

Choose E-mail Handler dialog box

[Related Topics](#)

Use this dialog box to select a mail protocol for the Rational E-mail Reader:

- MAPI
- SMTP

You will need to provide a MAPI profile or the SMTP/POP3 server and log on information.

- ▶ Click the **Next** button.

Note:

To configure e-mail for a Rational product or a RequisitePro project using the MAPI protocol, you must configure the MAPI profile for each product or project on each client machine. Use your e-mail administration tool to add the MAPI profiles to each PC that will access the specific Rational products or projects.

Choose Profile dialog box

[Related Topics](#)

Use this dialog box to select a MAPI profile to identify a mailbox that will be associated with the Rational database or project. To create a new profile, close the Rational E-mail Reader application, create a new profile, and return to this application.

- ▶ Select a **MAPI profile** from the list and click **Next**.

Note:

To configure e-mail for a Rational product or a RequisitePro project using the MAPI protocol, you must configure the MAPI profile for each product or project on each client machine. Use your e-mail administration tool to add the MAPI profiles to each PC that will access the specific Rational products or projects.

Configure Mail Server Account dialog box

[Related Topics](#)

Use this dialog box to specify a logon ID and password for accessing the POP3 server.

▶ Type the following entries and then click **Next**.

Login Type a POP3 server **log on** for the e-mail address you specified in the previous dialog box. This is the mailbox you are associating database or project.

Password Type the user's password for the e-mail logon ID.

Configure Mail Server dialog box

[Related Topics](#)

Use this dialog box to identify the SMTP server, the POP3 server, and a POP E-mail address.

▶ Type the following entries and then click **Next**.

SMTP Server Type the name of the SMTP server.

POP Server Type the name of the POP3 server.

POP E-mail Address Type the **e-mail address** that will be associated with a single Rational database or RequisitePro project.

Note:

The e-mail administrator must define this e-mail account for the project or database. Do not use a personal e-mail address; this e-mail account must be dedicated to the specific Rational database or project.

Configure Rational E-mail Handler dialog box

[Related Topics](#)

Use this dialog box to select an e-mail handler for a specific Rational product. In addition, you can use the Browse feature to select a product database or RequisitePro project.

► Make the following selections and then click **Finish**:

E-Mail Handler

Select a Rational product from the list.

Choose and Configure Database

Click **Browse** to select a Rational product database or a RequisitePro project.

Enabling e-mail using the Rational E-mail Reader

[Related Topics](#)

Use the E-mail Reader to enable e-mail for Rational Software Corporation products. The e-mail administrator must define an e-mail account for the project or database. Do not use a personal e-mail address; this e-mail account must be dedicated to the specific Rational database or project.

1. Start the E-mail Reader by browsing to the Program Files/Rational/common folder and clicking on mailreader.exe. The Rational E-mail Reader dialog box appears.
2. Click the **Add** button to create a mailbox for specific Rational product. The Choose E-mail Handler dialog box appears.
3. At the E-mail Provider field, select the e-mail protocol that is available on your network:
 - MAPI
 - SMTP
4. Click the **Next** button.

For MAPI, the Choose **Profile** dialog box appears.

For SMTP, the Choose **Mail Server** dialog box appears.

5. For MAPI, at the **Choose MAPI Profile** field, select a profile to associate with your Rational database or RequisitePro project. The Configure Rational E-mail Handler dialog box appears. Skip to Step 8.
6. For SMTP, type the name of your **SMTP server** and **POP server** (on many networks, these are the same), then click **Next**.
7. For SMTP, in the Configure Mail Server Account dialog box, enter the **Pop E-mail Address**, **Login**, and **Password** to associate with a single Rational database or RequisitePro project, then click **Next**.
8. In the Configure Rational E-mail Handler dialog box, at the **E-Mail Handler** field, select a specific Rational product. Click **Configure**. A set-up dialog box appears for the Rational product you selected.
9. Configure the mailbox for the product you are using: [RequisitePro mailbox](#) or [ClearQuest mailbox](#).

Using the Rational E-Mailreader

[Related Topics](#)

The Rational E-Mailreader allows administrators to enable e-mail capabilities for various Rational Software products. The E-Mailreader uses a different e-mail handler for each Rational product. In addition, a unique mailbox is associated with each Rational product database (or RequisitePro project).

Prior to configuring an e-mail handler for a specific Rational product, the E-mail Reader requires that you identify:

- an email protocol (SMTP/POP3 or MAPI);
- an SMTP server and POP server with a unique email address, or a unique MAPI profile for each Rational database or RequisitePro project;
- a log on and password for the POP server (SMTP protocol only);
- an e-mail handler for a specific Rational product.

Rational E-mail Database and Project Mailboxes tab

[Related Topics](#)

Use this tab for managing e-mail handlers that configure e-mail for Rational products. An e-mail handler associates a mailbox with a specific Rational product database or RequisitePro project. This tab contains the following fields and controls.

Mailbox Name	The Mailbox Name identifies a unique Rational database or project. The Product field associates the mailbox with a Rational product. These entries are configured using the other controls.
Add	Add another mailbox entry.
Remove	Delete the selected mailbox entry.
Properties	View or modify the selected mailbox entry.
Start Server/Stop Server	Initiates the server processing of the e-mail. Click it again to halt the server processing.
Close	Stop the server processing and close the Rational E-mail Reader application.

Reader Information and Options tab

[Related Topics](#)

This tab allows you to set the following options.

- | | |
|------------------------------|---|
| Notify Admin On Error | Automatically sends an e-mail notification to the Admin E-Mail Address if an error occurs. We recommend you use this option. |
| Admin E-mail Address | Provides an e-mail address for error notifications. |
| Pool Interval | Determines how often the E-mail Reader checks the status. |
| Log File Path | The location of a trace log. Start the trace log only when you need to troubleshoot a problem. The size of this file might increase rapidly. The trace log holds all the information that the E-mail Reader processes, including attachments. |
| Start Trace Now | Begins appending information to the trace log immediately. You have the option of creating a new trace log, rather than having information appended to the existing trace log. |

RequisitePro E-mail Setup dialog box

[Related Topics](#)

Use this dialog box to identify a RequisitePro project and provide the user name and password access to the selected project. This RequisitePro project will be associated with the MAPI profile or SMTP mailbox you have specified in the E-mail Reader setup procedure.

The RequisitePro E-mail Setup dialog box contains the following fields:

Project Path	The full path and project file (.rqs file) for a specific RequisitePro project. Click Browse to navigate to a project.
User Name	Type a RequisitePro user name. This user must have administrator permissions for the selected project.
Password	Type the user's password.

Selecting a RequisitePro project

[Related Topics](#)

Use this dialog box to configure inbound and outbound e-mail for a specific RequisitePro project. The enabled e-mail allows discussion participants to read and reply to discussion items via e-mail.

Discussion participants must have an e-mail address specified in their RequisitePro user information in order to receive discussion-related e-mail. Note that users can enable or disable their outbound discussion e-mail within RequisitePro by using the RequisitePro Tools/E-mail Setup option.

Discussions are specific to individual RequisitePro projects and can include comments, issues, and questions related to a RequisitePro requirement or larger project issues. Discussion items and responses to those items are stored in the RequisitePro database and can be read by project users regardless of whether they have e-mail enabled.

Begin enabling e-mail for a project by following the procedure:

▶ [Enabling e-mail using the E-mail Reader](#)

When the RequisitePro E-mail Setup dialog box appears:

1. Click **Configure** to navigate to a RequisitePro project. The Project Path dialog box appears.
2. Browse to your RequisitePro project directory and select an RQS file for an existing RequisitePro project.
3. Click **Open**. The dialog box closes and the project is entered in the Project Path field on the RequisitePro E-mail Setup dialog box.
4. Type a RequisitePro user name in the User Name field.
Note: This user must have administrator permissions for the selected project in RequisitePro.
5. Type a password for the user and click OK.

```
{button Field Descriptions,JI('>second','dia_RequisitePro_E_mail_Setup')}
```

Configuring a ClearQuest database

[Related Topics](#)

To configure a ClearQuest database to receive records or record modifications by e-mail, use the Rational E-Mailreader to specify ClearQuest as an e-mail handler, then configure your ClearQuest database.

In the **Configure Rational E-mail Handler** dialog box:

1. In the **E-mail Handler** list, choose ClearQuest.
2. Click **Configure** to select a ClearQuest database to use.

In the **Configure ClearQuest Database—Step 1** dialog box:

1. Use the **Database** list to choose the ClearQuest database you want to configure.
2. In the **User** and **Password** fields, log in to the database you specified.
3. Click **Next**.

In the **Configure ClearQuest Database—Step 2** dialog box:

You can configure a default action and default field for each record type. For example, if you are using the TestStudio schema with your ClearQuest database, a suggested default action is Modify, and its recommended default field is Description. If you choose this setting, your users only need to put the defect ID number in the subject line of the e-mail; any comments they make in the body of the e-mail are appended to the Description field.

For each record type for which you want to set defaults:

1. Select the record type in the **Types** list.
2. In the **Default Action** list, choose the default action to apply if the user does not specify an action in the subject line when a user submits or modifies a ClearQuest record by e-mail.
3. In the **Default Field** list, choose the default field to modify if the user does not specify a field in the body of the message when the user submits or modifies a ClearQuest record by e-mail.
4. Click **Finish** to return to the Rational E-Mailreader.

Note:

- § When you configure the Rational E-mail Reader for ClearQuest, specify the action that your users will be using most often. For example, the most common actions might be the Modify action, which allows users to modify the specified record.
- § Use a dedicated e-mail address for the Rational E-mail Reader.
- § Although, you can run the Rational E-mail Reader server on any machine, we recommend that you run it on a machine that is always running and connected to your network to ensure seamless e-mail service for your ClearQuest database.

Understanding e-mail rules

[Related Topics](#)

As an administrator, you can use the [stateless](#) Email_Rule record type to set up the parameters used to determine [when e-mail is sent](#), under which [conditions](#), to which [users or user groups it is sent](#) , and the [content of the e-mail message](#).

You create e-mail rules by submitting Email_Rules records from the ClearQuest Client. To add or modify an e-mail rule, you must have Super User or Schema Designer privileges.

Most ClearQuest predefined schemas include the Email_Rule record type, except for the Blank schema. If you are creating a new schema or the schema you are using does not have this functionality, you must [install](#) the Email [package](#), to add the Email_Rule record type. If you are creating a new record type, you have the option of creating a new Email_Rule for the new record type.

Important Note:

- Renaming or modifying a record type or action in the schema that has existing e-mail rules requires the rules to be updated to reflect the new names. You can [modify your action notification hooks](#) as necessary to include or remove support for e-mail rules.

Creating e-mail rules

[Overview](#)

[Related Topics](#)

Users with Super User or Schema Designer privileges can create e-mail rules in the ClearQuest client.

1. Select **Actions > New** in the ClearQuest client.
2. Select **Email_Rule**. ClearQuest from the list of record types to creates a new e-mail rule record.
3. In the [Rule Controls tab](#), indicate the fields or queries you want the e-mail rule to use.
4. In the [Action Controls tab](#), indicate the actions, action types, or states you want the e-mail rule to use.
5. In the [Display fields tab](#), indicate what you want to include in the e-mail message.
6. In the [To Addressing Info tab](#), indicate which users or user groups to which you want to send the e-mail message.
7. In the [CC Addressing Info tab](#), indicate which users or user groups you want to carbon copy. You can also CC the user who last modified the record involved.

Important Note:


- Renaming or modifying a record type or action in the schema that has existing e-mail rules requires the rules to be updated to reflect the new names. You can [modify your action notification hooks](#) as necessary to include or remove support for e-mail rules.
- The Email_Rule record type is not available in the Common or Blank schemas.

Setting up action control filters

[Overview](#)

[Related Topics](#)

As an optional filter, use the Action Controls tab to further limit the conditions under which the e-mail rule is triggered. Before using this tab, you must complete the required information in the [Rule Controls tab](#). The e-mail rule triggers when the combination of the specified record-type, fields, and Action Controls selections are met.

- § In the categories [Actions](#), [Action Types](#), [Source States](#), or [Destination States](#), make a selection from the Choice dialog box that displays when you click the Browse  button.

Setting up e-mail rule filters

[Overview](#)

[Related Topics](#)

When creating a new e-mail rule in ClearQuest, do the following in the Rule Controls tab:

1. Enter the name of the e-mail rule in the **Name** field.
2. Select the record type to which the rule applies in the **Record Type** field.
3. Enter the names of one or more fields associated with the record in the **Fields to Check for Change** list box. If any field in the list is changed, and you have not specified anything in the Actions control, the e-mail rule is triggered.
4. Select a public query to use as an additional e-mail trigger in the **Filter Query** field.
For example, if you have a query that finds all records associated with a beta release, you can include that query in the e-mail rule. So if a record is included in the query specified, e-mail is sent.
5. Clear the **Active Rule** check box to temporarily disable the rule.

Setting up the message contents

[Overview](#)

[Related Topics](#)

When creating a new e-mail rule in ClearQuest, do the following in the Display Fields tab:

1. Enter the return address for the e-mail message. This address must be entered in text format (e.g., *name@company.com*). Use the same account specified in the mailreader configuration for roundtrip e-mail.

Note The From Address is not applicable when using the MAPI protocol.

2. In the **Subject Fields** control, enter the fields to display in the subject line of the message.
3. In the **Fields to Display** control, enter the fields to display in the message body.
 - § Mark **Show Previous Values** if you want to include the previous field values in the message.
 - § Mark **Include Entire Defect** if you want to include all the records fields in the message.

Note:

The From ("Current User") Address field automatically defaults to the current ClearQuest user's e-mail address. This feature will fail if a valid e-mail address is not included in the user's information.

Addressing the message

[Overview](#)

[Related Topics](#)

ClearQuest uses the list of users created in the ClearQuest Designer as valid values in the Email_Rule record type. If a user does not have a valid e-mail address in their user information, that user won't receive e-mail.

1. On the **To Addressing Info** tab, enter the users and/or groups to which you want to send the e-mail message. The **User field** lists all ClearQuest users. The users must have e-mail address specified in their user information. The **Groups field** lists all ClearQuest groups. All users under this group must have their e-mail address specified in their user information to receive e-mail.
 - n With the **To Addresses (Fields)** list, you can address e-mail based on which users or groups are listed in a field. For example, if you specify the Owner field, e-mail is sent to the user listed in the Owner field of the modified record.
 - n With the **To Addresses (Text)** list, you can send e-mail to a specific address that may not necessarily belong to a ClearQuest user. Enter the e-mail address in text format (e.g., *name@company.com*).
2. On the **CC Addressing Info** tab, enter any additional users and groups to which you want to send the e-mail message.

If you want to send the message to the user who modified the record, click the **CC Actioner** check box.

Editing the Email_Rule record type

[Overview](#)

[Related Topics](#)

You should not make any changes to the Email_Rule record type. Changing the name of the record type or changing the name of any of its fields will break the code that supports e-mail rules.

If you want to add additional information to e-mail messages, you must modify your action notification hooks to create the appropriate e-mail messages.

To otherwise modify an e-mail rule record, see [Creating e-mail rules](#).

💡 Use a base action if you want all action (or most actions) to have e-mail.

Important Note:

- Renaming or modifying a record type or action in the schema that has existing e-mail rules requires the rules to be updated to reflect the new names. You can [modify your action notification hooks](#) as necessary to include or remove support for e-mail rules.

Modifying the action hook notification

[Overview](#)

[Related Topics](#)

The schemas that include the Email_Rule record type use that record type to implement e-mail notifications. If you are modifying one of these schemas, either adding new record types or new actions to existing record types, and want to continue supporting e-mail rules, you must modify the appropriate action notification hooks.

Schemas that include the Email_Rule record type also include a set of global scripts that you call to process the e-mail rules. The only script you really need to call is the RSEM_ProcessEmailRules script.

You pass a record to this script, which the script then checks against the e-mail rules in the database. If the record matches one of the rules, the script generates an e-mail message and sends it. Typically, you would call RSEM_ProcessEmailRules from your action notification hook, as in the following example for a record whose type is **Request**:

```
Sub Request_Notification(actionname, actiontype)
    ' actionname As String
    ' actiontype As Long
    ' action = Modify

    RSEM_ProcessEmailRules (Request)

End Sub
```

Note:

Pre-defined schemas have the Email_rule record type BASE. This action calls RSEM_ProcessEmailRules, and the action type is called for all actions. You can add the RSEM_ProcessEmailRules call to new records.

ClearQuest predefined schemas

[Overview](#)

[Related Topics](#)

ClearQuest includes with several predefined schemas that you use as is or modify to suit your organization's needs.

Blank	Contains system fields only. Use Blank to create a schema from scratch.
Common	Contains metadata that is common to the remaining schemas.
DefectTracking	Contains the fields necessary to start using ClearQuest to track defects in a software development environment.
AnalystStudio	For use with Rational AnalystStudio. Contains customizations for use with the Rational RequisitePro and Rose.
DevelopmentStudio	For use with Rational DevelopmentStudio. Contains fields and rules that work with Rational Purify, Visual Quantify, and Pure Coverage.
TestStudio	For use with Rational TestStudio. Contains fields and rules that work with Rational TeamTest, RequisitePro, Purify, Visual Quantify, and Pure Coverage.
Enterprise	For use with Rational EnterpriseStudio. Contains fields and hooks that work with all Rational products.
UnifiedChangeManagement	For use with Rational ClearCase. ClearQuest and ClearCase support the UCM process.

Copying a schema

[Related Topics](#)

You can copy an entire schema into a schema repository or copy a partial schema into another schema by using the CQLOAD command line utility.

Preparing to use CQLOAD

1. In a command prompt, change to the ClearQuest directory.
-Or-
Add the ClearQuest installation directory to your path.
2. Make sure that the schema to which you want to apply CQLOAD is not checked out in ClearQuest Designer.

Copying a schema into a schema repository

1. Export the schema (all revisions) you want to copy using the command [exportschema](#).
2. Import the copied schema into the schema repository using the command [importschem](#).

Copying a partial schema into an existing schema

1. Export part of a schema (one or more revisions) using the command [exportintegration](#).
2. Import the copied schema into an existing schema using the command [importintegration](#).

Note:

If you run CQLOAD while ClearQuest Designer is running, you'll need to exit the Designer and re-login to see your changes.

Understanding Rational's E-mail reader

[Overview](#)

Rational E-mail Reader is an e-mail server program that must be configured to allow ClearQuest users to submit and update (modify and change state) requests via e-mail.

When you've configured the Rational E-mail Reader, ClearQuest users can communicate with the ClearQuest database in two ways:

- § In conjunction with e-mail notification, ClearQuest users can respond to records they've been notified of and modify those records.
- § Remote users can submit new records without having to logon to the ClearQuest database or access the ClearQuest client or ClearQuest web client.

The Rational E-mail Reader enforces a required format when processing e-mail. The action names and ID are required for non-submitting actions. The required format is in the message body. If any other action is needed or different fields modified, the user must explicitly enter them in the subject and body of the e-mail message, according to the [required format](#).

Important Note:

- Adding new record types to the schema, renaming the record type, or modifying the actions of the existing record types, may require modifications to support the e-mail rules. You can [modify your action notification hooks](#) as necessary to include or remove support for e-mail rules.

ClearQuest e-mail submission format

When submitting or modifying ClearQuest records by e-mail, you must use the following format. Remember if you're submitting a new record, you must include legal values for all required fields.

Subject: <record type> <action> <id>
If modifying an existing record, the record ID must follow the action in the subject line.

Body format: <fieldname: legal value>
<fieldname: legal value>
{<multiline_fieldname: This is an example of a field value for a field whose data type is multi-line text field.
}

Format example for submitting a new record

This example e-mail message submits a defect record and fills in the specified fields.

Subject: defect submit

Body: Headline: inventory report is not running correctly
 Severity: 1-Critical
 Project: ClassicPOS
 {Description: When running an inventory report, application crashes if more than 50 items are included in the report.
 }
 Priority: 2-Give High Attention

Format example for changing state

Subject: defect closed SAMPL00001234

Note_Entry: Followed the steps outlined in the description, and confirmed that the error no longer occurs.

Test Method: Manual

Format for modifying a record using the default action

This is the remote ClearQuest user's response to receiving e-mail notification of the defect. In this case, the ClearQuest administrator has set up the default action as MODIFY, the default field to be Description, and the record type to be defect. The defaults are used when no action is specified in the subject line. Notice that because the Description field is a multi-line text field, { } are used.

Subject: defect SAMPL00000017

Body: Have tried to reproduce the problem with the inventory reports not allowing more than 50 items and it works just fine. Please re-try and provide more information.

Result: Use the MODIFY action to update SAMPL00000017, and appends the value in Description.

Note:

- You must enter legal values for each field you are modifying.
- Follow the required format and be sure to account for all mandatory fields.
- Multi-line text fields such as the Notes field in the TeamTest schema or Description field in the defaultapp, must be surrounded by { }.

• Regarding **(RE:)** is not required at the beginning of the subject line.

ClearCase and SourceSafe add-ins

Both the ClearCase and SourceSafe add-ins have been replaced by packages. To learn more about packages, see [Understanding packages](#) and [Administering Rational ClearQuest](#).

importintegration

[Overview](#)

[Related Topics](#)

The **importintegration** subcommand allows you to import a partial schema as a modification to an existing schema. Before using this command, you must export the partial schema using [exportintegration](#) subcommand. To import an entire schema into the schema repository, use the [importschemata](#) subcommand.

Syntax:

```
cqload importintegration <login> <password> <schema name> <record type>  
<"integration name"> <integration version> <schema pathname> <"comment or  
description"> <new form >
```

Where:

ClearQuest login

ClearQuest password

schema name

record type

integration name

integration version

schema pathname

form name

Represents:

The ClearQuest login name of the user. This user must have Super User privileges.

The ClearQuest password for the respective user.

The name of the schema associated with the integration. This name is the simple name of a schema that is defined in your schema repository (e.g. TeamTest). This schema will increase by one revision number after running CQLOAD.

The record type in the target schema (e.g. Defect) to which you want to add an integration.

The name you give the integration. It can be any alphanumeric indicator of the integration you are loading.

The version you are loading. It should be numeric.

The pathname of a schema integration file that has been produced by exportintegration. There are a number of integrations delivered out-of-the-box in the <installation-dir>/addin's directory.

The forms to which the new tabs (created by the integration) will be added. If no form updates, type "" to indicate no form update.

Example:

```
cqload importintegration admin "" Testit Defect Email_Integ 1 "c:\program  
files\rational\clearquest 1.1\schema.txt" ""
```

The above example imports the partial schema into the Defect record type of the Testit schema.

exportintegration

[Overview](#)

[Related Topics](#)

The **exportintegration** subcommand, part of the CQLOAD command line utility, exports revisions of a schema that would constitute pieces that could be added to another schema. This is useful for advanced users who want to create an integration for use at different (non-network connected) sites.

Exportintegration differs from the [exportschema](#) subcommand in that it exports partial schemas, not the entire schema. To import the data into another schema, use the [importintegration](#) subcommand.

Syntax:

```
Cqload exportintegration <login> <password> <schema_name> <begin_rev>  
<end_rev> <record_type> <schema pathname>
```

Where:

login

password

schema name

Begin_rev

End_rev

record_type

schema pathname

Represents:

ClearQuest login name of the user. This user must have Super User privileges.

ClearQuest password for the respective user.

name of the schema associated with the integration.

First schema revision whose changes you want to save.

Last schema revision whose changes you want to save.

Record type in the schema whose revisions you are saving.

Pathname of the file that will contain the results of exporting the schema revisions.

Example:

```
cqload exportintegration admin "" Enterprise 5 5 defect c:\temp\  
scriptchanges.txt
```

The above example exports only changes made in version five of the Enterprise schema.

```
cqload exportintegration admin " enterprise 5 8 defect c:\temp\  
newscripts.txt
```

The above example exports changes made in versions five through eight.

importschema

[Overview](#)

[Related Topics](#)

The **importschema** subcommand, part of the CQLOAD command line utility, imports an entire schema from a textual representation and adds it to your schema repository. It can be useful if you want to share schemas with sites that can not access your schema repository or have a different schema repository. Before using **importschema**, you must export the schema using the [exportschema](#) command. To import a partial schema, use the [importintegration](#) subcommand.

Syntax:

```
Cqload importschema <login> <password> <schema pathname>
```

Where:

Represents:

login	The ClearQuest login name of the user. This user must have Super User privileges.
password	The ClearQuest password for the respective user.
schema pathname	The pathname to the file that contains the textual representation of a schema that has been saved by the exportschema subcommand.

Example:

```
Cqload importschema admin "" c:\schema.txt
```

The above example imports the schema whose textual representation was contained in c:\schema.txt into the current schema repository.

Note:

C:\schema.txt was created using the cqload exportschema command. During that process, the name of the exported schema was saved into this file. So when you import this schema, that schema name will be used to create the schema with cqload importschema. If that name is already in use in your schema repository, the import will not work.

exportschema

[Overview](#)

[Related Topics](#)

The **exportschema** subcommand, part of the CQLOAD command line utility, is used to export entire schemas to a text file. This can be used to create files that can be used by importschema.

Syntax:

```
Cqload exportschema <login> <password> <schema name> <schema pathname>
```

Where:

Represents:

ClearQuest login	The ClearQuest login name of the user. The user must have Super User privileges.
ClearQuest password	The ClearQuest password for the respective user.
schema name	The name of the schema in your schema repository that is to be exported to a text file. (e.g. TeamTest).
schema pathname	The text file version of a schema to be saved.

Example:

```
Cqload exportschema admin "" DefectTracking c:\schema.txt
```

The above example exports the contents of the DefectTracking schema to the file c:\schema.txt.

List of actions **defined** in the record type selection in the Rules Controls tab.

List of all action types **used** in the record type they selected in the Rule Control tab.

List of states **defined** in the record type selected. Source and Destination states may be used in combination or separately.

Glossary

A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z

A

[access control](#)

[action](#)

[actiondef](#)

[administrator](#)

[aging chart](#)

[API](#)

[attachment](#)

[Attachment field](#)

[Attachment object](#)

[AttachmentField object](#)

[AttachmentFields object](#)

[Attachments object](#)

B

[bar chart](#)

[base action](#)

[behavior](#)

C

[change-state action](#)

[chart](#)

[checkout/checkin](#)

[COM](#)

[control](#)

[cursor](#)

D

[database](#)

[DatabaseDescription object](#)

[database set](#)

[dependent field](#)

[Designer toolbar](#)

[destination state](#)

[distribution chart](#)

[duplicate](#)

[duplicate action](#)

E

[e-mail rule](#)

[Entity object](#)

[EntityDef object](#)

[expression](#)

F

[field](#)

[fielddef](#)

[FieldInfo object](#)

[filter](#)

[Filter dialog](#)

[form](#)

[formdef](#)

[Form Layout toolbar](#)

G

H

[hook](#)

[hookdef](#)

[history](#)

[History field](#)

[History object](#)

[Histories object](#)

[HistoryField object](#)

[HistoryFields object](#)

[HookChoices object](#)

I

[import action](#)

[initialization hook](#)

J

K

L

[line chart](#)

[Link object](#)

M

[metadata](#)

[modify action](#)

N

[notification hook](#)

O

[operator](#)

P

[package](#)

[parent](#)

[pie chart](#)

[poll interval](#)

[privileges](#)

[property](#)

Q

[query](#)

[QueryDef object](#)

[QueryFilterNode object](#)

R

[record form](#)

[record type](#)

[record type family](#)

[result set](#)

[ResultSet object](#)

S

[schema](#)

[schema designer](#)

[schema repository](#)

[Session object](#)

[source state](#)

[SQL](#)

[state](#)

[statedef](#)

[state record type](#)

[state transition](#)

[state transition matrix](#)

[state type](#)

[stateless record type](#)

[submit action](#)

[submit form](#)

T

[trend chart](#)

U

[UCM](#)

[UNC Pathname](#)

[undo checkout](#)

[unduplicate action](#)

[unique key](#)

[user](#)

[user group](#)

V

[validate](#)

[validation hook](#)

[version](#)

W

[workspace](#)

X

Y

Z

access control

Access control limits the use or modification of actions to designated users. Access for actions is set through the action's Properties dialog and can be open to all users, limited to a specific group, or controlled by a hook. The Behaviors table determines access to fields.

action

Whenever you modify a record, you invoke an action from the Action menu to register the changes. Actions may result in a [state transition](#) or they may simply modify information in the record's fields. In ClearQuest, the Action menu displays only the appropriate legal actions.

ClearQuest allows you to modify a record or to transition a record from one [state](#) to another state. For example, you can transition a record from the open state to the closed state.

In ClearQuest Designer: the ClearQuest [administrator](#) modifies the Actions table and [state transition matrix](#) of each [record type](#) to define the legal actions for records of that type. The definition of each action is stored in the ClearQuest [schema repository](#).

actiondef

ClearQuest's system definition of the properties of an action that can be applied to a record type, such as its name, type (submit, state change, duplicate, unduplicate, modify, import, delete), hook programs that are executed when the action starts and ends. For change_state action types, it also defines the source and destination states.

administrator

The person responsible for setting up schemas and databases at your company.

The ClearQuest administrator uses ClearQuest Designer to create and modify [schemas](#), databases, and forms. In addition, the administrator performs other tasks, such as creating user groups and establishing permissions, maintaining the database and setting up e-mail notification.

aging chart

Aging charts show how many records have been in the selected states for how long. Use aging charts to answers the questions: “How many defects have been open for one week? For two weeks? For three weeks?”

API

Application Programming Interface.

ClearQuest contains a robust interface that administrators can use to customize the behavior of their databases. The API consists of a set of objects, methods, and functions that can be called from hook code to perform tasks such as getting or setting the value of a field.

attachment

ClearQuest allows you to associate a file with a particular record. Attachments are stored in the ClearQuest user database, along with other data contained in the record.

attachment field

An attachment field is a field whose type is ATTACHMENT_LIST. An attachment field stores attached files.

Attachment object

In the ClearQuest Designer API: An Attachment object stores information about a single attached file.

Attachments object

In the ClearQuest Designer API: An Attachments object is a collection for [Attachment objects](#). The overall collection contains the attached files for a single field (represented by an [AttachmentField object](#)).

AttachmentField object

In the ClearQuest Designer API: An AttachmentField object represents the attached files for a single field. This object stores a reference to an [Attachments object](#).

AttachmentFields object

In the ClearQuest Designer API: An AttachmentFields object is a collection object that represents all of the attached files for a given record. This object stores references to one AttachmentField object for every attachment field in the record.

bar chart

A bar chart illustrates comparisons among individual items. Categories are organized horizontally, values vertically, to focus on comparing values. For example, the bar chart “Defects by Engineer” displays the names of the engineers along the horizontal or x axis and the type of defect along the vertical or y axis.

base action

A base action behaves like a global action in that it fires whenever any other action fires. For example, you might write a base action that sends an e-mail notification when specific events occur. When any action fires, the base action also fires. First the base action checks to see what the other action did, and if the action meets the criteria, the base action sends an e-mail notification.

behavior

The behavior of a field defines the access restrictions for the field. The behavior for a given field can be READONLY, OPTIONAL, MANDATORY, or USE_HOOK. To set the behavior for a field, modify the field's entry in the Behaviors table.

change-state action

A change-state action moves a record from one state to another state.

chart

A chart is a graphical representation of a selected set of records, created usually for the purpose of comparing attributes of those records. There are several different kinds of charts including distribution charts, trend charts, and aging charts. Results can be displayed using several different kinds of graphics, including [bar chart](#) and [pie chart](#) graphics.

checkout/checkin

The two-part process that allows you to edit a [schema](#) and add a new version of the schema to the ClearQuest [schema repository](#). You can modify a schema only after you check it out of the schema repository. A [version](#) of the schema can be associated with a database only after you check it in to the schema repository.

A checkout allows you to add fields, record types, forms, [states](#) and [actions](#) to a schema.

A checkin adds a new version of the schema to the ClearQuest schema repository. Once you check a schema into the schema repository, you can make changes to it only by creating a new version of the schema. You can save intermediate changes to a schema, without checking it into the [schema repository](#) and without updating the version of the schema.

COM

COM is the underlying architecture that forms the foundation for higher-level software services, like those provided by OLE. OLE services span various aspects of commonly needed system functionality, including compound documents, custom controls, interapplication scripting, data transfer, and other software interactions.

ClearQuest's API defines ClearQuest-specific COM objects that can be used in hook code.

control

A graphic element such as a text box, list box, button or picture that you place on a form to display data, enter data, perform an action, or make the form easier to read.

cursor

The cursor is a placeholder used while navigating through a result set. The cursor indicates which row is currently being reviewed.

database

In ClearQuest, the term database refers to the user database that contains all user data and a copy of the [schema](#) associated with the database.

DatabaseDescription object

In the ClearQuest Designer API: A DatabaseDescription object contains information about a particular database. You can use this object to get information about the database.

database set

A database set consists of a [schema repository](#) and all of the databases associated with that repository.

dependent field

A dependent field is one whose value is affected by the values in other fields. To set up a dependent field, you must create hooks that set the value of the field when the original field (or fields) changes. Typically, you would modify one of the following hooks: the field default value hook, the field value-changed hook, or the field choice-list hook.

Designer toolbar

In ClearQuest Designer: The Designer toolbar provides easy access to some of the more commonly used menu items.

destination state

When you perform an action that causes a state transition, the Destination State is the state to which the record is sent. The record originates from the [source state](#).

distribution chart

Distribution charts are used to measure how many records fall into defined categories or match the values you indicate.

For example, use a distribution chart to see the current status of a group of records, or see who has been assigned the most/least change requests. Another example is a chart that details which records have the highest priority.

duplicate

In ClearQuest, the term duplicate refers to the nature of a record, [action](#) or field in the [state transition matrix](#).

A record that contains data already recorded in a previous entry is a duplicate. In the case of defect tracking, a duplicate identifies a defect that has previously been reported.

e-mail rule

A set of parameters that determine when e-mail is sent regarding a ClearQuest record, including who is sent the message and what fields of the record are included in the message. ClearQuest administrator can set up e-mail rules with the Email_Rule record type.

ClearQuest client users must have e-mail enabled to ensure that e-mail is sent.

duplicate action

A duplicate action marks a given record as a duplicate of another record.

entity object

In ClearQuest Designer: An entity object is a runtime object that represents a record in the database.

entitydef object

In ClearQuest Designer: An entitydef object is a runtime object that represents the metadata for a record. This metadata describes the structure of the record, including the number of fields, their names, what data types they must contain, the names of the permitted actions and states for this [record type](#), and so on.

expression

Any combination of an operator, values, and field name that evaluate a single value. Filters use expressions to define query criteria.

field

A field represents a singular piece of data in a record. Fields can contain simple data types such as numbers and strings or they can contain more complex information such as references to other fields, dates, or the current state of a record.

fielddef

ClearQuest's system definition of the properties of a field in a record type, such as its name, data type, default value, and choice list. It also defines hooks that are executed for permission control and field value changes

FieldInfo object

In the ClearQuest Designer API: A FieldInfo object contains information about a particular field. You can use this object to obtain the field's value and other attributes.

filter

in ClearQuest, filters are restrictions you place on a query to limit the number of records returned. Typically, a filter specifies which field values are needed to identify the specific records you want to work with. For example, you can set up a filter that limits the query to records submitted after a certain date. Records that were submitted before the given date are not returned in the query results.

Filter dialog

In ClearQuest, use the Filter dialog to edit the criteria of a given filter.

form

A form provides a visual interface for entering data into a new record, for modifying the data in an existing record, or for specifying query information. You can create [record forms](#) or [submit forms](#) for your schema.

formdef

ClearQuest's system definition of the properties of a form in a record type.

Form Layout toolbar

In ClearQuest Designer: The Form Layout toolbar allows you to adjust the alignment and size of controls in a form.

hook

Hooks are entry points, like triggers, for pieces of code that ClearQuest executes at specified times to more fully customize the product.

Actions can have hooks for access control, initialization, notification, committal, and validation. Fields can have hooks for specifying default values, choice lists, and permissions and for handling tasks associated with the field when it is validated or its value changes.

Records can use record scripts that allow you to trigger actions that are specific to a record type. Global scripts are another type of hook. Global scripts allow you write a piece of code, such as an e-mail notification that can be called from any hook, regardless of the record type for which it is defined.

hookdef

ClearQuest's system definition of the properties of a hook, such as its type and language.

history

ClearQuest allows you to track all modifications of each record. The history of a record includes the creation date and each modification made to the record, such as assigning a defect to an engineer, adding details to the description field and resolving a defect.

history field

In ClearQuest Designer: A History field stores information about the actions that have taken place on a record. Every record has an implicit history field associated with it. You cannot create new history fields. You can place a history control on a form to allow the user to view the history of a record.

History object

In the ClearQuest Designer API: A History object stores a text string describing an action that was initiated on a record.

Histories object

In the ClearQuest Designer API: A Histories object is a collection that stores the History objects associated with a single [history field](#).

HistoryField object

In the ClearQuest Designer API: A HistoryField object represents the history entries displayed in a single [history field](#).

HistoryFields object

In the ClearQuest Designer API: A HistoryFields object is a collection that stores all of the history information for a given record. This object stores references to one [HistoryField object](#) for every history field used on the record type's form.

HookChoices object

In the ClearQuest Designer API: A HookChoices object represents the choices stored for a given field.

import action

An import action is used when records are imported from another database. During an import action, the new records are added to the database with only a limited amount of validation. In particular, records are not validated to determine whether or not they could have legally reached their current state.

initialization hook

In ClearQuest Designer: An initialization hook can be associated with an action to initialize the fields of a record to some default values. Because this hook provides access to all the fields of the record, you should use it primarily for complex initialization. Compare with the default value hook for fields.

line chart

A line chart shows trends in data at equal intervals. Generally time elements are displayed along one axis and values displayed along the alternate axis.

Link object

In the ClearQuest Designer API: A Link object represents a link between a duplicate and its original record. You cannot create Link objects directly.

metadata

Metadata is information that describes other information. In ClearQuest, metadata is used to specify the structure of records. Databases use metadata to perform searches.

modify action

A modify action allows users to modify the fields of a record without changing the record's state.

notification hook

In ClearQuest Designer: A notification hook can be associated with an action to send notifications or to trigger other actions. For example, a notification hook can send an e-mail message to a group of people to alert them to changes in a particular record.

Operator

Operators act on field values to create a filter expression. Valid operators are:

- IN** Looks for single or multiple values (that is, several different states).
- EQUAL** Looks for one value (that is, a specific record description or date.)
- CONTAINS** Lets you look for text within a value (that is, words or words that might exist in the records you're looking for).
- IS NULL** Looks for fields that have no value entered. Tip: To look for fields that have any value, select the Not check box with the IS NULL operator.
- BETWEEN** Lets you look for a range of numeric values such as dates.
- GREATER THAN** Lets you look for values greater than the value specified (that is, records entered after a certain date).
- GREATER THAN** Lets you look for values less than the value specified (that is, records entered before a certain date).

original

An original record is a record that has one or more [duplicate](#) records associated with it. ClearQuest updates duplicate records using information in the original record.

Note: An original object can itself be a duplicate of another object.

package

A package contains [metadata](#), such as records, fields, and states, that define specific functionality. Installing a package in a [schema](#) provides a way to quickly add that functionality to the schema, thus eliminating the need to build the functionality from scratch.

parent

A parent record is the original record among two or more duplicate records. All other records are children (or duplicates) of the parent and should draw their state information from the parent.

privileges

Users must be granted privileges to access a database or to access the fields of a record. The ClearQuest administrator defines the privileges for each user using ClearQuest Designer.

pie chart

A pie chart shows the relationship of items to the sum of the items, or as a percentage of the whole. It always displays only one data series and is useful when you want to emphasize a significant element.

poll interval

The poll interval for a database is the amount of time a database waits before checking to see if a user's connection is still valid.

property

Properties are qualities that define the appearance or behavior of a field, record, action or form.

query

A query is a request to the system to return a set of records that match the specified search criteria. Queries use [filters](#) to set up the search criteria.

QueryDef object

In the ClearQuest Designer API: A QueryDef object contains the information for a query, including the fields to display and the search criteria. You must use this object in conjunction with a [ResultSet object](#) to initiate a query.

QueryFilterNode object

In the ClearQuest Designer API: A QueryFilterNode object contains information about the search criteria in a query. This object represents a single condition in the search criteria. Multiple QueryFilterNode objects can be created and grouped to perform complex searches.

record form

A record form is a form that can be used to display the contents of a record or to submit new records. Every [record type](#) must have at least one record form, which ClearQuest displays by default when you query the ClearQuest database. If a record type also has a [submit form](#), ClearQuest uses that form when submitting new records instead of the record form.

record type

A record type is a template that defines the actions, fields, forms, behaviors, and possibly [state](#) information associated with a record. Schemas can contain state record types that move from state to state, and stateless record types that remain static.

record type family

A record type family is a set of record types with common characteristics, grouped under one name. Record type families are created by the ClearQuest Administrator to allow querying across multiple record types. The record types in the family have a common set of fields for querying. You cannot submit a record to a record type family.

result set

A result set contains the data returned from a database search (query). This data is organized into rows and columns where each row represents a single record and each column represents a designated field of the record.

ResultSet object

In the ClearQuest Designer API: A ResultSet object initiates a query and provides methods to allow you to navigate through the search results.

schema

In ClearQuest, the term schema refers to all the attributes associated with a database. This includes field definitions, field behaviors, the state transition table, actions, report formats, and forms.

The ClearQuest administrator creates and modifies schemas in ClearQuest Designer. ClearQuest supports multiple schemas and multiple [versions](#) of each schema. Each version of a schema can be associated with multiple databases.

ClearQuest allows you to update and delete a schema. There must always be at least one schema in the ClearQuest [schema repository](#) .

schema designer

A schema designer has permission to create and modify [schemas](#) in ClearQuest Designer. The schema designer can add record types, define and modify fields, create and modify states and actions, add hooks to the schema, and update existing databases. A schema designer cannot perform [user administrator](#) tasks.

schema repository

The schema repository is a master database that contains all the data associated with existing [schemas](#). No user data is stored in the schema repository.

Session object

In the ClearQuest Designer API: A Session object represents the context in which users access a database. The Session object provides methods to allow the creation and modification of records and queries.

source state

When you perform an action that causes a state transition, the source state is the state from which the record originated. The record is sent to the [destination state](#) .

SQL

SQL stands for Standard Query Language and is a language supported by most databases for specifying queries.

state

The state of a record refers to the record's location in the record lifecycle. The ClearQuest administrator defines the possible states in which a record can exist. For example, a record is usually given the Submit state when it is first entered into the system. From there, it might proceed to the Open state while the defect is being examined, and then to the Fixed state when the defect has been corrected.

state record type

A state record type includes [states](#) and [actions](#) that move the record from one state to another. For example, the Defect record type might begin in the Open state, move to the Review state, and end in the Closed state. The [state transition matrix](#) provides a grid in which to define states and their associated actions.

state transition

A state transition occurs when a record moves from one state to a different state. Actions trigger state transitions based on the rules set up in your system's [state transition matrix](#).

state transition matrix

The state transition matrix defines the rules for moving from one state to another state. For each state, the administrator decides the appropriate set of state transitions for that state and enters them into the matrix.

state type

A state type is a label that is used to define a state's role in your state model.

Some ClearQuest schemas and packages require that each state in your state model be assigned or mapped to a specific state type in order for hooks to work. For example, the UnifiedChangeManagement schema uses state types to invoke certain ClearCase triggers. In addition, ClearQuest's Resolution package uses state types to invoke certain ClearQuest hooks when a record's state is transitioned to one of the "resolved" state type.

statedef

ClearQuest's system definition of properties of a state for record types that use states, such as its name and actions that are involved in changing a record type from one state to another.

stateless record type

A stateless record type remains static and is generally used for reference. For example, you might reference the Project stateless record type in the Defect [state record type](#). The ClearQuest user could create entries for all current projects and then associate the appropriate project with a defect.

submit action

A submit action allows users to create new records in the database.

submit form

A submit form is a specialized type of form that is used only for adding new records to the ClearQuest database. If a submit form is available, it is used instead of the default [record form](#) when adding new records. ClearQuest still uses the record form to display existing records in the database.

trend chart

Trend charts show how many records were transitioned into the selected states by day, week or month. In other words, they show you the rate at which new records are being submitted, resolved or moved into other states.

UNC Pathname

A Uniform Naming Convention pathname allows you to fully specify the location of a file. A UNC Pathname includes the host machine and directory information and is of the format:

```
\\machine_name\directory\file.ext
```

undo checkout

Cancels a [schema checkout](#). ClearQuest cancels all edits to a schema and reverts to the previously saved [version](#) of the schema when you undo a checkout. You can save intermediate changes to a schema, without checking it into the [schema repository](#) or updating the version of the schema. Once you check a schema into the schema repository, you can only make changes to it by creating a new version of the schema.

unduplicate action

An unduplicate action removes the mark from a record that identifies it as a duplicate of another record.

unique key

The database needs to know which column or combination of columns always has a unique value. For record types that contain states, the unique key is the ID. For stateless record types, the administrator must assign a unique key. For example, in a project table, the Project Name could be the unique key. In the case that there are multiple versions of the project, the Project Name and the Version can be the unique key.

user

A ClearQuest user is someone who submits records to a database using ClearQuest or who modifies existing records in a database. Users can also create their own custom forms to use when creating queries but cannot modify the public forms provided with the database. Compare with [administrator](#).

user administrator

A schema designer has permission to create and modify [schemas](#) in ClearQuest Designer. The schema designer can add record types, define and modify fields, create and modify states and actions, add hooks to the schema, and update existing databases. A [schema designer](#) cannot perform User Administrator tasks.

user group

A user group is a list of users with similar privileges and access permissions. ClearQuest uses user groups to limit access to certain actions. When access to an action is limited to a user group, only members of that group may perform the action.

validate

ClearQuest stores all schemas in the schema repository. Before checking in changes to a schema ClearQuest validates all changes, verifying that field types and behavior are valid. Some of the tests ClearQuest performs during the validation process are:

- Validates that you have entered unique labels and names for fields and actions.
- Validates that you have assigned a type to each field and a behavior for each state of each field.
- Validates that you have supplied a reference_to record type for each reference field.
- Validates that you have defined a source state and a destination state for all state transitions.
- Validates that you have defined a [unique key](#) for all stateless record types.

validation hook

A validation hook verifies that the fields in a record do not contain illegal values. Validation hooks can be associated with fields to verify the contents of the field immediately or with actions to verify the fields in an entire record.

version

ClearQuest allows you to modify or update a [schema](#). Each time that you checkout a schema, ClearQuest creates a new version, or revision, of the schema. ClearQuest stores each version of the schema in the [schema repository](#). You can associate any version of a schema to a database.

ClearQuest allows you to delete either the last version of the schema or the entire schema.

Workspace

The Workspace displays the currently available elements in the left pane of the ClearQuest component. Elements in the Workspace are displayed as a series of navigable folders that can be expanded and collapsed as needed.

In ClearQuest, the Workspace displays your personal and system queries, charts, and reports.

In ClearQuest Designer, the Workspace displays the elements of the currently selected [schema](#). Schema elements include field and behavior tables, [states](#) and the [state transition matrix](#), forms and stateless [record types](#), such as user and project tables.

UnifiedChangeManagement schema (UCM)

A ClearQuest schema that supports the UCM process by providing integration with Rational ClearCase.

