

AppBuilder
By Magic Software Enterprises

Magic Software AppBuilder

Version 3.2

Repository Administration Guide

Corporate Headquarters:

Magic Software Enterprises
5 Haplada Street,
Or Yehuda 60218, Israel
Tel +972 3 5389213
Fax +972 3 5389333

© 1992-2013 AppBuilder Solutions

All rights reserved.

Printed in the United States of America.

AppBuilder is a trademark of AppBuilder Solutions. All other product and company names mentioned herein are for identification purposes only and are the property of, and may be trademarks of, their respective owners.

Portions of this product may be covered by U.S. Patent Numbers 5,295,222 and 5,495,610 and various other non-U.S. patents.

The software supplied with this document is the property of AppBuilder Solutions and is furnished under a license agreement. Neither the software nor this document may be copied or transferred by any means, electronic or mechanical, except as provided in the licensing agreement.

AppBuilder Solutions has made every effort to ensure that the information contained in this document is accurate; however, there are no representations or warranties regarding this information, including warranties of merchantability or fitness for a particular purpose. AppBuilder Solutions assumes no responsibility for errors or omissions that may occur in this document. The information in this document is subject to change without prior notice and does not represent a commitment by AppBuilder Solutions or its representatives.

1. Repository Administration Guide	2
1.1 Introduction to Repository Administration	2
1.2 Installing the AppBuilder Repository Software	12
1.3 Managing Security	14
1.4 Managing a Repository	30
1.4.1 Working with the Service Control	30
1.4.2 Creating a Repository	33
1.4.3 Creating a Repository Alias	52
1.4.4 Deleting a Repository	53
1.4.5 Deleting a Repository Alias	55
1.4.6 Allowing Local Users to Connect to a Personal Repository	56
1.4.7 Full Repository Migration	57
1.4.8 Repository Replicator	61
1.4.9 Packing and Reindexing a Repository	69
1.4.10 Reorganizing a Repository	71
1.4.11 Reviewing the Log File	73
1.4.12 Broadcasting Messages	74
1.5 Using a Repository	75
1.5.1 Connecting to and Disconnecting from a Repository	75
1.5.2 Committing Session Changes	77
1.5.3 Querying Repository Objects	78
1.5.4 Printing Objects	78
1.5.5 Creating and Locking Repository Objects	82
1.5.6 Using the Object Browser	88
1.6 Tuning for Optimal Performance	92
1.7 Workgroup Repository Rebuild	97
1.8 Moving Data Between PC Repositories	109
1.8.1 Migration Overview	109
1.8.2 Understanding Migration Phases	110
1.8.3 Migration Export for Repository Objects	110
1.8.4 Performing a Migration Import	126
1.8.5 Using Migration Results	141
1.8.6 Deleting a Migration Object	141
1.9 Migrating between PC and Enterprise repositories	142
1.9.1 How Automated Migrations Work	142
1.9.2 Managing the Migration Server	143
1.9.3 Setting the Codepage for SBADATACONN	145
1.9.4 Configuring Mainframe Repository Properties	146
1.9.5 Specifying Upload, Download and Working Directories	147
1.9.6 Downloading Objects	149
1.9.7 Uploading Objects	164
1.9.8 Performing UOW Migrations	167
1.9.9 Setting Server Security Requirements	168
1.9.10 Understanding Performance Implications	168
1.9.11 Understanding Migration Restrictions	170
1.9.12 Renaming PC Migration Files	170
1.9.13 Understanding Mainframe Migration and Repository Security	170

Repository Administration Guide

Introduction to Repository Administration

This guide provides information about the AppBuilder Workgroup and Personal Repositories and how best to use and administer them in your development environment. A Workgroup Repository was previously referred to as a *Freeway Repository or GRE*, and a Personal Repository was referred to as a *Local Repository or LRE*. In both cases, the terms are interchangeable.

The cornerstone of AppBuilder is its repository technology. The repository gives AppBuilder its power. Technically, the repository is made up of a relational database and a set of files that store source code. It stores everything concerning the development of an application such as analysis models and diagrams, variable definitions, window designs, application code, and deployment options.

More importantly, a repository stores the inter-dependencies between any of these objects. The inter-dependencies are referred to as *relationships*.

Every source object created using AppBuilder is stored in the repository. When objects interact, relationships are formed and stored.

A repository stores information in two places: All objects and relationships are stored in the database. All files are stored external to the database in a directory. References to these files are stored within the database.

The following topics are discussed in this introduction:

- [What is a Personal Repository?](#)
- [What is a Workgroup Repository?](#)
- [Possible Repository Configurations](#)
- [Repository Administration Tool Overview](#)

There are three types of repositories available in AppBuilder. This guide covers administration of Workgroup and Personal Repositories. For information on the Enterprise Repository, refer to the *Enterprise Administration Guide*.

AppBuilder Repository Types

Workgroup Repository	A controlled, secure client/server development environment where the repository is accessible on a network server from remote workstations.
Personal Repository	Repository functionality on a client workstation that is configured to be easily uploaded and merged with a larger development effort. A personal repository (also known as a local repository) is a repository that only allows local client connections; consequently, it only supports development from a single user at a time. When developers need to share objects with one another, they need to migrate the objects to a Workgroup Repository or the central Enterprise Repository.
Enterprise Repository	The Enterprise repository is an MVS-based repository used for large scale development environments. The mainframe is designated as the central archive and site for a team of developers to conduct application development. When using an Enterprise repository, AppBuilder comes supplied with an upload / download process.

What is a Personal Repository?

A Personal Repository is a database repository where the client, server, and database all reside on the same local machine. The AppBuilder Personal Repository is also referred to as a local repository.

A Personal Repository supports multiple local users on a single machine. In addition, Workgroup and Enterprise Repositories, support multiple users simultaneously. Multiple Personal Repositories can be created on a single machine, each one defined to allow the same or different users access to the repository.

Benefits of a Personal Repository

The objective of the Personal Repository is to establish a repository locally where you can work independently, without a continuous network connection to a remote server. Organizations that use an Enterprise Repository will use Personal Repositories connecting to the Enterprise Repository for upload and download. Only during an upload, download, or query of remote objects is a connection required to the remote server. For detailed instructions for uploading and downloading objects, see [Migrating Between PC and Enterprise Repositories](#).

What is a Workgroup Repository?

A Workgroup Repository resides on a Windows server machine and is designed for use by multiple developers, how many, depends on the specification of the machine used. There is only one Workgroup Repository per dedicated machine. The benefit of the Workgroup Repository is that developers share information in a real-time environment, promoting a truly collaborative development model. A Workgroup Repository includes sophisticated locking mechanisms that insure only one developer changes an object at a time.



Do not install a Workgroup and Personal Repository on the same machine. If Personal Repositories are installed and then a Workgroup is installed, all Personal Repository files in the directory are written over by the Workgroup Repository files.

Benefits of a Workgroup Repository

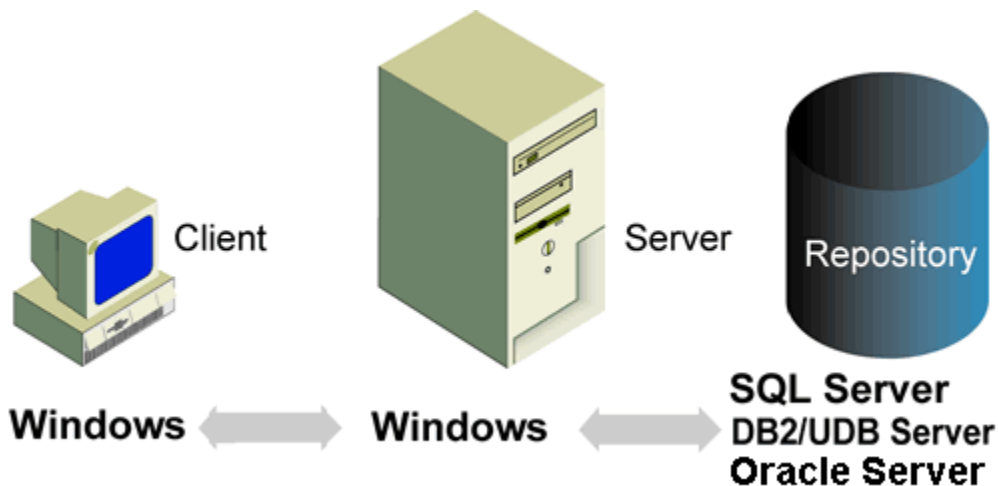
The AppBuilder Workgroup Repository and its management tools allow concurrent, object-based application development. A Workgroup Repository allows multiple developers to access objects in the repository in a real-time manner. This means that developers have immediate access to objects created or modified by other developers, and this avoids the latency inherent in the Personal Repository / Enterprise Repository model. The ability to immediately share objects between developers makes for more object reuse and therefore faster application development. Workgroup Repository technology supports:

- A flexible, scalable architecture
- An object-oriented information model
- Data migration between repositories
- Conference-based notification
- Industry standard APIs
- National language support (NLS)
- Rebuild technology.

Possible Repository Configurations

AppBuilder repository management software can be installed and configured to run on numerous platforms.

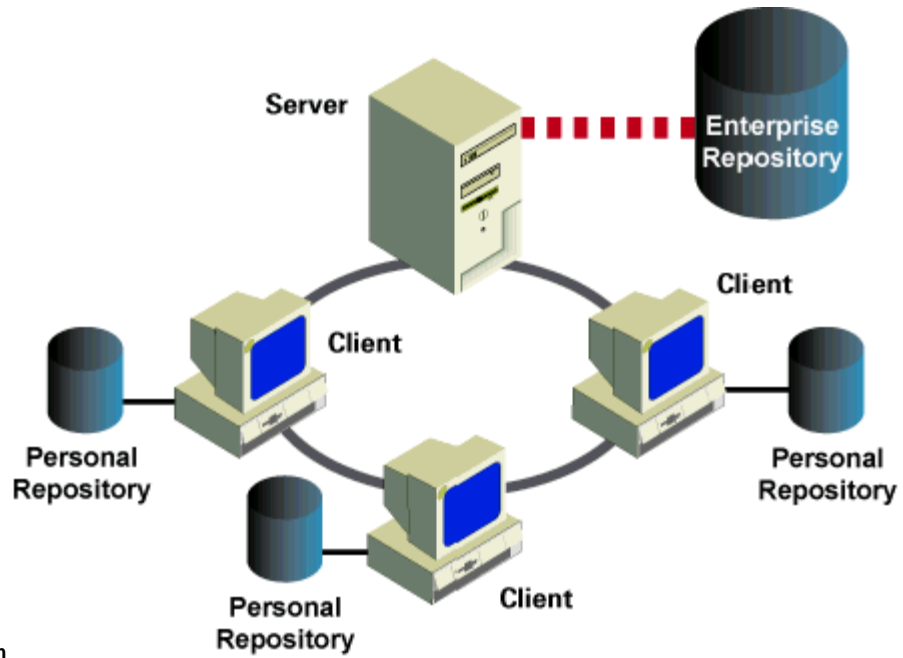
Supported AppBuilder platforms



[Supported AppBuilder platforms](#) illustrates a typical AppBuilder network configuration.

Personal Repository Network Diagram

In this scenario multiple clients each with one or more Personal Repositories are connected to an Enterprise Repository. Transfer of files is done through an automatic Upload/Download facility.

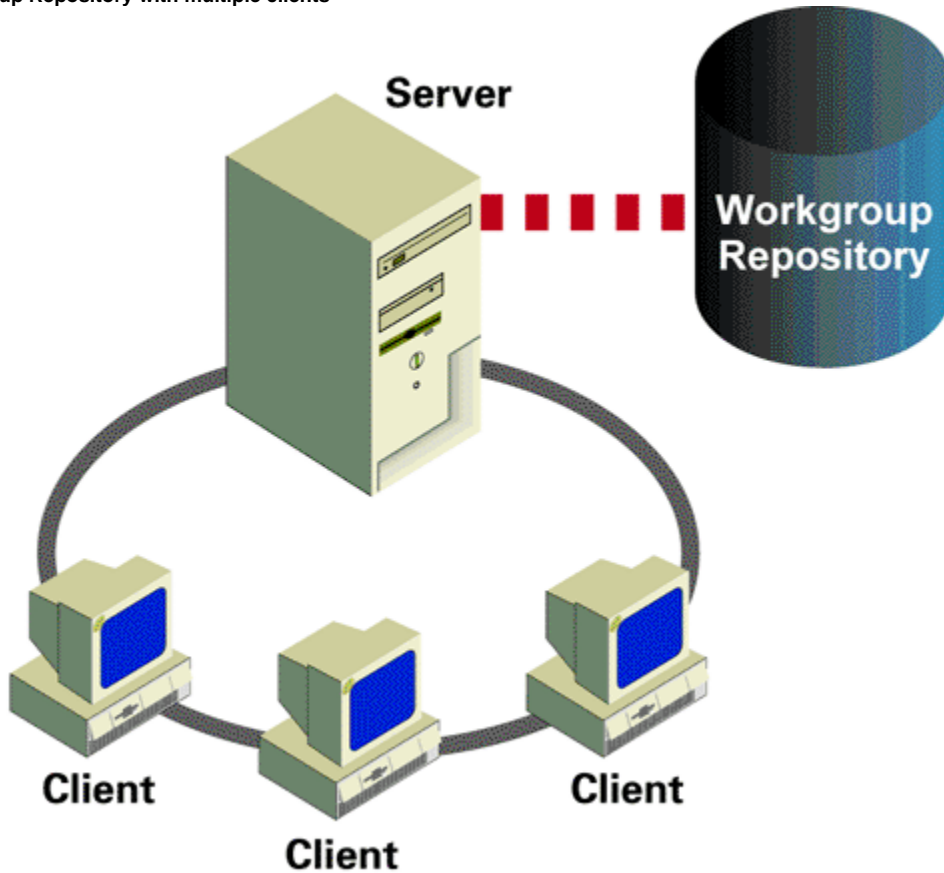


Personal Repository Network Diagram

Single Repository with Multiple Clients

The basic configuration in a distributed environment includes one or more client machines communicating with one server machine, as shown in [Single Workgroup Repository with multiple clients](#). A single repository can hold the data of any number of projects under development.

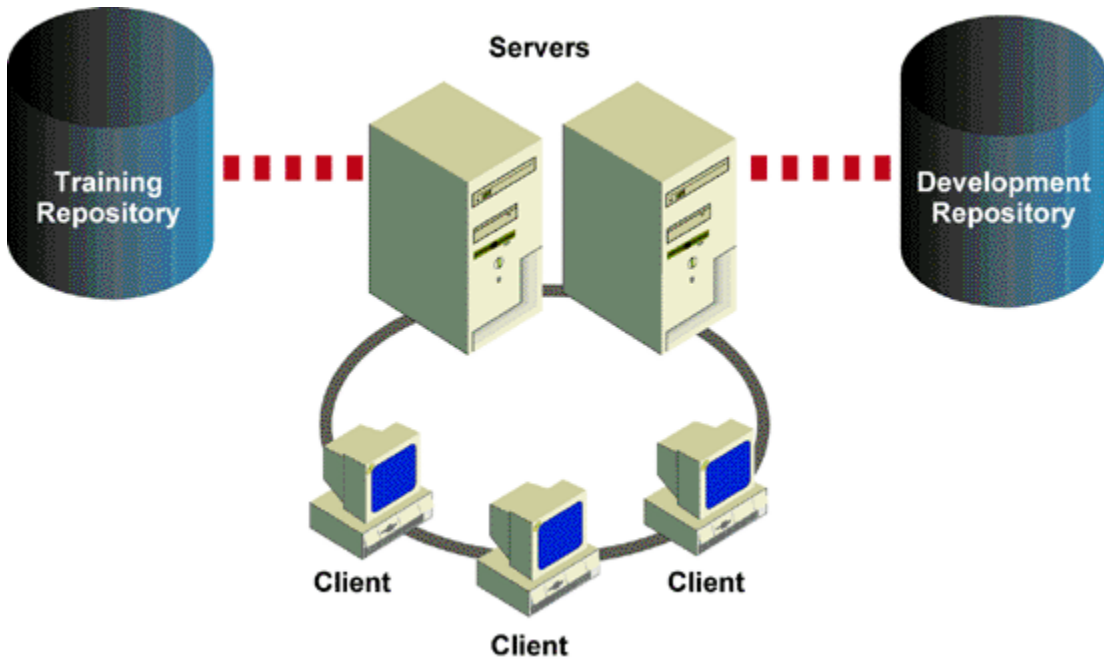
Single Workgroup Repository with multiple clients



Multiple Workgroup Repositories on Multiple Servers

More than one repository can be used. Clients with access to each server can possibly have access to each repository as shown in [Multiple](#)

Multiple Workgroup Repositories with multiple clients



Repository Administration Tool Overview

The focus of this section is how to administer your repository using the Repository Administration tool. Regardless of whether you are working with a Personal Repository or a Workgroup Repository, if you have administrative authority on the repository, you will manage the repository using the Repository Administration tool.

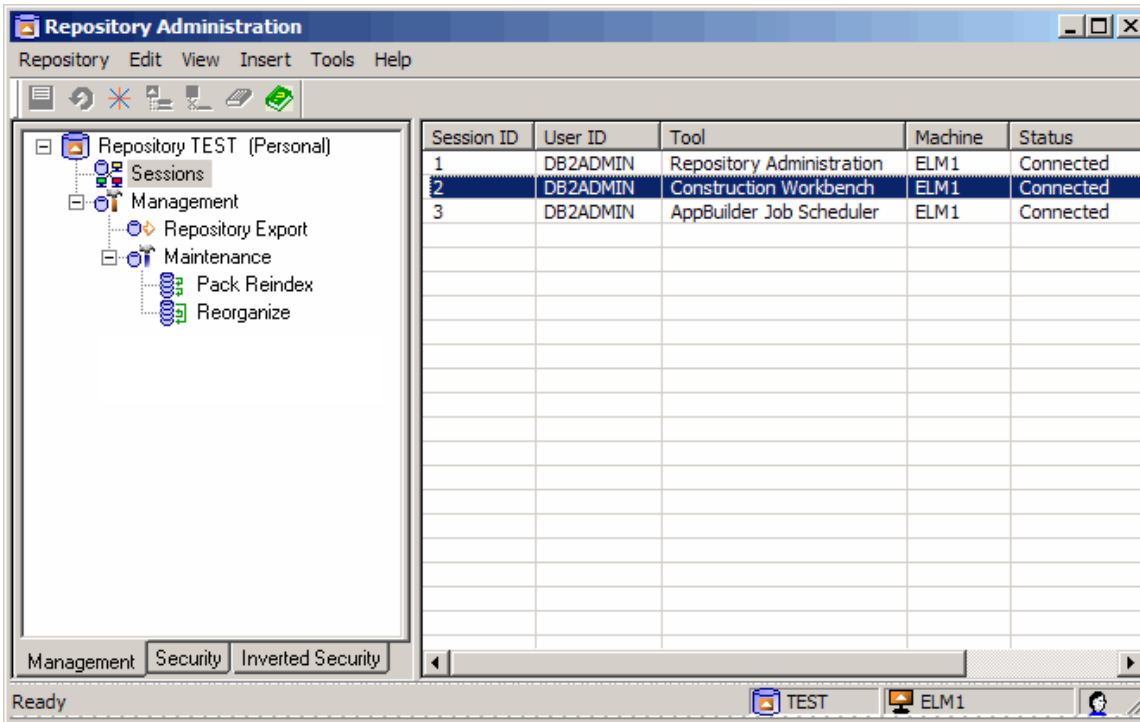
Use the Repository Administration tool to maintain repository security - creating new users, groups, and projects, and creating the relationships between them. Only users with system administrator authority can access the Repository Administration tool.


This section explains the Repository Administration tool's user interface. For information about using the Repository Administration tool, refer to the following topics:


- For information about setting up users, groups, and projects, refer to [Managing Security](#).
- For information about migration functionality, refer to [Moving Data Between PC Repositories](#) and [Migrating Between PC and Enterprise Repositories](#).

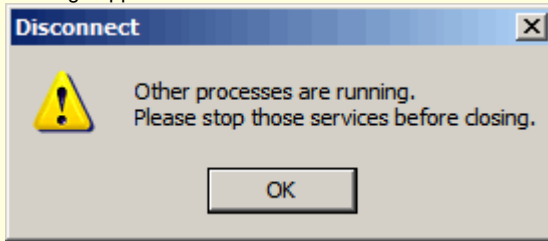
Like the Construction Workbench, the Repository Administration tool uses a hierarchy with tabs that offer different views of the repository information. This hierarchy window offers a hierarchical display of your repository administration information, whether it is repository management tasks under the *Management* tab, or project security information under the *Security* tab. There is also an *Inverted Security* tab that displays your repository security objects in a reverse hierarchy, so you can locate a user and see the groups and projects with which that user is associated.

Repository Administration tool



 The icons on the status bar indicate the repository alias, the machine to which you are connected, and the user ID that is connected.


 If you want to close the main Repository Administration Tool window while doing migrations, uploads or downloads, a warning message appears:



You must close the running processes before disconnecting the main Repository Administration Tool window.

Management Tab

When you first access the Repository Administration tool the Management tab displays. The Management tab offers tasks you can perform on the repository itself, including processes attached to the repository, a full repository migration, and pack and reindex utilities.

 Management functions are not available when you are connected to a remote machine.

When the Sessions object is highlighted in the Management tree, the status of the processes attached to that repository are displayed on the right pane. To display Session Details information for an uncommitted session, double-click on the session or right-click on the session and select *Details*. The system displays a dialog with the modified tables listed (see [Session Details dialog displaying tables for the selected uncommitted session](#)).

Status Report

The top right pane displays information about connected users, sessions, and tools currently active on the selected machine. This information is especially useful when working with a server machine.

Status Report pane

Session ID	User ID	Tool	Machine	Status
1	DB2ADMIN	Repository Administration	ELM1	Connected
2	DB2ADMIN	Construction Workbench	ELM1	Connected
3	DB2ADMIN	AppBuilder Job Scheduler	ELM1	Connected

The Status Report pane displays users connected to the Workgroup Repository. Use this information to see who is connected to the server before you stop the Repository Service. Stopping the service when users are connected results in users losing uncommitted work. Send a broadcast message to the connected users indicating that you need to stop the service and that all users should save their work and disconnect. Refer to [Broadcasting Messages](#) for steps on how to send a broadcast message.

In [Status Report pane](#), notice that user DB2ADMIN has a session for the Repository Administration tool, one for AppBuilder Job Scheduler, and one for Construction Workbench. The status for each of the sessions is also displayed. The Construction Workbench session status is Uncommitted because the user has modified or created objects in the repository without committing the changes. The following table outlines the purpose of each column in the Status Report pane.

Status Report pane

Column	Description
Session ID	Identifies the connected session
User ID	The user ID of the user connected to the session
Tool	Name of the tool that created the repository session
Machine	Name of the client machine where the tool is running
Status	Current status of the session: Connected or Uncommitted. <ul style="list-style-type: none"> • Connected means user is connected and has no uncommitted work • Uncommitted means user is connected and has updated or added repository objects that have not yet been committed to the repository. Those changes will be lost if the session is disconnected.

Refreshing the Status Report Pane

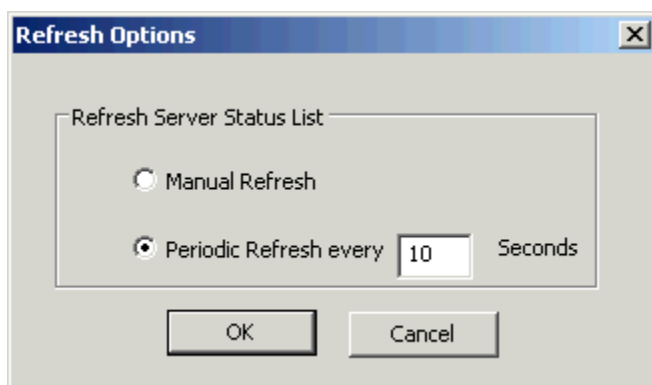
You can manually refresh the information on the Status Report Pane or set a periodic refresh. To do this:

Right-click on the *Sessions* option from the Management tab and select *Refresh Now* or *Refresh Options...*

Now you have the following options:

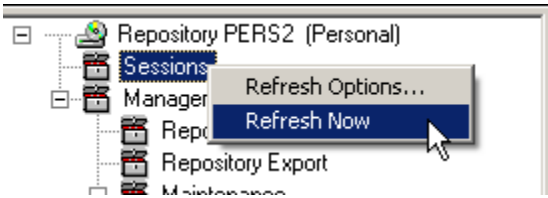
- When you select *Refresh Now*, the information is refreshed on the Status Report Pane.
- When you select *Refresh Options...*, the Refresh Options dialog displays allowing you to set up a periodic refresh or set the Manual Refresh option.

Refresh Options dialog



If you select manual refresh from this dialog, the Status Report window will not refresh until you right-click *Sessions* and select *Refresh Now*.

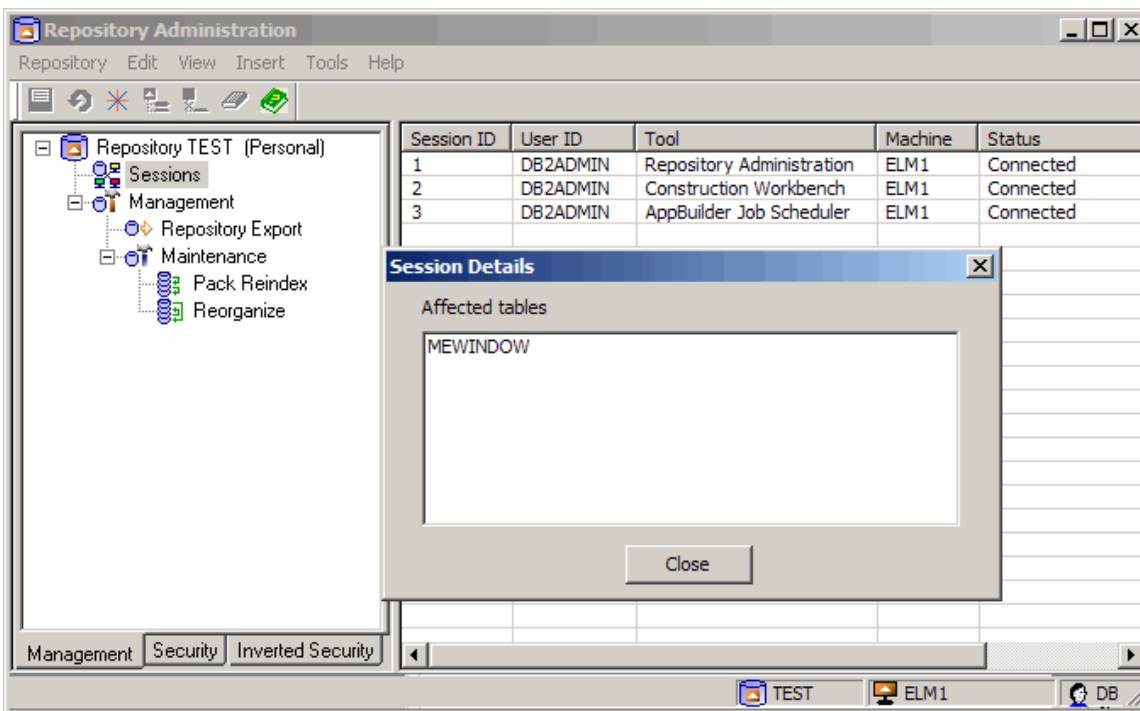
Manual Refresh



Status Details Dialog

Table information displayed on the Session Details dialog pertains to repository tables for the current uncommitted session. This dialog lists the repository tables that are holding object locks. In the following figure we can see only MEWINDOW, the table that is holding the object locks. To display this dialog, double-click the uncommitted session or right-click and select *Details*. The affected tables are displayed in the Session Details dialog.

Session Details dialog displaying tables for the selected uncommitted session



Only information on uncommitted (locked) objects displays in the Session Details dialog. If a user is connected to the repository but has no locked objects, the Details option is unavailable.



To stop the Service, or view session details on a remote machine, the Windows user account *must* be a member of the administrator group on the remote machine. The Repository Administration tool automatically restarts the Service whenever you connect to a local repository.

Security Tab

The Security tab displays the security model with the repository as the root object and Project, Group, and User objects the children of the root. The hierarchy of the objects displayed in the Security tab is as follows: Project, Group, and User. You can view the Projects and display which Groups and Users are associated with that project. The User is the leaf object in this tree. Functionality from the Edit, View, and Insert menus is enabled when you select the Security tab. For more information on the security model, refer to [Managing Security](#).

Inverted Security Tab

The Inverted Security tab is intended to display the security model in an inverted position, with the User object on top. From the Inverted Security tab, you can view the Users and identify which Groups and Projects with which a selected User is associated. Much of the functionality available

from the Security tab is also available from the Inverted Security tab.

Repository Administration Toolbar

The toolbar across the top of the screen offers an easy way to accomplish many of your administration tasks.

Repository Administration Toolbar

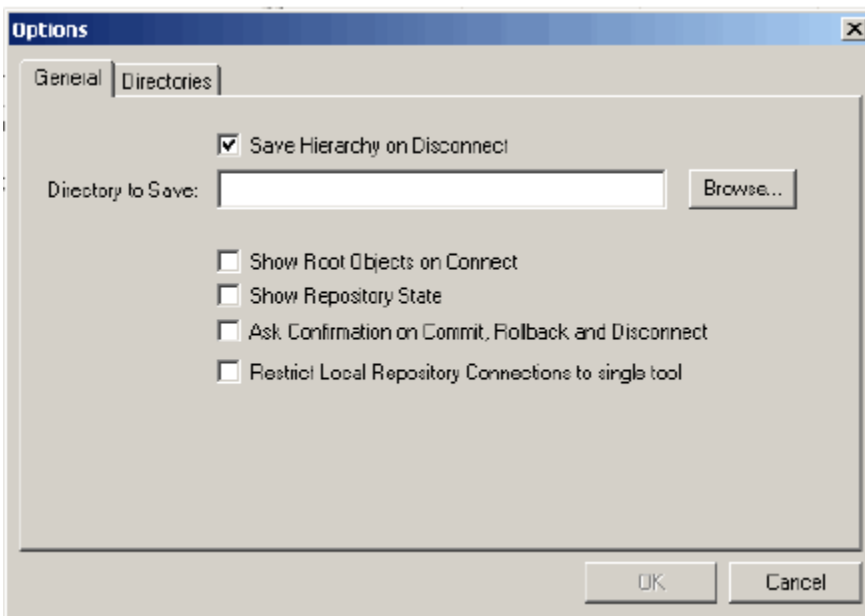


Repository Administration Toolbar descriptions

Number	Toolbar Description
1	Commit changes to the repository.
2	Rollback changes from the repository.
3	Show repository states. (Toggle option)
4	Show root objects in the hierarchy only. (Toggle option)
5	Show orphan objects in the hierarchy only. (Toggle option)
6	Clear all objects from the hierarchy tree.
7	Displays online documentation for AppBuilder.

The tool options that determine how the tool behaves each time you reconnect are set from the Options window. To access these options, click *Tools > Options* . The Options window displays.

Tools Options window



Tool Options General Tab

Option	Description
Save Hierarchy on Disconnect	Saves your hierarchy view and redisplay it on your next connection.
Directory to Save	Directory where the hierarchy view information is saved.
Show Root Objects on Connect	Displays root objects of the hierarchy on connect.
Show Repository State	Displays the type of lock placed on a repository object. Each lock type displays a different icon from the Security tab in the Hierarchy Diagram.

Ask Confirmation on Commit, Rollback, and Disconnect.	System prompts you for confirmation each time you commit or rollback an object, or disconnect.
Restrict Local Repository Connections to a Single Tool	Checking this option prevents other session connections to the repository when Upload/Download is in progress.

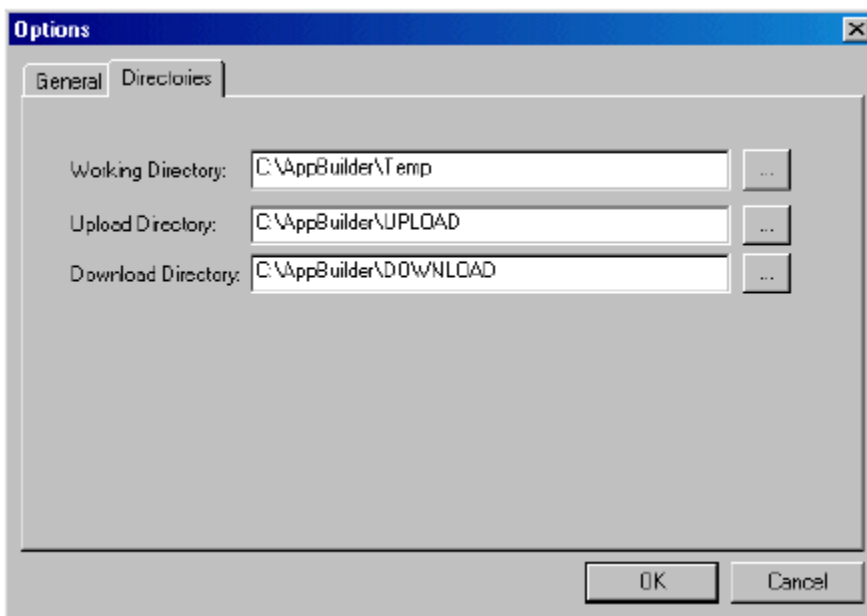
Select the Directories tab to set the upload, download, and working directory. The default working directory is C:\AppBuilder\Temp. The upload and download directories are only used with a Personal Repository. If you have a Workgroup Repository installed, these fields are blank.

Tools Options Directories Tab

Working Directory	A temporary directory for storing intermediate files for repository administration
Upload Directory	A directory to temporarily store files during the upload
Download Directory	A directory to temporarily store files during the download

Refer to [Specifying Upload, Download and Working Directories](#) for more information on these directories.

Directories tab



Showing Root Objects

Click the Show Root Objects icon on the toolbar to display or hide all root objects.



It is a toggle option. Root objects can be root projects from the Security tab or root users from the Inverted Security tab.



Groups are not loaded by default when selecting Show Root Objects. Only when you select Show Orphans while in the Security tab will you be able to see all groups that are not in a project.

Show Root Objects is an easy way to populate the hierarchy. This is only a view option. The repository is not affected by objects being cleared and shown in the hierarchy. If the objects are already visible in the hierarchy, Show Root Objects has no affect.

Showing Orphan Objects

Click the Show Orphan Objects icon on the toolbar to display or hide all objects that have no parents.



This is a toggle option. Show Orphan Objects is an easy way to populate the hierarchy. This is only a view option. The repository is not affected by objects being cleared and shown in the hierarchy.

Displaying Leaf Objects

Double-clicking on an object that has children displays the leaf object information for that hierarchy. When you double-click on a group, the system displays users for that group. When you double-click on a project, the system displays users associated with that project. The user is the lowest level object or *leaf object* in this scenario.

Similarly, from the Inverted Security tab, when you double-click on a user, the system displays the projects with which the user is associated. When you double-click on a group, the system displays the projects with which the group is associated. In this scenario, the project is the leaf object.

Using Cut, Copy, and Paste in the Hierarchy

Use Cut, Copy, and Paste on repository objects to move objects from one location in the hierarchy to another. This functionality is supported from the Security tab and the Inverted Security tab. From the Security tab, you can cut or copy users, groups, and projects, but you cannot paste a project into the Security tab. Paste projects only into the Inverted Security tab. The same is true for users in the Inverted Security tab. You can cut or copy users, groups, and projects, but you cannot paste a user in the Inverted Security tab. Paste users in the Security tab.

A cut effectively deletes the relationship between two objects and the Paste creates a new relationship in the new location. When you cut an object, before you paste, the item is greyed to indicate that it has been cut. If the Cut operation is cancelled before the Paste, the object remains in its original location. Only a single item can be sent to the clipboard at a time, but when a parent object is cut or copied, its children objects are also cut or copied.

Access Cut, Copy, and Paste from the right-click menu after you have selected an object, or access it from the Edit menu.

Menu Options

This section outlines and describes the options available on the Repository Administration tool menus.

Repository Menu

The available options in the Repository menu are:

- New repository or alias
- Delete a repository or alias
- [Repository Replicator](#)
- Start and Stop the repository service
- Connect or disconnect from the repository
- Commit changes to the repository, or rollback changes before committing to the repository
- Session Properties. Refer to [Understanding Session Properties](#)
- Print Setup
- Exit the Repository Administration tool

Edit Menu

You can cut copy and paste repository objects from the Repository Administration tool hierarchy. Refer to [Using Cut, Copy, and Paste in the Hierarchy](#) for information about this functionality.

- Cut (Ctrl+X)
- Copy (Ctrl+C)
- Paste (Ctrl+V)
- Delete Object (Ctrl+Delete). Deletes the object from the repository.
- Delete Relationship (Ctrl+Alt). Deletes the relationship between objects from the repository. Relationships exist from child objects to parent objects; therefore, a root object does not have a relationship. This option is unavailable when selecting a root object.
- Clear the Object (Delete). Clears the selected object from the hierarchy only. The object remains in the repository. This option is available for root objects only. When you select a child object, this option is unavailable.
- Repository Alias. Displays the Repository Alias window where you can select an alias and edit its parameters.
- Host Properties. Displays the Host Properties window where you can edit your host connection properties. This option is only available when you are connected to a Personal Repository.
- Properties. View and edit the properties for the selected object.
- Relationship Properties. View and edit the relationship properties for the selected object.

View Menu

The available options in the View menu are:

- Toolbar - Display or hide the toolbar.
- Status Bar - Display or hide the status bar.
- Repository States - Show repository states, that is object changes that have not been committed to the repository. An uncommitted object is denoted by an icon if this option is checked.
- Show Orphans - Display objects that have no parent objects.
- Show Roots - Show only the root objects. Do this when you want to limit the detail of your hierarchy.
- Clear all Objects - Clears all objects from the hierarchy without removing them from the repository. Objects can be displayed again by selecting another View option.
- Inverted Hierarchy - Displays the users and shows the groups and projects to which a user belongs.

- Expand One Level - Displays the next level of child objects for the selected object.
- Expand All - Expands the entire tree for the selected object showing all child objects that come from the selected parent object.
- Log - Displays the most recent log files.

Insert Menu

The options on this menu change depending on the tab you have selected and the object you have selected in the hierarchy.

- From the Security tab and the Inverted Security tab, you can insert Projects, Groups, and Users; however, you cannot insert anything when the leaf object is highlighted. The User object is the leaf of the Security tab and the Project is the leaf of the Inverted Security tab. Consider the following rules:

From the Security tab:

- With the root object highlighted, you can insert Projects, Groups, and Users.
- With the Project highlighted, you can insert Groups.
- With the Group highlighted, you can insert Groups and Users.
- With the User highlighted, the insert option is disabled.

From the Inverted Security tab:

- With the root object highlighted, you can insert Projects, Groups, and Users.
- With the User highlighted, you can insert a Group.
- With the Group highlighted, you can insert Groups and Projects.
- With the Project highlighted, the insert option is disabled.

Tools Menu

The available options in the Tools menu are:

- Migration Export allows you to create or open a migration object. You can also use the Migration Export action to export the contents of the Unit of Work or to export to XML. Refer to [Migration Export for Repository Objects](#) for more information.
- Migration Import allows you to import an existing migration object. You can also import from XML. Refer to [Performing a Migration Import](#).



The regular Migration Import and Export will not work for AppBuilder OO objects. For the OO objects, you should use the XML Import and Export [Tools > Migration Import or Export > XML].

- Repository Export allows you to export the entire contents of the current repository to .exp file. See [Using Repository Export](#).
- Pack Reindex allows you to clean up a repository that has been used for extended periods of time. See [Packing and Reindexing a Repository](#).
- Reorganize provides the means to eliminate unnecessary space in database tables. See [Reorganizing a Repository](#).
- Upload and Download when a Personal Repository is installed.
- Unit of Work - Opens the Unit of Work window.
- Query Content - Allows you to query the content of the selected repository.
- Broadcast Message - Allows you to broadcast a message to connected users. Also available from the right-click menu of the Management tab when you select the repository.
- Options - Set default options and directories for tool behavior.

Help Menu

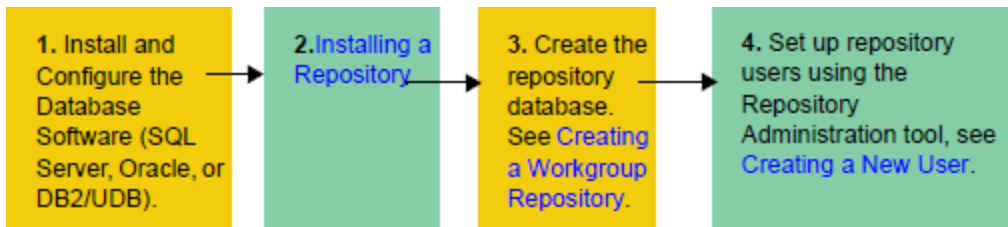
The available options in the Help menu are:

- Connects you to the Magic Software Enterprises web site.
- Connects you to the Customer Service web site.
- Displays information for the installed version of AppBuilder.

Installing the AppBuilder Repository Software

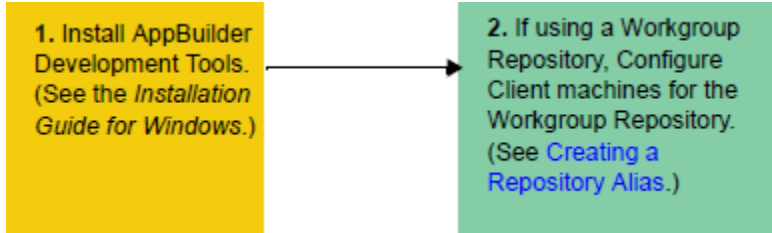
This section discusses the information you need to install and set up a Personal or Workgroup Repository environment. [AppBuilder repository administrator steps](#) outlines the tasks for which a repository administrator may be responsible. See your DBMS software documentation to complete Step 1 .

AppBuilder repository administrator steps



The AppBuilder installation process involves specifying the kind of repository to be installed on the machine ([Selecting the appropriate repository to install](#)). The installation process does not perform the repository installation. The repository installation is performed via the Repository Administration Tool. See [Creating a Personal Repository](#) or [Creating a Workgroup Repository](#). [AppBuilder developer steps](#) outlines the tasks for which an AppBuilder developer is responsible.

AppBuilder developer steps



For information about the system requirements and supported platforms, refer to the AppBuilder *Installation Guide for Windows* .

Installing a Repository

You select the appropriate repository when installing the AppBuilder software. InstallShield takes you through the installation process. At the Select Features screen, select *Personal Repository* to install a local repository on your machine; select *Workgroup Repository* to install a workgroup repository.

A Workgroup Repository and a Personal Repository cannot be installed on the same machine. Multiple Personal Repositories can be installed on one machine, but only one Workgroup Repository can be installed on a single machine.

Selecting the appropriate repository to install

In the options list, select the items you want to install, clear the items you want to remove. All selected products will be returned to the AppBuilder 3.0 base level.

<ul style="list-style-type: none"> <input checked="" type="checkbox"/> Construction Workbench <ul style="list-style-type: none"> <input checked="" type="checkbox"/> MLUI <input checked="" type="checkbox"/> Personal Repository <input checked="" type="checkbox"/> TurboCycler Developers Kit <input checked="" type="checkbox"/> Open Generation <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Java <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Java Reports <input checked="" type="checkbox"/> SOAP Web Services <input checked="" type="checkbox"/> EJB <input checked="" type="checkbox"/> HTML Client <input checked="" type="checkbox"/> Java Client <input checked="" type="checkbox"/> Microsoft .NET <ul style="list-style-type: none"> <input checked="" type="checkbox"/> Microsoft .NET Client <input checked="" type="checkbox"/> Microsoft .NET SOAP Web Services <input type="checkbox"/> Workgroup Repository <input type="checkbox"/> Repository Replicator 	<p>Description</p> <p>The Construction Workbench is the primary tool for constructing, preparing, testing, and packaging AppBuilder applications.</p>
--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------

0.00 MB of space required on the C drive
22454.78 MB of space available on the C drive

The installation process only selects which kind of repository to install. Because each kind of repository supports multiple types of DBMS, you must proceed to create the repository so that AppBuilder will open and function. See [Creating a Personal Repository](#) and [Creating a Workgroup Repository](#) for full information on creating an AppBuilder repository.

Managing Security

The Workgroup Repository system administrator sets security levels by assigning permissions that define access and actions that are allowable for the objects within the repository. Projects, groups, and users are managed using the Workgroup Repository software. Passwords are managed by the database or operating system. Refer to [Defining Native Security](#). After installing the repository software and creating the repository database, use the Repository Administration tool to set up users, groups, and projects. Refer to [Repository Administration Tool Overview](#) for a description of this tool.

The following topics are discussed in this section:

- [Understanding the Security Model](#)
- [Working with Users](#)
- [Working with Groups](#)
- [Working with Projects](#)
- [Defining Native Security](#)
- [Optimizing Repository Security](#)

Understanding the Security Model

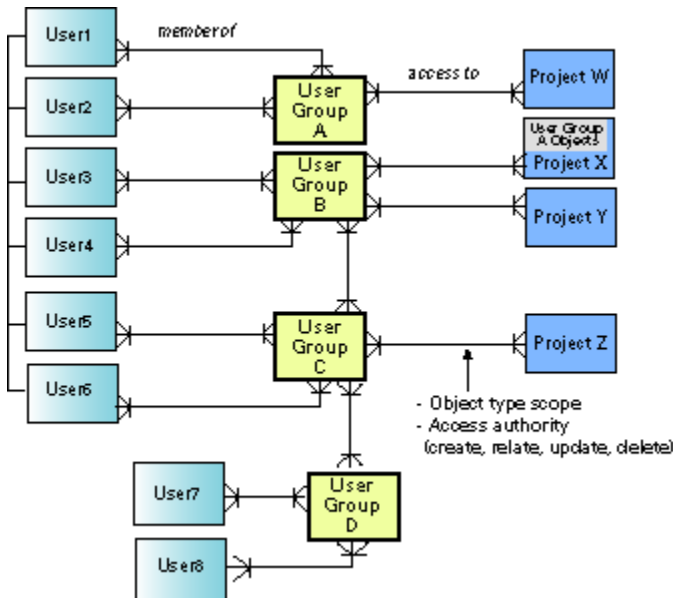
[Workgroup Repository Security Model](#) illustrates the basic Workgroup Repository security model. The security model consists of assigned associations and permissions between:

- [Projects](#)
- [Groups](#)
- [Users](#)

Associations and Permissions

This section presents the Projects, the Groups and the Users, that are associated in the security model.

Workgroup Repository Security Model



Projects

A Project is the area to which development objects are defined. For example, an ACCOUNT object, a RECEIVABLE object, and other objects needed for an accounting application could be created in a project called ACCOUNTING.

The Workgroup Repository can store multiple projects. When you connect to the repository, you must select one of the projects to which you have been given access. This is your active project until you change it in the Session Properties window. All newly created objects will belong to your active project. Refer to [Changing Your Active Project](#) for information about changing your active project.

Groups

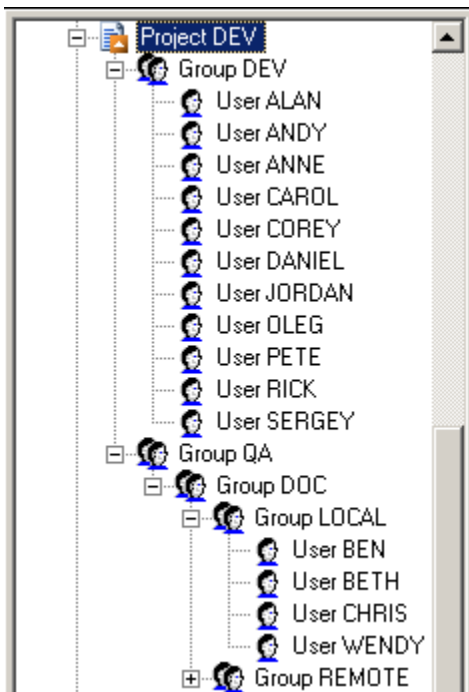
The Group defines users with common security permissions and scope. Multiple groups can be defined to a project, allowing different levels of permissions and scope. Additionally, a group can be defined to multiple projects, with the permissions in separate projects being different. For example, one user group can be permitted *relate* access to the objects in a project, while another group has *create*, *relate*, *update*, and *delete* access. Child subgroups can also be defined in the security model for the purpose of organization. All users in a child subgroup have the same security permissions as the users in the parent group.

Users

The user ID is stored in the repository security model for the authorization purposes. A user must have access to the database tables in which the repository objects are stored in order to access them in AppBuilder. Users have access to objects *only* in the projects to which they are defined. A user may work in only one project at a time. Refer to [Defining Native Security](#) for more information about defining user IDs.

The user's permission and scope is a function of the Group-Project relationship defined for that project. From the group level, it is possible to grant or remove authority levels from many users with one action. New users in a group have access to objects in the project associated with that group. [Repository Administration Project Hierarchy](#) shows the relationship between projects and groups as displayed in the Project Hierarchy window.

Repository Administration Project Hierarchy

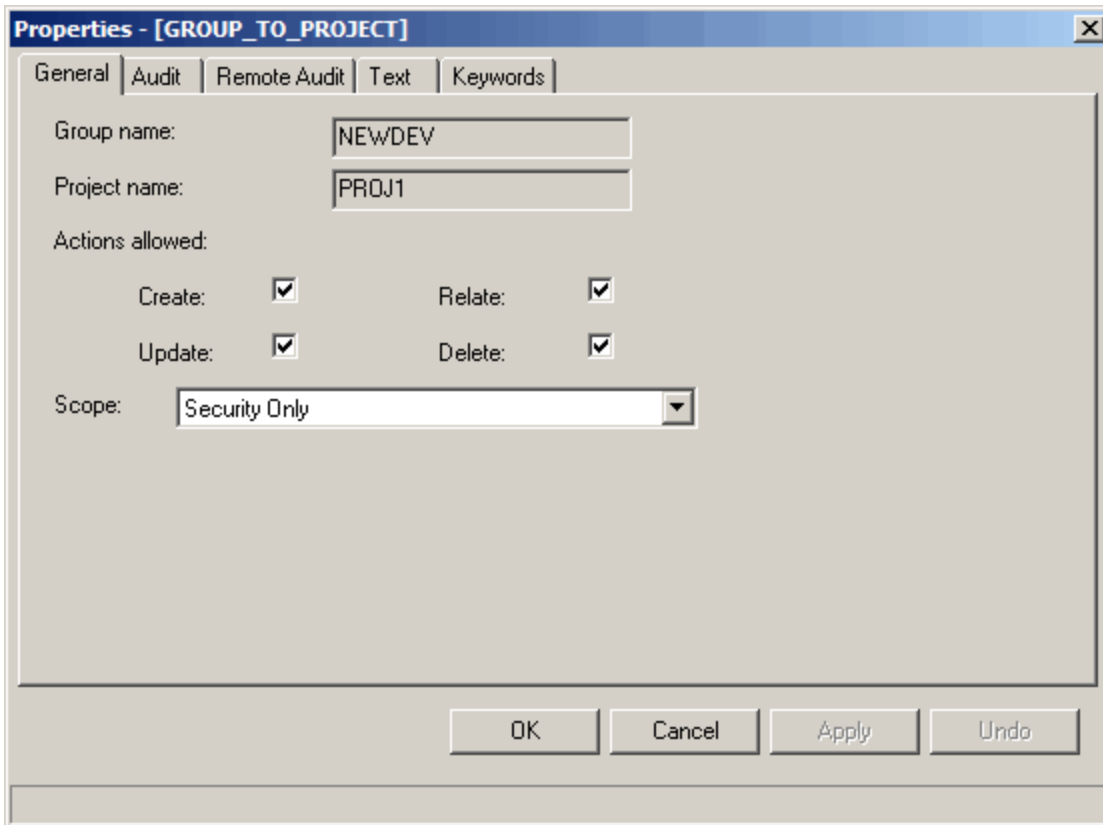


The access permissions that group members have to objects in the project is defined when a Group is added to a project hierarchy.

Setting Scope and Permissions

The Workgroup Repository security model allows the repository administrator to grant scope and permissions to each group individually. When creating the group, you define the scope and permission for the users within that group. [Relationship Properties for Doc Group](#) displays the Relationship Properties for a group, which includes the scope and permissions for all users in that group.

Relationship Properties for Doc Group



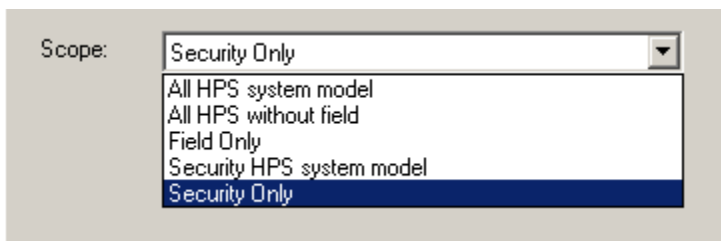
The Administrator assigns the following group security attributes:

- [Scope](#)
- [Permissions](#) (Create, Relate, Update, Delete)

Scope

Scope defines the set of objects that developers have access to.

Setting Group Scope and Permissions



[Definition of Scope Options](#) describes the five valid options for setting scope restrictions:

Definition of Scope Options

Scope	Function
All HPS system model	Specifies all object types defined in the AppBuilder Information Model
All HPS without field	Specifies all object types defined in the AppBuilder Information Model except for Fields
Fields only	Specifies Field objects only
Security HPS system model	Specifies all object types defined in the AppBuilder Information Model plus security objects
Security only	Specifies security objects only

Permissions

Permissions are set at the group level. They define the repository actions that are available to members of the Group. In the Repository Administration tool, permissions are defined on the Relationship Properties window for the Group (see [Relationship Properties for Doc Group](#)). To allow a developer to add, reuse, and modify (but not delete) objects in the repository, including objects that other developers have created, assign the developer membership in a group with create, relate, and update permission for all AppBuilder system model objects.

[Actions and their permission definitions](#) describes the permissions settings and descriptions of each option. Read access is implicit for *all* objects in *all* projects.

Actions and their permission definitions

Action	Permission Definition
<i>Create</i>	The authority to create objects.
<i>Relate</i>	The authority to make an object a child of another object.
<i>Update</i>	The authority to modify objects within the project if they are created by someone else.
<i>Delete</i>	Allows a developer to delete objects within the project if they are created by someone else.

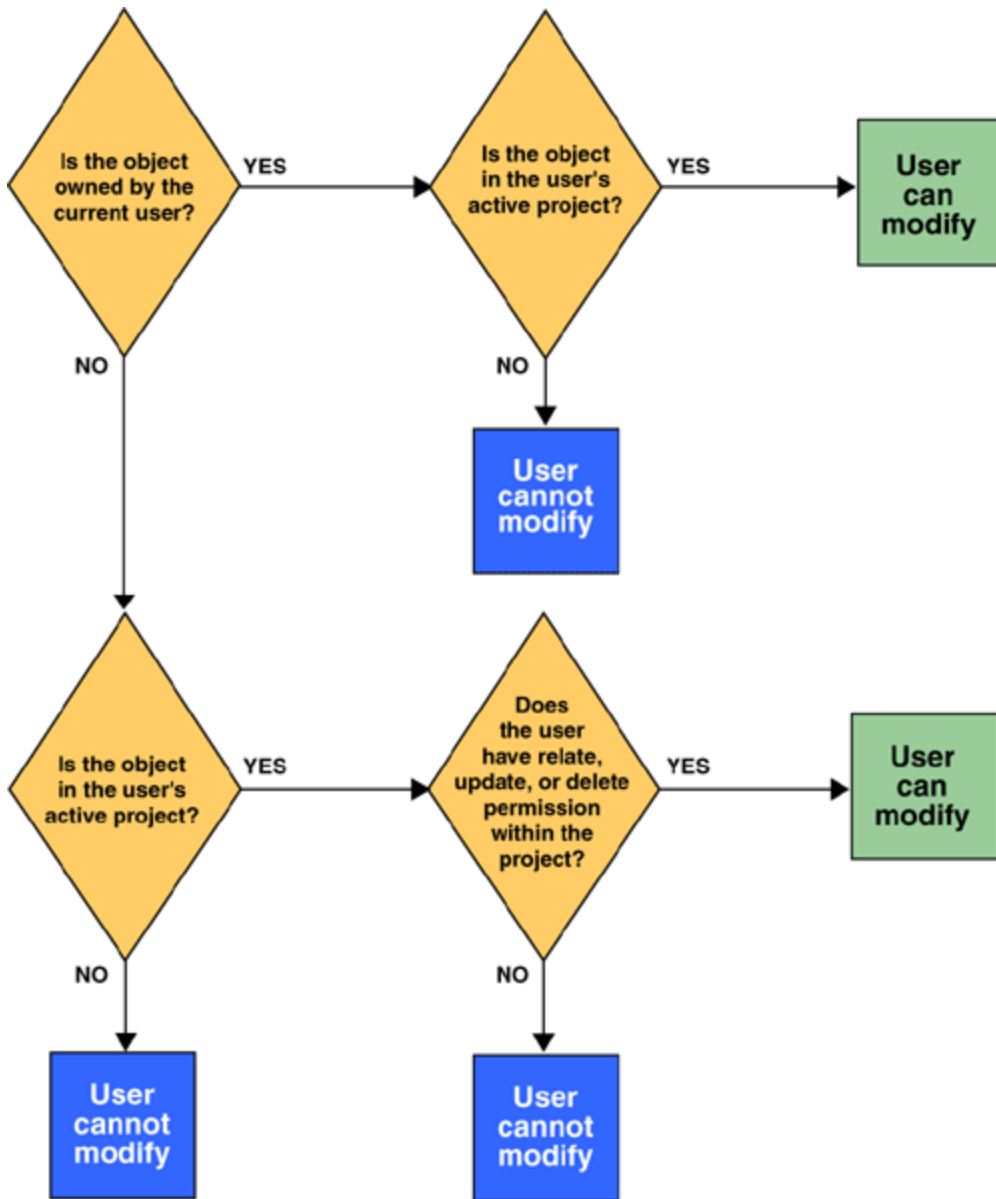
Verifying Ownership and the Active Project

The user who owns an object has exclusive *relate*, *update*, and *delete* permissions for that object until another user is granted the same authority. When a user attempts to update or delete an object, the Workgroup Repository security mechanism verifies that the user owns the object. The following rules apply:

- If the user owns the object and it is defined to the user's active project, the user can update or delete the object.
- If the user does *not* own the object, the user must have update and/or delete permission defined within the security model.

The project and owner of a given object are displayed on the Audit tab of the object's Properties window. [Verifying Ownership](#) illustrates these rules.

Verifying Ownership



Working with Users

After installing and setting up your Workgroup Repository, use the Repository Administration tool to set up users in the new repository. Refer to [Repository Administration tool](#) for an illustration of the Repository Administration tool.

Before a new user ID is valid in the repository, it must first be used to define a valid user in the underlying database, such as Microsoft SQL Server or DB2/UDB. You can either define a user in the database first and then create the user here, or you can create the user here first and then define the user in the database. Refer to the appropriate product documentation for your DBMS for complete information about defining users in SQL Server or DB2 /UDB.

The following topics are discussed in this section:

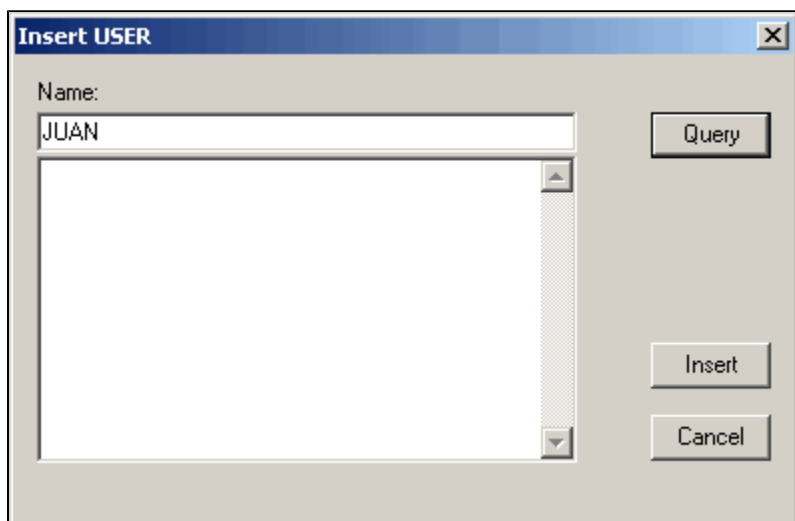
- [Creating a New User](#)
- [Associating an Existing User with a Group](#)
- [Viewing User Properties](#)
- [Viewing User Relationship Properties](#)
- [Setting User Security Levels](#)
- [Understanding the Default User ID](#)

Creating a New User

Before you create new user IDs in the repository, make sure each user ID is also valid in the database. Take the following steps to create a new user in a repository:

1. From the Security tab, highlight the group to which you will add this user. -Or, highlight the repository node. You do not have to add the user to a group right away;-you can first add it as a root object in the hierarchy and insert it into a group later.-
2. From the main menu bar, select **Insert > User** _Or, **right-click on the group and select _Insert Child > User** .The Insert User window displays.
3. Type the name of the user, and click **Insert**.

Insert User Window



The User Name must meet the following conditions:

- Cannot exceed 8 characters.
- Cannot include blank spaces or special characters.

The user is listed in the Hierarchy Diagram window under the selected group. The user name is displayed in the status panel at the bottom of the administration tool window.



If you create a new user for a Personal Repository or Workgroup Repository and try to connect to the repository as that new user, you may see a "Security Validation" error message. The new user must be added to the PUBLIC group before connecting to the repository.

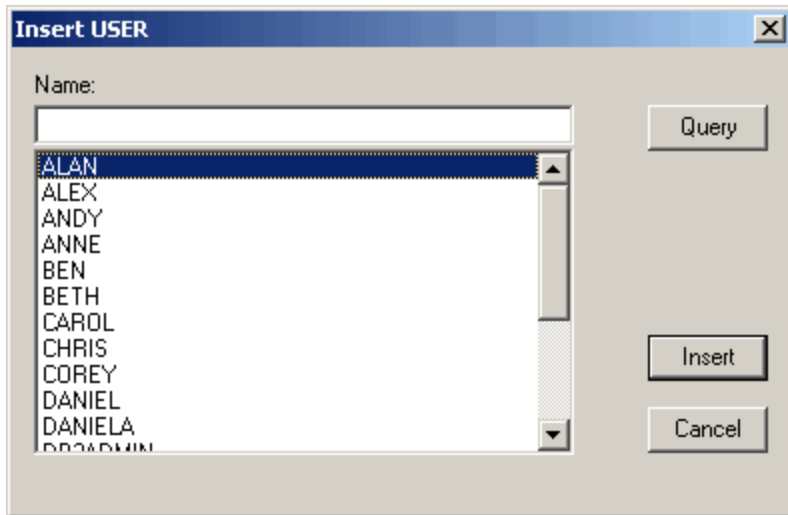
Associating an Existing User with a Group

After you have created a group in the repository, you need to either create a new user or select from the existing users in the repository. To create a new user, refer to [Creating a New User](#). To associate an existing user with the group, take the following steps:

Highlight the group in the Hierarchy window.

1. From the main menu bar, click **Insert > User** . Or, right-click on the group and select **Insert Child > User** .The Insert User window displays.
2. Click **Query** and select a user from the list. Press the **Ctrl** or **Ctrl+Shift** keys to select more than one user.
3. Click **Insert**.

Insert User window (Query)



The user is listed in the Hierarchy window under the selected group. The user name is displayed in the status panel at the bottom of the Repository Administration tool window.

Viewing User Properties

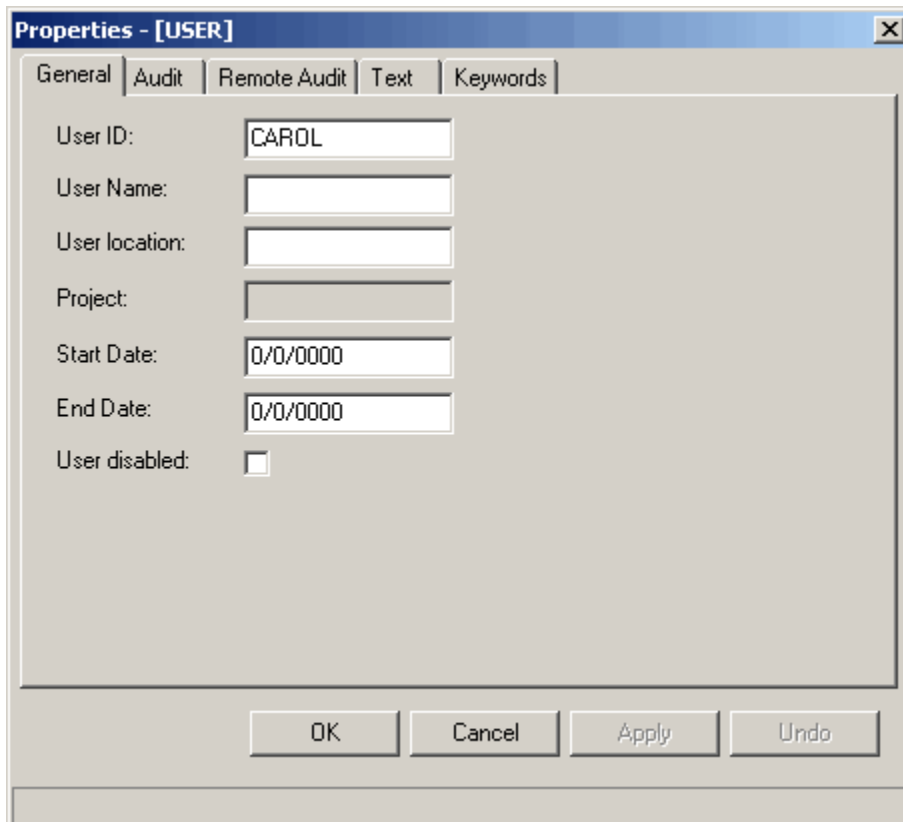
There is more than one way to view the properties of an object in the Hierarchy. To view the properties of a user:

Right-click on the User in the Hierarchy and select *Properties*.

Or,

1. Highlight the Group in the Hierarchy. The users associated with that group are listed in the right pane.
2. Right-click on the User and select **Properties**. The User Properties window displays.

Properties (User)



User Properties (General tab)

Property	Description
User ID	The ID assigned to this user.
User Name	The user's name.
User Location	The location where the user works.
Project	The project with which the user ID is associated.
Start Date	The date the user ID becomes active. Check the box and select a date from the drop-down calendar. If a date is not specified, the user is automatically active.
End Date	The date the user ID is deactivated. Check the box and select a date from the drop-down calendar. If a date is not specified, the user automatically remains active. If an end date is specified, the user is automatically disabled on that date.
User disabled	Check this box to permanently disable a user. If the start and end date parameters are used, it is not necessary to check this box. Users associated with the SYSADM group cannot be disabled. You must move the user to another group and then disable their access. To easily move a user from one group to another, use cut and paste. For more information, see Using Cut, Copy, and Paste in the Hierarchy .

For a description of the other tabs on this window, refer to [Properties window tab descriptions](#).

Viewing User Relationship Properties

Repository objects can have relationships to other objects in the repository. For instance, a user can be a child of a group. To view the relationships associated with an object easily, take the following steps:

1. Right-click the group in the Hierarchy Diagram window.
2. Select **Relationship Properties**. Or, highlight the user in the Hierarchy Diagram window; then, from the main menu bar click **Selected > Relationship Properties**.
3. The Relationship Properties window displays.

User Relationship Properties (General tab)

Property	Description
Parent name	The user's ID.
Child name	The group name
Sequence number	A number automatically assigned to each entity in sequence, used to define the order of relationships attached to a given entity. You can change this number to change the order of child entities under a parent entity. This value can be a number from 0-99.
Separator ID	Uniquely distinguishes duplicate relationships between one parent and its child entities.

For a description of the other tabs on this window, refer to [Properties window tab descriptions](#).

Setting User Security Levels

Security permissions for each user are set at the group level. All users associated with a group have the same level of security permissions. Refer to [Granting Permissions Security for the Group](#) for information about security permissions. A subgroup has the same security settings as its parent group.

Understanding the Default User ID

AppBuilder provides a default user ID called USERID with a password called PASSWORD with the AppBuilder installation to provide users initial access to the Workgroup Repository. The default user ID (USERID) is a member of the SYSADM group, which gives it full access authority to the AppBuilder development environment. After installing AppBuilder, use the default user ID to create your own system administrator IDs, and then remove the default user ID (USERID) from the SYSADM group. Later, lower the permissions level for USERID to prevent unauthorized access to the repository. Refer to [Modifying the Default Repository User ID](#) for more information about this procedure.

Working with Groups

This section covers the following topics related to groups:

- [Associating a Group with a Project](#)
- [Creating a Subgroup](#)
- [Granting Permissions Security for the Group](#)
- [Viewing Group Properties](#)
- [Viewing Group Relationship Properties](#)
- [Viewing Users in a Group](#)

Associating a Group with a Project

After you create a project, you need to associate one or more groups with the project. You can do this by creating a new group or associating a group that already exists in the repository. The following topics are discussed in this section:

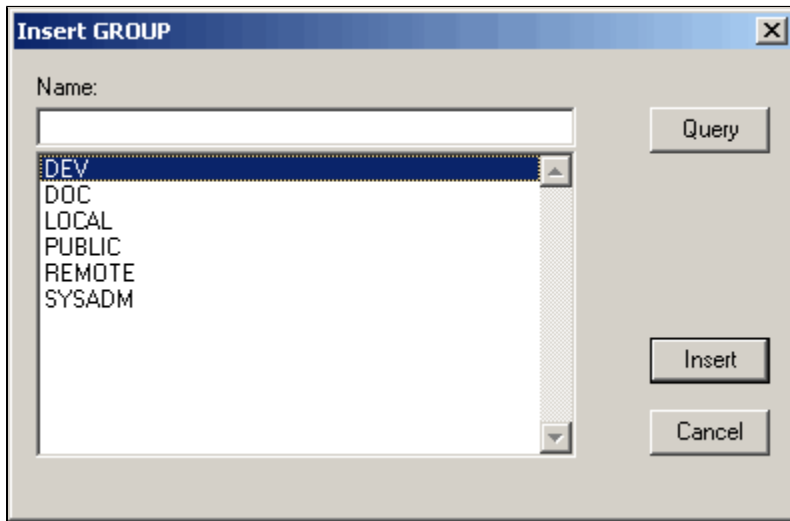
- [Creating a New Group](#)
- [Associating an Existing Group with a Project](#)

Creating a New Group

Take the following steps to create a new group in a repository:

1. Highlight the project in the Hierarchy window.
2. From the main menu bar, select **Insert > Group** .Or, right-click on the project and select **Insert Child > Group** .The Insert Group window displays.
3. Type the name of the group, and click **Insert**. The Group Name must meet the following conditions:
 - Cannot exceed 10 characters.
 - Cannot include blank spaces or special characters.

Insert Group Window



The group is listed in the Hierarchy Diagram window under the selected project.

After you have created a group, you must populate that group with one or more users. Refer to [Associating an Existing User with a Group](#).

Associating an Existing Group with a Project

After you have created a project in the repository, you need to either create a new group or select from the existing groups in the repository. To create a new group, refer to [Creating a New Group](#). To associate an existing group with your project, take the following steps:

1. Highlight the project in the Hierarchy Diagram.
2. From the main menu bar, click **Insert > Group** .Or, right-click on the project and select **Insert Child > Group** .The Insert Group window displays (see [Insert Group Window](#)).
3. Click **Query** and select a group from the list. Press the **Ctrl** or **Ctrl+Shift** to select more than one group.
4. Click **Insert**.

The group is listed in the Hierarchy Diagram window under the selected project. The group name is displayed in the status panel at the bottom of

the administration tool window.

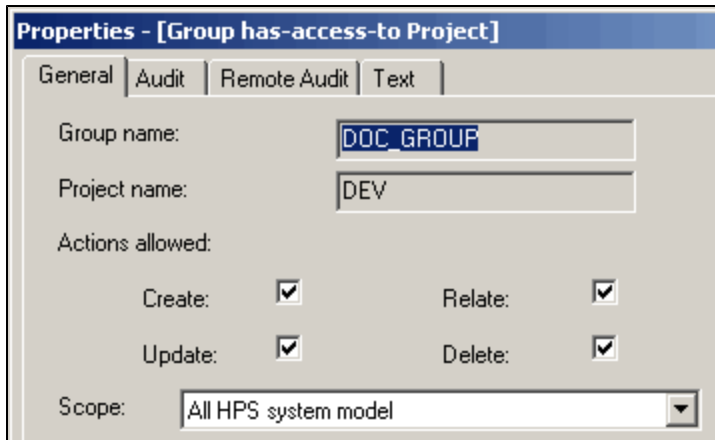
After you have created a group, you must populate that group with one or more users. Refer to [Associating an Existing User with a Group](#).

Granting Permissions Security for the Group

The status panel at the bottom of the repository tool main window displays specific information about the group, including security access permissions for users associated with the group. The security access information is set for the group as a whole; after the security is set for the group all users associated with that group, either current users or future users, will have that level of security.

A group can be a child of multiple projects; be sure to highlight the group within the correct project to set the permissions for that group in that project. The permissions apply to a specific group within a specific project only.

Security Information for the group



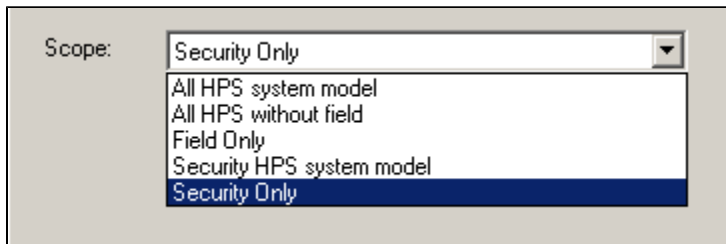
The Administrator assigns the following group security attributes:

- [Scope](#)
- [Permissions](#) (Create, Relate, Update, Delete)

Scope

Scope defines the set of objects to which the developers have access.

Setting Group Scope



[Definition of Scope Options](#) describes the five valid options for setting scope restrictions:

Definition of Scope Options

Scope	Function
All HPS system model	Specifies all object types defined in the AppBuilder Information Model.
All HPS without field	Specifies all object types defined in the AppBuilder Information Model except for Fields.
Fields only	Specifies Field objects only.
Security HPS system model	Specifies all object types defined in the AppBuilder Information Model plus Security objects.
Security only	Specifies Security objects only like User, Group, and Project.

Permissions

Permissions define the allowable actions available for the group within the repository. [Actions Permission Options](#) describes the permissions settings and descriptions of each option. Read access is implicit for *all* objects in *all* projects.

Actions Permission Options

<i>Create</i>	The authority to create objects.
<i>Relate</i>	The authority to make an object a child of another object.
<i>Update</i>	The authority to modify objects within the project if they are created by someone else.
<i>Delete</i>	Allows a developer to delete objects within the project if they are created by someone else.



To allow a developer to add, reuse, and modify (but not delete) objects to the repository, including objects that other developers have created, assign the developer membership in a Group with create, relate, and update permission for all AppBuilder system model objects.

Creating a Subgroup

You can also create a group within a group. The subgroup is considered a child of the parent group; it has the same scope and permissions as the parent group.

To create a subgroup, take the following steps:

1. Highlight the parent group in the Hierarchy Diagram.
2. From the main menu bar, click **Insert > Group**. Or, right-click on the group and select **Insert Child > Group**. The Insert Group window displays (see [Insert Group Window](#)).
3. Click **Query** and select a group from the list. Press the **Ctrl** or **Ctrl+Shift** to select more than one group.
4. Click **Insert**.

The subgroup is listed in the Hierarchy Diagram window under the selected parent group. The subgroup name is displayed in the status panel at the bottom of the administration tool window.

After you have created a subgroup, you must populate that group with one or more users. Refer to [Associating an Existing User with a Group](#).

Understanding the SYSADM Group

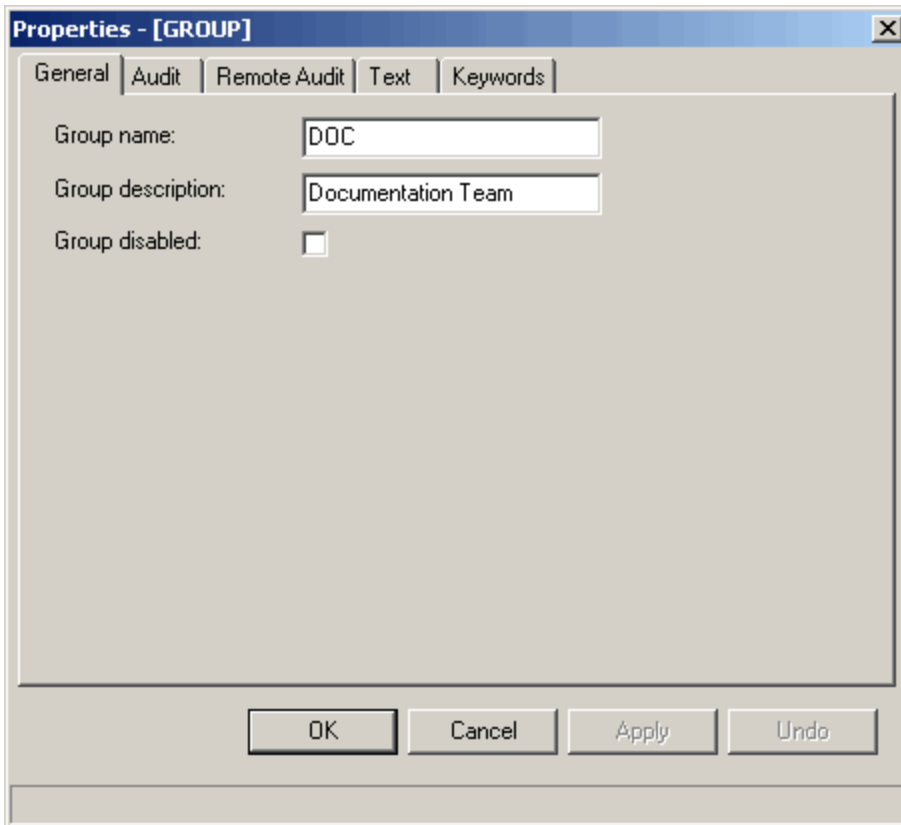
The group SYSADM comes with your AppBuilder installation. Users associated with this group have full access authority to the AppBuilder development environment. Changes made to permissions for this group will not take effect. Full permissions remain. The default user ID (USERID) is initially part of the SYSADM group. Refer to [Understanding the Default User ID](#) for more information on how to work with the default USERID.

Viewing Group Properties

To view the Properties of a Group, right-click on the Group in the Hierarchy and select *Properties*. The Group Properties window displays.

The general properties for any entity are unique. For a group, this is the group name and a description of the group. Every properties window has the same properties tabs. [Properties window tab descriptions](#) provides a description of each property tab.

Properties (Group)



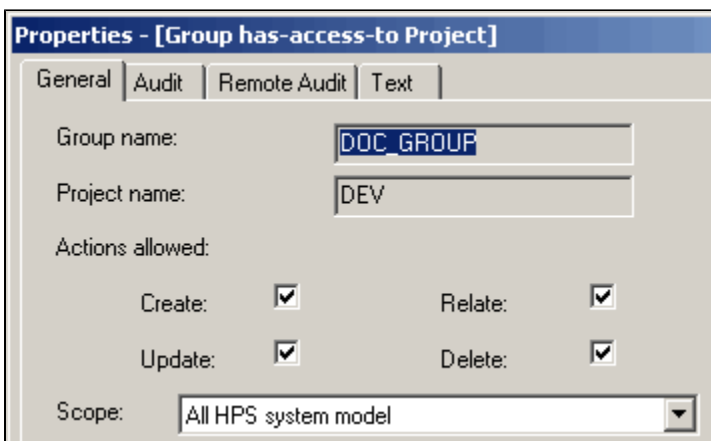
Viewing Group Relationship Properties

Repository objects have relationships to other objects in the repository. For instance, a group can be a child of a project or can be the child of another group. To easily view the relationships associated with an object, take the following steps:

1. Right-click the group in the Hierarchy Diagram window.
2. Select **Relationship Properties**. Or, highlight the group in the Hierarchy Diagram window; then, from the main menu bar click **Selected > Relationship Properties**.

The Relationship Properties window displays (see [Relationship properties for a group](#)).

Relationship properties for a group



The properties listed on the General tab refer to security settings for the group's access to the parent project. Security is set at the group level. All users within the group have the same security level. For a description of each field on the General tab, refer to [Granting Permissions Security for the Group](#).

For a description of the remaining property tabs on this window, refer to [Properties window tab descriptions](#).

Viewing Users in a Group

Double-click a group in the Hierarchy Diagram window to open the group and display the users associated with the group.

Working with Projects

This section discusses how to complete the following project related tasks using the Repository Administration Tool. Step-by-step instructions for the following tasks are included in this section:

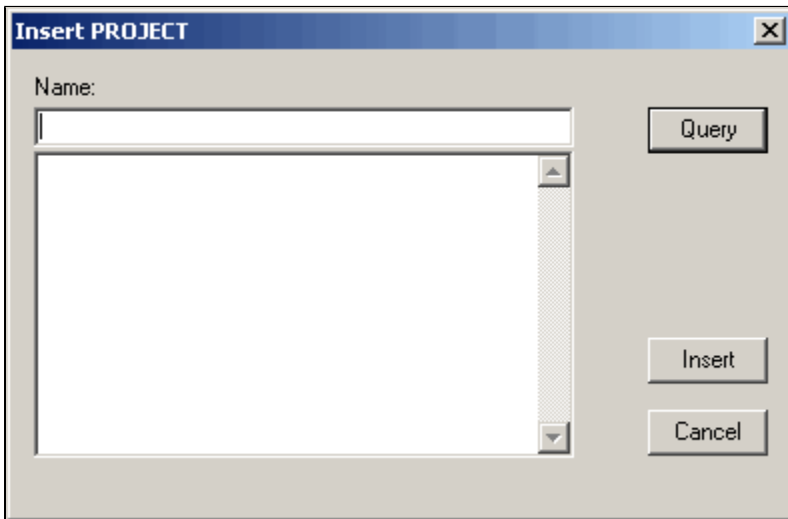
- [Creating a New Project](#)
- [Viewing all Projects that Exist in the Repository](#)
- [Viewing Project Properties](#)
- [Viewing Project Users](#)

Creating a New Project

Take the following steps to create a new project in a Workgroup Repository:

1. Highlight the repository in the Hierarchy window.
2. From the main menu bar, select **Insert > Project**.
3. Or, right-click on the repository and select **Insert Object > Project**. The Insert Project window displays.
4. Type the name of the project, and click **Insert**. The Project Name must meet the following conditions:
 - Project names must be four (4) characters or less if you plan on migrating files between the Workgroup and the Enterprise Repository.
 - Cannot exceed 10 characters.
 - Cannot include blank spaces or special characters.

Insert Project Window



The project name is displayed in the status panel at the bottom of the tool window. After you have created your project, you must associate a group with the project. Refer to [Associating a Group with a Project](#).

Viewing all Projects that Exist in the Repository

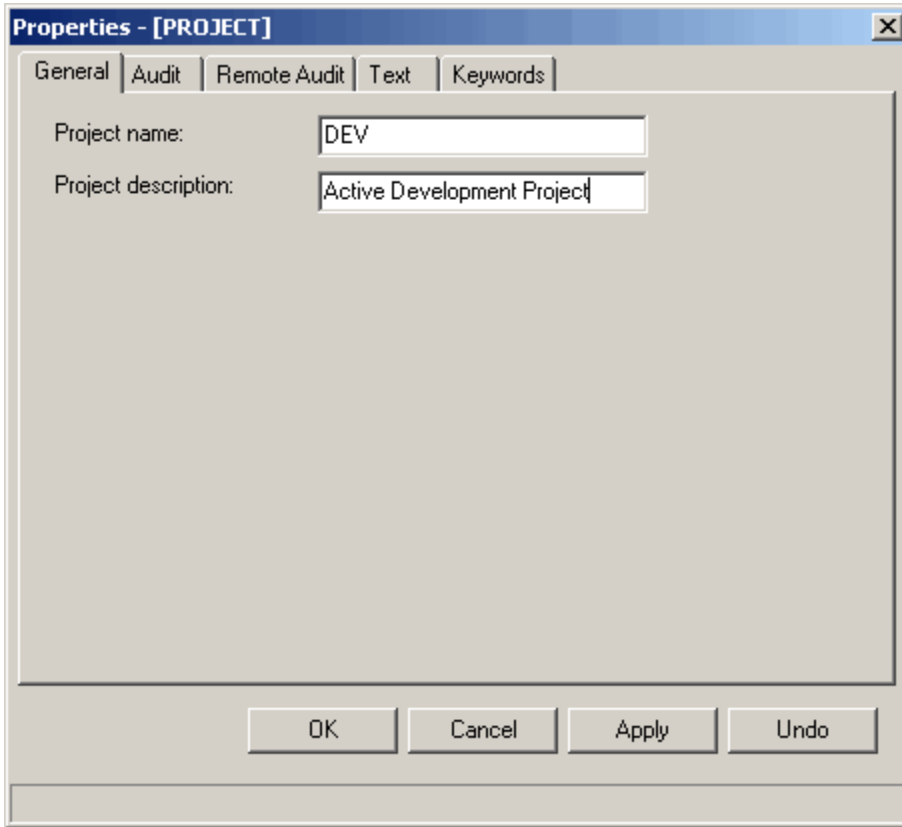
To view all the projects that exist in the repository from the Hierarchy window of the Repository Administration tool, take the following steps:

1. Select the Security tab of the Hierarchy window.
2. Highlight the repository in the Hierarchy Diagram window.
3. From the toolbar, select the **Show Root Objects** button.

Viewing Project Properties

To view the Properties of a Project, right-click on the Project in the Hierarchy and select *Properties* . The Project Properties window displays. The general properties for any entity are unique. For a project, this is the project name and a description of the project. Every properties window has the same properties tabs. [Properties window tab descriptions](#) provides a description of each property tab.

Properties (Project)



Properties window tab descriptions

Tab	Description
General	This is the general properties for the object. These properties will differ depending on the object.
Audit	The Audit tab contains information about the object as it exists on the PC.
Remote Audit	The Remote Audit tab contains information about the object as it exists on the Host.
Text	The Text tab is a place where a user can enter more information about the object. For example how to use the object.
Keywords	The Keywords tab can be used to enter words that can be searched on. For example, use the word "Development" for a project that is used by development.

Viewing Project Users

Double-click on a project in the Hierarchy Diagram window to open the project and display the users associated with the project.

Changing Your Active Project

The project you selected when connecting to the repository is your active project. From Construction Workbench, you can change your active project without having to disconnect and reconnect to the repository; however, because Repository Administration users should be members of the SYSADM project (only users with system administrator authority can access the Repository Administration tool), you should not attempt to change the active project from the Repository Administration tool. For information about changing your active project, refer to the *Development Tools Reference Guide* .

Deleting a Project

Deleting a project can be handled by right-clicking on the relevant project and deleting it from the repository. Deleting a project requires a confirmation that you do want the project deleted from the repository.

Defining Native Security

Initially, each user must be defined to the supporting Database Management System (DBMS) providing two levels of security:

- Security defined within the supporting DBMS (SQL Server or DB2/UDB)

The DBMS or operating system controls the user password.

- Security defined within the Workgroup Repository.

Once users have been defined within the native DBMS, the Workgroup administrator may add users within the repository and then assign the appropriate project permissions.

AppBuilder allows you to configure your security settings or use the defaults when appropriate. The following sections describe how to set security restrictions for specific servers or within the repository.

- [Using Default Projects and Groups](#)
- [Setting System Administrator Permissions](#)
- [Modifying the Default Repository User ID](#)
- [Configuring Native Security](#)

Using Default Projects and Groups

AppBuilder contains the following default projects and groups:

- **SYSADM** - The SYSADM project represents access entry into the entire development system and database. Any user in a group linked to the SYSADM project has full access to the repository. Give your project administrator full access to the repository immediately after installing Workgroup Repository software.
- **SEER1** and **SEER2** - These projects contain default repository objects.
- **PUBLIC** - The PUBLIC group, by definition, represents all Workgroup users. Consequently, if you want to give all users access to the objects in a project, you can do so globally by associating the PUBLIC group with that project.

AppBuilder is shipped with one default user (USERID) defined in its security model. When you set up and configure the default repository, create the same default user within the native DBMS using the password *PASSWORD*. Use *USERID* and *PASSWORD* to log onto the system and configure the initial security model.

By default, the user *USERID* is defined as a system administrator or a member of the project *SYSADM*. This configuration is necessary for an initial connection to be established because at least one user must pre-exist within both the DBMS and the repository. Remove the default configuration after updating the model with new users and passwords.

Setting System Administrator Permissions

The default project included in the AppBuilder repository, *SYSADM*, permits entry for its users into the entire development system and database. System administrator group members have full access and permissions to the repository, security model, and Workgroup database. This mechanism provides centralized control and administration, thus maintaining the security infrastructure. Typically, system administrators are defined in the *SYSADM* group, providing them with full access to the repository. All repository security checks are disabled for users in the *SYSADM* group. Limit this level of permissions to a select number of users, since numerous *SYSADM* users may create disorder in the repository.

Modifying the Default Repository User ID

The Workgroup Repository allows anyone using the default *USERID/PASSWORD* to have full access to AppBuilder development objects contained within the repository. Although the default repository only contains system objects, to prevent unauthorized access:

1. Create a new user and add that user to the *SYSADM* Group prior to removing the default *USERID* from the *SYSADM* group. Verify that the new user can connect to the repository before removing the default *USERID*.



Do not remove the default *USERID* object without first creating a new user with administrative permissions.

2. Move the default *USERID* to a group that has read-only permissions, ensuring that a read-only user is always defined in the repository. The default user *USERID* remains in the repository but outside the *SYSADM* group.

Using the Repository Administration tool, you can cut and paste the default *USERID* to a new group.

Configuring Native Security

The steps for configuring native security depend on which type of database you are using.

Configuring Native Security for DB2 Servers

For a DB2 repository, although it is necessary for the user ID to be recognized by the database, it is not necessary to create the default user ID on both the operating system and the database. Create the default user ID on the operating system and create a group on the operating system. AppBuilder provides the group FWYGRP to make this process more simple. Create the FWYGRP group on the operating system; then, associate the user ID with the group FWYGRP on the operating system. In doing this, the user ID is automatically recognized by the database through the FWYGRP group. When you create the repository, the FWYGRP group is automatically created in the database. The steps that follow, depend on the operating system you are using. Refer to your specific operating system documentation to accomplish the following three steps:

1. Create FWYGRP group on the server.
2. Create a local user account.
3. Add user to FWYGRP group.

Configuring Native Security for SQL Servers

To configure native security for SQL servers, first register the server and define a user to native security. The first user added becomes the System Administrator (sa) when Workgroup security is defined. If the SQL Server has already been registered in SQL Enterprise Manager, skip steps 1 through 5.

1. Double-click on the SQL Enterprise Manager icon located in the Microsoft SQL Server program group. The Server Manager window opens.
2. Select the SQL 7.0 icon.
3. Select **Server > Register Server**. The Register Server window opens.
4. Type the < *SQL Server name* > in the *Server* field.
5. Click *Servers* for a list of the available servers.
6. Select **Server** and click *OK*.
7. Click **Use Trusted Connection**. If there is a problem with the registration, re-examine the SQL Server setup.
8. Click **Register**.
9. Click **Close**. The machine name appears with a stoplight icon in the Server Manager window.
10. Select the < *machine name* >
11. Select **Security, Logins or Users** in the Database hierarchy. The Manage Logins window appears.
12. Type < *user ID* > in the **Login Name** field.
13. Type < *password* > in the **Password** field. You must define a valid password to use Workgroup. Workgroup will not function if a null password is defined.
14. Select the **Permit** column for the Workgroup database for database access.
15. Select **FREWAYGROUP** from the drop down combo box in the Group column. FREWAYGROUP appears in a drop-down combo box after selecting the Group column for FREWAY within Database Access.
16. Select **Add**.
17. Confirm the new user's password and click **OK**.
18. Click **Close** to exit the Manage Logins screen.
19. Select **File > Exit** to exit SQL Enterprise Manager.

After completing the procedure for Configuring Native Security for SQL Servers, the Workgroup user has permission to access the SQL Server tables. This is the first level of security a developer must have before operating within a Workgroup Repository. The user must also be recognized within the AppBuilder security model to access the objects within the repository. (See [Associating an Existing User with a Group](#).) When the starting point is a default repository, configure the entire security model ensuring that users have the correct level of permissions.

Optimizing Repository Security

The following recommendations can assist you in structuring and managing your repository security system for maximum performance.

- Define as few SYSADM users as possible.
- If the system administrator also develops applications with AppBuilder, define separate user IDs for administration functions and development functions, thus preventing unintentional SYSADM actions.
- Before the *USERID* user object is removed from the SYSADM group, ensure that the new SYSADM user can successfully attach to the repository, so you don't end up with a Workgroup Repository without a SYSADM user.
- If multiple projects are being developed within the same repository, define project security administrators in addition to the SYSADM users.
- Leave the user *USERID* in the repository with read-only permission to ensure that a read-only user will always be defined.
- Define sufficient users to the security model to avoid having two developers using the same user ID simultaneously.
- Use the client machine names, developer names, or TSO ID's whenever possible for user IDs.
- If objects will be migrated to the Enterprise Repository, define the Workgroup Repository security model to match the Enterprise Repository security model.

- If the development project also exists on the mainframe, match user IDs to TSO IDs to maintain consistency.

Managing a Repository

In this section we outline tasks a repository administrator can do to manage the repository as a whole. Before any management function is started, the system automatically disconnects the repository and reconnects it when the action is complete. The following operations are *not* supported for a remote repository: Repository Export, Pack, Reindex, and Reorganize. Later in this guide we discuss working with repository objects. The following topics are covered in this section:

- [Working with the Service Control](#)
- [Creating a Repository](#)
- [Creating a Repository Alias](#)
- [Deleting a Repository](#)
- [Deleting a Repository Alias](#)
- [Full Repository Migration](#)
- [Repository Replicator](#)
- [Packing and Reindexing a Repository](#)
- [Reorganizing a Repository](#)
- [Reviewing the Log File](#)
- [Broadcasting Messages](#)

Working with the Service Control

Before performing management tasks on the repository, the Repository Service must be stopped. The Repository Administration tool automatically starts and stops the Service as needed depending on the tasks you perform. When you attempt to perform a task that requires the Repository Service to stop, the following window is displayed.



The system stops the service before an AppBuilder installation upgrade. If you are using a Workgroup Repository, you must restart the repository service. You can do this through the Repository Administration tool or through your Windows Services Manager; see [Start Repository Service](#).

Stop the Repository Service?

Do You Want To Stop The Repository Service?

Authentication

Machine: LSOBHANDELL

Userid: DB2ADMIN

Password:

Broadcast Message Wait for: 120 seconds

Repository going down in 2 minutes.

Current Connections

- AppBuilder Job Scheduler
- Construction Workbench
- Repository Administration

Yes No

This window displays the Authentication information for the repository; it gives you an opportunity to broadcast a message to connected users; and it displays the processes that are currently connected to this repository. The current connections are only available when you are connected to a local machine.

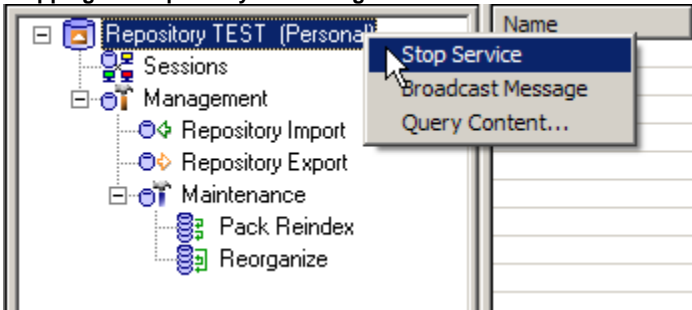
To broadcast a message:

1. Check the **Broadcast Message** check box and type the message.

1. Enter the number of seconds the Repository Service will wait before shutting down the service, and click **Yes** when you are ready to stop the service.
The system displays a message informing you of the number of minutes before the service goes down.
2. Click **OK**.

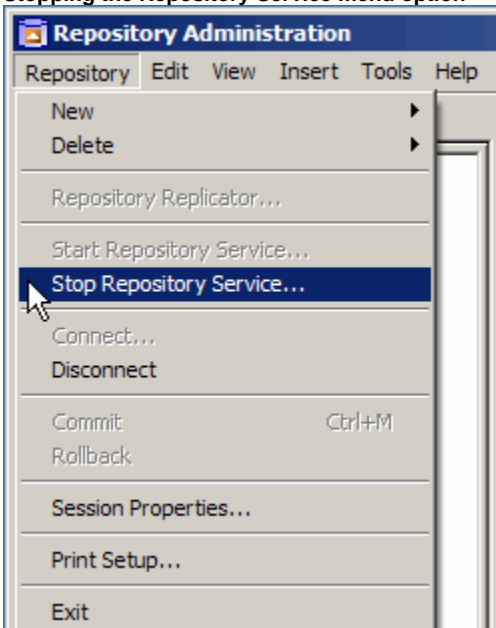
Although the system will prompt you to stop the service when needed, you do have control to manually start and stop the service. From the Management tab, right-click on the root object in the hierarchy and select **Stop Service**.

Stopping the Repository Service right-click menu



Or, you can select **Repository > Stop Repository Service** or **Repository > Start Repository Service** respectively.

Stopping the Repository Service menu option



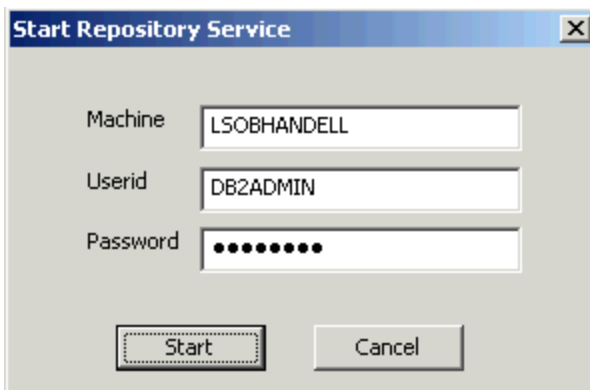
The Repository Administration tool automatically starts the service when you log in. However, after manually stopping the service or cancelling the login, you can manually start the service.



You must have administrator rights to start or stop the service manually. You must also have DB2 rights to connect/read the data.

To do this:

1. Click **Repository > Start Service** (see the above figure). When the service is stopped, the **Start Repository Service** option is available. The following dialog displays.
Start Repository Service



2. Enter the name of the machine where the repository resides.
3. Type the repository User ID and password, and click **Start**.
The system displays a message indicating that the service is started.

After manually starting the service, you have to manually connect to the repository. To do this, click **Repository > Connect**. For more information, refer to [Connecting to and Disconnecting from a Repository](#).

Service Control for a Remote Machine

The security information for the Workgroup Administrator is validated on the remote machine to control the Workgroup service on the remote machine. The Workgroup Administrator must be defined as an administrator on the remote machine in order to issue a request to start or stop the remote Service.

To start or stop the Repository Service, or view status details on a remote machine, the Windows user account must be a member of the administrator group on the remote machine.

Understanding the Freeway Service Manager

AppBuilder uses a number of services that manage who and how many clients are attached to the Workgroup Repository. The main repository service is called GRESVCNT (GRE Server Controller). It is responsible for a number of subordinate services called GRESRVNT (GRE Servants).

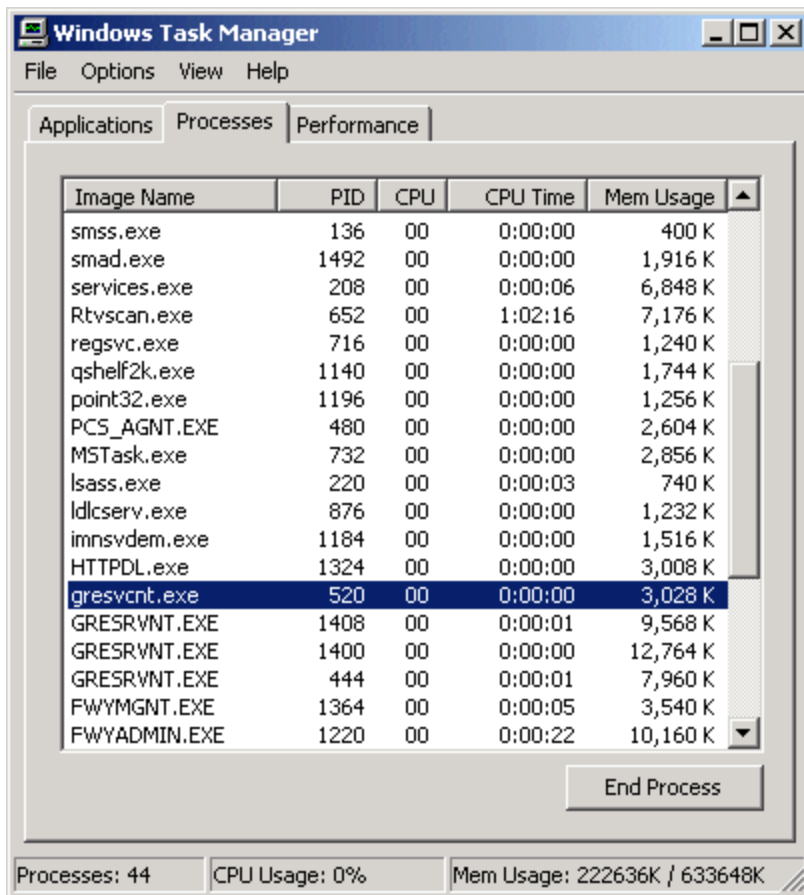
A development tool, such as Construction Workbench that connects to the repository requires a separate GRESRVNT service on the hosting machine. You will not run out of GRESRVNT processes because they are self-spawning. In other words, if a developer tries to open a new tool that attaches to the Workgroup Repository and there are no more available GRESRVNTs, a new one is started automatically. All GRESRVNT processes stay alive until the Freeway Server Controller is stopped.

The following figure displays the GRESVCNT process and the three GRESRVNT processes.



Do not attempt to close down processes from this panel. Always use the GRESVCNT -stop command.

Session Processes



Use the Report pane on the Repository Administration tool to see who owns which session number.

Creating a Repository

Using the Repository Administration tool, you can set up one or more Personal Repositories on your local machine or you can set up a Workgroup Repository on a server machine. This section outlines the procedures you need to set up either type of repository. To create a repository, you must have administrative rights on the machine. The following topics are discussed in this section:

- [Creating a Personal Repository](#)
- [Creating a Workgroup Repository](#)

Creating a Personal Repository

Using AppBuilder, you can create a single repository or multiple repositories on a local personal computer. You now have a choice between using DB2 Universal Database or Microsoft SQL Server 2000 Desktop Engine (MSDE 2000). Each Personal Repository creates a .INI file using the name assigned to the Personal Repository. For example, creating a new Personal Repository named NEWLRE generates a file named NEWLRE.INI.

Each Personal Repository is independent of the codepages of any other Personal Repository, mainframe settings, and job status. Multiple repositories make it possible to have one Personal Repository querying objects while a second processes a download job at the same time on the same machine.

The number of Personal Repositories that can be defined is limited by two factors:

- The number permitted by the database manager
- The amount of space on your system

The following topics are included in this section:

- [Personal Repository Connection Requirements](#)
- [Increasing available DB2 databases](#)
- [Using the Personal Repository Setup Wizard](#)

Personal Repository Connection Requirements

If your workstation is disconnected from the network, either the NetBEUI Protocol or MS Loopback Adapter must be installed. The Personal Repository uses named pipes to communicate with the database. Named pipes requires the NetBEUI Protocol or MS Loopback Adapter to function when a machine is not connected to a network.



Before you install AppBuilder and set up a Personal Repository, you must have a supported universal database (UDB) installed. For a complete list of supported universal database software, see the Software Support Matrix at <http://support.bluephoenixsolutions.com>.

The first time you create a Personal Repository, AppBuilder prompts you for a Freeway ID. Because the Freeway ID is used to generate short names for newly created objects, it is extremely important to have unique Freeway IDs across your organization to avoid short name conflicts. Freeway ID values can be between 400 and 1295. Your administrator can provide you with a unique number.

The Freeway ID is validated when you select *Next*. If you enter a number that is outside the valid range for the Freeway ID, the wizard displays a message box indicating that you have entered an invalid ID.

Increasing available DB2 databases

The restrictions defined by your repository administrator for your particular environment and the size of your local disk space are variables that can be controlled locally.

To increase the number of DB2/UDB databases available for Personal Repositories, complete the following steps:

1. Open the DB2 Control Center.
2. From the left side of the panel, under All Systems -> <Machine name>, right-click the **DB2 Instance** under the **Instances icon**.
3. Select the **Configure Parameters** option.
4. Select the **Environment** group.
5. Select the maximum number of concurrently active databases in the NUMDB line.
6. Change this value and click **OK**.

When DB2 restarts, the value increases.

Using the Personal Repository Setup Wizard



You cannot install a Personal Repository and a Workgroup Repository on the same machine.

1. From the Repository Administration tool, select **Repository > New > Repository**. The Personal Repository Setup Wizard displays. This screen lists the DBMS (DB2 and SQL Server) installed in the system. If multiple named SQL Server instances are installed in the system, this list displays the instance name as well. For example:
SQL Server : COMPUTERTNAME\INSTANCENAME)

Select a database for the Personal Repository

Personal Repository Setup Wizard

Select Database Management System
Select Database Management System that Personal Repository server will use.

Which Database Management System do you want to use?

DB2 Universal Database
DB2 Universal Database
SQL Server :(local)

Personal Repository Setup Wizard < Back Next > Cancel Create

- From the drop-down list, select the DBMS you want to use for your Personal Repository. You must have that DBMS installed on your machine. Click **Next**.

DB2 Universal Database

If you have selected this as your database, the following dialog displays.

Database Connection Parameters

The Connection Parameters dialog displays. [Connection Parameters dialog](#) provides descriptions for the fields on this dialog. It is possible to bypass additional dialogs and create the database from this dialog. To do so, click *Create*.

The additional dialogs give you the opportunity to change the location where the database will be installed and gives you an opportunity to set your host properties for upload and download. You can also set or modify your host properties from the Repository Administration tool. To do so, connect to a repository and click **Edit > Host Properties**.

If you would like to view the additional dialogs and specify additional settings, click *Next* from here.

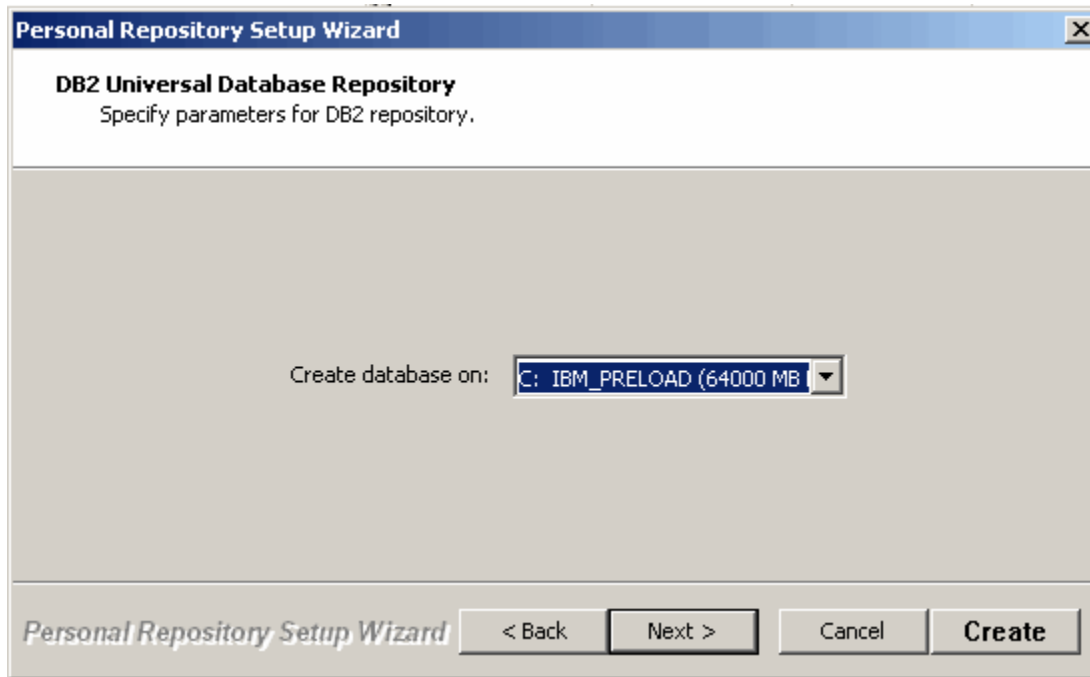
Connection Parameters dialog

Field	Description	More Information
Repository Name	The name of the repository you are creating	No more than 8 characters. The following names are not allowed: Default Empty AD AP HPS TURBO WP CP943 CP1253 Also, no special characters are allowed when creating a repository.
User ID and Password	User ID and Password of a current Windows Administrator using 8 or fewer characters	<i>DB2</i> : The user ID supplied is used to create a database in DB2/UDB, so the user ID must have administrative authority or be a part of a group with administrative authority in DB2/UDB. Any Windows user ID that is part of the administrator's group has sufficient privileges to create a Personal Repository. The user ID supplied in this field is the only repository user for this Personal Repository. The user ID is added to the AppBuilder security model. ¹ Use the same password with which the user ID logs into Windows. This password is verified by the Windows system and is used when connecting to the Personal Repository. The password <i>is</i> case sensitive. <i>SQL Server</i> : A valid SQL Server user name and password must be entered. For SQL Server, any user ID may be used, but it must first be defined with the "System Administrators" role in SQL Server. As shown in SQL database parameters The following table outlines topics to consider before setting up a Workgroup Repository with MS SQL., "sa" is the default administrator ID for an SQL Server installation. The user name and password are validated. A pop-up window is displayed if the user name or password is invalid.

Import Data	Imports AppBuilder default repository data	The Import Data field displays the default .EXP file used to populate the repository tables. When the Use Default check box is checked, this field is grey and the default file is used. If you uncheck this box, the Import Data field is activated and you can enter the path of your .EXP file or click the browse [...] button to navigate to the file location. In doing this, you populate the tables using your own .EXP file. See Importing an exported repository for more information.
Working Directory	Directory that stores temporary files during the Create action	The default temporary file location is C:\AppBuilder\Temp. You can change this temporary file location. To do so, create the new folder and click the browse [...] button to navigate to the new temporary file location. AppBuilder saves this new location as the default to use for additional newly created repositories. The size needed for your temporary location depends on the size of your archive file. If you are using the default EXP file we recommend allowing for 10MB of space.

1. DB2/UDB 7.1 and later allow you to create a user ID with more than 8 characters.

DB2 Parameters dialog



1. You can accept this default location or choose a different location from the drop-down list box.

DB2 Parameters

Field	Description
Database created on	Drive where the database files will be created. This is only for the DB2/UDB files pertaining to the database. This is <i>not</i> the AppBuilder home or Personal Repository supporting files directory. Make sure that the drive is a local hard drive. If the drive selected does not already have a DB2/UDB database created on it, the directory DB2 will be created in the root partition and the DB2/UDB files for the Personal Repository will be placed under that directory.

1. Click **Next** or **Create** to continue.
2. Click **Next** to continue, or click **Create** from here. If you clicked **Next**, the Host Configuration dialog displays.

Host Configuration dialog

Personal Repository Setup Wizard

Host Configuration
Please enter the host configuration parameters.

Host TCP/IP Parameters:

Host Name:

Port Number:

Timeout: (Seconds)

Host Repository Parameters:

Repository Name:

Version:

Project: Trace

Code Pages:

HOST:

PC:

Personal Repository Setup Wizard < Back Next > Cancel Create

3. Enter the host parameters for your mainframe machine and click *Next* or **Create**. If you click *Next*, a confirmation dialog displays.

Confirm Installation

Personal Repository Setup Wizard

Confirm Installation
Please confirm the personal repository installation.

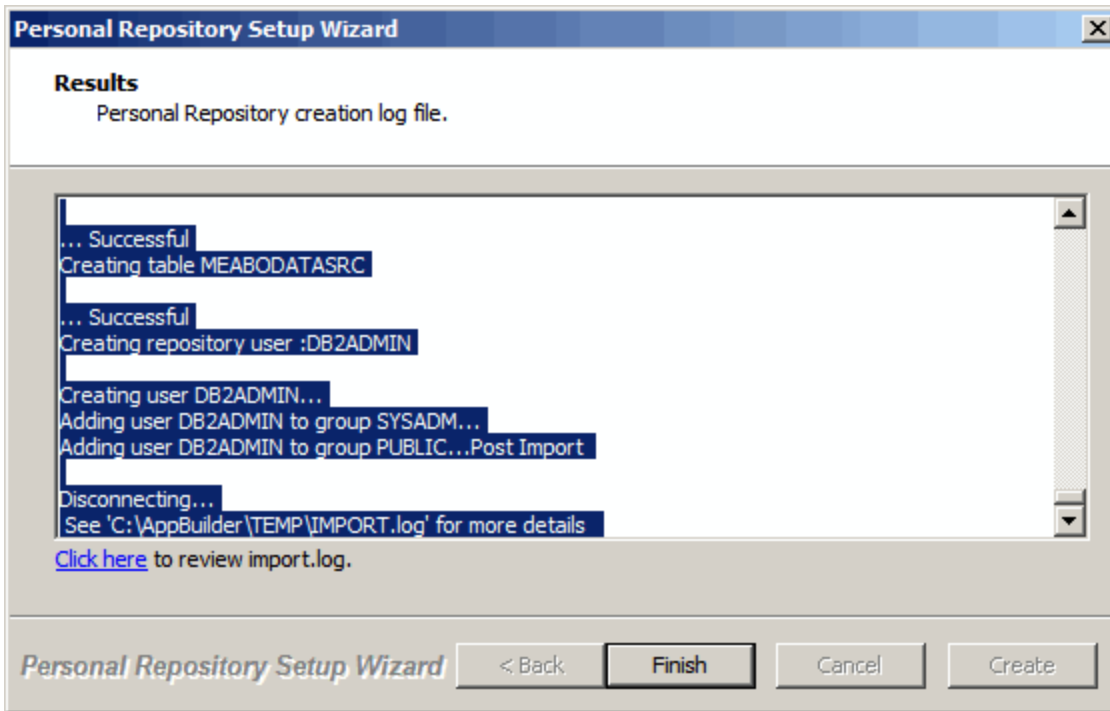
Personal Repository Setup Wizard is ready to create repository database


Press <Next> to start installation or press <Cancel> to exit

Personal Repository Setup Wizard < Back Next > Cancel

4. Click **Next** to create the Personal Repository. While the Repository Administration tool generates the Personal Repository, a status bar on the bottom of the window indicates its progress.
5. The Result window appears, indicating the creation status of the new repository. A log file is automatically displayed to indicate an error (**E**), warnings (**W**), and information (**I**). If an error is encountered and the Personal Repository fails to be created, the error message indicates the problem encountered. Use this information to correct the error and recreate the Personal Repository. At the bottom of the window you can also find a link to the **import.log** file, a link you can open using Notepad (see [Repository successfully created](#)).

Repository successfully created



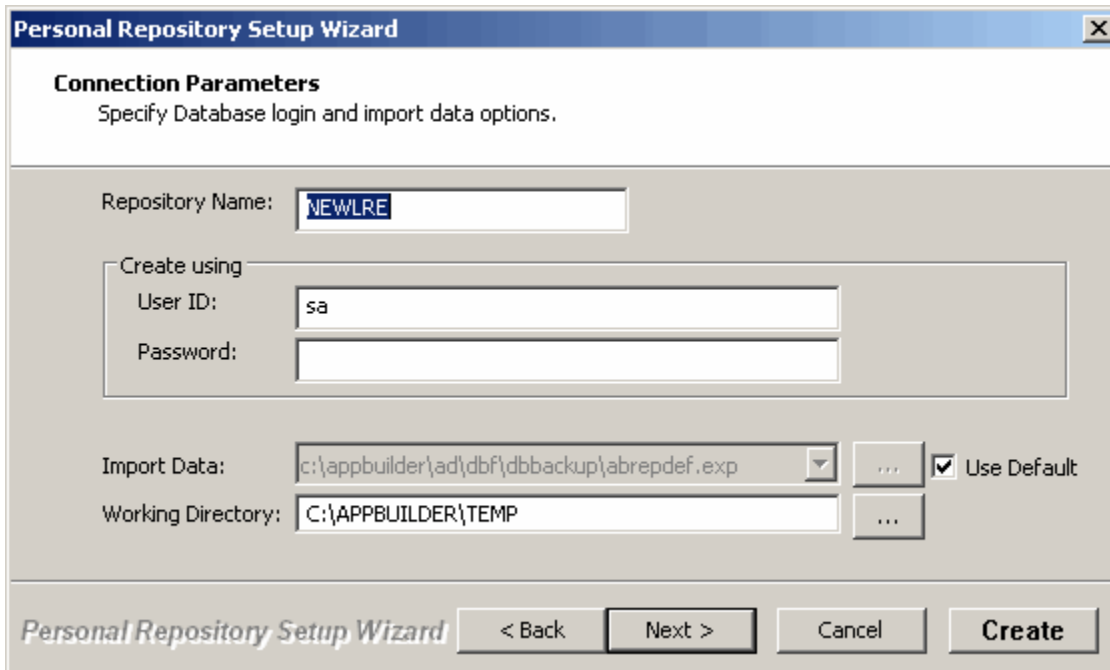
 Personal Repositories have an eight-character limitation for user IDs. However, you can create a user ID with more than eight characters using DB2/UDB 7.1 and later.

SQL Server

If you have chosen SQL Server as your database, the following screen displays. The following are the minimum required setup parameters when installing MSDE2000 for Personal Repository:

setup.exe SAPWD=<sa password> SECURITYMODE=SQL
 where the <sa password> is the password in quotes.

SQL database parameters



For a description of these fields, refer to [Connection Parameters dialog](#). It is possible to bypass additional dialogs and create the database directly

from this dialog. To do so, click **Create**.

The additional dialogs give you the opportunity to change the location where the database will be installed and give you an opportunity to set your host properties for upload and download. You can also set or modify your host properties from the Repository Administration tool. To do so, connect to a repository and click **Edit > Host Properties**.

If you would like to view the additional dialogs, click **Next** from the Connection Parameters dialog ([SQL database parameters](#)).

SQL Server Repository parameters

Personal Repository Setup Wizard

Microsoft SQL Server Repository
Specify parameters for MSSQL repository.

Data File: ... Size: 32 MB

Log File: ... Size: 8 MB

Truncate log on checkpoint

Personal Repository Setup Wizard < Back Next > Cancel Create

MSSQL repository setup fields

Field	Description
Data File	Specify the name and location of the database file to be created. Type or select the browse ... button.
Size	This option sets the initial size of the database. There is an eight-digit number limit.
Log File	Specify the location of the log file. Select the browse ... button to locate the log file.
Size	This option sets the initial size of the log file. There is an eight-digit number limit.
Truncate log on checkpoint	When checked, this will truncate the database transaction log from the checkpoint and back. It will not delete the active portion of the log.

You can accept these default values; however, if you choose to modify these parameters, do the following:

1. Select the Data File and the initial size of the database. Click the browse ... button.
2. Select the Log File and the initial size of the log file. Click the browse ... button.
3. Click **Next** to continue, or click **Create** from here.

If you click **Next**, the Host Configuration dialog displays.

Host Configuration dialog

Personal Repository Setup Wizard

Host Configuration
Please enter the host configuration parameters.

Host TCP/IP Parameters:

Host Name:

Port Number:

Timeout: (Seconds)

Host Repository Parameters:

Repository Name:

Version:

Project: Trace

Code Pages:

HOST:

PC:

Personal Repository Setup Wizard

4. Enter the host parameters for your mainframe machine. This project get added to the AppBuilder projects with public group authority. Then click **Next** or **Create** .
When you click **Next** , a confirmation dialog displays.

Confirm MSSQL repository installation

Personal Repository Setup Wizard

Confirm Installation
Please confirm the personal repository installation.

Personal Repository Setup Wizard is ready to create repository database

Press <Next> to start installation or press <Cancel> to exit

Personal Repository Setup Wizard

5. Click *Next* to start the installation.

Command Line Creation

You can also create a Personal Repository directly from the command line without going through the Repository Administration Tool and the Personal Repository Setup Wizard.

From the command line, enter:

```
RepoSetupCMD [<options>] -u <userid> -p <password>
```

where <options> are:

-v -> verbose, default is 'true'

-D -> for deleting repository
-s <DB server> -> database server name, defaults to (local) for SQL Server and defaults to instance name for DB2
-i <Import file> -> import data file, defaults to %HPSINI%\ad\df\ddbbackup\abreposef.exp
-w <Working dir> -> working directory, defaults to %TEMP%
-T <DBMS type> -> 1 - Sql Server, 2 - DB2, 3 - Oracle, defaults to Sql Server
-r <Repository> -> Repository name, defaults to NEWLRE

The following options apply to DB2 only:

-o <DB drive> -> for DB2, database created on, defaults to last used db2 data drive

The following options apply to SQL Server only:

-d <Datafile> -> Datafile name for SQL Server repository, defaults to default sql data dir

-n <Datafile size> -> Datafile size in MB, defaults to 32

-l <Logfile> -> Logfile name, defaults to default sql data dir

-z <Logfile size > -> Logfile size in MB, defaults to 8

-t <Truncate log> -> true or false, defaults to false

Examples:

To create a Personal Repository:

```
RepoSetupCMD -u sa -p sa
```

To create a Personal Repository named REPO1 using SQL Server as the database:

```
RepoSetupCMD -u sa -p sa -r REPO1 -T 1
```

To create a Personal Repository named REPODB using DB2 as the database:

```
RepoSetupCMD -u db2admin -p db2admin -r REPODB -T 2 -o c:
```

To delete a Personal Repository named REPO1:

```
RepoSetupCMD -u sa -p sa -D -r REPO1
```

Creating a Workgroup Repository

The Workgroup Repository Setup Wizard provides a user interface for you to set up a repository database. This section outlines how to create a database and remove an existing database using the Workgroup Repository Setup Wizard. Before you can use the wizard to create a database, you must have one of the following database software installed: SQL Server, DB2/UDB, or Oracle.

For an Oracle Workgroup Repository, you must first create the database using the Oracle tools. After you have created the database, the Workgroup Repository Setup Wizard creates the tables that are required for the repository.

You can click *Cancel* to close the Setup Wizard and configure the repository another time. If you choose to cancel at this time, later you can open the Setup Wizard from the Start menu. Click **Start > Programs > AppBuilder > Repository > Workgroup Repository Setup Wizard**.

The Repository Service must be stopped to set up a Workgroup Repository. The Setup Wizard prompts you before it automatically stops the service. Refer to [Working with the Service Control](#) or information about starting and stopping the Repository Service.

The following topics are included in this section:

- [Workgroup Repository Setup](#)
- [Creating an SQL Server Workgroup Repository](#)
- [Creating a DB2 Workgroup Repository](#)
- [Creating an Oracle Workgroup Repository](#)

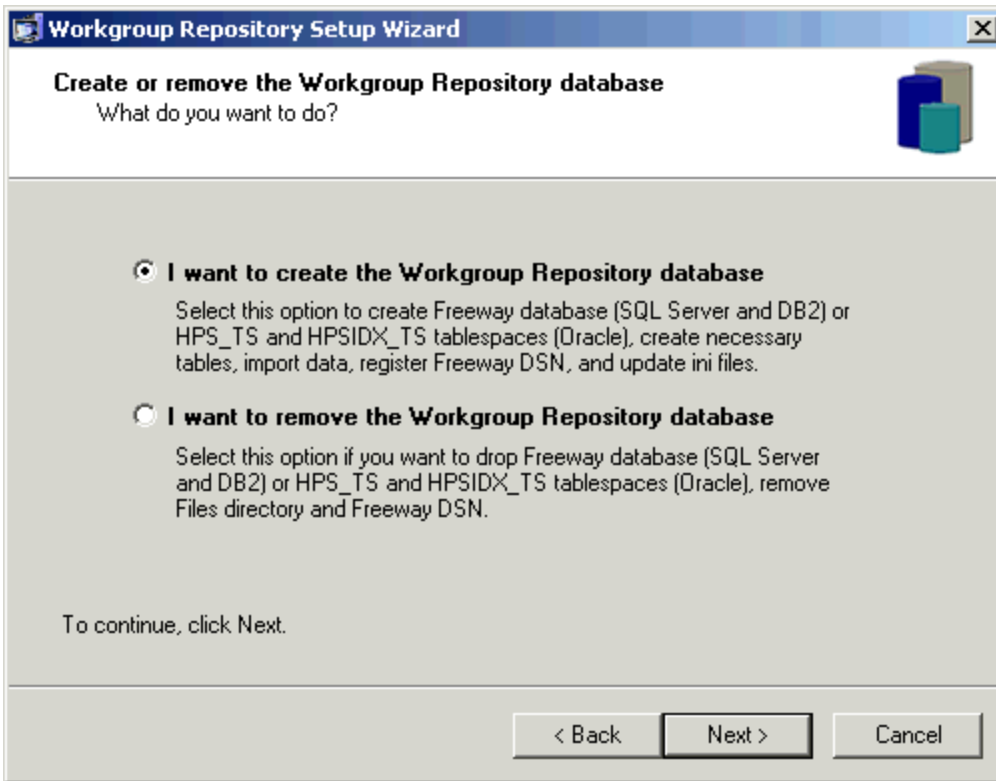
Workgroup Repository Setup

To set up the repository, do the following:

1. From the Workgroup Repository Setup Wizard Welcome dialog, click* **Next** to begin setting up the repository.

If a Workgroup Repository already exists on the machine, [Create or remove a Workgroup Repository database](#) is displayed. If a repository is not present, [Set the Freeway ID](#) is displayed.

Create or remove a Workgroup Repository database

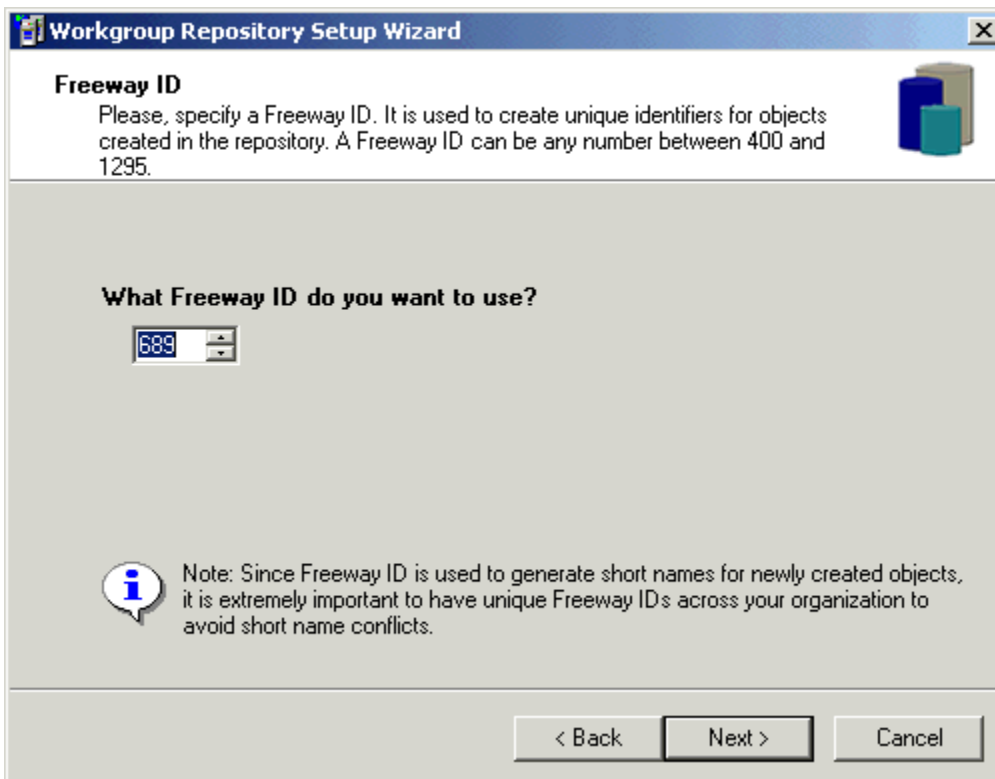


The wizard determines if the Workgroup is installed by checking for a FREEWAY DSN entry or for subdirectories in the `AppBuilder\Files` directory. From here, you can remove the Workgroup Repository database or create a new Workgroup Repository database.

To create a new database, take the following steps:

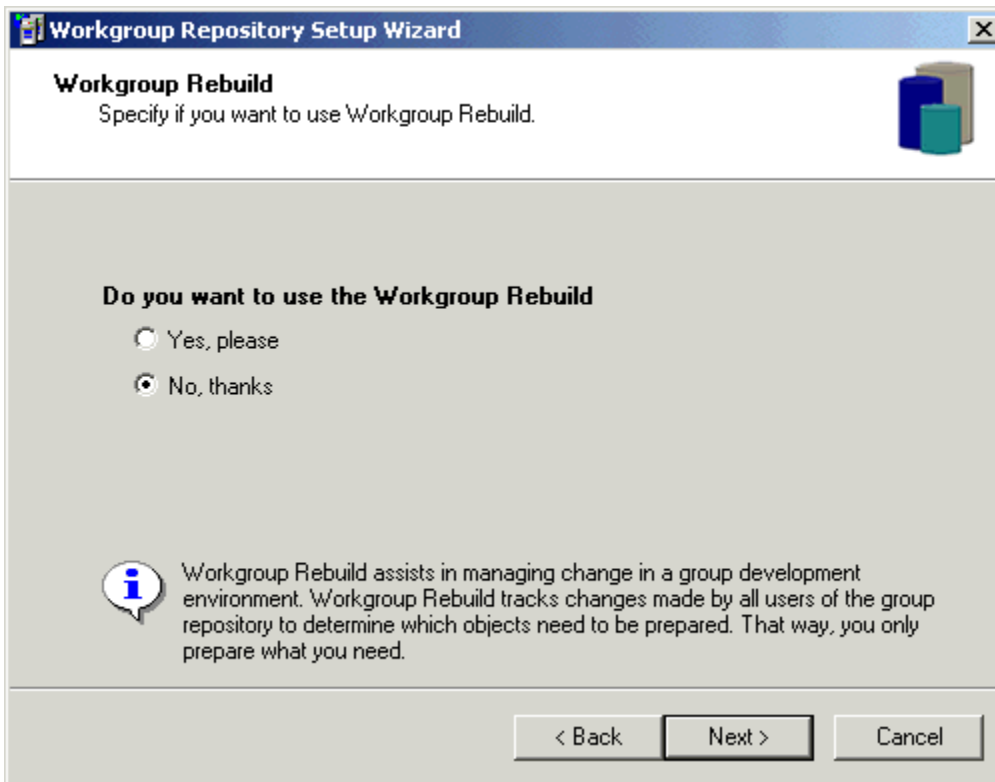
1. On [Create or remove a Workgroup Repository database](#), select **I want to create a Workgroup Repository database**.
2. Click **Next**.

Set the Freeway ID



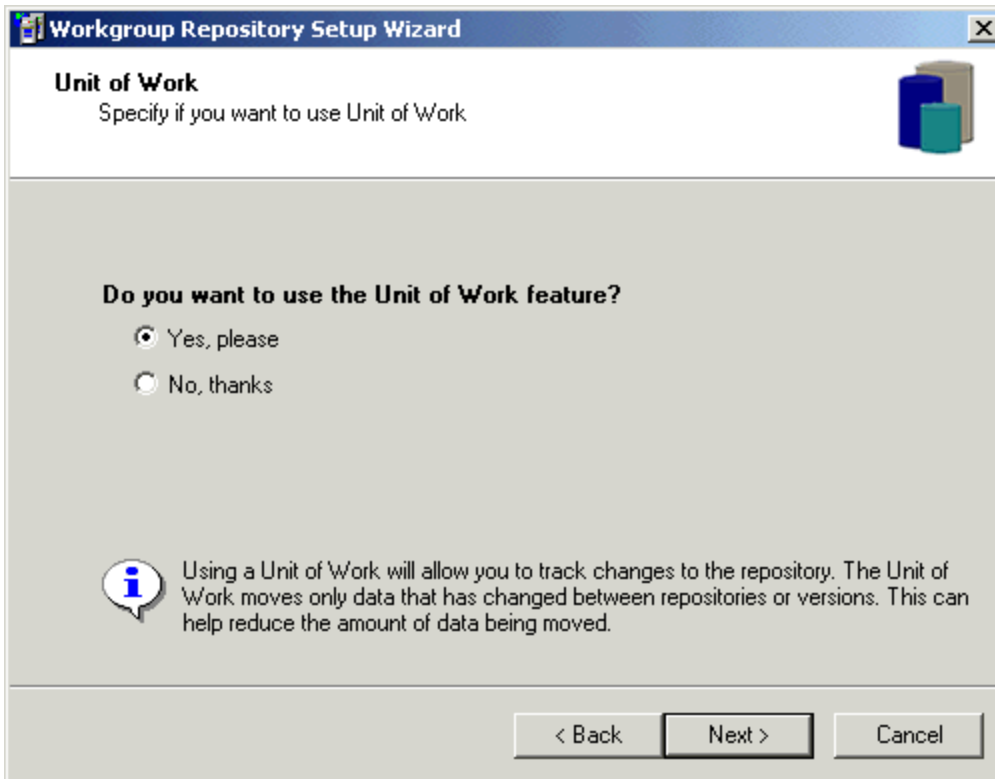
3. If the Freeway.ID file exists on the machine, the value from the file displays in the selection box. The Freeway ID is validated when you select **Next** . If you enter a number that is outside the valid range for the Freeway ID, the wizard displays a message box indicating an invalid ID has been entered.
Because the Freeway ID is used to generate short names for newly created objects, it is extremely important to have unique Freeway IDs across your organization to avoid short name conflicts.
If you wish to terminate the setup at this point, select **Cancel** . Changes are not saved if you select **Cancel** .
4. Click **Next** to continue.

Workgroup Rebuild



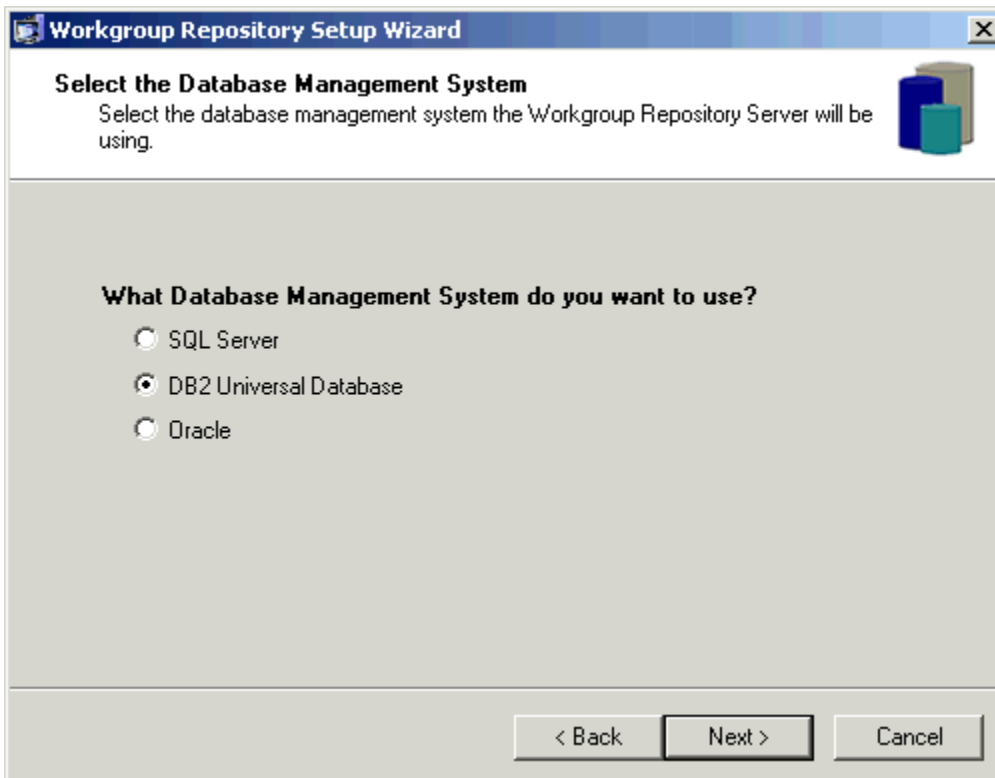
5. Make your selection here for whether you want the Workgroup Repository Rebuild feature. Rebuild allows you to prepare only portions of your application, reducing time and resources for a prepare. See [Workgroup Repository Rebuild](#) for more details.
6. Click **Next** to continue.

Unit of Work



7. Make your selection here for whether or not you want the Workgroup Repository Unit of Work (UOW) feature. UOW allows you to track your changes in the repository. Use this feature during repository migrations reducing the amount of data being moved.
8. Click **Next** to continue.

Choose the Database Management System



9. Select which database you want to configure:
 - SQL Server – Refer to [Creating an SQL Server Workgroup Repository](#).
 - DB2 Universal Database – Refer to [Creating a DB2 Workgroup Repository](#).

- Oracle – Refer to [Creating an Oracle Workgroup Repository](#).
The setup wizard automatically detects if you have database software installed on your local machine or if you have the software installed to connect to a remote database. If a database is not installed on the machine, the radio buttons are disabled. To install database software, refer to the specific software documentation.
If you wish to terminate the setup at this point, select **Cancel** . Changes are not saved if you select **Cancel**.

10. Click **Next** to continue.

Creating an SQL Server Workgroup Repository

The following table outlines topics to consider before setting up a Workgroup Repository with MS SQL.

Considerations for MS SQL

Consideration	Description
Character Set	The system administrator can set the codepage to whatever your organization is using. Be careful in setting up the environment, however, because database backups are sensitive to the character set of the SQL Server on which they are restored. For example, a repository database archived from a 437 US character set cannot be restored on an 850 English character set because of character set incompatibilities. Instead, use the Archive/Restore method to transfer objects.
Sort Order	The only supported sort order is Dictionary order-case insensitive. Selecting the correct character set and sort order during the MS SQL Server install procedure is critical. If the selected character set or sort order needs to be changed, rebuild the databases and reload the data.
MS SQL Server Service	The MS SQL Server service must be up and running before you start the Freeway Manager Service.

SQL Server parameters

SQL Server Setup Fields

Field	Description
User name	A valid SQL Server user name and password must be entered. For SQL Server, any user ID may be used, but it must first be defined with the "System Administrators" role in SQL Server. As shown in The following table outlines topics to consider before setting up a Workgroup Repository with MS SQL. , "sa" is the default administrator ID for an SQL Server installation. The user name and password are validated. A pop-up window is displayed if the user name or password is invalid.
Password	

Server	Identifies the machine that SQL Server is running on. The default value is local , which identifies the machine the user is currently logged into. The field can be left blank, have (local) , or have a "." to designate the local machine. It is also possible to enter a remote machine name. The button next to the field will bring up a list of available servers. The server button only appears if the SQLDMO files for SQL Server are present on your machine. The server is validated when you click Next . If an invalid server name is entered, a message displays. The server name must be valid in order to continue to the next dialog.
Create Database	This option is selected by default. This option creates the FREEWAY database and all of the Workgroup tables. Deselect this option if the FREEWAY database already exists. The drive selection displays the drive letter and free space on the drive. You can specify your local drive. When you click Next , the setup checks to see if the FREEWAY database already exists. If the database exists, a message is displayed. To create a new database, you can not proceed if the FREEWAY database already exists. You must delete the existing FREEWAY database (see Creating a Repository Alias) or deselect the Create database check box.
Express setup (SQL Server only)	Check this option to accept a non-customized typical installation.
Custom setup (SQL Server only)	Check this option to customize your database installation.
Import Data	The default repository is ABREPDEF.EXP for a workgroup repository installation. This box is checked to populate the repository tables. If you uncheck this box, the repository tables will remain empty. You can then populate the tables using an .EXP file. See Importing an exported repository for more information.
Working directory	Enter the path for your working directory. This is a temporary space that is used during the creation of the repository. The log files are stored in this location on repository creation, repository import & repository export. The files are import.log or export.log. It is possible to change the Working Directory. Use the browse ... button to select a new directory.

Express Setup

An Express Setup is a typical setup. If Express Setup is selected in [The following table outlines topics to consider before setting up a Workgroup Repository with MS SQL.](#), the next dialog displayed is [SQL Server Express Setup](#).

SQL Server Express Setup

The screenshot shows the 'Workgroup Repository Setup Wizard' dialog box. The title bar reads 'Workgroup Repository Setup Wizard'. The main heading is 'Specify Data and Transaction Log Files'. Below the heading, it says 'Specify the name of files for the database data and transaction log. For each file, specify initial size.' There is a small icon of a database cylinder to the right. The dialog is divided into sections for 'Data' and 'Transaction log'. Under 'Data', there is a 'File name:' field with 'c:\freeway.mdf' and a browse button (...), and a 'File size:' field with a spinner set to '32' and 'MB'. Under 'Transaction log', there is a 'File name:' field with 'c:\freeway.ldf' and a browse button (...), and a 'File size:' field with a spinner set to '8' and 'MB'. At the bottom, there is an 'Options' section with a checked checkbox 'Truncate log on checkpoint'. At the very bottom, there are three buttons: '< Back', 'Next >', and 'Cancel'.

SQL Server Express Setup Fields

Field	Description
Data File name	Specify the name and location of the database file to be created. Type or select the browse ... button.
File size	This option sets the initial size of the database. There is an eight-digit number limit.
Transaction Log File name	Specify the location of the log file. Select the browse ... button to locate the log file.
File size	This option sets the initial size of the log file. There is an eight-digit number limit.
Truncate log on checkpoint	This will truncate the database transaction log from the checkpoint and back. It will not delete the active portion of the log.

Custom Setup

The Custom Setup option gives you the ability to specify the database files you will be using to set up the Workgroup Repository. It also allows you to specify the file growth parameters. If you select Custom Setup on [SQL Server parameters](#) [The following table outlines topics to consider before setting up a Workgroup Repository with MS SQL.](#), the next dialog displayed is [SQL Server Custom Setup](#).

SQL Server Custom Setup

Workgroup Repository Setup Wizard

Name the Database Files
Specify the name of one or more files within which the database is contained. Please, specify the initial file size and file growth parameters for each of the files.

Database files

File name	Location	Initial size (MB)
freeway_data	c:\freeway.mdf	32

Automatically grow file

File growth

In megabytes: 1

By percent: 10

Maximum file size

Unrestricted filegrowth

Restricted filegrowth (MB): 33

< Back Next > Cancel

SQL Database Files Custom Setup Fields

Field	Description
Database files	Using this table, it is possible to split the database across multiple files. For each file you can specify the name, location and initial size. You must have at least one file specified. To add additional files, select the next empty row.
File name	When you enter a File name the system automatically updates the Location. By default each file is stored in the <i>Data</i> directory where SQL Server is installed.
Location	Change the directory by typing the name in the Location field or click the Browse button ... that appears when the cell is selected. The first file in the list has an extension of MDF. This is the primary database file. Additional files that are added have an NDF extension which denotes that the file is secondary.
Initial size	An integer value must be entered into the Initial size column for each database file.

Automatically grow file	This option is set for each file that is listed in the Database file table. When this option is selected, you can select different attributes associated with the selected database file. The following values can be changed: <ul style="list-style-type: none"> • File growth • Maximum file size
File growth	When the database file reaches the maximum defined limit, the size of the file increases. You have the option of determining how the file will increase: by megabytes or by a percentage.
Maximum file size	The file size grows to maximum size that the disk can hold (Unrestricted filegrowth). Or it grows to the upper limit that is specified under Restricted filegrowth .

After you have specified the custom settings for the database files, click **Next** to continue.

The next dialog is similar to [SQL Server Custom Setup](#); this time you are customizing the transaction log files.

SQL Server Transaction Log Files Custom Setup

Workgroup Repository Setup Wizard

Name the Transaction Log Files
Specify the name of one or more files that transaction log is contained within. Also, please specify the initial file size and file growth parameters for each of the files.

Transaction log files

File name	Location	Initial size (MB)
freeway_log	c:\freeway.log	8

Automatically grow file

File growth

In megabytes: 1

By percent: 10

Maximum file size

Unrestricted filegrowth

Restricted filegrowth (MB): 9

Truncate log on checkpoint

< Back Next > Cancel

SQL Server Transaction Log Files Custom Setup Fields

Field	Description
Database files	Using this table, it is possible to split the log file across multiple files. For each file you can specify the name, location, and initial size. You must have at least one file specified. To add additional files, select the next empty row.
File name	When you enter a File name the system automatically updates the Location . By default each file is stored in the Data directory where SQL Server is installed.
Location	Change the directory by typing the name in the Location field or click the Browse button ... that appears when the cell is selected.
Initial size	An integer value must be entered into the Initial size column for each database log file.

Automatically grow file	This option is set for each file that is listed in the Transaction Log File table. When this option is selected, you can select different attributes associated with the selected log file. The following values can be changed: <ul style="list-style-type: none"> • File growth • Maximum file size
File growth	When the log file reaches the maximum defined limit, the size of the file increases. You have the option of determining how the file will increase: by megabytes or by a percentage.
Maximum file size	The file size grows to maximum size that the disk can hold (Unrestricted filegrowth). Or it grows to the upper limit that is specified under Restricted filegrowth .
Truncate log on checkpoint	This will truncate the database transaction log from the checkpoint and back. It will not delete the active portion of the log.

After you have specified the custom settings for the Transaction Log files, click *Next* to continue. The system displays a confirmation dialog confirming that the wizard is ready to create the Workgroup Repository database.

Click **Next** to continue.

When the wizard is finished and the repository is created, click **Close**.

Creating a DB2 Workgroup Repository

Use the settings in the Workgroup Repository Setup Wizard to create a workgroup repository.

DB2 Parameters

DB2 Setup Fields

Field	Description
User name	A valid DB2 user name and password must be entered. A message displays if the user name or password is invalid. UDB uses the operating system security.
Password	

Instance	Identifies the instance of DB2/UDB used to create the Workgroup Repository database. The database instance can be local or remote . You must define a remote database instance in DB2/UDB before running the setup wizard. If a remote instance is selected, you must specify the disk drive in the edit field under the "Create database" check box. For Unix systems, the user must specify a directory path. If the edit field is left blank the Workgroup Repository database is created in a default location that is set by DB2/UDB.
Create database	This option is selected by default. This option creates the FREEWAY database and all of the Workgroup tables. Deselect this option if the FREEWAY database already exists. The drive selection displays the drive letter and free space on the drive. You can specify your local drive. When you click Next , the setup checks to see if the FREEWAY database already exists. If the database exists, a message is displayed. To create a new database, you can not proceed if the FREEWAY database already exists. You must delete the existing FREEWAY database (see Creating a Repository Alias) or deselect the Create database check box.
Import data	The default repository is ABREPDEF.EXP for a workgroup repository installation. This box is checked to populate the repository tables. If you uncheck this box, the repository tables will remain empty. You can then populate the tables using an .EXP file. See Importing an exported repository for more information.
Working directory	The value in the Working directory field is read from the TEMPORARY value in the FREEWAY section of the HPS.INI file. The directory is used to store files during the creation of the database and import of the data. The files stored in the working directory are removed when the setup is complete. You can specify a different working directory by typing in a new directory or click the browser ... button. The directory is validated when you click Next .

Click **Next** to continue. The system displays a confirmation dialog confirming that the wizard is ready to create the repository database. When the database is created, click **Close** to close the setup wizard.

Creating an Oracle Workgroup Repository

Follow the steps:

1. From the window [Choose the Database Management System](#), select Oracle as the database management system. The Oracle Workgroup Repository Setup window displays.

Oracle Parameters

2. Specify the appropriate setup parameters and click *Next* .

Oracle Setup Fields

Field	Description
User name	A valid Oracle user ID and password must be entered. This user ID must be set up in Oracle with the appropriate authority. A message displays if the user name or password is invalid.
Password	
Host string	The name of the connection to your Oracle database where you are creating the repository. Use the Oracle tools to set up a database before you start the Workgroup Repository Setup Wizard.
ODBC driver	Choose between: * Oracle ODBC Driver (Oracle 8i) * Oracle in OracleHome92 (Oracle 9i) * Microsoft ODBC for Oracle We recommend using native Oracle ODBC drivers.
Create database	This option is selected by default. With Oracle, this option creates the tablespaces that are used by AppBuilder to hold the repository. The administrator creates the database beforehand. AppBuilder creates the following objects: * HPS_TS tablespace * HPSIDX_TS tablespace * USERID user * HPSFWY user * HPS_USER_ROLE role * necessary tables and indexes
Import data	The default repository is ABREPDEF.EXP for a workgroup repository installation. This box is checked to populate the repository tables. If you uncheck this box, the repository tables will remain empty. You can then populate the tables using an .EXP file. See Importing an exported repository for more information.
Working directory	The value in the Working directory field is read from the TEMPORARY value in the FREEWAY section of the HPS.INI file. The directory is used to store files during the creation of the database and import of the data. The files stored in the working directory are removed when the setup is complete. You can specify a different working directory by typing in a new directory or clicking the browser ... button. The directory is validated when you click Next .

The second Oracle parameters dialog displays.

1. Click on each table cell to edit the filename, path, and initial size of each file.

Specify Data Files Setup Window

Specify datafiles for HPS_TS tablespace
Specify the name of one or more datafiles for HPS_TS tablespace . Please, specify the initial file size and file growth parameters for each of the datafiles.

Database files

File name	Location	Initial size (MB)
hps_ts1.dbf	C:\ORACLE\ORADATA\FREEWAY\	20
hps_ts2.dbf	C:\ORACLE\ORADATA\FREEWAY\	20
hps_ts3.dbf	C:\ORACLE\ORADATA\FREEWAY\	20

Automatically extend datafile when full (AUTOEXTEND)

File growth
Increment: MB

Maximum file size
 Unrestricted file growth
 Restricted file growth (MB):

< Back Next > Cancel

Database File Parameters

Field	Description
Database files	Click on each table cell to edit the filename, path, and initial size of each file. Edit the path when you want to split the files across drives for better performance.
Automatically extend datafile	Select this option when you want the selected data file to automatically expand as needed. You must set this option for each data file individually.
File growth (Increment)	Specify the increments at which you want the data file to expand.
Maximum file size	Choose between: * Unrestricted file growth - Allows the data file to expand without restriction. * Restricted file growth - Restricts the data file growth to the specified size.

1. Click **Next** to display the data files setup for HPSIDX_TS tablespace.
2. Click **Next**.
The system displays a confirmation message before you create the database.
3. Click **Next** to continue.

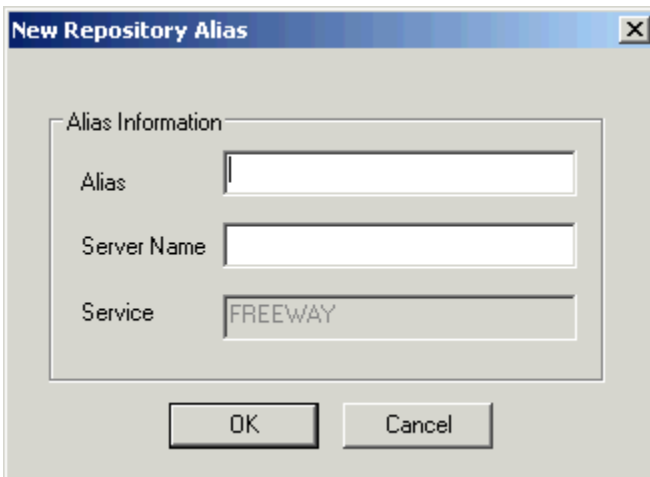
The system creates the database and gives you an opportunity to review the Fwyssetup.log and the Import.log files.

Creating a Repository Alias

Follow the steps:

1. To create a Repository alias, select **Repository > New > Repository Alias** .
The New Repository Alias window displays.

New Repository Alias window



You must obtain the server machine Type and the Server Name from your system administrator. Select the Workgroup server Type and Server Name that matches the server to which you are connecting.

2. Provide the following information:
 - Type the alias name of your choice
 - Type the name of the server machine to which you are connecting
 - The default value for Service is Freeway.
3. Click **OK** .
After creating a connection alias, the following security information is required to connect to the repository:
 - user ID - the assigned user name for the repository
 - password - the password as determined by native security. Refer to [Defining Native Security](#).

You are now able to connect to the Workgroup server machine using your assigned User ID and the password you provided. The next time you open the Repository Administration tool or Construction Workbench, the system prompts you for the repository name that you want to connect to and your respective User ID and password.



AppBuilder supports the Microsoft Windows 2000 and Windows Server 2003 implementation of Named Pipes. This software provides communications support. The named pipes can be transported on either NetBEUI or TCP/IP (using the TCP/IP NetBIOS Helper Service.)

Deleting a Repository

This section outlines the steps you take to delete a repository. The following topics are included in this section:

- [Deleting a Personal Repository](#)
- [Deleting a Workgroup Repository](#)

You can also delete a repository from the command line. For details of command-line operations, see [Command Line Creation](#).

Deleting a Personal Repository

Deleting the Personal Repository removes it from the local machine. All files related to the repository and the DB2/UDB database are removed. The Personal Repository name is also removed from the list of personal repositories. Complete the following steps to delete a Personal Repository:

1. From the Repository Administration tool, click **Repository > Delete > Repository** . The Personal Repository List window displays.

Personal Repository List

Available Personal Repositories:

Name
PERS1
PERS2
NEWLRE
TESTLRE
TEST2

User Id:

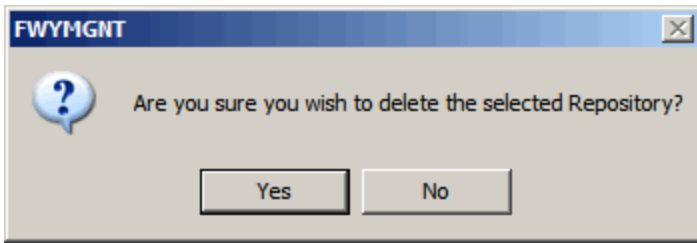
Password:

Database Type:

2. Select the repository to be deleted and provide the user ID and password for that repository.
3. Click **Delete**.

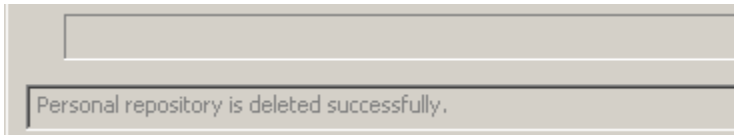
A message appears like in the following figure:

Message when deleting a repository



4. Click **Yes** to delete the personal Repository, or click **No** if you don't want to delete it.
5. The result of the deletion is displayed on the status bar.

Delete repository complete



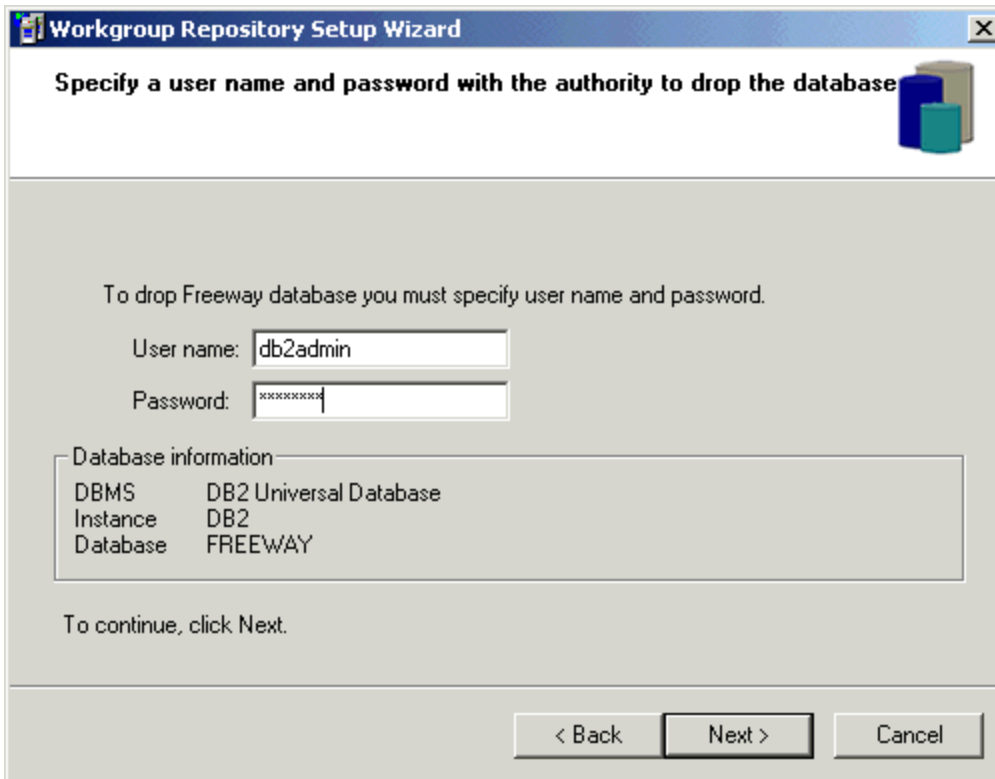
6. Click **Close** to close the dialog.

Deleting a Workgroup Repository

To remove a Workgroup Repository, take the following steps:

1. From the Repository Administration tool, select **Repository > Delete > Repository**. The Repository Service has to be stopped before it will allow you to remove the Workgroup Repository. The setup wizard prompts you to do this. Refer to [Working with the Service Control](#) for more information. After the service has stopped, the Workgroup Repository Setup Wizard displays.
2. On [Create or remove a Workgroup Repository database](#), select **I want to remove the Workgroup Repository database**. With this option, the setup wizard cleans the AppBuilder\Files directory. Depending on the DSN entry, it knows whether DB2, SQL Server, or Oracle is being used. For DB2 and SQL, it drops the FREEWAY database, and removes the DSN entry. For Oracle, it drops the following:
 - HPSFWY user
 - USERID user
 - HPS_USER_ROLE role
 - HPS_TS tablespace
 - HPSIDX_TS tablespaceIf the Oracle instance is local or if you are using Oracle version 9.0 (or above), the wizard removes data files used by HPS_TS and HPSIDX_TS tablespaces. The wizard removes the AppBuilder\Files directory and deletes the DSN entry for the repository database.
3. Click **Next**. The wizard asks you for a user ID and password. The user ID and password entered must have the appropriate authority to remove the database. This dialog displays the DBMS type and database name.

Removing Database – Enter User ID and Password



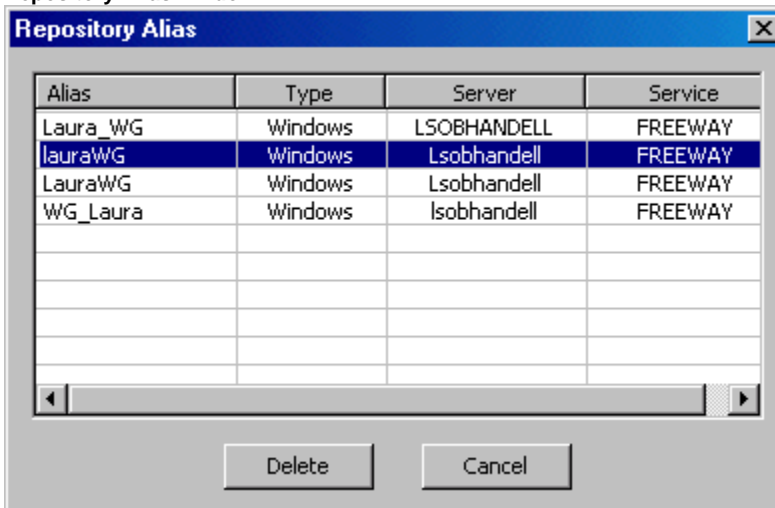
4. Click **Next**. The wizard displays a confirmation dialog, confirming that you want to remove the Freeway database, remove the Files directory, and the Freeway DSN.
5. Enter the appropriate password.
6. Click **Next** to continue. The wizard confirms that the database was successfully removed.
7. Click **Next** and then click **Close** to close the setup wizard.

Deleting a Repository Alias

To remove an alias connection to a server machine, take the following steps:

1. From the Repository Administration tool, click **Repository > Delete > Repository Alias**. The Repository Alias window displays.

Repository Alias window



2. Highlight the alias to delete and click **Delete**. Press Ctrl to select more than one alias. The system verifies that you want to delete the selected aliases.
3. Click **Yes** to complete the delete action.

Allowing Local Users to Connect to a Personal Repository

Administrative rights are required to create a Personal Repository. However, members of the local Windows Power Users group can use a Personal Repository if the Power Users group has read, write, and execute permission to the drive upon which AppBuilder is installed. Therefore, the first step in allowing a user to connect to a Personal Repository is to add the user to the Power Users group on the local machine. See your Windows documentation for steps on how to do this.

After the user is setup on the Windows machine as a power user, you must add the user to the database through the particular DBMS system used for the Personal Repository.

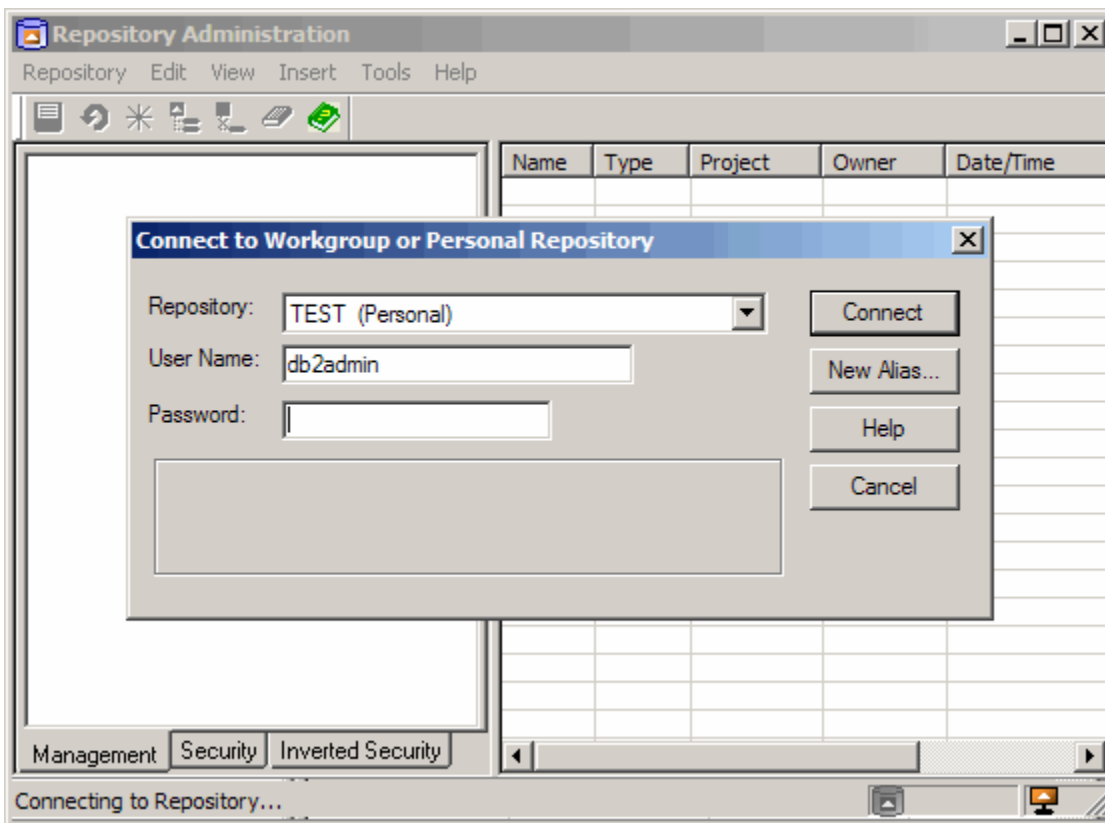


See the documentation for the DBMS for the procedure to add a user to the database.

Once the power user is set up in the database, you must add the user to an AppBuilder security group. Take the following steps to add a user to a security group using the Repository Administration tool:

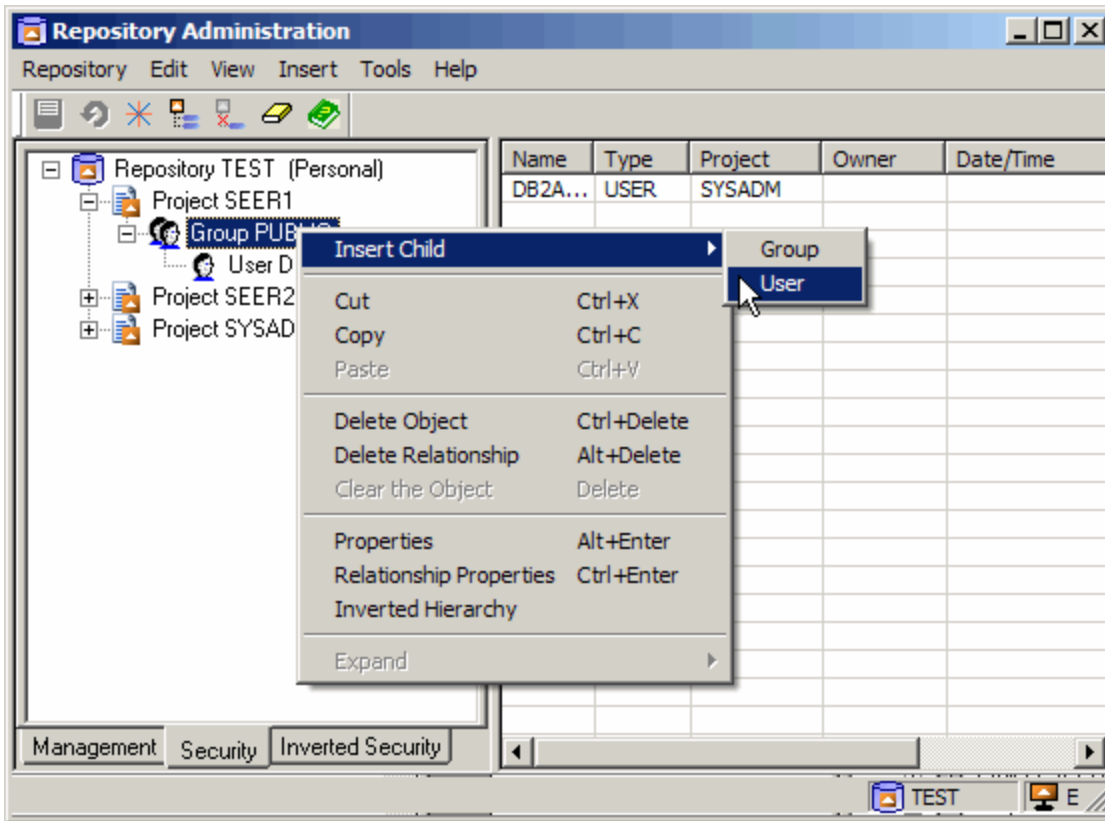
1. Open the Repository Administration tool and connect to the repository with which you have been working using the administrator ID.

Connection to Repository Administration tool as an Administrator



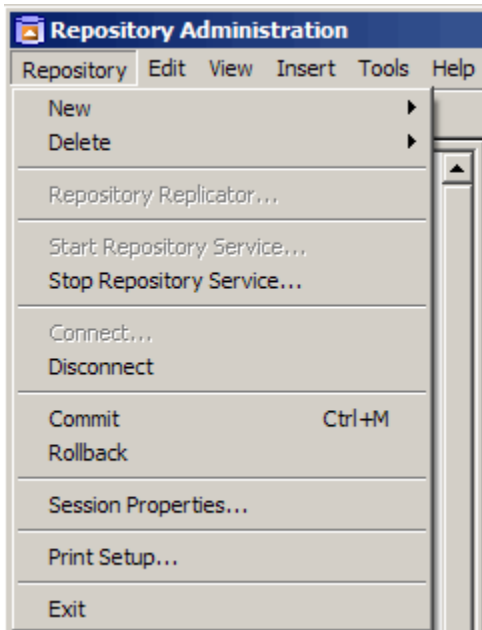
2. Select the *Security* tab, and add the user to a group that is defined in that repository. You can create a new group if necessary. Security permissions for the user are specified at the group level. For more information about setting up users and groups, refer to [Managing Security](#).

Adding a user to the Security tab



3. Commit the changes and disconnect from the Repository Administration tool.

Commit changes and disconnect



Full Repository Migration

The Repository Migration utility allows the repository administrator to migrate all data and files to or from the repository. This utility differs from migration import and export in that a migration import or export moves selected objects, whereas the Repository Migration utility moves all repository data in its entirety.

The Repository Migration utility extracts the contents of the repository tables into a database and platform-independent format that is transferred

between or across platforms.

The following table illustrates the Repository Migration support between different types of AppBuilder repositories.

Repository Migration Support

Source	Target	Actions
Personal Repository Archive (DB2)	Personal Repository	<ul style="list-style-type: none">• Truncates tables and imports data
Personal Repository Archive (DB2)	Workgroup Repository	<ul style="list-style-type: none">• Truncates tables and imports data• Removes UOW Personal• Creates UOW Workgroup
Personal Repository export file	Personal Repository	<ul style="list-style-type: none">• Truncates tables and imports data
Personal Repository export file	Workgroup Repository	<ul style="list-style-type: none">• Truncates tables and imports data• Removes UOW Personal• Creates UOW Workgroup• Performs a database-independent import
Workgroup Repository export file	Personal Repository	<ul style="list-style-type: none">• Truncates tables and imports data• Creates UOW Personal
Workgroup Repository export file	Workgroup Repository	<ul style="list-style-type: none">• Imports data

The following topics provide more information about using the Repository Migration utility:

- [Invoking the Repository Migration Utility from the Command Line](#)
- [Using Repository Export](#)
- [Importing an exported repository](#)

Invoking the Repository Migration Utility from the Command Line

To invoke the Repository Migration utility, at the command line type:

```
fwyutil -e | -i | -k | -r -n<exp filename> [-w<work_dir>] [-l<log>] -u<userID> -p<password> [-c<#>]
```

where:

- < *exp filename* > is the name of the export file exported to .
- < *work_dir* > is a temporary directory used to store files created during the export operation. If this directory is not specified, the files are stored in the current working directory.
- < *log* > is the name of the log file where information about the import or export is stored. If the log name is not specified, the information is stored in either import.log or export.log in the current working directory.
- < *userID* > is the userID
- < *password* > is the user password
- < *#* > is the number of object inserts between database commit operations

The command line switches are described in [Switches and Functions](#).

Switches and Functions

Switch	Function
-e	Perform an export
-i	Perform an import
-k	Pack the database
-r	Rebuild indexes

-n	Full file path & file name for Import/Export compressed data file
-w	Working directory for temporary files created/deleted by fwyutil
-l	Log file name for the fwyutil Import/Export log
-u	UserID for the Workgroup Administrator
-p	Password for the Workgroup Administrator
-c	The number of inserts before a database commit (optional). Use this option if you have a large repository but a small amount of space allocated to the transaction log. This option enables database cleaning during the fwyutil import by triggering a database commit after a specified number of object inserts.

To import data from a file called MYEXPORT.EXP in the root directory using a working directory called D:\TEMP, type:

```
fwyutil -i -nD:\MYEXPORT.EXP -wD:\TEMP -lD:\MYIMPORT.LOG -u<userID> -p<password>
```

To export data into a file called MYEXPORT.EXP in the root directory using a working directory called D:\TEMP, type the following (substituting your own userID and password for UID and PW):

```
fwyutil -e -nD:\MYEXPORT.EXP -wD:\TEMP -lD:\MYEXPORT.LOG -u<userID> -p<password>
```

Using Repository Export

The Repository Export utility creates the data files and then compresses them into a single file with an .exp extension. The Repository Export includes all data from the source repository, while a migration, for example, can be selective in terms of the repository contents. The data in the .exp file is written in a format that can be read by the repository management software on other operating system platforms.



Make sure that no changes are being made to the repository during the export operation

To export data, take the following steps:

1. From the Management tab, right-click **Management** and click **Repository Export**. Or, right-click **Repository Export** from the Management tree and select **Repository Export**.
The Repository Management Wizard: Export displays.

Migration Management Wizard: Export

Repository Management Wizard : Export

Repository Export
Specify file and working directory for repository export.

File: ...

Working Dir: ...


To continue, click Next

Repository Management Wizard < Back Next > Cancel

- Using the browse button [...], create an export file or navigate to an existing export file. The export filename can be no more than 8 characters.
- Specify the Working Directory. The default Working Directory is C:\AppBuilder\Temp.
- Click **Next**.
The system notifies you that the export process is ready to begin.
- Click **Next** to begin the export.
The Repository Service must be stopped before using Repository Export. If the service is not stopped, the Stop Repository Service window displays. Refer to [Working with the Service Control](#) for more information.
Information about the export operation is written to the following file:
<drive>:\appbuilder\temp\export.log
where <drive> is the drive on which the repository management software is installed. This log file is stored in the temporary folder specified above, c:\appbuilder\temp\export.log. Subsequent export operations overwrite the information in this file. Refer to [Reviewing the Log File](#) for more information about the log file.

Importing an exported repository

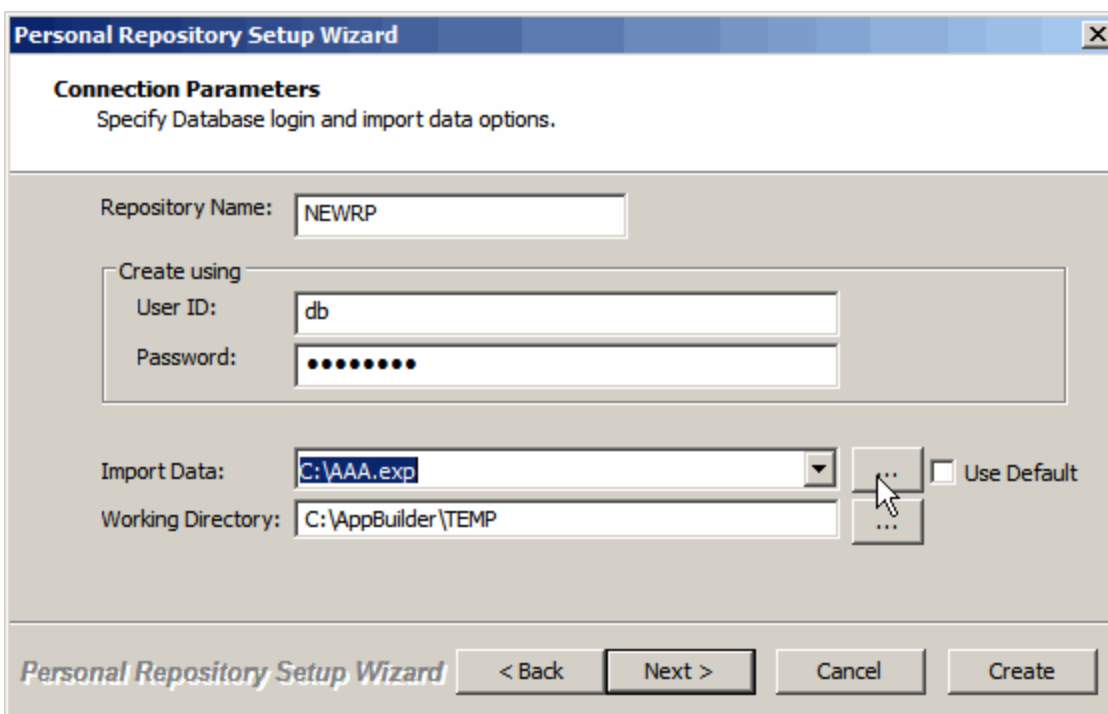
A repository export produces a .exp file which includes the contents of the repository tables and associated flat files with *rule source code* and *window panel* information. Before using, see [Using Repository Export](#).

 This operation overwrites any data in the existing repository.

In order to recreate a previous exported repository into a new one, follow the steps described below:

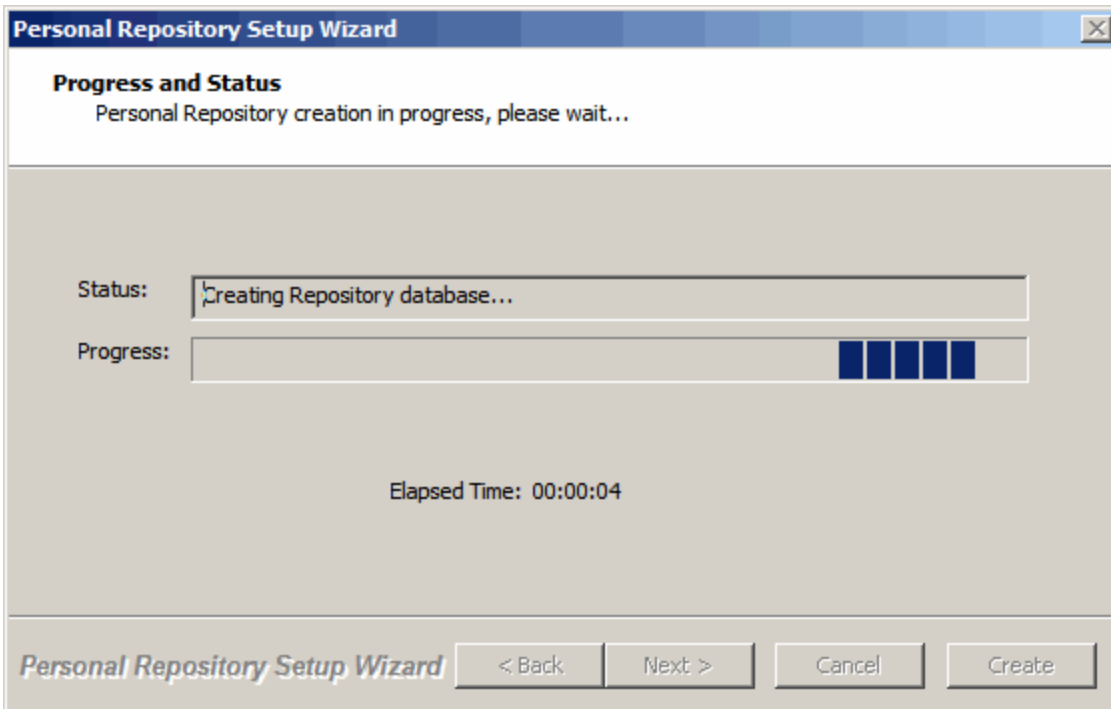
- From the Repository Menu, choose **New > Repository**. The Personal Repository Setup Wizard appears.
- From the drop-down list, choose the Database management system you desire, and click **Next**. The Connection Parameters window displays:

Connection Parameters window



- Type the name of the new repository in the Repository name field, the user ID and password. Uncheck the Use Default check box from the right.
- The Import Data field is now applicable. Use the browse button [...] to navigate to an existing export file (for example AAA.exp).
- Click **Create** to begin the import.

Repository progress and status



When the Repository import is finished, you can check the information about the import operation by clicking the import.log file link. This log file is stored in the temporary folder specified above, c:\appbuilder\temp\import.log. Subsequent import operations overwrite the information in this file. Refer to [Reviewing the Log File](#) for more information about the log file.

Repository Replicator

The Repository Replicator provides another way for you to quickly and easily migrate data from an Enterprise or Personal Repository to a Personal Repository. This functionality differs from a standard download in that a download moves a subset of repository objects and merges them into the existing workstation repository. The Repository Replicator is used only to migrate an entire mainframe or Personal repository to a workstation. While Download uses the API layer to access the repository, the Repository Replicator connects directly to the database. Repository replication does not transfer the security model from the mainframe repository; it preserves the workstation security model.



The Repository Replicator uses a destructive migration. It deletes the contents of the target repository before importing the source objects.

The following topics are discussed in this section:

- [Prerequisites for the Repository Replicator](#)
- [Running the Repository Replicator](#)

Prerequisites for the Repository Replicator

The following items are required to use the Repository Replicator:

- IBM DB2 Connect on the PC that is used to connect to the source repository database.
- Distributed Data Facility (DDF) must be configured and started on the mainframe. This will create a NetID that is used by DB2 Connect to communicate with the database.
- You must create a workstation database. Refer to [Creating a Repository](#) for instructions.

Running the Repository Replicator

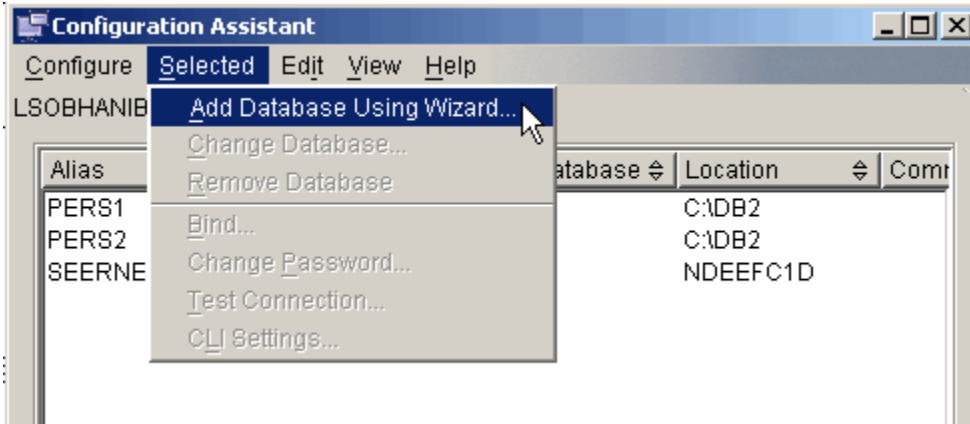
To use the Repository Replicator, do the following:

1. Create a connection to the Enterprise Repository using the [IBM DB2 Client Configuration Assistant](#).
2. Create a workstation repository as your target repository. Refer to [Creating a Repository](#) for instructions.
3. Start the Repository Replicator. See [Begin Replicating the Repository](#) for instructions.

Take the following steps to create a connection to an Enterprise Repository using the IBM DB2 Client Configuration Assistant:

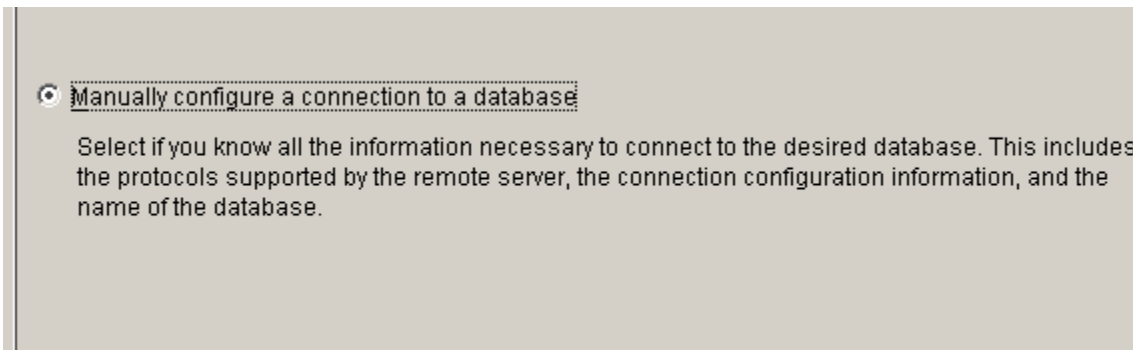
1. Click **Start > Programs > IBM DB2 > Set-up Tools > Configuration Assistant**.

DB2 Configuration Assistant



2. Click **Selected > Add Database Using Wizard**.
3. From the first wizard dialog, select **Manually configure a connection to a database** and click **Next**.

Configure DSN Manually



4. On the next dialog, select **TCPIP** as your communications protocol and click **Next** to continue. The wizard automatically selects **Connect directly to the server**. Keep this default.
5. On the next dialog, set your TCPIP communication parameters. Provide your Host name and the Port number. The Service name is not required.
6. Click **Next** to continue.
7. On the next dialog, specify the name of the database to which you want to connect. You must be aware of the database name to which you want to connect on the mainframe. For OS/390 and z/OS databases, this is the Location Name. The database alias and comment is not necessary.
8. Click **Next** to continue.
9. On the next dialog, you must register the database as a data source in ODBC (Open Database Connectivity). Then, use that registered name in the DSN field of the Repository Replicator Wizard (see [Source Repository parameters](#)). On this dialog, check **Register this database for ODBC** and select **As system data source**.

Registering with ODBC

Register this database for ODBC


As system data source
 As user data source
 As file data source

Data source name

Optimize for application

10. Click **Next** to continue.
11. On the next dialog, specify the node options. Select your mainframe operating system from the Operating system list box. Remote instance name and Comment are not necessary.
12. Click **Finish** .
The final two dialogs are not necessary.

Begin Replicating the Repository

 You must disconnect from the repository to activate the Repository Replicator option.

Take the following steps to replicate the Enterprise Repository onto a workstation repository.

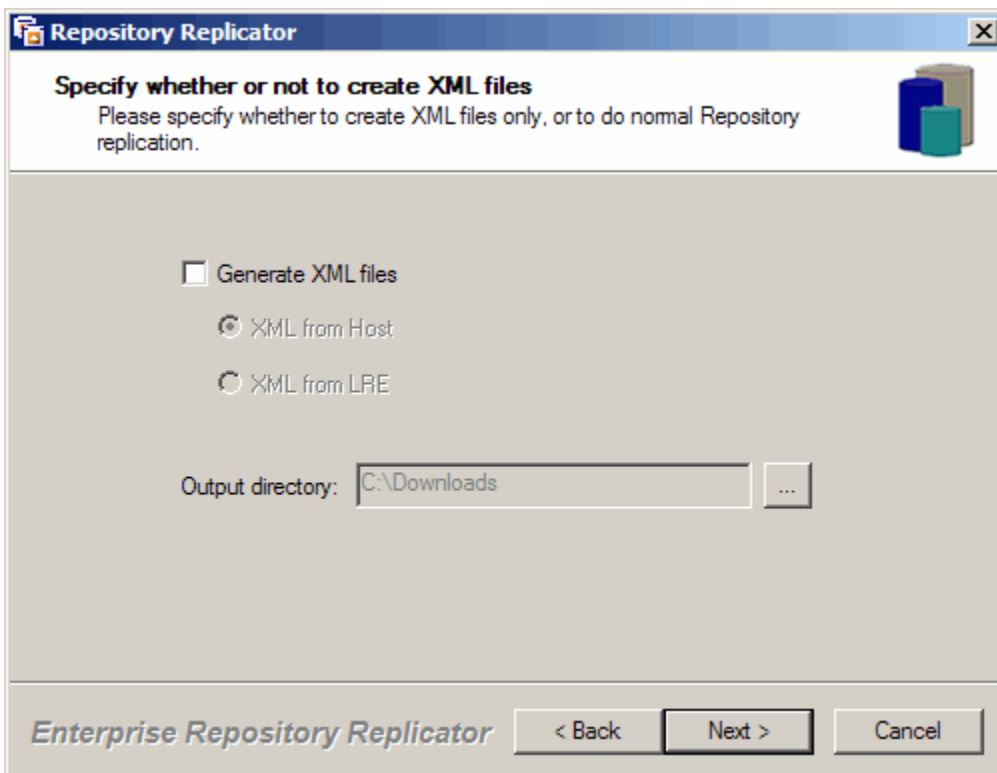
1. From within the Repository Administration tool, disconnect from the workstation repository. Select **Repository > Disconnect** .
2. To access the **Repository Replicator**, select **Repository > Repository Replicator**. Or, Click **Start > All Programs > AppBuilder > Repository > Repository Replicator**. The Repository Replicator window displays.

Repository Replicator tool



3. Click **Next** to continue, and the second dialog displays. It provides the option to generate XML files from the Enterprise Repository, a Personal Repository or to do a normal repository replication.

XML File Option



4. Select the Generate XML files option to create XML files from the appropriate source repository. If you are doing a normal repository replication, leave the XML file option unchecked.
5. Specify the output directory for those files. Select **Next** to continue.
If you have selected a Personal Repository, the Personal Repository parameters dialog, shown in the figure below, appears.

Personal repository parameters

Repository Replicator

Specify target workstation repository parameters
Please specify the target workstation repository parameters.

User name:

Password:

Repository:

To continue, click Next

Repository Replicator < Back Next > Cancel

6. Select the target repository from the drop-down list. Enter the workstation repository User name and password for the database and click **Next** to continue.
If you have selected an Enterprise Repository, the Enterprise Repository parameters dialog, shown below, displays.

Source Repository parameters

Repository Replicator

Specify Enterprise Repository parameters
Please specify user name, password, data source name, qualifier, and version for the Enterprise Repository.

User name:

Password:

DSN:

Qualifier:

Version:

Codepage:

To continue, click Next

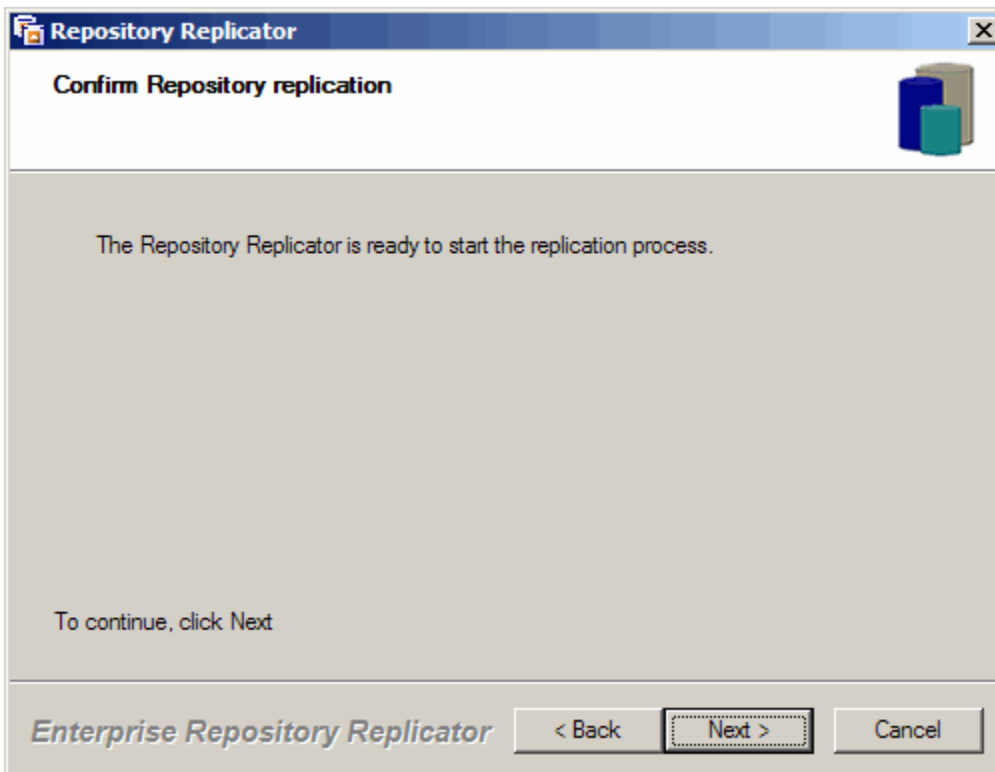
Repository Replicator < Back Next > Cancel

The following table provides information on each field entry.
Source Repository parameters

Field	Description
User name	Enter your user name for the source Repository you are replicating.
Password	Enter the password for the source Repository.
DSN	Enter the Data Source Name for the Enterprise Repository. This is the data source you registered with ODBC using DB2 Connect.
Qualifier	Database qualifier for the Enterprise Repository.
Version	Enter the repository version that you are replicating.
Codepage	Select the codepage for the Enterprise Repository.

7. Click **Next** to continue.

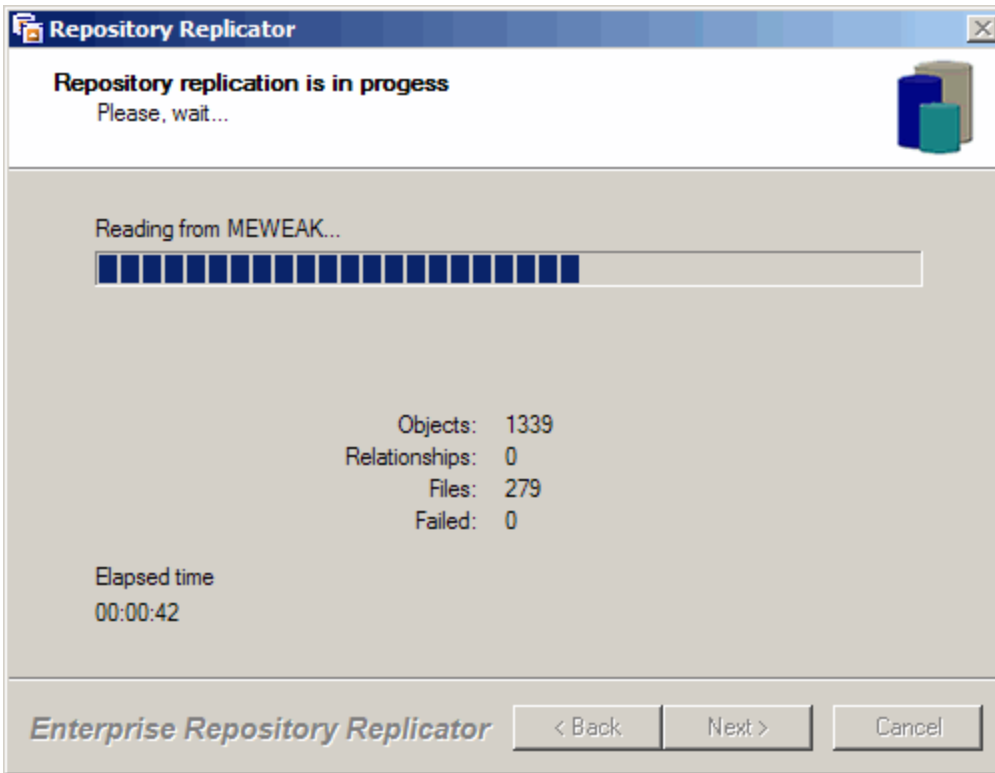
Replication confirmation



8. From the confirmation dialog, do one of the following:

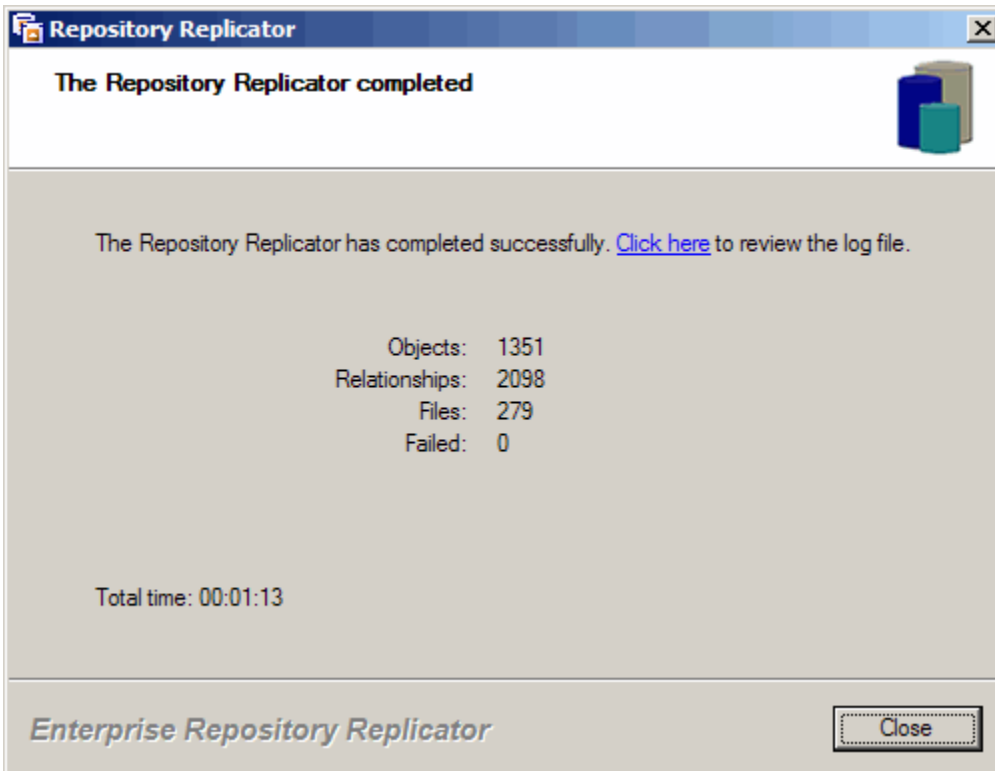
- Click **Next** to begin the replication process.
- Click **Back** to return to the parameters dialog and modify the parameters.
- Click **Cancel** to exit the Repository Replicator wizard.
The Repository Replicator begins the replication process.

Repository Replicator in Process dialog



When the replication process is complete, a summary dialog displays.

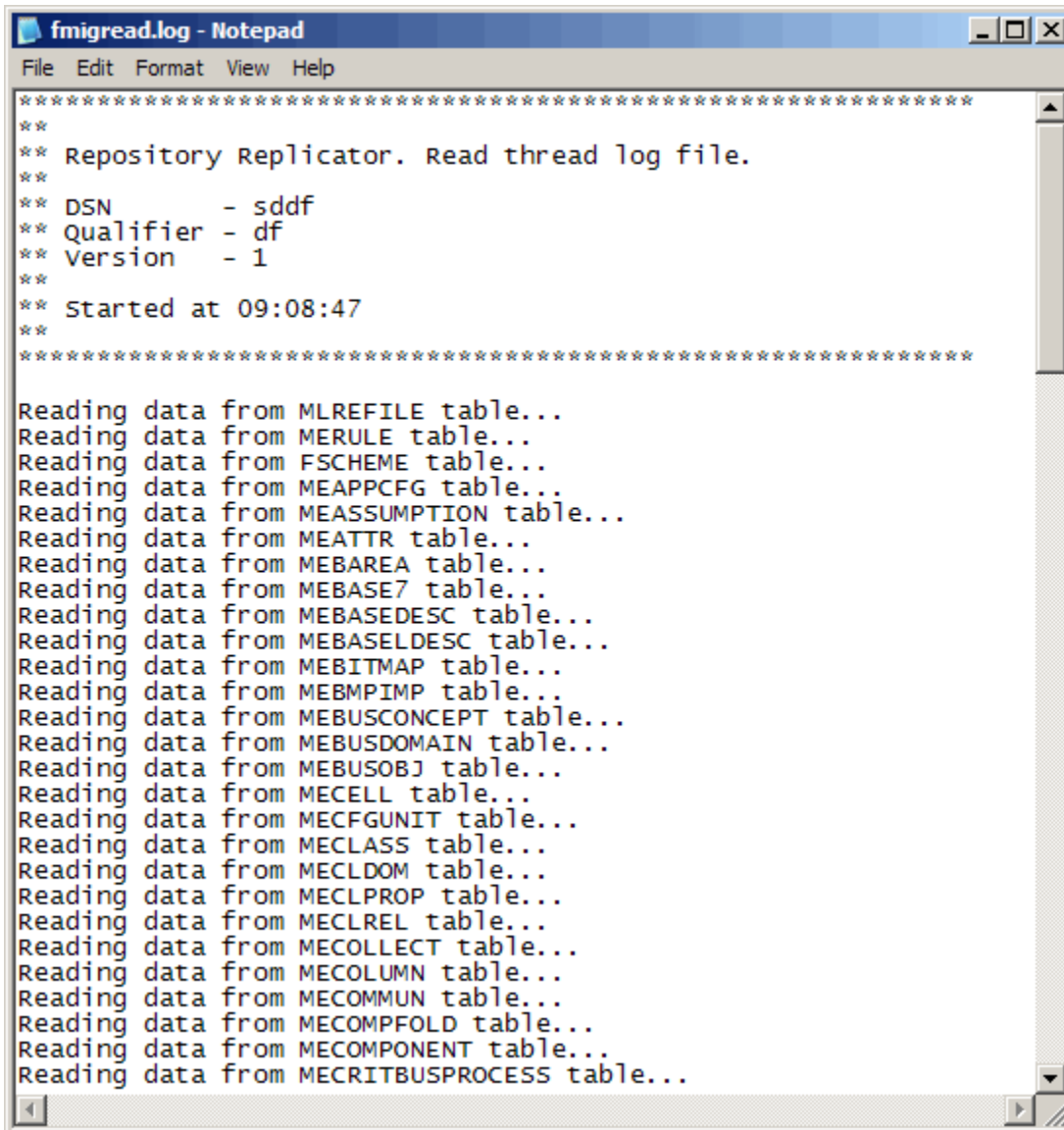
Replication Complete



The process produces two log files: Read Thread and Write Thread. These files are located in the _ERReplicator folder within your system's Temp directory. For example:
 C:\Documents and Settings\User1\Local Settings\Temp_ERReplicator.

9. Click the appropriate link on the final dialog to view the log files. One log displays the Read Thread Log File, and the other the Write Thread Log File.

Read Thread Log File



```
fmigread.log - Notepad
File Edit Format View Help
*****
**
** Repository Replicator. Read thread log file.
**
** DSN      - sddf
** Qualifier - df
** Version  - 1
**
** Started at 09:08:47
**
*****

Reading data from MLREFILE table...
Reading data from MERULE table...
Reading data from FSCHEME table...
Reading data from MEAPPCFG table...
Reading data from MEASSUMPTION table...
Reading data from MEATTR table...
Reading data from MEBAREA table...
Reading data from MEBASE7 table...
Reading data from MEBASEDESC table...
Reading data from MEBASELDESC table...
Reading data from MEBITMAP table...
Reading data from MEBMPIMP table...
Reading data from MEBUSCONCEPT table...
Reading data from MEBUSDOMAIN table...
Reading data from MEBUSOBJ table...
Reading data from MECCELL table...
Reading data from MECFGUNIT table...
Reading data from MECLASS table...
Reading data from MECLDOM table...
Reading data from MECLPROP table...
Reading data from MECLREL table...
Reading data from MECOLLECT table...
Reading data from MECOLUMN table...
Reading data from MECOMMUN table...
Reading data from MECOMPFOLD table...
Reading data from MECOMPONENT table...
Reading data from MECRITBUSPROCESS table...
```

Write Thread Log File


```
fmigwrite.log - Notepad
File Edit Format View Help
*****
**
** Enterprise Repository Replicator. write thread log file.
**
** Repository - PERS1
**
** started at 17:56:01
**
*****

Loading data into FSCHEME table...
0 row(s) inserted.
Loading data into MEAPPCFG table...
15 row(s) inserted.
Loading data into MEASSUMPTION table...
0 row(s) inserted.
Loading data into MEATTR table...
86 row(s) inserted.
Loading data into MEBAREA table...
0 row(s) inserted.
Loading data into MEBASE7 table...
0 row(s) inserted.
Loading data into MEBASEDESC table...
0 row(s) inserted.
Loading data into MEBASELDESC table...
0 row(s) inserted.
Loading data into MEBITMAP table...
10 row(s) inserted.
Loading data into MEBMPIMP table...
11 row(s) inserted.
Loading data into MEBUSCONCEPT table...
0 row(s) inserted.
Loading data into MEBUSDOMAIN table...
0 row(s) inserted.
Loading data into MEBUSOBJ table...
```

10. Click **Close** to close the Repository Replicator.

Packing and Reindexing a Repository

The packing and reindexing utilities provide a method to clean up repositories that have been used for extended periods of time. If you are going to pack the repository, you should reindex the repository also; however, you can reindex the repository without packing.

 Create a repository export file before doing a pack or reindex. Packing the repository permanently deletes files that cannot be restored without a backup. Refer to [Using Repository Export](#).

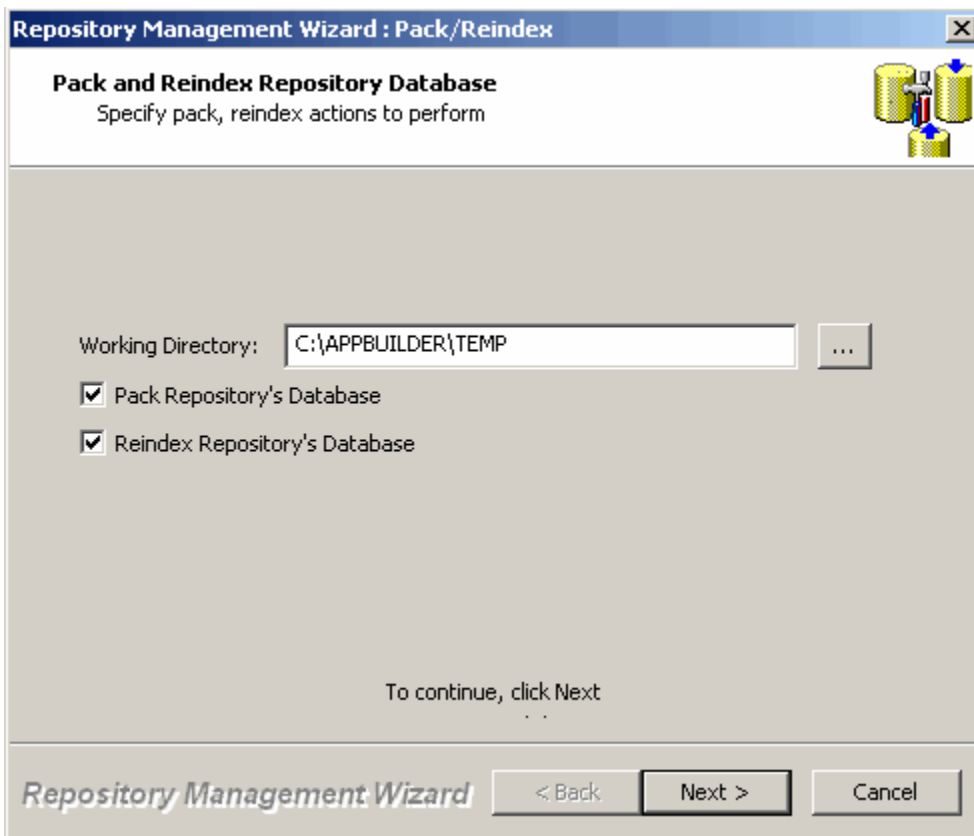
When an object is removed from the repository, for instance through Construction Workbench, the object is not removed from the database. It is marked as **deleted** in the database tables, and does not show up in any queries of the repository. Packing a repository permanently removes all entities, relationships, and files that have been marked as deleted. This is why we recommend creating an export file before packing. Refer to [Using Repository Export](#) for more information. After you pack a repository, it is good practice to reindex that repository.

Reindexing a repository drops and recreates all of the current indexes placed on the repository tables. You should reindex after you pack a repository; however there are other reasons to reindex without packing first. It is recommended that you reindex a repository after it has been packed, or if the repository has been used for long periods of time or just restored using the `fwyutil` command.

Take the following steps to pack/reindex a repository:

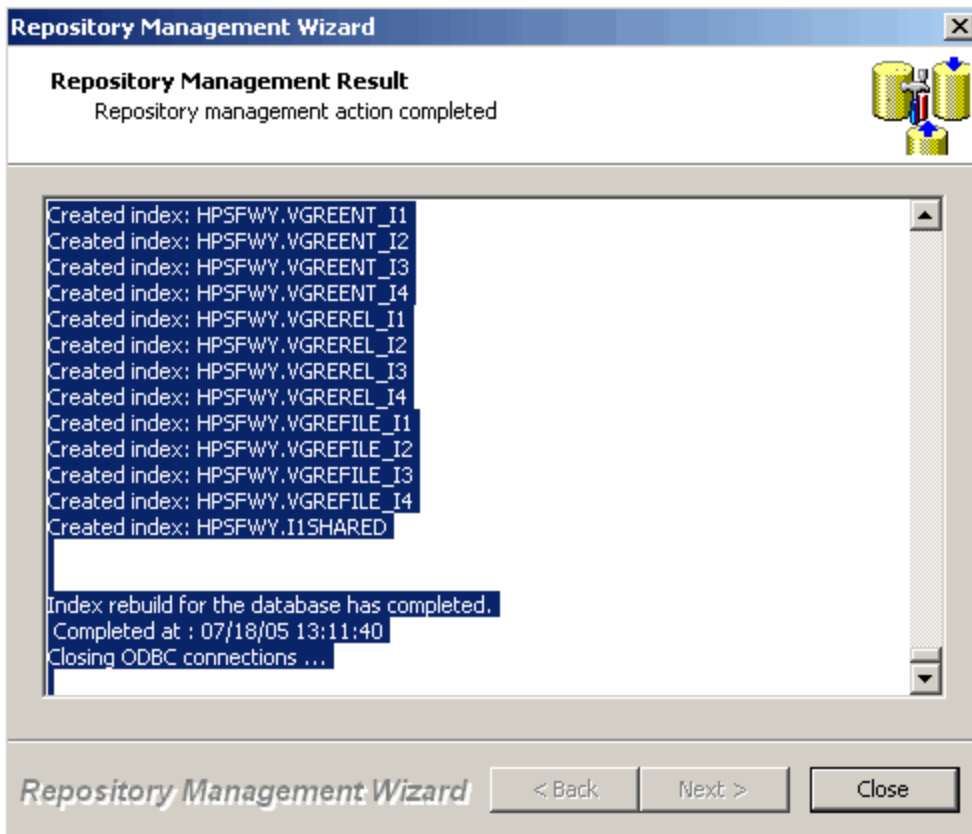
1. From the Repository Administration tool, select **Tools > Pack Reindex**. The Repository Management Wizard displays.

Pack & Reindex wizard



2. Select one or both check boxes and click **Next**.
You can pack and reindex the repository in one step by selecting both check boxes.
A confirmation dialog displays.
3. Click **Next** to begin the operation.
When complete, the following dialog displays.

Pack repository complete



4. Click **Close**.

The log files are stored in the AppBuilder/Temp directory. Refer to [Reviewing the Log File](#) for information about how to view the log file from within the Repository Administration tool.

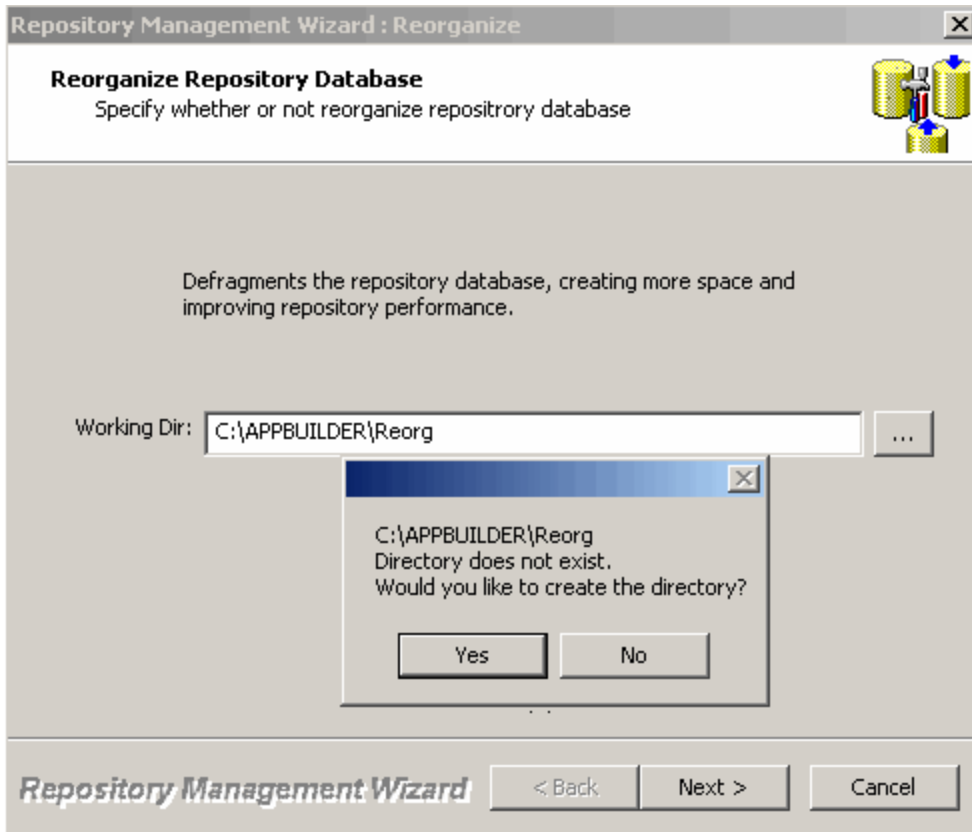
Reorganizing a Repository

Over time, numerous updates, deletes, and inserts can impact your repository performance. Often the physical placement of newly inserted rows does not match the logical sequence that is defined by the index. Reorganize your repository to further improve repository performance and response time. While packing the repository eliminates table rows for objects that are marked deleted, the space remains in the database. Reorganizing eliminates the space by defragmenting the repository and synchronizing the indexes with the physical tablespaces. Reorganizing your repository will greatly improve performance, access ability, and recover wasted disk space. Reorganize only after packing and reindexing the repository first.

To reorganize the repository:

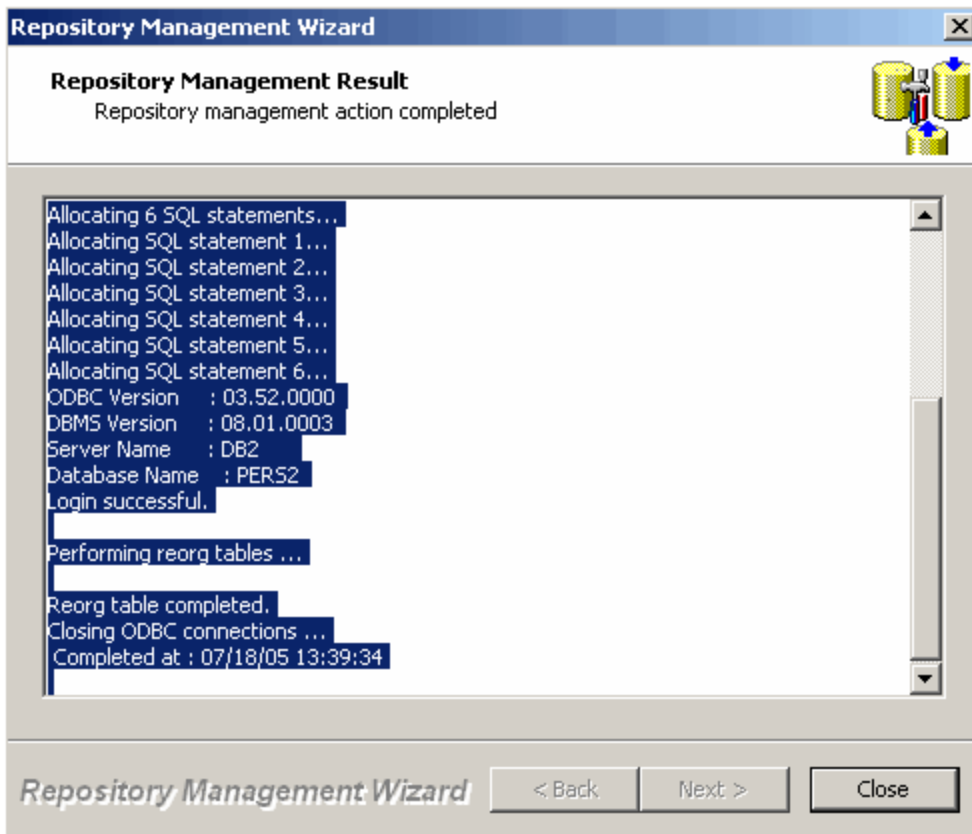
1. From the Repository Administration tool select **Tools > Reorganize**. The Repository Management Wizard displays.

Reorganize Repository – Changing the default directory



2. To accept the default temp directory, click **Next** to continue.
A confirmation dialog displays.
3. Click **Next** to begin the Reorganize operation.
Or, you can change the Working Directory, and if necessary, AppBuilder will create the directory for you.
4. Type the new directory path or click the browse ... button to navigate to the new directory.
If necessary, the system will prompt you to create the new directory.
5. Click **Yes** to create the new directory.
6. Then, click **Next** to continue.
A confirmation dialog displays.
7. Click **Next** to begin the Reorganize operation.
When complete, the following dialog displays.

Reorganize Complete



8. Click **Close**.

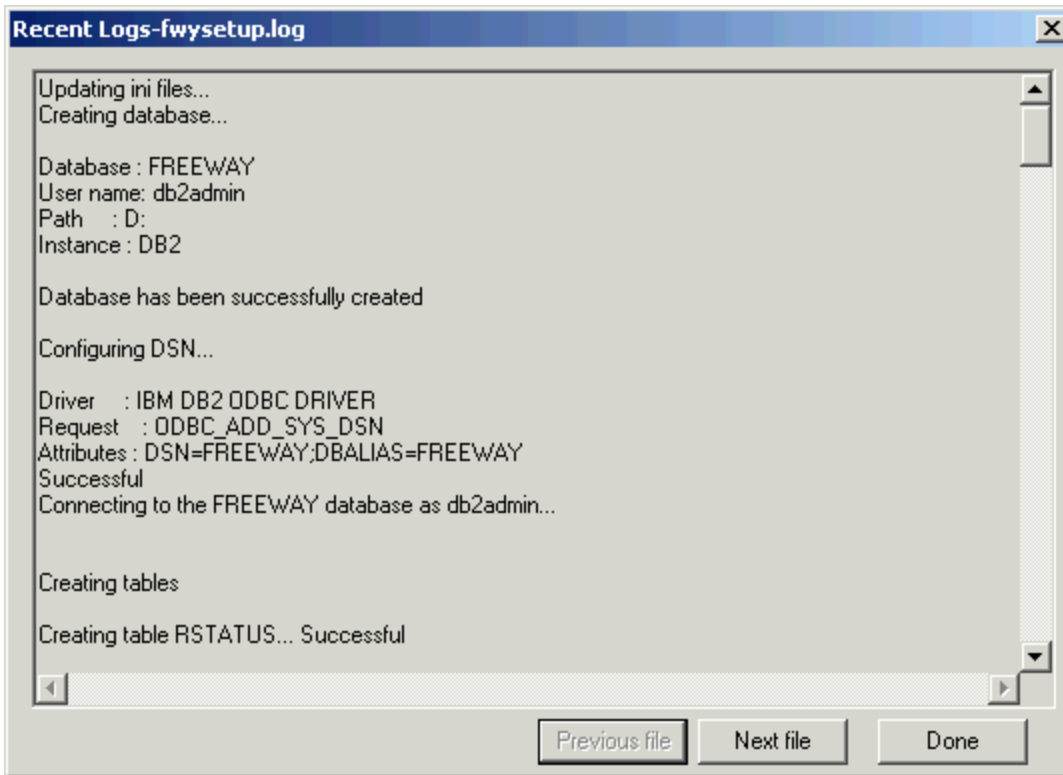
The log files are stored in the AppBuilder/Temp directory. Refer to [Reviewing the Log File](#) for information about how to view the log file from within the Repository Administration tool.

Reviewing the Log File

A log file keeps track of all of the events that occur during procedures, such as *create* and *delete*. It contains helpful information about what occurred during the action and explains errors, if any.

- To view the log file, select **View > Log**. This displays a log file for the last action that you performed.
- To view the next log file, select **Next file**.
- To close the Recent Logs window, click **Done**.

Sample Log File

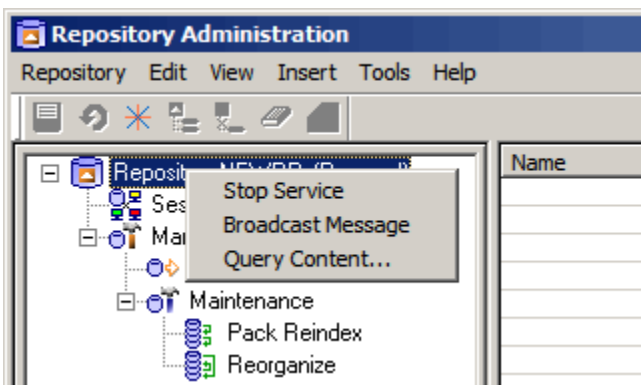


Broadcasting Messages

Broadcast messages to users connected to the server to notify them that the server is coming down, make specific requests, or deliver project information. The broadcast mechanism allows you to send a message to *all* the connected users of the designated Workgroup Repository. Do the following:

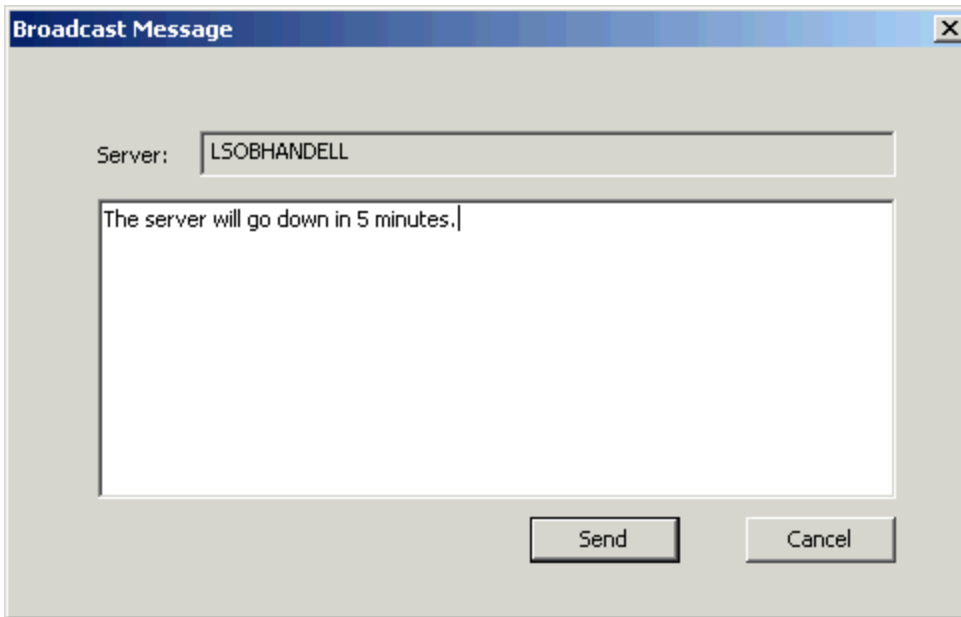
1. From the Repository Administration tool, select **Tools > Broadcast Message**.
Or, from the Management tab, right-click the root object in the hierarchy and select **Broadcast Message**.

Right-click menu



2. The Broadcast Message window appears with the name of the server in the Server field. Type the message you want to broadcast to users in the box and click *Send*.

Broadcast Message Window



Using a Repository

This section provides the information needed for a developer to perform and understand basic tasks while working with repositories. Refer to [Managing A Repository](#) for information about the Repository Administration tool and system administration. If you do not have system administrator authority, use other AppBuilder tools such as Construction Workbench to accomplish the following tasks:

- [Connecting to and Disconnecting from a Repository](#)
- [Committing Session Changes](#)
- [Querying Repository Objects](#)
- [Printing Objects](#)
- [Creating and Locking Repository Objects](#)
- [Using the Object Browser](#)

Connecting to and Disconnecting from a Repository

When you open the Repository Administration tool or Construction Workbench, the system prompts you to connect to a repository. AppBuilder prompts you for the repository name to which you want to connect (see [Connecting to a Repository](#)). To connect to a Workgroup Repository, the client machines must be configured for the appropriate server. Refer to [Creating a Repository Alias](#) for more information.

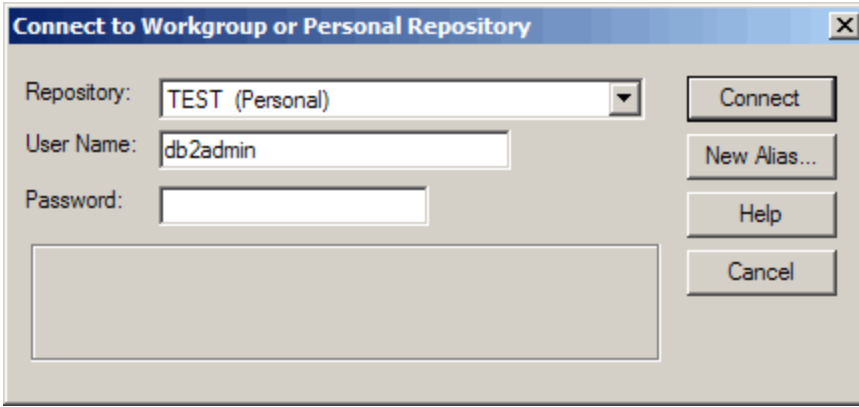


AppBuilder supports the Microsoft Windows 2000 and Windows Server 2003 implementation of Named Pipes. This software provides communications support. The named pipes can be transported on either NetBEUI or TCP/IP (using the TCP/IP NetBIOS Helper Service.)

From the connection window:

1. Select the repository from the Repository drop-down list.
2. Provide the appropriate user ID and password for the specified repository.

Connecting to a Repository



It is possible to disconnect from one repository and reconnect to a different repository. [Reconnecting to a different repository](#) illustrates how to reconnect depending on where you are in AppBuilder.

Reconnecting to a different repository

Product Component	Reconnect Action
Repository Administration	From the Repository menu, click Disconnect ; then click Connect . For more information about the Repository Administration tool, refer to Repository Administration Tool Overview .
Construction Workbench	From the File menu, click Disconnect ; then click Connect . For more information about Construction Workbench, refer to the <i>Development Tools Reference Guide</i> .



If your workstation has power-setting options specifying it to hibernate, and the machine does go into hibernation while you are working with AppBuilder, the (Freeway) connection to the repository will be broken by the hibernation.

Understanding Session Properties

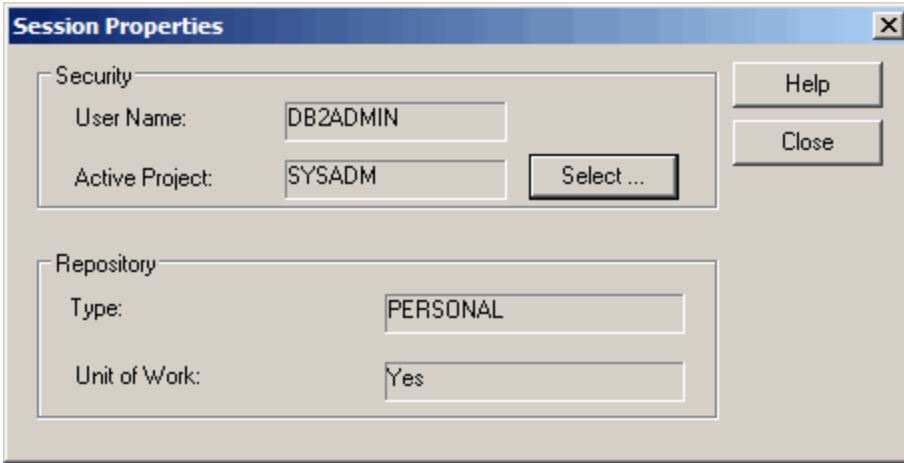
You can view the properties for a session from any of the following product components:

Viewing session properties of the product components

Product Component	To view Session Properties...
Repository Administration	From the <i>Repository</i> menu, click Session Properties .
Migration Export window	From the File menu, click Session Properties .
Migration Import window	From the File menu, click Session Properties .
Construction Workbench	From the File menu, click Session Properties .

The Session Properties window displays. The session properties vary, depending on whether or not you have installed the Unit of Work feature.

Session Properties from Repository Administration tool



This figure displays typical session properties for a session connected to the Repository Administration tool.

Session Properties

Field	Description
User Name	The User ID with which you connected to the repository.
Active Project	The project with which you connected to the repository. You can change your active project without disconnecting and reconnecting; however, from the Repository Administration tool, the project should remain SYSADM because repository management work should be done in the SYSADM project.
Type	This field shows the type of repository you are accessing during the current session.
Unit of Work	This field confirms if the Unit of Work is installed or not (available answers: Yes or No)

Committing Session Changes

When you are finished editing the objects and relationships in a repository session, you must commit the changes for them to be saved. You can commit changes in either the Construction Workbench or Repository Administration tool. For more information about sessions, refer to [Understanding Session IDs and Object Locking](#).

To commit changes:

- From Construction Workbench, click **File > Commit**.
- From the Repository Administration tool, click **Repository > Commit**.



Once you have committed your changes, you cannot rollback changes.

Understanding the Object Commit

When working within the AppBuilder tools, changes that you make to repository objects are only made for your session. Your session is created automatically when you connect to the repository. The objects that you modify are locked to prevent changes from being made in other sessions.

When you commit your changes, the locks on the objects you changed are released and the changes are permanently stored in the repository. A broadcast message is sent to all the tools that are connected to the repository indicating that changed, deleted, or created objects are stored in the repository and available for use. The other tools automatically refresh with the latest changes to the objects if they need them. Hierarchies that display the changed objects are refreshed on all connected tools. If tools such as Rule Painter or Window Painter have the object open in a read-only view, the user is prompted to refresh the object.

Understanding the Object Rollback

If you have *not* committed the changes that you made during your session, you have the option to rollback any changes that have been made

during the session. From any of the AppBuilder tools, select **Rollback** or **Rollback session changes** and modifications are reversed.

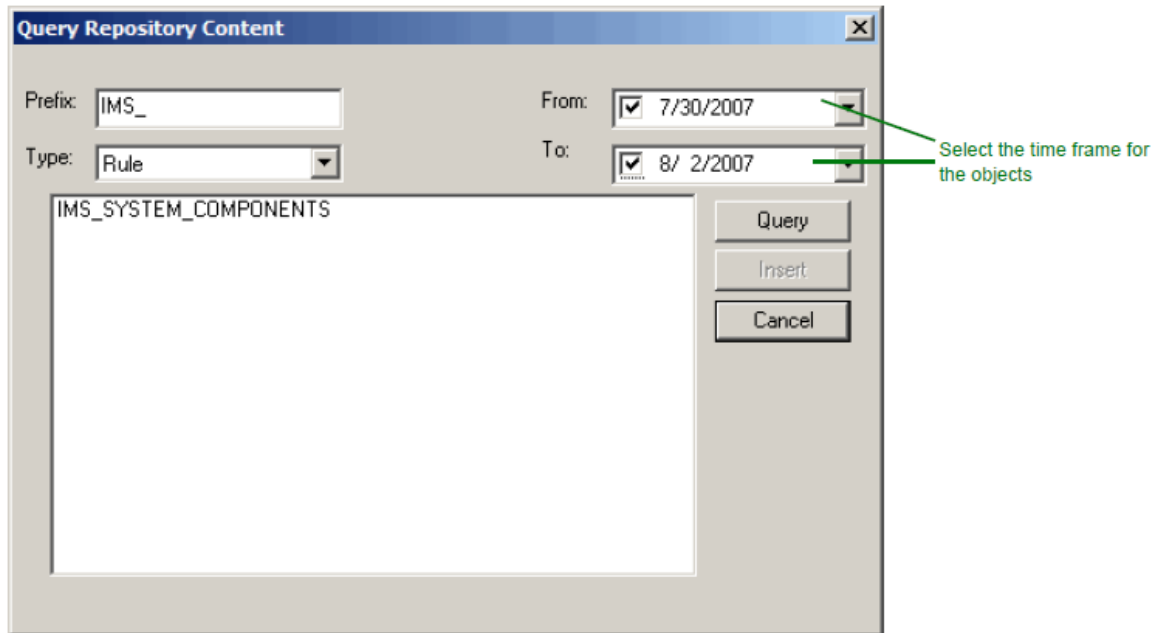
When you rollback your changes, the locks on the objects you changed are released. An unlock message is broadcast to all the tools that are connected to the repository.

Querying Repository Objects

There are many times when you will query a repository for the objects within. This section outlines the basic steps to running a repository query. These basic steps are true regardless of the type of repository you are querying. For a workgroup repository, Query Content is available from the right-click menu when selecting the root object in the hierarchy.

1. Right-click the repository or version and click **Query Content** . Or, Highlight the repository or version and click **Tools > Query Content** . The Query Objects window displays.

Object Query window



2. Narrow your search by entering part of the object name in the Prefix field.
3. Choose an object type from the drop-down box at the top of the window.
4. You can also choose the time frame for the objects of search.
5. Click **Query**.
A list of objects in the repository for that object type and Query value is displayed.



In the HPS.ini file, you can set a query limit to the number of objects that are returned from your query. Doing so improves performance. If you happen to start a query that is returning 20,000 objects, the response is slow; therefore, setting a query limit returns the specified number of objects or all the objects if less than the specified number exist in the repository. The setting can either be removed entirely or commented out if you do not want a query limit. By default, the setting does not exist in the HPS.INI file.

```
[FREWAY_SERVER]  
OBJECT_QUERY_LIMIT=500
```

If the object does not display within the limit you set, consider using a partial Query value to narrow your search. See the figure above. Depending on the type of repository you are looking in, you can narrow the search by partial name, prefix, or date.

If there are no query objects to display, then the **No objects found** status displays at the bottom of the query dialog. The query contents dialog displays status only when a query returns no results.

6. Select one or more objects from the Object instances pane and click **Load** .
If you are selecting only one object, you can double-click that object to load. When selecting more than one object, you must click **Load** .

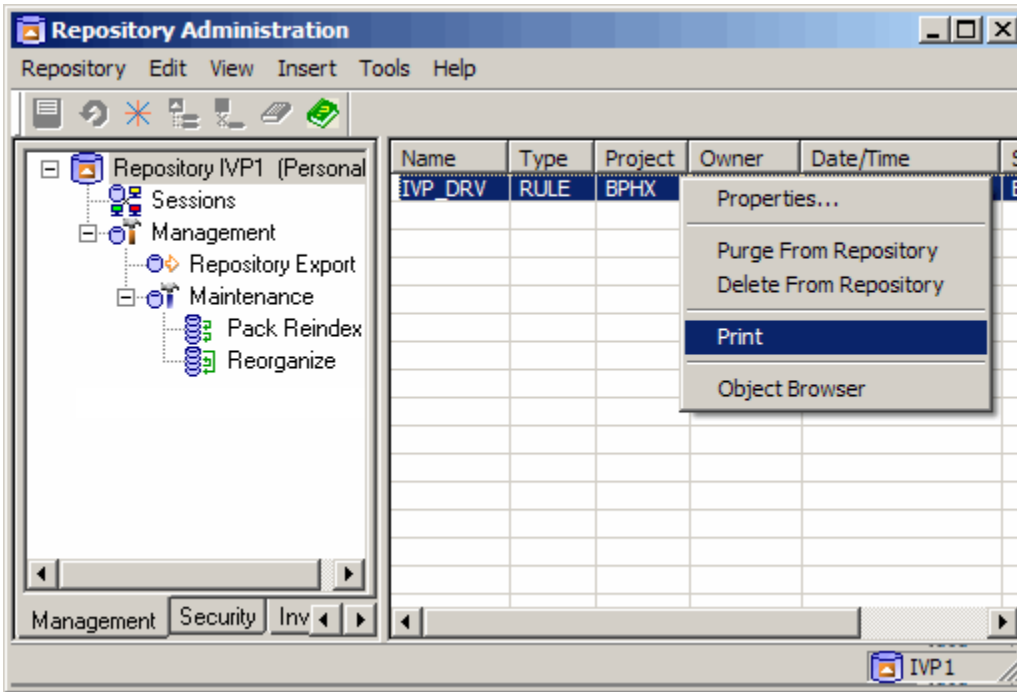
For more information about querying objects for migration, see [Relate the AppBuilder Objects](#).

Printing Objects

Once you have loaded the objects into the Repository Administration tool, you can print selected objects or send them to a print file:

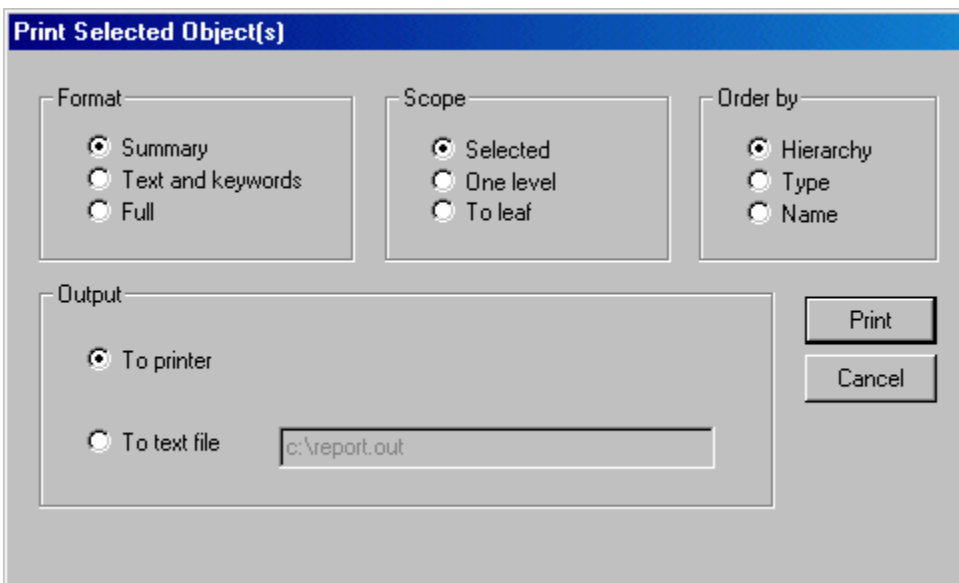
1. Right-click the selected object in the Repository Administration tool and select **Print**.

Printing a selected object



The Print Selected Objects window displays.

Print Selected Objects window



From this window, you can generate different types of reports for the selected objects. The following table outlines the different print options and what you can expect from each option.

Print options

Option	Description
Format Options	
Summary	Provides summary information for the selected entity and its children depending on the scope chosen. This information is similar to what is displayed in the Repository Administration tool. For each entity the following is displayed: <ul style="list-style-type: none"> • Entity name • Shortname • Entity type • Project • Owner
Text and Keywords	Provides a report with the following: <ul style="list-style-type: none"> • Entity name • Entity type • Text description • Keywords
Full	Provides a full report for the selected entity and its children depending on the scope chosen. See Full report for selected entity for an example of a full report on a Rule entity.
Scope	
Selected	Displays information for the selected entity only.
One level	Displays information for the selected entity and the child of that entity.
To leaf	Displays information for the selected entity and all children entities down to the leaf entity.
Ordered by	
Hierarchy	Displays the entities in their hierarchical order.
Type	Displays the entities in order of type (alphabetically).
Name	Displays the entities in order of name (alphabetically).
Output	
To printer	Print Setup window displays. Send the report to the printer you designate in the Print Setup window.
To text file	Send the report to the text file that you specify in the field. The default text file is C:\report.out. You can change the name and location of this file.

Example Printouts

Examples of the following reports are displayed:

- [Full Report](#)
- [Text and Keyword Report](#)
- [Summary Report](#)

Full Report

Full Report displays the complete report for the selected entity.

Full report for selected entity

```
FULL REPORT
NAME: Ims_system_components
TYPE: Rule

FWY_User:          USERID
FWY_Project:       SEER1
FWY_Date:          01-22-03
FWY_Time:          16:29:20
FWY_Owner:         SEER1
FWY_LockOwner:
RevDisplay:        1
VerDisplay:        VLS_1
DisplayChild:     ID 2429 -mErule-
Name:              IMS_SYSTEM_COMPONENTS
RemoteCreationDate: 93/05/07
RemoteCreationTime: 16:45
RemoteCreatedBy:   SEER1
RemoteMaintenanceDate: 93/05/07
RemoteMaintenanceTime: 16:45
RemoteMaintainedBy: SS0177
Project:           SEER1
ProjLockType:     N
LockId:           SEER1
ChangeNumber:     2
OwnerId:          SEER1
QAStatus:
LocalMaintenanceDate: 03/01/22
LocalMaintenanceTime: 16:29:20
ShortName:        RSYSIMS
Exec_Environ:     IMS
Rule_Source:
Sys_Source:       N/A
Rule_Name:        Mainframe System Components
Rule_Str_Id:      SYNC
Rule_TranId:
Rule_Imp_Name:    RSYSIMS
Package:
Isolation_Mode:   0

Text:
This rule is part of a system component.

Keywords:
mainframe
system components|
```

Text and Keyword Report

Text and Keyword Report displays the text and the keywords corresponding to the selected entity.

Text and Keyword report for selected entity only

```

TEXT AND KEYWORDS REPORT

NAME: Ims_system_components
TYPE: Rule

Text:
This rule is part of a system component.

Keywords:
mainframe
system components

```

Summary Report

Summary Report displays the summary of the selected entity and child entity to leaf.

Summary report of selected entity and child entities to leaf

Selected object

Child entities follow, ordered by hierarchy

SUMMARY REPORT					
Ims_system_components	RSYSIMS	Rule	SEER1	SEER1	
Ims_spa_component	HPISPAU	Component	SEER1	SEER2	
Get_ims_pcb_addr	HPIGPCB	Component	SEER1	SEER2	
Stop_tp_rule_region	HPISTPR	Component	SEER1	SEER2	
Mainframe	SEER2	Keyword object	SEER1	SEER1	
System	SEER1	Keyword object	SEER1	SEER1	
Components	SEER1	Keyword object	SEER1	SEER1	
Spa_input_view	VSPAINP	View	SEER1	SEER1	
Spa_output_view	VSPAOUT	View	SEER1	SEER1	
Get_ims_pcb_addr_view	VGETPCB	View	SEER1	SEER1	
Rule_stop_init	VRULESTP	View	SEER1	SEER1	
Spa_data	SPA_DATA	Field	SEER1	SEER1	
Spa_filler_byte1	SPA_FILLER_BYTE1	Field	SEER1	SEER1	
Spa_function_code	SPA_FUNCTION_CODE	Field	SEER1	SEER1	
Spa_length	SPA_LENGTH	Field	SEER1	SEER1	
Spa_tran_code	SPA_TRAN_CODE	Field	SEER1	SEER1	
Spa_return_code	SPA_RETURN_CODE	Field	SEER1	SEER1	
Ims_pcb_list_addr	IMS_PCB_LIST_ADDR	Field	SEER1	SEER1	
Ims_rule_id	IMS_RULE_ID	Field	SEER1	SEER1	
Return_key	RETURN_KEY	Field	SEER1	SEER1	

Creating and Locking Repository Objects

This section provides information about what happens within a repository when a user creates a repository object and a relationship inside an AppBuilder hierarchy diagram. For more information about creating an object, refer to the *Development Tools Reference Guide*. This section discusses the following topics:

- [Understanding Lock Types](#)
- [Understanding Session IDs and Object Locking](#)
- [Checking Locked Status](#)
- [Clearing Locks](#)
- [Understanding Locking Messages](#)
- [Broadcasting Object Changes](#)

To insert a new object into your hierarchy:

1. Right-click on the parent object and select Insert Child or Insert Sibling, depending on the relationship you want for the new object.
2. Click on the appropriate object you want to create.
Only the objects that are allowed in that position in the hierarchy are displayed in the menu. The Insert Object window displays.
3. Type the name of the new object and click **Insert**.
The new object is inserted into the hierarchy and created in the repository with a unique identifier.
4. To open and modify the object, double-click on it in the hierarchy diagram. The object opens in the tools area of the Construction Workbench.

Understanding Lock Types

Objects are locked when changes are made to them in the repository, preventing other users from performing certain actions on those objects. The Workgroup Repository uses an automatic object locking mechanism that:

- Prevents users from simultaneously updating the same object
- Preserves repository integrity when multiple objects are modified
- Preserves the integrity of relationships between objects

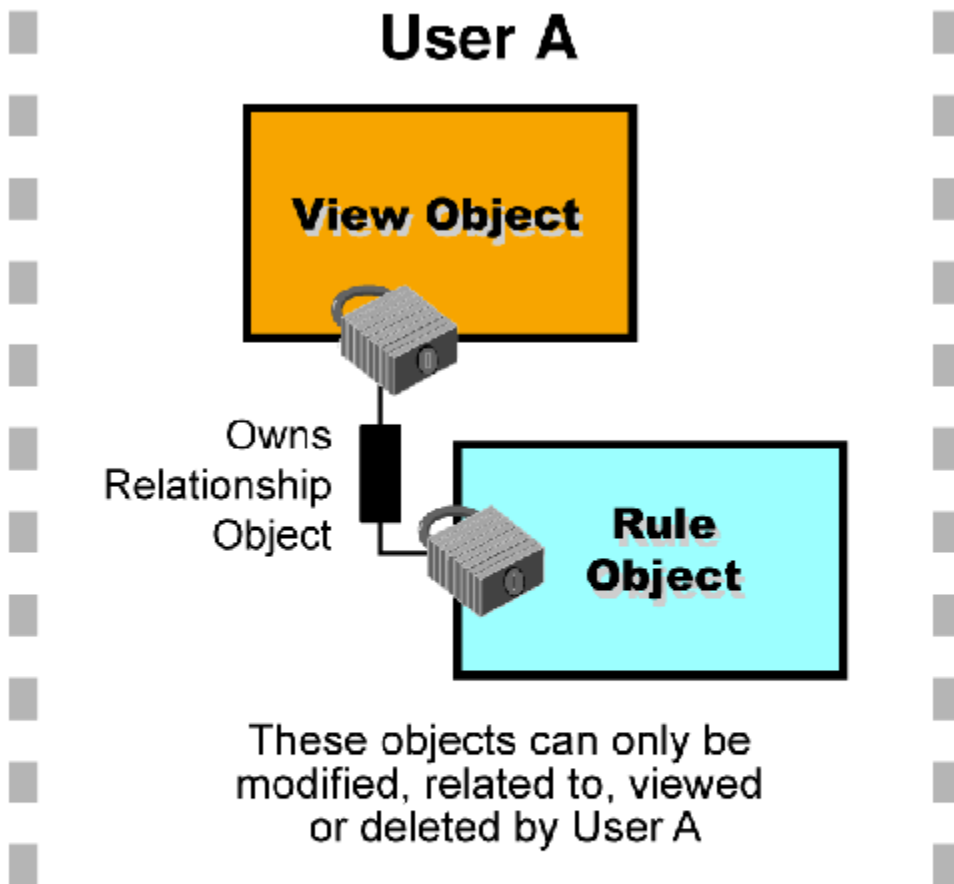
Exclusive Lock

One of the types of locks that AppBuilder places on an object in a repository is the *Exclusive Lock*. Only one person can open an object for modification at a time. The first person to open the object gets the exclusive lock. Subsequent developers will receive an error message when attempting to modify an object under an exclusive lock. When the first person commits or rolls back the object, the lock is released.

When an object is under an exclusive lock, other users *cannot* :

- Modify or delete the object
- Relate objects to the locked object
- View the changes to the object (although the object can be seen as it existed before it was changed)

Exclusive Locks



Shared Lock

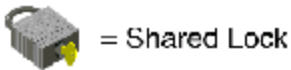
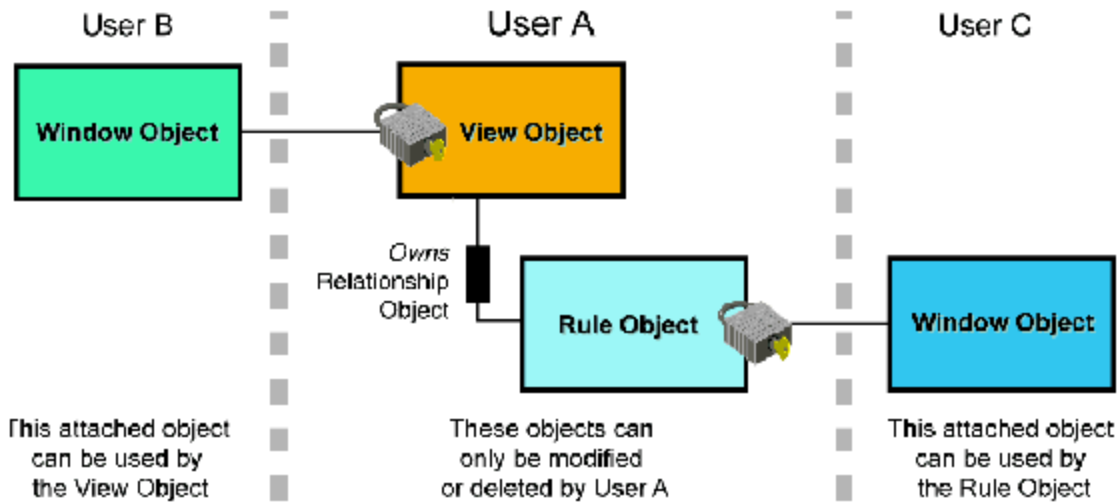
A Shared Lock occurs when a developer changes a *relationship* between two entities. In this case, the entities on either side of the relationship are locked from update by other developers.

A Shared Lock prohibits other users from modifying or deleting the shared object, until the session is committed or rolled back. However, other


objects can be related to the shared objects. These *related objects* are defined as shared locks when you:

- Modify an object that relates other objects together.
or
- Create a new relationship between other objects.

Shared Locks



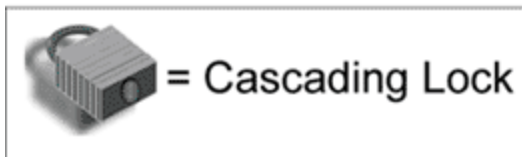
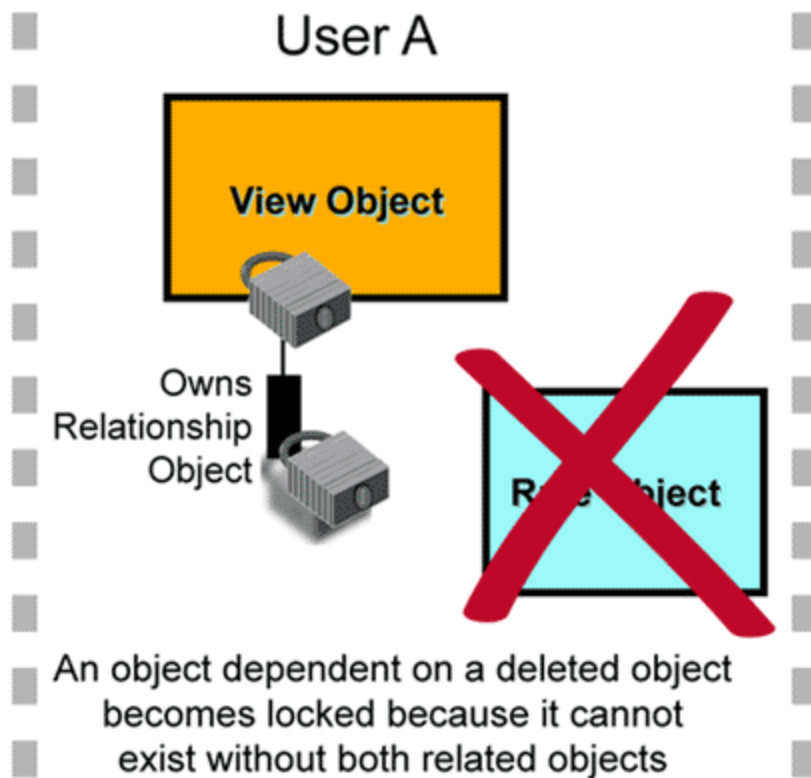
In [Shared Locks](#), the *Owns* relationship object that exists between a *View* object and a *Rule* object has been changed by User A, causing both the *Rule* and *View* objects to receive shared locks. Other users can attach objects to the *Rule* and *View* objects in their modeling, but they *cannot* modify or delete the *Rule* or *View* objects. User A always has privileges to modify or delete these objects. Other users can create objects that are used by the *Rule* or *View* object, as User B and C have done.

 If only one user has a shared lock on an object, that user can update or delete the object and the shared lock automatically converts into an exclusive lock. However, if another user has a shared lock on the object, the object cannot be changed or deleted until that user releases the lock.

Cascading Lock

A Cascading lock is identical to an [Exclusive Lock](#), except that it applies to objects not directly updated. This type of lock is placed on an object that needs to be deleted because the object on which it depends has been deleted. The locked item is physically deleted when a commit is performed. Using the same example as [Shared Lock](#), if a *View* object by definition connects to a *Rule* object and the *Rule* object is deleted, the *Owns* relationship object becomes locked because it *cannot* exist without *both* the *View* and *Rule* objects. Refer to Cascading Lock for more information.

Cascading Lock



Understanding Session IDs and Object Locking

The Workgroup Repository is a real-time, multi-developer environment; therefore, it is possible for many developers to connect into the repository at the same time. To enable this, the repository uses a number of services, which can be started before they are required.

The main repository service is called GRESVCNT (GRE Server Controller). It is responsible for a number of subordinate services called GRESRVNT (GRE Servants).

In this section, we limit our discussion of these services to how they affect repository objects. Each of the GRESRVNTs is allocated a session identifier when it is started (e.g. 01, 02, 03 etc.). This number is known as the Session ID. When a developer starts Construction Workbench and connects to the repository, they are allocated one of the GRESRVNT services, and hence a session number. AppBuilder uses the Session ID for various internal processes, but it is most noticeable when AppBuilder attempts to lock objects. When the developer creates or changes an object, AppBuilder attempts to lock the object in question. When locking an object, AppBuilder uses the session number as the lock identifier. The following is a brief walkthrough of how locking works in the AppBuilder Workgroup Repository.

Procedure: Repository Object Locking

Follow the steps:

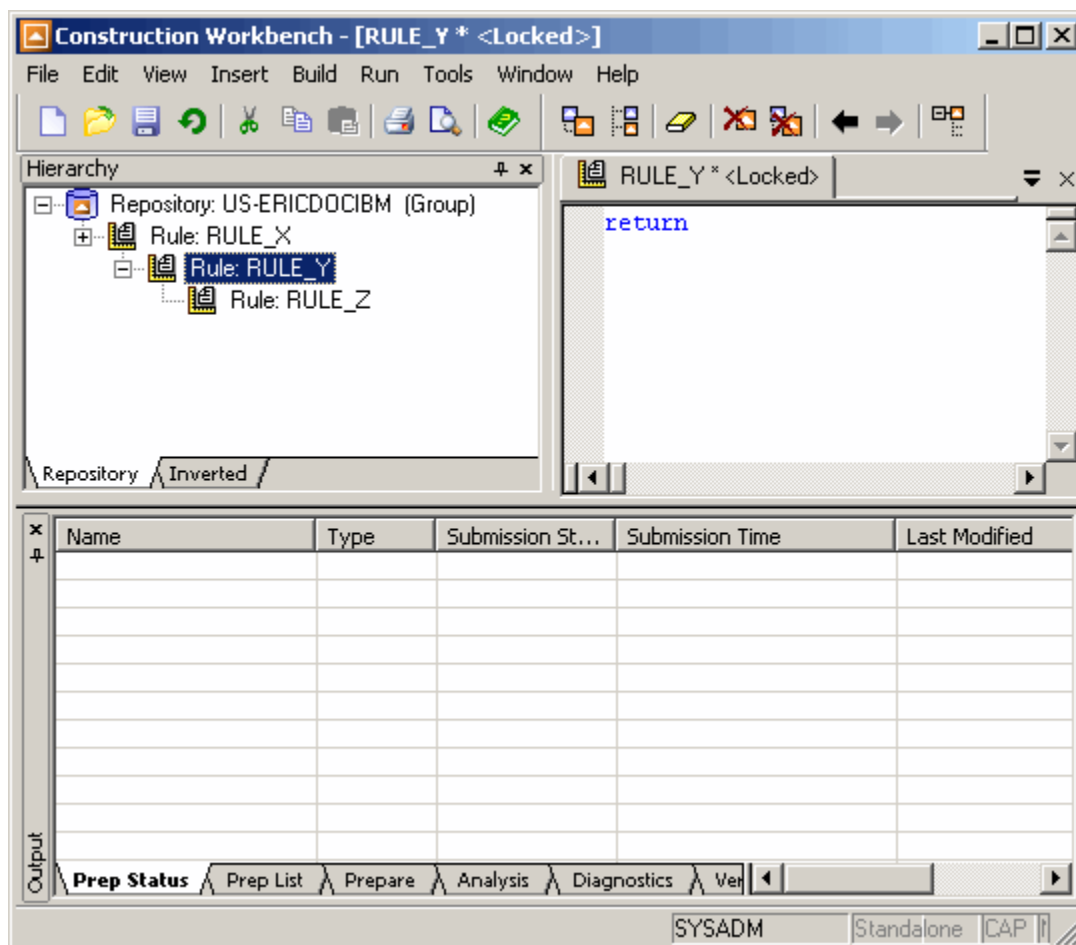
1. Connect to the Workgroup Repository using Construction Workbench.
2. A Session ID is allocated to that session.
3. Query the repository for a Rule object and inserts the Rule object into the hierarchy.
4. Double-click on the Rule object to open the rule editor and view the source code.
5. Attempt to modify the Rule source code.
GRESRVNT session reads the repository to see if the Rule object is already locked by another session. If it is not locked, then within the repository, AppBuilder places the session ID associated with the developer in the LOCK column of the repository table for that Rule object.
The word <locked> appears in the title bar of the rules code editor.
6. If the Rule is already locked by another developer (session ID), then the first developer receives a message stating that the object has been locked by another session.
This type of lock is known as an *exclusive lock*. In other words, the first developer has exclusive control over the object. No one else can update the object while the first developer holds the lock.

- When you, as the first developer, either commit or roll back, all locks which belong to that developer (that session ID) are released. Within the repository, AppBuilder simply changes the LOCK value for each locked object to zero (for that specific session ID). However, exclusive locks do not only affect the entity to which they apply. An exclusive lock on an entity also locks the relationships that the entity has with other entities. This is known as a [Cascading Lock](#) and, like the exclusive lock, it is designed to stop two developers from inadvertently changing the same object in some way.

Cascading Lock Example

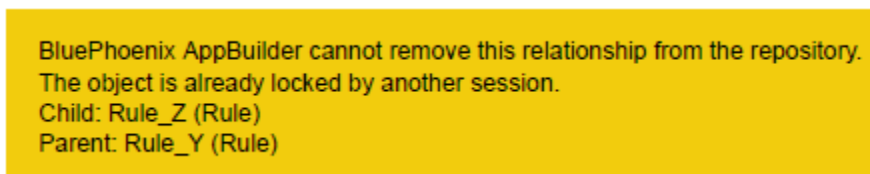
[Cascading Lock - Hierarchy Example](#) illustrates the use of a Cascading Lock.

Cascading Lock - Hierarchy Example



In [Cascading Lock - Hierarchy Example](#) another developer is permitted to modify RULE_Z, but if he or she tries to remove the relationship between RULE_Y and RULE_Z an error message displays (see [Locking message](#)). The same is true if he or she tries to delete the relationship between RULE_X and RULE_Y.

Locking message





Of course, with good project management, two developers should not be trying to make a change to the same object at the same time.



Checking Locked Status

In the Repository Administration tool, the repository state icon reflects the state of a locked *object* in the repository. The icons display for locked objects only. They indicate the type of lock that is on each locked object. To view or hide these icons, from the Security hierarchy tab, click **View > Repository States**, or press **F3**. This is a toggle option.



Object Locked

- Object has been locked (by the object owner) 
- Object has been locked (by another user) 

Relationship Locked

- Relationship from the selected object has been locked (by the object owner) 
- Relationship from the selected object has been locked (by another user) 

Both Object and Relationship Locked

- Both the object and the relationship have been locked (by the object owner) 
- Both the object and the relationship have been locked (by the another user) 

You can also see which users have locked objects using the Status Details tab of the Server Control window. Refer to [Status Details Dialog](#) for more information.

Clearing Locks

When a developer performs a commit or a rollback, AppBuilder clears all locks for that object. However, there are occasions when you may need to remove an object lock manually; for example, if the connection between the developer workstation and the Workgroup Repository ends unexpectedly, object locks may not be released.

AppBuilder is designed so that the next time a developer connects to the repository after an unexpected shutdown the system automatically rolls back the locks. When the service begins, it examines the sessions and rolls back any that it finds.

If there is a problem with AppBuilder not releasing object locks, and it is evident that these objects are not locked by any other developer, use a manual rollback command. The manual rollback command unlocks all repository objects based on the Session ID.

To perform a manual rollback, at the command line type:

```
gresrvnt -r -i<Session_ID> -u <userid> -p <password>
```

where:

- <Session_ID> is the session containing uncommitted work to be rolled back.
- <userid> and <password> must have administrator level authority to the repository database.

For example:

```
gresrvnt -r -i4 -u SERMKP -p XXXXXXXX
```



There is a space between the -u and the <userid> and a space between the -p and <password>.

Refer to [Understanding Session IDs and Object Locking](#) for more information.

Understanding Locking Messages

[Locking Messages](#) lists the AppBuilder locking error messages and steps to rectify the error. Sometimes these messages are due to user error, sometimes they are not. This section should help you understand why you are getting a specific error message and what you can do to rectify the situation.

Locking Messages

Repository message	Reason for message	Action to take
The object is already locked by another session.	A second user attempted to modify or delete an object that is under an exclusive or cascading lock.	Wait until the object lock is released.
A parent or child of the object is locked by another session.	A user attempted to change or delete an object when its parent or child object was under an exclusive lock.	Wait until the object lock is released.

The object is currently being shared by one or more sessions.

- A user attempted to change or delete an object when one or more users are sharing the object.
- A user attempted to delete an object that requires a cascading lock, and it is already being shared by one or more users.

Wait until the object lock is released.

Broadcasting Object Changes

The system broadcasts object changes to all connected tools when the updates are committed or rolled back. In many of the tools, the view of the objects is updated automatically. However, some tools that display complex images, such as drawings or source code, prompt the user to update the view of the object. AppBuilder prompts you to do one of the following:

- **Refresh** - updates the view of the object.
- **Continue** - leaves the view of the object as is. The words <out of sync> appear in the tool title bar.
- **Close** - closes the tool with no changes.

Using the Object Browser

You can use the Object Browser to view repository entities, owners, relationships, and status within the repository. Using the Browser windows, you can view repository objects, their relationships, and view the entire hierarchy in the repository.

The following topics are covered in this section:

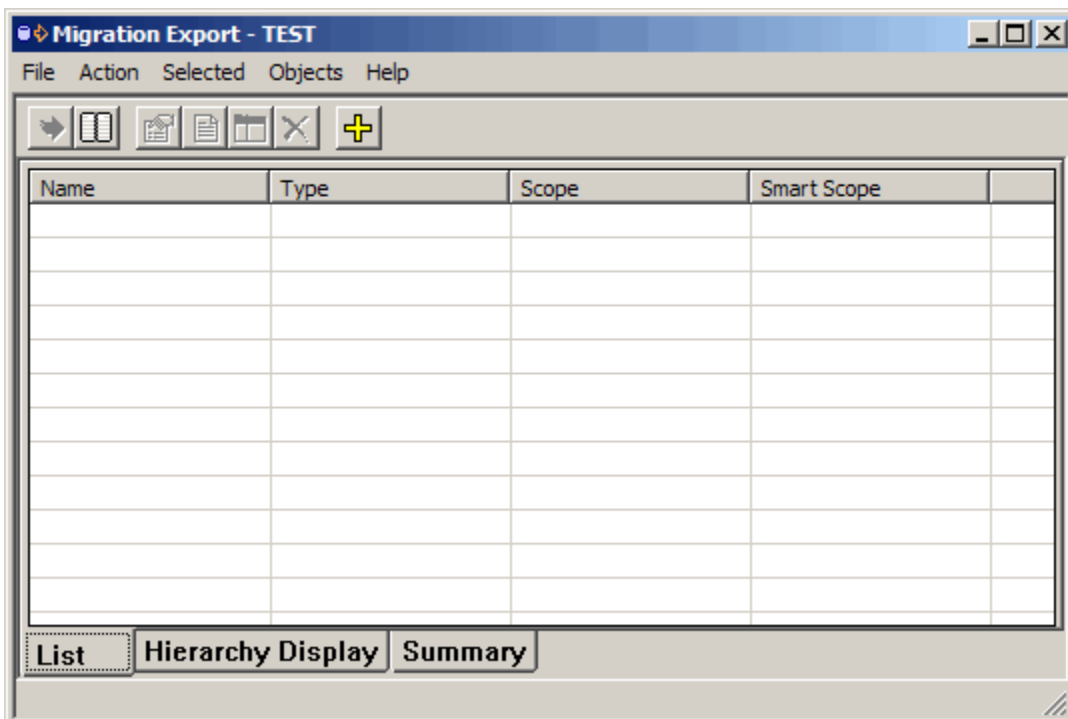
- [Opening the Object Browser](#)
- [Viewing and Editing Object Properties](#)
- [Viewing Child Entities and Relationships](#)
- [Viewing Parent Entities](#)

Opening the Object Browser

To access the object browser window, do the following:

1. In the Repository Administration Tool, select Tools> Migration Export. You can choose to Create a new Export file or to open an existing one. The Migration Export window opens.

Migration Export Window

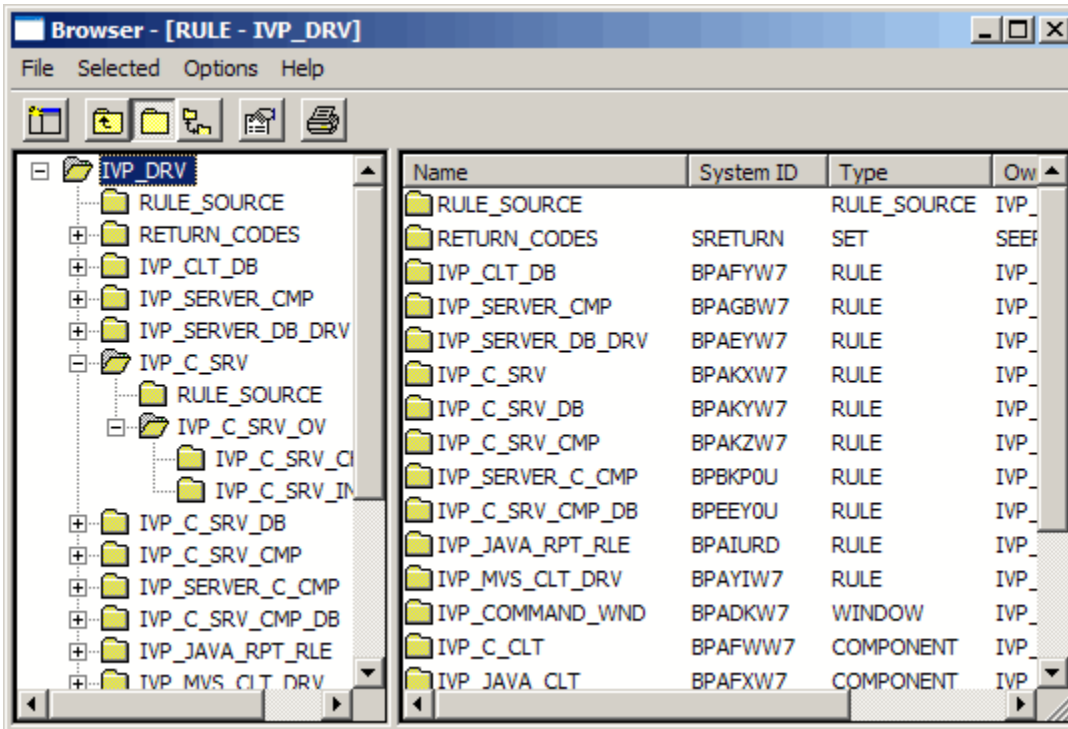



2. Add a root object to the export list by selecting Objects > Add Root Objects or by clicking on the **Add Root Objects** toolbar button.

3. Select the object that you want to open in the Browser and click on the Open Browser toolbar button.

The Browser window displays. The selected object displays in the right pane and the children of that object display in the left pane.

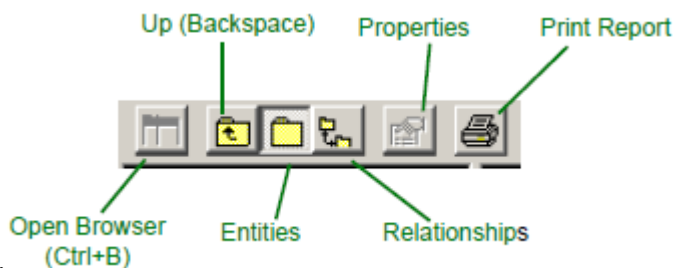
Browser window



The hierarchy pane in the left area of the window displays the hierarchy associated with the object. If an object has a plus sign  in front of it, the object has associated child objects. The detail pane on the right displays any child objects for the selected object. Some properties of the child objects are displayed:

- **System ID** – the unique identifier (HPSID) of this object in the repository
- **Type** – the object type
- **Owner** – the owner or creator of this object
- **Project** – the project in which this object was created

The Browser toolbar, shown in [Browser Toolbar](#), provides a shortcut to performing the tasks in the Browser. Many of these functions are also available from the Object Browser menus.



Browser Toolbar

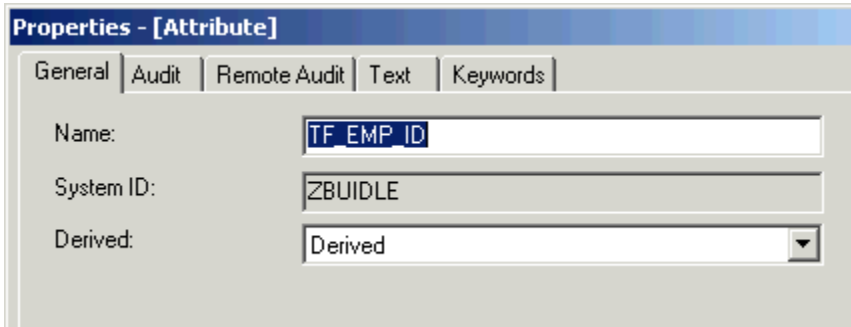
Viewing and Editing Object Properties

To use the Object Browser to view properties and relationships of objects in your repository, take the following steps:

1. Select the object in the hierarchy.
2. Choose **Selected > Properties**. The **Properties** window ([Browser Object Properties window](#)) for that object appears.
Or, right-click on the object and select **Properties**.
Or, highlight the object and select the **Properties** icon from the toolbar.
Or, highlight the object and press **Alt+Enter**.
3. Type the new values in the fields that can be edited and click **OK**.

4. Click **Apply** to continue to edit fields.
5. Commit to the changes. Click **Undo** to undo changes.

Browser Object Properties window



Viewing and Editing Relationships

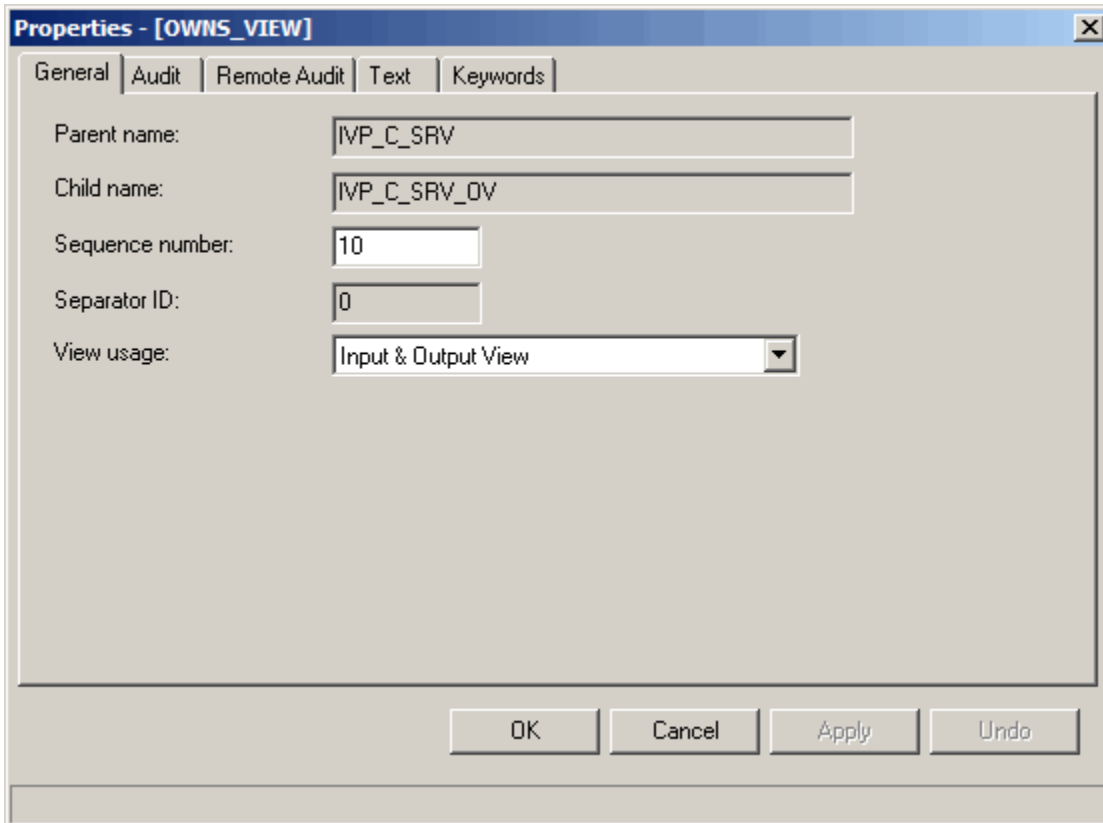
To view or edit the properties of a relationship in the Object Browser, complete the following steps:

1. In the hierarchy, select the child object in the relationship.
2. Select **Selected > Relationship properties** or press **Alt+R**.
3. The **Properties *window for that relationship appears (Browser Relationship Properties Window)**. To edit the information, type the new values and click **Apply** or **OK**.

If permissions to modify properties for that object are required, the following message appears in the status bar:

Security validation failed - update permission required

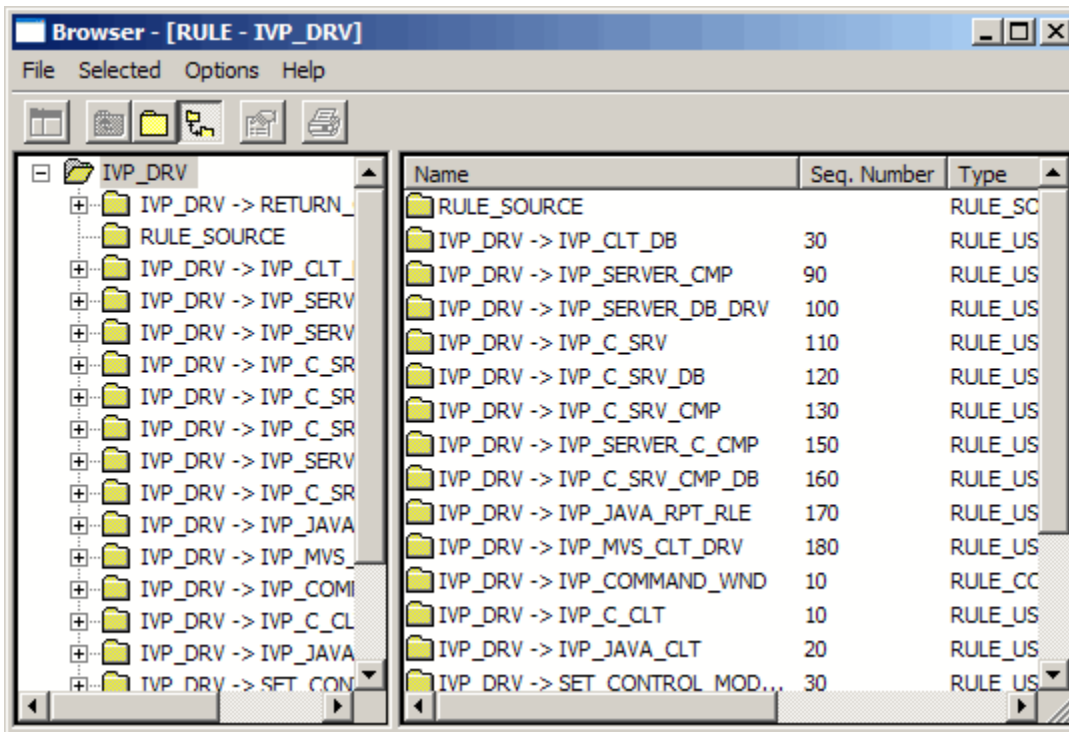
Browser Relationship Properties Window



To view object relationships from the Options menu, complete the following steps:

1. In the Browser window, select **Options > View Relationships**. The relationships are displayed in the detail pane designated with an arrow (->) in the entry ([Browser Relationships Example](#)).

Browser Relationships Example

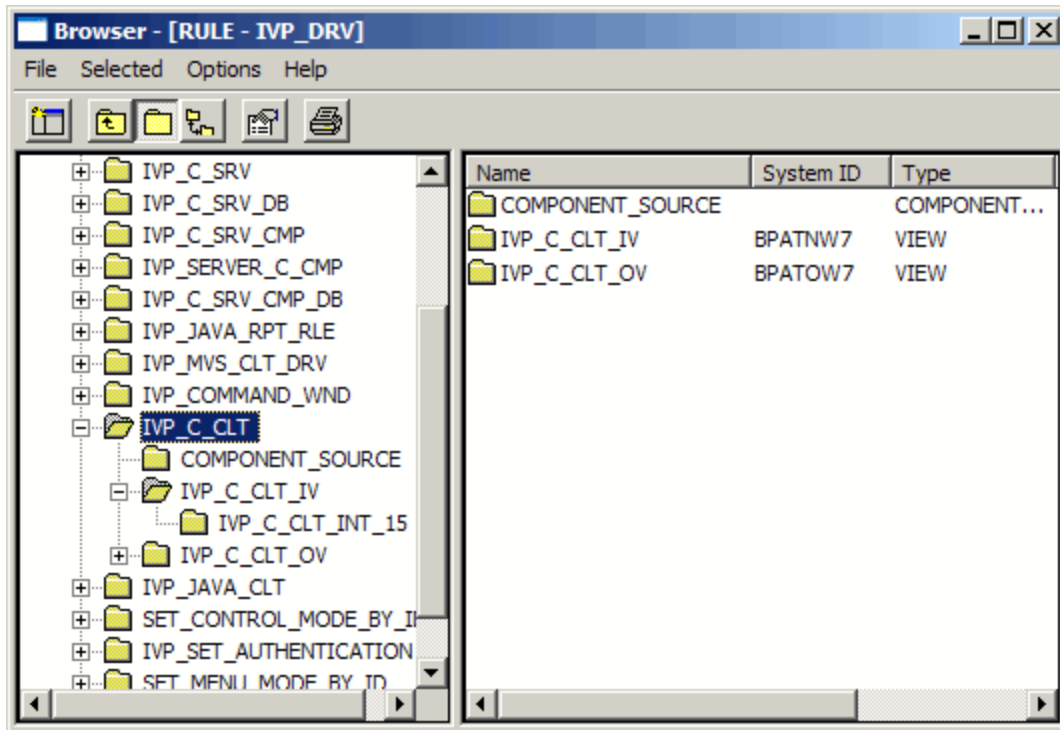


2. In the detail pane, select the relationship that you want to edit.
3. Select **Selected > Properties**. The **Properties** window for that relationship appears ([Browser Relationship Properties Window](#)).
4. Type the new values and click **OK**.

Viewing Child Entities and Relationships

You can use the Object Browser to view the child entities in your repository. If an object has a plus sign in front of it in the hierarchy, the object has associated child objects. To view the child entities in the Object Browser, select the object in the hierarchy. The child objects appear in the detail pane. If you click the plus sign in the hierarchy, the hierarchy explodes to reveal the children of that object in the hierarchy. You can continue to explode the hierarchy if a plus sign is displayed next to the object name.

Browser Hierarchy Relationships



To view the relationships between parent and child objects in the Object Browser, complete the following steps:

1. Select the object in the hierarchy.
2. Select **Options > View Relationships** . Relationship objects appear in the window. See [Browser Relationships Example](#) for an example of how the relationships are displayed.

Viewing Parent Entities

To use the Object Browser to view and edit the parent entities in your repository, complete the following steps:

1. Select the object in the hierarchy.
2. Choose **Selected > Explode Upward** (or click **Backspace** or **Up** in the toolbar).
 - If the selected object is a root in the hierarchy, the Object Browser changes to show all parents of the object.
 - If the selected object is a child in the hierarchy, all parents of the object appear in a new Object Browser window.
 - If the selected object has no parents, no action is taken.

Tuning for Optimal Performance

Tuning for Optimal Performance

This section describes how to improve the efficiency of repository access and storage. The following steps are *only recommendations*. Performance tuning procedures vary from system to system, depending on many factors, such as the amount of memory available, hard drive specifications, processor speed, and the type of data being stored.

For more information about database-specific performance tuning and procedures, consult a qualified Database Administrator (DBA) familiar with your system.

Performance tuning procedures include:

- [Evaluating Disk Utilization](#)
- [Configuring a UDB Database for Memory Usage](#)
- [Configuring the Mainframe Query Interval for Downloads](#)
- [Changing the Interval Time](#)
- [Configuring Windows Pagefile Size](#)
- [Configuring Download Processing](#)
- [Considerations for UDB Server Response](#)
- [Considerations for SQL Server Response](#)
- [Changing Windows Process Priority](#)

Evaluating Disk Utilization

Repository tables and the source files for each Repository each require a location with sufficient disk space.

In the [FREEWAY] section of HPS.INI, the FILE_DIR variable identifies the location of the source files for each repository. Change this variable to reflect a drive letter and location with sufficient disk space. For example:

```
FILE_DIR=<C>:\AppBuilder\FILES
```

The [FREEWAY] section also includes a TEMP variable. Change this variable to reflect a drive letter and location with sufficient disk space. For example:

```
TEMPORARY=<C>:\AppBuilder\TEMP
```

Configuring a UDB Database for Memory Usage

To change the interval time before you download, complete the following steps:

1. Open the UDB control center and right-click the database instance for the Repository that you want to update.
2. Select the **Configure Performance Wizard** and select **Server** or **Go to the Second Page**.
3. Change the target memory to use 90-100% of available memory.
4. Once the slider value has been changed, click **Done** at the bottom right of the panel.



If this value is set to 100% and applications other than UDB are running on the server, increasing this value might impact performance, and the value might require a downward adjustment.

Configuring the Mainframe Query Interval for Downloads

The default setting for the interval at which the mainframe is queried during download is 20 seconds. To reduce the likelihood of FTP time outs, prevent unnecessary FTP traffic, unnecessary requests of the migration server, and the resulting DB2 queries, change the interval to 99 seconds.

Changing the Interval Time

To change the interval time before you download, complete the following steps:

1. Select **Programs > AppBuilder > Repository > Repository Administration**.
2. Select **Edit > Host Properties** to configure the mainframe properties for the repository.
3. Change the **Timeout** setting to 99 seconds.

To change the interval time during the download:

1. Change the interval value in the window that is displayed during the download process. See [Current Job Status window](#).
2. Press the **New Interval** button during the download.

Configuring Windows Pagefile Size

Configure the Pagefile so that it does *not* exceed twice the available RAM size. A Pagefile that is too large might decrease performance.

Configuring Download Processing

There are two refresh options available for download processing: [Refresh Unit\(s\)](#) and [Refresh Directory](#). The refresh setting you choose affects the download processing speed. Refresh Unit(s) determines whether or not the download is compared to the UOW. The Refresh Directory option determines whether or not your connected session is updated after the download.

Refresh Unit(s)

From the Download window, click **Options > Refresh Unit(s)**. These are the available values for this option.

Download/Refresh Unit(s) Options

Value	Description
-------	-------------

Never	Never compare UOW to download (No Refresh). For empty Personal Repositories where it is not necessary to track the UOW, set the value to Never.
Always	Always compare UOW to download (Refresh).
Prompt	Prompt you for the desired action before migration import. Displays a window box when the files have been moved through FTP from the mainframe and prior to the import.

Refresh Directory

From the Download window, click **Options > Refresh Directory**. These are the available values for this option.

Download/Refresh Directory Options

Value	Description
Never	Never refreshes contents of repository session with updates.
Always	Always refreshes contents of repository session with updates.
Prompt	Prompts you to refresh contents of repository session with updates.

Considerations for UDB Server Response

This section provides suggestions for improving UDB server response. The following topics are discussed in this section:

- [Heap Size Value](#)
- [RunStats](#)
- [Adjusting Log File Parameters](#)

Heap Size Value

The Workgroup Repository requires that the value for APPLHEAPSZ is set to a value greater than or equal to 256. In UDB Version 8, the default is 256; however, in UDB Version 7 the default value for APPLHEAPSZ is 128. Increase this value to at least 256. The 256 setting is adequate for small repositories only. Run tests to be sure that the log files are large enough to handle a large import or export from within Freeway Manager.

RunStats

We recommend periodically running "Run Statistics" on the UDB database, or after a large number of objects have been updated, deleted, or created. For example, run statistics on the database after large migration imports or after running a Pack or Reindex on the repository.

Run Statistics updates the database statistics to better optimize the SQL statements that are executed by the repository server. There are two ways to do this.

Using the IBM DB2 Control Center:

1. Navigate to the FREEWAY database and then to the tables within that database.
2. On the right side highlight all the tables with a Schema of HPWFWY.
3. Right-click on the highlighted tables and select **Run Statistics**.

Using the IBM DB2 Command Line Processor window:

1. Type `connect to freeway user <userid>`
2. Type `reorgchk update statistics on table all`

Adjusting Log File Parameters

When working with large repositories, actions such as a full repository export/import can fill up the log files and result in a failure to execute the action. These actions can typically perform a large number of updates to the database between syncpoints, thus requiring a larger amount of log file space. The IBM DB2 settings LOGFILSIZ, LOGPRIMARY, LOGSECOND may need to be adjusted.

From the IBM DB2 Control Center:

1. Navigate to the Freeway database. Right-click on the FREEWAY database and select **Configure Database Logging...**
2. A wizard program starts. Change these settings within the wizard program.

Considerations for SQL Server Response

The most common problems encountered with SQL Server are performance issues and disk space deficiencies due to frequent filling of the master database SYSLOGS segment. To address this, we provide the following recommendations and tips:

- [Using Temporary Database \(tempDB\)](#)
- [Increasing Memory Allocation](#)
- [Setting Number of User Connections](#)
- [Clearing Transaction Logs \(SYSLOGS\)](#)

Using Temporary Database (tempDB)

One way to improve SQL Server response when using a Workgroup is to use the temporary database (tempDB). The temporary database is used for work space to sort and create temporary tables for some join operations. SQL Server provides the option to store the tempDB in RAM, as opposed to writing it to a table. In many situations, this enhances performance significantly. However, if used inappropriately, it can consume memory, making memory unavailable for other system processes, thereby degrading performance. The following guidelines help avoid that occurrence.

Procedure – Storing tempDB in RAM

Do the following to store tempDB in RAM:

1. Store tempDB. The default of tempDB in memory is "none".
2. Change this setting to 10 MB or more, depending on available memory. Subsequent duplicate requests are cached in memory instead of being written to the drive and are sent more quickly. A tempDB setting of zero is the default.



Microsoft SQL server has editable settings to optimize performance. Unfortunately, setting the tempDB variable over your maximum RAM will disable SQL server (making the interface required to change it inaccessible). If you allocate more memory than is available, SQL Server will not restart. Follow the steps described in this section to fix the problem without requiring a rebuild of the master database.

Procedure – Modifying tempDB Settings

Do the following to modify tempDB settings:

1. Type `<sqlservr -c -f>` on the command line to start in single user mode.
2. Go into ISQL.
3. Type `<sp_configure "tempdb in RAM">` and execute.
4. Type `<sp_configure "tempdb in RAM" ,0>` and execute.
5. Type `<reconfigure with override>` and execute. This will reset it to 0.

Increasing Memory Allocation

To improve SQL Server response when using Workgroup, increase the default memory allocation for SQL Server in the SQL Server configuration parameters. The increase cannot exceed 80 percent of the total memory for the machine.



Default memory is measured in 2 MB increments; not 1MB increments.

Procedure – Increasing the Default Memory Allocation

Follow the steps:

1. Change the setting in Enterprise Manager under:
Console Root\Microsoft SQL Servers\SQL Server Group\Configuration\server_name
2. Right click and select **Properties**.
3. Click on the Memory Tab (or from the Management Console, select the *machine_name* and then **Tools** from the menu).
4. Select **SQL Server Configuration properties**.
5. Select the **Memory** tab.

Setting Number of User Connections

One way to improve SQL Server response when using Workgroup is to set the number of user connections optimally. For repository logins, validate that the User Connections setting is at a high enough level to support all of the AppBuilder client requests. This setting is similar to the available sessions in Workgroup Repository settings, averaging three (3) settings per client machine. For example, ten (10) client workstations using AppBuilder should have a user connection setting of *at least* 30.

Take the following steps to set the number of user connections:

1. Change the setting in Enterprise Manager under:
Console Root\Microsoft SQL Servers\SQL Server Group\server_name
2. Right click and select **Properties**.
3. Click on the **Connection** tab (or from the Management Console, select the *machine_name* and then **Tools** from the menu).
4. Select **SQL Server Configuration properties**.
5. Select the **Connection** tab.

Clearing Transaction Logs (SYSLOGS)

One way to improve SQL Server response is to clear the transaction logs. The GRE Servant Process transaction log fills up quickly, as does the Master transaction log of the master database, impacting repository performance negatively. If the GRE Servant transaction log fills, repository performance degrades and speed declines. If the Master log fills, SQL server will *not* start and may become disabled. Delete the GRE Servant Process transaction log regularly to avoid performance problems.

If the log is full, SQL Server gives the following error message:

```
Can't allocate space for object 'syslogs' in database <database name> because the 'logsegment' segment is full. If you ran out of space in Syslogs, delete the transaction log. Otherwise, use ALTER DATABASE or sp_extendsegment to increase the size of the segment.
```


Two options are available to correct the error:

- Clear the log completely
or
- Extend the segment.

Clearing the GRE Servant and Master Transaction Logs

Follow the steps:

1. To clear the log, delete the GRE Servant and Master Transaction Log. Open ISQL/W and type:
DUMP TRANSACTION <database name> WITH NO_LOG.

 This is an unrecoverable log dump.

2. Execute the command.
3. With successful execution, the system displays a message indicating that the action returned no data. Shut down SQL server and restart to see the effects.

Extending the Workgroup and Master Transaction Log Segments

To extend the segment, use the SQL stored procedure `sp_extendsegment`.


Clearing the Master Transaction Log to Start SQL

If SQL Server will not start, clear the log using the following procedure:

1. Type:
<sqlservr -c -f>
on the command line.
2. Go into ISQL. (Type ISQL/? for Help).
3. Type:
C:\> ISQL -Usa -P -Q"DUMP TRANSACTION MASTER WITH NOLOG"
4. To stop and restart SQL Server, type:
net stop mssqlserver
net start mssqlserver

Changing Windows Process Priority

We recommend installing a Workgroup Repository on a dedicated machine. Changing the Windows process priority is not necessary on a dedicated machine. The database service for DB2 or SQL server should have equal priority with GRESRVNT.EXE as they typically split most processing between the database and the AppBuilder repository server. However, if you are not using a dedicated machine for the Workgroup Repository, it may help performance to increase the Windows process priority. Each Workgroup process thread (GRESRVNT . EXE) can have its Windows priority changed from normal to high.

 Windows will warn you that changing priority may cause undesirable results including system instability. Proceed with caution.

Increase the priority of the repository on Windows under Task Manager to get more CPU cycles:

1. Select `gresrvnt.exe`, right click, and choose **Set Priority**.
1. Select **High** priority.

Workgroup Repository Rebuild

Workgroup Repository Rebuild

AppBuilder 3.1 Repository Administration Guide for Workgroup and Personal Repositories

The Rebuild Facility is available from within Construction Workbench when you are connecting to a Rebuild-enabled Workgroup Repository. Rebuild is only supported for Workgroup Repositories on Microsoft Windows Server and SQL Server, or DB2/UDB. Use Rebuild in conjunction with an Application Configuration or Partition that selectively updates and rebuilds only objects affected by a particular change. The Rebuild Facility updates a record each time an object is prepared for a particular Application Configuration or Partition.

Workgroup Rebuild is designed to assist with change management in a development environment. With Rebuild, you know what executables are affected by your changes, so you only build what is required without having to do extensive analysis on your changes. The analysis is normally done automatically when you are connected to a Rebuild-enabled Workgroup Repository. By properly partitioning applications, you can rely on AppBuilder to tell you what needs to be prepared.

For instance, if an application is deployed in Java on the client, and COBOL on the server using CICS, set up an Application Configuration to partition the application into two logical pieces. Create one partition for the client and configure it to prepare Java locally. Create another partition for the server and configure it to prepare remotely to the target CICS region. Once this configuration is in place, build the application for the first time and start tracking the rebuild status of the objects.

The following topics are discussed in this section:

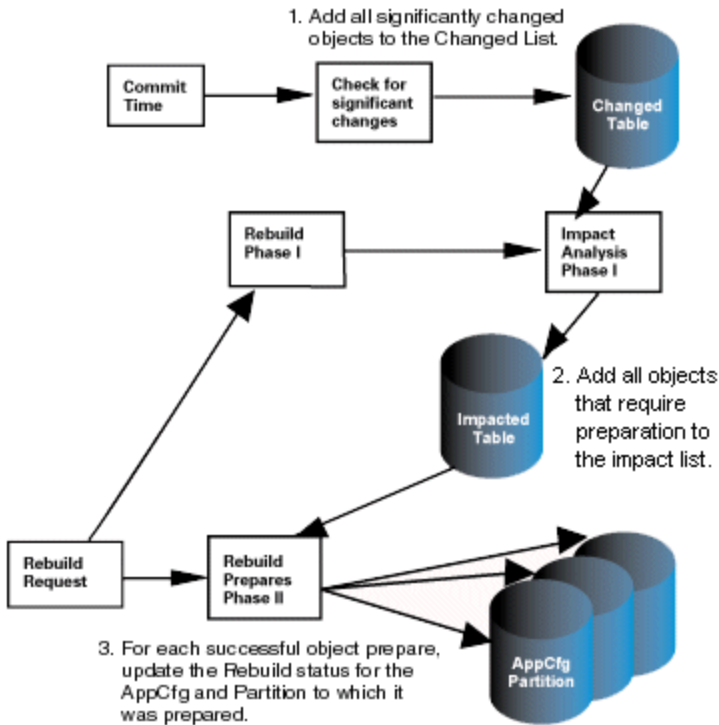
- [Rebuild Analysis](#)
- [Using Rebuild](#)

Rebuild Analysis

Rebuild analysis happens in a three-phase process. Part of the processing happens when you commit your changes to the repository. The remaining processing occurs when the user requests a Rebuild action. The following things happen during a Rebuild action:

1. The changed objects are identified and stored in a change list.*
2. The system analyzes the change impact to determine which additional objects need to be prepared. It looks at the impact time of each object to determine if the object in the impact list needs to be reprepared.
3. Every object in the selected Application Configuration or Partition is checked against the impact list to see if it has been prepared since the last impact.

Figure 7-1 Rebuild overview



The Rebuild process is discussed in more details in the following topics:

- [Significant Change Analysis](#)
- [Impact Analysis](#)
- [Rebuild within a Partition](#)
- [Notes on Rebuild Functionality](#)
- [Significant Changes by Object Types](#)

Significant Change Analysis

During development, the changed objects are identified and stored in a change list. When you commit the changes to the repository, the change list table is updated. The impact type is used to handle special circumstances other than the typical significant change.

Table 7-1 Changed list

Field	Purpose
Object ID	To relate this back to the affected object.
Impact type	Indicates whether this is source only change or other special circumstances.

Impact Analysis

During Rebuild Analysis, a full impact analysis is performed on the whole repository to determine which additional objects need to be prepared. Each object that requires preparation is added or updated in the Impact List table with the impact time. The following table displays sample table fields of the Impact table and the purpose of each field.

Table 7-2 Impact Table

Field	Purpose
Object ID	Identifies the impacted object.

Object Type	Identifies the type of the impacted object.
ImpactByID	Identifies the object that caused the impact.
ImpactByType	Identifies the type of the object that caused the impact.
Impact time	Identifies the time the object was last analyzed.

After the analysis is complete at Phase I, the objects are ready for Rebuild.

When an object is rebuilt, the rebuild time of the object is compared to the impact time of the changed object in the Impact List, and is prepared only if the rebuild time is different than the impact time. The rebuild time is then updated with the impact time if the object is prepared successfully. Only those objects in the selected Partition are analyzed for Rebuild in Phase II. The following table displays sample fields in the Rebuild Status Table and the purpose of each field.

Table 7-3 Rebuild Status table

Field	Purpose
Object ID	Identifies the rebuilt object.
Object Type	Identifies the rebuilt object type.
AppCfg ID	Identifies the parent Application Configuration of which this object is a child.
Partition ID	Identifies the parent Partition of which the object is a child.
Rebuild time	Identifies the time the object was last rebuilt.

Impact Analysis Examples

Impact analysis determines which objects are affected by a specific change or set of changes. For example, a rule might be affected by changing a field in a view of a child rule; however, a source change to a child rule does not affect a parent rule. Every impacted object is added to the Impact table.

*** = changed

@ = impacted

The following figure illustrates a Rule hierarchy. A change to Field 1 impacts the rules associated with it.

Figure 7-2 Field change

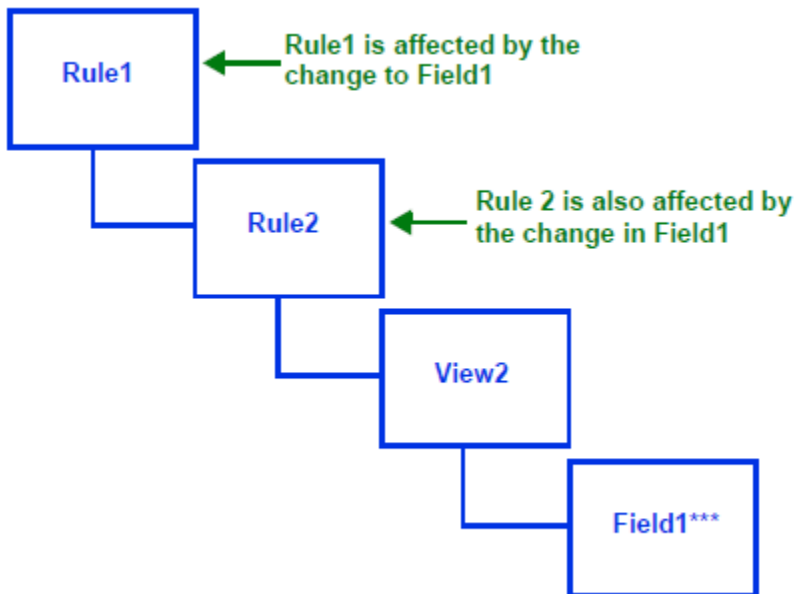
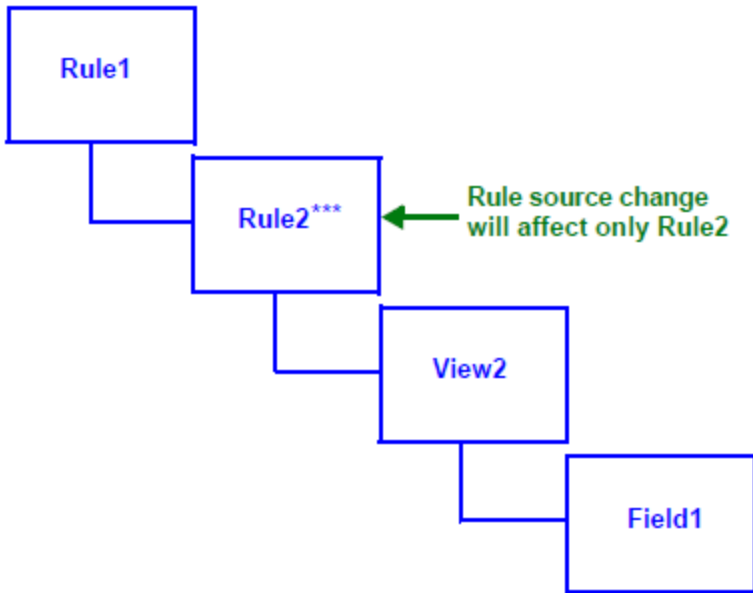


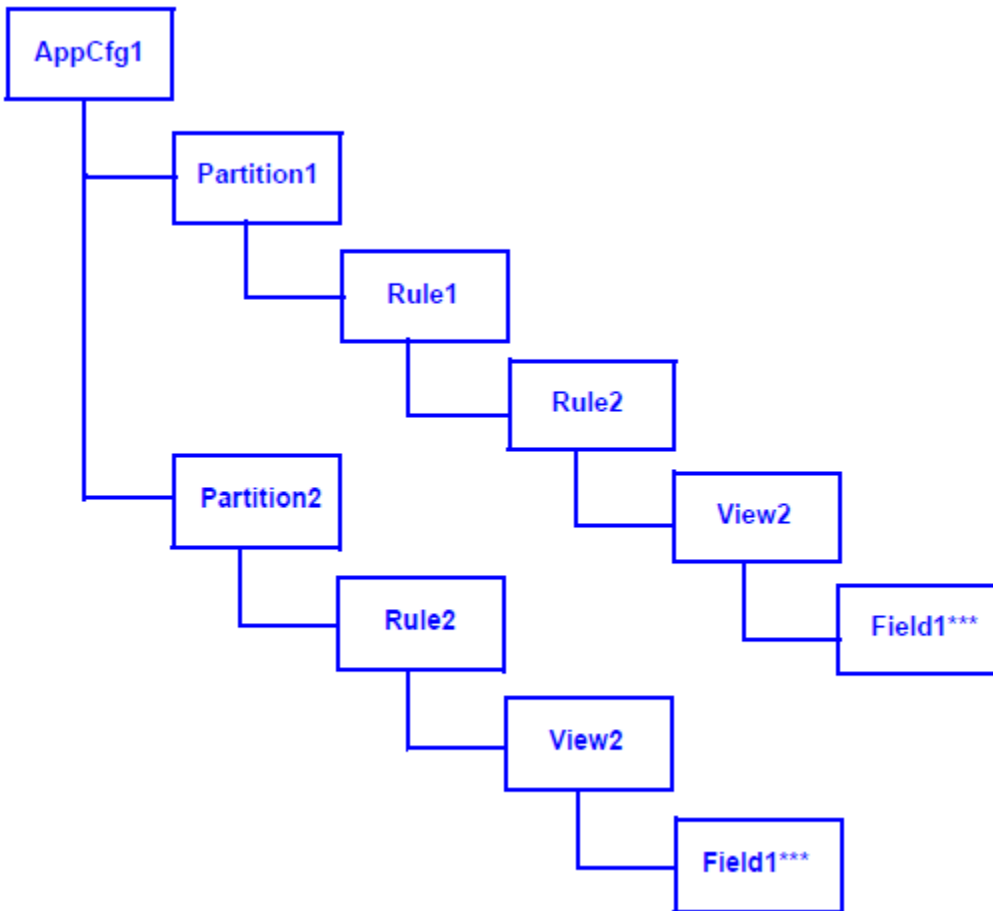
Figure 7-3 Rule source change



Rebuild within a Partition

The impact analysis results, which are stored in the Impact table, are used to determine which objects within a partition will be rebuilt.

Figure 7-4 Partition example



This example illustrates why it is best to rebuild from the Application Configuration object. In this example, Partition 1 and Partition 2 are server Partitions that contain different sets of objects. If a single change is made to Field 1, Rule 1 and Rule 2 are added to the impact table. However, if

only Partition 2 is rebuilt, then only Rule 2 is built, as Rule 1 is not in Partition 2.

If Partition 1 is built after Partition 2, not only will Rule 1 be built, but Rule 2 will be built again because it has not yet been built for that specific partition.

If you rebuild AppCfg1, both Partitions and all the children are marked for rebuild.



If you create a new window with text and place it on the panel, add the relationship to Rule 2, commit it, and run Rebuild. The new window and both rules are marked for Rebuild. According to the Impact list, only Rule 2 should be marked for Rebuild. To avoid this situation, delete the window from the repository and rerun Rebuild. This ensures that only Rule 2 is marked for prepare.

Reports can be prepared only for the host and not for the PC. If a child set exists in the rebuild list, its parent object will also be there. The exception is a Report object, which can only be rebuilt on the host. Therefore these objects will not be on the rebuild list if you are working with a Workgroup Repository.

Additionally, if a Window object or a Report object is on the rebuild list, the parent object is also on the rebuild list. Therefore, if WindowB has a parent, WindowA, and a child Set; when the Set is on the rebuild list, everything up to WindowA is also on the rebuild list.

Notes on Rebuild Functionality

All information related to the build status of an object is stored only in the repository. Rebuild does not prevent overwriting of libraries built from a different repository, partition, or application configuration.

What Happens When You Change a Partition?

Changes that affect the Partition or Application Configuration as a whole are not tracked by Rebuild. When making significant changes to how an application is partitioned or how a partition is deployed, it is recommended that you mark the partition as *UnPrepared* and rebuild the entire partition. For more information, refer to [Changing the Rebuild Status](#).



There is one exception: when the changes affect the content of a Servlet, the affected partition is displayed in the Prep Status tab.

For a Servlet partition, any changes made to any of the objects beneath the partition trigger the Partition to be marked for Rebuild. Other partitions do not have this behavior, as the Partition object is not needed to be updated when a child object is modified. This behavior is required because of the packaging that is involved. The Partition must be marked for Rebuild, so that the child object changes are incorporated into the final package of the application.

Rebuilding Migrated Partitions

For Partitions that are migrated into your application, there are 3 options. In order to use the disabled options of Mark All functionality for Rebuild, you must first build the entire Partition.

The Hierarchy Prepared of Mark All menu can be used to mark the selected partition or Application configuration as built. This is because the initial build updates the Rebuild Status table for that Partition. This will then enable the Rebuild Prepared and UnPrepared of Mark All menu. An initial build will perform an analysis of the entire repository and it will update the rebuild status. After the initial build, the Rebuild functionality can track the changes for rebuild. For this reason, the Mark All option is only available after the initial Partition build. For details on how the Mark All functionality options affect the Partitions and their objects, see [Changing the Rebuild Status](#).

If you select Mark All from the Application Configuration object, it rebuilds all the Partitions under that Application Configuration.

Significant Changes by Object Types

The following table indicates which objects are impacted when changes are made to the attributes of an object or relationship.

Table 7-4 Significant changes by objects types

Objects/Relationships	Attributes	Impacts
Bitmap	System Id	Report
	Type	Rule
	Implementation Name	Window

Component	System Id	Component
	Name	Rule
	Execution Environment	
	Language	
	Subroutine	
	DBMS Usage	
	Implementation Name	
File	System Id	Component
	Name	Rule
	File Type	
	Implementation Name	
Field	System Id	View
	Name	
	Field Picture — Storage	
	Field Picture — Display	
	Screen Literal — Long	
	Field Format	
	Field Length	
	Field Fraction	
	Range — Minimum Value	
	Range — Maximum value	
	Reference Table Name	
	Screen Literal — long	
	Implementation Name	
Form	System Id	Report
	Name	
	Form Environment	
	Base	
	Country Language	
Help	System Id	Window
	Name	
	Format	
	Description	
	Country Language	
Help Text	System Id	Window
	Name	
	Country Language	
Panel	System Id.	Window
	Name	

	Coordinate System	
	X-Resolution	
	Y-Resolution	
	Description	
	Country Language	
	Base	
	GUI	
Physical Event	System Id	Rule
	Name	
	Event Type	
	Event Class	
	Event Scope	
Report	Report Id	Report
	Name	Rule
	Execution Environment	
	Page Size	
	Line Size	
	Left Margin	
	Top Margin	
	Printer Type	
	Orientation	
	Implementation Name	
Rule	System Id	Rule
	Name	
	Execution Environment	
	DB2 Plan Name	
	DBMS Usage	
	Execution Mode	
	CICS/IMS transaction Id	
	Implementation Name	
	Package	
	Isolation Mode	
Section	System Id	Report
	Name	
	Implementation Name	
Set	System Id.	Component
	Name	Report
	Field Picture — Storage	Rule
	Set Format	Window

	Set Length	
	Set Fraction	
	Implementation Name	
	Style	
	Representation Length	
Symbol	System Id	Set
	Name	
	Define	
	Encoding	
	Display	
Values	System Id	Set
	Name	
	Value	
View	System Id	Component
	Name	File
	Implementation Name	Report
		Rule
		View
		Window
Window Content	System Id	Window
	Name	
	GUI	
Report/Section	Type	Report
	Break Sequence Number	
	Page Placement	
	Break Field	
	Break Qualifier	
	Left Margin	
	Print Option	
Set/Value	Sequence Number	Set
	Symbol	
Entity/View	Usage	Component
		File
		Report
		Rule
		View
		Window
View/Entity	Sequence Number	Field
	Occurs Number of Times	View

Null Indicator for DB2

Using Rebuild

The following procedures are listed in this section:

- [Rebuilding a Partition](#)
- [Rebuilding an Application Configuration](#)
- [Sorting the Rebuild Report](#)
- [Saving the Rebuild Report](#)
- [Selecting the Active Configuration for Rebuild or Rebuild Report](#)
- [Changing the Rebuild Status](#)

Rebuilding a Partition

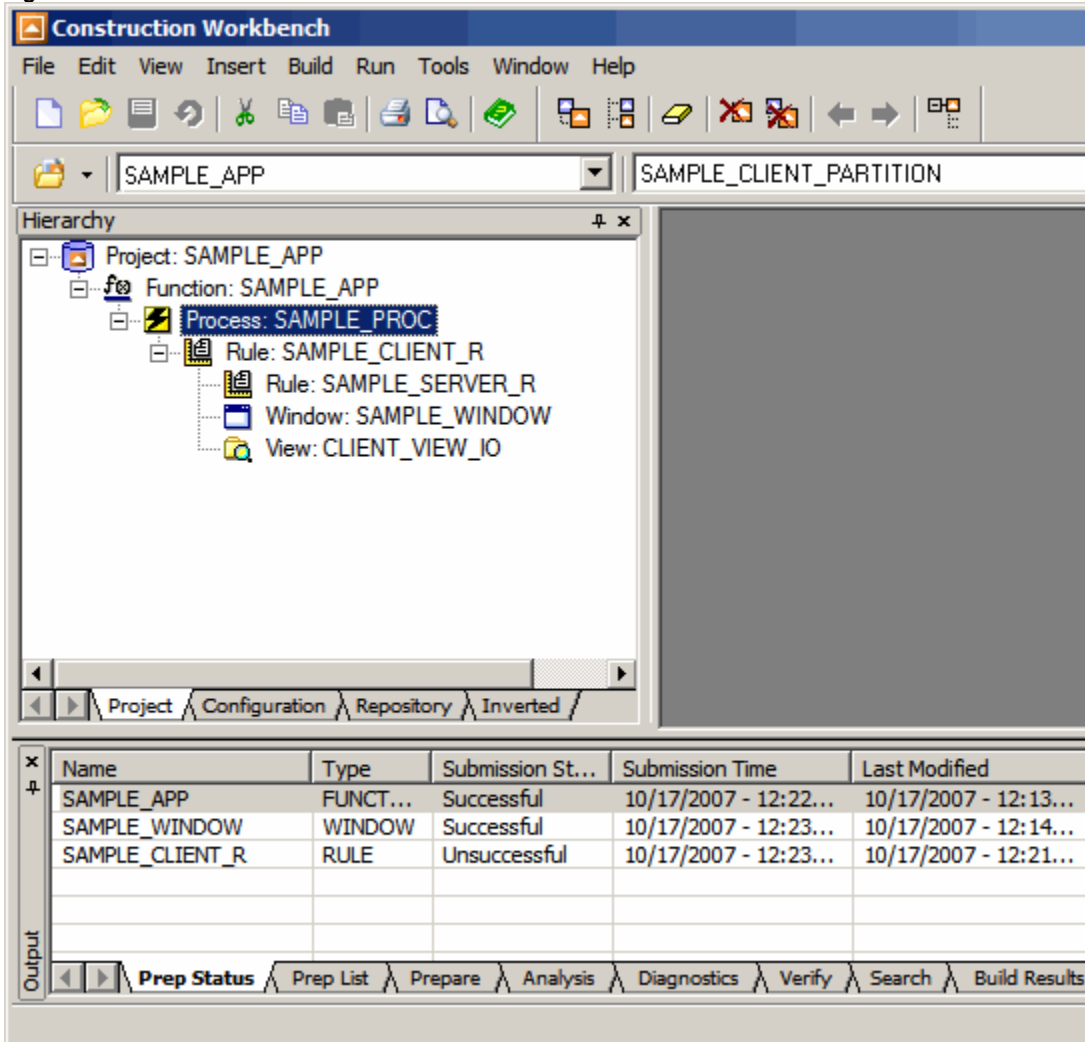
All the objects that are changed in the Partition and contained in the Impact List will be rebuilt. Once the changed object has been rebuilt, the rebuild time is replaced with the impact time of the changed object. The preparation order for Rebuild is the same as for a Super Prepare. Prepares do not update any significant fields in the repository.

Rebuilding a Partition


Follow the steps below to rebuild objects in a Partition:

1. Connect to a Workgroup Repository.
2. From within Construction Workbench, choose **File > Open Project**. You can either open an existing project or create a new one.

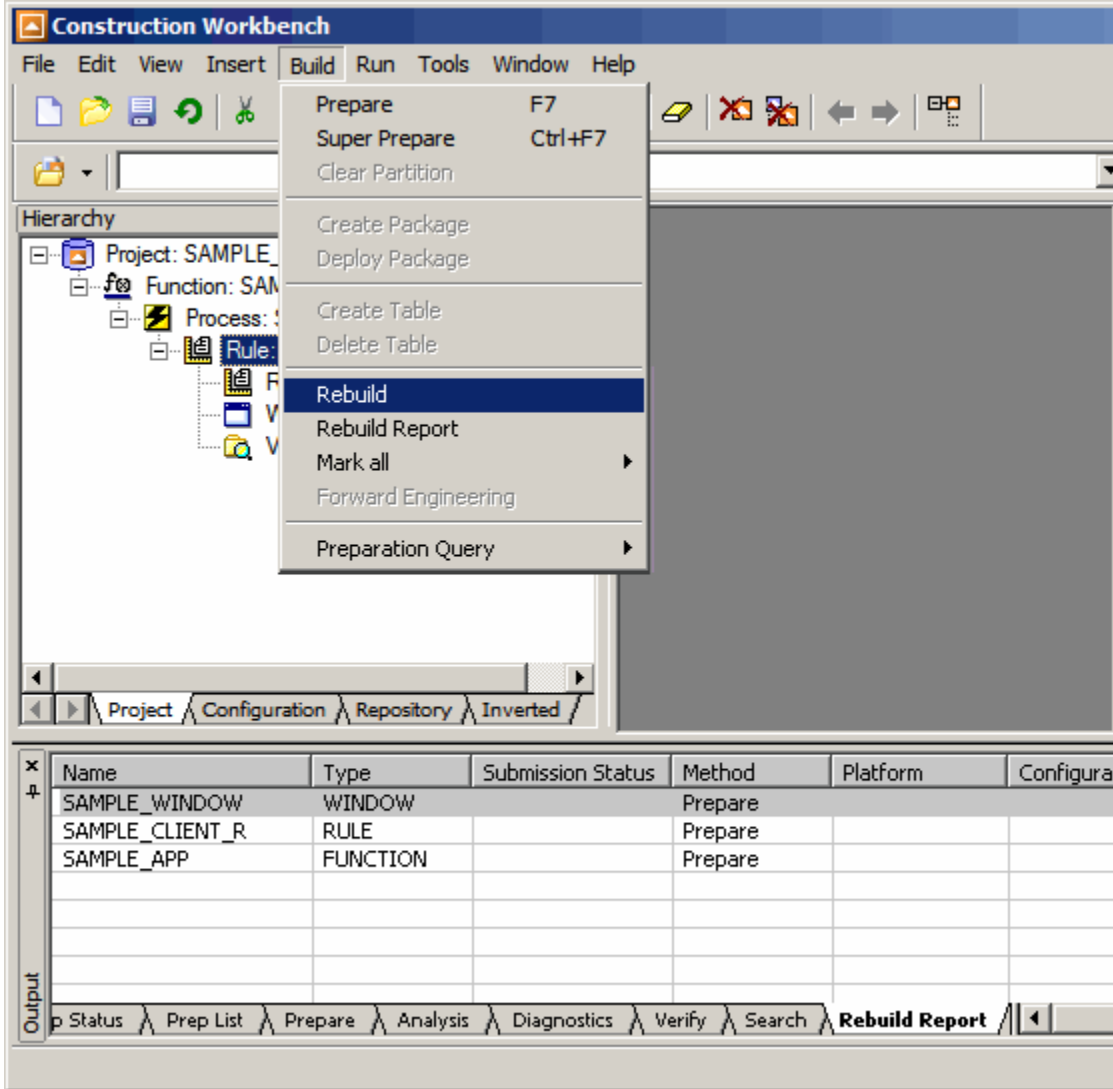
Figure 7-5 Construction Workbench Rebuild Interface




- You can rebuild from the **Project** tab or from the **Configuration** tab (Figure 7-5).
When rebuilding from the **Project** tab, the default Application Configuration and the default Partition will be submitted for Rebuild. Any items that are selected in the Hierarchy window have no impact on what is rebuilt.
When rebuilding from the **Configuration** tab, select the Partition object that you want to rebuild, rebuild report, or **Mark All**.
- From the Construction Workbench, select **Build > Rebuild** (Figure 7-6).

 From the Project tab, you can only run Rebuild when you are in Distributed Mode. Choose **File > Project Options** and select **Distributed Application**.

- The Rebuild status displays in the status box located at the bottom of the Construction Workbench.
- Figure 7-6 Project tab rebuild**



 For Partitions that are migrated into your application, you must first build the entire partition before you can use the Mark All functionality for Rebuild. This is because the initial build updates the Rebuild Status table for that partition. After the initial build, the Rebuild functionality can track the changes for rebuild. For this reason, the Mark All option is only available after the initial partition build.

Running a Rebuild Report on a Partition


Follow the steps:

- Select the Partition using the options in the drop-down box at the top of the Construction Workbench
- From the Hierarchy, select the object for which you want to obtain a Rebuild Report.
- Select **Build > Rebuild Report** (Figure 7-6).

4. The report results display at the bottom of the Construction Workbench.

After running a Rebuild Report, you have the option to submit one or more selected objects or to submit all of the objects for Rebuild preparation. Use the following steps:

1. Right-click on the Rebuild Report.
2. Click **Submit All** to select all objects for Rebuild, or click **Submit** to select one or more chosen objects.

 The **Submit** option is available only when you have already selected one object or a list of objects from the Rebuild Report.

The object or objects are removed from the Report pane and submitted for Rebuild preparation.

Rebuilding an Application Configuration

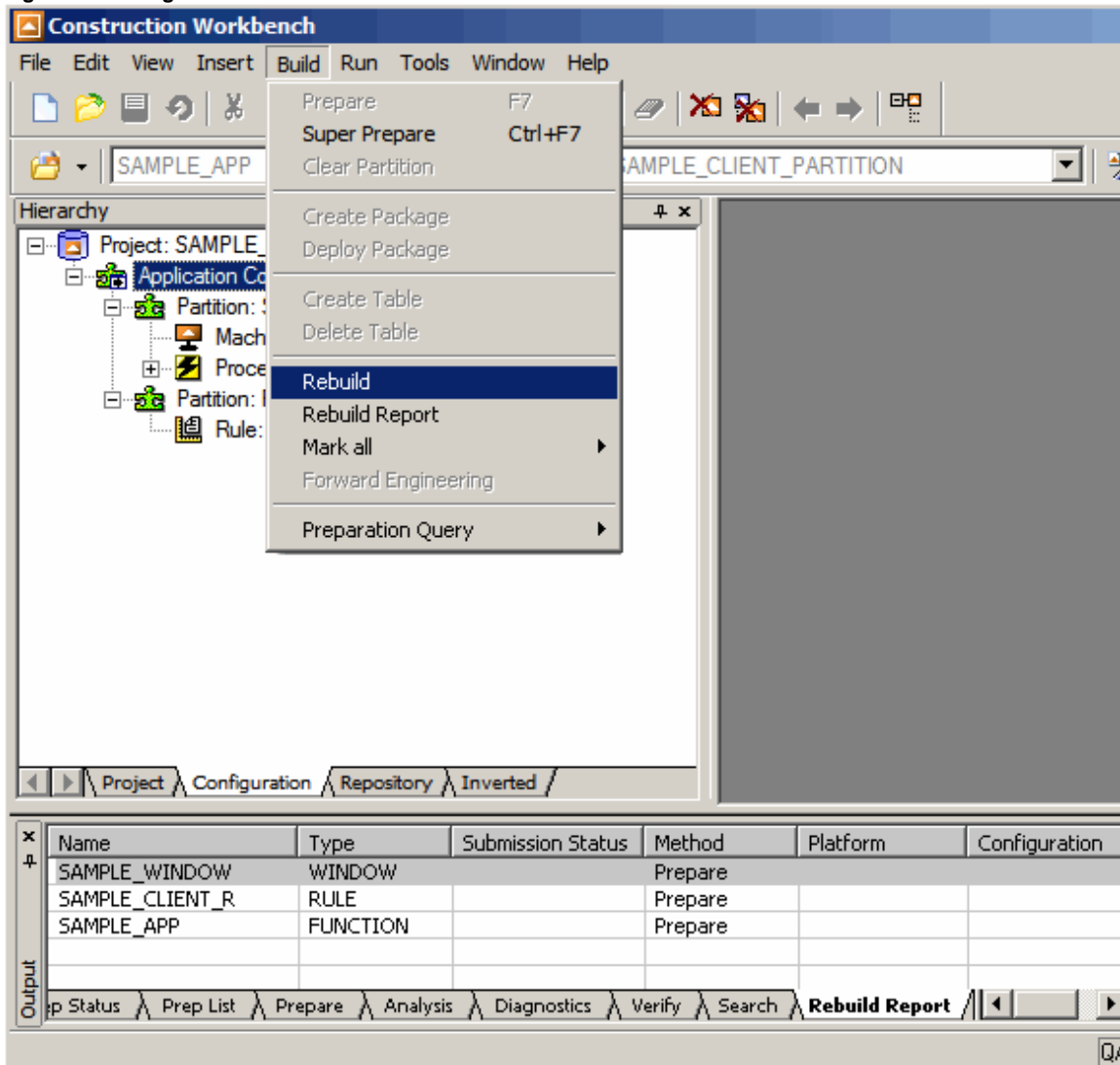
Rebuilding an Application Configuration results in a rebuild of each child Partition.

Rebuilding an Application Configuration

Follow the steps:

1. From Construction Workbench, select the **Configuration** tab of the Hierarchy window.

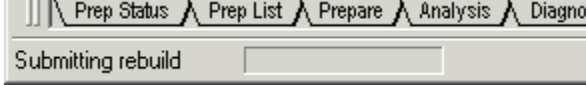
Figure 7-7 Configuration tab



2. Select the Application Configuration you want to rebuild.
3. From Construction Workbench, select **Build > Rebuild**, or right-click on the Application Configuration name and select **Rebuild**.

- The status of the rebuild is displayed in the lower left corner of the window.

Figure 7-8 Rebuild status display



Running a Rebuild Report on an Application Configuration

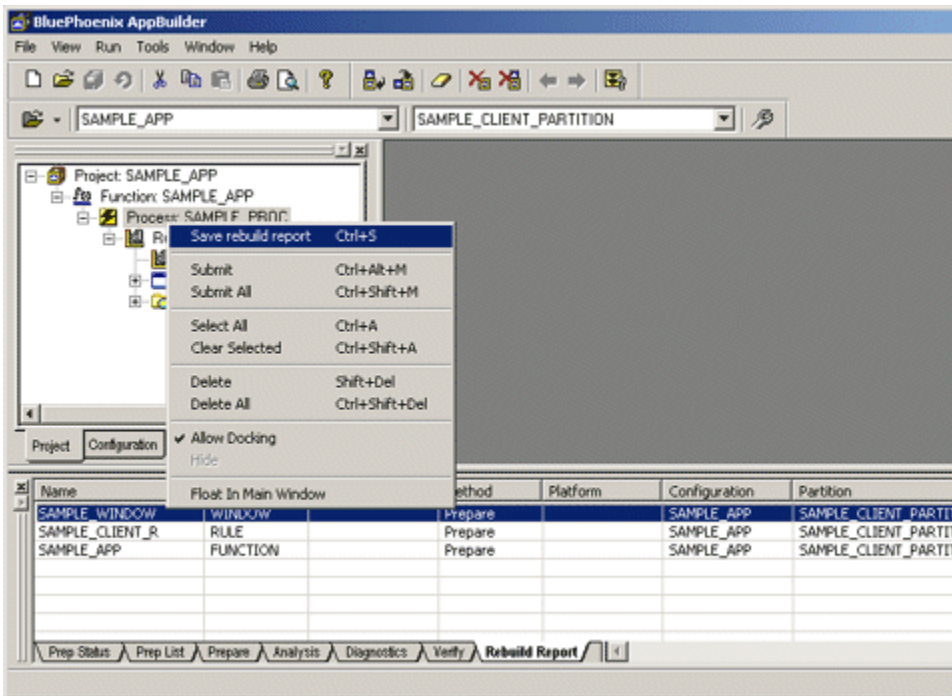
Follow the steps:

- Select the Application Configuration using the options in the drop-down box at the top of the Construction Workbench.
- From Construction Workbench, select **Build > Rebuild Report** (Figure 7-6).
- The report results display at the bottom of the Construction Workbench window.

Sorting the Rebuild Report

You can sort the report results in the window by clicking on the column heading.

Figure 7-9 Sorting and saving the report results



Saving the Rebuild Report

Save the Rebuild Report by right-clicking the Rebuild Report window and selecting **Save Rebuild Report**. Name the file, choose the destination location, and click **OK** (Figure 7-9).

Selecting the Active Configuration for Rebuild or Rebuild Report

When in distributed mode, the active configuration is used to submit a rebuild when working in the Project, Repository, or Inverted tab of the Hierarchy window. To set the active configuration, go to **File > Project Options**, select the Application Configuration in the drop-down list and then select one of the available Partitions.

Changing the Rebuild Status

Changing the rebuild status is most useful when making significant changes to the way an application is deployed or when you are deploying an application for the first time.

Mark All allows you to rebuild the children of a selected object. There are three options for Mark All:

- Hierarchy Prepared – Marks all objects in a Partition as prepared. The Partition does not have to be previously prepared.
- Rebuild Prepared – Marks all rebuilt objects in a Partition as prepared. To enable this option, the Partition must be previously prepared. This option works only for updates. **Rebuild Prepared** does an update to all the objects of the selected Partition, so the Partition should have already been rebuilt.
- Unprepared – Clears the rebuild status for all objects in the Partition. This causes all the objects in the Partition to be selected and prepared in the next Rebuild. When an Application Configuration is selected, the Mark All operation affects all child Partitions of this Application Configuration.

To use these options, from the Construction Workbench, click **Build > Mark All**, then choose the selected option.

Moving Data Between PC Repositories

AppBuilder provides three ways to move data from one repository to another. Each method is intended for a unique purpose.

Table 8-1 AppBuilder tools for moving data

Migration Import / Export	Migration is used to move specific objects from one repository to another. Objects can be migrated individually or as entire object hierarchies. There are two forms of migration: Selective migration allows you to pick specific objects to be migrated. Direct migration simply allows you to migrate the entire contents of a Unit of Work.
Repository Migration Utility	Copies the entire database for back up purposes. Repository Migration is not limited to import and export on the same type of database. If you are doing a backup using this utility, back up the freeway.id file as well. These utilities back up the data without including repository information and settings. Refer to Full Repository Migration for more information.
Repository Replication Tool	Copies the entire source database and imports it into an existing repository. You can use normal repository replication or use XML as the data transfer method. For details, see Repository Replicator .

AppBuilder repository migration tools provide a graphical user interface (GUI) on the client workstation that allows users to move objects between group or local repositories and the mainframe repository. The migration tool functions are accessed in the Repository Administration Tool. The following topics explain migrating data between repositories:

- [Migration Overview](#)
- [Understanding Migration Phases](#)
- [Migration Export for Repository Objects](#)
- [Performing a Migration Import](#)
- [Migration Import After Analysis](#)
- [Using Migration Results](#)
- [Deleting a Migration Object](#)

Migration Overview

Most development environments require data migration, that is the moving of data from one repository to another. For instance, AppBuilder objects can be moved from a development to a quality assurance repository. After the application is installed in the production environment, the objects can be moved to a "maintenance" repository. These objects can be migrated individually or as entire object hierarchies. Migrations are performed by creating a migration object, relating the AppBuilder repository objects to the migration object, and exporting the migration object to produce a set of migration files. The migration files are then imported into another (target) repository.

Resolving Short Name Conflicts on Migration

A short name (SYSTEMID) conflict occurs when the incoming object's long name (NAME) is the same as the existing object's long name but the short name is different. When a short name conflict occurs on a migration, the analyze details log is displayed on the workstation. The objects that have conflicts in the short name are highlighted. The conflict status can be changed by selecting one of the three options on the popup menu. The user can choose one of following options and press OK:

Table 8-2 Options for Short Name Conflicts

Option	Description
Overwrite	This option deletes the existing object from within the repository and all relationships and files attached to it. Then it imports the incoming object and the relationships and files that come with it.
Merge	This option causes an update of the like object including the short name of the incoming object. It preserves all the relationships and files of the existing object. It also adds or updates the relationships and files from the incoming migration.
Cancel	This option has the same effect that "Update off" would have for a non-conflicted object. It toggles off all relationships and files of the conflicting object.

Understanding Migration Phases

There are four phases to a repository data migration:

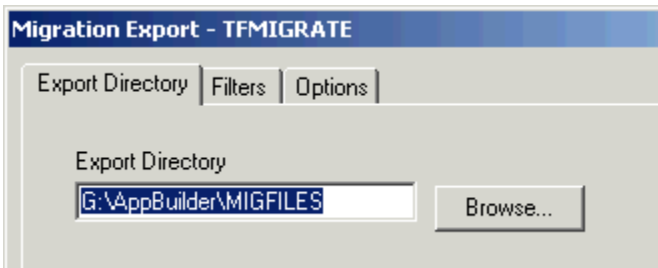
- Export
- Load
- Analyze
- Import

Both the Import and the Export phases of migration are performed on a client workstation.

The first phase, Export, is executed while connected to the *source* repository. The subsequent three stages, Load, Analyze, and Import, are executed while linked to the *target* repository.

During the Export phase, data from the source repository is extracted into migration files. These files are created in the directory you specify. If you specify a directory in the Export Directory field that does not yet exist, the system prompts you to create the directory.

Figure 8-1 Migration Export Setup Window



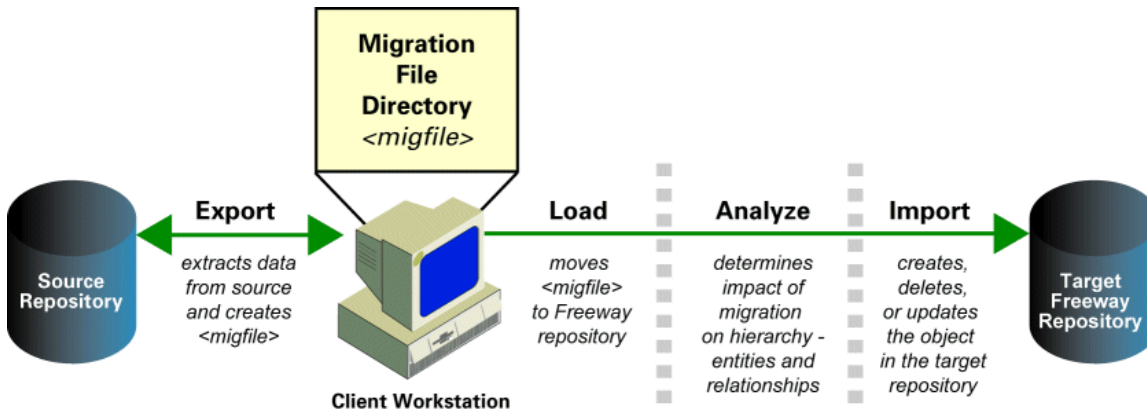
The Load and Analyze phases are executed during an Import of data into a repository. The Load phase, creates a set of temporary .dat files that are used in the Analysis and Import phases.

After completing the Load phase on the migration object, the Analyze phase can be executed. Analyze generates information about the entities and relationships included in the migration hierarchy. During Analyze, you decide whether or not to migrate the objects to the target repository.

The Analyze phase must be executed to determine which objects exported from the source repository are to be created or updated in the target repository. You have the option of manipulating which objects will be imported.

Once the Analyze phase is complete, execute the Import phase to actually create, delete, or update the repository object in the target repository. During this phase, actual ownership for the objects being maintained is assigned. The object retains the security information related to project and owner.

Figure 8-2 Four Phases of Migration



Migration Export for Repository Objects

A Migration Export is used to create a set of migration files that can be imported into another repository. You can perform an Export from the

Repository Administration tool; however, you must have system administrator authority. The export process is comprised of the following tasks:

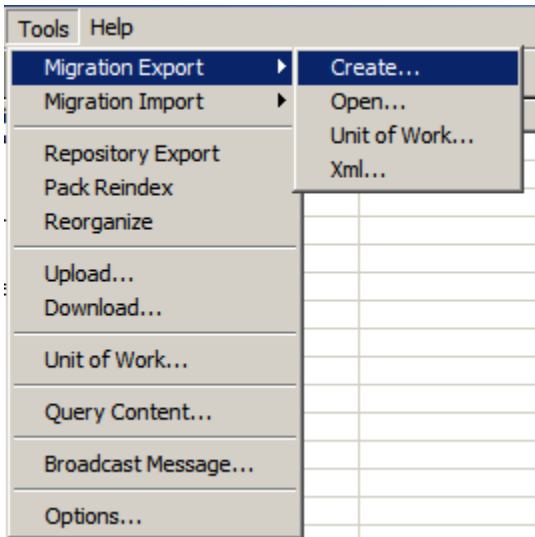
1. [Create a Migration Object](#) or [Open a Migration Object](#)
2. [Export a Unit of Work](#)
3. [Relate the AppBuilder Objects](#) to the migration object that will be exported.
4. [Set the Migration Object Scope](#)
5. [Display the Export Hierarchy](#)
6. [Export the Migration Object](#)
7. [Check the Export Log File](#) to ensure the export was successful.

Create a Migration Object

Before you perform data migration exports, create a migration object:

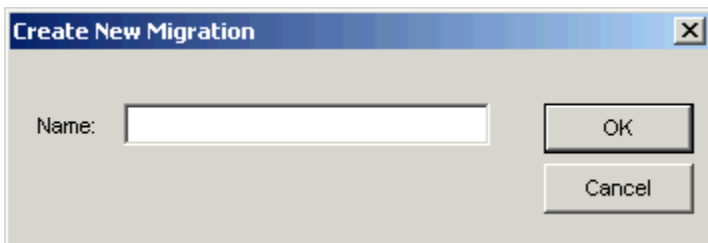
1. Connect to a repository.
You can use the migration facility from any tab in the Repository Administration tool.
2. Select **Tools > Migration Export > Create**.

Figure 8-3 Migration Export Tools Menu



The Create New Migration window displays.

Figure 8-4 Create New Migration window



3. Type a name for the migration object and click **OK**.
The system displays a message indicating whether or not the create was successful.
4. Click **OK**.

The Migration Export window displays. The next step is to [Relate the AppBuilder Objects](#).

Open a Migration Object

If a migration object already exists in the repository. You must open it before you can export it. You can also open the migration object to modify its contents.

1. Select **Tools > Migration Export > Open**.
The Migration Query dialog displays.

Figure 8-5 Migration Query dialog

For information about the Migration Export window tabs, refer to [Table 8-9](#). The next step is to [Set the Migration Object Scope](#).



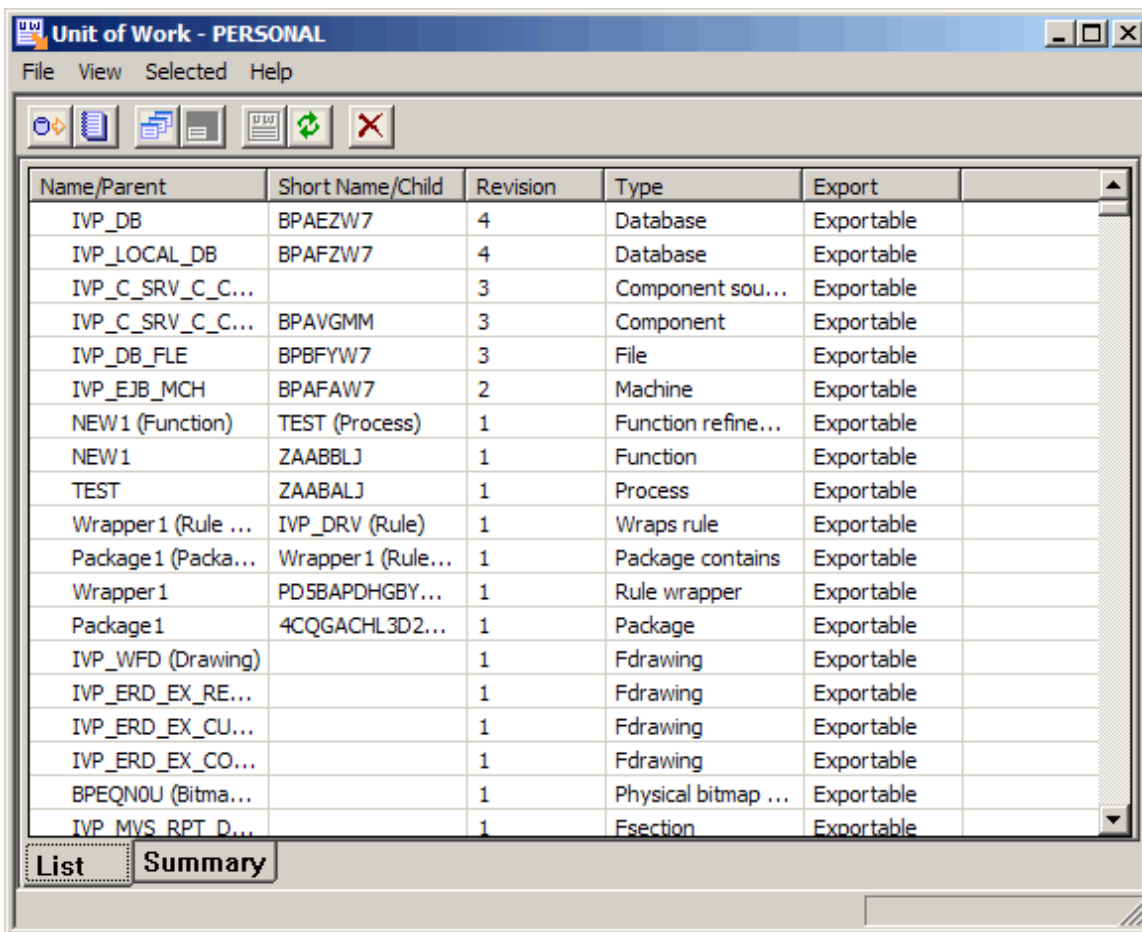
The Summary tab displays the export log file in text format for the UOW that is displayed in the UOW window. You must activate the Export option from the File menu before the log file is displayed in the Summary tab. This differs from the Export Log option on the View menu, in that the View > Export Log option displays the most recent log file that was generated, regardless of the current export. If there are no log files saved on your system, the Export Log option is unavailable from the View menu.

Export a Unit of Work

Unit of Work (UOW) is a facility to track your changes for ease in upload and download, and migration export. With a Personal Repository, the UOW feature is automatic and assists you in upload and download between the mainframe and your local repository. With a Workgroup Repository, UOW is an installable component; you must have the UOW feature installed. With a Workgroup Repository, use UOW to easily migrate your changes from one repository to another. To create a migration object of your UOW, you must first activate Migration Export:

1. Click **Tools > Migration Export > Unit of Work**.
Your Unit of Work window displays.

Figure 8-7 Unit of Work window with UOW loaded



From this window, you can sort and filter your display to make it easier to find what is in your UOW. You can also flush specific items from your UOW before export, and view the properties of the UOW and properties of a specific object in the UOW.

- To sort the UOW, click **View > Sort Order** or click the column heading to sort by that column.
- To filter the UOW display, click **View > Filter**.

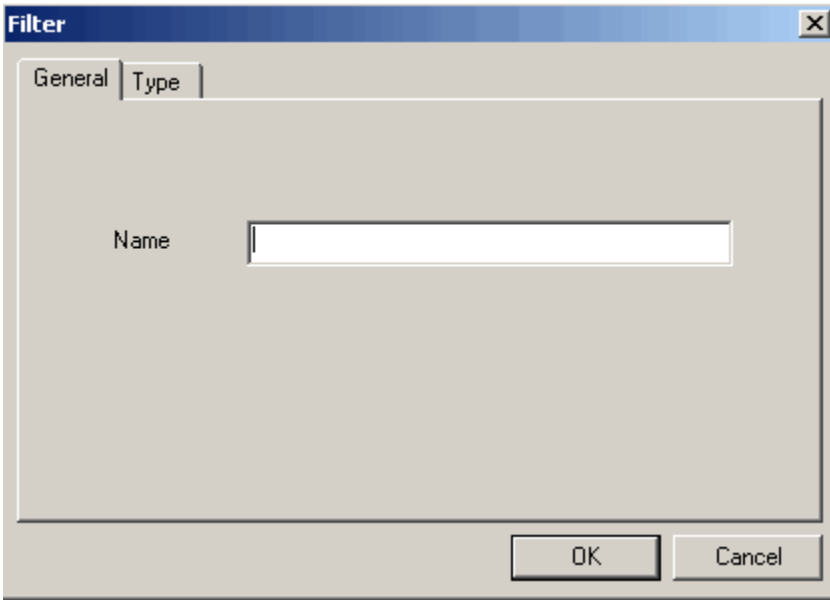
Filter the UOW

In addition to viewing content similarities, differences, or all content, you can filter the display.

1. Click **View > Filter Results**. The Filter window displays.
The filter option does not remove any objects from the UOW; it simply displays only those objects that meet your filter parameters.

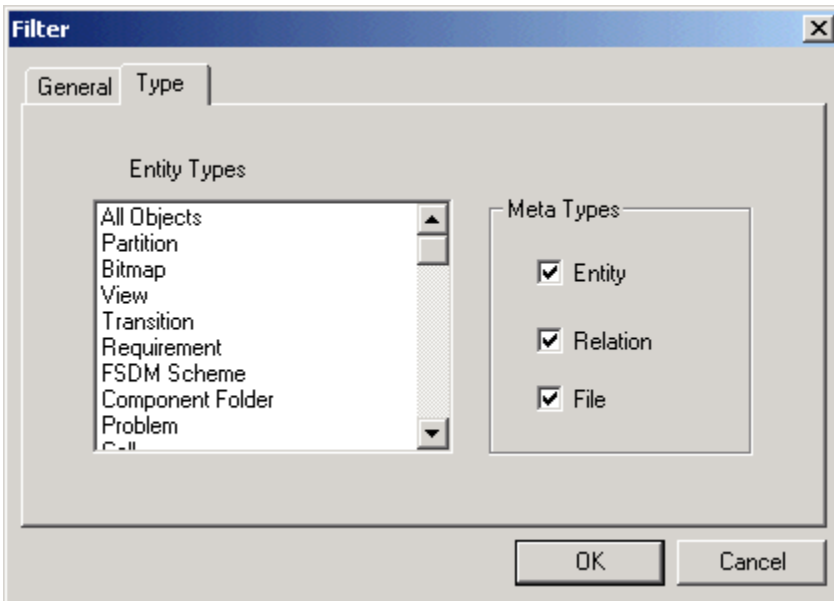
Making it easier to find what you are looking for in the UOW.

Figure 8-8 FilterUOW General tab



From the General tab, you can filter the display by name or partial name.

Figure 8-9 Filter UOW Type tab



From the Type tab, you can filter the display by Entity type or Meta type.

Refreshing the UOW

You can refresh the list in the UOW window. Click **View > Refresh** to refresh your list and include any new changes.

Flushing Selected Objects from the UOW

To flush specific objects from the UOW:

1. Highlight one or more items in your UOW display.
2. Click **Selected > Flush**.
3. Click **Yes** when prompted to flush the selected items.

Flushing All Objects from the UOW

After migrating your UOW, you may want to clear the UOW so it is empty for the next set of changes. To flush the entire UOW, from the UOW window, click **File > Flush** .

Viewing Properties of the UOW

You can view the properties of your entire UOW or the properties of selected items in the UOW.

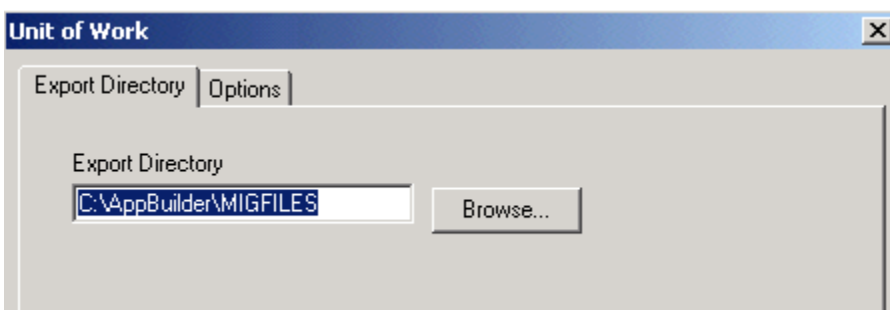
- To view properties of the entire UOW, click **View > Properties** .
- To view properties of selected items in the UOW, highlight the items in the UOW display and click **Selected > Properties** .

Performing the UOW Export

After you have loaded the UOW objects into your display, complete the Export operation.

1. Click **File > Export**. A second Unit of Work window displays.
2. Click **Browse** to navigate to the directory for your UOW export file.

Figure 8-10 Export Properties UOW



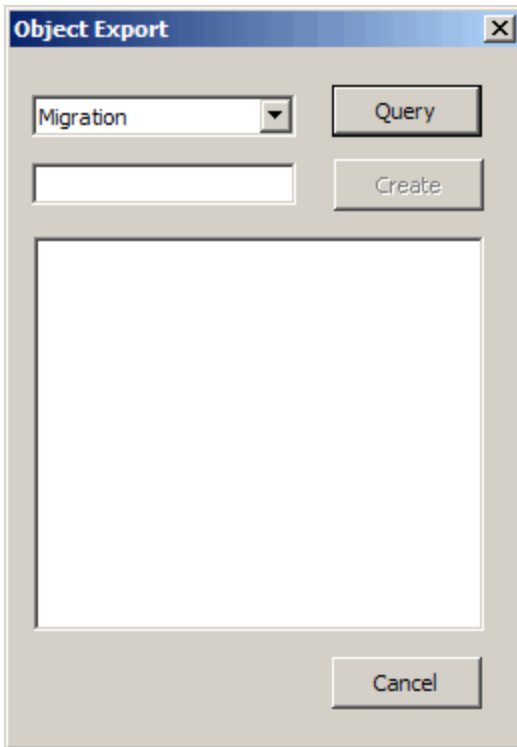
3. Set specific export options from the options tab. For a description of each export option, see [Table 8-12](#).

Exporting to XML

You can export objects in the source repository to XML files which can be used for data migration.

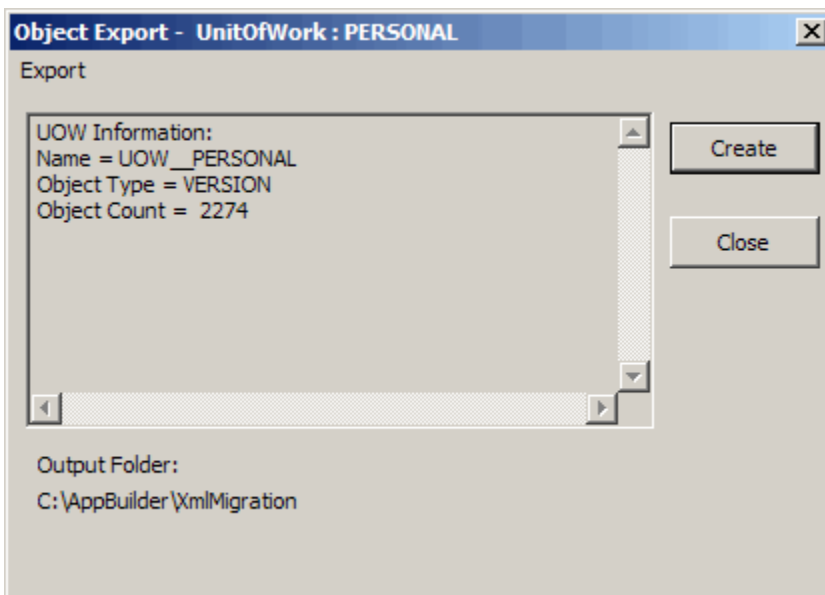
1. In the Migration Export command (see [Figure 8-3](#)), select **XML**.
2. The Object Export dialog displays, allowing you to specify whether to export a migration object or a unit of work as XML. You can also create an object of either kind from this dialog.

Figure 8-11 Object Export dialog



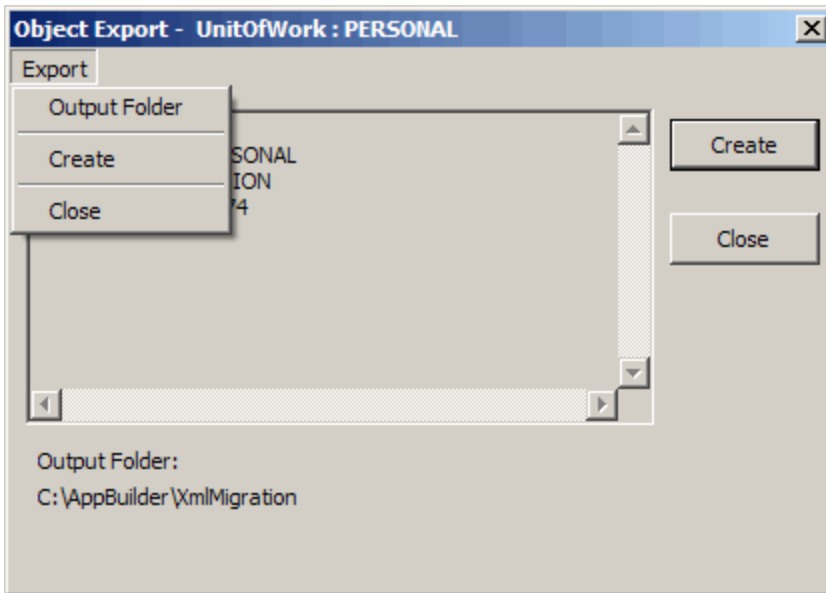
3. Once you have specified the object, double click on it.
4. The Export dialog appears, showing the contents of the Unit of Work or the Migration object.

Figure 8-12 Export dialog



5. There are several options available from this dialog. **Create** moves directly to creating the XML from the specified objects. **Close** cancels the operation. You can also change the output folder by using the Export menu at the top of the dialog.

Figure 8-13 Export Menu



- Once you have specified the settings you desire, click **Create**.

Export creates the following files in the migration folder. The log files are for display purposes, the XML files hold the repository object properties and data. The "mdf" in the extension stands for Migration Definition File.

Detail.Export.mdf.xml - holds migration object data.

Entity.Export.log

Entity.Export.mdf.xml

File.Export.log

File.Export.mdf.xml

Relationship.Export.log

Relationship.Export.mdf.xml

Summary.Export.log

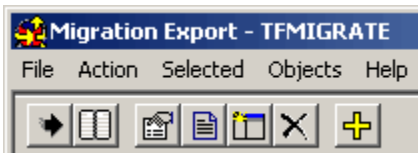
Actual source files are written out independently.

i.e. _97_3877.Src

Understanding the Migration Export Toolbar

The tools available in the Migration Export toolbar offer an alternative method to access the same functionality as the corresponding menu items in the Migration Export window drop-down menu ([Figure 8-6](#)).

Figure 8-14 Export Toolbar Icons

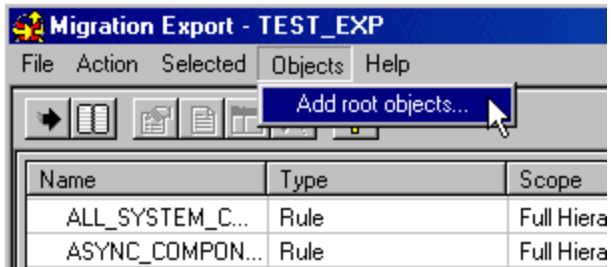


Relate the AppBuilder Objects

Follow the steps:

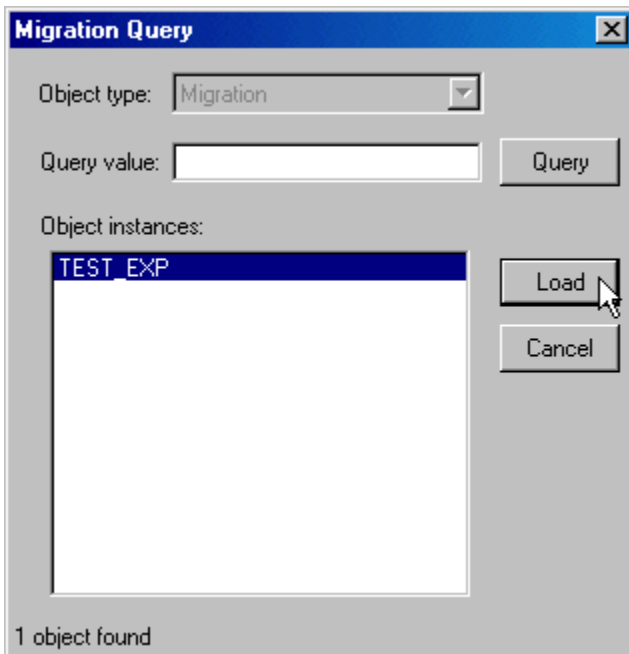
- From the Migration Export window, click **Objects > Add root objects** to display the repository objects that will be associated with the migration object.

Figure 8-15 Adding Root Objects



The Migration Query window displays with the Object type **Migration** displayed.

Figure 8-16 Migration Export Query window



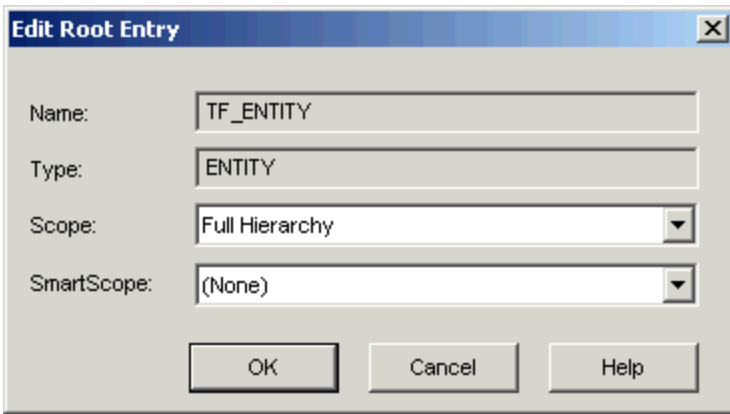
2. Click **Query**.
A list of Migration objects in the repository is displayed.
3. From the Object instances pane, select the migration object you want to export and click **Load**.

Set the Migration Object Scope

The root object that you have chosen to include in the export may or may not have children associated with it. Scope specifies the number (or level) of child objects that will be included with the root object. The default scope setting is Full Hierarchy, which means that all children of the root object will be included in the export. Modify entity scope settings for any object listed in the Migration Export window by highlighting that entity and clicking **Selected > Export properties...**

The Edit Root Entity Window displays.

Figure 8-17 Edit Root Entity Window



1. Select the scope and SmartScope settings, if any.
2. Click **OK**.

The following table lists the available scope options.

Table 8-3 Migration Scope Options

Scope Options	Description
Data Universe (Two Levels)	Exports the root entity and any child entities two levels down the hierarchy.
Drawing	All visible objects in the drawing, plus their relations to objects not represented in the migrated drawing will be exported up to two-levels. For ERD drawings, only the object types "Entity" and "Relation" are exported. For all other drawings, all pertinent objects existing in the source repository are exported.
Entity Model	Can be used with the following objects for export: <ul style="list-style-type: none"> • Business Object • Entity • Attribute • Data Type • Set (includes the symbols used to make up the set) • Identifier
Entity Only	Exports the migration object only.
Full Hierarchy	Exports the migration child object and all its child hierarchies in the migration.
One Level	Exports the root entity and any child entities one level down the hierarchy.
Preparable Entity	(Limited to preparable objects only.) Exports the root migration object and all of the objects in the hierarchy required to successfully prepare the root object. The scope of the root migration object depends on the level of the hierarchy. The following are objects included with the root object: Bitmap, Component, File, Physical Event, Report, Rule, Section, Set, Symbol, Value, View, Window. The following are objects included with a child entity: Section, Symbol, Value, View.
Table Model	Can be used with the following objects for export: <ul style="list-style-type: none"> • Database • Table • Key

The Hierarchy tab on the Migration Export window displays the selected scope hierarchy in text format. Refer to [Display the Export Hierarchy](#) for more information.

The next step is to [Export the Migration Object](#).

Using SmartScope Options

SmartScope narrows your scope settings further. SmartScope settings are also available from this window. These options can further limit the scope of objects selected for export using the options described in the following table.

Table 8-4 SmartScope Options

SmartScope Options	Description
ONLY_SYSTEM_COMPONENTS	Exported objects are associated with the SEER1 and SEER2 projects.
NO_SYSTEM_COMPONENTS	Exported objects are <i>not</i> associated with SEER1 and SEER2 projects.
RULE_TO_LEAF	Exported objects of the type listed in Table 8-5 that are in the hierarchy, regardless of which project they are associated with.
RULE_TO_LEAF_NO_SYS_COMPS	Exported objects of the type listed in Table 8-5 that are in the hierarchy and <i>not</i> associated with SEER1 or SEER2 projects.
SETS_TO_LEAF	Exported objects of the type listed in Table 8-6 that are in the hierarchy, regardless of which project they are associated with.
SETS_TO_LEAF_NO_SYS_COMPS	Exported objects of the type listed in Table 8-6 that are in the hierarchy and associated with SEER1 or SEER2 projects.
WINDOW_TO_LEAF	Exported objects are of the type listed in Table 8-7 , regardless of the project they are associated with.
WINDOW_TO_LEAF_NO_SYS_COMPS	Exported objects are of the type listed in Table 8-7 , <i>not</i> associated with SEER1 or SEER2 projects.
WITHOUT_FIELDS	Exports all objects, except field objects.
(None)	No options selected



The relationships to the field objects are created during the import, if the field objects exist in the target repository and the relationships are not already there.

Table 8-5 Rule_to_Leaf SmartScope Object Types

Entities	Relationships
bitmap	bitmap_has_bitmap_implementation
bitmap_implementation	component_owns_view
component	component_refers_to
field	component_uses_component
file	field_uses_language
help_text	file_is_accessed_by_component
language	file_is_accessed_by_rule
panel	file_is_forwarded_to_file
physical_event	file_is_keyed_to_file
report	file_owns_view
rule	physical_event_has_rule
section	physical_event_owns_view
set	report_contains_section
symbol	report_refers_to_set
value	rule_converses_report
view	rule_converses_window
window_content	rule_depends_on_rule

windows	rule_has_bitmap
	rule_owns_view
	rule_refers_to_set
	rule_triggers_physical_event
	rule_uses_component
	rule_uses_rule
	section_owns_view
	section_uses_language
	set_contains_symbol
	set_contains_value
	symbol_uses_language
	view_includes_field
	view_includes_view
	window_content_has_help_text
	window_content_has_panel
	window_has_bitmap
	window_has_window_content
	window_owns_view
	window_refers_to_set



Any entity or relationship of the type listed in [Table 8-5](#) will be exported regardless of whether these objects appear as part of a window's subhierarchy. For example, if you export the Process, and choose the Rule_to_Leaf SmartScope, the export will include BITMAP_A, BITMAP_B, RULE AND VIEW.

Table 8-6 Sets_to_Leaf SmartScope Object Types

Entities	Relationships
language	set_contains_symbol
set	set_contains_values
symbol	symbol_uses_language

Table 8-7 Window_to_Leaf SmartScope Object Types

Entities	Relationships
bitmap	bitmap_has
bitmap_implementation	bitmap_implementation
field	field_uses_languages
help_text	set_contains_values
language	set_contains_symbols

panel set	symbol_uses_language
symbol	view_include_field
value	view_includes_view
view	window_has_bitmap
window	window_owns_view
window_content	window_refers_to
	window_content_has_help_text
	window_content_has_panel
	window_has_window_content

Display the Export Hierarchy

To display the objects and relationships in .txt format, from the Migration Export window ([Figure 8-6](#)), select **Action > Display** and then select the **Hierarchy Display** tab. It is possible to produce this information before the actual Export action.

Figure 8-18 Sample Export Hierarchy Display

```
* Full Hierarchy *
-> COMPONENT (DB2)
  -> VIEW (SQLCA)
    -> FIELD (SQLCABC)
    -> FIELD (SQLCAID)
    -> FIELD (SQLCODE)
    -> VIEW (SQLERRD)
      -> FIELD (SQLERRDF)
    -> FIELD (SQLERRM)
    -> FIELD (SQLERRP)
    -> FIELD (SQLEXT)
    -> VIEW (SQLWARN)
      -> FIELD (SQLWARN0)
      -> FIELD (SQLWARN1)
      -> FIELD (SQLWARN2)
      -> FIELD (SQLWARN3)
      -> FIELD (SQLWARN4)
      -> FIELD (SQLWARN5)
      -> FIELD (SQLWARN6)
      -> FIELD (SQLWARN7)
  -> VIEW (SQL_PARAMETER_LIST)
    -> FIELD (A_PARAMETER_POINTER)
    -> FIELD (CALL_TYPE)
    -> FIELD (CODE_POINTER)
    -> FIELD (PARAMETER_LIST_CONSTANT)
    -> FIELD (PROGRAM_NAME)
    -> FIELD (STATEMENT_NUMBER)
    -> FIELD (STATEMENT_TYPE)
    -> FIELD (TIME_STAMP_ONE)
    -> FIELD (TIME_STAMP_TWO)
    -> FIELD (V_PARAMETER_POINTER)
    -> FIELD (SECTION_FLD)
```

Export the Migration Object

To export the migration object, from the Migration Export window, click **Action > Export**.

Figure 8-19 Migration Export Window

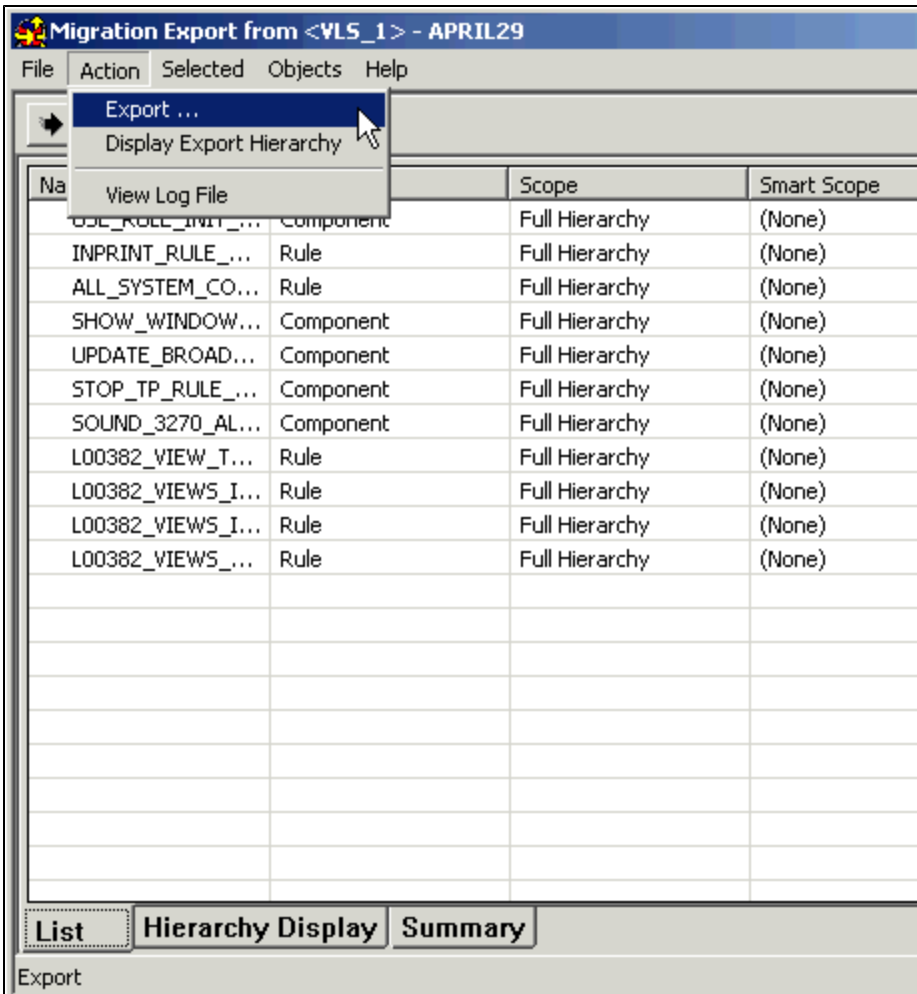


Table 8-8 Migration Action Menu Options

Menu Items	Explanation
Export	Exports the objects you have added as object roots and writes the data to migration files.
Display Export Hierarchy	Displays the objects and relationships in .txt format.
View Log File	Displays information that was recorded during the most recent export. If there are no log files saved on your system, the View Log File option is unavailable.

Table 8-9 Migration Export Window Tabs

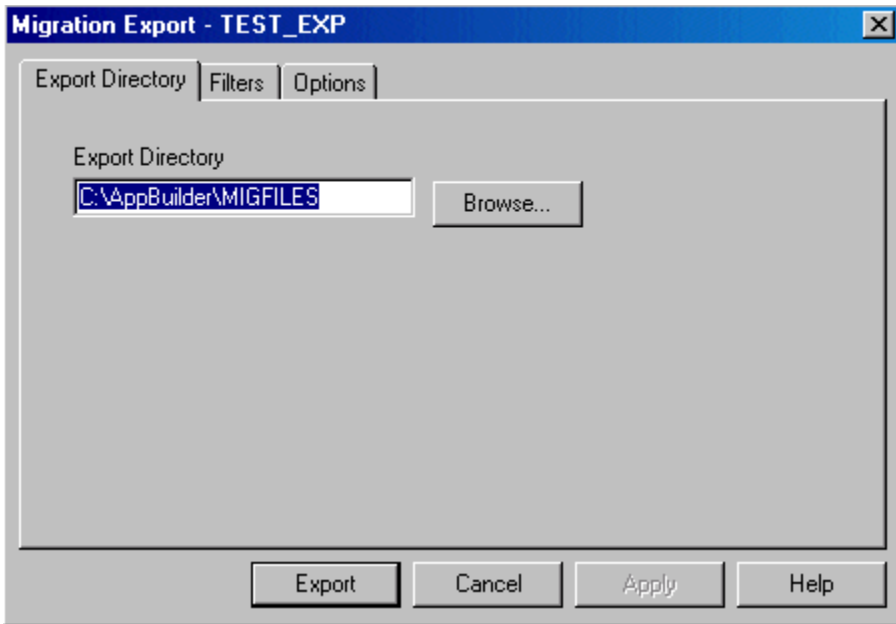
Window Tab	Description
List	Lists the entities you have related to the migration object.
Hierarchy Display	Produces a text file that displays the hierarchy of the objects that will be exported. This information is based on the scope you select. Refer to Set the Migration Object Scope for more information.
Summary	Displays the export log file in text format for the migration object that is displayed in the Migration Export window. You must activate the Export option from the Action menu before the log file is displayed in the Summary tab. This differs from the View Log File option on the Action menu, in that the View Log File option displays the most recent log file that was generated, regardless of the current export. If there are no log files saved on your system, the View Log File option is unavailable.

Setting Export Migration Options

Selecting **Action > Export** opens the Migration Export setup window ([Figure 8-1](#)). The Export window contains three tabs for setting additional migration configurations:

- [Export Directory](#)
- [Filters](#)
- [Options](#)

Figure 8-20 Migration Export Tabs



Export Directory

In the field provided on the **Export Directory** tab, you define the target directory for migration objects. [Figure 8-20](#) shows the Export Directory field where you type the path destination for your migration object files. If the directory you specify does not exist, the system prompts you to create it. There is also a **Browse** button that allows you to locate a directory for your migration export.

Filters

Using the **Filters** tab, you can establish project, date, and time filters for the migration objects you are exporting ([Figure 8-21](#)). The Filters window allows you to set specific filters for objects to be included or excluded in the migration. The time field must be used in conjunction with the date field.

Figure 8-21 Migration Export Filters

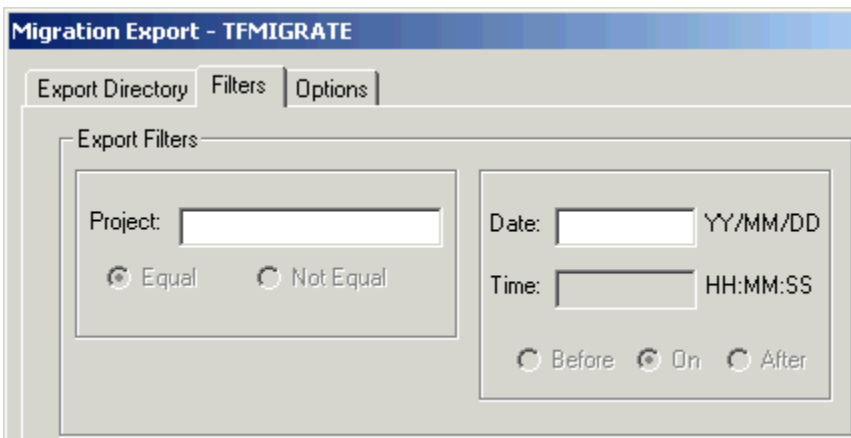


Table 8-10 Migration Export Filters

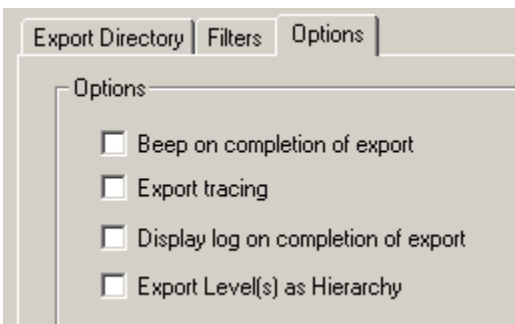
Filter	Description
--------	-------------

Project	Type the name of a valid project in this field. <ul style="list-style-type: none"> • Equal radio button includes all the objects in the specified project, consistent with the scope defined. • Not Equal radio button excludes all the objects in the specified project.
Date and Time	Type the date and time using the formats examples to export data based on date and timestamps associated with the objects. <ul style="list-style-type: none"> • Before radio button specifies to export objects before the defined date • On radio button specifies dates that match the timestamp exactly • After radio button specifies dates after the defined timestamp

Options

The **Options** tab allows you to specify notification, tracing, display log, and hierarchy settings for the migration object. Click on the check box next to the option you want to function during your migration export.

Figure 8-22 Migration Export Options



The following table describes the actions in the **Options** tab of the Migration Export configuration window. These settings are located in the [Freeway] section of the HPS.INI file. For more information about INI settings, refer to the *INI Settings Reference Guide*.

Table 8-11 Migration Export Options

Option	Description
Beep on completion of export	The workstation notifies completion of the export with a beep sound. BEEP= FALSE /TRUE The default for this setting is False . When set to False , there is no beep. This setting is also available from the Migration Import window.
Export tracing	The Tracing option determines if migration information is written to a file. MIGRATION_TRACE= FALSE /TRUE The default for this setting is False . When set to False , the import information is not written to a file. Select this check box to set this option to True. When this option is set to True, AppBuilder writes the migration information to a file called either LOAD.OUT, ANALYZE.OUT, ANAIMP.OUT, IMPORT.OUT or EXPORT.OUT, depending on the relevant operation. If an error occurs in a migration operation, a less detailed version of these .OUT files is created regardless of the tracing option selection. This option is also available from the Migration Import window.
Display log on completion of export	Automatically displays the LOG file after export. EXPORT_DISPLAY_LOG= FALSE /TRUE The default for this setting is False . When set to False , the log file is not automatically displayed. You must select Action > View Log File or select the Summary tab.

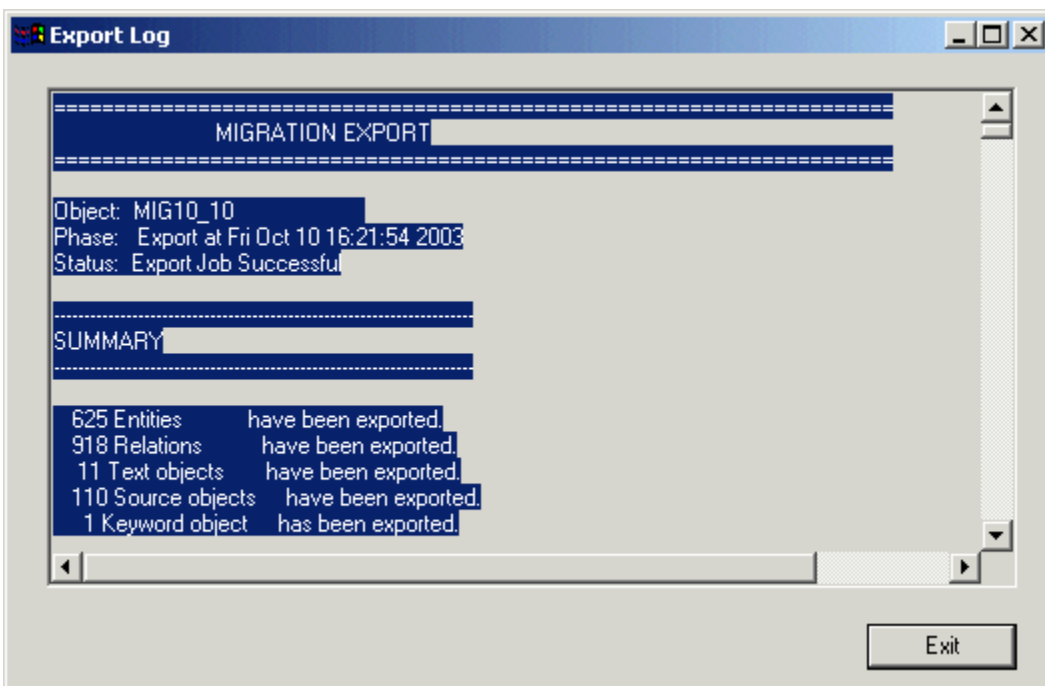
Export Level(s) as Hierarchy	LVLEXP_AS_HIERARCHY= FALSE /TRUE The default for this setting is False . If set to False (Unchecked) — AppBuilder exports all objects and relationships in the level selected as Entity Only. Objects are automatically merged into the target repository (no implied deletes of relationships take place in the target repository). If set to True (Checked) — AppBuilder exports the entire hierarchy. The Export Level(s) as Hierarchy option must be checked for deletions to occur in a migration.
------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Check the Export Log File

To check the log file after an export, from the Migration Export window (Figure 8-6), click the **Summary** tab or click **Action > View Log File**. Verify that the export was successful. The Summary tab is only populated after the current export. Using the menu option, you can view the most recent log file that is stored on your system.

From the Options tab, you can also select **Display log on completion of export** to generate a log and display it after export. The system writes an EXPORT.OUT file with information about whether the migration succeeds or fails and puts the files in the directory specified in the Migration Export Window. It can be opened and viewed using any standard text editor, such as Notepad. The following graphic shows a sample migration log file.


Figure 8-23 Sample Migration Log File



Performing a Migration Import

A migration import consists of the following tasks:

1. Connect to a repository.
You can activate the Migration Import and Export functionality from any tab in the Repository Administration tool.
2. Find the directory where the migration files you want to import are located.

 Copy the migration files locally to improve performance of your import.

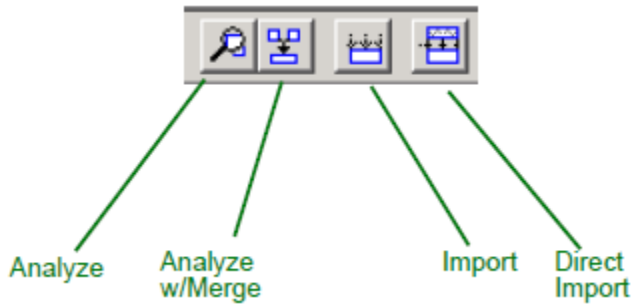
3. Analyze the migration files against the content of the target repository. Refer to [Performing the Migration Analysis](#) for more information.
4. Load — The Load Phase is included automatically when you start analysis. *The target repository is not changed by analysis and load.* Analysis deduces what needs to be created, updated, and deleted by the Import Phase.
5. Execute the import.

Direct Import: This performs the Load Phase, Analysis Phase, and Import Phase without pausing to allow user interaction. The Direct Import is practical when performing a large migration overnight. When set up correctly, Direct Import does not display any messages or dialog boxes until import is complete.

Migration Import Toolbar

The Migration Import window provides a toolbar for easy access to the import options.

Figure 8-24 Migration Import Toolbar

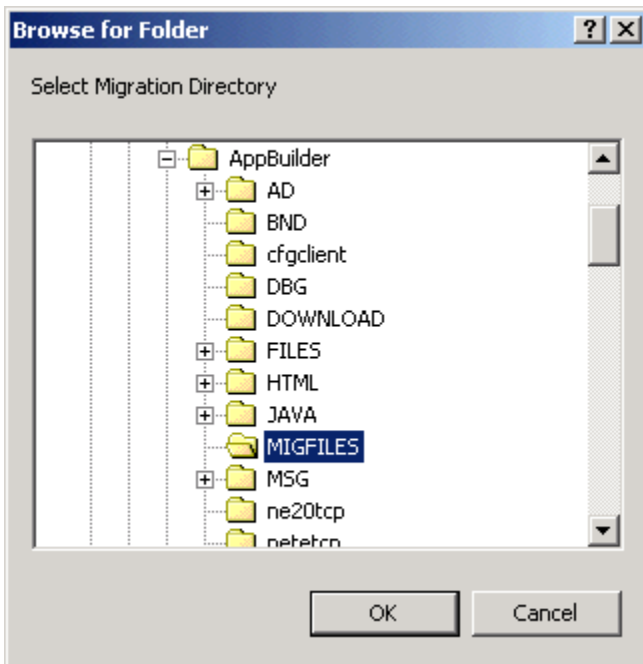


Connecting to a Target Repository

To begin the migration import operation:

1. Select **Tools > Migration Import**.
2. Select **Open** (to select an existing Migration file).
3. The Migration Import window opens. Select **File > Choose Directory** to locate the directory the migration files were exported to during the Export phase.
4. Locate the directory in the Browse for Folder dialog and click **OK**.

Figure 8-25 Browse for Folder Window



5. The Migration Import window opens and the status bar displays the directory path.

Figure 8-26 Directory Location in the Status Bar



Performing the Migration Analysis

During this phase, the content of the migration files is analyzed against the repository. There are two options for analyze: **Analyze** and **Analyze with Merge**. **Analyze** carries the object relationships from the source repository to the target repository. This is referred to as a **destructive** merge because objects and relationships in the target repository may be overwritten or deleted when they don't match what is coming in from the source repository.

An **Analyze with Merge** keeps the relationships in the target repository intact. It merges any new objects coming from the source repository into the existing hierarchical structure of the objects in the target repository. Refer to [Using the Analyze w/Merge](#) for more information about this option.

1. Select **Action > Analyze**, or **Analyze w/Merge**. An analysis is performed on files in the selected migration object in the directory.
2. On completion, the Migration Import analysis is displayed. It contains information about the migration import phase and status, and the results summary for entities and relationships comparisons. The import has not been executed yet.

Figure 8-27 Migration Import Analysis display

Name/Parent	ShortName/Child	Status/A...	Type	MetaType	Errors	Project
HPS_COMM_ERR...	RDNAERR	Update	Rule	Entity	Action succ...	HPS
L00382_VIEW_T...	A8QMML	Compare	Rule	Entity	Action succ...	USB
L00382_VIEW_T...	ZA8QNML	Compare	Window	Entity	Action succ...	USB
W53700709	ZA8Q1ML	Compare	Window Content	Entity	Action succ...	USB
W53700709	ZA8Q2ML	Compare	Panel	Entity	Action succ...	USB
W53700709	ZA8Q3ML	Compare	Help	Entity	Action succ...	USB
L00382_VIEW_T...	ZA8QOML	Compare	View	Entity	Action succ...	USB
L00382_CHECK1	ZA6GLML	Compare	Field	Entity	Action succ...	USB
L00382_CHECK2	ZA8F9ML	Compare	Field	Entity	Action succ...	USB
L00382_CHECK3	ZA8PAML	Compare	Field	Entity	Action succ...	USB
L00382_VIEWS_I...	A8QPML	Compare	Rule	Entity	Action succ...	USB
L00382_VIEWS_I...	ZA8QUML	Compare	View	Entity	Action succ...	USB
L00382_CHAR_30	ZA8OCML	Compare	Field	Entity	Action succ...	USB
L00382_VIEWS_...	A8QVML	Compare	Rule	Entity	Action succ...	USB
L00382_VIEWS_...	ZA8QXML	Compare	View	Entity	Action succ...	USB
L00382_VIEWS_I...	A8QWML	Compare	Rule	Entity	Action succ...	USB
L00382_VIEWS_I...	ZA8QZML	Compare	View	Entity	Action succ...	USB
L00382_VIEWS_...	A8Q4ML	Compare	Rule	Entity	Action succ...	USB
A8QMML	ZA8QNML	Compare	Rule converses Window	Relation	Action succ...	USB
ZA8ONML	ZA8O1ML	Compare	Window has Window Cont...	Relation	Action succ...	USB

Import the files in the Migration Directory OBJ 1665 ENT 625 REL 918



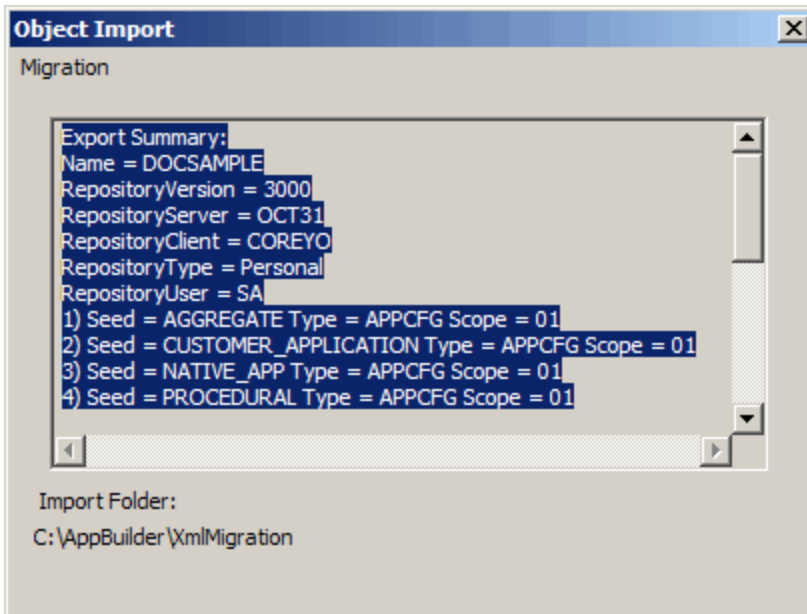
An object will have the status of Update when it has been modified, when the name of the Project with which it is associated has been changed, or if the name of the user who owns it has been changed.

Importing XML Files

You can use the Migration Import to import XML files containing objects and relationships into your repository of choice. To do so,

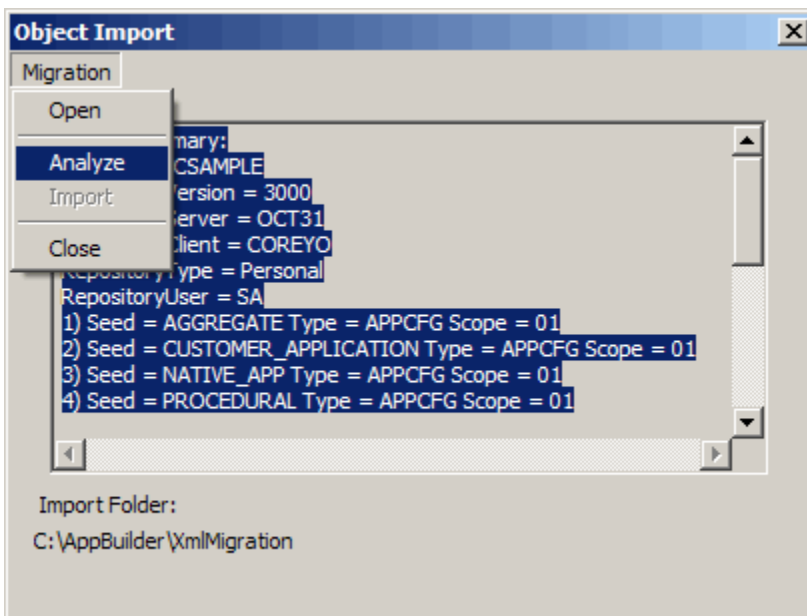
1. In the Repository Administration tool, select **Tools > Migration Import**.
2. Select **XML** (to specify the XML migration folder). The Object Import dialog appears.

Figure 8-28 Object Import dialog



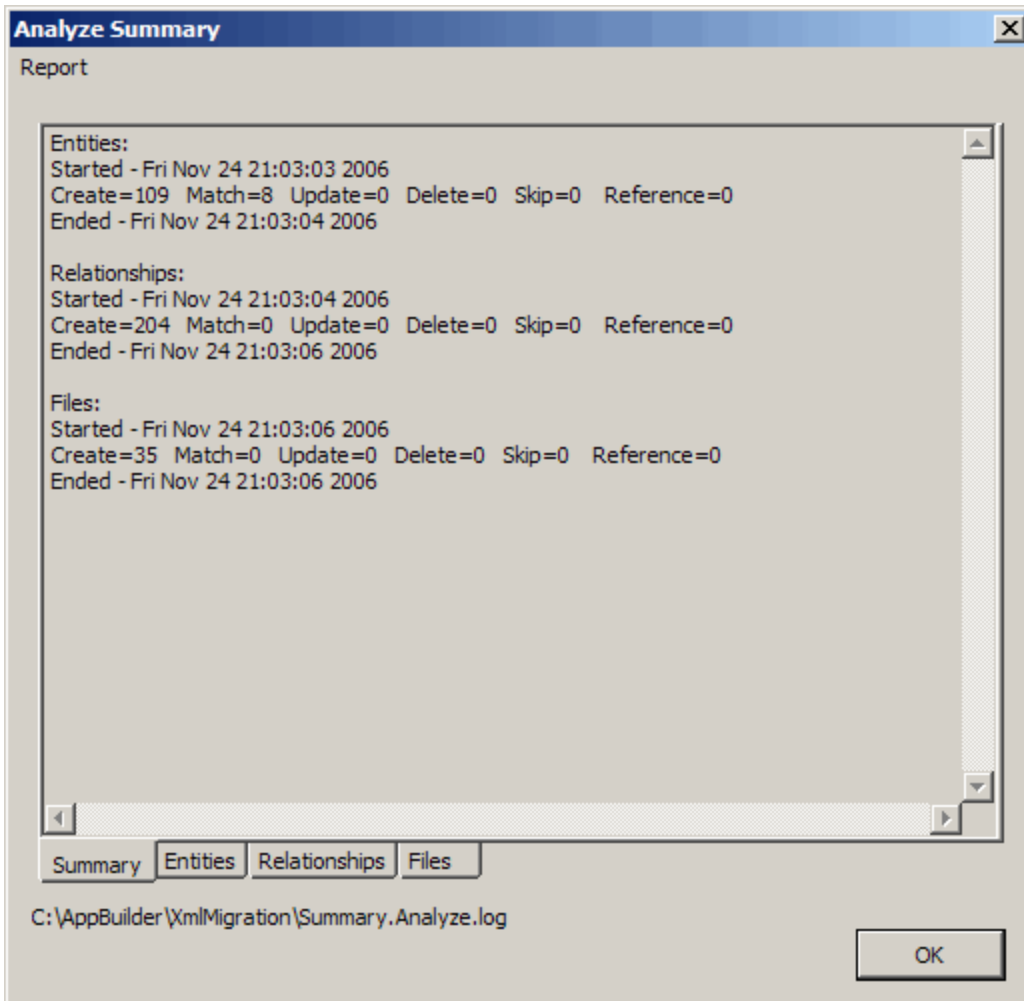
3. The Object Import dialog lists the objects found in the XML files. From the Migration menu, select **Analyze**.

Figure 8-29 Migration Menu of the Object Import dialog



4. The files are analyzed and the results are provided in the Analyze Summary dialog.

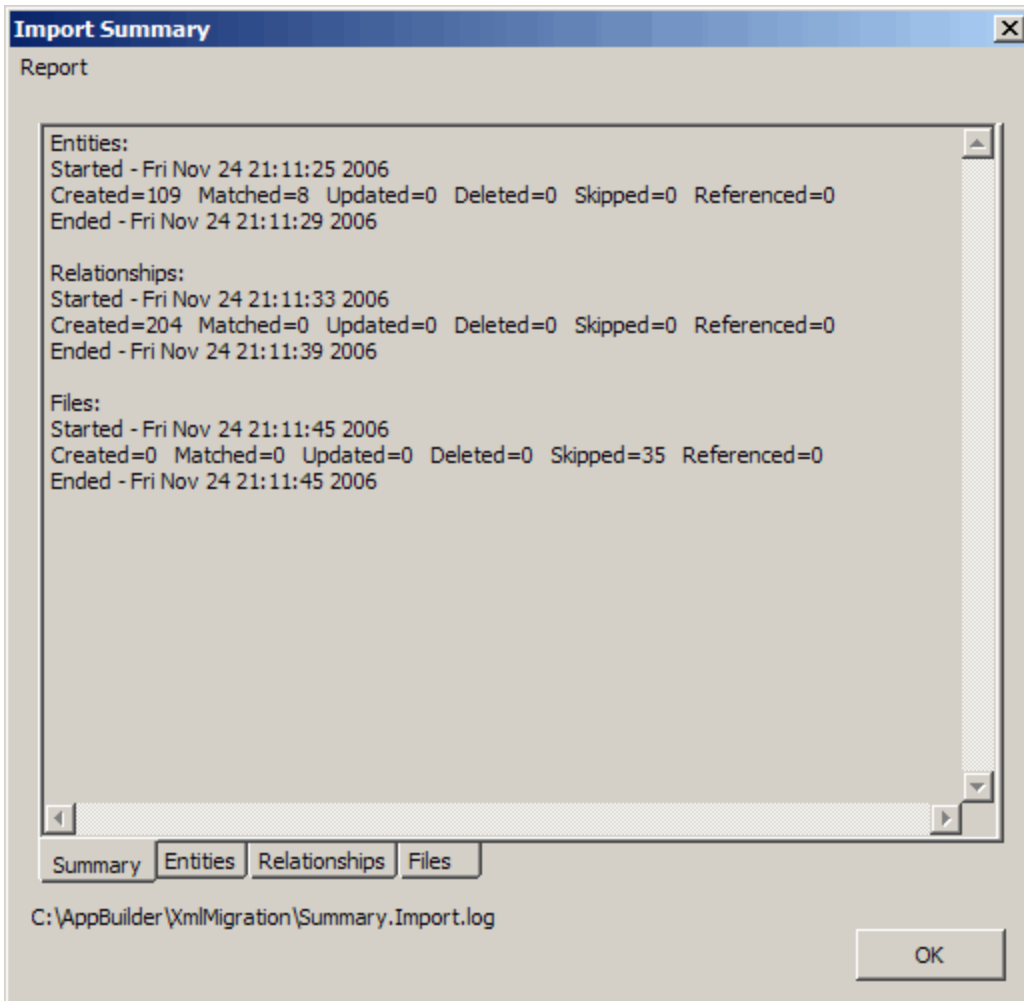
Figure 8-30 Analyze Summary dialog



There are multiple tabs providing a summary as well as data specific to entities, relationships, and files.

5. Click **OK** . The Analyze Summary dialog closes and you are once again at the Object Import dialog.
6. Select **Import** . You may be notified that the session is using a Unit of Work. If it is OK to disable the Unit of Work for this session, click **Yes** .
7. Once the Import has been completed, a summary dialog appears providing a report on what entities and relationships and files were imported.

Figure 8-31 Import Summary Report



8. Click **OK** . You are returned to the Object Import dialog. (See [Figure 8-29](#)).
9. Select **Migration > Close**.

Filtering the Migration Import

The Workgroup Migration Import window provides filters so that you can more easily view your analysis and customize the actual migration import. To filter the import analysis, do the following:

1. From the Migration Import window, click **View > Filter Result**.
The Filter window displays with the General Tab displayed.

Figure 8-32 General Tab Filter Options

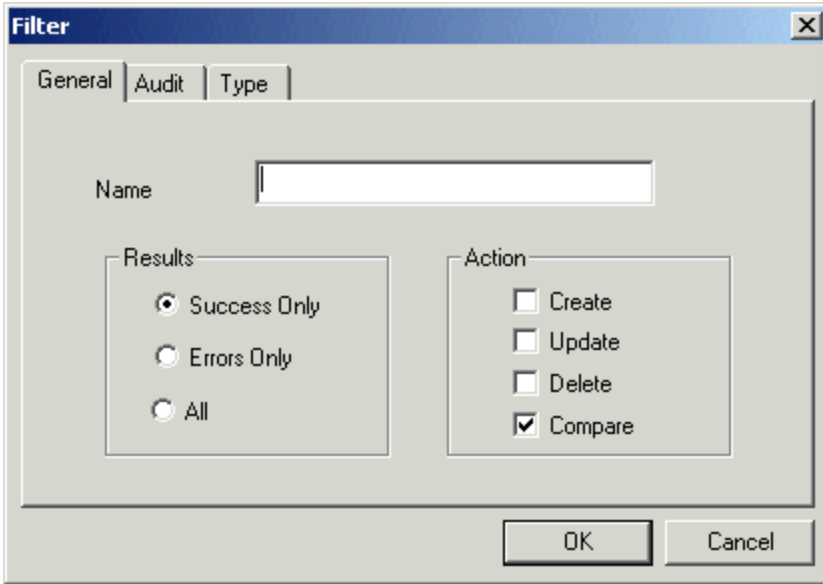


Table 8-12 General Tab Filter Options

Option	Description
Name	Display only entities with this full or partial name.
Results	
Success Only	Display entities with a successful status.
Errors Only	Display entities with an error status.
All	Display all entities regardless of status.
Action	
Compare	An identical object currently exists in the target repository.
Create	The object does <i>not</i> currently exist in the target repository.
Update	The object currently exists in the target repository but is <i>not</i> identical to the object about to be imported. Note: An object will have the status of Update when it has been modified, when the name of the Project with which it is associated has been changed, or if the name of the user who owns it has been changed.
Delete	The object exists in the target repository, but <i>not</i> in the source. Performing the physical import causes the object in the target repository to be deleted.

Import Status Toggle

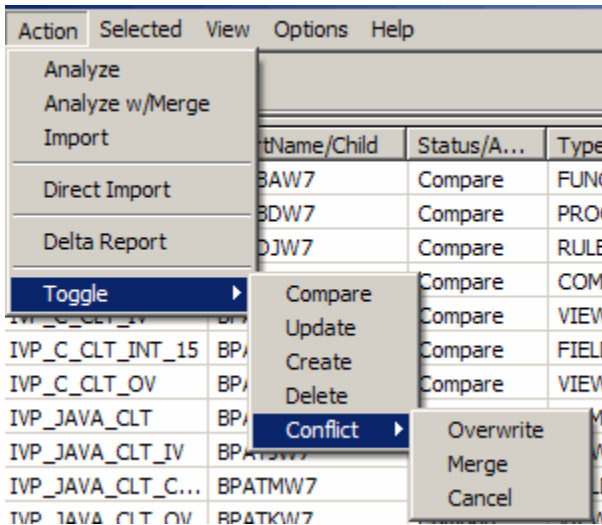
The Toggle commands control the object import status for the migration object during migration. The import status determines whether the import object should be compared, created, updated, or deleted in the target repository. (See the column in [Figure 8-27](#) labeled Status/Action.) It also specifies whether to flag changes between the source and target objects.

The import status can be toggled between *on* and *off*. For example, if the status of an object is **Create**, the physical import will *create* it in the target repository. Toggle the import status to **Create Off** and the physical import will *not create* it in the target repository. The import status **Compare** can be toggled to **Compare Off**; and so on.

You can control the status of imported objects, either individually or collectively, by using this Toggle command. You can, for example, set all relationships and associated files, such as source, text, and keywords, of an object to toggle with the object. When an object is set to **Create off**, all relationships for that object will be set to **Create off**. From the Migration Import window, click **Options > Settings > Auto Toggle Relations**. This is also true for all files associated with an object. For example, Rule objects and Window objects can have associated File objects. When an object is set to **Create off**, all files associated with that object will be set to **Create off**. From the Migration Import window, click **Settings > Auto Toggle Files**. Refer to [Migration Import INI Settings](#) for more information on the auto toggle settings.

To toggle only specific selected objects in the import list, right-click on the entity and select **Toggle Action**. To toggle all entities of a specific type, click **Action > Toggle** and select the status you want to toggle. All entities with that status will be toggled.

Figure 8-33 Toggle Import Status



The following section explains the import status categories in more detail:

- **Compare** - An identical object currently exists in the target repository. When a physical import is performed, the object being imported and the object in the target repository are compared. If the object in the target repository is modified *after* the analysis operation but *before* the import, the import operation reports that the target object has been modified since the analyze was performed. However, the object is not updated by the import operation.
- **Update** - The object currently exists in the target repository, but is not identical to the object to be imported. The object being imported overwrites the object in the target repository when the import is implemented.
- **Create** - The object does not currently exist in the target repository. When the physical import is performed, if it still does not exist in the target directory, it will be created.
- **Delete** - The object exists in the target repository, but not in the source. The physical import causes the object in the target repository to be deleted. This status applies only to *dependent* objects, such as text, source, keywords, and relationships. Primary objects cannot be deleted using the migration facility.
- **Conflict** - The new object being imported into the target repository will conflict with an object that already exists in the target repository. The options (Overwrite, Merge, or Cancel) are explained in [Table 8-2](#).

Figure 8-34 Right-click menu operation for Toggle

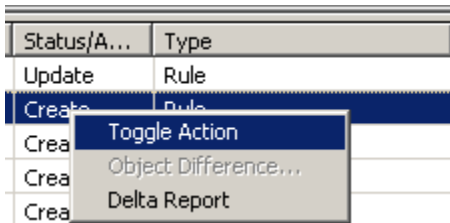
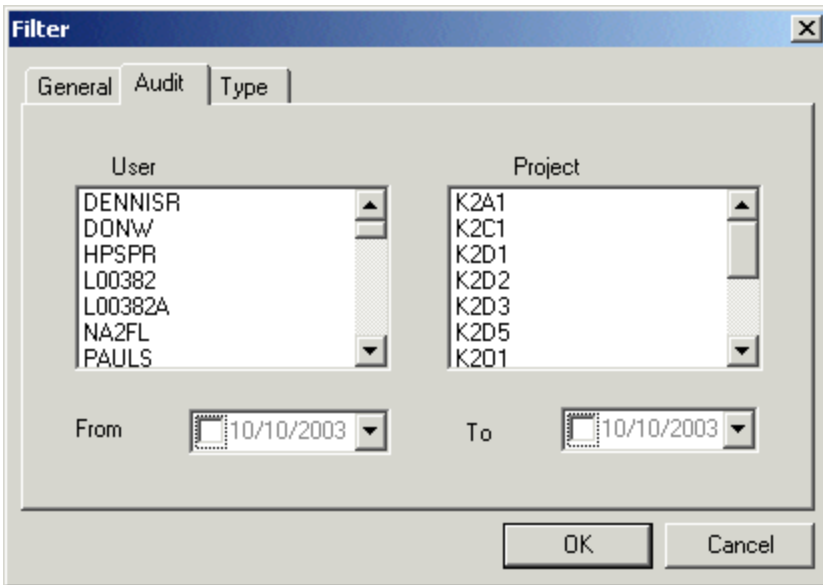
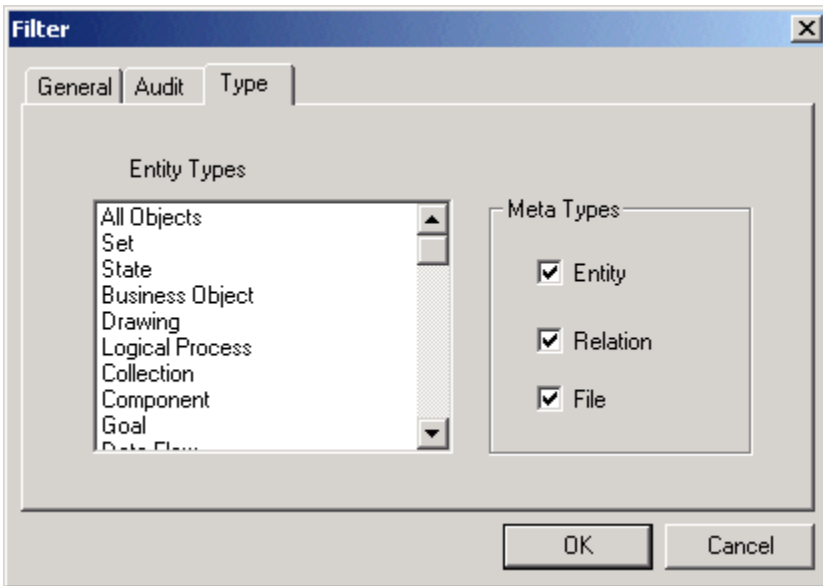


Figure 8-35 Audit Tab Filter Options



You can include objects for a specific user, project, or date and time frame.

Figure 8-36 Type Tab Filter Options



You can include objects by entity type.

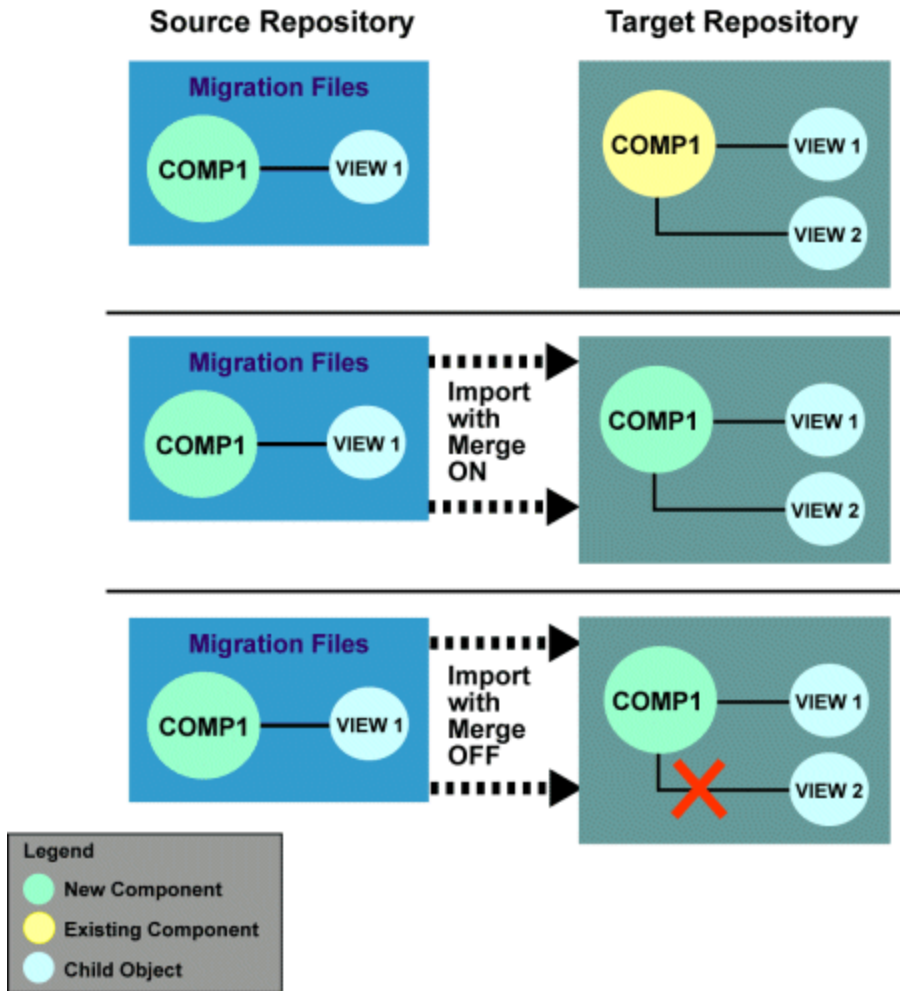
Using the Analyze w/Merge

Analyze w/Merge controls how the system affects specific objects during an import. Analyze with Merge keeps the parent/child relationship intact in the target repository.

For example, a set of migration files (source) and a repository (target) contain the Object A. Object A has a child object in the target repository, but not in the source repository. The **Merge** option determines what happens to the relationship between Object A and the child object when the source is imported to the target. If the **Merge** option is used, the relationship between the object and the child is unchanged. If the **Merge** option is *not* used, the *relationship* is deleted; however, the child object remains unchanged in the target repository.

The **Merge** option has the same effect when an object in the target repository has an associated file (text, keywords, or source). If the file object is present in the target but is not present in the source, the *Merge* option determines whether the file is moved or deleted when an import is performed.

Figure 8-37 Merge Option Example

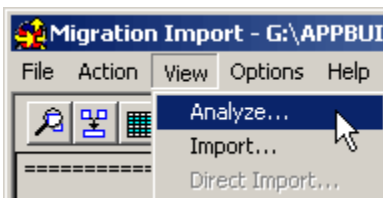


Viewing an Existing Analysis or Import

When you perform an import analysis or an import, AppBuilder creates two report files in the migration directory: *Analysis.rpt* and *Import.rpt*. It is possible to view a report from the last analysis or import that you have performed. In order for the view options to be available, you must be in a migration directory; however, when you open the Import window, AppBuilder automatically puts you in the directory of your last import. The Summary tab and Analysis tab are blank when you first open the Migration Import window. If you choose to view a report from an earlier analysis or import, you must do so from the View menu:

1. Select **View > Analyze** to examine the results of an existing analysis report.
2. Select **View > Import** to examine the results of an existing import report.

Figure 8-38 Viewing the Import Analysis



After you have performed an analysis or import from the Action menu, the results of the action are displayed in the window tabs. After you perform an Analysis, the Analyze tab is populated. After you perform an Import, the Summary tab is populated.

Generating a Delta Report Before an Import

A Delta Report compares the objects associated with the migration object with the objects that exist in the repository and shows you specifically how each object has been changed. The report is generated for entities and relationships that have changed. The report shows the property differences followed by the file differences (Source, Text, Keywords). The Delta Report for migration is an Excel file called *migdelta.dif*. It is saved to the migration directory.

Take the following steps to generate a Delta Report:

1. Run Analyze. Click **Action > Analyze** or **Action > Analyze w/Merge**.
2. Click **Action > Delta Report**.

The Delta Report is displayed from the Delta Report tab. To print this report from the Delta Report tab, click **File > Print**.


 If there are no changes to objects in the repository, the Delta Report is empty.

Figure 8-39 Example Delta Report

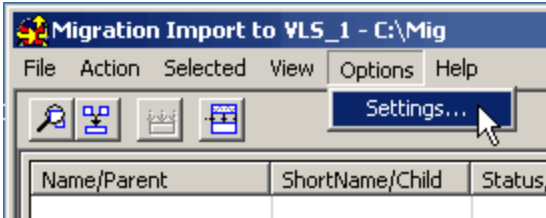
MIGRATION DELTA REPORT		
Produced on Tuesday, June 08, 2004 at 12:35:42		
Legend :		
+	:Object Property/Text for Migration	
-	:Object Property/Text for Repository	
File Name	:C:\Mig_Imports\DB2Stress\migdelta.dif	

Import will UPDATE Entity <Rule>	HPS_COMM_ERROR_RULE	
Property Name	Migration(+)	Repository(-)
-----	-----	-----
LocalMaintenanceDate	04/04/21	03/01/22
LocalMaintenanceTime	14:13:38	16:29:38
FWY_User	DB2ADMIN	USERID
-----	-----	-----
Import will CREATE Entity <Rule>	L00382_VIEW_TYPES_DIS	
Property Name	Migration(+)	Repository(-)
-----	-----	-----
Name	L00382_VIEW_TYPES_DIS	
FWY_Project	USB	
FWY_Owner	L00382A	
ShortName	A8QMML	
FWY_User	DB2ADMIN	
LocalMaintenanceTime	15:27:18	
Sys_Source	N/A	
LocalMaintenanceDate	03/10/10	
ChangeNumber	2	
OwnerId	L00382A	
Rule_Str_Id	SYNC	
ProjLockType	N	
Project	USB	
Rule_Imp_Name	A8QMML	
RemoteMaintenanceTime	14:34	
RemoteMaintainedBy	L00382A	
Isolation_Mode	0	
RemoteMaintenanceDate	98/09/28	
Exec_Environ	PC	
RemoteCreatedBy	L00382A	
RemoteCreationTime	14:34	

Migration Import INI Settings

Before you perform a migration import, you can adjust the following INI settings in the HPS.INI file from the Migration Import window. For more information about INI settings, refer to the *INI Settings Reference Guide*.

Figure 8-40 Migration Import Options Menu INI Settings



1. Click **Options > Settings**.

The Settings window displays. A check mark indicates that the setting is set to True. No check mark indicates that the setting is set to False.

Figure 8-41 Settings from the Migration Import window — Freeway tab

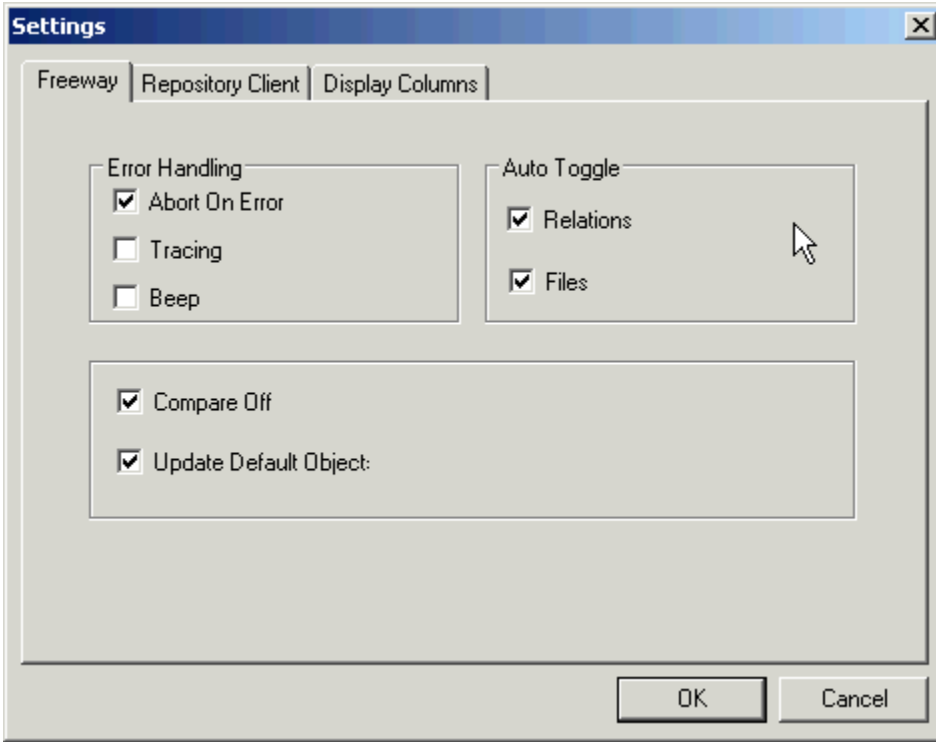


Table 8-13 Settings from the Migration Import window — Freeway tab

INI Setting Option	Description
Abort On Error	<p>This setting is used to allow your import to continue processing even if an error occurs.</p> <p>[FREEWAY] MIG_ABORT_ON_ERROR= TRUE / FALSE</p> <p>The default for this setting is True . When set to True , the import stops on all errors, and no additional data is imported past that point in the migration. When this setting is set to False, the import continues even if an error occurs; however, the result of the migration is still "failed" because all of the data was not imported.</p>
Tracing	<p>The Tracing option determines if migration information is written to a file.</p> <p>MIGRATION_TRACE= FALSE /TRUE</p> <p>The default for this setting is False . When set to False , the import information is not written to a file. Select this check box to set this option to True. When this option is set to True, AppBuilder writes the migration information to a file called either LOAD.OUT, ANALYZE.OUT, ANAIMP.OUT, IMPORT.OUT or EXPORT.OUT, depending on the relevant operation. If an error occurs in a migration operation, a less detailed version of these .OUT files is created regardless of the tracing option selection. This option is also available from the Migration Export window.</p>

Beep	<p>Workstation notifies completion of the export with a beep sound</p> <p>BEEP= FALSE /TRUE The default for this setting is False . When set to False , there is no beep. This setting is also available from the Migration Export window.</p>
Compare Off	<p>The default action for Compare is that it is on. If an identical object currently exists in the target repository. When you import, the object being imported and the object in the target repository are compared. If the object in the target repository is modified <i>after</i> the analysis operation but <i>before</i> the import, the import operation reports that the target object has been modified since the analyze was performed. However, the object is not updated by the import operation.</p> <p>If you do not want the Compare action, check the Compare Off setting. [FREEWAY] COMPARE_OFF= TRUE</p> <p>The COMPARE_OFF ini setting only appears in the hps.ini file when it is checked and set to True.</p>
Update Default Objects	<p>This setting is used to control whether or not default system objects are included in an import.</p> <p>[FREEWAY] IMPORT_DEFAULT_OBJS= FALSE /TRUE</p> <p>The default for this setting is False . When set to False , the system automatically filters out the <i>updated</i> default system objects that belong to the SEER1 or SEER2 project from the import. If there are user created objects in the SEER1 or SEER2 project, they are also filtered out of the import. The analyze step does not report these objects. Check this option to set it to True. When set to True, the <i>updated</i> objects that belong to the SEER1 and SEER2 project are imported like any other object. Note: This setting applies to <i>updated</i> objects only. If the default objects are being <i>created</i> through a migration import, they are included in the migration and this setting does not apply.</p>
Relations	<p>In a migration analyze, you have the option of turning on and off the create feature for a specific entity before it is created in the repository. This setting automatically toggles create off for relations of an entity that was toggled create off.</p> <p>[FREEWAY] AUTO_TOGGLE_RELATIONS= FALSE /TRUE</p> <p>The default for this setting is True . When this setting is set to False , you must manually toggle off the create setting for each entity relationship when an entity's create setting is toggled off. When set to True, the create setting for an entity's relationship will match that of the entity.</p>
Files	<p>In a migration analyze, you have the option of turning on and off the create feature for a specific entity before it is created in the repository. This setting automatically toggles create off for files of an entity or relationship that was toggled create off. Rule and Window objects are examples of entities that have associated files.</p> <p>[FREEWAY] AUTO_TOGGLE_FILES= FALSE /TRUE</p> <p>The default for this setting is True . When this setting is set to False , you must manually toggle off the create setting for each entity file and relationship when an entities create setting is toggled off. When set to True, the create setting for an entity's files and relationship will match that of the entity.</p>

Figure 8-42 Settings from the Migration Import window — Repository Client tab

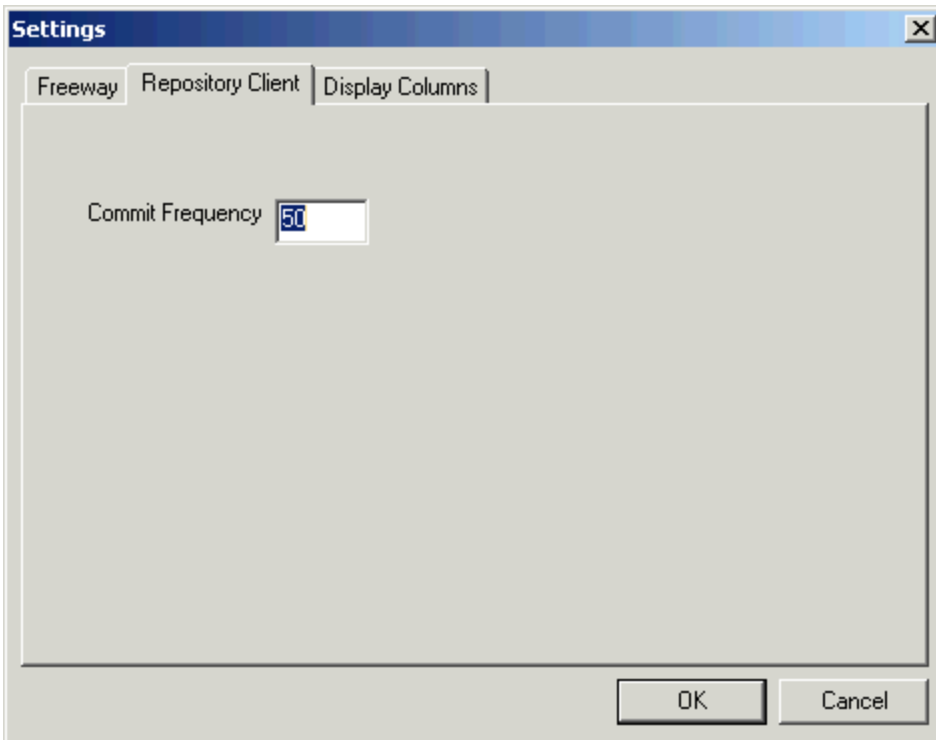
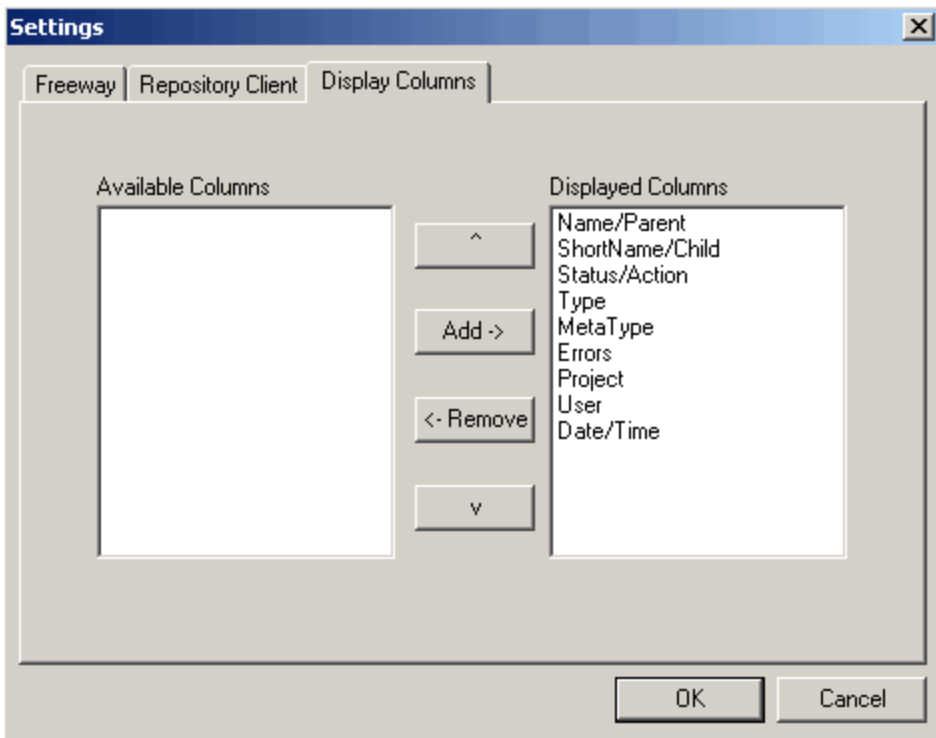


Table 8-14 Settings from the Migration Import window — Repository Client tab

Setting Option	Description
Commit Frequency	The value in the Commit Frequency field determines the number of objects that are migrated before an automatic commit. If you set this value to 0, you must perform a manual commit after the migration has completed. This setting is in the hps.ini file. [REPOSITORY CLIENT] RECYCLE_DATABASE_COUNT= <integer value>

Figure 8-43 Settings from the Migration Import window — Columns tab



From this tab, it is possible to customize which columns display in the Import Analysis window. To move a column name from one pane to the other, highlight the name and select the left and right arrow buttons. To order the columns within the Selected Columns pane, highlight a name and click the up and down arrow buttons as appropriate.

Table 8-15 Settings from the Migration Import window — Display Columns tab

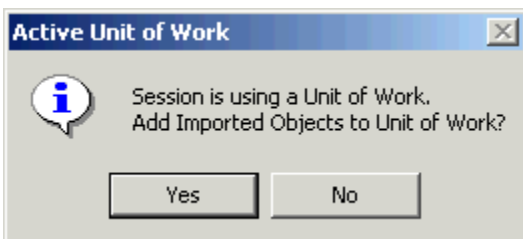
Settings Option	Description
All Columns	This is a list of possible columns you can display in your Import Analysis window. Move each column name that you want to display from the All Columns pane to the Selected Columns pane.
Selected Columns	These are the columns that you have chosen display in the Import Analysis window. You can order the columns in this pane by highlighting a name and clicking the up and down arrow buttons. The top most item is the leftmost column in the Import Analysis window.

Migration Import After Analysis

When the analysis of the migration object has been completed and modifications have been made, you are ready to perform the migration import: From the Migration Import window, select **Action > Import**. A window appears asking if you want to add imported items to your Unit of Work. Your Unit of Work is a collection of objects that have changed or been created or deleted. Unit of Work is an easy way for you to track the changes you have made:

- Click **Yes** to have the imported objects placed in your Unit of Work; then, you can easily migrate those objects when migrating your Unit of Work.
- Click **No** when you want the objects in your repository but not associated with your Unit of Work; then, when you modify these imported objects, they are added individually to your Unit of Work.

Figure 8-44 Active Unit of Work Window



The progress of the migration is displayed in the status bar during the import. The status bar message confirms that the import succeeded or

failed. To check the results, select the Import tab or the Summary tab from the Migration Import window. You can also view results of a previous import:

1. Open the directory for the import. Click **File > Choose Directory**, and select the import folder.
2. Select **View > Import** in the Migration Import window.
The migration import report for that migration is displayed.

Performing a Direct Import

In a Direct Import, all the objects specified in the migration files are imported to the target repository without an intervening analysis operation. The analyze and import processes are coupled into one action. A direct import *disables* the options to analyze, view, or select objects before doing the physical import to the target repository. This lower level of processing results in a direct import being faster than a selective import.

Perform a direct import *only* when the import status of the data does not need to be changed. For example, Group 1 and Group 2 selectively migrate their data to a central repository where integration of the data is performed. A direct import is very useful in this scenario, because Group 1 and Group 2 are complete and approved, no revisions need to be made, and mirror images of the latest object hierarchies can be directly imported.

Direct Import has a menu item to automatically toggle objects between **Compare** and **Compare Off** before the import takes place. To use this option, in the Migration Import window, select **Settings > Compare Off**.

To perform a direct import:

1. Select **Action > Direct Import**.
2. View the results of the import from the Import tab and the Summary tab on the Migration Import window.

You can also view results of a previous import:

1. Open the directory for the import. Click **File > Choose Directory**, and select the import folder.
2. Select **View > Direct Import** in the Migration Import window.

The migration import report for that migration is displayed.

Using Migration Results

Each phase of a migration produces a different set of output files. These files have either a . LOG or . OUT file extension.

- LOG files contain details of a successful migration. This information includes the type of entity, the long name of the entity, the scope of the migration child object to which the entity belongs, and an object status line.
- OUT files contain troubleshooting information to use if the migration was unsuccessful.

If any phase of the migration fails, *only* the OUT files are produced. LOG files may still be created, but they will be empty. If the migration phase is unsuccessful, the appropriate OUT file is displayed.

To display the LOG or OUT file after a successful execution, navigate to the appropriate migration folder. LOG files are located in the directory the migration is targeted to. The generated .OUT file is placed in the same directory the migration files reside in, as defined in the Migration Export window ([Figure 8-1](#)).

The following table lists the output files produced during each migration operation.

Table 8-16 Migration Output Files

Operation	Output Files
Export	EXPORT.LOG, EXPORT.OUT
Load	LOAD.LOG, LOAD.OUT
Direct Import	ANAIMP.LOG, ANAIMP.OUT
Analysis	ANALYZE.LOG, ANALYZE.OUT
Import	IMPORT.LOG, IMPORT.OUT

Deleting a Migration Object

Removing a Migration object from a repository is similar to removing any object from a repository:

1. Query the contents of the repository.
2. Load the objects of your choice.
3. Right-click on the objects to delete.
4. Select **Delete From Repository**.

For more specific instructions:

1. Right-click the repository and select **Query Content**. Or, Click **Tools > Query Content**. The Query Objects window displays.
2. Click **Query** and select one or more objects and click **Load**. To load one object, you can double-click on that object. When loading multiple objects, you must click **Load**. The objects are loaded into the Status Report pane.
3. Right-click on each object to remove and select **Delete From Repository**. The migration object is not deleted from the repository until you commit your actions.
4. From the Repository Administration tool, click **Repository > Commit**.

The **Purge From Repository** option deletes the entity locally without altering the UOW; therefore, the entity is not deleted from the server upon import.

Migrating between PC and Enterprise repositories

Migrating Between PC and Enterprise Repositories

Typically, the migration process is a manual process managed by an administrator who collects and moves the data to the target repository. The purpose of the Migration Server is to automate the migration process from a single user Personal Repository to the Enterprise Repository, eliminating the need for an administrator to manually move the data. The Migration Server automates the movement of data between a Personal Repository and the Enterprise Repository over TCP/IP.

The following topics are discussed in this section:

- [How Automated Migrations Work](#)
- [Managing the Migration Server](#)
- [Setting the Codepage for SBDATACONN](#)
- [Configuring Mainframe Repository Properties](#)
- [Specifying Upload, Download and Working Directories](#)
- [Downloading Objects](#)
- [Uploading Objects](#)
- [Performing UOW Migrations](#)
- [Setting Server Security Requirements](#)
- [Understanding Performance Implications](#)
- [Understanding Migration Restrictions](#)
- [Renaming PC Migration Files](#)
- [Understanding Mainframe Migration and Repository Security](#)

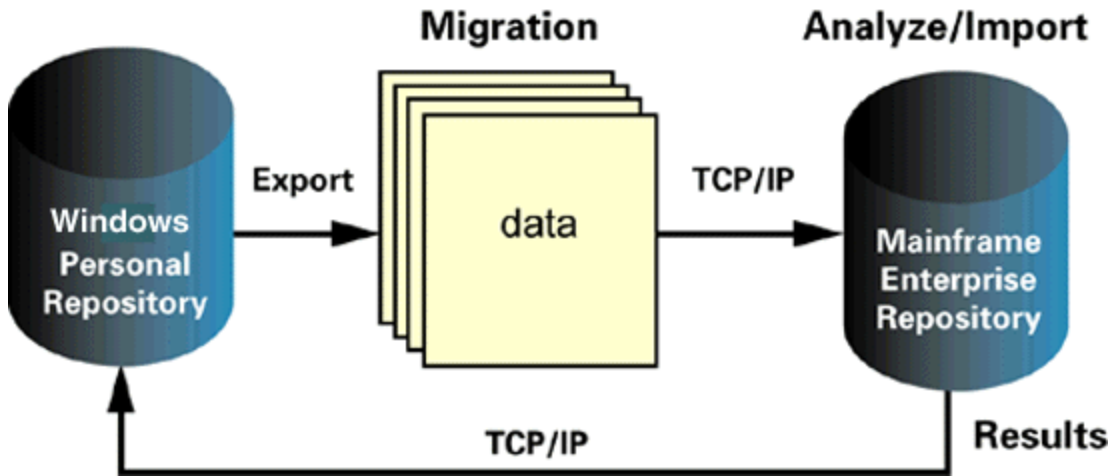
How Automated Migrations Work

How Automated Migrations Work

Automation of the migration process requires use of a TCP/IP communication protocol. The migration files are automatically uploaded, and the Analyze and Import functions are executed. Results of the import are then automatically transferred to the workstation for review.

Downloads are initiated with a Query to the target Enterprise Repository. Using the Query, the objects and scopes are selected for download. Upon completion, the download process is initiated. The Enterprise Repository exports the data and makes the files available for the client to download. The Personal Repository then downloads the migration files, imports them, and displays the results to the user. The Personal Repository downloads the migration files into a local download directory using FTP.

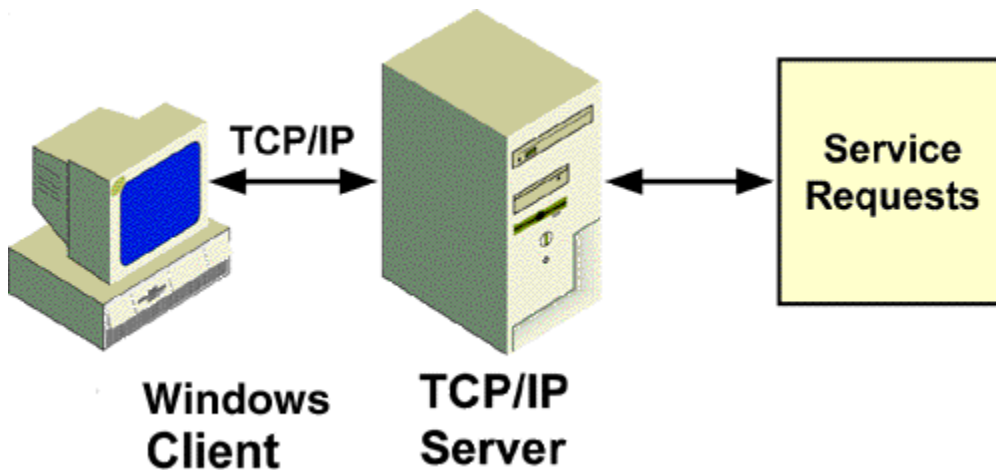
Figure 9-1 Automation of a Unit of Work Migration



The Migration Server services Windows Personal Repository migration requests. The server initiates the request to start an Analyze and Import, start an Export, or Query a repository for certain types of objects based on the long name of the object.

The Migration Server listens on a specific pre-configured port for the Personal Repository service requests. The request might contain the length of data, the actual service request, and any needed parameters that the service request needs for fulfillment.

Figure 9-2 Migration Server Overview



To fully utilize the OS/390 capabilities and make use of existing repository services, most of the communications between the client and server spawn batch jobs for processing. This enables the mainframe to control the resource load.

Managing the Migration Server

Managing the Migration Server

The Analyze/Import service (upload) spawns a batch method handling job. The purpose of this job stream is to tailor the migration request based on the target repository. A single method handling job stream needs to exist for each repository that can be connected to the TCP/IP server. This job stream is made available to the server by the CRSRVJCL method.

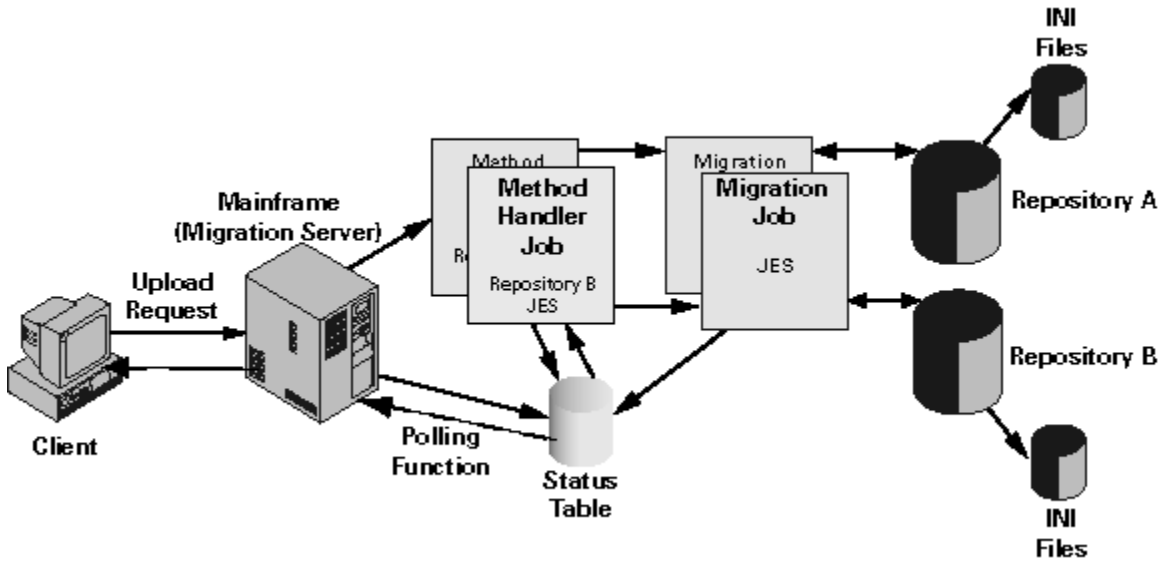
The create server JCL (CRSRVJCL) is an administration method run under the ADM model on the Enterprise Repository. CRSRVJCL creates the method handling job stream based on the .INI settings for a given repository. Without this, the server must dynamically tailor the job streams each time a sync repository service request is made. The CRSRVJCL method manages this tailoring and supplies the server with a ready-made method handler. CRSRVJCL should only be executed once per repository unless INI or data set changes create the need for another execution. The following topics comprise this section:

- [Method Handler Functionality](#)
- [TCP/IP Services](#)

Method Handler Functionality

The Method Handler tailors the migration jobs. It allocates the settings to call the necessary executable and .INI files used to tailor the migration job. Each repository has its own .INI settings that specify the handling of migration settings, what DB2 plans to execute with, and other process commands. The Method Handler uses the proper settings for each repository to ensure proper execution. When the Method Handler submits the migration Analyze/Import or Export job into the internal reader, the standard migration process executes. [Figure 9-3](#) illustrates the Method Handler flow.

Figure 9-3 Method Handler Batch Job Process



TCP/IP Services

The TCP/IP server provides the following services:

Table 9-1 TCP/IP Services

Service	Description
Analyze/Import (Upload)	<ol style="list-style-type: none"> 1. The first step of the Analyze/Import process is an Export process on the Windows Personal Repository. Once completed, the migration files are transferred to the mainframe through FTP. 2. The system then performs an Analyze followed by an Import. The results are written to the log file. 3. The client workstation retrieves the log files from a predefined location and displays the status of the process back to the user for review. 4. Upload consists of a series of migration files, followed by a request to import those objects into the target repository. This service triggers the submission of a batch analyze and import job.
Export (Download)	<p>The Export process requires a migration object and settings to perform the request.</p> <ol style="list-style-type: none"> 1. Specify a series of objects on the workstation to download by querying and selecting objects and scopes. 2. The workstation polls to check on the status of the Export process. When a complete status is determined by the workstation, the workstation initiates an FTP of the migration files and begins an import into the Windows Personal Repository. 3. The Export service creates a migration object with settings as specified by the Windows Personal Repository requests. Once the server creates the objects on the Enterprise Repository, the Export process is initiated. 4. The Export writes out the migration files to complete the download.
Query Host Repository	<p>Use the Query Host Repository function to query a repository by long name.</p> <ol style="list-style-type: none"> 1. Specify a starting long name value and type (for example, Rule). 2. The long name values are passed back to the requesting client forty rows at a time until complete. 3. The repository query facility uses the DB2 CAF to make all database requests. 4. The DBRM for the repository query function must be bound into the repositories DB2 plan via the CRSRVJCL method.
Status Query	<p>The Status Query service is a dedicated service for reporting the status of any given Analyze/Import or Export process. This service reports the current status of any given task.</p>

Shutdown	The Shutdown service is used to bring down the TCP/IP migration server in an orderly manner, closing the socket, DB2 connection, etc. This service is <i>only</i> accessed by the mainframe TCP/IP Client.
FTP Address	This service returns the FTP address location to which the migration files are to be transferred. This location is also used to access the result files generated by the migration process.

Setting the Codepage for SBDATACONN

Setting the Codepage for SBDATACONN

The International Components for Unicode (ICU version 2.0) comes with AppBuilder and is used for the Codepage information for the Personal Repository.

For more information, visit: <http://oss.software.ibm.com/icu>.

Codepage conversion takes place while messaging using a Winsock connection between the Personal Repository and the mainframe migration server. For the codepage information to be transmitted during FTP sessions, a manual entry in the [PERSONAL_REPOSITORY] section of the HPS.INI file must be applied. The setting reads as follows:

```
[PERSONAL_REPOSITORY]
SEND_SBDATACONN_VALUE=TRUE
```

Setting the SEND_SBDATACONN_VALUE to **TRUE** presets your FTP session with the mainframe to your specific codepage settings and ignores the mainframe default. It sends the codepage value to the FTP server for each FTP session, asking the server not to act upon a static setting, but rather use the value supplied with SBDATACONN.

For clients using the same codepages on mainframe and client machines, it is not necessary to use the SBDATACONN setting because the mainframe and client are always set to the appropriate codepage.

To set the codepage, take the following steps:

1. Set the **SEND_SBDATACONN_VALUE** to **TRUE** (or **YES**).
Tells the Personal Repository to send the FTP **quote site** command to the mainframe migration server during an FTP session. This allows the client Personal Repository to dictate codepage conversions for the mainframe migration server. If this is not set, the default setting is used.
2. Set the **USER_REMOVEABLE_UOW** to **TRUE**.
This setting in the [PERSONAL_REPOSITORY] section of the HPS.INI file enables the menu item to remove selected items in the Upload window. If this setting has been added previously, remove it from the HPS.INI (or set it to **FALSE**), and the **Selected > Remove** menu item is removed.
3. Set the **LVLEXP_AS_HIERARCHY** to **FALSE**.
This setting in the [FREEWAY] section of the HPS.INI file is used to define the LEVEL MIGRATION EXPORT functionality of the Windows-based Personal Repository and Workgroup repository. It dictates how a LEVEL MIGRATION EXPORT from the Personal or Workgroup Repository acts on import.
 - If set to **FALSE**, the objects are exported independently (Entity Only) and merged on import into a target repository; therefore, no deletes can occur.
 - If set to **TRUE**, the objects are exported as a hierarchy; therefore, implied deletes can occur. This mimics the behavior of a Full Hierarchy Migration Export in a Workgroup Repository migration. Refer to [Migration Export for Repository Objects](#) for more information about migration export.

Issuing a Custom FTP Command

It is possible to issue a custom command for an FTP session during the upload/download processing when using the Personal Repository.

In the [PERSONAL_REPOSITORY] section of the HPS.INI file, set the FTP_CUSTOM_QUOTE option to a specific argument to issue the command during the FTP session on the host server.

For example, set this option as follows:

```
FTP_CUSTOM_QUOTE=site sbd=tcPIP.dutch.tcxplbi
```

or

```
FTP_CUSTOM_QUOTE=site sbdataconn=(ibm-285,ibm-850)
```

For more information on INI settings, refer to the *INI Settings Reference Guide*.

Configuring Mainframe Repository Properties

Configuring Mainframe Repository Properties

AppBuilder uses the Enterprise Repository properties to communicate between a Personal Repository and an Enterprise Repository. This information can be configured for each individual Personal Repository so that each repository can communicate with different mainframes and repositories. The mainframe properties for the Personal Repository can be modified at any time before logging into the Personal Repository.

Take the following steps to modify the mainframe properties:

1. From the Repository Administration tool, log in to the repository.
2. Choose **Edit > Host Properties**. The Host Repository Settings window displays.

Figure 9-4 Host Repository Settings Window

The screenshot shows a dialog box titled "Host Repository Settings - TEST". It is divided into three main sections. The first section, "Host TCP/IP Parameters:", includes text boxes for "Host Name", "Port Number" (set to 0), and "Timeout" (set to 20) with "(Seconds)" next to it. The second section, "Code Pages:", features two dropdown menus: "HOST" is set to "ibm-037 (US English EBCDIC)" and "PC" is set to "cp1252 (Windows Latin 1)". The third section, "Host Repository Parameters:", contains text boxes for "Repository Name", "Version", and "Project", along with an unchecked "Trace" checkbox. At the bottom of the dialog are "Cancel" and "OK" buttons.

3. Fill in the data in the Host Repository Settings window. The following table describes the guidelines for each of the fields. Consult your local mainframe administrator for these settings.
4. Click **OK**.

The system stores the values in the file: <AppBuilder>\<LRE_NAME>.INI.

Table 9-2 Host Repository Settings

Host TCP/IP Parameters Section	
Host Name	Name of the mainframe machine where the consolidation repository resides

Port Number	Number of the port on which the server daemon is listening
Timeout	Number of seconds that the Personal Repository attempts to connect to the mainframe. If the Personal Repository does not have a response from the mainframe within the amount of time specified in the timeout, it generates a timeout error and quits attempting to connect.
Code Pages	
Host:	Host codepage is the codepage that the mainframe is running
PC:	PC codepage is the codepage on the local machine where the Personal Repository is located
Host Repository Parameters Section	
Repository Name	Name of the repository on the mainframe machine
Version	Version of the repository set up on the mainframe machine
Project	Project space that Personal Repository will be using. The Project Name is limited to 4 characters for upload/download compatibility with the host. If you are uploading and downloading using CICS, there is a CICS limitation that does not allow underscores or special characters in Project Names. This field is mandatory. If not completed, a warning message appears saying that the field cannot be blank and will revert back to the old value.
Trace	The trace flag "HOST_TRACE" is sent to the host for tracing the host upload or download process. This flag is not read from the host ini file.

Specifying Upload, Download and Working Directories

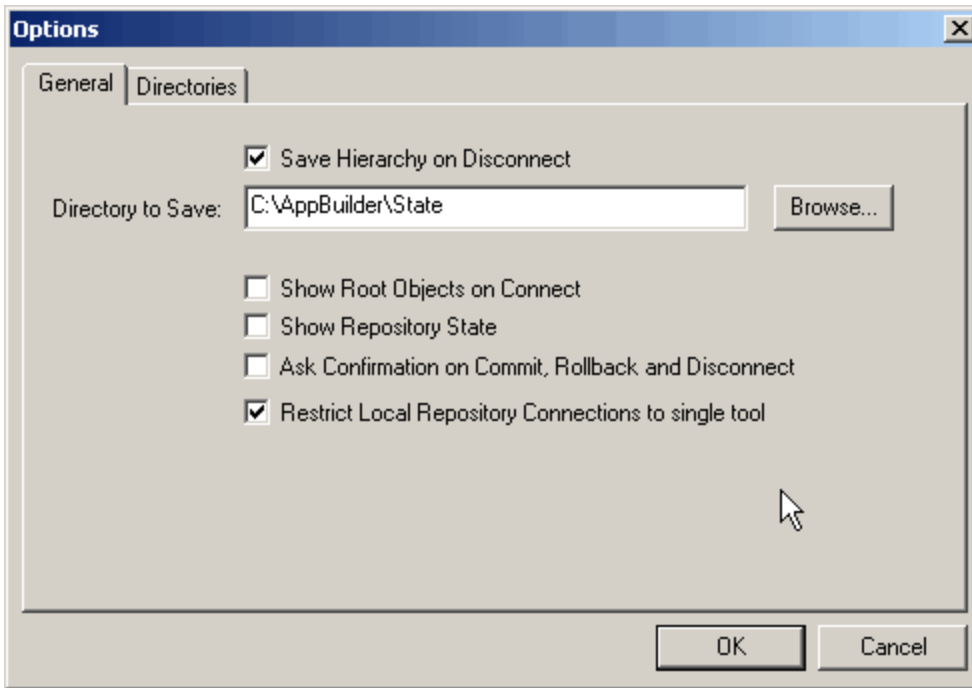
Specifying Upload, Download and Working Directories

All Personal Repositories have the same upload and download directories within AppBuilder. The Personal Repository uses these directories to store files associated with uploading and downloading objects from the Enterprise Repository. File transfer protocol (FTP) is used to upload and download the files between the Personal Repository and the mainframe.

Take the following steps to customize these directories:

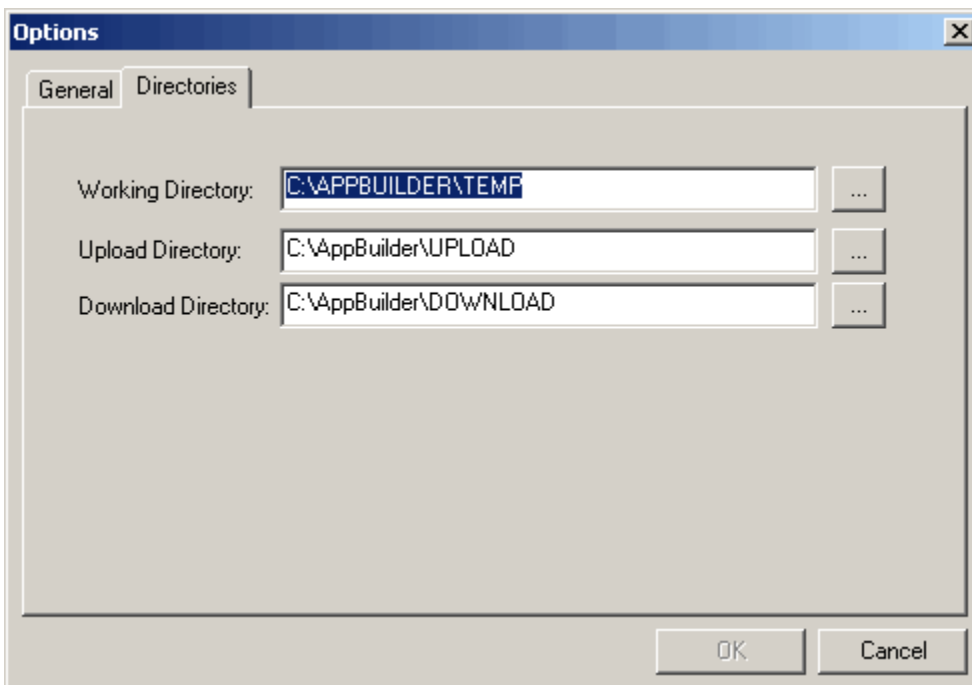
1. From the Repository Administration tool, click **Tools > Options**.
The Options window displays.

Figure 9-5 Upload, Download Options



2. To restrict connections, select the Restrict Local Repository Connections check box:
By selecting this check box you can prevent access to the Construction Workbench, thus restricting multiple jobs from updating the Personal Repository at the same time.
3. Select the **Directories** tab on the Options window.
The Upload and Download directories display.

Figure 9-6 Upload/Download directories



The default directories are displayed. You can modify these directories.



Caution

Select your directory carefully. Periodically, the entire contents of these directories are deleted. Use the default directories. You can change these directories if you require additional space.

4. To modify the directory, click the browse ... button and navigate to the directory you choose. Use the following table as a guideline.

Table 9-3 Upload, Download and Working Directories

Directory	Function	Description
Upload Directory	The directory where files associated with the upload to the mainframe will be kept.	A temporary directory; information stored in this directory is deleted often. The default setting is C:\AppBuilder\Upload. If you need additional space, to change this directory, click the browse ... button and navigate to the directory of your choice.
Download Directory	The directory where files associated with the download from the mainframe will be kept.	A temporary directory; information stored in this directory is deleted often. The default setting is C:\AppBuilder\Download. If you need additional space, to change this directory, click the browse ... button and navigate to the directory of your choice.
Working Directory	A temporary location that is used during the Upload and Download processes.	A temporary directory; information stored in this directory is deleted often. The default setting is C:\AppBuilder\Temp. If you need additional space, to change this directory, click the browse ... button and navigate to the directory of your choice.

5. Once these directories are set, click **OK** to save them.

Downloading Objects

Downloading Objects

This section describes the windows and tools available to operate the download function. The download function works only with a Personal Repository. If you are connected to a Workgroup Repository, this option is not available. This section includes the following topics:

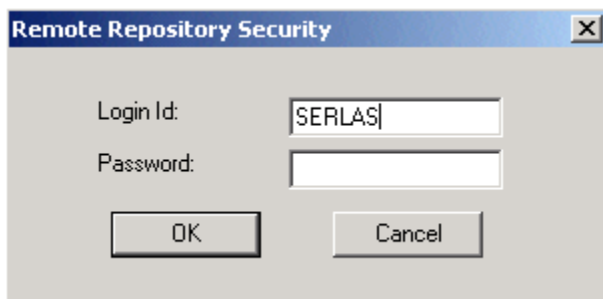
- [Using the Download Window](#)
- [Using the Query Window](#)
- [Using the Load Detail Menu Option](#)
- [Working with Objects in the Download Window](#)
- [Using the Current Job Status Window](#)
- [Selecting Download Scope](#)

To perform a download, take the following steps:

1. From the Repository Administration tool, select **Tools > Download**. The Remote Repository Security window displays.
2. Type the mainframe login ID and password that your mainframe administrator gave you.

If you type an incorrect ID/password combination, the system warns you and suggests you modify your host properties. To do this, refer to [Configuring Mainframe Repository Properties](#).

Figure 9-7 Download Security



During the download window initialization, the Personal Repository's INI file (<AppBuilder>\<Irename>.ini) is read for the host properties and current job status. If the current job status is inactive, the window opens and displays the unit-of-work. The Personal Repository only has one Unit of Work (UOW). It is labeled PERSONAL. Refer to [Performing UOW Migrations](#) for more information about UOW.

If a previous download job is in progress, the status window is opened to monitor the status of the mainframe on operation. If a previous upload job is in progress, you are prompted with a message to either close the window or close the window and automatically open the upload window with the status window present.

If you are using the Repository Administration Tool for an upload or download between a personal repository and an enterprise repository and disconnect, you are prompted: "Disconnecting will close all child windows. Do you wish to continue?" Selecting **Yes** closes the Repository Administration Tool and all child windows. Selecting **No** cancels the disconnection.



These menu items are disabled if a Download or Upload window is already open or if you are connected to a Workgroup Repository.

When the Download window of the Repository Administration Tool is open, the Upload menu item on the Tools menu of the Repository Administration Tool is disabled. Likewise, when the Upload window is open, the Download menu item on the Tools menu is disabled. The disabled menu command becomes enabled when the active Upload or Download window is closed. Only one window (whether Download or Upload) can be active at a time.

Using the Download Window

The Download window initially opens with *no data*. Use the Query window to populate its contents. If you close the Download window, the contents of the query are cleared. Data from closed download windows is *not* stored anywhere on the Personal Repository for repopulating the list when it is reopened. Data must be reentered.

Figure 9-8 Download window after host was queried and objects were loaded

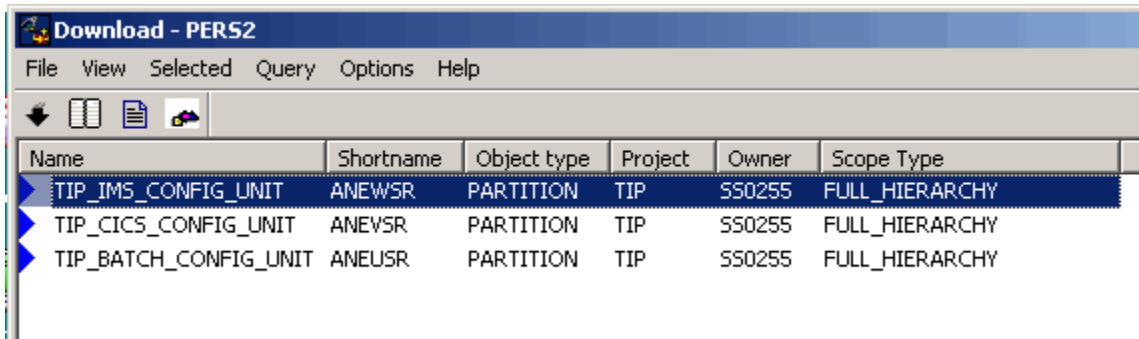


Figure 9-9 Download Toolbar



The following table outlines the menu items and tools available to manage the download.

Table 9-4 Download Menu Items

Menu Items	Selection Names	Descriptions and Settings
File	Download	<p>Initiates a download. Invoking this option causes several events to take place:</p> <ul style="list-style-type: none"> The Personal Repository sends a download request to the mainframe migration server with the object names, object types, and export scopes (i.e. Full Hierarchy, Entity Only, etc.). The mainframe migration server submits a migration batch job with the download list of objects to seed the mainframe migration export. When the batch job is submitted, the migration server can accept other requests. The status window is displayed showing the status of the job until it is complete. Closing the status window box does not affect the progress of the job once it is submitted; the mainframe is not queried for a status until the status window box is opened again. While the status window box is open, the mainframe is queried for job status every <n> seconds as specified in the entry field of the status window box. When complete, the mainframe migration files and log files are retrieved from the mainframe migration server to the download directory. The objects are then migrated into the Personal Repository with the UOW disabled. Imported objects are compared against objects in the UOW. Matching objects are removed (refreshed) from the UOW.
	Load Detail	<p>Opens or loads a mainframe migration *.DETAIL file as a reference to automatically load entities into the Download window. A mainframe migration DETAIL file is a member of the log files retrieved from the mainframe after upload/download operations. It can be found in the temporary FTP directories used for these operations.</p>
	Exit	<p>Closes the Download window. Closing the Download or Upload window does not affect the status of any submitted download or upload jobs.</p>

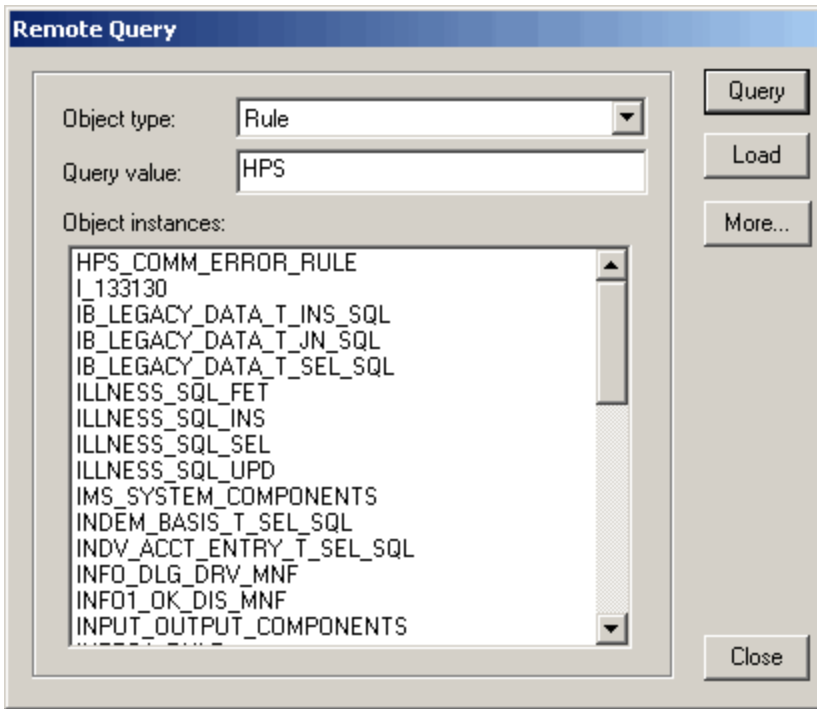
View	Host Properties	Displays the mainframe properties, codepages, and TCP/IP information of the Personal Repository that you are logged into in read-only format.
	Repository Security	Changes the user ID and the password for the download session. Invokes the same window box that is presented when opening an Upload or Download window for the first time.
	Log files	Displays the current mainframe log files that exist in the download directory on the workstation.
Selected	Remove Items...	Removes the selected objects in the Personal Repository download list.
	Change Scope...	Changes the scope of the selected object in the download list. Most queried objects, when loaded into the download window, have a default scope type of Full Hierarchy. You can select multiple objects and change the scope at one time.
Query	Query Host...	Opens the Query window box. The Query window box is used to load objects into the Download window. The Download window does not maintain a list of these objects after it is closed. If you would like to maintain a list to download at another time, use a detail file. Refer to Load Detail .
Options	All or Nothing	The Personal Repository quits the process if any failures happen during the migration import. Prohibits import of any objects if a failure occurs.
	As Many as Possible	The Personal Repository imports as many objects into the repository as possible and ignores errors on independent objects. Successful imports are committed.
	Replace on Import	<ul style="list-style-type: none"> • Always - the migration import of a download always replaces the relations that are deleted on the host. • Never - the migration import always merges the relations, does not delete the relations that were deleted on the host thereby causing a merge. • Prompt - prompts you to select an operation after the download and just before the migration import. <p>When "Yes" is selected from the Replace vs Merge dialog, the relations deleted on the host will be replaced and will be deleted.</p> <p>When "No" is selected from the "Replace vs Merge" dialog, the relations deleted on the host will not be deleted. The drawing model scope of the download will apply implicit merge. Any changes or deletes to the relations of a drawing should use the "full hierarchy" download scope to update or replace the relations.</p>
	Refresh Unit(s)	<ul style="list-style-type: none"> • Always - the UOW always refreshes after the migration import of a download. • Never - bypasses the refresh of the UOW after the migration import. • Prompt - prompts you to select an operation after the download and just before the migration import.
	Refresh Directory	<ul style="list-style-type: none"> • Always - Refreshes contents of repository session with updates. • Never - Refreshes contents of repository session with updates. • Prompt - Prompts you to refresh contents of repository session with updates.
	Import Source	You can download hierarchies with no source (rule source, text, keyword) updates. Refreshes a hierarchy (entities and relationships) without overwriting any source files on which you are currently working.
	Clear Job Status	Sets the PC (<AppBuilder>\<Irename>.ini) job status to inactive. The Personal Repository status of the submitted job is reset. No status checks are invoked until another upload or download is submitted and the status window is reopened for the new job.

Using the Query Window


Use the Query window to populate the Download or the Upload window with the object or objects from the source repository:

1. To open the Query window, select **Query > Query host...** .
 2. Select the object type from the drop-down list populated with various AppBuilder object types.
 3. Select the specific objects to be uploaded or downloaded from the Query value drop-down list.
- The mainframe query is a single-threaded operation; therefore, the mainframe connection remains established until the query is complete. A maximum of forty objects is returned per mainframe query. If the object that you want is *not* in the returned list, click **More** to display the next forty objects.

Figure 9-10 Remote Query Window




4. Select one or more entries in the Object instances pane. Select multiple objects by holding the CTRL key while clicking on objects in the list.
5. Click **Load**.
Or, when selecting only one object, you can double-click on the object to load it.
This populates the Download window or Upload window with the selected items. Each time the Query window is opened, the object type is set to the last queried type.

 The More button is disabled when the host query has reached the end of the table.

Using the Load Detail Menu Option

Use the Load Detail menu option to read from the mainframe migration DETAIL log files in the upload/download FTP directories. The resulting list contains the entities in a *.DETAIL file retrieved from the mainframe after a successful download or upload operation.

Use this file to repeat a download on the selected files. You can edit the *.DETAIL file, save it, and rename it for later use. Click **File > Save**. The resulting edited and saved DETAIL file can be reused for batch purposes.

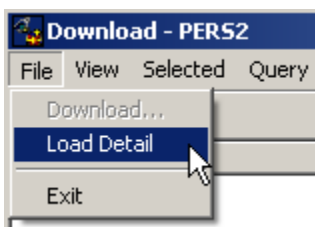
 Removing your modified DETAIL file from the FTP directory ensures that future upload and download operations will not overwrite the file. You can also rename the file, however, the extension must use the .DETAIL filename extension; for example, MYGROUP.DETAIL

You can verify that the objects you are downloading exist in the host repository or you can simply download without verification. If you choose not to verify the objects, Load Detail assumes the entities are in the host repository. If an object fails, the entire download fails. The host migration server will not initiate a download if loaded objects are missing from the host repository.

To load the entities listed in a *.DETAIL file:

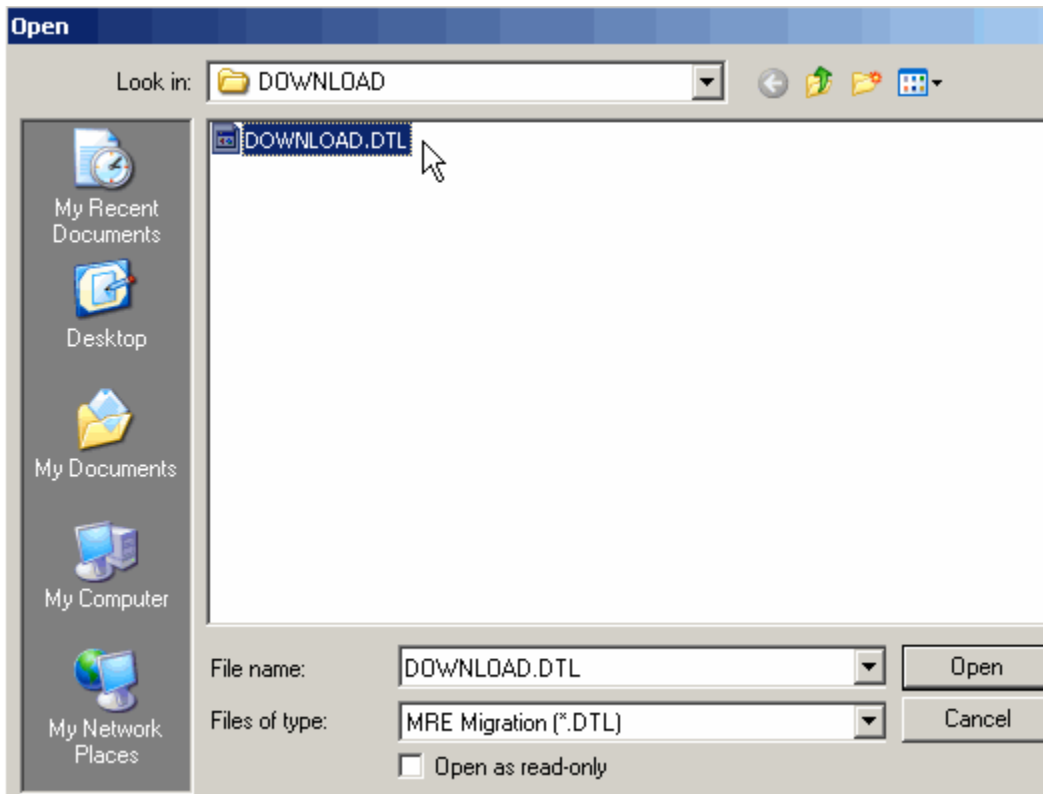
1. Select the **File > Load Detail** option of the Download window.

Figure 9-11 The Load Detail Menu Option



2. Select one of the DETAIL files from the list that displays and click **Open**. The system prompts you to validate the objects on the host.

Figure 9-12 The DETAIL File Results



3. You can click **Yes** or **No**:
 - Yes validates that each entity is in the host repository before the download. An automatic query begins, and the entities contained in the selected DETAIL file are loaded. All AppBuilder entities logged as "updated" or "created" in the mainframe import IMPORT.DETAIL file are included. The *.DOWNLOAD.DETAIL mainframe export file lists all the entities as "exported" in the list.
 - No assumes the entities are in the host repository. If an object fails, the entire download fails. The host migration server will not initiate a download if loaded objects are missing from the host repository. You can edit the DETAIL file with any standard text editor. To reduce the number of entities that automatically load, select and delete the unwanted entities from the list. The entities are listed in the Download window.
4. Click **File > Download** to begin the download process.
5. Save the file after you have modified it for later use. Click **File > Save**. The resulting edited and saved DETAIL file can be reused for batch purposes.



Removing your modified DETAIL file from the FTP directory ensures that future upload and download operations will not overwrite the file. You can also rename the file, however, the extension must use the .DETAIL filename extension; for example, MYGROUP .DETAIL

Working with Objects in the Download Window

The Download window displays the objects that you have loaded. You can perform several actions:

- Remove objects from the download list
- Verify the mainframe connection parameters
- Select the download scope for each object
- Select the download mode
- Perform additional queries

Follow the steps:

1. To remove objects from the Download window, select objects to remove and click **Selected > Remove items**.
2. To verify that the mainframe-connection parameters are correct, select **View > Host Properties**. If any are incorrect, from the Repository Administration window select **Edit > Host Properties** to change them.

3. To set the download scope for each object or multiple objects, highlight the objects in the Download window and click **Selected > Change Scope...** Refer to [Selecting Download Scope](#).
4. To change the download mode, click **Options** and select the desired mode:
 - **All or Nothing** to specify that all objects must be imported successfully or the import will be aborted.
 - **As Many As Possible** to require that the Personal Repository attempt to import all objects but failures do **not** abort the import.
 - **Import Source** to download hierarchies with no source (rule source, text, keyword) updates and refresh the hierarchy (entities and relationships) without overwriting any source files.
 For an explanation of these options, see [Controlling Concurrency](#).
5. When the download scope and download mode are specified for all entities, select **File > Download** to initiate the download. After a short pause, the Job Status window displays. Refer to [Using the Current Job Status Window](#).
6. When the host objects are successfully loaded, the system prompts you to compare these host objects to your UOW. This provides information if you are already working with these objects on your personal workstation.
7. The system then prompts you to Import the Hierarchy Structure. Click Yes if you want to import the relationship objects as well as the selected objects.
8. When the download is completed, the mainframe log files are displayed. To view these logs, select **View > Log files**.

Using the Current Job Status Window

The Status window indicates the progress of an upload or a download. When the window progress bar is full, a request is generated by the Personal Repository to the mainframe migration server for the status of the submitted job. The time interval between status checks is controlled by a numeric entry (in seconds) below the progress bar. Each time the status indicator is open, the time interval is set to the last setting entered. The amount of time a download takes is affected by many factors, such as:

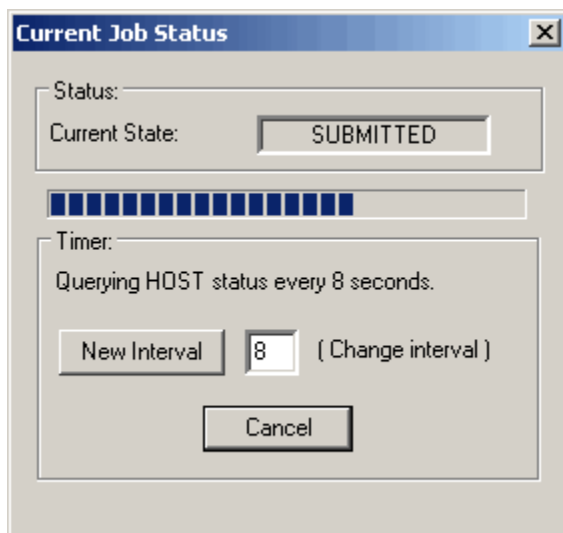
- The current workload of the mainframe migration server
- The size of the hierarchy requested. For example, *Entity only* downloads are much faster than *Full hierarchy* downloads.
- The type of media the mainframe repository data is stored on: disk or tape

The Current Job Status window is used for querying a job status regardless of whether the submitted operation is an upload or a download.

To change the time interval used to query the mainframe server for the status of your job, change the numeric value and click **New Interval**, or click **Cancel** and close the window.

To cancel the download job, you must select **Options > Clear Job Status**. This clears the job on the local machine. The Cancel button on this window simply stops the workstation from pinging the host. When you reopen the Download window, the Job Status dialog displays showing the same status before you clicked **Cancel**.

Figure 9-13 Job Status dialog



Selecting Download Scope

Scope defines the set of objects that can be accessed in the repository. To configure the download scope, select an object and select **Selected > Change Scope**. The Download Entity Scope Types window displays. The default scope value is [Full Hierarchy](#). The following scope types are discussed in this section:

- [Entity Only](#)
- [Full Hierarchy](#)
- [Preparable Entity](#)
- [One Level](#)
- [Drawings](#)
- [Entity Model](#)

- [Table Model](#)

The Replace or Merge dialog window has "Yes" and "No" buttons. When the scope for a Download is Drawing, this option is ignored and the operation is always a merge. When the scope for the Download is an Entity Model or Table Model, the two options of this dialog are used. The "Yes" choice in the dialog replaces and deletes the relations that are missing in the download. A "No" choice in the dialog merges and ignores the missing relations in the download (see [Figure 9-14](#)).

Figure 9-14 Replace or Merge dialog

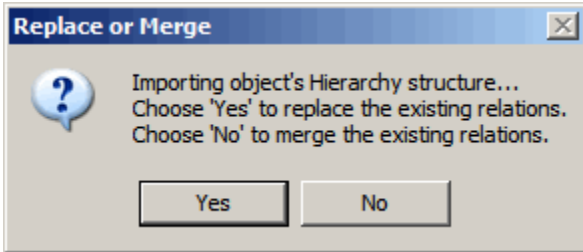
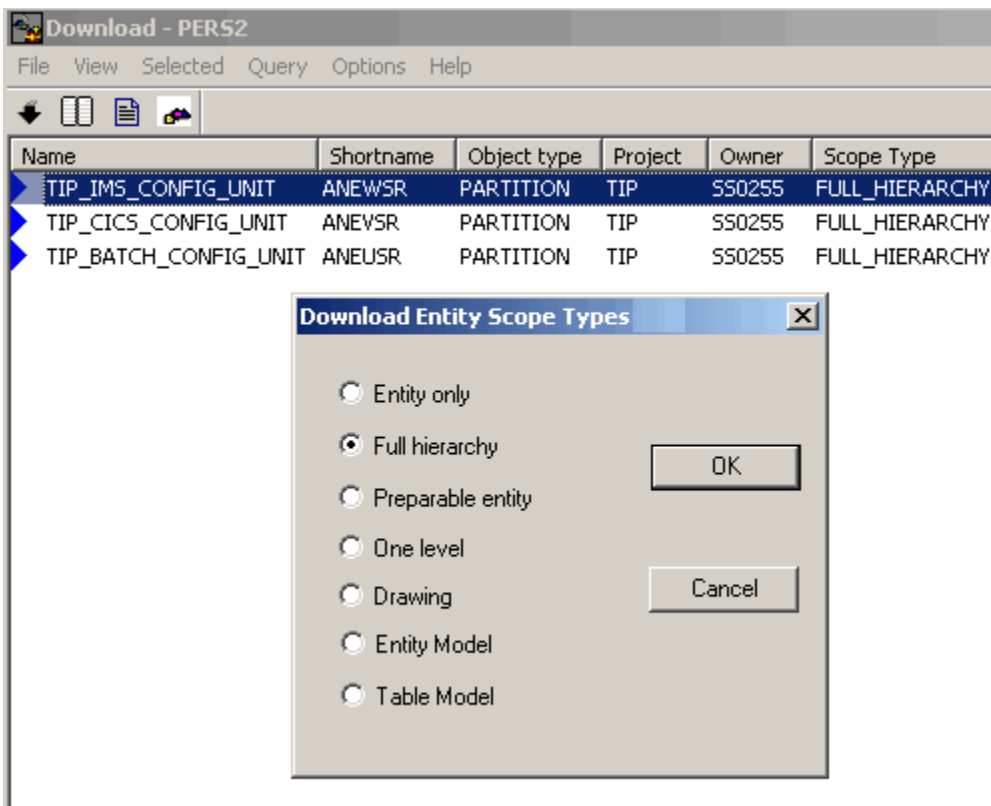


Figure 9-15 Download Entity Scope Types



The migration scope for Drawing objects is limited to Drawing and Full Hierarchy.

Entity Only

Exports the root migration object and its files (source, text and keywords) only. Children objects are not in the scope of this download, so no hierarchy changes are made. The Replace vs. Merge option has no effect on this scope. Entity only always does a merge, because the relationships of the objects are not considered within the migration scope.

Downloading the EXAMPLE_RULE object as Entity Only does not affect this hierarchy. It only updates the rule source, text and keywords if applicable.

Hierarchy 1. Hierarchy on the target (before download):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
  View: EXAMPLE_RULE_VIEW_B
```

Hierarchy 2. Hierarchy on the source (before download):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  View: EXAMPLE_RULE_VIEW_B
```

Hierarchy 3. Hierarchy on the target (after import of Entity Only download of EXAMPLE_RULE):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
  View: EXAMPLE_RULE_VIEW_B
```

Even though the source did not have the Rule EXAMPLE_CHILD_RULE, it is not deleted on the target because it is not within the object's downloaded scope.

Full Hierarchy

Full hierarchy exports the root migration object and all of its child hierarchies in the migration. This migration scope is the default. It is useful for migrating entire branches of a hierarchy. This type of migration can delete relationships on the target if the relationship object was deleted from the source (exported) hierarchy.

Hierarchy 1 shows the hierarchy as it exists before a full hierarchy download of EXAMPLE_RULE. Hierarchy 2 shows the hierarchy as it exists on the source. On the source, EXAMPLE_RULE removed the set EXAMPLE_SET and added a new window EXAMPLE_WINDOW. Hierarchy 3 shows the resulting hierarchy on the target after a replace is done.

Hierarchy 1. Hierarchy as it exists on the target (before download):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
  View: EXAMPLE_RULE_VIEW_B
```

Hierarchy 2. Hierarchy as it exists on the source (before download):

```
Rule: EXAMPLE_RULE
  Rule: EXAMPLE_CHILD_RULE
  View: EXAMPLE_RULE_VIEW_B
  Window: EXAMPLE_WINDOW
```

Hierarchy 3. Hierarchy on target (after import of FULL HIERARCHY download of EXAMPLE_RULE):

```
Rule: EXAMPLE_RULE
  Rule: EXAMPLE_CHILD_RULE
  View: EXAMPLE_RULE_VIEW_B
  Window: EXAMPLE_WINDOW
```

In this example, the set has been deleted, and the window has been added. The resulting hierarchy on the target looks identical to the source migration. If the merge option had been selected on the import, then the set would still exist (the relationship would not have been deleted on

import), and the window would still be added.

Preparable Entity

This scope option is limited to preparable objects only. It exports the root migration object and all of the objects in the hierarchy required to successfully prepare the root object. The scope of the root migration object depends on the level of the hierarchy. The following objects will be part of a root object export if the Preparable Entity scope is selected because they are required for a successful prepare:

- Bitmap
- Component
- File
- Physical Event
- Report
- Rule
- Section
- Set
- Symbol
- Value
- View
- Window

The following objects will be part of a child object export if the Preparable Entity scope is selected because they are required for a successful prepare:

- Section
- Symbol
- Value
- View

Hierarchy 1 shows the hierarchy as it exists in the target before a Preparable Entity download of EXAMPLE_RULE. When the Rule EXAMPLE_RULE is downloaded with a scope of Preparable Entity from the source (Hierarchy 2), the input and output views under EXAMPLE_CHILD_RULE are deleted because they are considered in scope; however, the set under EXAMPLE_CHILD_RULE is not deleted because it is not needed to prepare EXAMPLE_RULE and is out of the scope (see Hierarchy 3).

Hierarchy 1. Hierarchy as it exists on the target (before download):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
    View: EXAMPLE_CHILD_VIEW_I
    View: EXAMPLE_CHILD_VIEW_O
    Set: EXAMPLE_CHILD_SET
  View: EXAMPLE_RULE_VIEW_B
```

Hierarchy 2. Hierarchy as it exists on the source (before download):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
    View: EXAMPLE_CHILD_VIEW_B
  View: EXAMPLE_RULE_VIEW_B
  Window: EXAMPLE_WINDOW
```

Hierarchy 3. Hierarchy on target (after import of PREPARABLE ENTITY download of EXAMPLE_RULE):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
    View: EXAMPLE_CHILD_VIEW_B
    Set: EXAMPLE_CHILD_SET
  View: EXAMPLE_RULE_VIEW_B
  Window: EXAMPLE_WINDOW
```

Views are also further scoped depending on the type of view. Root migration objects consider the following types of views in scope: IN, OUT,

INOUT, GLOBAL, and WORK. Child migration objects of a Preparable download consider the following types of views in scope: IN, OUT, INOUT, and GLOBAL. So, for example if a child rule had a WORK view attached to it, it would not be in the source migration because it is not needed to prepare a root migration object.

One Level

This option exports the root object and any objects one level down in the hierarchy. This migration only includes the direct children of the root object. The objects that are direct children of the root migration object are subject to deletes in the hierarchy. Objects below the 1st level are considered Entity Only objects and are treated like the Entity Only download scope.

Hierarchy 1. Hierarchy as it exists on the target (before download):

```
Rule: EXAMPLE_RULE
  Set: EXAMPLE_SET
  Rule: EXAMPLE_CHILD_RULE
    View: EXAMPLE_CHILD_VIEW_I
    View: EXAMPLE_CHILD_VIEW_O
    Set: EXAMPLE_CHILD_SET
  View: EXAMPLE_RULE_VIEW_B
```

Hierarchy 2. Hierarchy as it exists on the source (before download):

```
Rule: EXAMPLE_RULE
  Rule: EXAMPLE_CHILD_RULE
    View: EXAMPLE_CHILD_VIEW_B
  View: EXAMPLE_RULE_VIEW_B
  Window: EXAMPLE_WINDOW
```

Hierarchy 3. Hierarchy on the target (after import of One Level download of EXAMPLE_RULE):

```
Rule: EXAMPLE_RULE
  Rule: EXAMPLE_CHILD_RULE
    View: EXAMPLE_CHILD_VIEW_I
    View: EXAMPLE_CHILD_VIEW_O
    Set: EXAMPLE_CHILD_SET
  View: EXAMPLE_RULE_VIEW_B
  Window: EXAMPLE_WINDOW
```

The relationship to set EXAMPLE_SET has been deleted, a relationship to EXAMPLE_WINDOW has been created, and all objects under EXAMPLE_CHILD_RULE have been left untouched.

Drawings

Drawings can be downloaded with the scope of Drawing or Full Hierarchy. The download exports the root object (limited to drawing objects) and all of the objects contained in the drawing plus other objects that are related to these objects but are not in the drawing.

If the scope is Drawing:

When a Drawing is downloaded with a scope of "Drawing", all visible objects in the Drawing are added to the MIGXX migration file. This information provides the "seeds" used to perform the download and assists the PC during the import phase of the LRE.

All visible objects in the drawing, plus their relations to objects not represented in the downloaded drawing are downloaded up to two-level.



If there have been changes to the relations of a drawing, use the Full Hierarchy scope to replace the existing relations.

When downloading drawings in the scope of "Drawing", you have the following cases:

- for PDD, STD or DD Drawing – children of the visible objects are extracted up to one-level.
- for ERD Drawing – the following objects are extracted one-level, including any parent Entity of the Relationship:
 - Drawing (seed)*
 - Entity (seed)*

- Business Object
 - Relationship
 - Identifier
 - Attribute
- The relations for this drawing are:
- Has_Identifier (entide)
 - Is_Described_by_Attribute (entatt)
 - Is_related_via_Relationship (entrIn)
- for WFD Drawing – all children of the visible objects are extracted to n-levels.

If the scope is Full Hierarchy:

All visible objects in the drawing, plus their relations to objects not represented in the download, are represented. When downloading drawings in the scope of "Full Hierarchy", you have the following cases:

- When a WFD Drawing is downloaded with a scope of "Drawing" or "Full Hierarchy", all children of the visible objects are extracted to n-levels.
- When a PDD, STD, DD, or ERD Drawing is downloaded with a scope of "Full Hierarchy", all parents and children of visible objects are extracted to n-levels as long as they can be defined in a Drawing.

You can observe that with a Full Hierarchy scope, all drawings are treated the same; there is no special processing for ERD drawings.

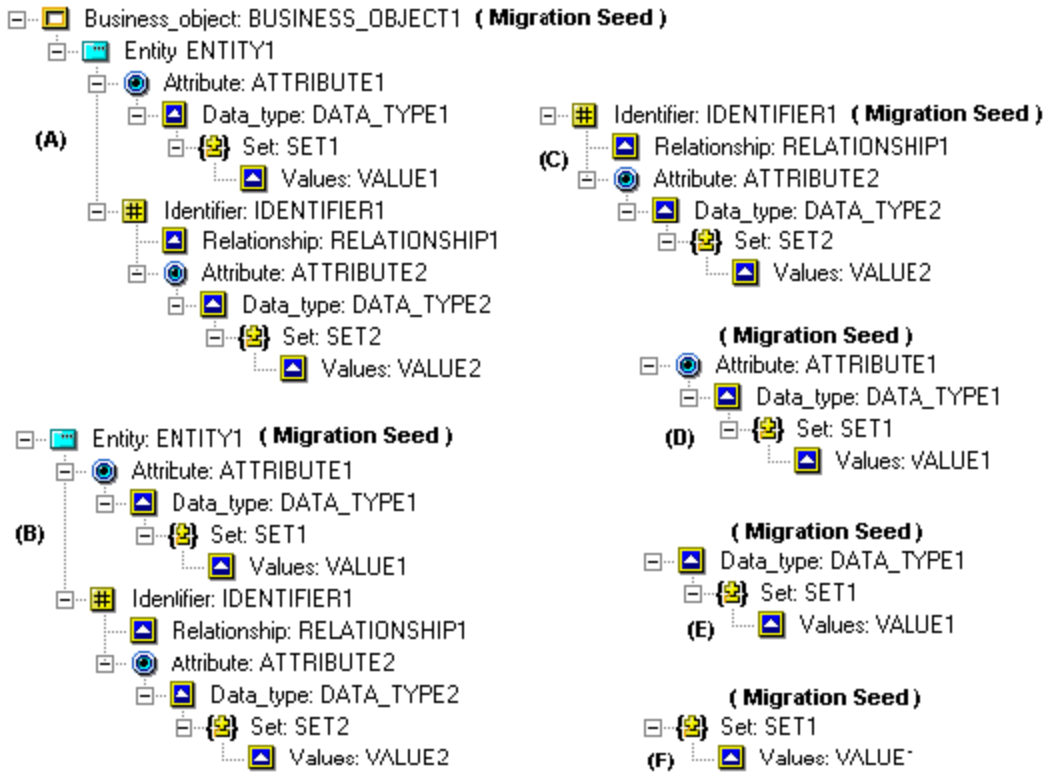
Entity Model

This section discusses which objects can be downloaded using the Entity Model scope and what specifically is loaded into the Personal Repository for each object downloaded. Entity Model scope is applicable for the following entity types:

- [Business Object](#)
- [Entity](#)
- [Attribute](#)
- [Data Type](#)
- [Set](#)
- [Identifier](#)

Figure 9-16 Entity Model scope illustration

Migration Scope - Entity Model



For - (A) & (B) & (C)

Additionally, Download any Entity (less children), that is a child of the downloaded Relationship.

Exception:

(A) If Entity is direct child of Business Object (Migration Seed)

Business Object

The download loads the following objects into the Personal Repository when used for business object:

Table 9-5 Objects loaded during download for business object

Object	Relationship
the business object itself	
each entity connected to the business object	(owns entity)
each attribute for each downloaded entity	(is described by attribute)
each identifier for each downloaded entity	(has identifier)
each attribute for each downloaded identifier	(is composed of attribute)
each relationship for each downloaded identifier	(is composed of relationship)
each data type for each downloaded attribute	(is typed by data type)
each set for each downloaded data type	(is constrained by set)
each value for each downloaded set	(contains value)
each relationship, which is connected to min. one downloaded entity	

each entity	itself not owned by the business object but which is connected to a downloaded relationship. No attributes etc. for this kind of entity.
-------------	------------------------------------------------------------------------------------------------------------------------------------------

Entity

The download loads the following objects into the Personal Repository when used for an entity:

Table 9-6 Objects included in download for an entity

Object	Relationship
the entity itself	
each attribute for the downloaded entity	(is described by attribute)
each identifier for the downloaded entity	(has identifier)
each attribute for each downloaded identifier	(is composed of attribute)
each relationship for each downloaded identifier	(is composed of relationship)
each data type for each downloaded attribute	(is typed by data type)
each set for each downloaded data type	(is constrained by set)
each value for each downloaded set	(contains value)
each relationship that is connected to the downloaded entity	
each (different) entity that is connected to a downloaded relationship, but no attributes etc. for this kind of an entity	

Attribute

The download loads the following objects into the Personal Repository when used for an attribute:

Table 9-7 Objects loaded during a download for attribute

Object	Relationship
the attribute itself	
the data type for the downloaded attribute	(is typed by data type)
each set for the downloaded data type	(is constrained by set)
each value for each downloaded set	(contains value)

Data Type

The download loads the following objects into the Personal Repository when used for data type:

Table 9-8 Objects loaded during a download for data type

Object	Relationship
the data type itself	
each set for the downloaded data type	(is constrained by set)
each value for each downloaded set	(contains value)

Set

The download loads the following objects into the Personal Repository when used for a set:

Table 9-9 Objects loaded during a download for set

Object	Relationship
--------	--------------

the set itself	
each value for the downloaded set	(contains value)

Identifier

The download loads the following objects into the Personal Repository when used for an identifier:

Table 9-10 Objects loaded during a download for identifier

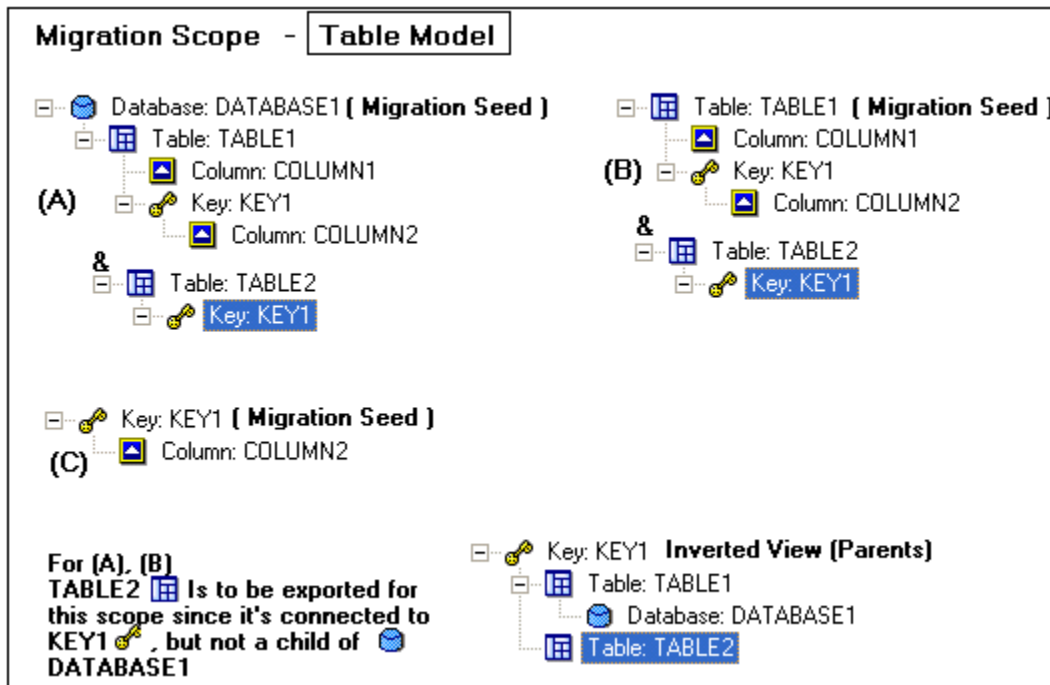
Object	Relationship
the identifier itself	
each attribute for the downloaded identifier	(is composed of attribute)
each relationship for the downloaded identifier	(is composed of relationship)
each data type for each downloaded attribute	(is typed by data type)
each set for each downloaded data type	(is constrained by set)
each value for each downloaded set	(contains value)
each entity that is connected to a downloaded relationship, but no attributes etc. for this kind of an entity	

Table Model

This section discusses which objects can be downloaded using the Table Model scope and what specifically is loaded into the Personal Repository for each object downloaded. Table Model scope is applicable for the following entity types:

- Database
- Table
- Key

Figure 9-17 Table Model scope illustration



Database

The download loads the following objects into the Personal Repository when used for a database:

Table 9-11 Objects loaded during a download for database

Object	Relationship
the database object itself	
each table connected to the database	(has table)
each key for the downloaded tables	(has key, is referenced by key)
each column for the downloaded keys	(has column)
each column for the downloaded tables	(has column)
each table, itself not owned by the database, but which is connected to a downloaded key, but no additional columns, keys etc. for this kind of a table	(has key),

Table

The download loads the following objects into the Personal Repository when used for a table:

Table 9-12 Objects loaded during a download for table

Object	Relationship
the table itself	
each key for the downloaded table	(has key, is referenced by key)
each column for the downloaded keys	(has column)
each column for the downloaded table	(has column)
each additional table, which is connected to a downloaded key, but no additional columns, keys etc. for this kind of a table	(has key)

Key

The download loads the following objects into the Personal Repository when used for a key:

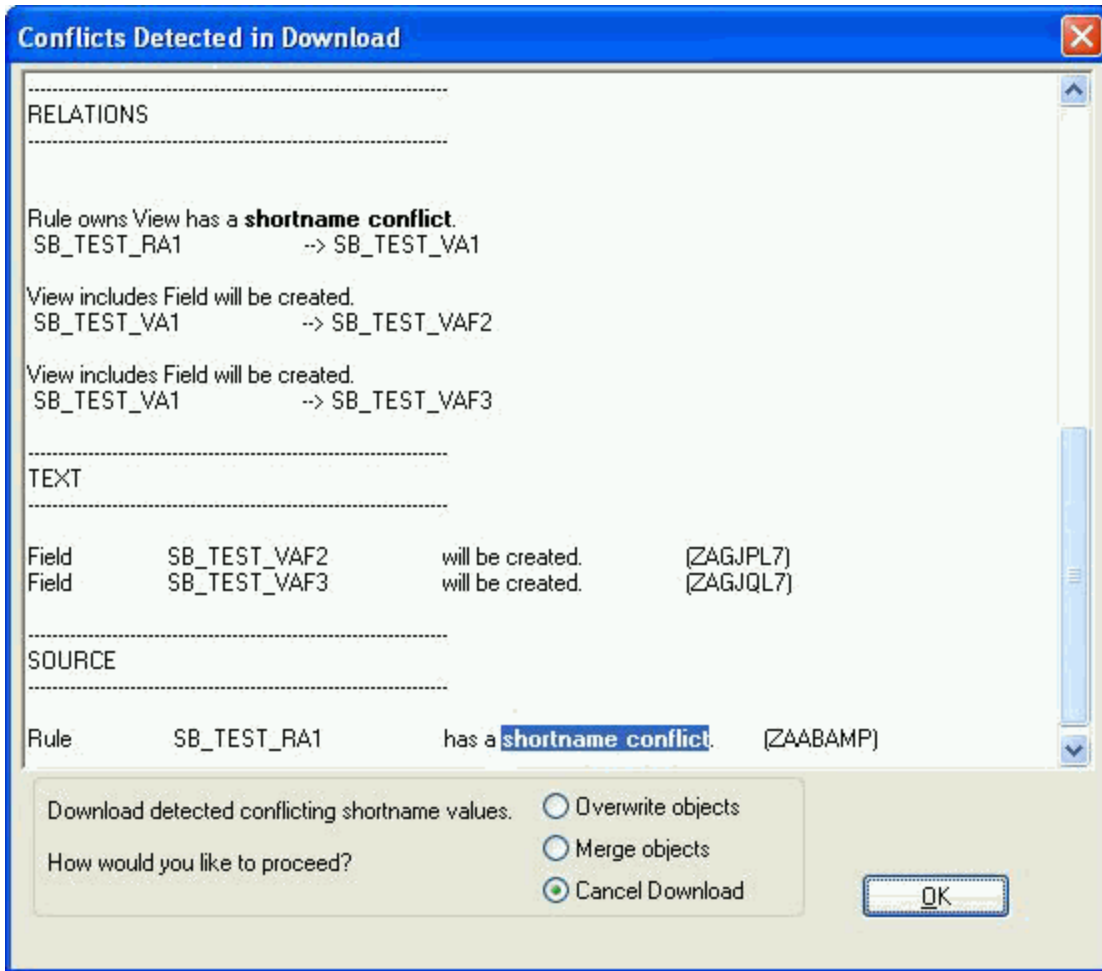
Table 9-13 Objects loaded during a download for key

Object	Relationship
the key itself	
each column for the downloaded key	(has column)

Resolving Short Name Conflicts on Downloads

A short name (SYSTEMID) conflict occurs when the incoming object's long name (NAME) is the same as the existing object's long name but the short name is different. When a short name conflict occurs on a download, the analyze details log is displayed on the workstation. The objects that have conflicts in the short name are highlighted.

Figure 9-18 Download conflict



The dialog explains "Download detected conflicting shortname values. How would you like to proceed?" You can choose one of the following options and press the **OK** button.

Table 9-14 Options for Downloads

Option	Description
Overwrite objects	This option deletes the object from within the repository and all relationships and files attached to it. Then it imports the incoming object and the relationships and files that come with it.
Merge objects	This option causes an update of the like object including the short name of the incoming object. It preserves all the relationships and files of the existing object. It also adds or updates the relationships and files from the incoming download.
Cancel Download	This option aborts the entire process. Analyze report will still be available. The user must make manual modifications within the target repository before the download can be performed.

Uploading Objects

Uploading Objects

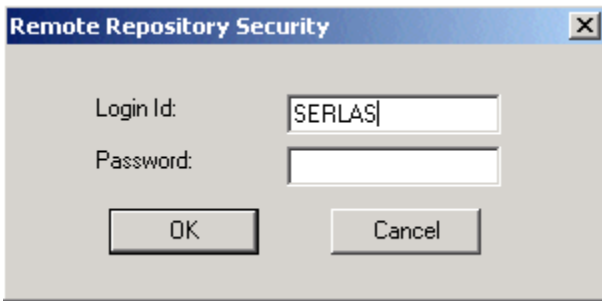
This section describes the windows and tools available to operate the upload function. As with the download, the upload function works only with a Personal Repository. If you are connected to a Workgroup Repository, this option is not available.

The first time you display the Upload window, you are prompted for a user identification and password.


Take the following steps to upload objects from a Personal Repository to an Enterprise Repository:

1. From the Repository Administration tool, click **Tools > Upload**.
The Remote Repository Security window displays.
2. Type the mainframe login ID and password that your mainframe administrator gave you and click **OK**.

Figure 9-19 Upload Security



During the Upload window initialization, the Personal Repository's INI file (<AppBuilder>\<Irename>.ini) is read for the host properties and current job status. If the current job status is inactive, the window opens and displays the unit-of-work. The Personal Repository only has one Unit of Work (UOW). It is labeled PERSONAL. Refer to [Performing UOW Migrations](#) for more information about UOW. If a previous upload or download job is in progress, you are prompted with a message to either close the window or close the window and automatically open the upload window with the status window present.

 These menu items are disabled if a Download or Upload window is already open or if you are connected to a Workgroup Repository.

3. The Upload window is always initialized with a list of objects currently in your UOW. The UOW that the Personal Repository uses is always named PERSONAL. If you want to add additional objects to the upload, use the Query window to populate the Upload window with objects. Refer to [Using the Query Window](#) for steps on how to do this.

Figure 9-20 Upload UOW Window

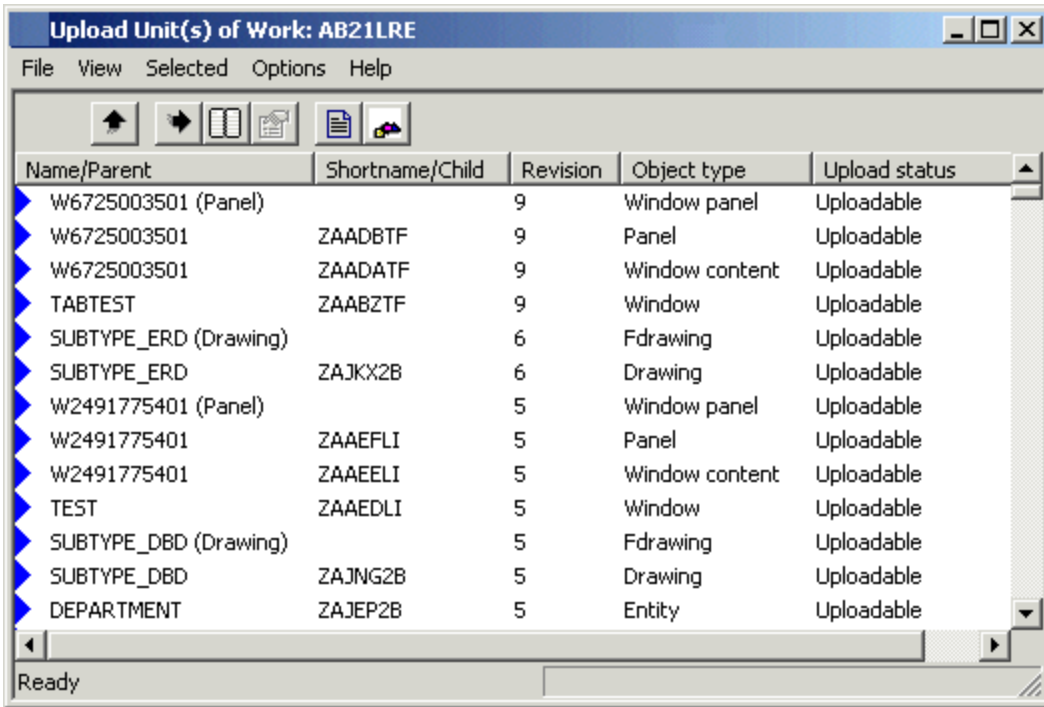


Figure 9-21 Upload UOW Toolbar

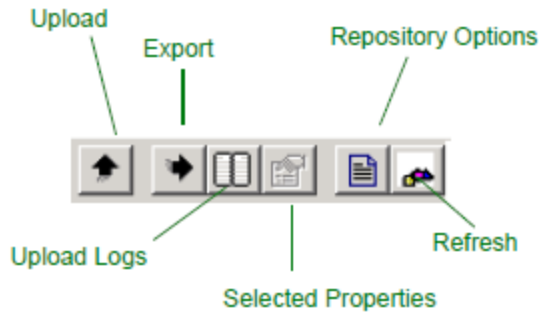


Table 9-15 lists the menu item tools and selections available to configure the upload.

Table 9-15 Upload Menu Items

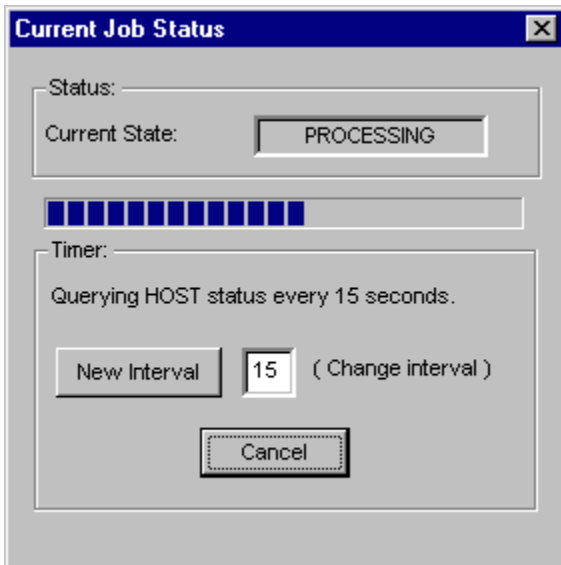
Menu Item	Selection Name	Description
File	Upload	Initiates an upload. When this option is invoked, a series of events take place: * A migration of all the objects listed in the Upload window is performed, and the output files are placed in the upload directory. * The Personal Repository sends an upload request to the mainframe. * The migration files are <i>sent</i> (FTP) to the mainframe migration server. * The mainframe migration server submits the job. Once the batch job is submitted, the migration server can accept other client requests. * The status window is displayed showing the status of the job until complete. Closure of the status window does not effect the progress of the job once it is submitted, the mainframe is not queried for a status until the status window is opened again. * While the status window is open, the mainframe is queried for job status every <#> seconds (specified in the status window). Once complete, the migration log files are <i>got</i> (FTP) from the mainframe migration server. * The Personal Repository parses the detail file from the mainframe migration to determine which objects need to be removed (successfully updated on the mainframe) from the UOW. * The object change number is updated.
	Migrate	Migrates out the UOW objects. This is not an upload and will not contact the mainframe or clear the UOW. This process creates a set of migration files that can be imported into other repositories. If you want to clear the objects after doing a migration, use the Remove option.
	Print	Creates and displays a text file from the information in the browser window. You can print the text file from your third-party editor.
	Exit	Closes the Upload window. Closing the Download or Upload window does not affect the status of any submitted download or upload jobs.
	View	Host Properties
	Repository Security	Use this selection to change the user ID and the password used for the upload session. Invokes the same window that is presented when opening an upload or download window for the first time.
	Refresh	Refreshes the UOW list. Queries the object in the repository UOW again, picking up updated objects (if any) since the last Commit .
	Log files	Displays the current mainframe log files in the upload directory on the workstation.
	Migration Export	Opens migration window for setting export configuration options.
Selected	Properties	Displays the properties window for the selected object. If multiple objects are selected, the properties for the first selected item are displayed, and the menu items are disabled.
	Remove	Removes selected items from the Personal Repository UOW. This command removes them from the window. Commit this change to the repository for this action to be permanent. From the Repository Administration tool, select Repository > Commit .
Options	All or Nothing	Select this to quit the process if any failures happen during the migration import. Will not import <i>any</i> objects if a failure occurs.
	As Many as Possible	Select this to inform the mainframe migration server to import as many objects into the repository as possible and ignore errors on independent objects. Will import and commit those that were successful. If there is a failure, this option will upload only a partial group of objects.

	Clear Job Status	Sets the workstation (<AppBuilder>\<Irename>.ini) job status to inactive. The Personal Repository status of the submitted job is reset, and no status checks are invoked until another upload or download is submitted and the status window is reopened for the new job.
	Refresh Directory	Refreshes the repository directory with updates.

After you have populated the Upload window, you can perform several actions:

- Remove objects from the upload list
 - Verify the mainframe connection parameters
 - Select the upload scope for each object
 - Select the upload mode
 - Perform additional queries
4. To remove objects from the Upload window, select the objects to remove and click **Selected > Remove items**.
 5. To verify that the mainframe-connection parameters are correct, select **View > Host Properties**. If any are incorrect, from the Repository Administration window select **Edit > Host Properties** to change them.
 6. To set the scope for each object or multiple objects, highlight the objects in the upload window and click **Selected > Change Scope**. Refer to [Selecting Download Scope](#) for more information.
 7. To change the upload mode, click **Selected > Change Mode**.
 8. Select **Options > All or Nothing** to specify that all objects must be imported successfully or the import will be aborted.
 - Select **Options > As Many As Possible** to require that the Personal Repository attempt to import all objects but failures do *not* abort the import.
 - Select **Options > Import Source** to download hierarchies with no source (rule source, text, keyword) updates and refresh the hierarchy (entities and relationships) without overwriting any source files.
For an explanation of these options, see [Controlling Concurrency](#).
 9. When the upload scope and upload mode are specified for all entities, select **File > Upload**. After a short pause, the Job Status window displays.

Figure 9-22 Current Job Status window



10. From the Current Job Status window, to change the time interval used to query the mainframe server for the status of your job change the numeric value and click **New Interval**, or click **Cancel** and close the window.
11. To cancel the job, select **Options > Clear Job Status**. This clears the job on the local machine.
12. When the upload is completed, the mainframe log files are displayed. To view these logs, select **View > Log files**.

Performing UOW Migrations

Performing UOW Migrations

To perform Unit of Work (UOW) migrations, you typically perform these procedures:

- [Starting and Stopping the Migration Server](#)
- [Cleaning Up the UOW Migration Entity](#)

The Personal Repository only sends Unit of Work (UOW) migrations, also known as "Delta" migrations, to the TCP/IP migration server.

The mainframe repository only supports UOW migrations during import.

Starting and Stopping the Migration Server

The `CRSRVJCL` administrator method, discussed in [Managing the Migration Server](#), creates a startup JCL for the server on the mainframe repository. This startup JCL can be submitted manually from TSO to start the migration server, or it can be submitted using any automated mechanism. The user ID starting the server job should have surrogate authority.

To stop the migration server, use the skeleton JCL provided with the installation document. By submitting the JCL, the system will start a TCP/IP client. The client initiates a `SHUTDOWN` method on the server. The `SHUTDOWN` method stops the server in an orderly manner by releasing all the socket and DB2 connections.

Cleaning Up the UOW Migration Entity

The upload/download batch jobs create many intermediate files on the mainframe. They also create rows in the migration server DB2 tables. To avoid populating DB2 tables and flooding files, a migration administrator method (`MIGCLUP`) is provided. Invoke the method `MIGCLUP` from administrator management services. This method deletes all the intermediate and result files. It also deletes unnecessary files in the migration server DB2 tables.

Setting Server Security Requirements

Setting Server Security Requirements

Mainframe server security is based on the address space executing the TCP/IP server. The job must be setup to execute under an administrator ID. The Method Handling job is spawned with the same administration ID as the migration server. To set the security requirements, set these permissions:

- [FTP Authorization](#)
- [Repository Authorization](#)
 - [Surrogate Authority](#)
 - [DB2 Authorization](#)

FTP Authorization

FTP security must be set up in RACF for each user with access to the TCP/IP server because FTP is used to send all migration files. All users of the TCP/IP migration server must have access to FTP.

To check security validation on the client, make sure that you can connect to the intended server using FTP. If this access check fails, all services are denied, including services that do *not* use migration files.

Repository Authorization

For access to a repository, permissions must be set to at least one project within the version of the repository being accessed. If you do not have permissions set in advance, all service requests are denied.

Security is checked upon each service request. This is necessary because you can change the target repository with each call. The security check is based on two criteria: the repository and the command.

Surrogate Authority

Surrogate Authority is granted to a user who starts a migration job and wants to submit other migration jobs using a different user ID. The system administrator grants the user surrogate permissions. The authority can later be disabled by using a `.INI` variable [`USEPASS`] in the `@server INI`. If the `USEPASS` is set to 'n', the password can be viewed in the log file.



Surrogate authority in RACF must be set up so that the administrator ID can submit jobs on behalf of the user ID and password sent by the Windows client machine.

DB2 Authorization

The query facilities `DBRM` are bound with the plan for each repository activated for the TCP/IP migration server.

Understanding Performance Implications

Understanding Performance Implications

Performance for the upload (**Analyze/Import**) and download (**Export**) functions is based on the load and the priority of the jobs spawned by the server.

TCP/IP configuration and tuning have a substantial effect in maximizing the processing environment. Refer to the following manual for TCP/IP tuning guidance:

IBM TCP/IP: *Performance Tuning Guide*
Document Number: SC31-7188-01

Or find it at the IBM web site at <http://www.s390.ibm.com>.

Regression Checking

Regression checking is performed by evaluating the change number of a given object in the repository compared to the change number present in the migration files. If a change number of the repository object is equal to or greater than an updated object in a migration file, the object has been updated since download.

Handling Results of Migration

The Analyze/Import and Export services both spawn jobs to the internal reader to handle the migration requests. The server generates a unique identifier upon submission of the migration job. This identifier enables the client to query the status of a given migration.

Controlling Concurrency

It is advantageous in most development environments to have the capability to perform concurrent imports and exports by different users in a repository. Setting the UOW to the parameters **All or nothing** or **As many as possible** can be used to control concurrency based on the number of users accessing a given repository and version. This option also sets limits using DB2 resource locking limits and the implementation of row locking at a given site.

The **All or nothing** setting is the same as an Analyze/Import running with a commit count of zero. The entire UOW must be successful or all changes are rolled back.

The **As many as possible** setting is the equivalent of running an Analyze/Import job with a commit count of one.

Debugging a Migration

Notification of errors and bugs is sent to the client through TCP/IP or FTP in the Informational, Warning, and Error messages. These messages are displayed and exist for all repository services.

Tracing and logging are performed by both the method handler and migration jobs. This information is available through the JES output of the job in question.

Tracing Operations During Upload and Download

Tracing of repository operations during uploads and downloads can be implemented using trace files supplied in AppBuilder. Two files with specific functions are available:

- [LRE_FTP.trace](#)
- [LRE_SOCKET.trace](#)

Both files are in the \appbuilder\nt\sys\bin directory when tracing is turned on. The purpose of having two separate files is to allow the client machine to separate the main communication methods used when communicating between mainframe and client: FTP for the transfer of migration files and Sockets (Winsock) during the messaging process between mainframe and client. These are basic text files that can be viewed with any text editor.

Tracing is turned on and off by entries in the HPS.INI file in the root AppBuilder directory. In the [PERSONAL_REPOSITORY] section of the HPS.INI file, tracing occurs if the values FTP_TRACE and WINSOCKET_TRACE are set equal to "TRUE". The entries in the INI file would be:

```
[PERSONAL_REPOSITORY]
FTP_TRACE=TRUE
WINSOCKET_TRACE=TRUE
```

LRE_FTP.trace

LRE_FTP.trace details the application logic flow of FTP commands used during upload and download and provides the return codes for the FTP

calls. It also contains the information of the codepages, filenames, and the retrieval/placement location of those files used for the FTP process.

LRE_SOCKET.trace

LRE_SOCKET.trace details the messages being passed back and forth between mainframe and client before and after the FTP process is initiated. In the LRE_SOCKET.trace file, entire messages are displayed, however, passwords are kept secure using a "*" naming convention.

Each file is appended to, not overwritten, when an upload or download is initiated. The files must be deleted manually.

Understanding Migration Restrictions

Understanding Migration Restrictions

The following restrictions exist in AppBuilder migration processing:

- Only one upload or download can be submitted per user per repository at one time.
- All repositories that are activated for the TCP/IP server must exist in the same DB2 subsystem.
- The lack of RACF access might create situations that are difficult to debug.
- The DOWNLOAD and UPLOAD methods that exist on the migration object are not to be used outside of the migration server.
- The mainframe repository can only accept a four-character project name from the Windows Personal Repository.

Renaming PC Migration Files

Renaming PC Migration Files

The following table lists the naming equivalencies to use for migrations between PC and mainframe migration files for export to the mainframe and import to the PC:

Table 9-16 Migration File-naming Comparison

PC File	Host File
control.fil	migcntl
elog.fil	elogkey
ent.fil	entity
rel.fil	relate
rlog.fil	rlogkey
inmig.fil	inmig
migxx.fil	migxx
text.fil	text
key.fil	keyword
srct.fil	source
srcbhdr.fil	srcbhdr
srcbdat.fil	srcbdat

Transfer these files between the PC and the mainframe using FTP. Transfer the files using ASCII format, except for srcbdat.fil (srcbdat) files. These are binary files and must be transferred using the binary setting in FTP.

Understanding Mainframe Migration and Repository Security

Understanding Mainframe Migration and Repository Security

Migration between PC repositories is accomplished from a PC workstation; however, the ability to migrate is dependent upon defined levels of authorization in the respective security models for each repository.

During the Export phase, a user exports objects from the source repository. These objects are stored in temporary files, loaded into migration tables, compared to the target repository tables, and then finally migrated into the target repository tables. The initial migration Import assigns ownership in the target repository for the entities included in the migration files. The owner is the pre-existing OWNER defined in the attributes of the object.

When importing objects into the Enterprise Repository, the project name must be four (4) characters long and the project to which the object is defined must exist within the security model in the target Enterprise Repository. If the project does not exist, the import will fail.

Additionally, the user executing the import must have *create* and *update* permissions to the target project, or the import will not successfully import the relationships between some objects.

Any user who successfully migrates an object is identified by the object audit information in the target repository as the last user. This is the case for the initial migration and for successive migrations.