_tag attribute. Works in conjunction with an optimized stylesheet that doesn't generate these itself. This defaults to on.
--wsdladdress	Defines the output WSDL file's location address
-v|--verbose	display verbose information
-?/-h/--help	show usage page
Output	(srcdir|<outdir>)/filename.ecl

This generates WSDL from ESDL definition.

Examples:

© 2026 HPCC Systems®. All rights reserved

```
esdl wsdl MathService.ecm MathService
```

© 2026 HPCC Systems®. All rights reserved

esdl publish

esdl publish <filename.(ecm|esdl|xml)> <servicename> [options]

filename	The ESDL (*.ecm, *.esdl, or *.xml) file containing the service definitions.
servicename	The name of the service to publish. Optional if the ESDL definition contains only one service.
--overwrite	Overwrite the latest version of this ESDL Definition
-I, --include-path	Locations to look for included ESDL files. They can be absolute or relative paths. If you need to specify multiple directories, you can use multiple -I options or use a single entry with the directories separated with the environment separator character. For Linux, use a colon (:) and for Windows, use a semi-colon (;). The paths can also be set using an environment variable--ESDL_INCLUDE_PATH.
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Publishes an ESDL service definition to the system datastore.

Examples:

```
esdl publish MathService.ecm MathService -s nnn.nnn.nnn.nnn --port 8010
```

© 2026 HPCC Systems®. All rights reserved

esdl list-definitions

esdl list-definitions [options]

-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

This command lists published definitions

Example:

```
esdl list-definitions -s nnn.nnn.nnn.nnn --port 8010
```

© 2026 HPCC Systems®. All rights reserved

esdl delete

esdl delete <ESDLDefinitionID> [options]

ESDLDefinitionID	The ID of the ESDL service definition to delete
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to delete an ESDL Service definition. If the Service definition is bound, you must first unbind it.

Example:

```
esdl delete MathService.2 -s nnn.nnn.nnn.nnn --port 8010
```

© 2026 HPCC Systems®. All rights reserved

esdl bind-service

esdl bind-service <TargetESPProcessName> <TargetESPBindingPort> <ESDLDefinitionId> (<ESDLServiceName>) [command options]

TargetESPProcessName	The target ESP Process name
TargetESPBindingPort | TargetESPServiceName	Either target ESP binding port or the target ESP service name
ESDLDefinitionId	The Name and version of the ESDL definition to bind to this service (must already be defined in Dali)
ESDLServiceName	The Name of the ESDL Service (as defined in the ESDL Definition) Required if ESDL definition contains multiple services
--config <file | XML>	Configuration XML (either inline or as a file reference)
--overwrite	Overwrite the latest version of this ESDL Definition
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to bind a Dynamic ESDL-based ESP service to an ESDL definition.

To bind an ESDL Service, provide the target ESP process name (ESP Process which will host the ESP Service as defined in the ESDL Definition.)

You must also provide either the port on which this service is configured to run (ESP Binding) or the name of the service you are binding.

Optionally provide configuration information either directly inline or using a configuration file XML in the following syntax:

```
<Methods>
  <Method name="myMthd1" url="<RoxieIPRange>:9876/path?param=value" user="me" password="mypw"/>
  <Method name="myMthd2" url="<RoxieIPRange>:9876/path?param=value" user="me" password="mypw"/>
</Methods>
```

Example:

```
esdl bind-service myesp 8003 MathService.1 MathService --config MathSvcCfg.xml
                  -s nnn.nnn.nnn.nnn -p 8010
```

Configuring ESDL binding methods

The DESDL binding methods can optionally provide context information to the target ECL query. The way this information is configured, is by appending child elements to the Method (<Method>...</Method>) portion of the ESDL Binding.

For example, the following XML provides a sample ESDL Binding.

© 2026 HPCC Systems®. All rights reserved

```
<Methods>
  <Method name="AddThis" url="<RoxieIPRange>:9876" querytype="roxie" queryname="AddThis"/>
</Methods>
```

If this Method requires context information, for example about gateways, then you could include the Gateways Structure (<Gateways>...</Gateways>) depicted as follows.

```
<Methods>
  <Method name="AddThis" url="<RoxieIPRange>:9876" querytype="roxie" queryname="AddThis">
    <!--Optional Method Context Information start-->
    <Gateways>
      <Gateway name="mygateway" url="1.1.1.1:2222/someservice/somemethod/>
      <Gateway name="anothergateway" url="2.2.2.2:9999/someservice/somemethod/>
    </Gateways>
    <!--Optional Method Context Information end-->
  </Method>
</Methods>
```

The DESDL ESP does not pose any restrictions on the layout of this information, only that it is valid XML. This provides the flexibility to include context information in any valid XML format.

Roxie (query) ECL developers need to decide what information they will need from the ESP request and design how that information is laid-out in the ESP request and ESDL binding configuration.

In the following example, every "AddThis" request processed by the ESP and sent to Roxie would contain the sample gateway information in the request context.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
<soap:Body>
  <roxie.AddThis>
   <Context>
    <Row>
     <Common>
      <ESP>
       <ServiceName>wsmath</ServiceName>
       <Config>
        <Method name="AddThis" url="<RoxieIPRange>:9876" querytype="roxie" queryname="AddThis">
          <Gateways>
            <Gateway name="mygateway" url="1.1.1.1:2222/someservice/somemethod/>
            <Gateway name="anothergateway" url="2.2.2.2:9999/someservice/somemethod/>
          </Gateways>
        </Method>
       </Config>
      </ESP>
        <TransactionId>sometrxid</TransactionId>
     </Common>
    </Row>
   </Context>
   <AddThisRequest>
    <Row>
     <Number1>34</Number1>
     <Number2>232</Number2>
    </Row>
   </AddThisRequest>
  </roxie.AddThis>
</soap:Body>
</soap:Envelope>
```

The ECL query consumes this information and is free to do whatever it needs to with it. In some instances, the query needs to send a request to a gateway in order to properly process the current request. It can interrogate the context information for the appropriate gateway's connection information, then use that information to create the actual gateway request connection.

© 2026 HPCC Systems®. All rights reserved

Configuring ESDL binding for Proxy Mode methods

You can specify that ESDL service methods be proxied to another ESP instance. Set up the proxy in the dynamic configuration associated with the dESDL service.

Under the Methods tag where you would add Method tags, you can also add Proxy tags, as shown here:

```
<Methods>
  <Method name="myMethod" url="http://10.45.22.1:292/somepath"/>
  <Method name="myMethod2" url="http://10.45.22.1:292/somepath"/>
  <Proxy method="myMethod3" forwardTo="http://10.45.22.1:292" />
  <Proxy method="myWild*" forwardTo="http://10.45.22.1:292" />
</Methods>
```

The Proxy tag also supports wildcards:

```
<Proxy method="myWild*" forwardTo="http://10.45.22.1:292" />
```

This example binds all methods matching the pattern: myWild*

© 2026 HPCC Systems®. All rights reserved

esdl list-bindings

esdl list-bindings [options]

-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to list bindings on a server.

Example:

```
esdl list-bindings -s nnn.nnn.nnn.nnn -p 8010
```

© 2026 HPCC Systems®. All rights reserved

esdl unbind-service

esdl unbind-service <ESPBindingID> [options]

ESPBindingID	The ESDL Binding ID
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to unbind ESDL service based bindings.

To unbind a given ESDL binding, provide the ESP process name and the ESDL binding ID

Available ESDL bindings to unbind can be found using the "esdl list-bindings" command

Example:

```
esdl unbind-service myesp.8003.MathService
```

© 2026 HPCC Systems®. All rights reserved

esdl bind-method

esdl bind-method <TargetESDLBindingID> <TargetMethodName> [options]

TargetESDLBindingID	The id of the target ESDL binding (must exist in Dali)
TargetMethodName	The name of the target method (must exist in the service ESDL definition)
--config <file | XML>	Configuration XML (either inline or as a file reference)
--overwrite	Overwrite the latest version of this ESDL Definition
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to publish ESDL Service based bindings.

To bind an ESDL Service, provide the target ESP process name (ESP Process which will host the ESP Service as defined in the ESDL Definition.)

You must also provide the port on which this service is configured to run (ESP Binding), and the name of the service you are binding.

Optionally provide configuration information either directly inline or using a configuration file XML in the following syntax:

```
<Methods>
  <Method name="myMthd1" url="http://<RoxieIPRange>:9876/path?param=value" user="me" password="mypw"/>
  <Method name="myMthd2" url="http://<RoxieIPRange>:9876/path?param=value" user="me" password="mypw"/>
</Methods>
```

Example:

```
esdl bind-method myesp.8003.MathService AddThis --config myMethods.xml
```

© 2026 HPCC Systems®. All rights reserved

esdl unbind-method

esdl unbind-method <ESDLBindingID> <MethodName> [options]

ESDLBindingID	The ID of the ESDL Binding
MethodName	The name of the target method (must exist in the service ESDL definition)
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to unbind a method configuration associated with a given ESDL binding. To unbind a method, provide the ID of the ESDL binding and the name of the method you are unbinding.

Example:

```
esdl unbind-method myesp.8003.MathService AddThis
```

© 2026 HPCC Systems®. All rights reserved

esdl get-binding

esdl get-binding <ESDLBindingId> [options]

ESDLBindingId	The target ESDL binding id <ESPProcessName>.<Port>.<ServiceName>
-s, --server	The IP Address or hostname of ESP server running ECL Watch services
--port	The ECL Watch services port (Default is 8010)
-u, --username	The username (if necessary)
-pw, --password	The password (if necessary)
--version <ver>	ESDL service version
--help	display usage information for the given command
-v, --verbose	Output additional tracing information

Use this command to get DESDL Service based bindings.

To specify the target DESDL based service configuration, provide the target ID of the ESDL binding, which is normally in the format <ESPProcessName>.<Port>.<ServiceName>

Example:

```
esdl get-binding  myesp.8003.MathService -s nnn.nnn.nnn.nnn -p 8010
```

© 2026 HPCC Systems®. All rights reserved

esdl manifest

esdl manifest <manifest-file> [options]

-I | --include-path <path>	Search path for external files included in the manifest. Use once for each path.
--outfile <filename>	Path and name of the output file.
--output-type <type>	Overrides the value supplied in the manifest attribute Manifest/@output-Type. Allowed values are 'binding' or 'bundle'. Default is 'bundle'.
--help	Display usage information for the given command.
-v, --verbose	Output additional tracing information.
-tcat, --trace-category <flags>	Control which debug messages are output; a case-insensitive, comma-delimited combination of: dev, admin, user, err, warn, prog, info. Errors and warnings are enabled by default if not verbose, and all are enabled when verbose. Use an empty <flags> value to disable all.

For a detailed explanation of the variables and options used in the manifest command, see the: https://github.com/hpcc-systems/HPCC-Platform/blob/master/tools/esdlcmd/README.md.

For additional manifest file examples, refer to the repository at: https://github.com/hpcc-systems/HPCC-Platform/tree/master/initfiles/examples/EsdlExample/Manifest.

The **manifest** command creates an XML configuration file for an ESDL ESP from an input XML manifest file. The type of configuration output depends on the manifest file input and on command-line options.

The Manifest File

A manifest file is an XML-formatted template combining elements in and outside of the manifest's `urn:hpcc:esdl:manifest` namespace. Recognized elements of this namespace control the tool while all other markup is copied to the output. The goal of using a manifest file with the tool is to make configuring and deploying services easier:

- The manifest file format abstracts some of the complexity of the actual configuration.
- By allowing you to include external files like ESDL Scripts and XSLTs into the output, you can store and maintain them separately in your repo.

The result of running the manifest tool on a manifest file is an XML artifact suitable for use with the ESDL ESP. Supported output includes:

- *binding*: The output is an ESDL binding that may be published to dali.
- *bundle*: The output is an ESDL bundle file that may be used to launch an ESP in application mode.

Example Manifest File:

```
<em:Manifest xmlns:em="urn:hpcc:esdl:manifest">
  <em:ServiceBinding esdlservice="WsFoobar" id="WsFoobar_desdl_binding" auth_feature="DEFERRED">
    <Methods>
      <em:Scripts>
        <em:Include file="WsFoobar-request-prep.xml"/>
        <em:Include file="WsFoobar-logging-prep.xml"/>
      </em:Scripts>
      <Method name="FoobarSearch" url="127.0.0.1:8888">
```

© 2026 HPCC Systems®. All rights reserved

```
        <em:Scripts>
          <em:Include file="FoobarSearch-scripts.xml"/>
        </em:Scripts>
      </Method>
    </Methods>
    <LoggingManager>
      <LogAgent transformSource="local" name="main-logging">
        <LogDataXPath>
          <LogInfo name="PreparedData" xsl="log-prep"/>
        </LogDataXPath>
        <XSL>
          <em:Transform name="log-prep">
            <em:Include file="log-prep.xslt"/>
          </em:Transform>
        </XSL>
      </LogAgent>
    </LoggingManager>
  </em:ServiceBinding>
  <em:EsdlDefinition>
    <em:Include file="WsFoobar.ecm"/>
  </em:EsdlDefinition>
</em:Manifest>
```

The tool is permissive and flexible, copying through most markup to the output. Recognized elements in the manifest namespace may be treated differently. They are only required in order to take advantage of the automated processing and simplified format of the manifest file. This example highlights the recommended usage of manifest elements to use the tool's capabilities. Although you could replace some of the elements below with verbatim bundle or binding output elements we won't cover that usage here.

Manifest Elements and Attributes

em:Manifest — Required root element. By default the tool outputs a bundle, though you may explicitly override that on the command line or by providing an `@outputType='binding'` attribute.

em:ServiceBinding — Valid for both bundle and binding output. Enables recognition of em:Scripts and em:Transform elements.

em:EsdlDefinition — Relevant only for bundle output. Enables element order preservation and recognition of em:Include as a descendant element.

em:Include — Causes external file contents to be inserted in place of the element. Processing is context dependent; the parent dictates how the file is handled. Facilitates code reuse in a configuration as code development environment.

em:Scripts / em:Transform — Trigger preservation of element order for all descendant elements and enable em:Include recognition.

Manifest Attributes

Manifest Root Attributes

Attribute	Required?	Value	Description
@outputType	No	string	A hint informing the tool which type of output to generate. The command line option --output-type may supersede this value to produce a different output.

© 2026 HPCC Systems®. All rights reserved

Attribute	Required?	Value	Description
@xmlns[:prefix]	Yes	string	The manifest namespace urn:hpcc:esdl:manifest must be declared. The default namespace prefix should not be used unless all other markup is fully qualified.

ServiceBinding Attributes

Attribute	Required?	Value	Description
@esdlservice	Yes	string	Name of the ESDL service to which the binding is bound. Output on the Binding/Definition element. Also used to generate a value for Definition/@id for bundle type output.
@auth_feature	No	string	Used to declare authorization settings if not present in the ESDL Definition, or override them if they are.
@returnSchemaLocationOnOK	No	Boolean	When true, a successful SOAP response (non SOAP-Fault) will include the schema location property. False by default.
@namespace	No	string	String specifying the namespace for all methods in the binding. May contain variables replaced by the ESP at runtime.
@created	No	string	Timestamp of binding creation.
@espbinding	No	string	Set to match the id. Otherwise the value is unset and unused.
@espprocess	No	string	Name of the ESP process this binding is running on.
@id	No	string	Runtime name of the binding. When publishing to dali the value is [ESP Process].[port].[ESDL Service]. When not present in the manifest a default value is generated of the form [@esdlservice]_desdl_binding.
@port	No	string	Port on the ESP Process listening for connections to the binding.
@publishedBy	No	string	Userid of the person publishing the binding.

© 2026 HPCC Systems®. All rights reserved

JSON and XML to ECL Tools

The HPCC Systems Client Tools include utilities to scan JSON and XML data files and generate ECL record definitions. These command-line tools simplify the process of integrating external data sources by automatically generating ECL RECORD structures from sample data files.

Both tools are available in the client tools package and on bare metal installations in the */opt/HPCCSystems/bin* directory. The general purpose of both is to scan data files and derive an ECL RECORD structure that can read those files.

json2ecl

json2ecl [OPTIONS**] [**FILES...**]**

The **json2ecl** utility examines JSON data and deduces the ECL RECORD definitions necessary to parse it. The tool supports both standard JSON documents and NDJSON (Newline Delimited JSON) format. This tool analyzes the structure of JSON input and generates corresponding ECL RECORD definitions with XPATH annotations suitable for use with ECL's JSON() dataset format.

JSON data can be supplied as one or more files or via standard input. The resulting ECL definitions are returned via standard output.

For efficiency, when scanning arrays or NDJSON streams, the tool stops processing after 500 consecutive elements that don't modify the schema. This prevents unnecessary processing of large uniform datasets while ensuring schema accuracy. Use --full-scan to scan all elements if needed.

When multiple files are provided, they are parsed as if they have the same record structure. This is useful when not all fields are defined in a single file, allowing the tool to discover the complete schema across all files.

The tool automatically detects and handles NDJSON format. When multiple top-level JSON objects are detected, the NOROOT flag is automatically added to the generated ECL code.

FILES	One or more JSON files to process. Multiple files are parsed as if they have the same record structure, useful when not all fields are defined in a single file. If omitted, data is read from standard input.
OPTIONS	
-h, --help	Display help message and exit.
-s, --string-type TYPE	Specify the ECL string type to use for string fields. Valid options are: UTF8, STRING, VARSTRING, UNICODE. Default: UTF8
--full-scan	Scan all array elements or NDJSON elements without limit. By default, for efficiency, the tool stops processing after 500 consecutive elements that don't modify the schema. Use this option to scan all elements if needed.
--version	Display version information and exit.

Known Limitations:

- Quoted numbers (e.g., "1234") are treated as numbers, not strings.
- JSON null values cannot be distinguished from empty strings.

© 2026 HPCC Systems®. All rights reserved

- Field names containing spaces may not be handled correctly due to underlying JSON parser behavior (e.g., 'this is a test' may become 'this_sis_sa_stest'). Use underscore or camelCase naming instead.

Usage Notes:

- The tool automatically detects field types from the JSON data (strings, integers, floating-point numbers, booleans, objects, arrays).
- Nested objects are converted to child RECORD structures with automatically generated names.
- Arrays are handled appropriately based on their content type (primitive arrays or arrays of objects).
- All generated fields include XPATH annotations that correspond to the JSON path to that field.
- ECL reserved words used as field names are automatically prefixed with 'f_' to avoid conflicts.
- The default string type is UTF8. Use the -s option to change to STRING, VARSTRING, or UNICODE as needed.
- When processing multiple files, ensure they represent the same logical data structure with different field coverage.

Examples:

Scan JSON data from a file

```
json2ecl data.json
```

Reads JSON data from *data.json* and writes the ECL record definition to standard output.

Scan JSON from standard input

```
cat sample.json | json2ecl
```

Reads JSON from standard input and outputs the ECL record definition.

Process multiple files with STRING type

```
json2ecl -s STRING file1.json file2.json
```

Processes multiple JSON files, combining their schemas, and generates ECL record definitions using STRING type instead of the default UTF8.

Redirect output to a file

```
json2ecl data.json > layout.ecl
```

Generates the ECL record definition and saves it to *layout.ecl*.

Scan all elements in large file

```
json2ecl --full-scan large_dataset.json
```

Processes all elements in the JSON file without the default 500 element limit.

Sample Input and Output

Given a JSON file *foo.json* with the following content:

```
{
```

© 2026 HPCC Systems®. All rights reserved

```
  "foo": "bar",
  "start": 12,
  "end": 98.76
}
```

Running the command:

```
json2ecl foo.json
```

Generates the following ECL output:

```
FOO_LAYOUT := RECORD
    UTF8 foo {XPATH('foo')};
    UNSIGNED start {XPATH('start')};
    REAL f_end {XPATH('end')};
END;

// ds := DATASET('~data::foo', FOO_LAYOUT, JSON());
```

Note that the generated code includes:

- XPATH annotations for each field indicating the JSON path to the data
- Appropriate ECL types based on the JSON values (UTF8 for strings, UNSIGNED for positive integers, REAL for floating-point numbers)
- A commented-out sample DATASET declaration showing how to use the generated layout
- Field names prefixed with 'f_' if they conflict with ECL reserved words (e.g., 'end' becomes 'f_end')

Using JSON Layouts

ECL record structures generated by json2ecl can be used directly with ECL's DATASET declaration using the JSON() format:

```
// Generated layout from json2ecl
FOO_LAYOUT := RECORD
    UTF8 foo {XPATH('foo')};
    UNSIGNED start {XPATH('start')};
    REAL f_end {XPATH('end')};
END;

// Use the layout to read JSON data
ds := DATASET('~data::foo', FOO_LAYOUT, JSON());
```

For more complex structures with nested records:

```
// Nested JSON layout example
PERSON_LAYOUT := RECORD
    UTF8 firstName {XPATH('firstName')};
    UTF8 lastName {XPATH('lastName')};
    UNSIGNED age {XPATH('age')};
    DATASET(ADDRESS_REC) addresses {XPATH('addresses')};
END;

// Read the JSON file
people := DATASET('~data::people', PERSON_LAYOUT, JSON());
```

Best Practices:

- **Review Generated Code:** Always review the generated ECL code. The tools provide a starting point, but you may need to refine field types or lengths based on your specific requirements.

© 2026 HPCC Systems®. All rights reserved

- **Use Representative Samples:** Ensure your sample data files include representative examples of all field variations. The tools analyze the data to determine schema, so incomplete samples may result in incomplete schemas.
- **Process Multiple Files:** If your data has optional fields that don't appear in every record, process multiple representative files together to capture the complete schema.
- **Test with Real Data:** Always test the generated record definitions with your actual data to verify correct parsing and type handling.
- **Handle Large Files:** For very large files, the default 500-element scan limit is usually sufficient. Only use --full-scan if you suspect the schema changes throughout the file.
- **String Type Selection:** Choose the appropriate string type (-s option) based on your data. UTF8 is the default and recommended for most use cases, especially with international data.
- **Handle Reserved Words:** Be aware that ECL reserved words in field names are automatically prefixed. Check the generated code for these prefixes (e.g., 'f_' prefix).
- **Save Generated Layouts:** Save the generated ECL layouts to files and include them in your version control system for tracking changes over time.
- **XML Namespaces:** For XML files with namespaces, you may need to manually clean up or handle the xmlns:* attribute fields in the generated code.

© 2026 HPCC Systems®. All rights reserved

xml2ecl

xml2ecl [OPTIONS] [FILES...]

The **xml2ecl** utility examines XML data and deduces the ECL RECORD definitions necessary to parse it. This tool analyzes XML structure and generates corresponding ECL RECORD definitions with XPATH annotations suitable for use with ECL's XML() dataset format or PARSE function.

XML data can be supplied as one or more files or via standard input. The resulting ECL definitions are returned via standard output.

FILES	One or more XML files to process. Multiple files are parsed as if they have the same record structure, useful when not all fields are defined in a single file. If omitted, data is read from standard input.
OPTIONS	
-h, --help	Display help message and exit.
-s, --string-type TYPE	Specify the ECL string type to use for string fields. Valid options are: UTF8, STRING, VARSTRING, UNICODE. Default: UTF8
--full-scan	Scan all array elements without limit. By default, for efficiency, the tool stops processing after 500 consecutive elements that don't modify the schema. Use this option to scan all elements if needed.
--version	Display version information and exit.

xml2ecl examines XML data and deduces the ECL RECORD definitions necessary to parse it. The tool analyzes XML element structure, attributes, and content to generate appropriate ECL RECORD definitions.

Multiple Files: When multiple files are provided, they are parsed as if they have the same record structure. This is useful when not all fields are defined in a single file, allowing the tool to discover the complete schema across all files.

Attribute Handling: XML attributes are included in the schema with their original names. When an element has both attributes and text content, the text is stored in a field named '_inner_value'.

Performance: For efficiency, when scanning repeated elements, the tool stops processing after 500 consecutive elements that don't modify the schema. This prevents unnecessary processing of large uniform datasets while ensuring schema accuracy. Use --full-scan to scan all elements if needed.

Known Limitations:

- Namespace declarations (xmlns:* attributes) are preserved as fields in the generated ECL schema.
- Mixed content (text and elements interleaved) may not be fully supported.

Examples:

Scan XML data from a file

```
xml2ecl data.xml
```

Reads XML data from *data.xml* and writes the ECL record definition to standard output.

Scan XML from standard input

```
cat document.xml | xml2ecl
```

© 2026 HPCC Systems®. All rights reserved

Reads XML from standard input and outputs the ECL record definition.

Process multiple files with STRING type

```
xml2ecl -s STRING file1.xml file2.xml
```

Processes multiple XML files, combining their schemas, and generates ECL record definitions using STRING type instead of the default UTF8.

Redirect output to a file

```
xml2ecl data.xml > layout.ecl
```

Generates the ECL record definition and saves it to *layout.ecl*.

Scan all elements in large file

```
xml2ecl --full-scan large_dataset.xml
```

Processes all elements in the XML file without the default 500 element limit.

Sample Usage:

Given an XML file *customer.xml* with the following content:

```
<?xml version="1.0" encoding="UTF-8"?>
<Customer id="12345">
  <FirstName>Ember</FirstName>
  <LastName>Shiloh</LastName>
  <Age>28</Age>
  <Email>ember.shiloh@example.com</Email>
  <Address>
    <Street>456 Oak Avenue</Street>
    <City>Portland</City>
    <State>OR</State>
    <ZipCode>97201</ZipCode>
  </Address>
</Customer>
```

Running the command:

```
xml2ecl customer.xml
```

Generates ECL output with:

- XPATH annotations for each field indicating the XML path to the data
- Appropriate ECL types based on the XML content (UTF8 for text, UNSIGNED for positive integers)
- Nested RECORD structures for nested XML elements (e.g., Address)
- XML attributes included as fields with their original names (e.g., id)
- A commented-out sample DATASET declaration showing how to use the generated layout

When an XML element contains both attributes and text content, the text content is stored in a field named '_inner_value'.

Usage Notes:

- The tool automatically detects field types from the XML data (text content, numeric values, nested elements).

© 2026 HPCC Systems®. All rights reserved

- Nested elements are converted to child RECORD structures with automatically generated names.
- Repeating elements (arrays) are handled appropriately with XPATH annotations.
- XML attributes are included in the schema with their original names.
- When an element has both attributes and text content, the text is stored in a field named '_inner_value'.
- All generated fields include XPATH annotations that correspond to the XML path to that field.
- The default string type is UTF8. Use the -s option to change to STRING, VARSTRING, or UNICODE as needed.
- Namespace declarations (xmlns:* attributes) are preserved as fields. You may need to manually remove or handle these in your ECL code.
- When processing multiple files, ensure they represent the same logical data structure with different field coverage.

Using XML Layouts:

ECL record structures generated by xml2ecl can be used with ECL's DATASET declaration using the XML() format or with the PARSE function:

```
// Using DATASET with XML() format
CUSTOMER_LAYOUT := RECORD
    UTF8 id {XPATH('@id')};
    UTF8 FirstName {XPATH('FirstName')};
    UTF8 LastName {XPATH('LastName')};
    UNSIGNED Age {XPATH('Age')};
END;

// Read XML file
customers := DATASET('~data::customers.xml',
                     CUSTOMER_LAYOUT,
                     XML('Customer'));
```

Alternatively, using the PARSE function for more control:

```
// Read the raw XML file
xmlFile := DATASET('~data::customers.xml',
                   {STRING line},
                   CSV(SEPARATOR(''), QUOTE('')));

// Parse XML data
parsedCustomers := PARSE(xmlFile,
                         line,
                         CUSTOMER_LAYOUT,
                         XML('Customer'));

OUTPUT(parsedCustomers);
```

Best Practices:

- **Review Generated Code:** Always review the generated ECL code. The tools provide a starting point, but you may need to refine field types or lengths based on your specific requirements.
- **Use Representative Samples:** Ensure your sample data files include representative examples of all field variations. The tools analyze the data to determine schema, so incomplete samples may result in incomplete schemas.

© 2026 HPCC Systems®. All rights reserved

- **Process Multiple Files:** If your data has optional fields that don't appear in every record, process multiple representative files together to capture the complete schema.
- **Test with Real Data:** Always test the generated record definitions with your actual data to verify correct parsing and type handling.
- **Handle Large Files:** For very large files, the default 500-element scan limit is usually sufficient. Only use --full-scan if you suspect the schema changes throughout the file.
- **String Type Selection:** Choose the appropriate string type (-s option) based on your data. UTF8 is the default and recommended for most use cases, especially with international data.
- **Handle Reserved Words:** Be aware that ECL reserved words in field names are automatically prefixed. Check the generated code for these prefixes (e.g., 'f_' prefix).
- **Save Generated Layouts:** Save the generated ECL layouts to files and include them in your version control system for tracking changes over time.
- **XML Namespaces:** For XML files with namespaces, you may need to manually clean up or handle the xmlns:* attribute fields in the generated code.

© 2026 HPCC Systems®. All rights reserved