CA Gen

Client Server Encyclopedia Version Control User Guide

Release 8.5

CA Technologies Product References

This document references the following CA Technologies products:

- CA Gen

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Appendix B: Migration Rules 143

Chapter 1: Introduction

CA Gen Client/Server Encyclopedia Version Control is designed to help you synchronize the definitions of two versions of the same object in different models within one Client/Server Encyclopedia (CSE). This and other model management tasks are described in Version Control Tasks.

CSE Version Control Functions

When you start the Version Control Client and display the main window, you will notice six functions under the Model menu:

- Adopt
- Migrate
- Compare
- Trial Adopt
- Trial Migrate

Following is a high-level overview of these Version Control functions in the order they appear on the menu.

Adopt

Adoption is the Version Control function that enables you to propagate the ancestry of a source model object to an object that is logically the same in the destination model. You will use Adoption when two objects in different models are logically the same, but are not recognized as versions of the same object by Migration and Compare Report. Adoption does not affect object definitions. During Adoption, all selected aggregate objects and their component objects that can be adopted will be adopted. Any objects that cannot be adopted will be reported as not adopted, ignored, or already adopted. When a parent object is adopted, but a component of that object is not adopted because it is not logically the same as any component in the source model, that component is reported as unadopted. Adoption can be successful even when some objects cannot be adopted.

For answers to other questions you have about Adoption, see FAQs About Adoption in the chapter "Performing Adoption."

Migrate

Migration is the Version Control function that enables you to selectively copy aggregate objects from the selected source model to the selected destination model. Copied objects replace equivalent objects, if they exist. Otherwise, copied objects are added to the destination model. For a graphical depiction, see Migration Replaces Equivalent Objects and Adds New Objects in the chapter "Migrating Objects from One Model to Another."

Migration copies each object you select in context as an aggregate object with related objects. Migrating the related objects maintains the integrity of the aggregate object's definition. There are three kinds of related objects: component objects, which Migration always migrates with the aggregate, companion objects, which Migration migrates with the aggregate if they do not already exist in the destination model, and enabling objects, which you must select for migration along with the aggregate object if they do not already exist in the destination model.

If any selected object cannot be successfully migrated, no objects are migrated. This ensures that Migration always preserves the integrity of the destination model.

For answers to other questions you have about Migration, see FAQs About Migration in the chapter "Comparing Objects in Different Models."

Compare Report

The Compare Report is the Version Control function that compares selected aggregate objects and their subordinated aggregate objects in two models in the same encyclopedia and reports differences. The resulting Compare Aggregate Object Report lists:

- Objects that exist only in the source model or in the destination model
- For objects that exist in both models, that is, for equivalent objects
  - Which object pairs are the same versions in both models
  - Which object pairs are different versions in each model

You will find this information useful when you begin to identify objects to migrate from one model to another. For example, objects in different versions are candidates for migration—usually you will select the later version of the object to replace the earlier version. For a sample scenario illustrating this, see Evaluating the Compare Aggregate Object Report. For answers to other questions you have about Compare Report, see FAQs About Compare Report in the chapter "Comparing Objects in Different Models."

Trial Adopt Report

Trial Adoption is the Version Control function that produces a report of the results you could expect if you performed Adoption on the same selection of objects. Trial Adoption allows you to examine potential adoption results without changing object ancestry.

Trial Migrate Report

Trial Migration is the Version Control function that produces a report of the results you could expect if you performed Migration on the same selection of objects. Trial Migration allows you to examine potential migration results without actually adding or replacing any objects in the destination model.

Note: When you select objects for Migration, Adoption, or Compare Report, you select aggregate objects. The use of aggregate objects enables Version Control to preserve the design context of objects.

Aggregate Sets

Aggregate sets is the Version Control function that enables you to reuse a set of aggregate objects you have selected for Adoption, Migration, or any of the Reports. You can select an aggregate set you created for one function and reuse with another function. You can select an aggregate set you used for a trial report and modify it to use with another run of that report.

Differences from Host Encyclopedia Functions

If you have used Version Control on the Host Encyclopedia, you will recognize much of the functionality incorporated into the CSE Version Control. The following table describes a summary of differences in functionality between the CSE and the Host:

Functionality	CSE	Host	Considerations
Migration and Trial Migration	X	X	
Adoption and Trial Adoption	X	X	
Compare Aggregate Objects	X	X	
Reporting	X	X	

Functionality	CSE	Host	Considerations
Aggregate sets	X	X	In CSE Version Control, an aggregate set cannot be created as output from a Version Control function.
Model Unadoption		X	
Model families		X	In CSE Version Control, all models in one encyclopedia implicitly belong to the same family.

Migration

Migration differs with encyclopedia in the following areas:

- Migration of Constraint aggregate objects
- Effect of model checkout status on migration
- Availability of migration to a new model

The following table describes a summary of differences in migration between the CSE and the Host:

Migration Function	CSE	Host	Considerations
Constraint is an aggregate object	X		
Migration possible when destination model is checked out	X		Protection of individual objects in destination model is respected.
Migration of objects to a new model possible		X	For this functionality in CSE, use one of the following commands from the Encyclopedia Client: Copy From This Encyclopedia or Create Model From Subset.
Object IDs included in Migration report	X		
Can specify number of errors as a threshold for terminating migration processing			

Adoption

Adoption differs with encyclopedia in the following areas:

- Terminology
- Requirements for common ancestry
- Model checkout status
- Adoption requirements for system-defined objects
- Adoption report messages and options

The following table describes a summary of differences in adoption between the CSE and the Host:

Adoption Function/Area	CSE	Host	Considerations
Term for the model whose objects' ancestry is propagated during adoption is Source model	X		Host term (corresponding to source model) is Related model.
Term for the model from which you select objects for ancestry replacement is	X		Host term (corresponding to destination model) is Adoptee model.
The model containing objects targeted for ancestry replacement can be checked out.		X	
You must adopt system-defined objects to establish common ancestry	X		In a CSE, system-defined objects share common ancestry across all models and encyclopedias without being adopted. When a model is created in the encyclopedia, the system-defined objects are automatically adopted.
Can specify number of errors as a threshold for terminating migration processing		X	
Adoption report messages include Not adopted and Unadopted.	X		For message meanings, see the table in Adoption Report Messages in the chapter, "Performing Adoption."

Adoption Function/Area	CSE	Host	Considerations
Constraint and Default Edit Pattern are selectable options.	X		Both object types are adoptable on both platforms.
Index (non-identifier based) can be selected and adopted			

Compare Aggregate Objects

The Compare Aggregate Object Report differs with encyclopedia in the following areas:

- Report options
- Objects reported
- Objects that can be selected

The following table describes a summary of differences in report comparison between the CSE and the Host:

Compare Report Function	CSE	Host
Provides option of limiting report to selected sections: Different, Same, Source Only, Destination Only	X	
Provides option of reporting all sections: Different, Same, Source Only, Destination Only	X	X
Reports only on objects that can be explicitly selected for migration (no non-selectable)	X	
Constraint is an aggregate object	X	

Aggregate Sets

The following table describes a summary of differences in aggregate set function between the CSE and the Host:

Aggregate Set Function	CSE	Host
Aggregate set can be created/modified during compare, adoption, or migration process		X
Multiple aggregate sets can be used as input for compare, adoption, or migration	X	

Aggregate Set Function	CSE	Host
When there are aggregate objects in set that are not found in model, produces Discarded Aggregate Objects report		X
When there are aggregate objects in set that are not found in model, produces objects retrieved vs. objects in set message	X	
Constraints is a selectable aggregate object	X	

Differences in Reports

The following table describes a summary of differences in reports for version control between the CSE and the Host:

Report (Object Range)	CSE	Host	Considerations
Aggregate Set Reports		X	
Compare Aggregate Object Report	X	X	
Trial Adopt Model Report (All objects)	X	X	
Adopt Model Report (All objects)	X	X	
Trial Adopt Objects Report (Selected)	X	X	
Adopt Objects Report (Selected)	X	X	
Trial Migrate Aggregate Object Report	X	X	
Migrate Aggregate Object Report	X	X	
Migrate Model Report		X	Migrate to new model is not a supported function.
Where Exists Report		X	

Change Capture for Version Control

Change Capture differs with encyclopedia. The CSE allows the reporting of change and subsequent migration to be done at a more granular level than the Host Encyclopedia allows. The following table describes a summary of differences:

Change Capture Function	CSE	Host
Objects marked changed on basis of adding, deleting, or moving an association between two objects.	X	
Objects marked changed on basis of change in description property.	X	
Constraint is treated as an aggregate object	X	

Version Control Tasks

The following table describes How Version Control functions can be used to satisfy Version Control tasks:

Objective	Task
Maintain multiple models that represent the same system at different stages of development, testing, and production.	Migrate new and changed objects through the different model versions.
Synchronize models used for parallel development so that all objects in one model are the same version as equivalent objects in the other model.	Migrate new and changed objects back and forth between models on a regular basis.
Reuse and synchronize common objects.	Migrate changed common objects from the common objects model to each development model where they are used.
Synchronize models with objects that are logically the same but were initially implemented independently.	Adopt objects that are logically the same, and then migrate the equivalent objects to synchronize object versions.

Getting Ready

When you select the source model and destination model for any Version Control function that function verifies that you are authorized for the function and the models you select meet the requirements in the areas of language code and schema level. If restrictions apply to a selected model in either of these areas, a warning message or error message appears. Models that are unusable, models that reside on different encyclopedias, and models that do not meet schema requirements for the particular Version Control function, do not appear on the selection lists for source model and destination model. Child models appear on the destination model selection list only for the Compare Report; child models cannot be selected as destination models for any other Version Control function. There are no restrictions on whether the source or destination model is checked out for any Version Control function.

Model Requirements

This section describes the model requirements enforced by Version Control as well as recommendations for meeting these requirements. You need not investigate in advance whether particular models can be used for a given Version Control function. You are notified through message dialogs if the models you select cannot be used successfully.

Language Code

Both the source and destination models must be stored with the same language code. See Verify Language Code Combination is Valid for more information. A Zero Language Code Warning message indicates that one or both selected models have a language code of zero.

If a Zero Language Code Warning message appears when you select the models, convert the selected model to establish the correct code page value. For details, see the *Online Help* associated with the message dialog, or see the *Client Server Encyclopedia User Guide.*

Important! If you attempt a migration where the destination model uses a different language code than the source model, migration can cause unpredictable results.

Schema Level

You can Migrate or Trial Migrate objects between models only if the destination model's schema is equal to or greater than the source model's schema. You can Compare, Adopt, or Trial Adopt objects between models regardless of schema level. Supported model schemas are Prior Schema and Current Schema.

Note: For the schema designation for Current Schema, Prior Schema, and Second Prior Schema, see the *Release Notes*.

The following table describes the prerequisites to perform the listed functions:

Function	Source Model Schema	Destination Model Schema
Migration	Prior Schema Current Schema	Prior Schema or Current Schema Current Schema
Trial Migration	Prior Schema Current Schema	Prior Schema or Current Schema Current Schema
Adoption	Prior Schema or Current Schema	Prior Schema or Current Schema
Trial Adoption	Prior Schema or Current Schema	Prior Schema or Current Schema
Compare	Prior Schema or Current Schema	Prior Schema or Current Schema

Authorization

You must have proper authority on the models you select to perform any Version Control function. The following table describes the authorizations required for source and destination models:

Function	Minimum Authority on Source Model	Minimum Authority on Destination Model
Migration	Read	Migrate To
Trial Migration	Read	Read
Adoption	Read	Migrate To
Trial Adoption	Read	Read
Compare	Read	Read

The following table describes the encyclopedia access requirements to perform Version Control functions:

Aggregate Set Process	Minimum Authority
Add	Encyclopedia access. The set administrator is the userid used when adding the set.
Copy, Detail	Encyclopedia access

Aggregate Set Process	Minimum Authority
Change Administrator, Delete, Modify, Rename	Set administrator

Model Status

The following statuses indicate that the model is usable:

- Modifiable
- Read-only
- Checked-in
- Waiting-for-verify

Note: No Version Control functions can be performed on a model with a status of Unusable.

The following table discusses the model status requirements to perform Version Control functions:

Function	Source Model	Destination Model
Migration	usable	modifiable
Trial Migration	usable	modifiable
Adoption	usable	usable
Trial Adoption	usable	usable
Compare	usable	usable

Child Model Usage

You can use a child model as the source model for any Version Control function. You can also use a child model as the destination model for Compare Report.

The following table discusses the models that can be used as source and destination models:

Function	Source Model	Destination Model
Migration	Child Model	N/A
Trial Migration	Child Model	N/A

Function	Source Model	Destination Model
Adoption	Child Model	N/A
Trial Adoption	Child Model	N/A
Compare	Child Model	Child Model

Model/Subset Checkout Status

You can use any Version Control function without regard to the checkout status of the source or destination models or their subsets. However, if the destination model or any of its subsets are checked out when you perform a migration, the protection level of individual objects will be respected. For details, see Attempt to Reference/Modify/Delete Object(s) Which are Checked Out in the chapter, "Migrating Objects from One Model to Another."

Model Residence Requirement

Source and destination models must be in the same CSE. Version Control is not supported across encyclopedias.

Recommendations

The following are the actions you need to take to enhance your success with Version Control:

- Become familiar with the models you select for migration or adoption prior to using these functions.
- Make sure the models you plan to use together are stored with the same language code.
- If your models were created with an earlier version of CA Gen, consider converting the models.
- If the model you want to use as the source model is in a different encyclopedia than the destination model, either copy the model to the encyclopedia where the destination model resides, or extract the model to the required encyclopedia and use the child model as the source model.

Know Your Models

It is important that you be familiar with both models before performing a migration or adoption. One way to gain familiarity with the models is through studying the Compare Aggregate Object Report, the Trial Migrate Aggregate Object Report, and the Trial Adopt Objects Report or the Trial Adopt Model Report.

Verify Language Code Combination is Valid

Evaluate the language code of each model via the Detail pushbutton on the Model Selection dialog. Both the source and destination models must be stored with the same non-zero language code. If you select a model with a zero language code, a Zero Language Code Warning is displayed. Selecting a model with a zero language code can cause unpredictable results. For details, see *Online Help* for Zero Language Code Warning or see the *Client Server Encyclopedia User Guide*.

Consider Converting Existing Models

Converting Second Prior Schema or Prior Schema models to Current Schema enables you to take advantage of the new functionality associated with the Current Schema.

Note: For the schema designation for Current Schema, Prior Schema, and Second Prior Schema, see the *Release Notes*.

Note that the only actions you can take on a Second Prior Schema model are delete or convert. For more information about the Convert command, see the *Toolset Help;* for more information about model conversion, see the *Client Server Encyclopedia User Guide*.

Use Extract or Copy when Models are on Different Encyclopedias

If the model you want to use as the source model for any Version Control function is on a different CSE than the destination model, you can extract that model to the same encyclopedia using the Extract / Model command from the Encyclopedia Client. You can then use the child model as the source model for Migration, Adoption, or Compare Report. This is an alternative to using the Copy / From Another Encyclopedia command to create a new source model like the one on the other encyclopedia.

Guided Tour

Five Version Control functions share many of the same dialogs. The following discussion walks you through the Version Control dialogs you are likely to use the most, and offers usage tips. Once you are familiar with these dialogs, you will find them easy to use when you encounter them. You can find it helpful to start the Version Control Client and proceed through this section.

Starting the Version Control Client

Follow these steps:

1. Start the CA Gen Version Control Client either by clicking on the Version Control Client icon or by entering the following command line command:

   ```
   iefvc vcfunc open
   ```

2. Enter the CSE Servers host name, service name or port, your CA Gen user ID, and the password if your account requires one, and click OK.

   Note: The hostname field can contain a hostname, a hostname with domain name , or an Internet Packet version 4 (IPv4) or version 6 (IPv6) address. The hostname can be a maximum of 1024 characters in order to accomodate host and domain names that contain non-ASCII characters.

3. If you are authorized for more than one encyclopedia, select the required encyclopedia, then select Logon.

   For more information, see one of the following procedures:

   - Performing Object Comparison (Chapter 2)
   - Performing Adoption or Trial Adoption (Chapter 4)
   - Performing Migration or Trial Migration (Chapter 5)

Using a Filter

Many of the Version Control dialogs allow you to filter what is displayed to make selection easier. Access *Online Help* from any filter entry field for detailed instructions on creating and using filters.

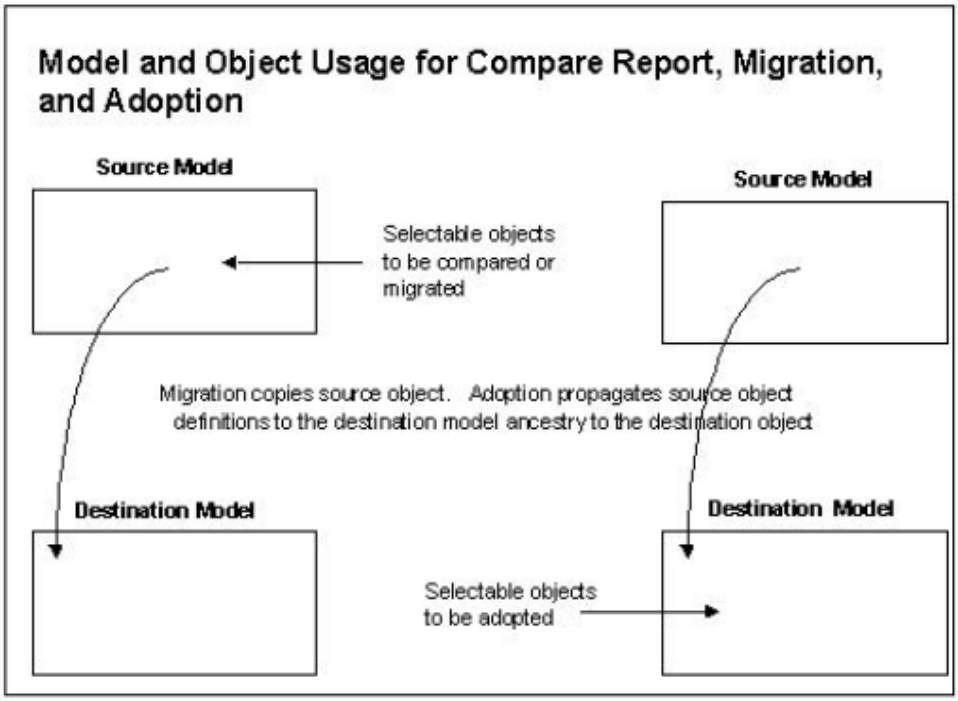
Type the appropriate characters in the filter field and press tab to display the following selections:

Selections	Type
beginning with an S	S or S%
with an embedded Z	%Z
with B as the third character	__B (underscore, underscore, B)

Determining Which Model Is the Source Model

In selecting two models to compare, you specify one as the source model and the other as the destination model. For Compare Report, Migration, and Trial Migration, the source model is the model whose object occurrences are displayed for selection. For Adoption and Trial Adoption, the destination model is the model whose object occurrences are displayed for selection. Select a source and destination model for every Version Control operation you perform.

The following figure shows the relationship of source and destination models for the various Version Control functions. As the figure suggests, when object occurrence selection lists display, the Compare Report and Migration/Trial Migration functions display source model objects for selection. Adoption/Trial Adoption displays destination model objects for selection:



Evaluating the Model Checkout Status

The Model Selection List displays the checkout status next to the name of each model available for selection as a source or destination model. You can perform any Version Control function, regardless of whether either model or one or more of their subsets are checked out. However, protection levels associated with objects in a checked out model or subset will be respected by Migration. See Attempt to Reference/Modify/Delete Object(s) Which are Checked Out in the chapter, "Migrating Objects from One Model to Another."

The following table lists the model's checkout status for a corresponding checkout user ID:

Checkout User ID	Means
blanks	the model is not checked out; none of its subsets are checked out
a User ID	the model is checked out by the identified user
********	one or more subsets of the model are checked out

Selecting the Object Range

When determining whether to use All or Selected as the Object Range for Adoption, see Determine Whether to Adopt All or Adopt Selected Objects in the chapter "Performing Adoption."

When determining whether to use All or Selected as the Object Range for Compare Report or Trial Adoption with small to moderate models, see the following table.

The following table discusses the advantages of each of the Object Range options:

All	Selected
No object selection time needed. Resulting report is comprehensive; there is no chance of leaving out information you want. Provides a good baseline report if using Version Control on the same models on a regular basis. Desirable for running the Compare Report; use Compare results on all objects to create aggregate set of required objects for next process.	Processing time is generally shorter. Report can be easier to analyze because it focuses on objects of greatest interest. Report can take less time to analyze because of its shorter length. Desirable with aggregate set for Trial Adoption, Adoption, Trial Migration, and Migration.

Note: For large models, selected objects can be the preferable alternative because Version Control functions can be resource intensive. If you choose selected objects, you can select objects individually or select one or more aggregate sets composed of objects previously selected.

Specifying How to Save Reports

Each time you use a Version Control function, a report is generated. The default directory is the directory where the Version Control Client is installed; each report has a default name. If you accept the defaults, you will retain only the latest version of each type of report generated.

If you want to archive reports or retain a series of reports for tracking changes, consider developing file-naming standards to ensure filename uniqueness for each report type. Also consider creating directories indicating date ranges and direct all reports, say for the quarter, into the same directory.

Selecting Aggregate Objects

You can select Aggregate objects used in any phase of an application development life cycle for Version Control functions. To know more about list of aggregate objects, see one of the following:

- For a list of aggregate object types available for Compare Report, Migration, and Trial Migration, see Become Familiar with Aggregate Object Types in the chapter, "Creating and Maintaining Aggregate Sets."
- For a list of aggregate object types available for Adoption and Trial Adoption, see Become Familiar with Aggregate Object Types for Adoption, in the chapter "Performing Adoption."

To select objects for any Version Control function, you first select the aggregate object type, then select the specific aggregate object occurrence from a list of objects of that type that exist in the model. If a selected object has subordinate aggregate objects, you can expand to list the subordinates for selection. If the set of objects you plan to select is one that you can use more than once, consider defining it as an aggregate set. Then select the set each time you need those particular aggregate objects.

How Aggregates Help Minimize Object Selection

When an object type is defined as an aggregate object type for a particular Version Control function, that function considers objects related to the aggregate object when evaluating it. One advantage of being able to select aggregate objects is that you do not have to select all the low-level components of the objects of interest to compare objects, migrate objects, or adopt objects. For example, you do not select literals, fields, special fields, and edit patterns when you want to compare, migrate, or adopt a screen occurrence. These objects are the components of the aggregate object type called screen, and therefore are comprehended automatically when you perform a Version Control function on a screen object occurrence.

Strategies for Simplifying Object Selection

There are various ways to reduce the number of object occurrences on scrollable lists. The following are some tips:

- Use aggregate sets, when possible.
- Use the Filter feature. See Using a Filter.
- If selecting a low-level aggregate of a type where there are numerous occurrences, begin with the highest level aggregate object type up the chain of subordinates from that occurrence and expand until you reach the level of the object of interest.

For example, to select a particular procedure step action block, you could do one of the following:

- Select the related parent business system and expand to its procedures
  - Select the related procedure and expand to its procedure steps
  - Select the related procedure step and expand to its procedure step action blocks
  - Select the procedure step action block of interest
- Select the object type of interest initially, then use the slider bar to scroll; object occurrences are listed alphabetically.

How Version Control Functions Use Aggregate Objects

Each Version Control function uses aggregate objects somewhat differently. The following list shows different ways:

- For details on the changes to related objects that cause aggregate objects to be marked as changed, see Change Capture for Version Control.
- For migration details related to component objects, companion objects, and enabling objects, see Aggregate Object Expansions and Special Cases.
- For details on the process of determining which component objects are adopted along with an aggregate object, see Adoption Rules.

For a list of subordinate aggregate objects used by Compare Report, see Become Familiar with Subordinate Aggregate Object Types.

Chapter 2: Comparing Objects in Different Models

This chapter includes a Compare Report overview and addresses questions you have about the Compare Report.

FAQs About Compare Report

The following are the Frequently Asked Questions (FAQs) and their short answers about Compare Report functionality:

- What is the purpose of compare report? To compare objects in two models on the same encyclopedia and report which objects are the same, which objects are in one model but not the other, and which objects are different versions
- How is equivalence between objects determined? By testing for common ancestry and for applicability of Special Equivalency Rules
- How is version difference between equivalent objects detected? By comparing the timestamps of the session objects that were associated with each equivalent object during the last change session
- How are object differences reported? This question is answered with an annotated mock-up of the report format
- Why would I need version control? It is a very convenient way to compare all objects or selected objects in different models
- When would I use version control? Before performing any other Version Control function on the selected models
- How do I use compare report results? Analyze the results on a case-by-case basis

What Is the Purpose of Compare Report?

Compare Report is a Version Control function that compares all aggregate objects or selected aggregate objects and their subordinate aggregate objects in two models in the same Client/Server Encyclopedia (CSE) and reports. The following is a list of objects for which Compare Report function can be used:

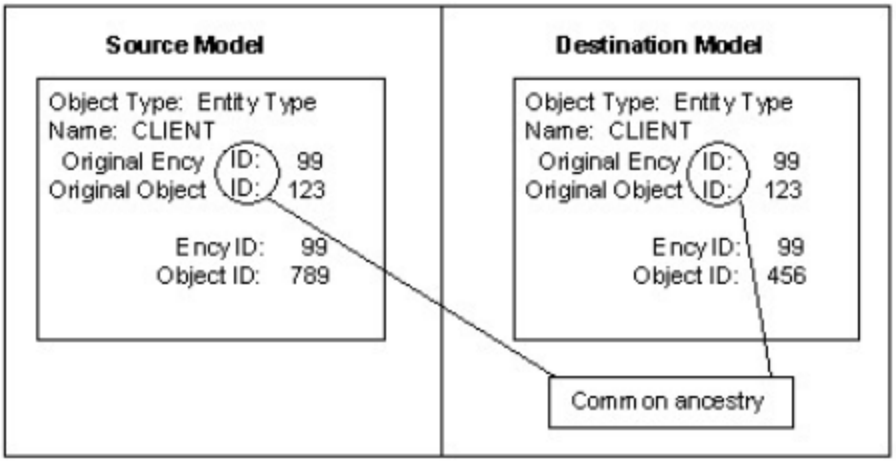
- Objects that exist in one model but not the other, that is an object for which no equivalent object exists in the other model.
- Equivalent objects that exist in both models, but in different versions.
- Equivalent objects that exist in the same version in both models.

How Is Equivalence Between Objects Determined?

An object in the destination model is equivalent to an object in the source model if it has the same Original Encyclopedia ID and the same Original Object ID, or if there is a Special Equivalency Rule for that object type, and the object in question meets the rule.

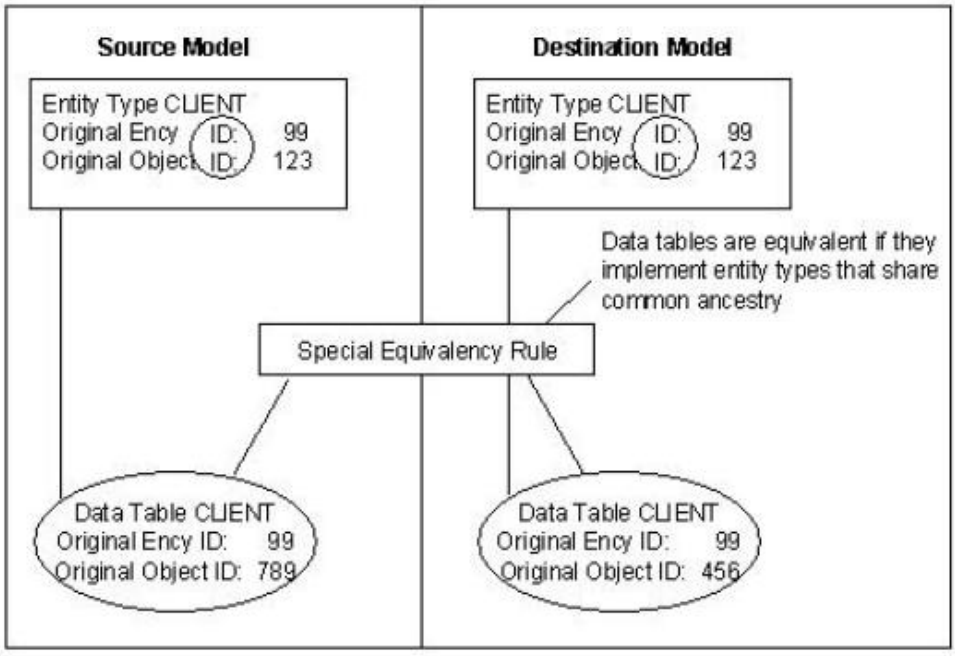
The combination of an object's Original Encyclopedia ID and Original Object ID, which is unique within a model, defines that object's ancestry. If two aggregate objects in different models have the same Original Encyclopedia ID and Original Object ID, they share common ancestry. The two aggregate objects are considered versions of the same object.

The following figure illustrates equivalence between two objects, due to common ancestry. The entity type, CLIENT, in the source model shares common ancestry with the entity type, CLIENT, in the destination model. Equivalence is established by matching Original Encyclopedia IDs and Original Object IDs:



The figure, Object Equivalence Through Special Equivalency Rule, illustrates equivalence between objects that meet the special equivalency rule for data tables. The data table, CLIENT, in the source model is equivalent to data table, CLIENT, in the destination model. Equivalence between the data tables is established by the Special Equivalency Rule for data tables, which states that data tables are equivalent if they implement entity types that share common ancestry.

Note: You can find the other object types to which a special equivalency rule applies in the table Special Equivalency Rules. The rule for each object type is also provided.



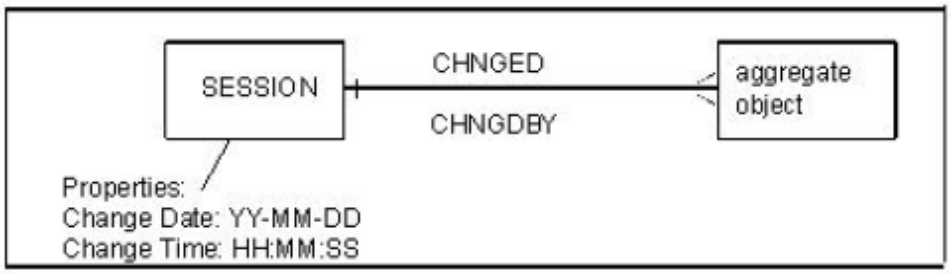
How Is Difference Between Equivalent Objects Detected?

Equivalent objects are different if the session in which they were last changed is different. When any process changes an object, that object is associated with a session object that includes the date and the time of change to the second. Compare Report evaluates the date and time of change. If a change to an object is made in one model, the date and time of its session object will be different than the date and time of the session object associated with the equivalent object in the other model; thus the objects will compare as different.

Under certain circumstances, objects can be reported as different when they are not actually different. For example, if identical changes to an object are made independently in each model, the date and time associated with the session object will be different and the objects will compare as different.

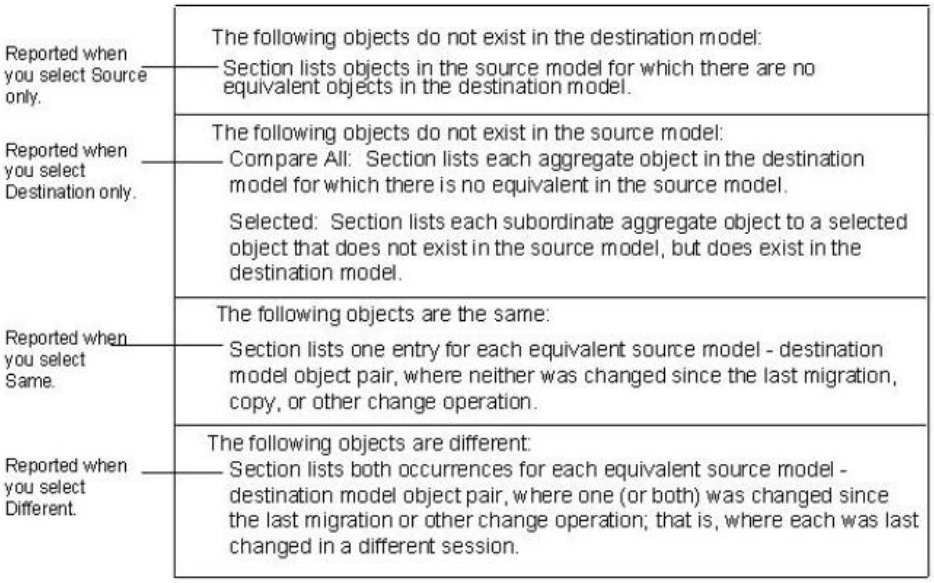
An aggregate object can be associated with many different sessions. However, it is associated with only one session at a time via the CHNGED/CHNGDBY association—namely, the session that reflects the last update to the object with respect to Version Control.

The following figure illustrates the type of association formed between a changed aggregate object and the session object:



How Are Object Differences Reported?

The differences that are reported depend on the report sections you select to include. When you select sections to report, the options are: Source only, Destination only, Same, Different, and All. The following figure shows the mapping of these selections to the headers on the four report sections. Note that All would apply to all sections:



If you compare selected objects, each object you select from the source model and each of its components are reported in one of three ways:

- As being the same version as an equivalent object in the destination model,
- As being a different version from an equivalent object in the destination model, or
- As not existing in the destination model.

If you select Destination only, this additional the following information is reported:

- If you compare selected objects and any selected objects have subordinates that exist in the destination model but do not exist in the model from which you select objects, they are reported as not existing in the source model.
- If you compare all objects, all aggregate objects in the destination model for which there are no equivalent objects in the source model are reported as not existing in the source model.

Why Would I Need Version Control?

There are a number of situations where you can get the information you need from the Compare Aggregate Object Report. For example, run the Compare Report for the following results:

- To determine whether model-level adoption is indicated, see the figure in Sample Compare Report for Adoption in the chapter "Performing Adoption."
- To identify objects to consider for migration. See the figure, Compare Report That Identifies Objects to Consider for Migration.
- To evaluate model versions after a successful migration. See the figure, Compare Report on Migrated Objects.
- To evaluate changes made in both models to the subordinate objects of a selected aggregate object. See the figure, Sample Compare Report on an Aggregate and Its Subordinate Objects.

When Would I Use Version Control?

The Compare Report is often the first step in planning a migration to copy new and changed objects from one model to another. The report identifies objects that have changed and objects that exist in one model but not the other. Based on the difference analysis information in the report, you can determine which objects to migrate. You also can be able to identify possible candidates for adoption by looking for objects with the same name listed as existing in the source model only and as existing in the destination model only.

There are special cases for using the Compare Report, and there are situations where you can use the Compare Report on a regular basis. The following are two such cases cited in terms of model management objectives:

If You Use Version Control To	Use the Compare Report To
Synchronize models used for parallel development	Evaluate changes and additions made to both development models so that you can plan a strategy for migration that captures all enhancements without loss of any objects you want to retain.
Maintain models of the same application in different stages, such as development and test	Limit migration from test to development to only what has been added or changed since the initial copy or the last migration.

How Do I Use Compare Report Results?

There are certain situations where synchronizing object versions is needed; in this case, the list of objects reported as different can be used as the basis of selecting objects for migration. If you create an aggregate set of objects listed as different, you will save object selection time when performing a series of Trial Migrations. There are other situations where you can want to retain differences between equivalent objects in different models; for example, if the models represent different releases of an application. For recommendations on when to consider migration and other alternatives, see Evaluate Report Data to Determine the Next Step.

Preparing for the Compare Report

Generating a Compare Report on the models of interest is usually the first task you will perform with the CSE Version Control Client.

Consider Requirements and Recommendations

You can either verify model and authorization requirements before you begin or you can simply proceed and take action only if you receive a message on an unmet requirement. For details on model requirements and recommendations, see Getting Ready.

Scheduling Considerations

Scheduling a migration can be necessary if:

- The migration is very large.

- The destination model is checked out where objects affected by migration have protection levels that prevent the necessary referencing, deleting, or modification to be completed.

Large migrations are resource-intensive; scheduling considerations can be an issue. It is recommended that you perform small migrations to reduce the resources needed.

Object Protection Considerations

Consideration of protections is applicable only if the destination model or any of its subsets are checked out when you perform a migration. For information on determining checkout status, see Evaluating the Models' Checkout Status. Object-level protection applies to aggregate objects equivalent to those you select for migration. It also extends to objects related to the aggregate object.

For the migration rules based on object protections and an illustrative example, see Attempt To Reference/Modify/Delete Object(s) Which Are Checked Out.

Determine Whether to Compare All or Compare Selected Objects

When you make your report selections on the Compare Options dialog, you will notice the Object Range option. This lets you specify whether to compare all objects in both models, or to compare aggregate objects you select from a selection list.

If you are using Version Control for the first time on a model, you can want to use the compare all objects option. The advantage of comparing All objects is that you can be sure the report will be comprehensive. For additional considerations, see the table, Comparing Advantages of All and Selected in Selecting the Object Range.

When using Compare All, it is not necessary to become familiar with aggregate object types before you run the report for the first time. It is reasonable to defer this until you begin analyzing the report results and planning the objects to select for the aggregate set to use for your next function. Then you can focus on only the aggregate object types you have in your model. For details on aggregate object types for Compare Report, see Become Familiar with Aggregate Object Types.

Identify Models as Source and Destination

When you select Compare All, it does not matter which model you specify as the source and destination. The information you can derive from the report is the same.

When you select aggregate objects to compare, consider the following tips:

- If changes have been made to only one of the models, specify this model as the source model.

- If the models are two development models that are being worked on in parallel and the report is limited to selected objects, consider running the report twice. Select one model as the source one time and select the other model as the source the next time.
- If you are aware of all new or changed objects in the source model, you can not need to run the Compare Report; you can have all the information you need to select objects for migration.

Determine the Sections to Report

Choose any combination of the options listed in the following table for the report. The following table describes all the options:

Section Option	Meaning of Option
All	Same as selecting all these options: -Source only -Destination only -Same -Different **Note:** The individual options are disabled when you select All, since they are included automatically with All.
Source Only	Lists each source model object for which there is no equivalent object in the destination model.
Destination Only	If comparing selected objects from the source, lists the subordinate aggregate objects of equivalent objects in the destination model that have no equivalent in the source model. If comparing all, lists all objects in the destination model that are not in the source model.
Same	Lists objects that exist in the same version in both source model and destination model.
Different	Lists objects that exist in both source and destination models, but in different versions.

Performing Object Comparison

Use the following procedure to compare objects in different models and generate a Compare Aggregate Object Report.

Note: For syntax on the command line alternative, enter COMPMODL from the directory where the Version Control client is installed.

How to Generate the Compare Aggregate Object Report

This section describes the various procedures to be followed for generating the Compare Object Report.

Access the Compare Function

Follow these steps:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Compare Report option. Model, Reports then Compare.
3. Select the source model and destination model. For considerations, see Determining Which Model is the Source Model.
    a. (Optional). Specify a filter for the Source Model names as described in Using a Filter.
    b. From the Source Model list, select the model that contains the objects you intend to compare.
    c. (Optional). Specify a filter for the Destination Model names.
    d. From the Destination Model list, select the model that contains the objects you intend to compare the source model objects to.
    e. Click OK.
4. Specify the Compare options. Press F1 for *Online Help* on any field.
    a. Specify whether to report on all objects or selected objects. For details, see Selecting the Object Range.
    b. If reporting on selected objects that have been defined to an aggregate set, check Aggregate Sets.
    c. Specify a report destination or accept the displayed default. For tips, see Specifying How to Save Reports.
    d. Specify a unique filename for the report or accept the displayed default.
    e. Specify the sections to include on the report or accept the default, All. For details, see Determine the Sections to Report.
    f. Click OK.
    g. If you selected Compare All, skip to the Compare Report Confirmation (Step 11). If you selected Aggregate Set, skip to Aggregate Sets (Step 9). Otherwise, continue.

5. Select an aggregate object type for which you want to list aggregate object occurrences. A description of each aggregate object type is listed in a table under Become Familiar with Aggregate Object Types.

   a. Highlight the aggregate object type of interest. You can find it helpful to select a higher-level aggregate object to the object of interest. This strategy will ensure that the report includes information on enabling objects to the object of interest. It will also ensure that the report includes information about related objects at the same level as the object of interest. For example, if the object of interest is an entity type, select instead its parent subject area. For a listing of the hierarchies, see Become Familiar with Aggregate Object Expansion in the chapter, "Migrating Objects from One Model to Another."

   b. Click List, or to limit the display to a specific range of occurrences of the selected aggregate object type, enter a filter value as described in Using a Filter and click List.

6. Select one occurrence, select multiple occurrences, or click Select All.

   Selecting one occurrence and expanding it displays the list of objects subordinate to the selected occurrence. This option is useful if you want to select objects of a low-level type having an extremely large number of occurrences within the model, but only a small number when accessed in this way.

   a. If you selected only one object and want to expand that object, click Expand and proceed with the Aggregate Object Expanded Occurrences dialog (Step 7).

   b. If you selected multiple individual occurrences, clicked Select All, or selected an object that cannot be expanded or that you do not want to expand, click Add, and then click Exit. The Aggregate Object Types List appears. Either continue the selection process as described in Step 5 or click Cancel and proceed to generate the report as described in Step 10 for the Selected Object List.

7. Select one occurrence, select multiple occurrences individually, or click Select All. Continue in one of the following ways:

   a. If you selected only one object and want to expand that object, click Expand again. Continue as described at the beginning of this step. (You can repeat this step of selecting and expanding until you reach the lowest-level object in the chain.)

   b. If you selected multiple objects, all objects, or a single object that either cannot be expanded or that you don't want to expand, click Add, then click Exit. If you performed only one expansion, the Aggregate Object Occurrences dialog with the parent object appears. If you performed multiple expansions, the Aggregate Object Expanded Occurrences dialog with the parent object appears. Click Exit repeatedly until you display the Aggregate Object Types dialog.

   c. There is no need to select a subordinate aggregate object if you have selected its parent, since it will be automatically processed when the parent object is fully expanded. This is the reason that selecting Add at the parent level is not recommended here; the assumption is that you made the selection at the highest level you need.

8. To continue selecting objects to compare, return to step 5. To generate the report with the objects currently selected, click Cancel.
9. Select the aggregate set(s) to use.
10. Verify that the listed objects are the objects to be compared. If so, click Proceed. (If not, see *Online Help* for other options.)

    Compare Report Confirmation
11. Click Yes to proceed with the Compare Report.

    Compare processing begins. A Compare progress indicator appears, updating the count of objects processed until the compare is complete.

    Compare Processing Messages
12. Review the following messages, then click Continue to generate the Compare Aggregate Object Report. The report appears in a Review panel. You can also access it under the specified (or default) filename from the specified (or default) directory.
    a. Request being forwarded to server
    b. Preparing the compare report
    c. Comparing … objects (displays incrementing object count)
    d. Compare complete

Evaluating the Compare Aggregate Object Report

This section begins with a scenario with the corresponding Compare Aggregate Object Report. Evaluating report data to determine what to do next is addressed last.

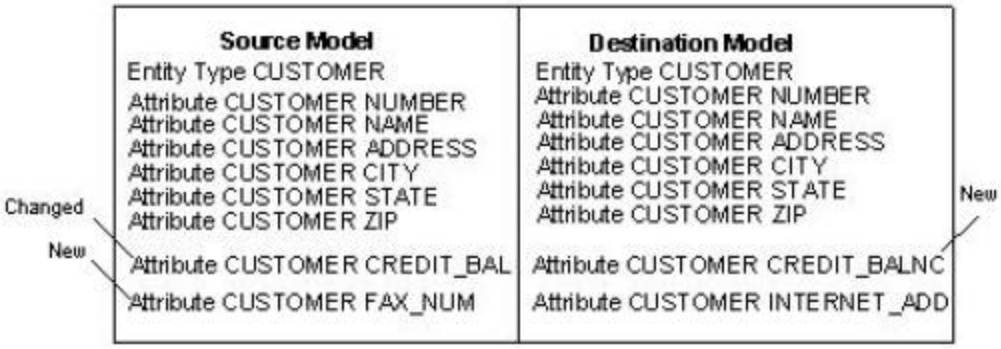
Scenario for Sample Compare Report

Consider two models used for parallel development, where the goal is to keep them synchronized. The current status is one of the following:

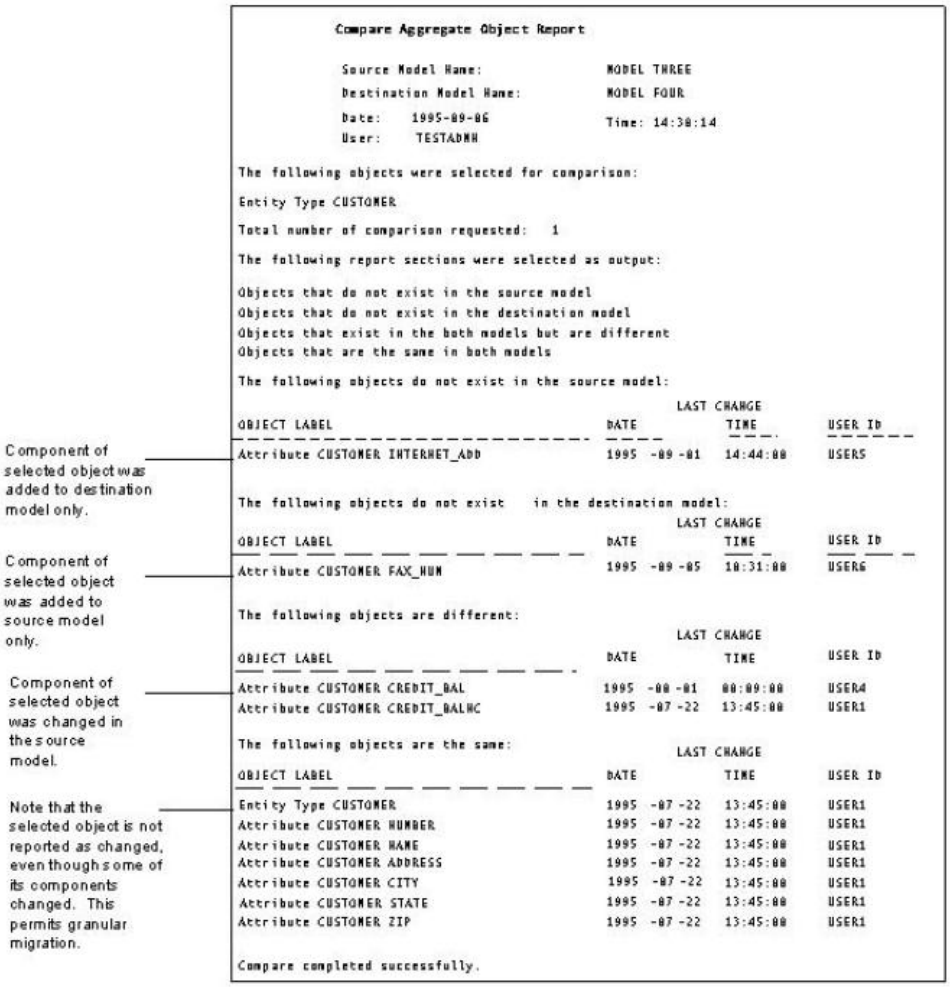
- Most objects are in the same version in both models
- Attribute CUSTOMER CREDIT_BAL was changed in the source model

- Attribute CUSTOMER FAX_NUM was added to the source model
- Attribute CUSTOMER INTERNET_ADD was added to the destination model

The following report is a sample view of the data displayed in each of the sections as evaluated by the Compare Report process:



When you report on a single high-level aggregate object, here the Entity Type, CUSTOMER, the Compare Report process reports on not only the selected object, but also on its subordinate aggregate objects. The following is sample Compare Report on an Aggregate and Its Subordinate Objects:



```
                Compare Aggregate Object Report

                 Source Model Name:              MODEL THREE
                 Destination Model Name:         MODEL FOUR
                 Date:    1995-09-06             Time: 14:30:14
                 User:    TESTADMN

The following objects were selected for comparison:

Entity Type CUSTOMER

Total number of comparison requested:   1

The following report sections were selected as output:

Objects that do not exist in the source model
Objects that do not exist in the destination model
Objects that exist in the both models but are different
Objects that are the same in both models

The following objects do not exist in the source model:

                                                        LAST CHANGE
OBJECT LABEL                                 DATE           TIME       USER ID
-------------------------------------------- ----           -----      --------
Attribute CUSTOMER INTERNET_ADD              1995 -09 -01   14:44:00   USER5


The following objects do not exist   in the destination model:

                                                        LAST CHANGE
OBJECT LABEL                                 DATE           TIME       USER ID
-------------------------------------------- ----           -----      --------
Attribute CUSTOMER FAX_NUM                   1995 -09 -05   10:31:00   USER6


The following objects are different:

                                                        LAST CHANGE
OBJECT LABEL                                 DATE           TIME       USER ID
-------------------------------------------- ----           -----      --------
Attribute CUSTOMER CREDIT_BAL                1995 -08 -01   00:00:00   USER4
Attribute CUSTOMER CREDIT_BALHC              1995 -07 -22   13:45:00   USER1


The following objects are the same:

                                                        LAST CHANGE
OBJECT LABEL                                 DATE           TIME       USER ID
-------------------------------------------- ----           -----      --------
Entity Type CUSTOMER                         1995 -07 -22   13:45:00   USER1
Attribute CUSTOMER NUMBER                    1995 -07 -22   13:45:00   USER1
Attribute CUSTOMER NAME                      1995 -07 -22   13:45:00   USER1
Attribute CUSTOMER ADDRESS                   1995 -07 -22   13:45:00   USER1
Attribute CUSTOMER CITY                      1995 -07 -22   13:45:00   USER1
Attribute CUSTOMER STATE                     1995 -07 -22   13:45:00   USER1
Attribute CUSTOMER ZIP                       1995 -07 -22   13:45:00   USER1

Compare completed successfully.
```

Component of selected object was added to destination model only.

Component of selected object was added to source model only.

Component of selected object was changed in the source model.

Note that the selected object is not reported as changed, even though some of its components changed. This permits granular migration.

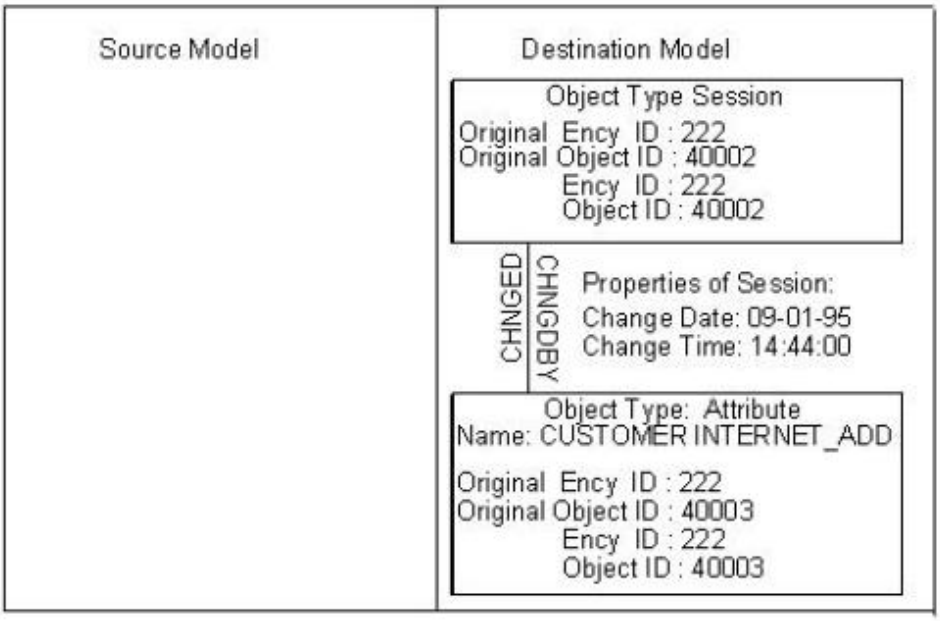
Compare Report's View of the Scenario

The Compare Report evaluates the following:

- The Original Encyclopedia ID and the Original Object ID of each reported object
- The change date and time properties of the session object associated with each reported object

Aggregate Object Listed as Not in Source Model

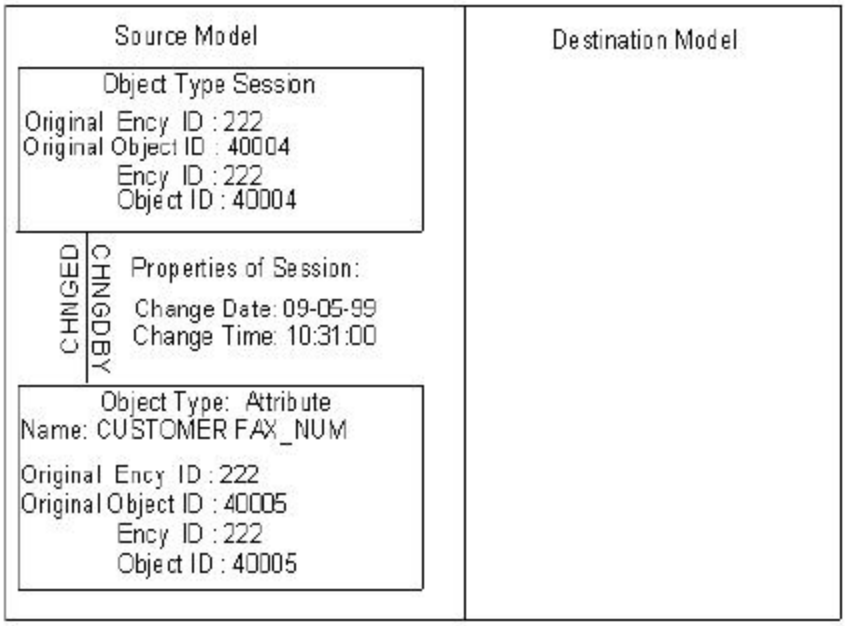
The aggregate object CUSTOMER INTERNET_ADD was added to the destination model. The following is a sample view:



The Compare Aggregate Object Report indicates that the attribute CUSTOMER INTERNET_ADD does not exist in the source model. This is because the source model contains no attribute that is a component of the selected entity type CUSTOMER that is equivalent to CUSTOMER INTERNET_ADD on the destination model. See the second figure in Evaluating the Compare Aggregate Object Report.

Aggregate Object Listed as Not in Destination Model

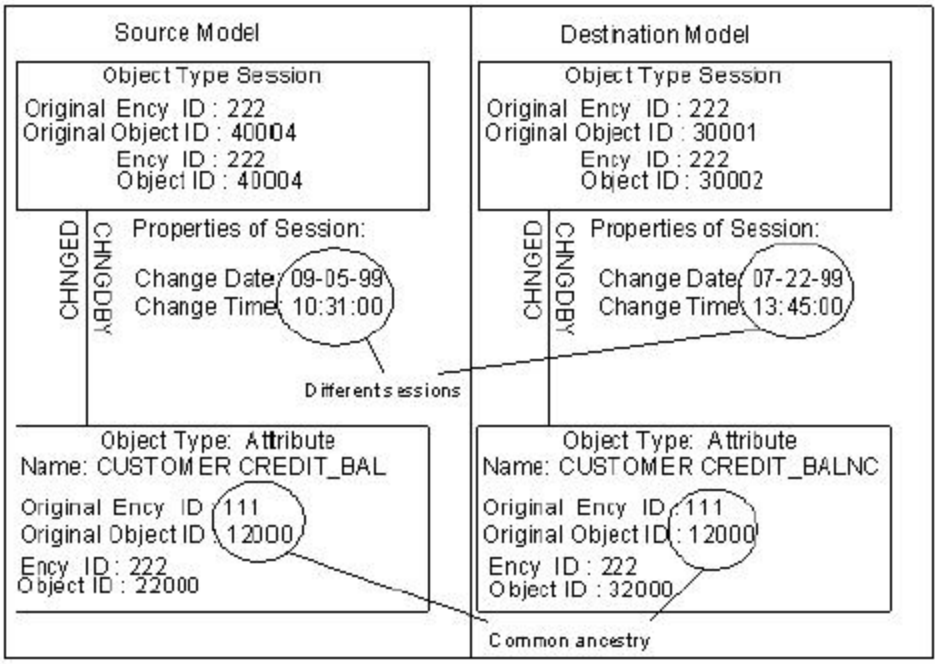
The aggregate object CUSTOMER FAX_NUM was created in the source model. It exists only in the source model; it does not exist in the destination model. The following is a sample view:



The Compare Aggregate Object Report indicates that the attribute CUSTOMER FAX_NUM does not exist in the destination model because no object in the destination model has the same Original Encyclopedia ID and Original Object ID. See the second figure in Evaluating the Compare Aggregate Object Report.

Aggregate Objects Listed as Different

In the source model, a property of attribute CUSTOMER CREDIT_BAL is updated. If an object's property is updated and it is an aggregate object, that object is marked as changed. The following is a sample view:

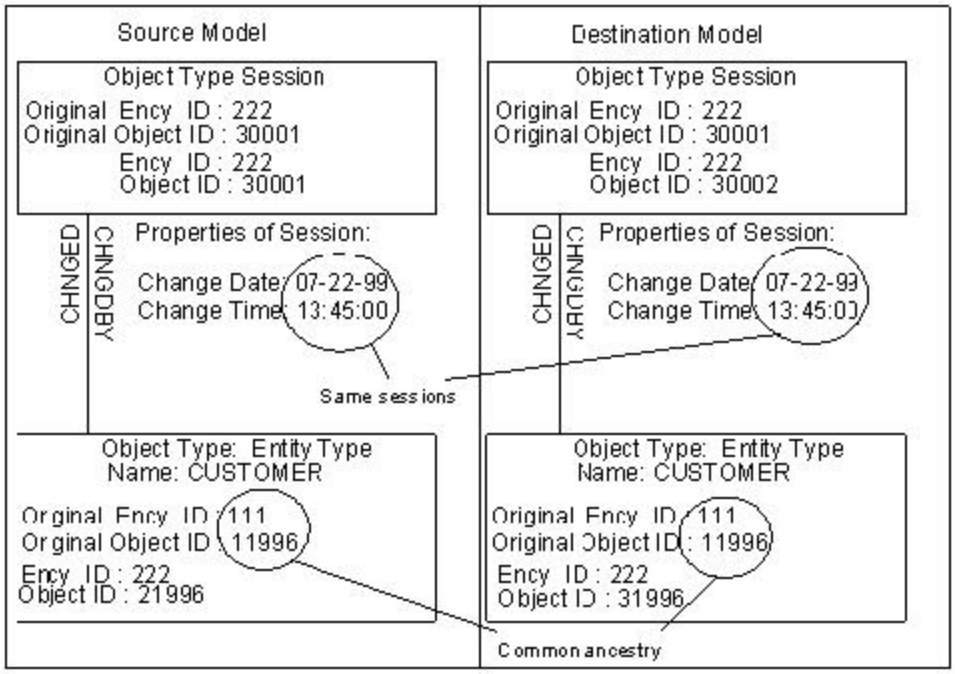


The Compare Aggregate Objects Report indicates that attribute CUSTOMER CREDIT_BAL in the source model is different from attribute CUSTOMER CREDIT_BALNC in the destination model. The report shows both labels since they are different. Although the attributes are equivalent, their Session Change Date and Change Time are different. See the figure, Sample Compare Report on an Aggregate and Its Subordinate Objects.

Note: When you change the property of an object, the object or its parent aggregate object is marked as changed. The parent aggregate object is marked if the changed object is not an aggregate object.

Aggregate Objects Listed as the Same

Assume that the destination model was created by copying the source model and that the entity type CUSTOMER existed in the source model at the time of the copy. Subsequent migrations could have occurred. The following is a sample view:



The Compare Aggregate Object Report indicates that entity type CUSTOMER as well as several of its attributes are the same in the source and destination models. The additions and changes were marked at the attribute rather than the entity type level, since attribute is an aggregate object. See the figure, Sample Compare Report on an Aggregate and Its Subordinate Objects.

Note: Compare Report uses the Change Date and Change Time of the session object to determine whether equivalent objects are the same version. If equivalent objects were last changed in the same session, the objects are reported as being the same.

Evaluate Report Data to Determine the Next Step

For each difference reported, you determine to do one of the following:

- Migrate the object
- Adopt the object, then migrate it
- Accept the difference, as is
- Continue analysis

Migrate Objects

Migrating objects enables you to maintain the same version of like objects across models. Migration either replaces an equivalent object, if it exists in the destination model, or creates a new instance of the object in the destination model:

- For each object pair reported as different, evaluate the content, then:
  - If one version is acceptable as is, consider making the objects the same in both models by migrating the preferred version to the model containing the version to be replaced.
  - If neither version is acceptable as is because each has unique component objects that you want to retain, consider migrating one object's unique components to the other model, then migrating the updated aggregate object back.
  - If both versions are acceptable as is because the same updates were made to each at different times, consider migration to prevent future reporting of a difference.
- For each object reported in the section, the following objects do not exist in the source model; consider adding that object to the source model through migration.
- For each object reported in the section, the following objects do not exist in the destination model; consider adding that object to the destination model through migration.
- Aggregate objects listed in the section, the following objects are the same do not require migration. If such objects exist in the aggregate set to be used for migration, they should be removed.

For report example reports listing possible candidates for migration, see the following:

- Compare Report That Identifies Objects to Consider for Migration.
- Sample Compare Report on an Aggregate and Its Subordinate Objects.

Adopt Objects

Adoption is used to create common ancestry between two objects in different models that are to be synchronized through Migration. For each instance of the same object name reported as both not existing in the source model and as not existing in the destination model, evaluate; if the objects were designed for the same purpose, that is, if they are logically the same, perform an adoption, then migrate. Create an aggregate set to use for both functions.

For reports listing possible candidates for adoption, see the following sample reports under What Reports Can Help Me?

- Trial Adopt Model Report
- Compare Report That Suggests Model-Level Adoption
- Trial Migrate Report that Identifies Isolated Objects to Consider for Adoption

Accept the Difference, As Is

Consider making no changes for each object pair reported when both versions are acceptable as is, where one is an enhanced version of the other, and the intent is to maintain them separately.

Continue Analysis

To pursue visual inspection of items of interest, do one of the following:

- Checking out a subset from each model and reviewing the pertinent information from the toolset.

Defining reports containing the applicable information using the CSE Public Interface views.

Chapter 3: Creating and Maintaining Aggregate Sets

Using aggregate sets saves you the time needed to reselect the same objects for one operation that you previously selected for another.

Preparing to Create an Aggregate Set

You can use aggregate sets with any version control function.

Consider Requirements

The encyclopedia administrator can take any action on an aggregate set. Minimal requirements for actions are listed in the following table:

Aggregate Set Action	Encyclopedia Access	Model Authorization
Add, copy, detail aggregate set	Authorization	Read
Change administrator, delete, modify, or rename aggregate set	Set administrator	Read

Become Familiar with Aggregate Object Types

Aggregate Object Types are the same for Compare Report, Trial Migration and Migration, but different for Trial Adoption and Adoption. The following lists include those for Compare Report, Trial Migration, and Migration:

Aggregate Object Type	Description
ACTION BLOCK	Common, Default, and Derivation, PAD, or PrAD
ACTIVITY CLUSTER	Activity cluster
ATTRIBUTE	Attribute or foreign key attribute of entity type or subtype
BATCH JOB	Batch job

Aggregate Object Type	Description
BATCH JOB STEP	Batch job step
BUSINESS AREA	Business area
BUSINESS SYSTEM	Business system
BUSINESS SYSTEM IMPL	Business system implementation
COMMAND	Command
COMPONENT IMPLEMENTATION	Component implementation type
COMPONENT MODEL	Component model
COMPONENT SPECIFICATION	Component specification type
CONFIGURATION INSTANCE	Configuration Instance
CONSTRAINT	Linkage between tables
CRIT SUCCESS FACTOR	Critical success factor
CURRENT DATA STORE	Current data store
CURRENT INFO SYSTEM	Current information system
CUSTOM PROXIES	Custom Proxies
CUSTOM VIDEO PROPERTY	Custom video property
DATA CLUSTER	Data cluster
DATA COLUMN	Data column definition
DATA TABLE	Data table definition
DATABASE	Database definition
DEFAULT DIALOG BOX	Dialog
DEFAULT EDIT PATTERN	User default edit pattern
DENORMALIZED COLUMN	Denormalized column
DFLT PRIMARY WINDOW	Primary window within a procedure step
DIALECT	Dialect
DIALOG FLOW	Dialog flow between procedure steps
ENTITY TYPE	Entity type
ENVIRONMENT	Environment
EVENT	External event
EXIT STATE	Exit state
EXTERNAL OBJECT	External object

Aggregate Object Type	Description
FACILITY	Computing or communication facility
FOREIGN KEY COLUMN	Foreign key column
FUNCTION	Function definition
GOAL	Business planning goal
INDEX	Index definition
INFORMATION NEED	Information need
INTERFACE TYPE	Interface type
ISP MATRIX	Isp matrix (user- or system-defined)
LINK TABLE	Link table definition
LOCATION	Location of business assets
NAVIGATION DIAGRAM	Navigation diagram for window presentation
OBJECTIVE	Business planning objective
ONLINE LOAD MODULE	Online load module
OPERATIONS LIBRARY	Operations library
ORGANIZATIONAL UNIT	Non-root organizational unit
PERFORMANCE MEASURE	Performance measure
PKG FOR PSTEP	Load module packaging
PROCEDURE	Procedure
PROCEDURE STEP	Procedure step
PROCESS	Process definition
RELATIONSHIP MEMBER	Relationship membership
SCREEN	Screen
SCROLL AMOUNT VALUE	Scroll amount value
SERVER MANAGER	Server manager
SPECIFICATION TYPE	Specification type
STORAGE GROUP	Storage group
STRATEGY	Business strategy
SUBJECT AREA	Subject area
SYSTEM PF KEY	System program function key
SYSTEM WORK ATTR SET	System-defined work attribute set

Aggregate Object Type	Description
TABLESPACE	Tablespace
TACTIC	Business tactic
TECH DESIGN DEFAULT	Technical design default
TEMPLATE	Screen template
TRANS OPERATION	Transaction operation
TYPEMAP	Type map
USER DEF OBJ CLASS	User-defined generic object class
USER DEFINED OBJECT	User-defined generic object
WEB SERVICE DEFINITION	Web service definition
WINDOW LOAD MODULE	Window load module
WORK ATTRIBUTE	Work attribute of work attribute set
WORK ATTRIBUTE SET	Work attribute set
z/OS LIBRARY	z/OS library

Aggregate Sets created for Compare Report can be used for Adoption even if they contain aggregate object types not valid for Adoption; invalid object types are ignored.

The objects listed in the following table are not valid for Adoption and Trial Adoption. The following table shows aggregate object types available only for compare and migration:

Aggregate Object Types	Description
BATCH JOB	Batch job
BATCH JOB STEP	Batch job step
BUSINESS SYSTEM IMPL	Business system implementation
DEFAULT DIALOG	Dialog
DFLT PRIMARY WINDOW	Primary window within a procedure step
PKG FOR PSTEP	Load module packaging
SCREEN	Screen
SCROLL AMOUNT VALUE	Scroll amount value
SYSTEM PF KEY	System program function key
SYSTEM WORK ATTR SET	System-defined work attribute set

Aggregate Object Types	Description
TECH DESIGN DEFAULT	Technical design default

Additional differences between aggregate object types selectable for compare and migration and those selectable for adoption are shown on the following table. The following table shows similar aggregate object types for Version Control functions:

Aggregate Object Type for Compare and Migration	Similar Aggregate Object Type for Adoption
ACTION BLOCK	COMMON ACTION BLOCK
DIALECT	DIALECT (NON-DFLT)
FUNCTION	FUNCTION (NON-ROOT)
ORGANIZATIONAL UNIT	ORG UNIT (NON-ROOT)
ISP MATRIX	USER DEF MATRIX
SUBJECT AREA	USER SUBJECT AREA

Become Familiar with Subordinate Aggregate Object Types

Certain aggregate object occurrences in a selection list can be expanded into subordinate aggregate objects. When you expand an object occurrence, the objects that are selectable are the first level subordinates to the expanded object. The maximum number of expansions down a particular subordinate hierarchy chain depends on the aggregate object. The hierarchy of subordinates can help you navigate through the selection dialogs.

If you select an object that has subordinates, Compare Report compares the selected object and if an equivalent is found, compares the subordinates. The following table lists the aggregate objects for Compare Report that have subordinate aggregate objects:

Aggregate Object Types	Subordinate Aggregate Object Types
ATTRIBUTE	Default or derivation algorithm ACTION BLOCKS
BATCH JOB	BATCH JOB STEP
BUSINESS SYSTEM	BUSINESS SYSTEM IMPL COMMAND DEFAULT EDIT PATTERN EXIT STATE PROCEDURE SYSTEM PF KEY TEMPLATE

Aggregate Object Types	Subordinate Aggregate Object Types
Configuration Instance	
DATABASE	TABLESPACE
DATA TABLE	DATA COLUMN DENORMALIZED COLUMN INDEX FOREIGN KEY COLUMN "From" CONSTRAINT LINK TABLE
ENTITY TYPE	ATTRIBUTE RELATIONSHIP MEMBERSHIP
FUNCTION	Child FUNCTION PROCESS
LINK TABLE	INDEX FOREIGN KEY COLUMN "From" CONSTRAINT
ORGANIZATIONAL UNIT	Child ORGANIZATIONAL UNIT
PROCEDURE	BATCH JOB PROCEDURE STEP
PROCEDURE STEP	ACTION BLOCK BATCH JOB STEP DEFAULT DIALOG DFLT PRIMARY WINDOW DIALOG FLOW PKG FOR PSTEP SCREEN
PROCESS	ACTION BLOCK Child PROCESS
SUBJECT AREA	Child SUBJECT AREA ENTITY TYPE
TABLE SPACE	DATA TABLE
USER DEF OBJ CLASS	USER DEFINED OBJECT
WORK ATTRIBUTE SET	WORK ATTRIBUTE

Select Model Containing Objects for Aggregate Set

When creating an aggregate set to use for a Compare Report, Trial Migration, or Migration, select the same model that you plan to use as the source model. When creating an aggregate set for Adoption or Trial Adoption, select the same model that you plan to use as the destination model.

Creating an Aggregate Set

To create an aggregate set to be used for Compare Report, Trial Adoption, Adoption, Trial Migration, and/or Migration, use the following procedure.

Access the Aggregate Set Function

Follow these steps:

1. Access Version Control. See Starting the Version Control Client.

   If you log on with a group ID, all members of the group can edit, rename or delete the aggregate set you create or change the aggregate set's administrator.

2. Select the Aggregate Sets option, Aggr Sets.
3. Name the new aggregate set. Select Actions then New.
4. Type a unique name and click OK.
5. Select the model with the objects to be selected.

   a. (Optional). Specify a filter as described in Using a Filter.

   b. From the list, select the model that contains the objects you intend to select for the set.

      For Compare Report, Trial Migration, or Migration, select the same model you will select as the source model. For Trial Adoption or Adoption, select the same model you will select as the destination model.

   c. Click OK.

   Select an aggregate object type for which you want to list aggregate object occurrences. See the tables in Become Familiar with Aggregate Object Types.

   Use the following procedure to highlight the aggregate object type of interest.

d. You can find it helpful to select a higher-level aggregate object to the object of interest. This strategy will ensure that the report includes information on enabling objects to the object of interest. It will also ensure that the report includes information about related objects at the same level as the object of interest. For example, if the object of interest is an entity type, select instead its parent subject area. For a listing of the hierarchies, see one of the following in Become Familiar with How Aggregate Objects Expand in Preparing for Migration:

   – Subordinates of Subject Area, Entity Type, and Attribute

   – Subordinates of Database, Tablespace, Data Table, and Link Table

   – Subordinates of Business System, Procedure, Batch Job, and Procedure step

   – Subordinates of Function, Child Function, and Process

   – Subordinate of Organizational Unit

   – Subordinate of User Class

   – Subordinate of Work Attribute Set

   Select the lowest–level aggregate object possible to migrate. This strategy will reduce the number of objects that have to be migrated, as well as optimizing performance. Use the Compare Aggregate Objects Report to identify objects that have changed. Use the Trial Migration Report to determine the anticipated outcome of migrating the objects of interest. Use the expansion tables to identify component objects. If selecting a parent aggregate object, it is not necessary to select any of its component objects, since components are migrated automatically.

e. Click List, or to limit the display to a specific range of occurrences of the selected aggregate object type, enter a filter value as described in Using a Filter and click List.

6. Select one aggregate object occurrence, select multiple occurrences, or click Select All. For use with Compare Report, Trial Migration, or Migration, selecting one occurrence and expanding it displays the list of objects subordinate to the selected occurrence. This option is useful if you want to select objects of a low-level type having an extremely large number of occurrences within the model, but only a small number when accessed in this way.

   - If you selected only one object and want to expand that object, click Expand and proceed with the Aggregate Object Expanded Occurrences dialog (step 7).
   - If you selected multiple individual occurrences, clicked Select All, or selected an object that cannot be expanded or that you do not want to expand, click Add, then click Exit. The Aggregate Object Types List appears. Either continue the selection process as described in step 5 or click Cancel and proceed to generate the report as described in step 9 for the Selected Object List.

   To identify object types that require other objects to be adopted in the same session, or previously adopted, see the guidelines under Example Where Object in Adopt Selected Depends on Another. If you are adopting all model objects in multiple sessions, list each aggregate object type in the recommended sequence, then use Select All to select all occurrences for each aggregate object type that has occurrences.

7. Select one occurrence, select multiple occurrences individually, or click Select All. Continue in one of the following ways:

   - If you selected only one object and want to expand that object, click Expand again. Continue as described at the beginning of this step. (You can repeat this step of selecting and expanding until you reach the lowest-level object in the chain.)
   - If you selected multiple objects, all objects, or a single object that either cannot be expanded or that you don't want to expand, click Add, then click Exit. If you performed only one expansion, the Aggregate Object Occurrences dialog with the parent object appears. If you performed multiple expansions, the Aggregate Object Expanded Occurrences dialog with the parent object appears. Click Exit repeatedly until you display the Aggregate Object Types dialog.

   There is no need to select a subordinate aggregate object if you have selected its parent, since it will be automatically processed when the parent object is fully expanded. This is the reason that selecting Add at the parent level is not recommended here; the assumption is that you made the selection at the highest level you need.

   Try to migrate at the lowest level possible. For example, do not migrate an entity type when only an attribute has changed; migrate only the changed attribute.

8. To continue selecting objects, return to step 5.
9. To end the selection process, click Cancel until you display the Selected Object List.
10. Verify that the listed objects are the objects to be selected. If so, select Actions, Save then Exit. (If not, see help for other options.)

Editing an Aggregate Set

You can edit an aggregate set to make changes in its contents or to add objects from the same or a different model. You would edit an aggregate set in one of the following situations:

- Preparing for Trial Migration after creating an aggregate set for Compare. When you create an aggregate set for comparison purposes, you generally select objects at a high level to ensure you get all needed information on subordinate aggregate objects. When you prepare for Trial Migrate, you select objects at the lowest level possible. You can edit a set to remove high-level objects and add the needed low-level objects when moving from the comparison phase to the migration phase of your synchronization plan.
- Preparing for Migration after evaluating Trial Migrate Report. You can need to add objects based on messages produced during Trial Migration. For example, the message object <requires> object identifies an object you need to add to the aggregate set for the next Trial Migration.
- Creating a set to be used for a model that is composed of objects from two or more other models. You can create the set with one of the source models then edit the set to add objects from the other source model.

Access the Aggregate Set Function

Follow these steps:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Aggregate Sets option, Aggr Sets.
3. (Optional.) Filter the displayed list of aggregate sets by aggregate set name or by administrator.
   a. Accept the default, Name, or select Administrator.
   b. Enter a character or character string for the selected filter type and press Tab.
4. Highlight the aggregate set to edit and select Actions then Edit.
5. Select the model containing the objects to be selected.
   a. (Optional). Specify a filter as described in Using a Filter. From the list, select the model that contains the objects you intend to select for the set. Normally, you will select the model used to create the aggregate set. However, you can select a different model.
   b. Click OK.
6. If you select a model that contains objects that belong to the set, those objects are retrieved. The Retrieval Status indicates the number retrieved. Click OK.

7. To remove any object(s) from the list, highlight the object(s) and select Actions then Remove.
8. To add new objects, proceed as you would if creating a new aggregate set.
9. When you finish editing the aggregate set, select Actions, Save then Exit.

Changing the Administrator for an Aggregate Set

Only the aggregate set administrator (or encyclopedia administrator) can change the administrator for the target set. The following are the steps for changing administrator:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Aggregate Sets option, Aggr Sets.
3. Highlight the aggregate set for which you want to change the administrator, and select Actions then Change Admin.
4. Select the user ID of the new administrator for the selected set and click OK.
5. To exit the Aggregate Sets function, select Actions then Cancel.

Copying an Aggregate Set

If you have already defined an aggregate set containing most or many of the objects you need for the current operation, it saves time to create a copy and modify it rather than create a new one from scratch. The following are the steps for copying an aggregate set:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Aggregate Sets option, Aggr Sets.
3. Highlight the aggregate set to copy from the displayed list, and select Actions then Copy.
4. Specify a new aggregate set name that is unique among the aggregate sets for this model and click OK.
5. To exit the Aggregate Sets function, select Actions then Cancel.

Renaming an Aggregate Set

Renaming can be used on demand or to indicate something generic. You can develop a naming convention to indicate the function where you plan to use the aggregate set next. For example, abccompr can be created for Compare Report, then renamed to abctadpt for use with Trial Adoption and modified to include occurrences to be tested for adoption readiness. After the Trial Adopt report has been run and changes made based on report analysis, the aggregate set could be renamed to abcadpt to indicate the next is for the adoption function. Subsequent renaming could be abctmigr to abcmigr to indicate readiness for Trial Migration or Migration. Alternatively, your convention can be to indicate the number of uses, tmigr1, tmigr2, tmigr3, and so forth.

Only the aggregate set administrator (or encyclopedia administrator) can rename an aggregate set.

Follow these steps:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Aggregate Sets option, Aggr Sets.
3. Highlight the aggregate set to rename from the displayed list, and select Actions then Rename.
4. Specify a new aggregate set name that is unique among the aggregate sets for this model and click OK.
5. To exit the Aggregate Sets function, select Actions then Cancel.

Deleting an Aggregate Set

When you are finished using an aggregate set, it is a good housekeeping practice to delete it. Only the aggregate set administrator (or encyclopedia administrator) can delete an aggregate set.

Follow these steps:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Aggregate Sets option, Aggr Sets.
3. Highlight the aggregate set you want to delete, and select Actions then Change Delete.
4. To confirm the deletion, click Yes.
5. To exit the Aggregate Sets function, select Actions then Cancel.

Detail an Aggregate Set

Any user with access to the encyclopedia can detail any aggregate set.

Follow these steps:

1. Access Version Control. See Starting the Version Control Client.
2. Select the Aggregate Sets option, Aggr Sets.
3. Highlight the aggregate set you want to detail, and select Actions then Detail.
4. When finished examining the details, click OK.

To exit the Aggregate Sets function, select Actions then Cancel.

Chapter 4: Performing Adoption

This chapter discusses the prerequisites and procedures required to perform adoption.

FAQs About Adoption

Read these sections for an Adoption overview that addresses basic questions you have about adoption.

The following are the Frequently Asked Questions (FAQs) and their short answers about Adoption functionality:

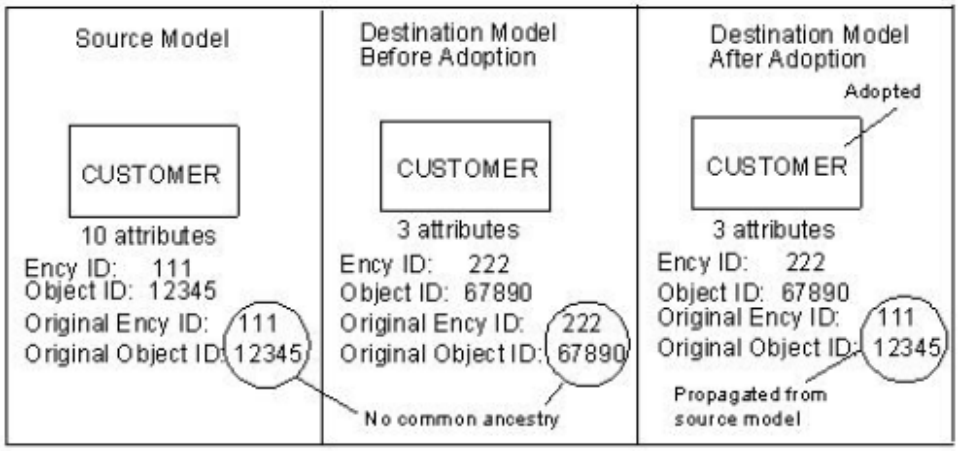
- What is adoption's purpose? To establish common ancestry between objects that are logically the same so they can be migrated
- Why would I need adoption? Equivalence was severed or never existed
- When would I use adoption? Before migration
- How do I identify objects to adopt? Look at the reports to identify candidates, then verify
- What reports can help me? Compare Aggregate Object Report, Trial Adopt Model Report, and Trial Migrate Aggregate Object Report can help you identify candidates for adoption; Adopt Report contains details on adoption outcome
- How can I avoid mistakes? Verify candidates are object versions
- How Does Adoption Work? Objects you select for adoption get the Original Encyclopedia ID and Original Object ID of the objects that are to replace them during migration

What Is Adoption's Purpose?

Adoption is the Version Control function that establishes common ancestry between objects in different models that are versions of the same object but are not recognized as equivalent objects by Migration or Compare Report. Before adoption, such objects cannot be correctly migrated; the Compare Report indicates they are unrelated. After adoption, such objects can be migrated successfully; the Compare Report indicates they are versions of the same object.

Adoption creates common ancestry between the objects that are logically the same by replacing the Original Encyclopedia ID and Original Object ID of the destination model object with the IDs belonging to the related object in the source model.

The following figure shows the adoption of CUSTOMER in the destination model. The Original Encyclopedia ID and Original Object ID of CUSTOMER in the destination model are changed to match the original IDs of the object in the source model that is logically the same. This is because two entity types are considered logically the same when they have the same name:



Why Would I Need Adoption?

Normally, you do not. Objects that are logically the same do not need to be adopted when they are already equivalent. When a new object is added to a model through migration, it is equivalent to the object on which it is based. When new objects are created in a model through model copy, each object is equivalent to the object on which it is based.

Adoption is performed to establish equivalence between objects that are logically the same, only when those objects are not already recognizable by Compare Report and Migration as equivalent. Lack of equivalency between objects that are logically the same can occur when:

- A model is uploaded from a tran file created by the download with upload option.
- Technical Design transformation is performed.
- Objects were created independently in different models for the same purpose.

When Would I Use Adoption?

When needed, adoption should be performed on an object before attempting to migrate it.

How Do I Identify Objects to Adopt?

Objects that require to be adopted by examining reports can be identified in at least three ways, which are listed as follows:

- You can identify objects that are candidates for adoption by requesting a Trial Adopt Model report that lists only objects that can be adopted. This is appropriate if you believe that only selected objects can need to be adopted.
- You can identify objects that are candidates for adoption by requesting a Compare Report on all objects that exist in the source model only and the destination model only, and assessing whether the lists are identical. This is the appropriate option if you believe a model-level adoption can be needed.
- You can identify objects that are candidates for adoption when you have no reason to suspect adoption is needed by examining the Trial Migrate Aggregate Object Report for renamed occurrences. Migration renames an object being migrated if another object with the same name already exists in the model. The object with the same name in this instance is different from the one the migration would replace.

Note: Regardless of the report, it is up to you to verify that these objects actually are versions of the same object. Reports only indicate objects can be candidates for adoption.

What Reports Can Help Me?

Successful adoption is illustrated by the figure in Sample Adopt Objects Report.

The following reports indicate how useful these types of reports can be. Each of the sample reports lists possible candidates for adoption in a different way.

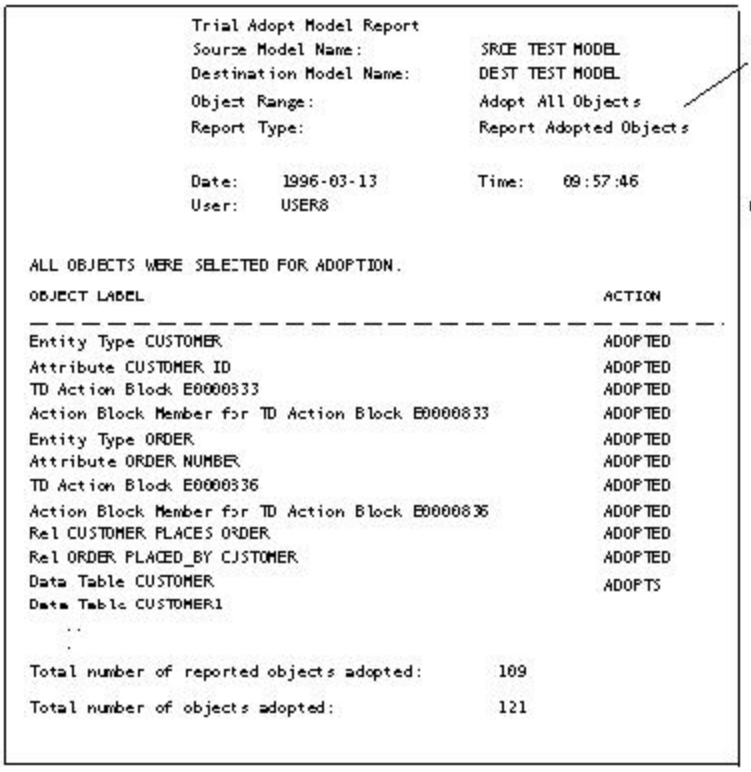
- Trial Adopt Model Report that Identifies Objects to Consider for Adoption
- Compare Report that Suggests Model-Level Adoption
- Trial Migrate Report that Identifies Isolated Objects to Consider for Adoption

All the above reports use the data shown in the following table:

Source Model	Destination Model
Entity Type CUSTOMER Attribute CUSTOMER ID	Entity Type CUSTOMER Attribute CUSTOMER ID
Entity Type ORDER Attribute ORDER NUMBER	Entity Type ORDER Attribute ORDER NUMBER
Rel CUSTOMER PLACES ORDER Rel ORDER PLACED_BY CUSTOMER	Rel CUSTOMER PLACES ORDER Rel ORDER PLACED_BY CUSTOMER

Sample Trial Adopt Model Report

One strategy for identifying objects to select for adopting is to report on objects that can be adopted. You can then use the set of objects reported as ADOPTED or ADOPTS to create an aggregate set to use when you perform the actual adoption.

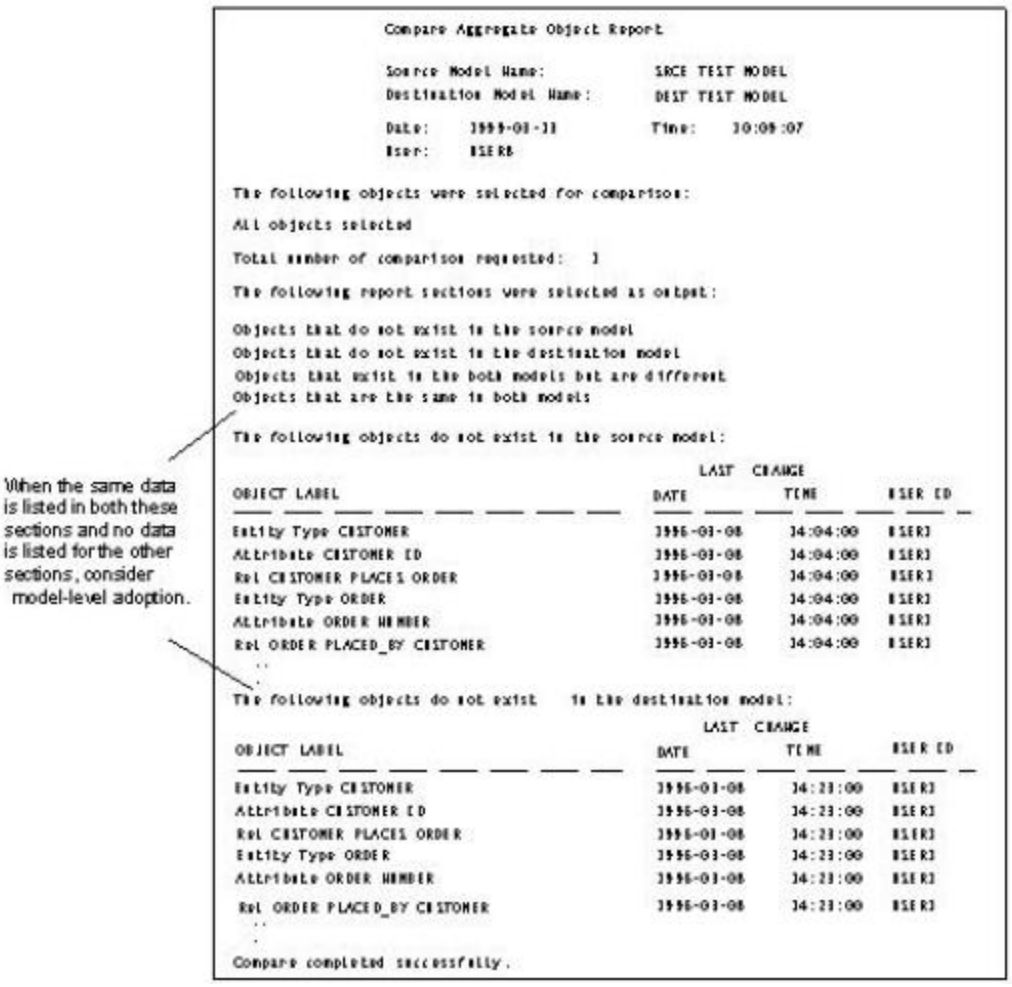


```
                Trial Adopt Model Report
                Source Model Name:          SRCE TEST MODEL
                Destination Model Name:     DEST TEST MODEL
                Object Range:               Adopt All Objects
                Report Type:                Report Adopted Objects

                Date:    1996-03-13         Time:    09:57:46
                User:    USER8

ALL OBJECTS WERE SELECTED FOR ADOPTION.

OBJECT LABEL                                                    ACTION
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Entity Type CUSTOMER                                            ADOPTED
Attribute CUSTOMER ID                                           ADOPTED
TD Action Block E0000333                                        ADOPTED
Action Block Member for TD Action Block E0000833                ADOPTED
Entity Type ORDER                                               ADOPTED
Attribute ORDER NUMBER                                          ADOPTED
TD Action Block E0000336                                        ADOPTED
Action Block Member for TD Action Block E0000836                ADOPTED
Rel CUSTOMER PLACES ORDER                                       ADOPTED
Rel ORDER PLACED_BY CUSTOMER                                    ADOPTED
Data Table CUSTOMER                                             ADOPTS
Data Table CUSTOMER1
   ..
   .
Total number of reported objects adopted:       109

Total number of objects adopted:                121
```

An easy way to identify candidates for adoption acoption is to run the Trial Adopt report with report options Adopt All and Report Adopted Objects.

Sample Compare Report Before Adoption

One strategy for determining whether model-level adoption can be indicated is to request a Compare Report on all objects that includes all report sections. Model-level adoption is suggested if the same data is listed in both the do not exist sections and no data is listed as same or different. Aggregate sets are not used for model-level adoption. The following is a sample view:

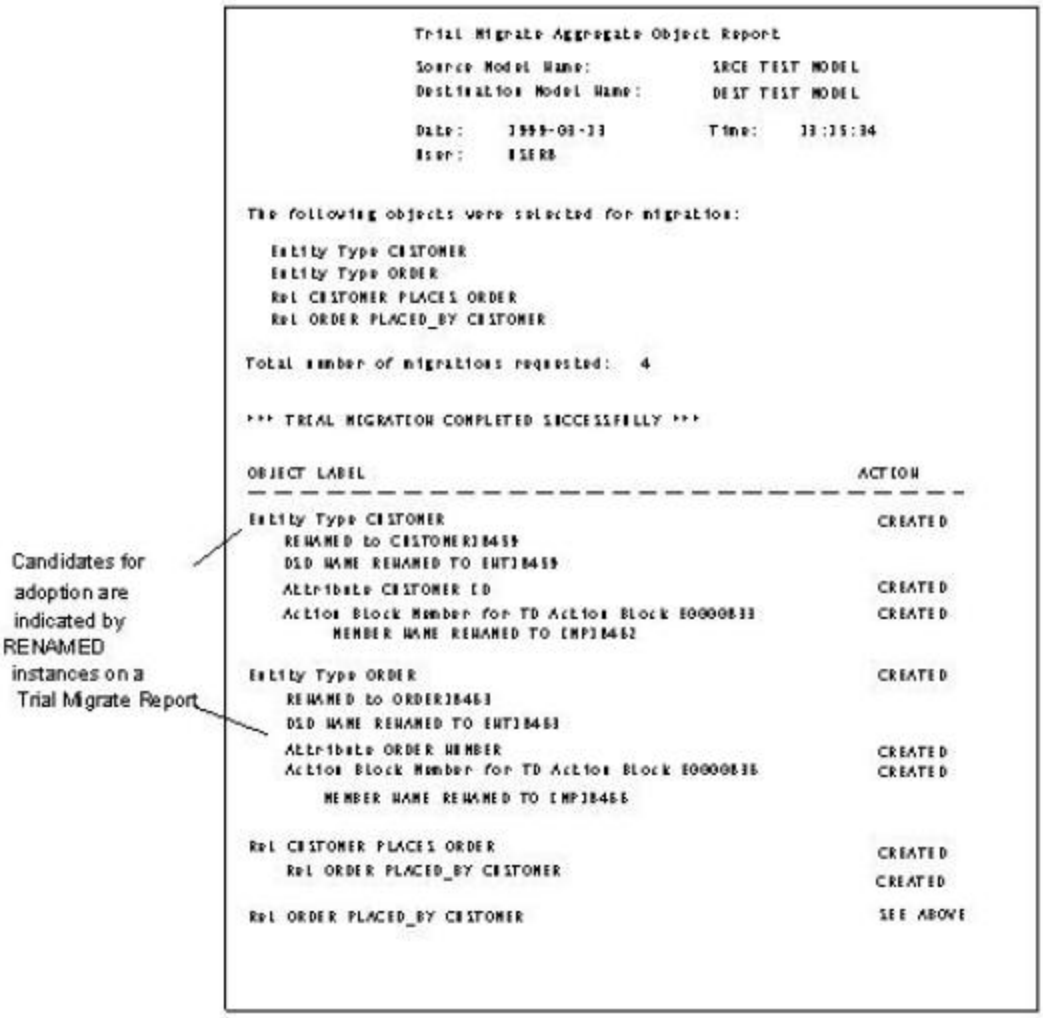


```
                Compare Aggregate Object Report

                Source Model Name:          SRCE TEST MODEL
                Destination Model Name:     DEST TEST MODEL

                Date:     1999-08-31        Time:     10:09:07
                User:     USER8

The following objects were selected for comparison:

All objects selected

Total number of comparison requested:   1

The following report sections were selected as output:

Objects that do not exist in the source model
Objects that do not exist in the destination model
Objects that exist in the both models but are different
Objects that are the same in both models

The following objects do not exist in the source model:

                                                    LAST  CHANGE
OBJECT LABEL                                  DATE          TIME        USER ID
--------------------------------------------  ------------  ----------  -------
Entity Type CUSTOMER                          1996-03-08    14:04:00    USER1
Attribute CUSTOMER ID                         1996-03-08    14:04:00    USER1
Rel CUSTOMER PLACES ORDER                     1996-03-08    14:04:00    USER1
Entity Type ORDER                             1996-03-08    14:04:00    USER1
Attribute ORDER NUMBER                        1996-03-08    14:04:00    USER1
Rel ORDER PLACED_BY CUSTOMER                  1996-03-08    14:04:00    USER1
  ..
  .
The following objects do not exist    in the destination model:
                                                    LAST  CHANGE
OBJECT LABEL                                  DATE          TIME        USER ID
--------------------------------------------  ------------  ----------  -------
Entity Type CUSTOMER                          1996-03-08    14:23:00    USER1
Attribute CUSTOMER ID                         1996-03-08    14:23:00    USER1
Rel CUSTOMER PLACES ORDER                     1996-03-08    14:23:00    USER1
Entity Type ORDER                             1996-03-08    14:23:00    USER1
Attribute ORDER NUMBER                        1996-03-08    14:23:00    USER1
Rel ORDER PLACED_BY CUSTOMER                  1996-03-08    14:23:00    USER1
  ..
  .
Compare completed successfully.
```

When the same data is listed in both these sections and no data is listed for the other sections, consider model-level adoption.

Sample Trial Migrate Aggregate Object Report

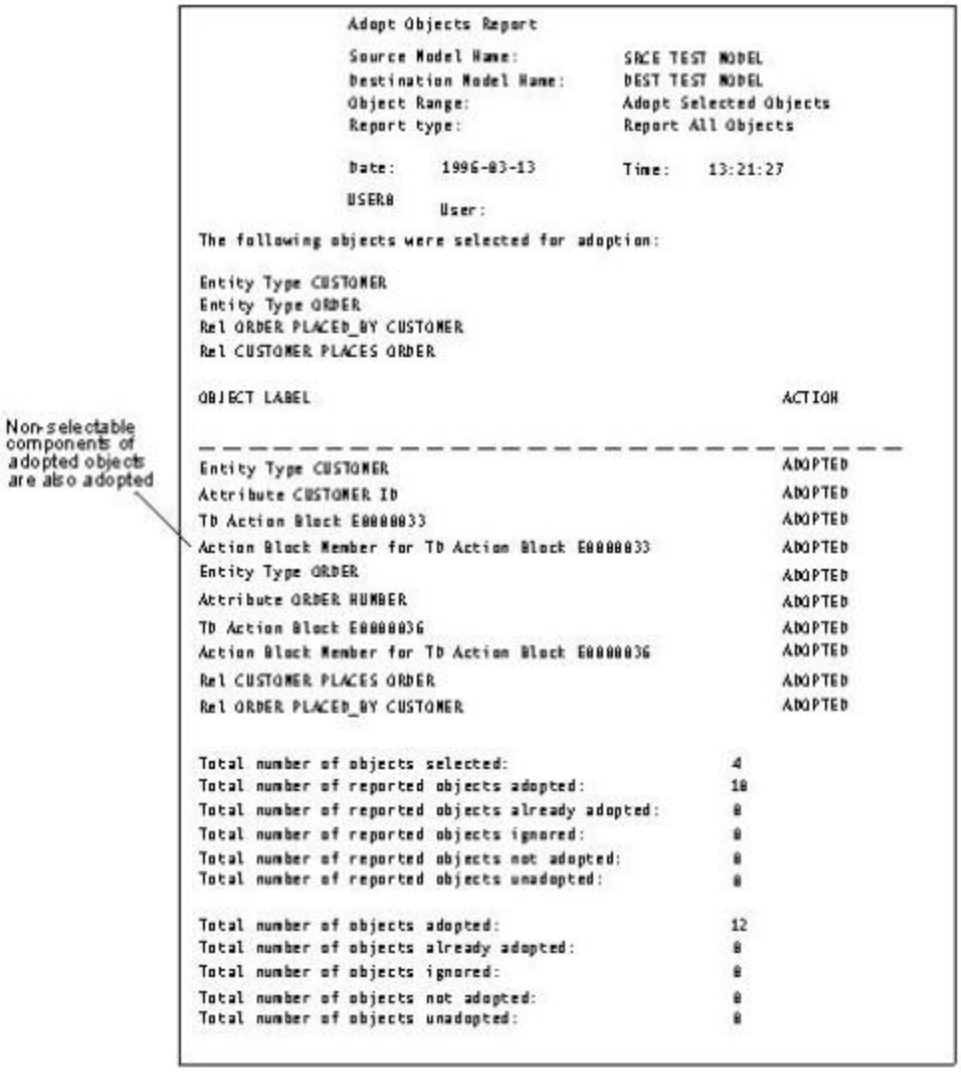
If you notice instances of RENAMED objects when you examine a Trial Migration report, examine the renamed objects to see if they are candidates for adoption. The following is a sample view:




```
                Trial Migrate Aggregate Object Report

                Source Model Name:          SRCE TEST MODEL
                Destination Model Name:     DEST TEST MODEL

                Date:     1999-08-13        Time:     13:15:34
                User:     USER8

The following objects were selected for migration:

   Entity Type CUSTOMER
   Entity Type ORDER
   Rel CUSTOMER PLACES ORDER
   Rel ORDER PLACED_BY CUSTOMER

Total number of migrations requested:  4

*** TRIAL MIGRATION COMPLETED SUCCESSFULLY ***

OBJECT LABEL                                                    ACTION
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Entity Type CUSTOMER                                            CREATED
    RENAMED to CUSTOMER18459
    OLD NAME RENAMED TO ENT18459
    Attribute CUSTOMER ID                                       CREATED
    Action Block Member for TD Action Block E0000833            CREATED
        MEMBER NAME RENAMED TO EMP18462

Entity Type ORDER                                               CREATED
    RENAMED to ORDER18463
    OLD NAME RENAMED TO ENT18463
    Attribute ORDER NUMBER                                      CREATED
    Action Block Member for TD Action Block E0000835            CREATED
        MEMBER NAME RENAMED TO EMP18466

Rel CUSTOMER PLACES ORDER                                       CREATED
    Rel ORDER PLACED_BY CUSTOMER                                CREATED

Rel ORDER PLACED_BY CUSTOMER                                    SEE ABOVE
```

Sample Adopt Objects Report

The following sample report shows the results of a successful adoption of selected objects:



```
                Adopt Objects Report

                Source Model Name:          SRCE TEST MODEL
                Destination Model Name:     DEST TEST MODEL
                Object Range:               Adopt Selected Objects
                Report type:                Report All Objects

                Date:     1996-03-13        Time:     13:21:27
                USER0     User:
The following objects were selected for adoption:

Entity Type CUSTOMER
Entity Type ORDER
Rel ORDER PLACED_BY CUSTOMER
Rel CUSTOMER PLACES ORDER

OBJECT LABEL                                                    ACTION

- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Entity Type CUSTOMER                                            ADOPTED
Attribute CUSTOMER ID                                           ADOPTED
TD Action Block E0000033                                        ADOPTED
Action Block Member for TD Action Block E0000033                ADOPTED
Entity Type ORDER                                               ADOPTED
Attribute ORDER NUMBER                                          ADOPTED
TD Action Block E0000036                                        ADOPTED
Action Block Member for TD Action Block E0000036                ADOPTED
Rel CUSTOMER PLACES ORDER                                       ADOPTED
Rel ORDER PLACED_BY CUSTOMER                                    ADOPTED

Total number of objects selected:                       4
Total number of reported objects adopted:               10
Total number of reported objects already adopted:       0
Total number of reported objects ignored:               0
Total number of reported objects not adopted:           0
Total number of reported objects unadopted:             0

Total number of objects adopted:                        12
Total number of objects already adopted:                0
Total number of objects ignored:                        0
Total number of objects not adopted:                    0
Total number of objects unadopted:                      0
```

Non-selectable components of adopted objects are also adopted

After adoption, the objects are recognized as equivalent. Compare this report with the Compare Report that Suggests Model-Level Adoption, where the same objects are not recognized as equivalent. This comparison points out that:

- Before adoption, objects are listed under the sections
- The objects do not exist in the source model
- The objects do not exist in the destination model

- After adoption, objects are listed under the section
- The objects are different

How Can I Avoid Mistakes?

Certain errors can be easily avoided by examining the Compare Aggregate Object Report. Potential adoption errors that can be detected by examining this report are described in the section, Identify Objects to Avoid Adopting. Other adoption errors can be easily prevented by running the Trial Adoption report, before attempting the adoption and correcting conditions causing actions such as NOT ADOPTED.

A mistake you could make that is not easily remedied is to adopt an object that is not really intended to be a version of the object with the same name in the other model. You cannot undo an adoption or selectively unadopt objects that have been inadvertently adopted. The reason you should take precautions to avoid such a mistake is that subsequent migration would replace one object with the other. To prevent this type of mistake, compare the definitions of the candidate objects with the definitions of the objects you believe to be logically the same, if you detect differences that indicate they are not versions of the same object, remove the name of the object from the list or aggregate set you plan to use as the basis for selection.

How Does Adoption Work?

After you identify those aggregate objects you plan to replace during a subsequent migration, you use Adoption to establish common ancestry for those objects and all of their components that meet adoption criteria. Adoption creates common ancestry between objects by replacing the Original Encyclopedia ID and Original Object ID of the identified object with the Original Encyclopedia ID and the Original Object ID of the object in the source model that is logically the same. An object whose IDs are replaced by Adoption is usually an object whose definition is replaced by the next migration. (However, once common ancestry is established between objects, migration can take place in either direction.)

System-defined objects always share common ancestry across all models and encyclopedias. Therefore these objects are not selectable for adoption. The exception is the IEF_SUPPLIED Work Attribute Set and its Work Attributes; these are selectable because they can be modified:

- System work attribute set and its work attributes and permitted values
- Default dialect
- System-defined objects
- System-defined object classes
- System-defined matrices

- System-supplied functions and their views (including data view groups, entity views, attribute views, and view implementations)
- IEF_SUPPLIED work attribute set and its work attributes and permitted values
- Pcroot
- Root function
- Root organizational unit
- Root subject area
- Scroll amount values
- Technical design
- Work attribute sets for system-supplied functions and their work attributes

Preparing for Adoption

Preparing for adoption begins with your having some indication from a report you have run that you need to perform an adoption. When needed, adoption is performed before migration. Adoption is needed only when objects you want to replace through migration do not meet migration's criteria for equivalence. Adoption establishes the required equivalence.

If you have never performed an adoption before, preparation guidelines include the following:

- Ensuring the target models meet requirements for adoption
- Becoming familiar with the list of aggregate object types and their selectable and un-selectable subordinates, so that you can determine a good selection strategy for aggregate object occurrences
- Understanding how adoption works well enough to know which model to specify as source model and which to specify as the destination model

Preparation guidelines that apply to each adoption include the following:

- Compile a list of candidates for adoption.
- Identify objects to rename prior to adoption.
- Identify objects that should not be adopted; that is, objects with like names but that are not logically the same, including:
  - Objects that can be identified through Compare Report because they share common ancestry with another object.
  - Objects that can be identified through PI views or manual inspection.
- Determine the scope of adoption.

- Adopt the model in a single session.
- Adopt the model in multiple sessions.
- Adopt selected objects.
- Identify objects to adopt together if adopting selected objects or identifying the sequence of object adoption if adopting an entire model in multiple sessions.
- Create an aggregate set to use for Trial Adoption and Adoption plus the subsequent Trial Migration and Migration.

Consider Requirements and Recommendations

You can either verify model and authorization requirements before you begin or you can simply proceed and take action only if you receive a message on an unmet requirement. For details on model requirements and recommendations, see Getting Ready.

Become Familiar with Aggregate Object Types for Adoption

The following tables list aggregate object types in the order they are presented on the selection list. A description is provided for each.

Aggregate Object	Description
ACTIVITY CLUSTER	Activity cluster
ATTRIBUTE	Attribute or foreign key attribute of entity type or subtype
BUSINESS AREA	Business area
BUSINESS SYSTEM	Business system
COMMAND	Command
COMMON ACTION BLOCK	Common, Default, and Derivation Action Diagrams
COMPONENT IMPLEMENTATION	Component implementation type
COMPONENT MODEL	Component model
COMPONENT SPECIFICATION	Component specification type
CONFIGURATION INSTANCE	Configuration Instance
CONSTRAINT	Foreign key linkage
CRIT SUCCESS FACTOR	Critical success factor
CURRENT DATA STORE	Current data store

Aggregate Object	Description
CURRENT INFO SYSTEM	Current information system
CUSTOM PROXIES	Custom Proxies
CUSTOM VIDEO PROPERTY	Custom video property
DATA CLUSTER	Data cluster
DATA COLUMN	Data column definition
DATA TABLE	Data table definition
DATABASE	Database definition
DEFAULT EDIT PATTERN	User default edit pattern
DENORMALIZED COLUMN	Denormalized column
DIALECT (NON-DFLT)	Non-default dialect
DIALOG FLOW	Dialog flow
ENTITY TYPE	Entity Type
ENVIRONMENT	Environment
EVENT	External event
EXIT STATE	Exit state
EXTERNAL OBJECT	External object
FACILITY	Computing or communication facility
FOREIGN KEY COLUMN	Foreign key column
FUNCTION (NON-ROOT)	Non-root function definition
GOAL	Business planning goal
INDEX	Index definition
INFORMATION NEED	Information need
INTERFACE TYPE	Interface type
LINK TABLE	Link table definition
LOCATION	Location of business assets
NAVIGATION DIAGRAM	Navigation diagram
OBJECTIVE	Business planning objective
ONLINE LOAD MODULE	Online load module
OPERATIONS LIBRARY	Operations library
ORG UNIT (NON-ROOT)	Non-root organizational unit

Aggregate Object	Description
PERFORMANCE MEASURE	Performance measure
PROCEDURE	Procedure
PROCEDURE STEP	Procedure step
PROCESS	Process definition
RELATIONSHIP MEMBER	Relationship membership
SERVER MANAGER	Server manager
SPECIFICATION TYPE	Specification type
STORAGE GROUP	Storage Group
STRATEGY	Business strategy
TABLESPACE	Tablespace definition
TACTIC	Business tactic
TEMPLATE	Screen template
TRANS OPERATION	Transaction operation
TYPEMAP	Type map
USER DEF MATRIX	User-defined matrix
USER DEF OBJ CLASS	User-defined object class
USER DEFINED OBJECT	User-defined object
USER SUBJECT AREA	Non-root subject area
WEB SERVICE DEFINITION	Web service definition
WINDOW LOAD MODULE	Window load module
WORK ATTRIBUTE	Attribute of work attribute set
WORK ATTRIBUTE SET	Work attribute set
z/OS LIBRARY	z/OS library

Become Familiar with Parents of Non-Selectable Components

Component objects in adoption are selectable if they are aggregate objects for adoption. Otherwise, they are non-selectable. Aggregate objects for adoption appear on the Adoption selection list. Object types that are not aggregates for adoption do not. Attribute, a component of entity type is selectable; batch job, a component of procedure, is not. To adopt an attribute, you would select it directly; to adopt a batch job, you would select the parent procedure.

For a list of objects that can be adopted only if you select the parent object, see Adoption Rules for Non-Selectable Objects in the appendix, "Adoption Rules."

Identify Models as Source and Destination

See the figure in Determining Which Model is the Source Model in the chapter "About Version Control." The following gives the guidelines to consider the direction to adopt:

Select	Model Meeting Criteria
Source Model	Its objects have the Original Encyclopedia IDs and Original Object IDs you want used as the basis for common ancestry It is the model you intend to use as the source when you perform migration between these two models, that is the model with the object definitions you want to retain in both models
Destination Model	Its object's ancestry is to be replaced It is the model you intend to use as the destination when you perform migration between these two models

Create an Aggregate Set of Objects for Adoption

Compiling a list of candidates for adoption begins with running reports on the model containing the objects with the ancestry you want to propagate (source model) and the model containing the objects whose ancestry you want replaced (destination model).

Compiling a list can be as simple as highlighting those objects on a report that you want to investigate as candidates for adoption. For suggestions on the type of entries to look for on each report, see the report examples under What Reports Can Help Me? in the chapter, Adopting Object to Establish Equivalence. Use the list to create a reusable aggregate set.

Identify Objects to Rename Prior to Adoption

Objects that you know are logically the same and have similar names, cannot be adopted until their names match, if matching names is a requirement for that object type.

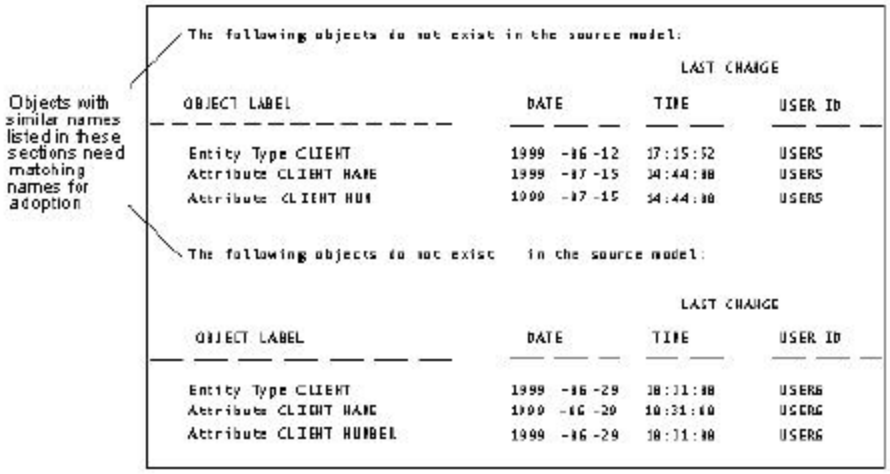
If you do not know whether the two objects with similar names are logically the same, examine their respective definitions to determine whether they actually have the same meaning and are used for the same purpose. If so, rename one (or both) of the objects so that the names match, then perform adoption.

Adoption Rules Based on Matching Names

When you are considering renaming candidates for adoption, be sure that matching names is relevant to the object's aggregate object type. See the Adoption Rules Tables in the appendix, Adoption Rules. While name matching is a common adoption requirement for selectable aggregate objects, it is not universal.

Example Where Renaming Is Indicated

The names CLIENT NUMBER and CLIENT NUM on the following figure suggest they can be different names for the same attribute, especially in light of the same parent entity name. The following is a sample view:



The source model can adopt an attribute in the destination model only if the source model contains an attribute with the same name as the selected attribute and if the parent entity types of these two attributes have common ancestry. If the parent entity types do not have common ancestry, this adoption requirement can be met by adopting the entity types in the same adoption session.

The adoption rules for entity types and attributes are discussed in the following table:

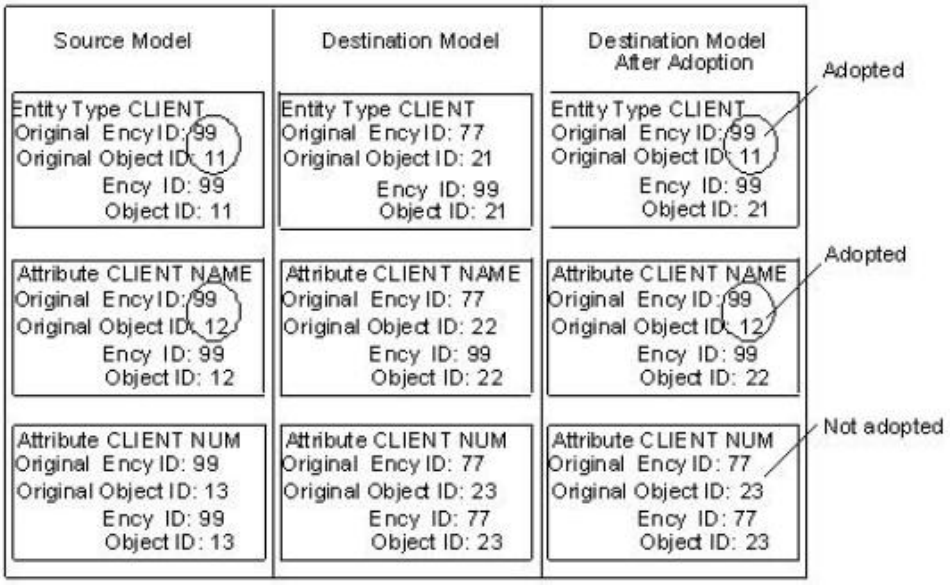
Selected Object	Adoption Rule is Adopted If:
ENTITY TYPE	Source model contains Entity Type with same name
ATTRIBUTE	Source model contains Attribute with same name Parent Entity Types share common ancestry

The adoption of Attribute CLIENT NAME and the adoption of Attribute CLIENT NUM depend on the following three things:

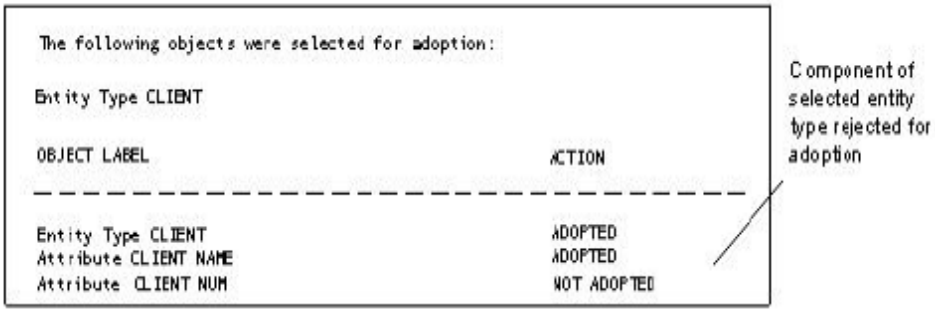
- An Attribute named CLIENT NAME in the source model
- An Attribute named CLIENT NUM in the source model
- The adoption of the parent object of the attributes, Entity Type CLIENT

Note: CLIENT NUM does not appear in the source model.

The following figure illustrates that Adoption would replace the Original Encyclopedia ID and Original Object ID of Entity Type CLIENT and Attribute CLIENT NAME, but would not replace IDs (adopt) of CLIENT NUM because it does not conform to the Adoption Rule for attributes.



The following figure illustrates how Trial Adoption would report the action on the attribute CLIENT NUM.



Identify Objects to Avoid Adopting

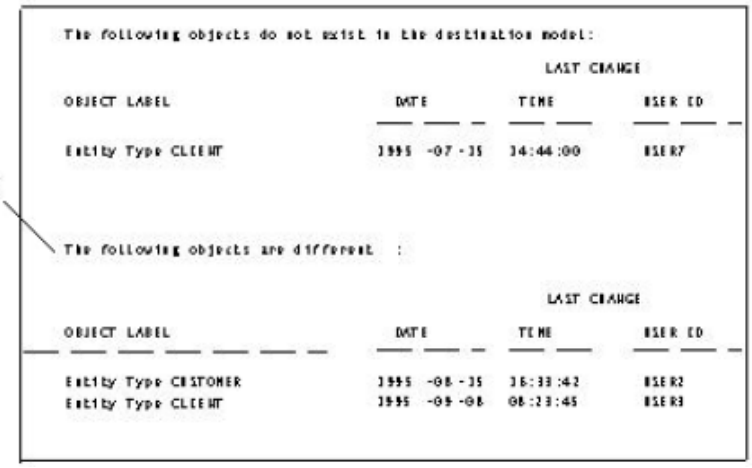
It is important to realize that objects in different models that share the same name and object type can not be logically the same. If you have any doubt that objects with the same names should be made equivalent, take the necessary precautions to verify whether the objects are intended to be independent or intended to be equivalent. You can use Public Interface views or download the models and examine the objects in question. If you discover any objects with the same name that should remain independent and retain their separate ancestry, avoid using the Adopt All option with the models that contain these objects.

There are two scenarios where such false candidates can be detected from a Compare Aggregate Object Report. Proceeding with adoption in one case has no ill effect; Adoption ignores the request to adopt and reports IGNORED on the Trial Adoption or Adoption report. Proceeding with adoption in the other case will result in removing the common ancestry the target object had with an object in the source model with a different name; Adoption honors the request to adopt and reports ADOPTED on the report. This section describes each of these scenarios, both with Compare Report entries to be on the alert for, and an illustration that depicts the technical perspective.

Example Where Overwriting Common Ancestry Would Occur

Consider the scenario where CLIENT in the destination model has the same name as CLIENT in the source model, but CLIENT in the destination model shares common ancestry with CUSTOMER in the source model. The two objects named CLIENT cannot be logically the same. You can be sure such objects are not meant to be equivalent.

Don't attempt adoption with an object listed in the Different section that has the same name as an object in the source model, if name is used as the adoption rule.

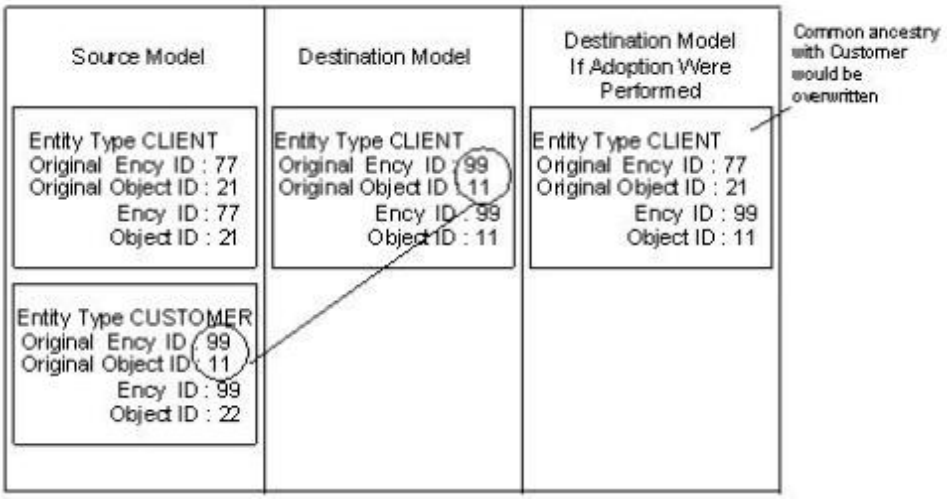


When the Compare process reports that CLIENT does not exist in the destination model, it means that no object in the destination model has an Original Encyclopedia ID and Original Object ID that matches CLIENT's in the source model.

For the Compare process to report two objects as different, they must be equivalent but last changed at different times, as indicated by the date and time of their respective session objects. It can be assumed from this that CLIENT in the destination model has an Original Encyclopedia ID and Original Object ID that matches CUSTOMER's in the source model.

Note: Adoption should not be performed on CLIENT.

The following figure shows what would happen if adoption were performed on CLIENT. (Adoption would succeed because Adoption would find an entity type named CLIENT in the source model and assign this object's IDs to CLIENT in the destination model.) Adoption of CLIENT in the destination model would result in overwriting the common ancestry between CLIENT and CUSTOMER.

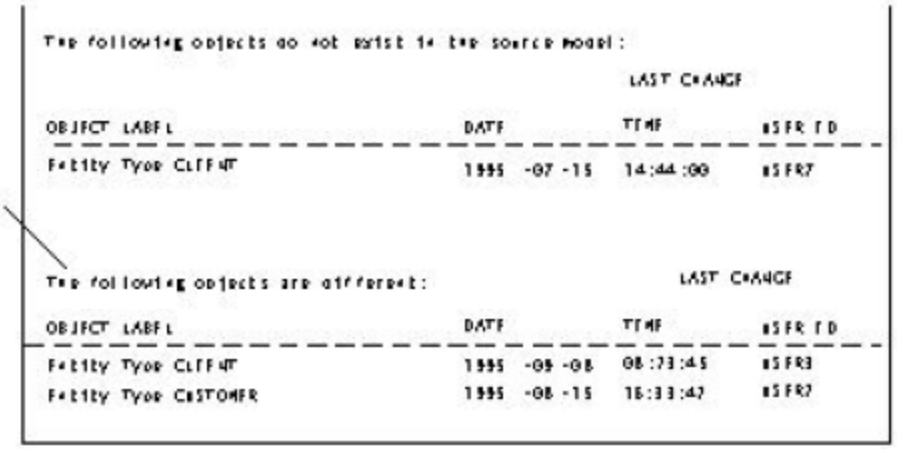


The general rule is to avoid performing adoption between two objects with matching names when one of the objects shares common ancestry with another object with a different name. This guideline expressed in terms of the Compare Report would be—Never attempt adoption of an object listed in the Different section of the Compare Aggregate Object Report.

Example Where Adoption Request Would Be Ignored

Consider the scenario where CLIENT in the source model has the same name as CLIENT in the destination model, but CLIENT in the source model shares common ancestry with CUSTOMER in the destination model. The two objects named CLIENT cannot be logically the same. You can be sure such objects are not meant to be equivalent.

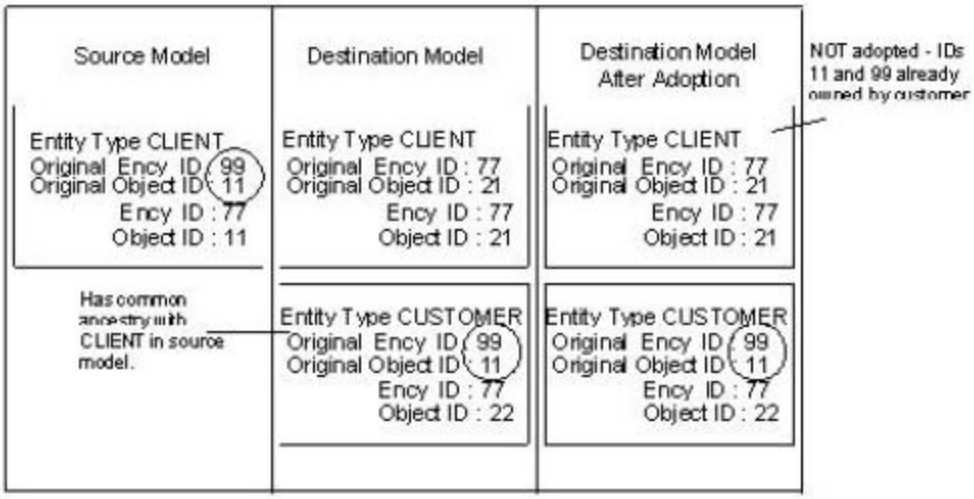
Don't attempt adoption with an object listed in the Different section that has the same name as an object in the destination model, if name is used as the adoption rule.



Since no object in the source model has an Original Encyclopedia ID and Original Object ID that matches CLIENTs in the destination model, the Compare process reports that CLIENT does not exist in the source model, even though there is an unrelated entity type called CLIENT in the source model.

Since CLIENT in the source model has an Original Encyclopedia ID and Original Object ID that matches CUSTOMER's in the destination model, the Compare process evaluates CUSTOMER and CLIENT as equivalent and reports that they are different based on different timestamps.

The following figure shows what would happen if adoption were requested. CLIENT in the destination model would not be adopted because another object in the destination model, CUSTOMER, already owns the Original Encyclopedia ID and Original Object ID that adoption would apply. That is, you could not replace the Original Encyclopedia ID and Original Object ID of CLIENT in the destination model with 99-11, because these values already identify CUSTOMER in the model, and no more than one occurrence of the combination is permitted in a model.



The general rule is to avoid performing adoption between two objects with matching names when one of them shares common ancestry with another object with a different name. This guideline expressed in terms of the Compare Report would be—Never attempt adoption of an object listed in the Different section of the Compare Aggregate Object Report.

To do so in a case like this would result in an IGNORED action since you cannot replace the IDs of an object with IDs that already exist in the model. See the message IGNORED in the table, Adoption Report Information Messages in Adoption Report Messages.

Determine Whether to Adopt All or Adopt Selected Objects

The following table has the guidelines to determine whether to adopt a model with the Adopt All option or to adopt selected objects in a model with the Adopt Selected option:

Select option:	If the source and destination models
Adopt All	Meet ALL of these criteria: Are considered the same (contain all or nearly all objects in common) Contain all or many objects that are logically the same, but that do not share common ancestry Contain no objects that are NOT designed to be equivalent, but have the same name and are of the same type
Adopt Selected	Meet ANY of these criteria: Are considered partially the same (contain only some objects in common) Contain one or more objects that are logically the same but that do not share common ancestry Contain objects that preclude using Adopt All (Objects that are not adoption candidates even though they are named the same.)

The third criteria for Adopt All is critical. Adopt All means all objects meeting adoption criteria will be adopted. Adoption criteria for most object types is having the same name. If the models contain any objects with the same names that have no similarity in purpose, you want to avoid selecting those objects during an adoption. When using adoption with such models, be sure to use the Adopt Selected and avoid selecting the identified object(s). For examples, see Identify Objects to Avoid Adopting.

The guideline to choose Adopt All when the two models are considered the same, and to choose Adopt Selected when the two models are considered only partially the same, is not a hard and fast rule. For example, consider the scenario where two models are the same, but Technical Design transformation was performed. This process deletes and recreates objects, so the TD objects do not share common ancestry. In this situation, it can be easier to use the Adopt All Objects option rather than to manually select all TD objects to be adopted.

There are many situations where you will want to weigh the advantages of each option. Consider the following:

Adopt All	Adopt Selected
No object selection time needed	Less processing time needed
No chance of missing an object you wanted to adopt	Less chance of adopting an object you do not want to adopt, since verifying a limited list is easier to do with accuracy

If you choose adopt selected, consider creating an aggregate set that you can reuse. You can use Compare Report results to identify objects to include in an aggregate set for trial adoption. You can use Trial Adopt Report results to modify the set, if needed.

The following table has the guidelines to determine whether to add, use, or modify an aggregate set:

Consider This Aggregate Set Usage	If the Model Comparison
Add a set	Is being performed with objects you plan to adopt or migrate, and no appropriate set exists
Use an existing set	Is to focus on only the objects identified in the set
Modify a set	Is to focus on many objects already defined with or without additional objects

Example Where Adopt All Is Indicated

Consider the scenario where two models are the same but have been uploaded into the encyclopedia independently, and therefore common ancestry has not been maintained between objects that are logically the same in the models. (This occurs when you upload models into an encyclopedia from a tran file created by the download with upload option.)

Note: If the tran file was created by the extract with apply option, common ancestry is maintained and adoption is not necessary.

When you run the Compare Report with the Compare All option on such models, the result is two identical lists under the source only and destination only sections. See the figure in Sample Compare Report Before Adoption.

What the Entries Mean

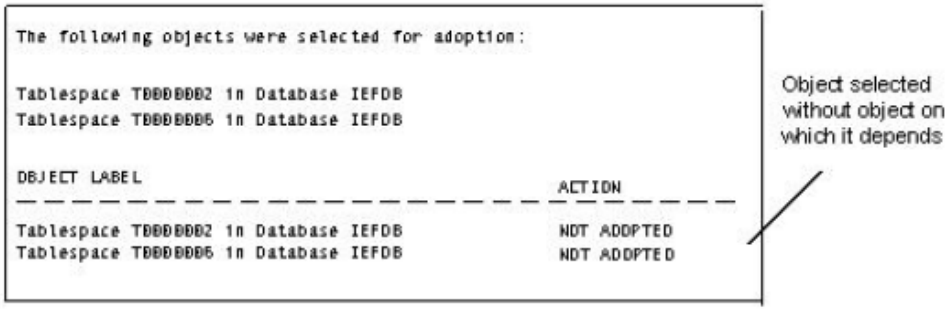
Identical lists indicate that Compare Report does not recognize the objects on the following lists as versions of the same object:

- The objects do not exist in the source model
- The objects do not exist in the destination model

When all objects in the source model share names with all objects in the destination model, you can want to consider model–level adoption.

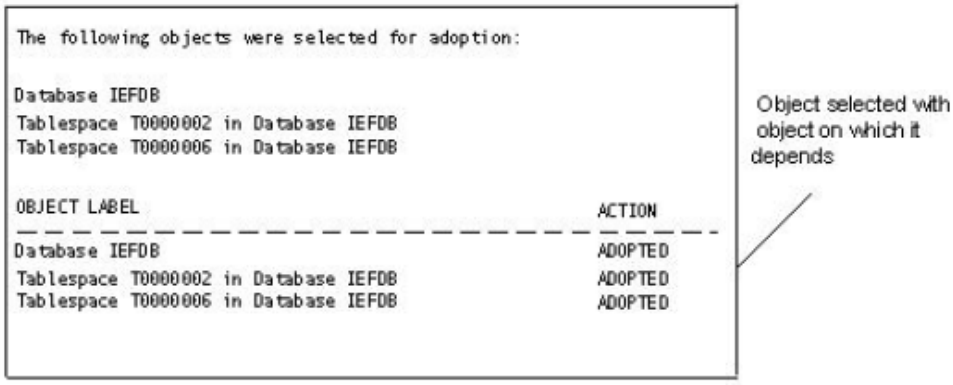
Example Where Object in Adopt Selected Depends on Another

Adopt Selected can be used in several types of scenarios. For example, the same logical object can be created independently in two models; you decide to replace one definition with the other. The main consideration for adoption in such a scenario is to determine whether you must adopt any other objects with the object of interest. Consider the scenario where you want to adopt tablespaces. The following is a graphical depiction:



What the Entries Mean

Tablespaces in the reported database cannot be adopted. This can be because the database they belong to has not been adopted. To find out, run the Trial Adoption report again and specify the parent database.



Tablespaces in the reported database could be adopted when specified with the parent database.

Recommended Action

The general guideline when specifying selected objects to adopt is to include any object on which adoption of the object of interest depends. To see if the object of interest is one that is dependent on other objects, see the following table:

Adoption of Object Type	Aggregate Objects Adopted in Same or Previous Session
ACTION BLOCK (BAA or BSD)	ENTITY TYPEs and WORK ATTRIBUTE SETs defined in views
ATTRIBUTE (Foreign Key Attribute)	RELATIONSHIP MEMBERs
COMMAND	BUSINESS SYSTEM
CUSTOM PROXIES	BUSINESS SYSTEM
CUSTOM VIDEO PROPERTY	BUSINESS SYSTEM
DATA TABLE	ENTITY TYPEs, RELATIONSHIP MEMBERs
DIALOG FLOW	PROCEDUREs, EXIT STATEs
FUNCTION	ENTITY TYPEs and WORK ATTRIBUTE SETs defined in views
ONLINE LOAD MODULE	BUSINESS SYSTEM

Adoption of Object Type	Aggregate Objects Adopted in Same or Previous Session
OPERATIONS LIBRARY	BUSINESS SYSTEM
PROCEDURE	BUSINESS SYSTEM, ENTITY TYPEs and WORK ATTRIBUTE SETs defined in views
PROCESS	ENTITY TYPEs and WORK ATTRIBUTE SETs defined in views
RELATIONSHIP MEMBER	ENTITY TYPEs
TABLESPACE	DATABASEs
TEMPLATE	BUSINESS SYSTEM
USER DEFINED MATRIX	USER DEF OBJ CLASS
WEB SERVICE DEFINITION	BUSINESS SYSTEM
WINDOW LOAD MODULE	BUSINESS SYSTEM
z/OS LIBRARY	BUSINESS SYSTEM

Example Where Many Objects Are Adopted in Separate Sessions

For large-scale adoptions, consider adopting the objects in multiple sessions. Some selectable objects must be adopted during the same session (or previous session); others have no dependencies.

The general rule is that subordinate objects cannot be adopted unless their parents share common ancestry. If you select parent objects for adoption in the same session as the dependent objects, the parent objects are processed first. Therefore, adoption in the same session is sufficient to meet the rule.

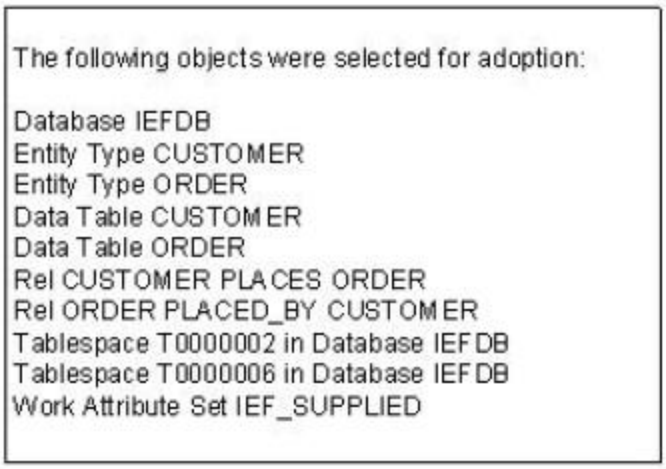
Use the following guidelines to group the model objects into multiple aggregate sets and plan the sequence of adoption. If you select aggregate object types in the following sequence, and select all occurrences of each type, you will avoid encountering problems due to a missing object on which a selected object depends.

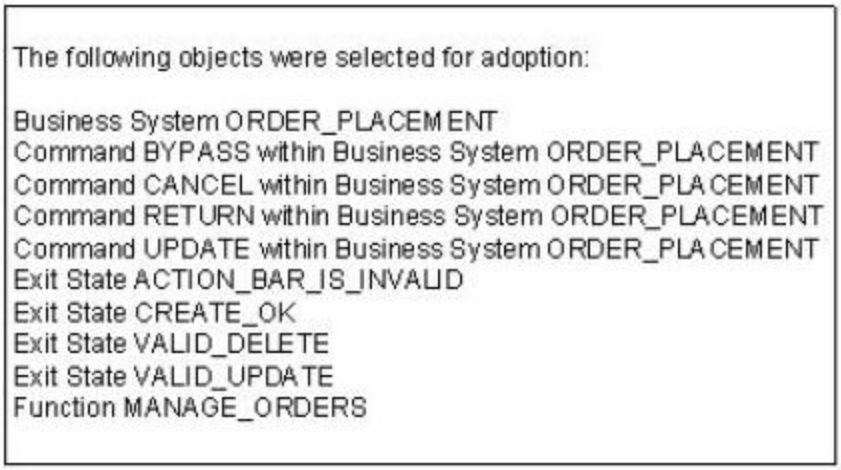
The following is the recommended Adoption Sequence:

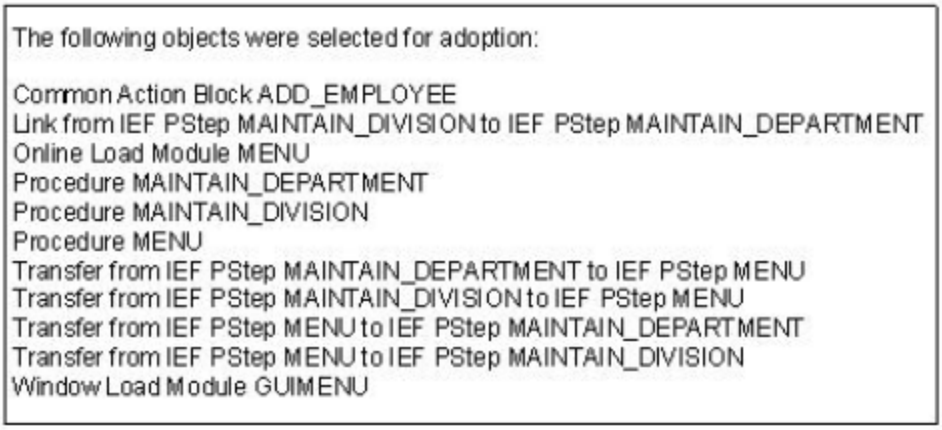
1. USER SUBJECT AREAs
2. ENTITY TYPEs
3. RELATIONSHIP MEMBERs
4. FOREIGN KEY ATTRIBUTEs
5. DATA TABLEs
6. DATABASEs

7. TABLESPACEs
8. WORK ATTRIBUTE SETs
9. BUSINESS SYSTEMs
10. COMMANDs
11. TEMPLATEs
12. EXIT STATEs
13. FUNCTIONs
14. PROCESSes
15. COMMON ACTION BLOCKs (BAA)
16. COMMON ACTION BLOCKs (BSD)
17. PROCEDUREs
18. DIALOG FLOWs
19. ONLINE LOAD MODULEs
20. WEB SERVICE DEFINITIONs
21. WINDOW LOAD MODULEs
22. CONFIGURATION INSTANCEs
23. CUSTOM PROXIES
24. OPERATIONS LIBRARY
25. z/OS LIBRARY
26. USER DEF OBJ CLASS
27. USER DEFINED MATRIX
28. All other Planning objects (no dependencies)

The following three graphics assume all occurrences of all aggregate object types numbered 1-7 are in an aggregate set adopted during session 1, 8-13 in session 2, and 14-21 in session 3.







Performing Adoption or Trial Adoption

Use the following procedure as a guide when performing an Adoption or Trial Adoption.

Note: For syntax on the command line alternative, enter ADPTMODL from the directory where the Version Control client is installed.

Access the ADOPT Function

Follow these steps:

1. Access Version Control. See Starting the Version Control Client. Select one of the following options, depending on whether you want to perform an adoption or simply report what would happen if an adoption were performed.
   - Model / Adopt
   - Model / Reports / Trial Adopt
2. Select the source model and destination model. For considerations, see Determining Which Model is the Source Model and Identify Models as Source and Destination.
   a. (Optional) Specify a filter for the Source Model names as described in Using a Filter.
   b. From the Source Model list, select the model that contains the objects with ancestry to be propagated.
   c. (Optional) Specify a filter for the Destination Model names.
   d. From the Destination Model list, select the model that contains the objects to be selected for adoption.
   e. Click OK.
3. Specify the Adoption options. Press F1 for *Online Help* on any field.
   a. If you want to generate an Adopt Model or Trial Adopt Model report, change the object range option from Selected to All. For considerations, see the table Guidelines for Selecting the Object Range Option for Adoption.
   b. If adopting or trial adopting selected objects that belong to an aggregate set, check Aggregate Set.
   c. Specify a report destination or accept the displayed default. For tips, see Specifying How to Save Reports.
   d. Specify a unique filename for the report or accept the displayed default.
   e. If you want to limit the report to adopted objects only, change the default Report Type from All objects to Adopted objects only.

4. The default report type includes objects where the action taken can be adopted, adopts, ignored, ignores, already adopted, already adopts, not adopted, and unadopted. The Adopted objects only report type includes only objects where the action taken is either adopted or adopts.

   a. Click OK.

   b. If you selected All for Object Range, skip to the Adopt Confirmation dialog (Step 11). If you checked Aggregate Set, skip to Select Aggregate Set (Step 9). Otherwise, continue.

5. Select an aggregate object type for which you want to list occurrences. See, Become Familiar with Aggregate Object Types for Adoption.

   a. Highlight the aggregate object type of interest.

   b. Click List, or to limit the display to a specific range of occurrences of the selected aggregate object type in the destination model, enter a filter value as described in Using a Filter and click List.

   c. Aggregate Object Occurrences

6. Select one occurrence, select multiple occurrences, or click Select All.

   The section Preparing for Adoption describes strategies for adoption that can assist you in determining objects to select for adoption. To identify object types that require other objects to be adopted in the same session, or previously adopted, review Guidelines for Selecting Objects to Adopt Together in Example Where Object in Adopt Selected Depends on Another.

   If you are adopting all model objects in multiple sessions, list each aggregate object type in the sequence recommended in Guidelines for Selecting Objects to Adopt in Multiple Sessions, then use Select All to select all occurrences for each aggregate object type that has occurrences. See Example Where Many Objects Are Adopted in Separate Sessions.

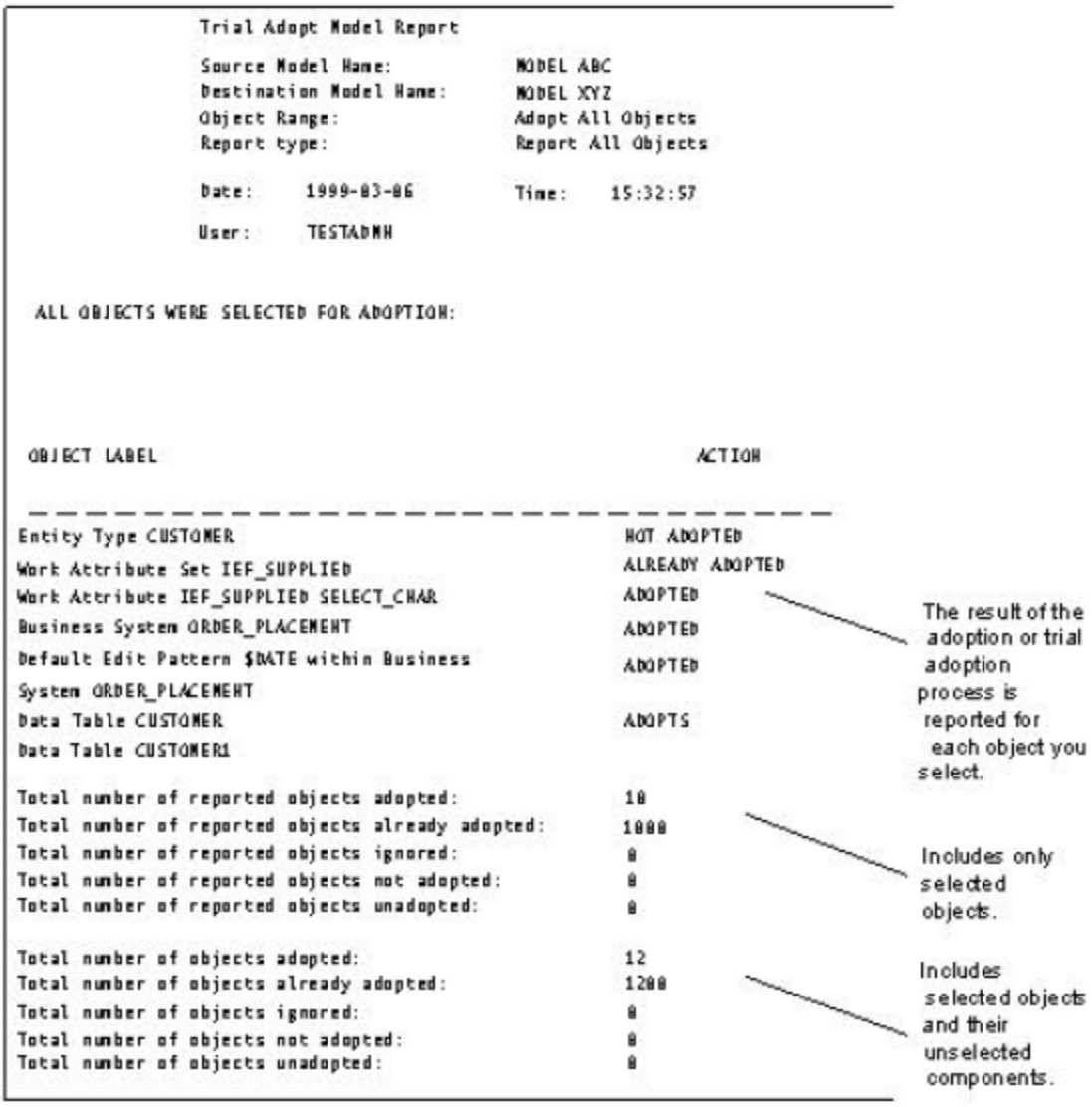
7. If you selected only one object and want to expand that object, click Expand or to limit the display to a specific range of aggregate object expanded occurrences, enter a filter value and click Expand. Proceed with the Aggregate Object Expanded Occurrences dialog (Step 7).

   If you selected multiple individual occurrences, clicked Select All, or selected an object that cannot be expanded or that you do not want to expand, click Add, then click Exit. The Aggregate Object Types List appears. Either continue the selection process as described in Step 5, or click Cancel and proceed to generate the report as described in Step 9 for the Selected Object List.

8. Select one occurrence, select multiple occurrences individually, or click Select All. Continue in one of the following ways:
   - If you selected only one object and want to expand that object, click Expand again. Continue as described at the beginning of this step. (You can repeat this step of selecting and expanding until you reach the lowest-level object in the chain.)
   - If you selected multiple objects or selected a single object that either cannot be expanded or that you do not want to expand, click Add, then click Exit. If you performed only one expansion, the Aggregate Object Occurrences dialog with the parent object appears. If you performed multiple expansions, the Aggregate Object Expanded Occurrences dialog with the parent object appears. Click Exit repeatedly until you display the Aggregate Object Types dialog or click Add to add the highlighted parent object before clicking Exit at any step.
9. To continue selecting objects to adopt, return to step 5. To generate the report with the objects currently selected, click Cancel and continue with Step 10.
10. Select the appropriate aggregate set(s) and click OK. Click OK again after examining retrieval status.
11. Verify that the listed objects are the objects to be adopted. If so, select Proceed. (If not, see *Online Help* for other options.)
12. Click Yes to proceed with the trial adoption or adoption.
13. Review the following messages, then click Continue to generate the Adopt Objects Report. The report appears in a Review panel. You can also access it from the directory you specified.
    - Request being forwarded to server
    - Preparing for adoption
    - *nnn* objects processed during adoption phase (displays incrementing object count)
    - *nnn* objects processed during unadoption phase (if you are performing an Adoption)
    - Rolling back changes to database (if you are performing a Trial Adoption)
    - Adoption complete

Evaluating the Adoption or Trial Adoption Report

Examples of messages that can be reported on an Adoption Report or Trial Adoption Report are shown on the following figure:



```
                Trial Adopt Model Report

                Source Model Name:          MODEL ABC
                Destination Model Name:     MODEL XYZ
                Object Range:               Adopt All Objects
                Report type:                Report All Objects

                Date:     1999-03-06        Time:    15:32:57

                User:     TESTADMN


 ALL OBJECTS WERE SELECTED FOR ADOPTION:



 OBJECT LABEL                                          ACTION

 ------------------------------------------------------------
Entity Type CUSTOMER                                   NOT ADOPTED
Work Attribute Set IEF_SUPPLIED                        ALREADY ADOPTED
Work Attribute IEF_SUPPLIED SELECT_CHAR                ADOPTED
Business System ORDER_PLACEMENT                        ADOPTED
Default Edit Pattern $DATE within Business             ADOPTED
System ORDER_PLACEMENT
Data Table CUSTOMER                                    ADOPTS
Data Table CUSTOMER1

Total number of reported objects adopted:              10
Total number of reported objects already adopted:      1000
Total number of reported objects ignored:              0
Total number of reported objects not adopted:          0
Total number of reported objects unadopted:            0

Total number of objects adopted:                       12
Total number of objects already adopted:               1200
Total number of objects ignored:                       0
Total number of objects not adopted:                   0
Total number of objects unadopted:                     0
```

The result of the adoption or trial adoption process is reported for each object you select.

Includes only selected objects.

Includes selected objects and their unselected components.

Adoption Report Messages

The adoption process can take any of eight different actions on an object, based on the evaluation of the object's current status in the model and the adoption rules for its object type. The eight types of actions are described in the following table:

Action	Indicates	Object Label
ADOPTED	This object now has the same Original Encyclopedia ID and Original Object ID as the object in the source model that is logically the same.	The same
ADOPTS	This object now has the same Original Encyclopedia ID and Original Object ID as the object in the source model that is logically the same.	Different
IGNORED	No change was made to the IDs of this object; another object in the destination model already has common ancestry with the object in the source model that is logically the same. (Two objects in the same model can not share common ancestry.)	The same
IGNORES	No change was made to the IDs of this object; another object in the destination model already has common ancestry with the object in the source model that is logically the same.	Different
ALREADY ADOPTED	This object already has common ancestry with its equivalent object on the source model. No action required.	The same
ALREADY ADOPTS	This object already has common ancestry with its equivalent object on the source model. No action required.	Different
NOT ADOPTED	This object was not adopted because it does not meet adoption criteria according to adoption rules for its object type; that is, no equivalent object was found in the source model.	N/A

Action	Indicates	Object Label
UNADOPTED	This component object of an adopted parent object was not adopted because it did not meet adoption criteria for its object type. During component unadoption processing, the component object's Original Encyclopedia ID and Original Object ID are replaced with the current encyclopedia ID and an unused object ID.	N/A

The Object Label listed in the report is not necessarily the name of the object. Object Label refers to text that uniquely identifies the object occurrence within a model; the label for an object can be different from the name of the object. Many adoption rules are based on objects that are logically the same having the same name; this does not imply that they have the same labels. Therefore, objects can be adopted based on identical names when they have different labels.

Chapter 5: Migrating Objects from One Model to Another

This chapter includes information about migrating objects from one model to another.

FAQs About Migration

Frequently Asked Questions (FAQs) and their short answers follow:

- What is migration's purpose? To copy an object's definition from one model to another model.
- Why would I need migration? Object changes and additions to one model need to be applied to another version of that model.
- When would I use migration? To synchronize two model versions.
- How do I identify objects to migrate? Look at the reports to identify candidates.
- What reports can help me? Compare Aggregate Object Report and Trial Migrate Aggregate Object Report can help you identify candidates for migration.
- The Trial Migrate Report can also help you identify conditions that require correction or change for successful migration.
- How can I ensure a successful migration? Perform Trial Migration, take corrective action based on messages.
- How does migration work? Objects you identify in one model replace their equivalent objects in the other model, if they exist; otherwise objects are added.

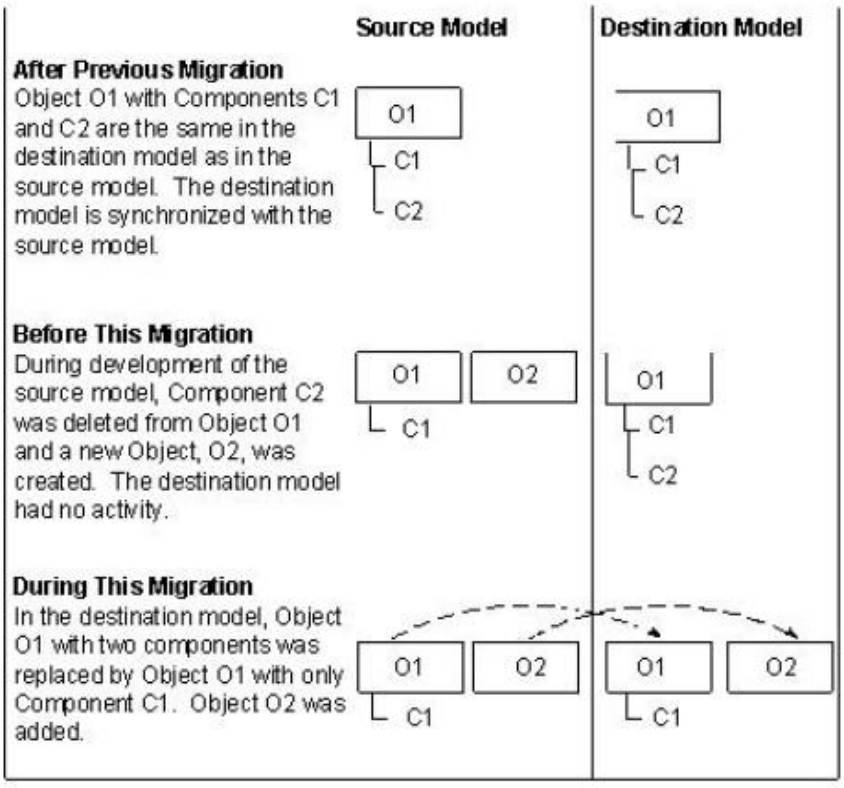
What Is Migration's Purpose?

Migration is the Version Control function that copies aggregate objects from the source model to the destination model, where models reside in the same C/S Encyclopedia. When an object equivalent to the object being migrated exists in the destination model, the object being migrated replaces the existing object. If the object being replaced has components that are not present in the object being migrated, those components in the destination model are deleted by the migration. When an equivalent object does not exist in the destination model, the object being migrated is added to the destination model.

When an aggregate object is migrated, all of its components are migrated with it. If any of its companion objects do not already exist in the destination model, they are also migrated with the aggregate. If the destination model does not contain all of the aggregate object's enabling objects, you must select the missing enablers to migrate along with the aggregate object in order for the migration to succeed.

Migration Example

The following figure illustrates the effect of migrating a changed object, called O1, and a new object, called O2, from the source model to the destination model. Migration does not change the source model. All changes occur only in the destination model.



Note: In this example, successful migration of O1, requires deletion of component C2. This depends on C2 not being referenced in the destination model. If C2 is referenced in the destination model, the reference must be removed prior to the migration.

Why Would I Need Migration?

Migration enables you to maintain the same version of equivalent objects across models. It enables you to keep models synchronized, so that when new objects are added to one model or existing objects are changed, those additions and changes can be applied to the other model. Migration also enables you to selectively copy objects from one model to another, even when model synchronization is not a goal.

When Would I Use Migration?

You can use Migration in any of the following situations:

- To synchronize models used for parallel development
- To propagate changes to a common objects model
- To apply test model fixes to the development model
- To replace problematic objects with working versions from a backup model
- To reuse an existing object in an unrelated project model

For details, see Version Control Tasks.

How Do I Identify Objects to Migrate?

You can identify equivalent objects in two models that are in different versions by running the Compare Report and seeing which objects are reported as different. You can identify objects that have been added to each of two models that are to be kept synchronized by taking note of those objects reported as not existing in the source model and not existing in the destination model. You can identify objects you must migrate with the objects of interest by running the Trial Migration function and examining the resulting report for instances of the <object> REQUIRES <object> message.

What Reports Can Help Me?

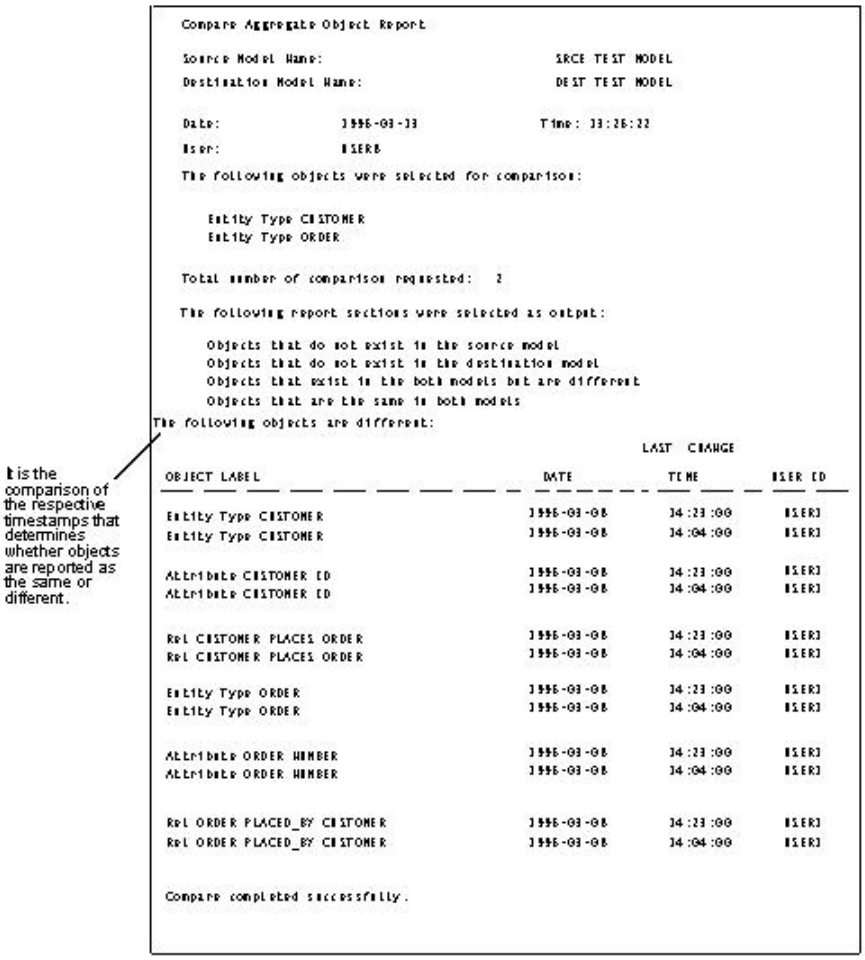
The following three reports illustrating the same scenario prior to, during, and after migration use the data in the table:

Source Model	Destination Model
Entity Type CUSTOMER Attribute CUSTOMER ID	Entity Type CUSTOMER Attribute CUSTOMER ID
Entity Type ORDER Attribute ORDER NUMBER	Entity Type ORDER Attribute ORDER NUMBER
Rel CUSTOMER PLACES ORDER Rel ORDER PLACED_BY CUSTOMER	Rel CUSTOMER PLACES ORDER Rel ORDER PLACED_BY CUSTOMER

Note: To select all this data for Compare Report, you only need to select the Entity Types, since the subordinate Attributes and Relationship Memberships are automatically processed along with the related Entity Types. To select all of this data for Migration, you need to select the Entity Types and the Relationship Memberships, since only the component Attributes are processed along with the related Entity Types. (Relationship Membership is subordinate to Entity Type but is not a component.)

Sample Compare Report on Objects Before Migration

The following sample Compare Aggregate Object Report reflects data as it would appear if the objects in the source model were in a different version than equivalent objects in the destination model. Objects listed as different on the Compare Aggregate Object Report can be defined to an aggregate set for easy use with Trial Migration and Migration operations.

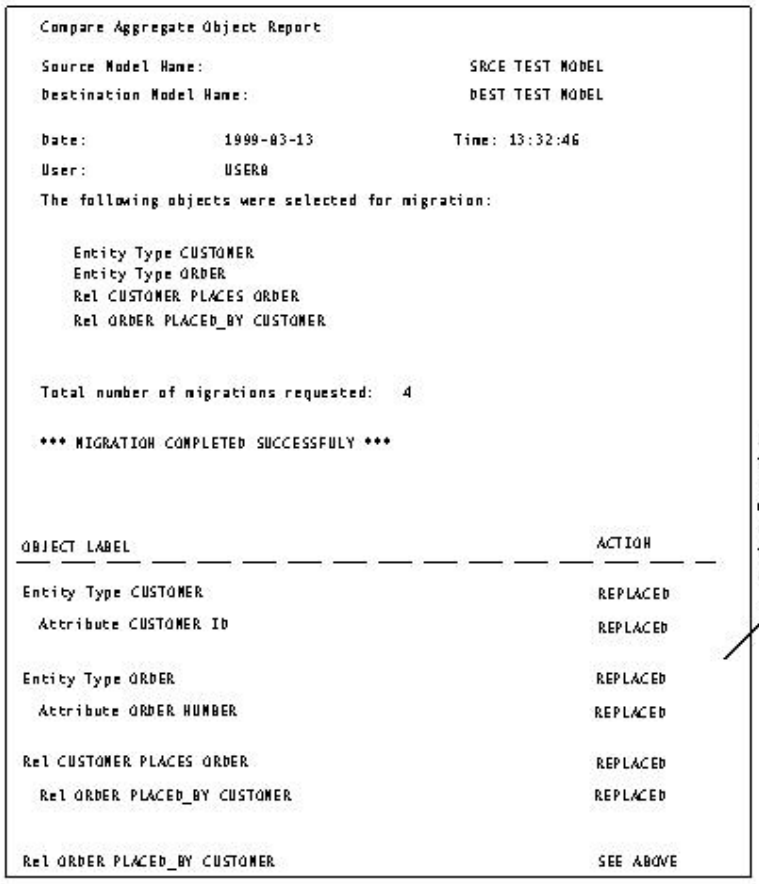


```
Compare Aggregate Object Report

Source Model Name:                          SRCE TEST MODEL
Destination Model Name:                     DEST TEST MODEL

Date:              1996-03-13               Time: 13:25:22
User:              USER8

The following objects were selected for comparison:

   Entity Type CUSTOMER
   Entity Type ORDER

Total number of comparison requested:  2

The following report sections were selected as output:

   Objects that do not exist in the source model
   Objects that do not exist in the destination model
   Objects that exist in the both models but are different
   Objects that are the same in both models

The following objects are different:

                                                          LAST  CHANGE
OBJECT LABEL                                    DATE          TIME        USER ID
---------------------------------------------------------------------------------
Entity Type CUSTOMER                        1996-03-08    14:23:00      USER1
Entity Type CUSTOMER                        1996-03-08    14:04:00      USER1

Attribute CUSTOMER ID                       1996-03-08    14:23:00      USER1
Attribute CUSTOMER ID                       1996-03-08    14:04:00      USER1

Rel CUSTOMER PLACES ORDER                   1996-03-08    14:23:00      USER1
Rel CUSTOMER PLACES ORDER                   1996-03-08    14:04:00      USER1

Entity Type ORDER                           1996-03-08    14:23:00      USER1
Entity Type ORDER                           1996-03-08    14:04:00      USER1

Attribute ORDER NUMBER                      1996-03-08    14:23:00      USER1
Attribute ORDER NUMBER                      1996-03-08    14:04:00      USER1

Rel ORDER PLACED_BY CUSTOMER                1996-03-08    14:23:00      USER1
Rel ORDER PLACED_BY CUSTOMER                1996-03-08    14:04:00      USER1

Compare completed successfully.
```

It is the comparison of the respective timestamps that determines whether objects are reported as the same or different.

Sample Migrate Aggregate Object Report

The following sample Migrate Aggregate Object Report shows objects successfully replaced. If objects could not be successfully migrated, an explanation of the condition preventing successful migration would be reported. For details, see Evaluate Migration Error Messages in the chapter, "Migrating Objects from One Model to Another."

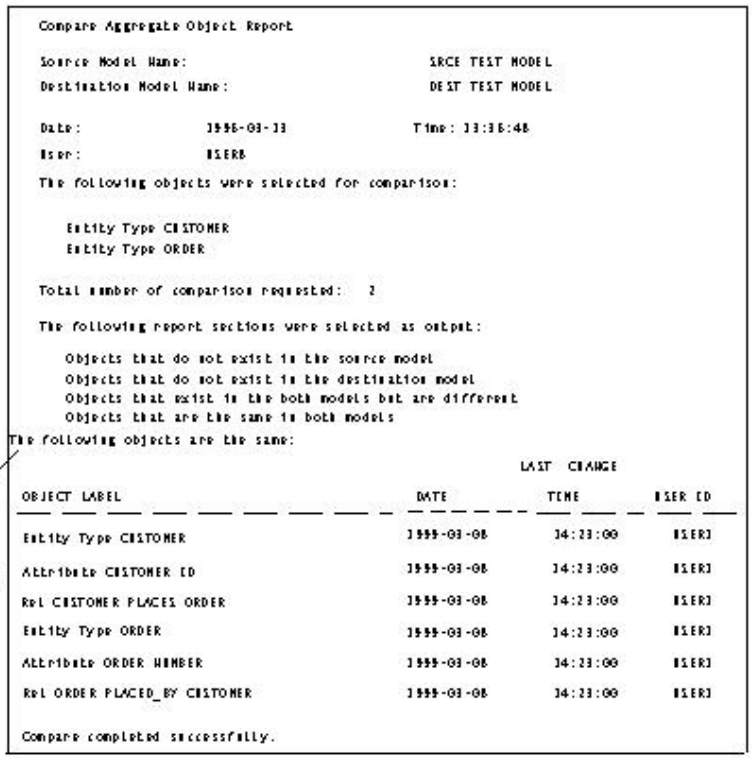


```
Compare Aggregate Object Report

Source Model Name:                          SRCE TEST MODEL
Destination Model Name:                     DEST TEST MODEL

Date:              1999-03-13               Time: 13:32:46
User:              USER0
The following objects were selected for migration:

    Entity Type CUSTOMER
    Entity Type ORDER
    Rel CUSTOMER PLACES ORDER
    Rel ORDER PLACED_BY CUSTOMER


Total number of migrations requested:   4

*** MIGRATION COMPLETED SUCCESSFULY ***



OBJECT LABEL                                                ACTION
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ----
Entity Type CUSTOMER                                        REPLACED
  Attribute CUSTOMER ID                                     REPLACED

Entity Type ORDER                                           REPLACED
  Attribute ORDER NUMBER                                    REPLACED

Rel CUSTOMER PLACES ORDER                                   REPLACED
  Rel ORDER PLACED_BY CUSTOMER                              REPLACED


Rel ORDER PLACED_BY CUSTOMER                                SEE ABOVE
```

Migration replaces the objects in the destination model with their equivalent objects from the source model.

Sample Compare Report After Migration

The following Compare Aggregate Object Report shows that the migrated objects are now evaluated as being the same version in the destination model as they are in the source model.



```
Compare Aggregate Object Report

Source Model Name:                          SRCE TEST MODEL
Destination Model Name:                     DEST TEST MODEL

Date:              1998-08-13               Time: 13:38:48
User:              USER8

The following objects were selected for comparison:

   Entity Type CUSTOMER
   Entity Type ORDER

Total number of comparison requested:   2

The following report sections were selected as output:

   Objects that do not exist in the source model
   Objects that do not exist in the destination model
   Objects that exist in the both models but are different
   Objects that are the same in both models
The following objects are the same:

                                                          LAST  CHANGE
OBJECT LABEL                                    DATE          TIME        USER ID
---------------------------------------------------------------------------------
Entity Type CUSTOMER                            1999-08-08    14:23:00    USER1
Attribute CUSTOMER ID                           1999-08-08    14:23:00    USER1
Rel CUSTOMER PLACES ORDER                       1999-08-08    14:23:00    USER1
Entity Type ORDER                               1999-08-08    14:23:00    USER1
Attribute ORDER NUMBER                          1999-08-08    14:23:00    USER1
Rel ORDER PLACED_BY CUSTOMER                    1999-08-08    14:23:00    USER1

Compare completed successfully.
```

After the object versions from the source model replace the object versions in the destination model, the same version of the objects exist in both models.

How Can I Ensure a Successful Migration?

Performing a successful migration can take multiple trials, where each trial identifies certain corrective actions to take. Attempting to foresee and prevent all possible conditions that could cause an unsuccessful migration could be very time-consuming if you are migrating more than just a few objects. Fortunately, Trial Migration saves you that time and effort. When weighing preventative versus corrective measures, consider the advantages of letting the software do the work. For the recommended corrective action for each possible message, see the What to do sections in Evaluate Migration Error Messages in the chapter, "Migrating Objects from One Model to Another."

How Does Migration Work?

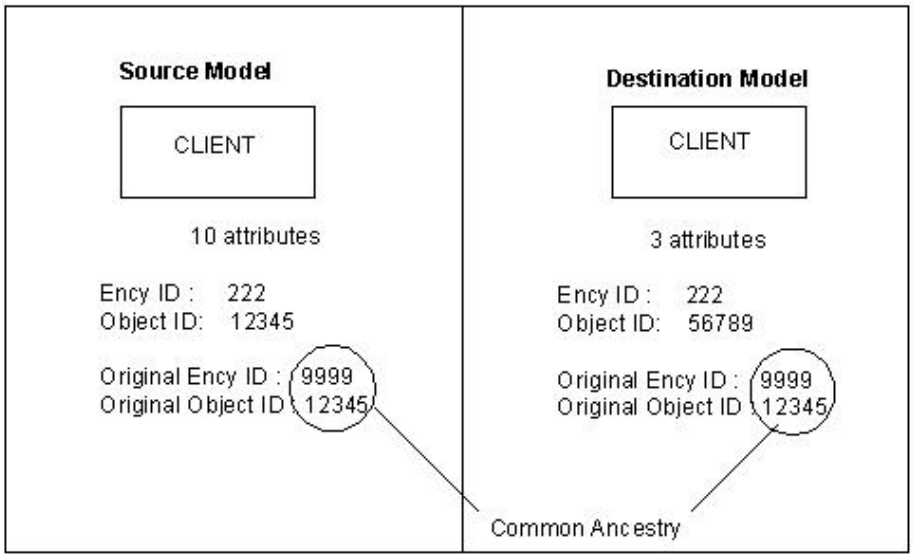
The following series of figures illustrating before and after views of migration suggests the effects you can expect of migration.

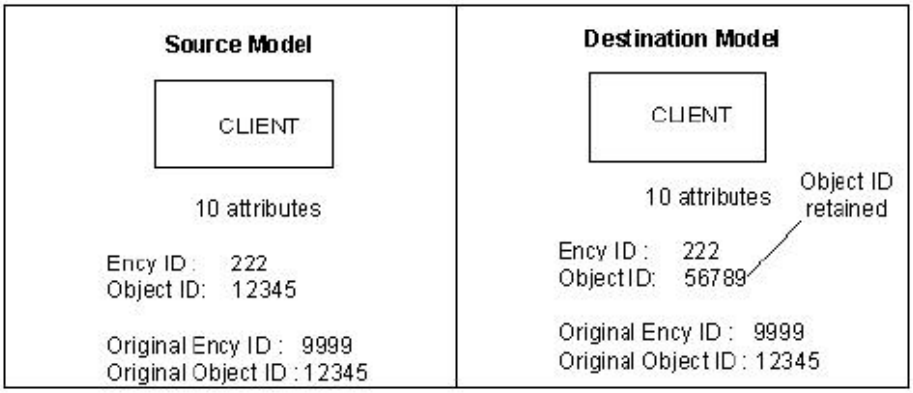
Migration Replaces Objects with Common Ancestry

The following pair of figures provide a before and after view of migration of an aggregate object, where an object in the destination model is equivalent to the object being migrated.

- **Different Objects Before Migration**—Equivalence Through Common Ancestry shows an aggregate object, CLIENT, with ten components in the source model and an equivalent object, CLIENT, with only three components in the destination model.

- **Same Objects After Migration**—Equivalence Through Common Ancestry illustrates what happens when the aggregate object that has ten components is selected from the source model for migration. Migration replaces the equivalent aggregate object in the destination model. Assuming that three of the ten components of the aggregate object in the source model are equivalent to the three components in the destination model, the three components of the replaced aggregate object are replaced in the destination model and the seven components that did not exist in the destination model are created. The following is a sample view:



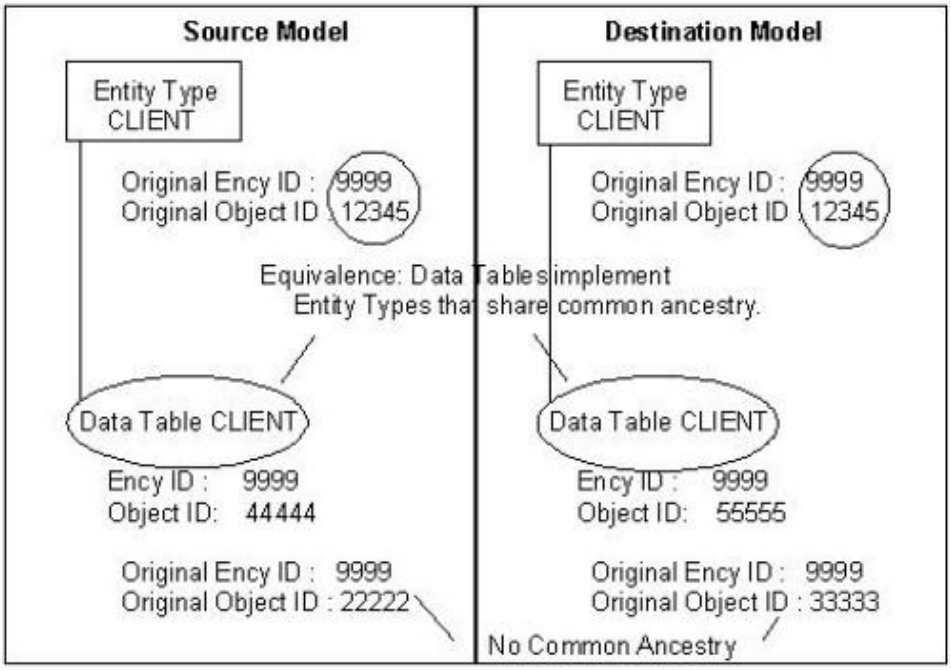


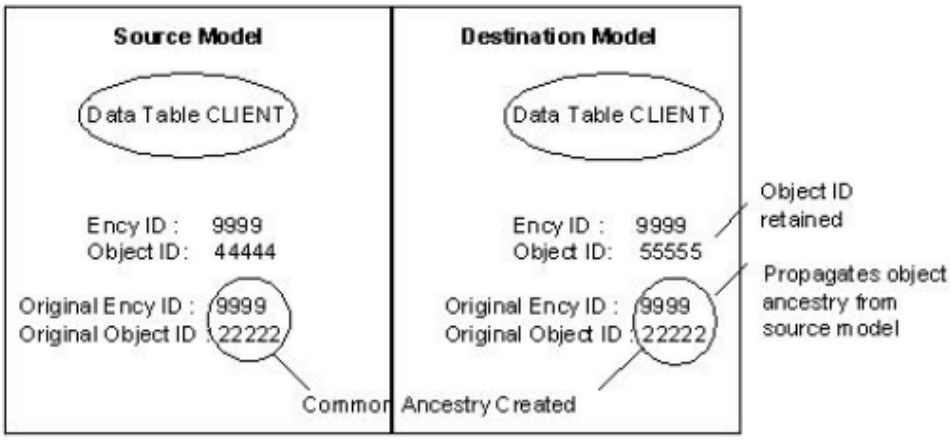
Migration Replaces Object When Special Equivalency Rule Applies

The following pair of figures provide a before and after view of migration of an aggregate object, where an object in the destination model is equivalent to the object being migrated based on Special Equivalency Rules.

- **Different Objects Before Migration**—Special Equivalency Rules Apply shows two aggregate objects, the data tables named CLIENT, that are equivalent based on Special Equivalency Rules.

- **Same Objects After Migration**—Migration Creates Common Ancestry illustrates that when such an object is migrated, migration creates common ancestry between the two objects that had previously been considered equivalent based on Special Equivalency Rules. That is, after migration, the data table CLIENT in the destination model has common ancestry with the data table CLIENT in the source model.

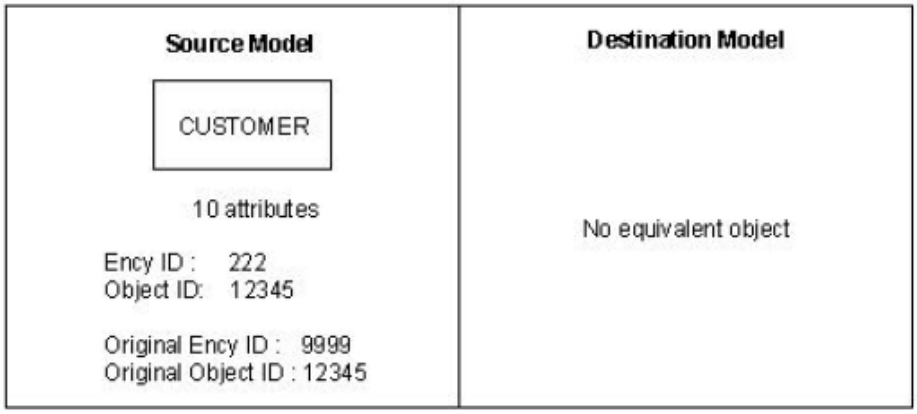


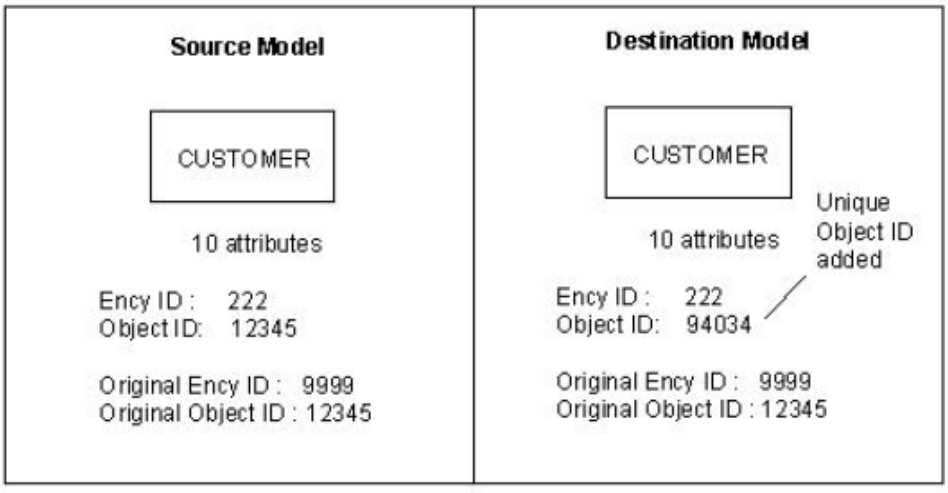


Migration Adds New Objects to Destination Model

The following pair of figures provide a before and after view of migration of an aggregate object, where the destination model does not contain an object that is equivalent to the object being migrated.

- Object That Does Not Exist in the Destination Model Before Migration shows an aggregate object named CUSTOMER that exists only in the source model before migration.
- Equivalent Objects in Source and Destination Model After Migration shows that when an object that existed only in the source model is migrated, an equivalent object is created in the destination model. In this case, CUSTOMER, and its attributes are added to the destination model.





Preparing for Migration

The following tasks are recommended to prepare for migration:

- Make sure the models satisfy Version Control prerequisites.
- Decide whether model conversion is necessary to ensure valid language code combinations or schema level combinations.
- Evaluate characteristics of the models that can affect your migration, such as model size, migration size, subsetting protection of objects you can need to access for migration, and so on.
- Generate the Compare Report and evaluate the comparison results.
- Identify objects to migrate, based on the Compare Aggregate Object Report.
- Perform needed adoption, if indicated by an analysis of the Compare Aggregate Object Report.
- Generate one or more Trial Migrate Aggregate Object Reports to identify objects requiring further analysis before performing Migration.

Consider Requirements and Recommendations

You can either verify model and authorization requirements before you begin, or you can simply proceed and take action only if you receive a message on an unmet requirement. For details on model requirements and recommendations, see Getting Ready.

Become Familiar with the Aggregate Object Types for Migration

The Aggregate Object Type List shows all the aggregate object types you can select for migration. The list of aggregate object types for migration is the same as the list of aggregate object types for Compare Report. See the tables in Become Familiar with Aggregate Object Types*:*

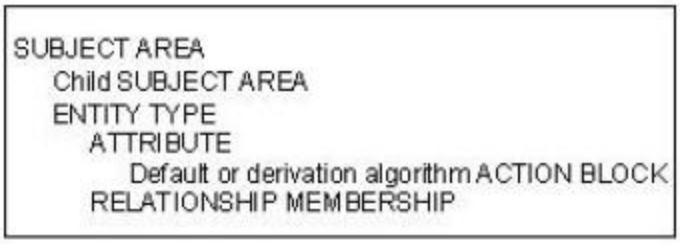
- Aggregate Object Types for Compare and Migration (Analysis)
- Aggregate Object Types for Compare and Migration (Design)
- Aggregate Object Types for Compare and Migration (Tech Design)
- Aggregate Object Types for Compare and Migration (Construction)
- Aggregate Object Types for Compare and Migration (Planning)

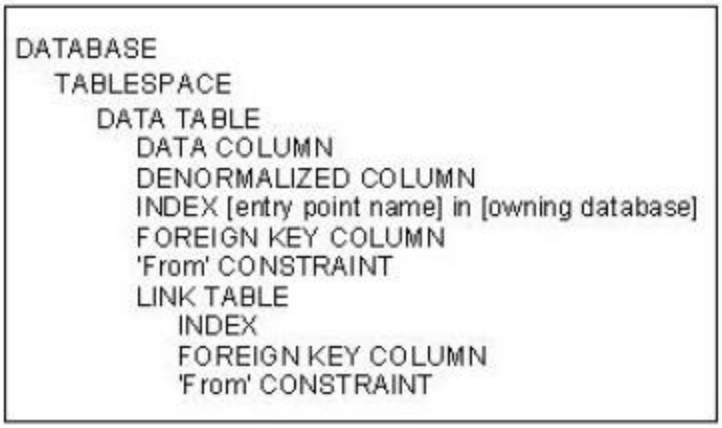
You can want to examine the expansion table for an object in the Aggregate Object Type List. The expansion table shows you the component objects, companion objects, enabling objects, and any special cases associated with the aggregate object. If so, see Aggregate Object Expansions and Special Cases.

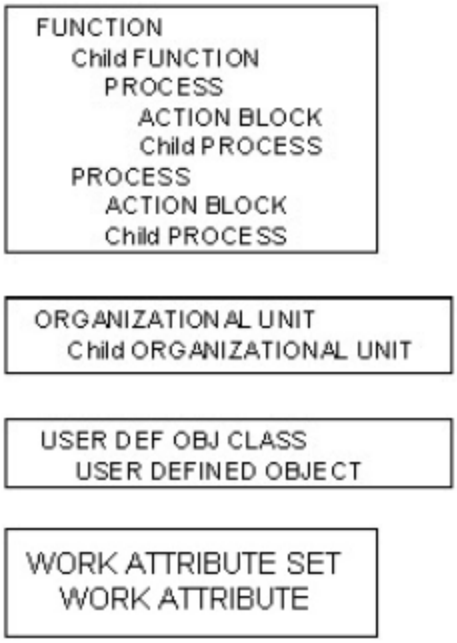
Become Familiar with Aggregate Object Expansion

Certain aggregate objects can have subordinate aggregate objects. When navigating through the selection lists to select objects to migrate, you can expand an aggregate object to display its subordinates for selection. Expansion hierarchies tell you what object types are selectable when you expand a given object type.

Each of the following graphics begins with the highest-level object in the hierarchy of subordinate aggregate objects. Each level of indentation represents a level of subordination.







Identify Models as Source and Destination

Consider carefully the direction in which to migrate, in other words, which model to specify as the source model. See Determining Which Model is the Source Model. Use the following guidelines:

Select	Model with the Object Definitions
Source Model	To retain in both models
Destination Model	To be replaced

Identify the Objects to Migrate

Always migrate at the lowest level possible. For example, if you select an entity type for comparison and the resulting report shows that the entity type has not changed but the entity type's component attributes have changed, select only the changed attributes for migration, not the entity type.

For tips on identifying objects to migrate, see How Do I Identify Objects to Migrate? For the procedure on producing a Compare Report to perform difference analysis on the models you plan to select for migration, see Performing Object Comparison.

For examples of identifying candidates for migration based on the Compare Report, see Migrate Objects.

Identify Objects to Adopt and Perform Adoptions

Strategies for identifying objects that require adoption before migration are discussed in the section, Adopt Objects. For details on how to perform adoption, see Performing Adoption or Trial Adoption.

Create or Modify Aggregate Sets for Trial Migrate and Migrate

If you are migrating objects previously defined to one or more aggregate sets, identify those sets and modify as needed. If not, create an aggregate set of objects you plan to migrate. See Creating an Aggregate Set. Use it first with Trial Migrate, then modify as needed for use with Migrate.

Performing Migration or Trial Migration

Use the following procedure as a guide when performing a Migration or Trial Migration.

Note: For syntax on the command line alternative, enter MIGRMODL from the directory where the CSE is installed.

Access the Migrate or Trial Migrate Function

Follow these steps:

1. Access Version Control. See the procedure Starting the Version Control Client in the chapter, "About Version Control."

   Select one of the following options, depending on whether you want to perform a migration or just report what would happen if a migration were performed.

   - Model, Migrate
   - Model, Reports, Trial Migrate

2. Select the source model and destination model. For considerations, see Determining Which Model is the Source Model.

   a. (Optional) Specify a filter for the Source Model names as described in Using a Filter.

   b. From the Source Model list, select the model that contains the objects you intend to migrate.

   c. (Optional) Specify a filter for the Destination Model names.

   d. From the Destination Model list, select the model that contains the objects you intend to replace with object definitions from the source model.

   e. Click OK.

3. Specify the Migration options. For *Online Help* on the field having focus, click F1. Specify the number of errors to be reported before terminating the migration, or accept the displayed default. Note that this value does not affect the success or failure of the migration. It only specifies the number of errors to report. Any error causes the migration to fail. Enter 0 (zero) to stop the migration process when the first error is encountered.

   a. Specify a report destination or accept the displayed default. For tips, see Specifying How to Save Reports.

   b. Specify a unique filename for the report or accept the displayed default.

   c. If migrating or trial migrating selected objects that belong to an aggregate set, check Aggregate Set, click OK and skip to Select Aggregate Set (Step 9).

   d. Click OK.

4. Select an aggregate object type for which you want to list occurrences. Note that the list of aggregate object types you can select for Migration is the same as the list for the Compare Report. A description of each aggregate object type is listed in a table under Become Familiar with Aggregate Object Types in this chapter.

    a. Highlight the aggregate object type of interest. Select the lowest-level aggregate object possible to migrate. This strategy reduces the number of objects that have to be migrated, as well as optimizing performance. Use the Compare Aggregate Objects Report to identify objects that have changed. Use the Trial Migration Report to determine the anticipated outcome of migrating the objects of interest. Use the expansion tables to identify component objects. If selecting a parent aggregate object, it is not necessary to select any of its component objects, since components are migrated automatically.

    b. Click List, or to limit the display to a specific range of occurrences of the selected aggregate object type, enter a filter value and click List.

5. Select one occurrence, select multiple occurrences, or click Select All. Selecting a single occurrence for expansion limits the Expanded Occurrence list to objects subordinate to the selected occurrence. This option is useful if you want to select objects of a low-level type having an extremely large number of occurrences within the model, but only a small number when accessed through the path of objects to which it is subordinate.

    - If you selected only one object and want to expand that object, click Expand. Proceed with the Aggregate Object Expanded Occurrences dialog (Step 7).
    - If you selected multiple individual occurrences, clicked Select All, or selected an object that cannot be expanded or that you do not want to expand, click Add, then click Exit. The Aggregate Object Types List appears. Either continue the selection process as described in Step 5 or click Cancel and proceed to migrate the objects you have selected, as described in Step 9 for Selected Object List.

6. Select one occurrence, select multiple occurrences individually, or click Select All. Continue in one of the following ways:

    - If you selected only one object and want to expand that object, click Expand again. Continue as described at the beginning of this step. (You can repeat this step of selecting and expanding until you reach the lowest-level object in the chain.)
    - If you selected multiple objects or selected a single object that either cannot be expanded or that you do not want to expand, click Add, then click Exit. If you performed only one expansion, the Aggregate Object Occurrences dialog with the parent object appears. If you performed multiple expansions, the Aggregate Object Expanded Occurrences dialog with the parent object appears. Click Exit repeatedly until you display the Aggregate Object Types dialog.
    - Try to migrate at the lowest level possible. Do not, for example, migrate an entity type when only an attribute has changed; migrate only the changed attribute.

7. To continue selecting objects to migrate, return to step 5. To stop the selection process, click Cancel.

   Select the aggregate set(s) to use and verify the retrieval status shows the expected number of objects retrieved.

8. Verify that the listed objects are the objects to be migrated. If so, select Proceed. (If not, see *Online Help* for other options.)

9. Select Yes to proceed with the trial migration or migration.

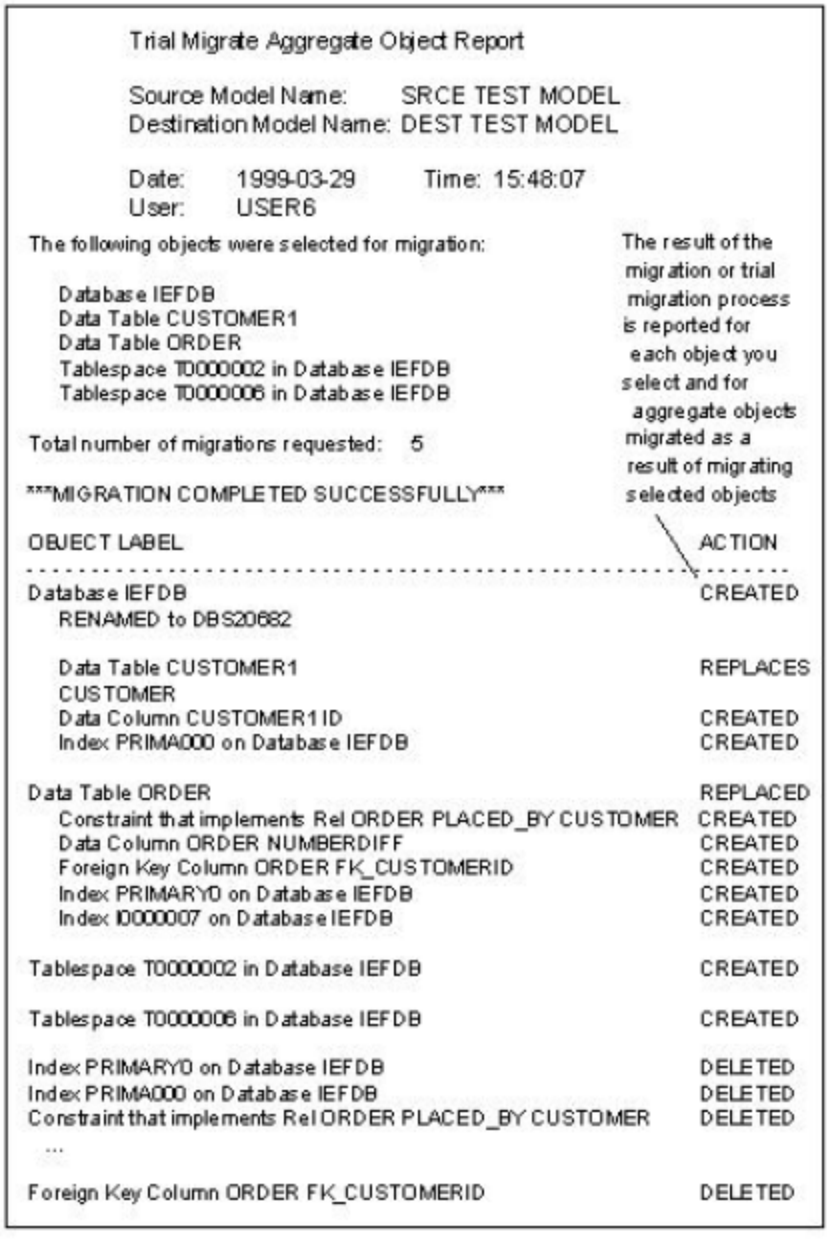
10. Review the following messages, then click Continue to generate the Migration or Trial Migration Report. The report appears in a Review panel. You can also access it from the directory you specified.

    - *n* objects processed during delete phase.
    - Where *n* is the number of objects that have been deleted from the destination model
    - *n* objects processed during copy phase.
    - Where *n* is the number of objects that have been copied from the source model to the destination model
    - Verifying protection for objects to be deleted.
    - Verifying required associations exist for object n.
    - Where *n* is the current count of associations that have been validated
    - Validating destination model.
    - Verifying property uniqueness.
    - Creating report.
    - Rolling back changes to the database (displays for a successful Trial Migration or an unsuccessful Migration)

Note: Additional messages are displayed for a failed migration or a trial migration.

Evaluating the Migration or Trial Migration Report

When you evaluate the Trial Migrate Aggregate Object Report, examine what would happen to the migrated objects, and examine any reported error messages. The following is a sample view:



Trial Migrate Aggregate Object Report

Source Model Name: SRCE TEST MODEL
Destination Model Name: DEST TEST MODEL

Date: 1999-03-29 Time: 15:48:07
User: USER6

The following objects were selected for migration:

Database IEFDB
Data Table CUSTOMER1
Data Table ORDER
Tablespace T0000002 in Database IEFDB
Tablespace T0000006 in Database IEFDB

Total number of migrations requested: 5

MIGRATION COMPLETED SUCCESSFULLY

The result of the migration or trial migration process is reported for each object you select and for aggregate objects migrated as a result of migrating selected objects

OBJECT LABEL	ACTION
Database IEFDB	CREATED
RENAMED to DBS20682	
Data Table CUSTOMER1	REPLACES
CUSTOMER	
Data Column CUSTOMER1.ID	CREATED
Index PRIMA000 on Database IEFDB	CREATED
Data Table ORDER	REPLACED
Constraint that implements Rel ORDER PLACED_BY CUSTOMER	CREATED
Data Column ORDER.NUMBERDIFF	CREATED
Foreign Key Column ORDER.FK_CUSTOMERID	CREATED
Index PRIMARY0 on Database IEFDB	CREATED
Index I0000007 on Database IEFDB	CREATED
Tablespace T0000002 in Database IEFDB	CREATED
Tablespace T0000006 in Database IEFDB	CREATED
Index PRIMARY0 on Database IEFDB	DELETED
Index PRIMA000 on Database IEFDB	DELETED
Constraint that implements Rel ORDER PLACED_BY CUSTOMER	DELETED
...	
Foreign Key Column ORDER.FK_CUSTOMERID	DELETED

The following table illustrates migration report information messages:

Action	Indicates	Object Label in Both Models
CREATED	The source model object did not exist in the destination model; therefore this object was added to the destination model. See also RENAMED, which occurs only in conjunction with CREATED.	N/A
DELETED	The source model object's components did not include this component found in the destination model; therefore, this object was deleted.	N/A
RENAMED	The source model object was added to the destination model, but renamed as specified because the destination model already contains another object with that object name; where the object with the same name is not equivalent to the object being migrated.	N/A
REPLACED	The source model object replaced the equivalent destination model object.	The same
REPLACES	The source model object replaced, and consequently renamed, the equivalent destination model object.	Different
SEE ABOVE	The object was already migrated as a consequence of migrating another object; see the previously reported action on this object.	N/A

Note: The RENAMED action on a Trial Migrate Aggregate Object Report sometimes indicates the need for adoption. See Sample Trial Migrate Aggregate Object Report.

When you examine the Trial Migrate Aggregate Object Report, you can notice certain outcomes that are different from what you expected. For example:

- Objects reported as renamed and added, when you intended that they replace the object with the same name.
- Objects reported as deleted, when you intended that they be saved.

Evaluate Renamed Objects

It is important to evaluate renamed objects on an individual basis. Objects with the same name can be related or unrelated.

Handling Renaming When Objects Are Unrelated

An object in the destination with the same name as an object being migrated can have no relationship with the object being migrated. In this case, you can want to rename either the unrelated object in the destination model or the source model object to be migrated. (This assumes you noticed this condition by examining a Trial Migration Report and that Migration had not yet been performed.) Migration, after either type of preventive intervention, will not rename the destination model object it replaces.

Handling Renaming When Objects Are Related

An object in the destination with the same name as an object being migrated can be logically the same as the object being migrated. If you want to replace such an object in the destination model with the object to be migrated, the relevant aggregate object must be made equivalent in the two models. To create object equivalency through common ancestry, perform adoption before migration.

How to Proceed If You Intended to Replace the Object

Follow these steps:

1. If you noticed the renaming on a Migrate Aggregate Object Report, delete the renamed aggregate object that was created by migration. If you noticed the renaming on a Trial Migrate Aggregate Object Report, skip this step.

   For an example, see Trial Migrate Aggregate Object Report in the chapter "Performing Adoption."

2. Use adoption to create common ancestry between the source model object to be migrated and the destination model object to be replaced.

   Migrate the aggregate object.

Evaluate Deleted Component Objects

When migration replaces an aggregate object, it also replaces the object's components. For example, if Entity Type E exists in the source and destination models and its Attribute A has been deleted only from the source model, migration deletes Attribute A from the destination model also. If Attribute A is not referenced in the destination model—has no usages—its deletion causes no problem, and the migration can proceed.

If you want to save components of aggregate objects that would be deleted by migration, perform migrations in both directions. Say, for example, you intend to migrate Entity Type E from Model A to Model B but want to keep several attributes of Entity Type E in Model B—and their permitted values. In this instance, migrate the required attributes from Model B to Model A. Then migrate the enhanced entity type from Model A to Model B. Because permitted values are components of attributes, they migrate with their respective attributes both ways.

How to Keep Components

Follow these steps:

1. Migrate the components that you want to keep to the model that contains the aggregate object you want to migrate from. To migrate components individually, the components must be aggregate objects in their own right.
2. Migrate the enhanced aggregate object (with the added components) as originally planned.

Evaluate Migration Error Messages

The migration process ensures that no changes are made to your model that would leave it in an invalid state. When errors are encountered that prevent a successful migration, the migration report contains a description of each type of problem followed by a list of occurrences that identify the objects causing the problem. The format is as follows:

- Object requires existence of related object
  - <object> REQUIRES <object>
- Attempt to reference/modify/delete object(s) which are checked out
  - <object type> <object label>
  - subset: <subset name>
  - Access: DELETE | MODIFY | ACCESS
  - userid: <userid>
- Object repeats implementation of object
  - <object> REPEATS <object>
- Object contains object
  - <object> CONTAINS <object>
- Object(s) cannot be/must be deleted
  - <object> USED BY <object>

- Object invalidates object
  - <object> INVALIDATES <object>
- Object is incomplete
  - <object> INCOMPLETE <object>

Note: In the following examples, assume equivalence exists.

Object Requires Existence of Related Object

The following is the description of this error message:

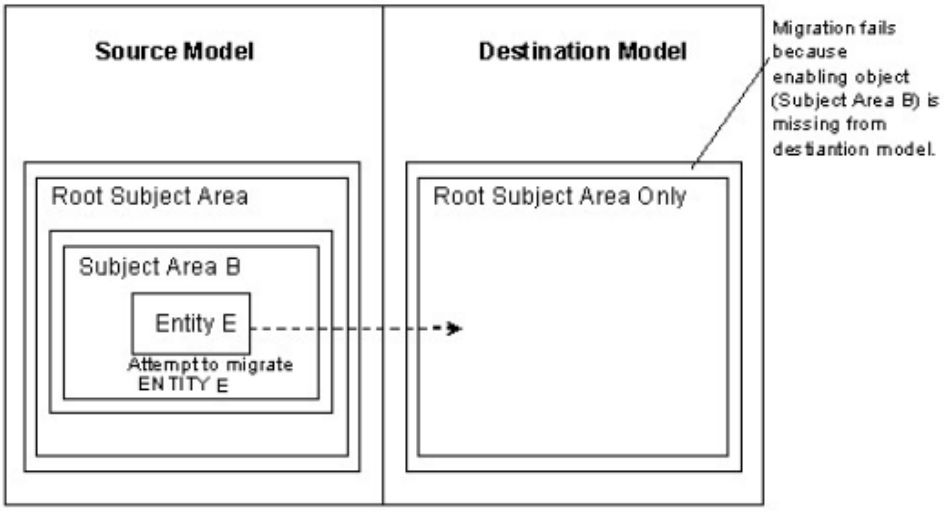
- Object requires existence of related object
  - <object> REQUIRES <object>

The related object is an enabling object, and is missing from the destination model. This means:

- The parent object of the object in the message was missing
- The aggregate object references the object listed as missing
- Object REQUIRES Object Example
- Object requires existence of related object

  Entity E ---REQUIRES

  Subject Area B



The following suggestions ensure the existence of the required enabling object.

- Verify that the enabling object exists in the destination model and is equivalent to the enabling object in the source model.
- If the enabling object does not exist in the destination model, migrate the enabling object at the same time. In this example, migrate Subject Area B and Entity E.

Attempt to Reference/Modify/Delete Object(s) which are Checked Out

The following is the description of this error message:

Attempt to reference/modify/delete object(s) which are checked out

- <object type> <object label>
- subset: <subset name>
- Access: DELETE | MODIFY | ACCESS
- userid: <userid>

This message indicates that the object listed in the message was checked out with the indicated protection during the migration. A migration can complete successfully when subsets are checked out, but migration does not:

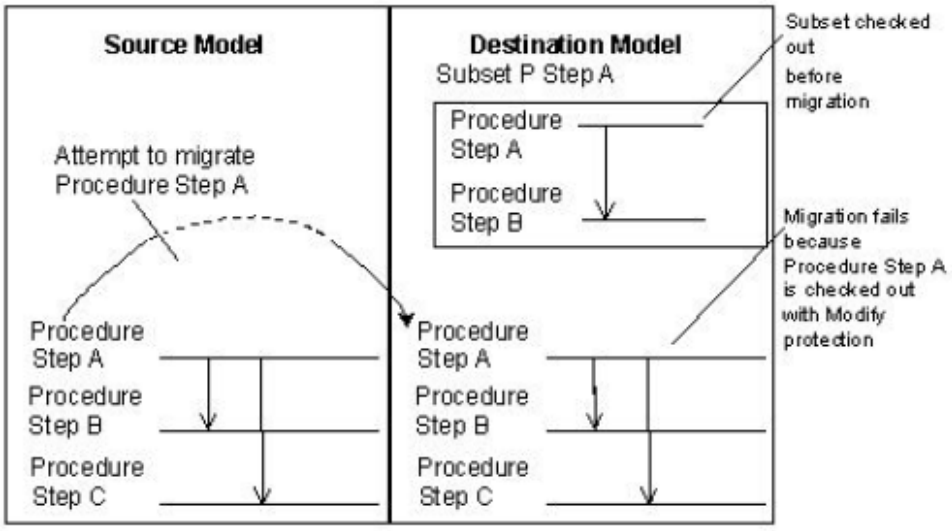
- Add or remove a referencing association to an object that is checked out with Delete protection in the destination model.
- Add or remove a modifying association if an object is checked out with Modify or Delete protection in the destination model.
- Replace an object checked out with Modify or Delete protection in the destination model.
- Delete an object checked out with Delete, Modify, or Access protection in the destination model.

The following is the description of this error message:

Attempt to reference/modify/delete object(s) which are checked out

- Procedure Step A
- subset: PSTEP A
- Access: MODIFY
- userid: JLM

The following figure references several objects including a Procedure step called Procedure Step A, which is to be migrated to the Destination Model. However, userid JLM checked out the Destination Model's Procedure Step A with an access level of Modify prior to the migration attempt.



Check in the subsets that would cause the errors.

Note: For more information about subsetting protections, see the *Client Server Encyclopedia User Guide*.

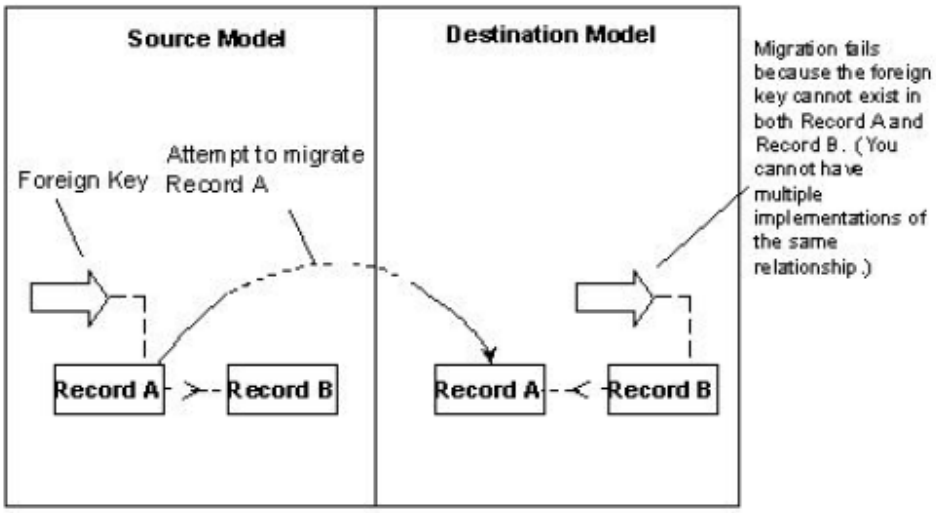
Object Repeats Implementation of Object

Object repeats implementation of object.

<object> REPEATS <object>

This message indicates that migration fails because it would have created more than one implementation of a relationship, thereby producing an invalid model.

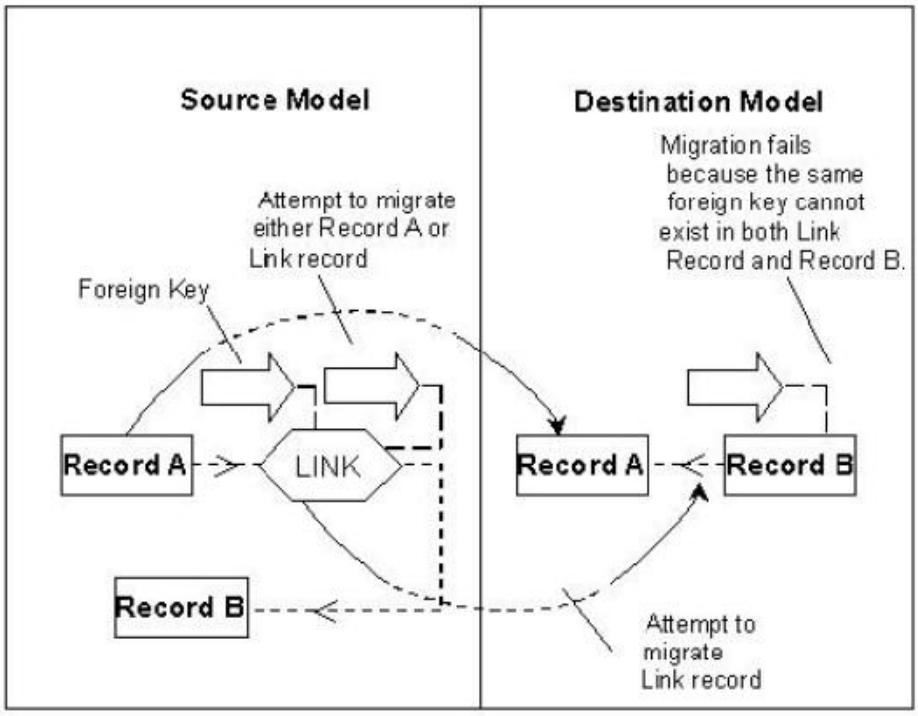
Object repeats implementation of object.



Do one of the following:

- Check the location of foreign key columns for conflicts
- Migrate data table A, data table B, and the relationship

Object repeats implementation of object.



Do one of the following:

- Check foreign key column in link table against other foreign key columns.
- Migrate data table A and data table B and relationship.

Object Contains Object

The following is a description of this error message:

Object contains object

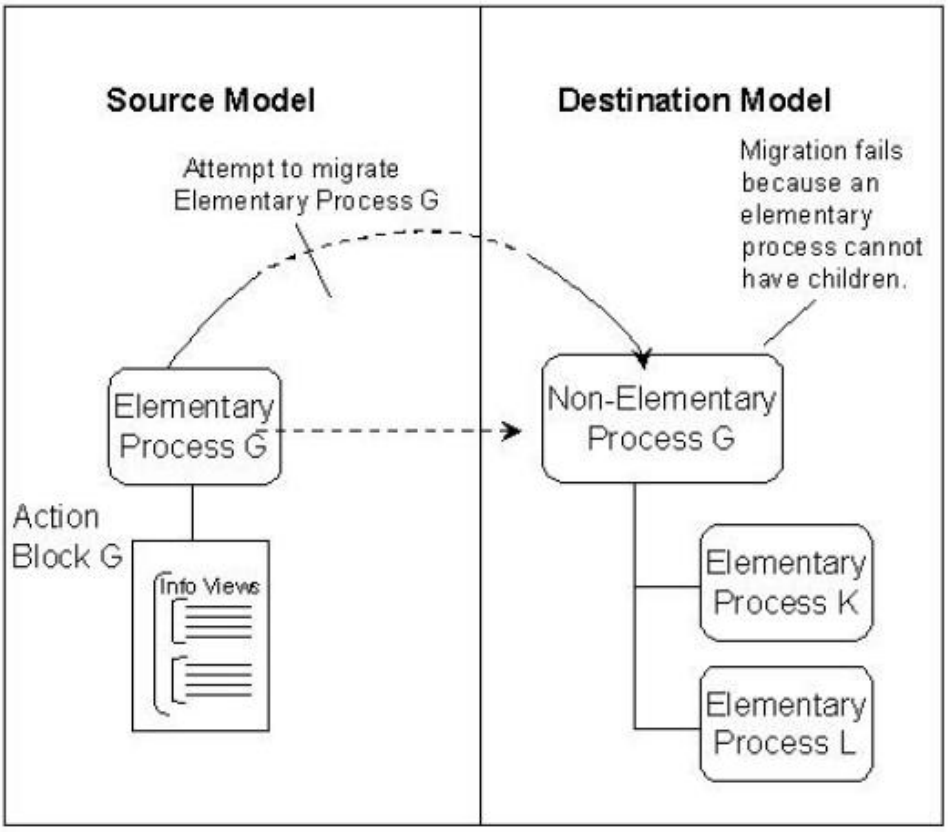
<object> CONTAINS <object>

This message indicates that migration failed because it would have produced an invalid model in which:

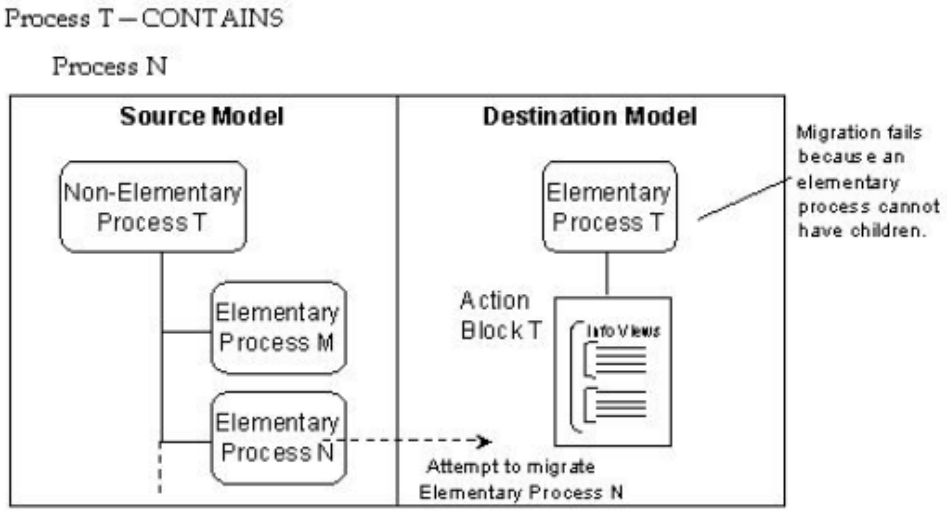
- An elementary process contained another process (a child), or
- An elementary process contains child events or external objects.

Process G—CONTAINS

Process L

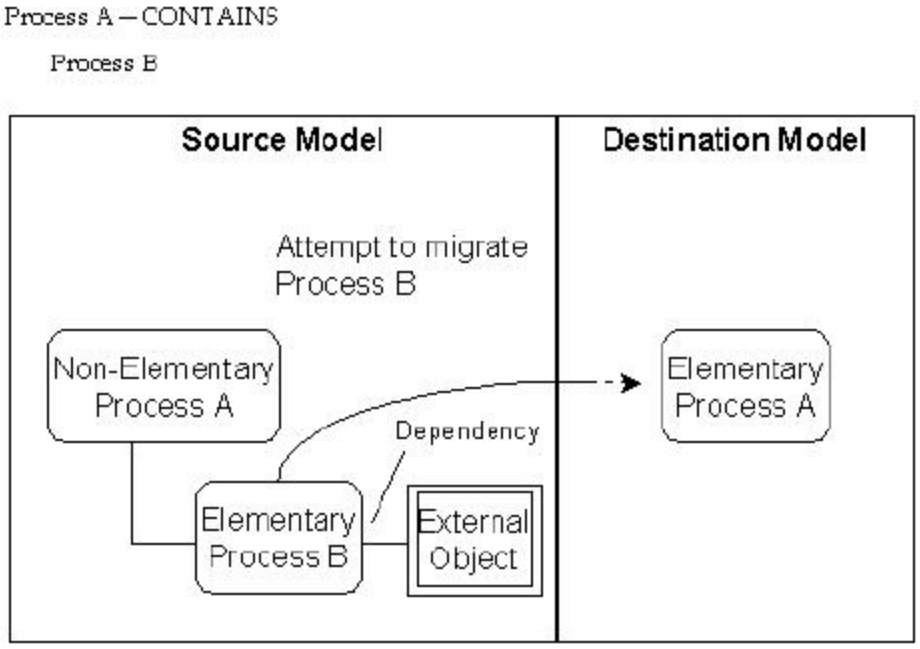


Do the following:



Delete Elementary Process K and Elementary Process L from the destination model or make Process G non-elementary in the source model.

Do the following:



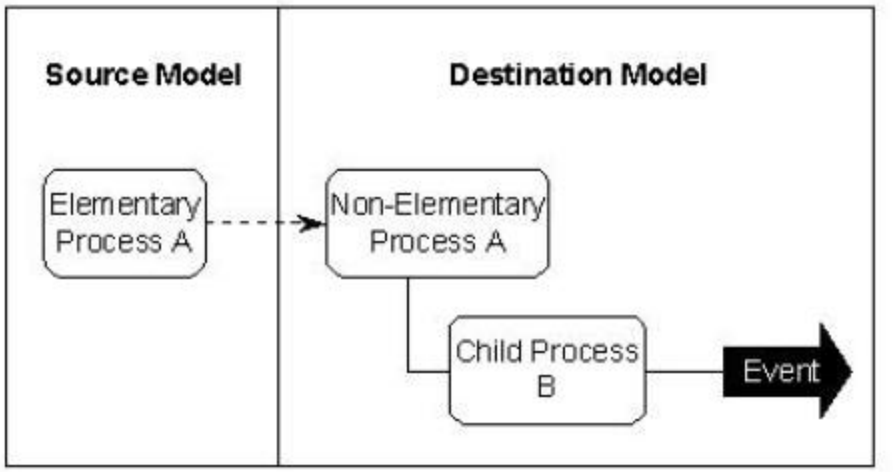
Migrate Process T with Process N.

Do the following:

Migrate Process A with Process B.

Process A—CONTAINS

Process B



Do the following:

Migrate Process A with Process B.

Object Cannot Be/Must Be Deleted

Object(s) cannot be/must be deleted

<object> USED BY <object>

This message indicates that migration tried to delete an object from the destination model that is referenced by (USED BY) another object in that model.

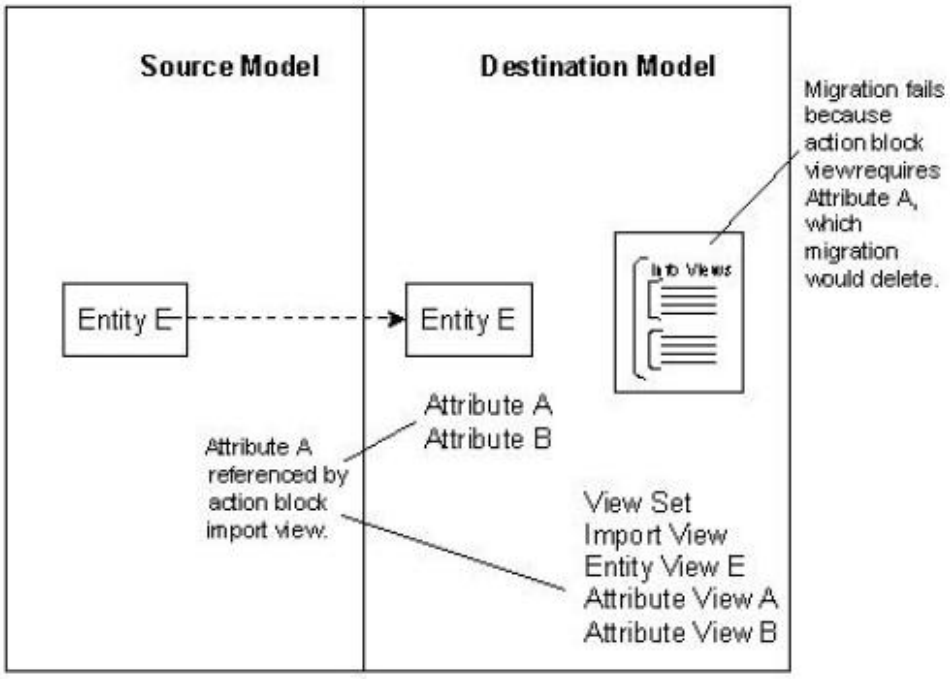
Object(s) cannot be/must be deleted Attribute Entity E A—USED BY

Attribute view Entity E A within Process Action Block B

What Happened

The migration of Entity Type E:

1. Copies Entity Type E and its component object, Attribute B, to the destination model and deletes Attribute A.
2. Tries to reestablish the link between Attributes A and B and their attribute views. Finds Attribute A missing.
3. Terminates the migration and rolls back ALL changes to the database when it cannot reestablish the link.



Do the following:

Determine whether the target object in the destination model has different components than the object being migrated. If so, remove references to those component objects in the destination model that do not exist in the source model.

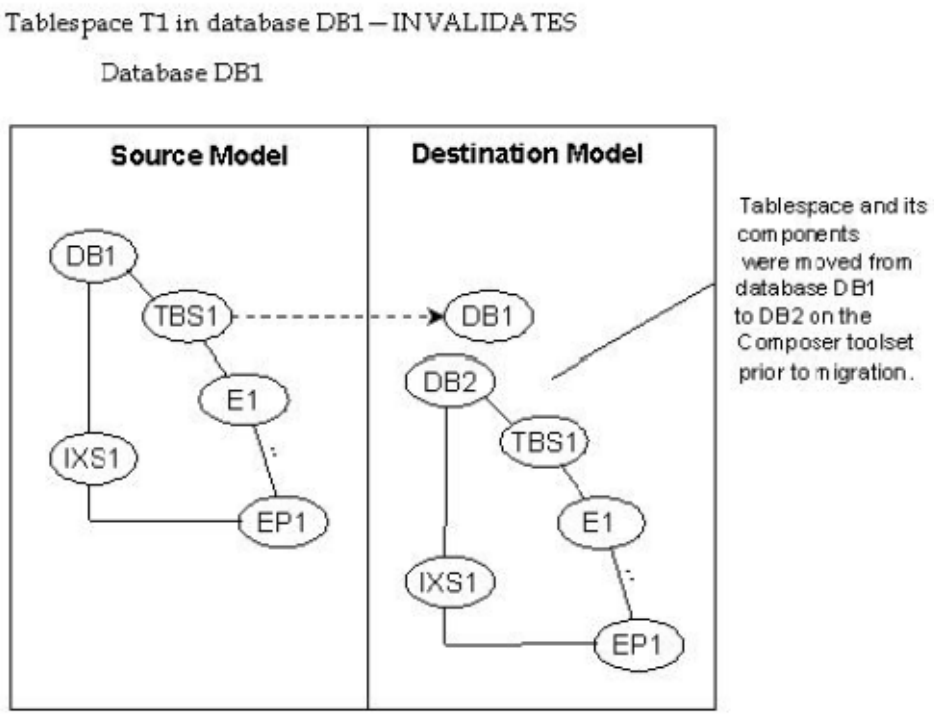
Object Invalidates Object

The following is the description of this error message:

Object invalidates object

<object> INVALIDATES <object>

This error can occur in several situations. As it applies to the following example, the error message indicates that migration fails because it would cause the tablespace to be in a different database from the indexspace.

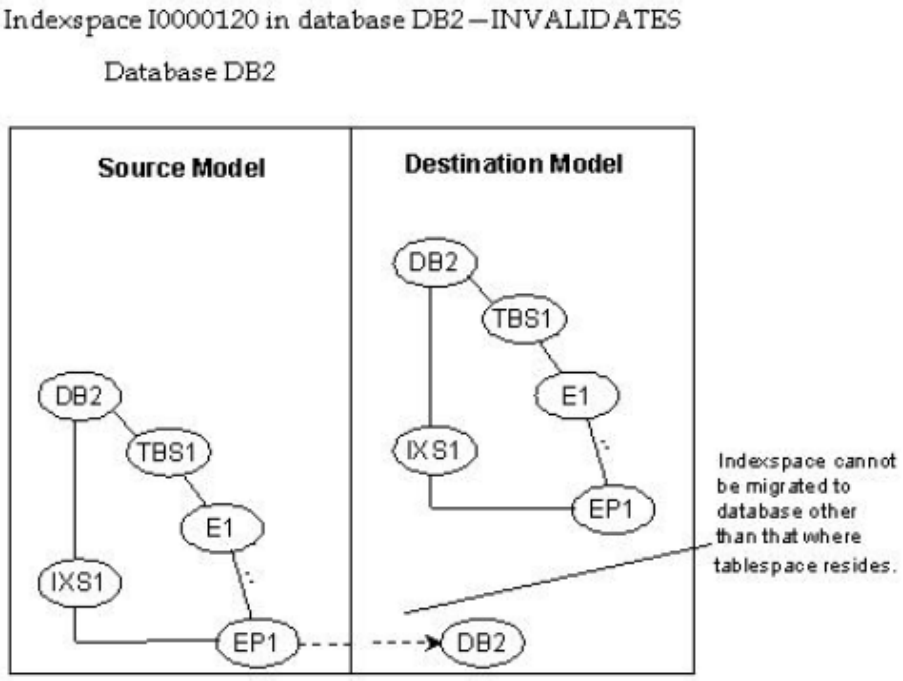


What Happened

Migrating the tablespace would cause the tablespace to be in a different database from the indexspace(s) for the data table. Therefore, the migration is not allowed to complete.

Do the following:

Migrate the data table along with the tablespace.

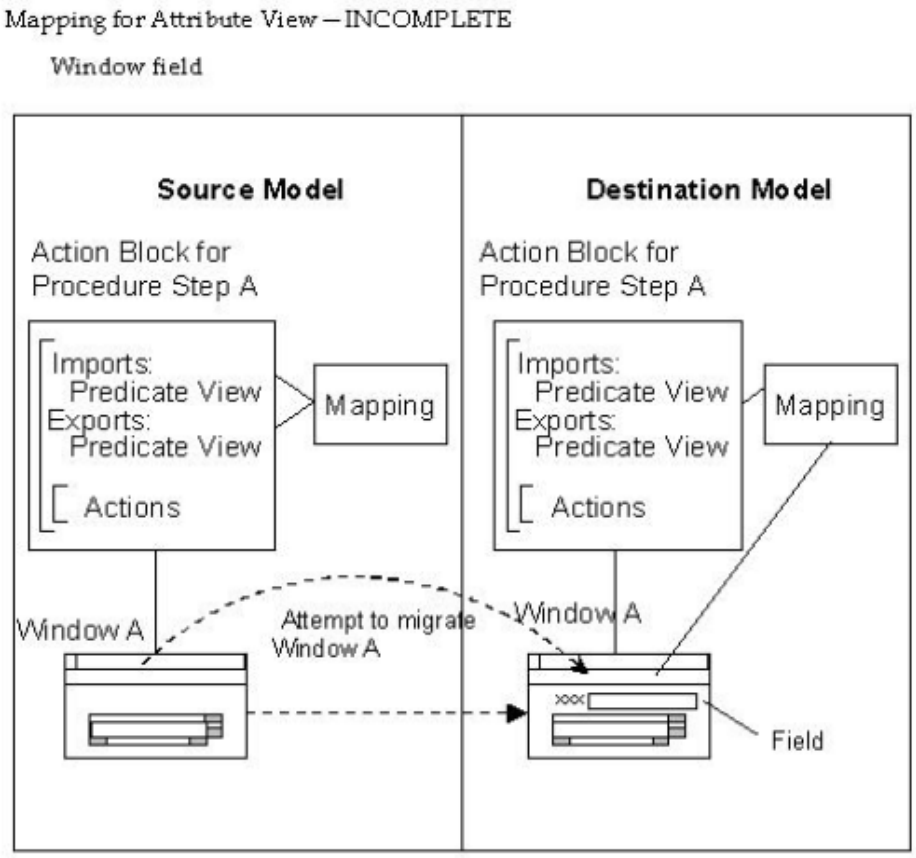


Object Is Incomplete

Object is incomplete

<object> INCOMPLETE <object>

This message indicates that migration fails because it would cause a mapping that is not associated to a window field to exist with only an export predicate view. If a mapping is not associated to a window field, it must be associated to both an export predicate view and an import predicate view.



Do the following:

Migrate Procedure Step A. If this does not fix the problem, delete the dialog or window from the Destination Model and repeat the migration.

Appendix A: Change Capture for Version Control

Change Capture for Version Control collects and saves information about changes to objects in the Client/Server Encyclopedia (CSE).

Change Capture Basics

The Compare Report uses this information to compare and report on changes to objects in two models that you select. For each object evaluated, the report identifies:

- When the last change was made (date and time)
- Who made the last change (user ID)

The change information is used to determine if equivalent objects are the same or different.

How Objects Change

Changes occur to objects on the CSE when users:

- Upload changes to a model from the toolset
- Apply changes to a parent model from a child model
- Package modules
- Generate code or databases
- Convert models
- Modify model through CSE client

Where Changes Are Marked

For every change that is captured, Change Capture marks an aggregate object as changed, whether the change was made:

- To the aggregate object itself, or
- To a non-aggregate object for which the aggregate object is a parent

Change Capture tracks changes to the same set of aggregate objects you can select for Migration or for Compare Report.

Example 1

If you add a procedure step to a model, Change Capture marks the procedure step aggregate object as changed.

For instance, if you add procedure step MAINTAIN_CUSTOMER, Change Capture marks the aggregate object procedure step labeled MAINTAIN_CUSTOMER as changed.

Example 2

If you remove a view in a procedure step action block, Change Capture marks the procedure step aggregate object and the procedure step action block aggregate object as changed because:

- Views are not aggregate objects
- Views are subordinate to both the procedure step aggregate object and the procedure step action block aggregate object

Example 3

If you update the name of a dependency in the Activity Dependency Diagram, its parent aggregate objects, the functions and processes to which it is linked, are marked as changed since a dependency is not an aggregate object.

How Changes Are Marked

Change Capture adds the session object that contains the following information for every group of changes that it captures:

- Date of change
- Time of change
- User ID of the person who made the change
- Identification number for the changing session

For details about the changing session, see How is Difference Between Equivalent Objects Detected? in the chapter, Comparing Objects in Different Models.

Information That Is Not Saved

Change Capture marks objects as changed. It does not maintain information about how an object was changed.

If you remove a view from a procedure step action block, the procedure step and procedure step action block are marked as changed. However, Change Capture does not record that a view has been removed.

Actions on Aggregate Objects Causing Them to be Marked Changed

The basic changes that cause an object to be marked as changed are as follows.

- **Add an Object**—If an object is added and it is an aggregate object, the object itself is marked as changed. If a non-aggregate object is added, its parent aggregate object is marked as changed.
- **Delete an Object**—Deleting an object involves dissociating two objects; see Remove an Association between Two Objects.
- **Modify Object Type**—If an object type is modified and it is an aggregate object, the object is marked as changed. If a non-aggregate object's type is modified, its parent aggregate object(s) are marked as changed.
- **Update Property of an Object**—If an object's property is updated, and it is an aggregate object, the object is marked as changed. If a non-aggregate object's property is updated, its parent aggregate object(s) are marked as changed.
- **Add an Association between Two Objects**—If you add an attribute to an entity, that association causes the entity to be marked as changed. Any time an association is created between two objects, a change to an aggregate object is marked.
- **Remove an Association between Two Objects**—If you remove an attribute from an entity, the deletion of that association causes the entity to be marked as changed. Any time you delete an association between two objects, a change to an aggregate object is marked.
- **Move/Reorder an Association between Two Objects**—If the order of attributes of an entity is changed, the entity is marked as changed. Any time an association is moved between two objects, a change to an aggregate object can be marked.

Actions on Related Objects Causing Aggregates to be Marked Changed

The following tables indicate when aggregate objects are marked as changed because of changes to related objects. Tables are ordered alphabetically by aggregate object name, beginning with Action Block (BAA):

Aggregate Object	Changes to Related Objects
ACTION BLOCK (BAA)	Add/Modify member name Add/Remove operation of an entity Add/Remove use of trigger module Add/Remove/Modify bind package defaults Add/Remove/Modify statement Add/Remove/Modify table usage Add/Remove/Modify view matching Add/Remove/Move views Use action block in a Business System for the first time
ACTION BLOCK (BSD)	Add view implementation Add/Modify member name Add/Remove operation of an entity Add/Remove use of trigger module Add/Remove/Modify bind package default Add/Remove/Modify statement Add/Remove/Modify table usage Add/Remove/Modify view matching Add/Remove/Move views **Note:** If the action block is for a reusable object, the reusable object is marked rather than the action block.
ATTRIBUTE	Add prompt Add prompt to permitted value Add/Modify prompt dialect text Add/Remove default value Add/Remove derivation algorithm Add/Remove reference to foreign key column by foreign key attribute Add/Remove reference of based attribute for foreign key attribute Add/Remove reference of attribute/relationship usage to foreign key attribute Add/Remove/Modify an alias Add/Remove/Modify permitted values Reorder attribute within entity

Aggregate Object	Changes to Related Objects
BATCH JOB	Remove batch job steps
BATCH JOB STEP	Add/Remove/Modify package list entry Add/Remove/Modify usage of program specification block (psb)
BUSINESS SYSTEM	Add/Modify default error field attributes Add/Modify default GUI dialog attributes Add/Modify default GUI field attributes Add/Modify default GUI group box attributes Add/Modify default GUI list box attributes Add/Modify default GUI literal attributes Add/Modify default GUI menu item attributes Add/Modify default GUI prompt attributes Add/Modify default GUI push button attributes Add/Modify default GUI status bar attributes Add/Modify default GUI toolbar attributes Add/Modify default GUI window attributes Add/Modify default literal attributes Add/Modify default normal field attributes Add/Modify default prompt attributes Add/Modify default special field attributes Add/Remove parameter delimiters Add/Remove parameter string delimiters Add/Remove default message box attributes Add/Remove custom video properties
BUSINESS SYSTEM IMPL	Add/Remove/Modify bind package default Add/Remove/Modify package list entry Add/Remove/Rename library
COMMAND	Add/Remove/Modify command synonym Add/Remove/Modify dialect sensitive text Add/Remove/Modify dialect text

Aggregate Object	Changes to Related Objects
COMPONENT IMPLEMENTATION	Add/Modify trigger Add/Remove attribute to/from identifier Add/Remove entity state transition Add/Remove mutually exclusive Add/Remove partitioning Add/Remove partitioning of subtype Add/Remove relationship to mutually exclusive Add/Remove relationship to/from identifier Add/Remove subtypes Add/Remove use of entity in content Add/Remove/Modify alias Add/Remove/Modify classifying attribute Add/Remove/Modify content for an Entity Add/Remove/Modify identifier Add/Remove/Modify parent Remove attribute Remove linkage
COMPONENT MODEL	Add/Remove an art object Add/Remove reference to subject area
COMPONENT SPECIFICATION	Add/Modify trigger Add/Remove attribute to/from identifier Add/Remove entity state transition Add/Remove mutually exclusive Add/Remove partitioning Add/Remove partitioning of subtype Add/Remove relationship to mutually exclusive Add/Remove relationship to/from identifier Add/Remove subtypes Add/Remove use of entity in content Add/Remove/Modify alias Add/Remove/Modify classifying attribute Add/Remove/Modify content for an Entity Add/Remove/Modify identifier Add/Remove/Modify parent Remove attribute Remove linkage
CONFIGURATION INSTANCE	Add/Remove references to business system, load modules, procedures steps, action blocks, windows, dialog boxes, databases, tablespaces, records and storage groups
CONSTRAINT	Add/Modify trigger Add/Remove/Modify extended constraint
CUSTOM PROXIES	Add/Remove custom proxies

Aggregate Object	Changes to Related Objects
DATA COLUMN	Add/Remove/Modify extended column
DATA TABLE	Add/Remove entity implementation Add/Remove constraint Add/Remove many to many implementation Add/Remove relationship implementation Add/Remove tablespace for extended table Add/Remove/Modify extended tables Move table to another tablespace Remove data column Remove denormalized column Remove index Remove foreign key column
DATABASE	Add/Remove tablespace for extended database Add/Remove/Modify data tables Add/Remove/Modify extended database Add/Remove/Modify log tables
DEFAULT EDIT PATTERN	Add/Remove dialect sensitive text
DENORMALIZED COLUMN	Add/Remove/Modify extended column
DFLT PRIMARY WINDOW	Add/Remove/Modify use of reusable object Add/Remove/Modify window contents
DIALOG	Add/Remove/Modify use of reusable object Add/Remove/Modify window contents
DIALOG FLOW	Add/Remove Flows on/Returns on Exit State Set/Remove commands Set/Unset autoflow command
ENTITY TYPE	Add/Modify trigger Add/Remove attribute to/from identifier Add/Remove entity state transition Add/Remove mutually exclusive Add/Remove partitioning Add/Remove partitioning of subtype Add/Remove relationship to mutually exclusive Add/Remove relationship to/from identifier Add/Remove subtypes Add/Remove use of entity in content Add/Remove/Modify alias Add/Remove/Modify classifying attribute Add/Remove/Modify content for an Entity Add/Remove/Modify identifier Add/Remove/Modify parent Remove attribute Remove linkage

Aggregate Object	Changes to Related Objects
EXIT STATE	Add/Remove dialect text Delete business system
FUNCTION	Add/Remove expected effects Add/Remove usage of activity Add/Remove/Move views Change parent In dependency diagram: Add/Remove information flow Add/Remove information view
FOREIGN KEY COLUMN	Add/Remove/Modify extended column
INDEX	Add/Modify indexspace Add/Remove data column Add/Remove denormalized column Add/Remove foreign key Add/Remove foreign key column Add/Remove storage group to/from indexspace Add/Remove tablespace for extended indexspace Add/Remove volume serial to/from indexspace dataset Add/Remove/Modify extended index Add/Remove/Modify extended indexspace Add/Remove/Modify partitioning value to/from index Change indexspace data set for indexspace Move (Add/Remove) indexspace to another database Remove constraint using foreign key columns contained in index

Aggregate Object	Changes to Related Objects
INTERFACE TYPE	Add/Modify trigger Add/Remove attribute to/from identifier Add/Remove entity state transition Add/Remove mutually exclusive Add/Remove partitioning Add/Remove partitioning of subtype Add/Remove relationship to interface type model Add/Remove relationship to mutually exclusive Add/Remove relationship to/from identifier Add/Remove subtypes Add/Remove use of entity in content Add/Remove/Modify alias Add/Remove/Modify classifying attribute Add/Remove/Modify content for an Entity Add/Remove/Modify identifier Add/Remove/Modify parent Remove attribute Remove linkage
LINK TABLE	Add many to many implementation Add/Remove tablespace for extended table Add/Remove/Modify extended tables Move many to many implementation to another tablespace
MATRIX	Add/Remove cell values
NAVIGATION DIAGRAM	Add/Remove a window usage Add/Remove a procedure step window usage
ONLINE LOAD MODULE	Add/Remove/Modify package list entry
OPERATIONS LIBRARY	Add/Remove/Modify action blocks.
ORGANIZATIONAL UNIT	Change/Reorder parent
PROCEDURE	Remove procedure steps

Aggregate Object	Changes to Related Objects
PROCEDURE STEP	Add/Remove packaging for pstep Add/Remove PF Key overrides Add/Remove unformatted input Add/Remove view implementation Add/Remove/Modify trancode associated to a procedure step Add/Remove/Modify view matching Add/Remove/Move views Remove dialog Remove "initiates" (source) dialog flow Remove screen Remove window Add/Remove web operation
PROCESS	Add/Remove expected effects Add/Remove usage from state transition to process Add/Remove usage of activity Add/Remove/Move views In dependency diagram: Add/Remove information flow Add/Remove information view Change/Reorder parent
SCREEN	Add screen implementation Add/Remove/Modify screen contents
SCROLL AMOUNT VALUE	Add/Remove dialect text
SERVER MANAGER	Add/Remove/Modify package list entry
SPECIFICATION TYPE	Add/Modify trigger Add/Remove attribute to/from identifier Add/Remove entity state transition Add/Remove mutually exclusive Add/Remove partitioning Add/Remove partitioning of subtype Add/Remove relationship to mutually exclusive Add/Remove relationship to/from identifier Add/Remove subtypes Add/Remove use of entity in content Add/Remove/Modify alias Add/Remove/Modify classifying attribute Add/Remove/Modify content for an Entity Add/Remove/Modify identifier Add/Remove/Modify parent Remove attribute Remove linkage

Aggregate Object	Changes to Related Objects
STORAGE GROUP	Add/Remove volume serial
SYSTEM PF KEY	Add/Remove association with command
TABLESPACE	Add/Remove database partition Add/Remove storage group Add/Remove volume serial Add/Remove/Modify dataset Add/Remove/Modify extended tablespace Add/Remove/Modify partitioning value to/from index Move data table to another tablespace Move tablespace to another database
TECH DESIGN DEFAULT	Add/Remove/Modify bind package default Add/Remove/Modify extended technical design Add/Remove/Modify package list entry Add/Remove/Rename trigger library
TRANS OPERATION	Add/Remove/Modify a constraint Add/Remove/Modify an external parameter Add/Remove delegation to another transaction Add/Remove reference to a command Add/Remove reference to an entity type Add/Remove reference to a procedure step Add/Remove reference to a work attribute set
TYPEMAP	Add/Remove reference to correspondences Add/Remove reference to action blocks
USER DEF OBJ CLASS	Remove user-defined object
WEB SERVICE DEFINITION	Add/Remove web service
WINDOW LOAD MODULE	Add/Remove/Modify package list entry
WORK ATTRIBUTE SET	Add/Modify prompt dialect text Add/Remove of alias definition Remove work attribute
z/OS LIBRARY	Add/Remove action blocks

Appendix B: Migration Rules

The following three sets of rules are documented in this appendix:

- **Aggregate Object Expansions Tables**—Aggregate object expansion is the process of determining which objects will be migrated along with the selected aggregate and whether proper conditions exist for the migration to succeed. The Aggregate Object Expansion and Special Cases tables list all expandable aggregate object types for migration. For each type of aggregate object, the table lists the:
  - Component objects
  - Companion objects
  - Enabling objects
  - Special cases
- **Special Case Aggregate Actions**—The actions related to the special cases are documented with the relevant aggregate objects.
- **Special Equivalency Rules**—This is a table of rules for establishing equivalence between aggregate objects that exist in different models and are meant to be equivalent but do not share common ancestry. Objects that are deemed equivalent based on one of these rules are treated by Compare Report and Migration as though they share common ancestry, even though they do not. Common ancestry is propagated by Migration when objects are replaced during migration based on equivalence established by these rules.

Aggregate Objects with No Expansions

The following aggregate objects are not expandable and are not listed with Aggregate Object Expansions and Special Cases:

- ACTIVITY CLUSTER
- BUSINESS AREA
- CRIT SUCCESS FACTOR
- CURRENT DATA STORE
- DATA CLUSTER
- DIALECT
- ENVIRONMENT
- EXTERNAL OBJECT
- FACILITY

- GOAL
- INFORMATION NEED
- LOCATION
- OBJECTIVE
- PERFORMANCE MEASURE
- STRATEGY
- TACTIC
- USER DEF OBJ CLASS

Aggregate Object Expansions and Special Cases

The expansions for aggregate objects that are expandable or have Special Cases are listed alphabetically.

Action Block

The description for action blocks is given in the following table:

Component Objects	Companion Objects	Enabling Objects
Action statements, views, DBRM, implementation unit, and bind package defaults.	Referenced commands, exit states, operations library, z/OS library, and action blocks USED by action block	Relationships, entity types, subtypes, attributes, work attributes, and work attribute sets referenced by any migrated views and/or action statements. Owning entity type or work attribute set. Windows and dialogs referenced by action statements. Business system if migrated action block is implemented into a business system. For BSD, window controls referenced by window control usages.

Special Cases

The following is a list of special cases:

- A view match to a view in a USED action block is migrated only if the corresponding view in the USED action block already exists in the destination model. Conversely, a view match from a view in an action block that USES the migrated action block is migrated only if the corresponding view in the using action block already exists in the destination model.
- If the action block being migrated is an elementary process action block and the process exists in the destination model (but is not elementary), the elementary process will automatically be migrated.
- If the action block being migrated is an elementary process action block and the process does NOT exist in the destination model, the elementary process will automatically be migrated.

Special Case Examples: Data Views and View Matching When USING Action Block Is Migrated

In the following situations, Action Block A is an action block that is directly or indirectly selected to be migrated and Action Block B is another action block.

- View Match Copied
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the source model between a view in Action Block A and a view in Action Block B.
  - Action Block B exists in the destination model.
  - The View in Action Block B exists in the destination model.
  - The View Match does not exist in the destination model.
  - Result: The View Match will be copied to the destination model.
- View Match Replaced
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the source model between a view in Action Block A and a view in Action Block B.
  - Action Block B exists in the destination model.
  - The View in Action Block B exists in the destination model.
  - The View Match exists in the destination model.
  - Result: The View Match will be replaced in the destination model.

- View Match Not Copied
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the source model between a view in Action Block A and a view in Action Block B.
  - The Action Block B exists in the destination model.
  - The View in the Action Block B does not exist in the destination model.
  - Result: The View Match will not be copied to the destination model.
- View Match Deleted
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the destination model between a view in Action Block A and a view in Action Block B.
  - The View in Action Block A does not exist in the source model
  - OR the View Match does not exist in the source model.
  - Result: The View Match will be deleted from the destination model.

Special Case Examples: Data Views and View Matching When USED Action Block Is Migrated

In the following situations, Action Block B is an action block that is directly or indirectly selected to be migrated, and Action Block A is another action block.

- View Match Copied
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the source model between a view in Action Block A and a view in Action Block B.
  - Action Block A exists in the destination model.
  - Action Block A uses Action Block B in the destination model.
  - The View in Action Block A exists in the destination model.
  - The View Match does not exist in the destination model.
  - Result: The View Match will be copied to the destination model.

- View Match Replaced
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the source model between a view in Action Block A and a view in Action Block B.
  - Action Block A exists in the destination model.
  - Action Block A uses Action Block B in the destination model.
  - The View in Action Block A exists in the destination model.
  - The View Match exists in the destination model.
  - Result: The View Match will be replaced in the destination model.
- View Match Not Copied
  - Action Block A uses Action Block B in the source model.
  - A View Match exists in the source model between a view in Action Block A and a view in Action Block B.
  - Action Block A does not exist in the destination model OR
  - Action Block A does not use Action Block B in the destination model OR
  - The View in Action Block A does not exist in the destination model.
  - Result: The View Match will not be copied to the destination model.
- View Match Deleted
  - Action Block A uses Action Block B in the source model.
  - Action Block A uses Action Block B in the destination model.
  - A View Match exists in the destination model between a view in Action Block A and a view in Action Block B.
  - The View in Action Block B does not exist in the source model OR the View Match does not exist in the source model.
  - Result: The View Match will be deleted from the destination model.
- View Match Not Changed
  - Action Block A does not exist in the source model OR Action Block A does not use Action Block B in the source model.
  - Action Block A uses Action Block B in the destination model.
  - A View Match exists in the destination model between a view in Action Block A and a view in Action Block B.
  - Result: The View Match will not be changed in the destination model.

Attribute

The following table describes the attributes:

Component Objects	Companion Objects	Enabling Objects
Permitted values and aliases Attribute relationship usage (foreign key attribute)	Default and derivation algorithm action blocks and referenced dialects	Parent entity type or subtype Foreign key columns attribute is implemented by, foreign key attribute or attribute derived from relationships, relationships referenced by component attribute, relationship usages

Special Cases

The following is a list of special cases:

- Prompts are created if they exist in the source model and do not already exist in the destination model. Prompts are replaced if they exist in both the source and destination model. Prompts are unchanged if they exist in the destination model but not in the source model.
- Prompt values are created if they exist in the source model and do not already exist in the destination model. Prompt values are replaced if they exist in both the source and destination models. Prompt values are unchanged if they exist in the destination model but not in the source model.

Special Case Examples: Attributes and Prompts

In the following situations, Attribute A is selected to be migrated:

- Prompt Created
  - Prompt A exists for Attribute A in the source model.
  - Attribute A exists in the destination model, but Prompt A is not in the destination model.
  - Result: Prompt A is created in the destination model.
- Prompt Not Changed
  - Attribute A does not have Prompt A in the source model.
  - Attribute A has Prompt A in the destination model.
  - Result: Prompt A is not changed in the destination model.

- Prompt Replaced
  - Attribute A has Prompt A in the source model.
  - Attribute A has Prompt A in the destination model.
  - Result: Prompt A is replaced in the destination model.

Special Case Examples: Permitted Values and Prompt Values

In the following situation, Attribute A is selected to be migrated:

- Prompt Value Created
  - Attribute A has Permitted Value A, which has Prompt Value A, in the source model.
  - Attribute A and Permitted Value A exist in the destination model.
  - Prompt Value A does not exist in the destination model.
  - Result: Prompt Value A is copied to the destination model.
- Prompt Value Not Changed
  - Attribute A has Permitted Value A in the source model.
  - Attribute A and Permitted Value A exist in the destination model.
  - Prompt Value A exists in the destination model for Permitted Value A.
  - Prompt Value A does not exist in the source model.
  - Result: Prompt Value A is not changed in the destination model.
- Prompt Value Replaced
  - Attribute A has Permitted Value A, which has Prompt Value A, in the source model.
  - Attribute A, Permitted Value A, and Prompt Value A exist in the destination model.
  - Result: Prompt Value A is replaced in the destination model.

Batch Job

The following table describes the batch job:

Component Objects	Companion Objects	Enabling Objects
Batch job steps		Associated procedure

Batch Job Step

The following table describes the batch jog step:

Component Objects	Companion Objects	Enabling Objects
Packaging, package list entries, PSB (program specification block)		Parent batch job

Special Cases

To migrate a batch job step, select batch job step, or migrate the procedure or the batch job.

Business System

The following table describes business system:

Component Objects	Companion Objects	Enabling Objects
Literal properties, error properties, prompt properties, field and special field properties, default GUI attributes, and business system implementation.	Default edit patterns, system-wide program function keys, and custom video properties.	None

Business System Implementation

The following table describes business system implementation:

Component Objects	Companion Objects	Enabling Objects
Target environment parameters which are properties of a business system implementation object; bind package default and package list entry	Referenced libraries	Parent business system

Command

The following table describes the command:

Component Objects	Companion Objects	Enabling Objects
Command synonyms	Referenced dialects	Parent business system

Component Implementation

The following table describes the component implementation:

Component Objects	Companion Objects	Enabling Objects
None	None	Parent subject area

Component Model

The following table describes the component model:

Component Objects	Companion Objects	Enabling Objects
Art objects	None	Scoping subject area, attributes, and action blocks

Component Specification

The following table describes the component specification:

Component Objects	Companion Objects	Enabling Objects
Offerings	None	Parent subject area, interface type offered by component specification

Configuration Instance

The following table describes the Configuration Instance:

Component Objects	Companion Objects	Enabling Objects
Configuration Instance		Referenced business system, load modules, procedure steps, action blocks, windows, dialog boxes, databases, tablespaces, records, and storage groups

Constraint

The following table describes the constraint:

Component Objects	Companion Objects	Enabling Objects
Extended constraints	Source foreign key column	Source and target data tables, target data column, identifier (if linkage implements an identifying relationship), and relationship implemented by the linkage

Note: Only constraints between data tables are selectable.

Current Information System

The following table describes the current information system:

Component Objects	Companion Objects	Enabling Objects
Current effect	None	Current database or store

Custom Proxies

The following table describes the web service definition:

Component Objects	Companion Objects	Enabling Objects
Custom proxy	None	Business system

Note: The association is recreated in the destination model between the Custom Proxy and the Interface if the Interface already exists in the destination model.

Custom Video Property

The following table describes the custom video property:

Component Objects	Companion Objects	Enabling Objects
None	None	Business System

Data Column

The following table describes the data column:

Component Objects	Companion Objects	Enabling Objects
Extended columns	None	Parent data table and attribute being implemented by the data column

Data Table

The following table describes the data table:

Component Objects	Companion Objects	Enabling Objects
Indexes, constraints from data tables, related link tables, relationship triggers, columns, extended tables for migrated constraints.	None	Parent tablespace and entity type being implemented by the data table, tablespaces referenced by extended tables

Database

The following table describes the database:

Component Objects	Companion Objects	Enabling Objects
Extended databases and database files	Technical design and default storage groups	Tablespaces referenced by extended databases

Default Dialog

The following table describes the default dialog:

Component Objects	Companion Objects	Enabling Objects
Window controls, custom edit patterns, and GUI events.	Referenced prompts, prompt values, commands, default edit patterns, and dialects.	Parent procedure step, views, and GUI event handlers referenced by window controls.

Note: Window controls are the objects contained in a window or dialog.

Default Edit Pattern

The following table describes the default edit pattern:

Component Objects	Companion Objects	Enabling Objects
None	Referenced dialects	Parent business system

Default Primary Window

The following table describes the default primary window:

Component Objects	Companion Objects	Enabling Objects
Window controls, custom edit patterns, and GUI events.	Referenced prompts prompt values, commands, default edit patterns, and dialects	Parent procedure step, views, and GUI event handlers referenced by window controls

Note: Window controls are objects contained in a window or dialog.

Denormalized Column

The following table describes the denormalized column:

Component Objects	Companion Objects	Enabling Objects
Extended columns	None	Parent data table, attribute and relationship being implemented by the denormalized column

Dialog Flow

The following table describes the dialog flow:

Component Objects	Companion Objects	Enabling Objects
None	Flows on, returns on exit states and commands	From and to procedure steps

Note: View matching is migrated with the procedure step not with the dialog flow.

Special Case Examples: Procedure Step and Dialog Flow

In the following situations, Procedure Step A is a procedure step that is directly or indirectly selected to be migrated and Procedure Step B is another procedure step.

- Dialog Flow Copied
  - A Dialog Flow flows from/to Procedure Step A in the source model.
  - The Dialog Flow flows to/from Procedure Step B in the source model.
  - Procedure Step B exists in the destination model.
  - The Dialog Flow does not exist in the destination model.
  - Result: The Dialog Flow is copied to the destination model.

- Dialog Flow Replaced
  - A Dialog Flow flows from/to Procedure Step A in the source model.
  - The Dialog Flow flows to/from Procedure Step B in the source model.
  - Procedure Step B exists in the destination model.
  - The Dialog Flow exists in the destination model.
  - Result: The Dialog Flow is replaced in the destination model.
- Dialog Flow Not Copied
  - A Dialog Flow flows from/to Procedure Step A in the source model.
  - The Dialog Flow flows to/from Procedure Step B in the source model.
  - Procedure Step B does not exist in the destination model.
  - Result: The Dialog Flow is not copied to the destination model.
- Dialog Flow Deleted
  - A Dialog Flow flows from/to Procedure Step A in the destination model.
  - The Dialog Flow does not exist in the source model.
  - Result: The Dialog Flow is deleted from the destination model.

Entity Type

The following table describes the entity type:

Component Objects	Companion Objects	Enabling Objects
Aliases, attributes, foreign key attributes, child subtypes, classifiers, identifiers, partitionings, entity state changes, entity state transitions, entity triggers, and contents.	None	Parent subject area, identifying relationships, entity types and work attribute sets used in component contents.

Note: To delete a relationship from an identifier, migrate the relationship with the entity type, then delete the relationship from the identifier.

Special Cases

Mutually exclusives are migrated if at least two of the relationships participating in the mutually exclusive exist in the destination model.

Special Case Examples: Entities and Mutually Exclusives

In the following situation, Entity A is an entity that is selected to be migrated.

- Mutually Exclusive Copied
  - Entity A contains a Mutually Exclusive A that has Relationship A and Relationship B as participants.
  - Entity A exists in the destination model.
  - Relationship A and Relationship B exist in the destination model.
  - Result: Mutually Exclusive A is copied to the destination model.
- Mutually Exclusive Not Copied
  - Entity A contains a Mutually Exclusive A that has Relationship A and Relationship B as participants.
  - Entity A exists in the destination model.
  - Relationship A exists in the destination model.
  - Relationship B does not exist in the destination model.
  - Result: Mutually Exclusive A is not copied to the destination model.

Event

The following table describes the event:

Component Objects	Companion Objects	Enabling Objects
None	None	None

Special Case

The migration of an event will not create the event in the destination model if there are no usages of the event.

Exit State

The following table describes the exit state:

Component Objects	Companion Objects	Enabling Objects
None	Referenced dialects	Parent business system (unless this is a global exit state)

Special Case

If migration of exit states changes their Code Generation (CG) value, you must regenerate all action blocks that use the migrated exit states.

Foreign Key Column

The following table describes the foreign key column:

Component Objects	Companion Objects	Enabling Objects
Extended columns	None	Parent data table, linkage, attribute, source column (data or foreign key)

Function

The following table describes the function:

Component Objects	Companion Objects	Enabling Objects
Views, expected effects	Dependent external objects, dependent events	Parent function (unless aggregate function is a root function). Entity types, subtypes, attributes, work attribute sets and work attributes referenced by any migrated views. Entity types and subtypes referenced by migrated expected effects.

Special Cases

The following is a list of special cases:

- Imported or exported dependency migrated only if the other functions importing or exporting the dependency exist in the destination model.
- A mutually exclusive or parallel dependency is migrated if the input function and at least two of the output functions participating in the dependency exist in the destination model.
- A closure dependency is migrated if the mutually exclusive dependency that is being closed and at least two of the input functions and the output function exist in the destination model.

Index

The following table describes the index:

Component Objects	Companion Objects	Enabling Objects
Indexspace, extended indexes, and extended indexspaces	None	Identifier (if an identifier-based index), parent data table, parent database, referenced data columns. Relationships if identifier contains relationships. Tablespaces referenced by component extended indexspaces.

Interface Type

The following table describes the interface type:

Component Objects	Companion Objects	Enabling Objects
Aliases, attributes and their components, child subtypes and their components, classifiers, identifiers, partitionings, entity state changes and transitions, entity triggers, contents, and interface type model. Art objects for interface type model.	None	Parent subject area, identifying relationships, entity types, and work attribute sets used in component contents. Entity types, entity subtypes, partitionings, and relationships referenced by interface type model.

ISP Matrix

The following table describes the ISP matrix:

Component Objects	Companion Objects	Enabling Objects
Cell values	User-defined class if the X and/or Y axis of the matrix is a user-defined class.	None

Special Case

A row or column in the matrix is migrated only if the aggregate object that the row or column references already exist in the destination model.

Link Table

The following table describes the link table:

Component Objects	Companion Objects	Enabling Objects
Indexes, constraints, columns, and extended tables	None	Source and destination data tables, tablespace, and the relationship implemented by the link table, tablespaces referenced by extended tables

Navigation Diagram

The following table describes the navigation diagram:

Component Objects	Companion Objects	Enabling Objects
Usages of windows, dialogs, and procedure steps	None	Referenced windows, dialogs and procedure steps

Online Load Module

The following table describes the online load module:

Component Objects	Companion Objects	Enabling Objects
Package list entries	None	None

Special Case

The migration of an online load module or the migration of packaging for an online load module, can create an invalid situation in the destination model, where a load module would not contain packaging for any procedure steps. If this situation is detected, migration will delete the load module to prevent the corruption.

Operations Library

The following table describes the operations library:

Component Objects	Companion Objects	Enabling Objects
None	None	Technical System

Organizational Unit

The following table describes the organizational unit:

Component Objects	Companion Objects	Enabling Objects
None	None	Parent organization unit (unless this is the root organization unit)

Packaging for Pstep

The following table describes the packaging for Pstep:

Component Objects	Companion Objects	Enabling Objects
Transaction codes	Load module (windowed, online, cooperative, or batch)	Associated procedure step

Note: The clear screen association is recreated in the destination model between the transaction code and packaging object if the transaction code is not already the clear screen transaction code for another procedure step in the destination model.

Special Case

If the migration of packaging causes an online load module to be unused in the destination model, the load module will be deleted.

Procedure

The following table describes the procedure:

Component Objects	Companion Objects	Enabling Objects
Procedure steps	None	Parent business system

Procedure Step

The following table describes the procedure step:

Component Objects	Companion Objects	Enabling Objects
Procedure step action diagram, screen, dialogs, primary window, views, local program function keys, unformatted input information, and packaging, and web operations.	System-wide program function keys referenced by local program function keys, and custom video properties.	Parent procedure, entity types, subtypes, attributes, work attribute sets and work attributes referenced by migrated views

Note: Packaging is the inclusion of a procedure step in a load module.

Note: The association is recreated in the destination model between the Web Operation and the Web Service if the Web Service already exists in the destination model.

Special Cases

- A dialog flow from the procedure step is migrated only if the procedure step at the other end of the dialog flow already exists in the destination model. A view match along the dialog flow is migrated only if an equivalent view exists in the destination model.

- Dialog flows that only exist in the destination model will be deleted only if the source and target procedure steps exist in both models. Dialog flows will be created/replaced if the source and target procedure steps exist in both models.

For examples of special cases involving dialog flows, see the Special Case Examples: Procedure Step and Dialog Flow.

Special Case Examples: Data Views and View Matching between Procedure Steps

In the following situations, Procedure Step A is a Procedure Step that is directly or indirectly selected to be migrated and Procedure Step B is another Procedure Step.

- View Match Copied
  - A Dialog Flow flows from/to Procedure Step A to/from Procedure Step B in the source model.
  - A View Match exists in the source model between a view in Procedure Step A and a view in Procedure Step B.
  - Procedure Step B exists in the destination model.
  - The View in Procedure Step B exists in the destination model.
  - The View Match does not exist in the destination.
  - Result: The View Match will be copied to the destination model.
- View Match Replaced
  - A Dialog Flow flows from/to Procedure Step A to/from Procedure Step B in the source model.
  - A View Match exists in the source model between a view in Procedure Step A and a view in Procedure Step B.
  - Procedure Step B exists in the destination model.
  - The View in Procedure Step B exists in the destination model.
  - The View Match exists in the destination model.
  - Result: The View Match will be replaced in the destination model.
- View Match Not Copied
  - A Dialog Flow flows from/to Procedure Step A to/from Procedure Step B in the source model.
  - A View Match exists in the source model between a view in Procedure Step A and a view in Procedure Step B.
  - The Procedure Step B exists in the destination model.
  - The View in Procedure Step B does not exist in the destination model.
  - Result: The View Match will not be copied to the destination model.

- View Match Deleted
  - A Dialog Flow flows from/to Procedure Step A to/from Procedure Step B in the destination model.
  - A View Match exists in the destination model between a view in Procedure Step A and a view in Procedure Step B.
  - One of the following:
    - The Dialog Flow does not exist in the source model *OR*
    - The View in Procedure Step A does not exist in the source model *OR*
    - The View Match does not exist in the source model.
  - Result: The View Match will be deleted from the destination model.

Process

The following table describes the process:

Component Objects	Companion Objects	Enabling Objects
usages Views, expected effects, event dependencies and information flows, process action block (if the process is an elementary process), and entity state transition usages.	Dependent external objects, dependent events	Parent process or function, entity types, subtypes, attributes, work attribute sets, and work attributes referenced by migrated views. Entity types and subtypes referenced by migrated expected effects.

Special Cases

The following is a list of special cases:

- Imported or exported dependency is migrated only if the other processes importing or exporting the dependency exist in the destination model.
- A mutually exclusive or parallel dependency is migrated if the input process and at least two of the output processes that participate in the dependency exist in the destination model.
- A closure dependency is migrated if the mutually exclusive dependency that is being closed and at least two of the input processes and the output process exist in the destination model.

Special Case Examples: Data Views and View Matching between Processes

In the following situations, Process A is a process that is directly or indirectly selected to be migrated and External Object B is another external object:

- View Match Copied
  - An Information Flow exists between Process A and External Object B in the source model.
  - A View Match exists in the source model between a view in Process A and External Object B.
  - The View Match does not exist in the destination model.
  - Result: The View Match will be copied to the destination model.
- View Match Replaced
  - An Information Flow exists between Process A and External Object B in the source model.
  - A View Match exists in the source model between a view in Process A and External Object B.
  - The View Match exists in the destination model.
  - Result: The View Match will be replaced in the destination model.
- View Match Deleted
  - An Information Flow exists between Process A and External Object B in the destination model.
  - A View Match exists in the destination model between a view in Process A and External Object B.
  - One of the following:
    - External Object does not exist in the source model
    - Information Flow does not exist in the source model
    - View in Process A does not exist in the source model
    - View Match does not exist in the source model.
  - Result: The View Match will be deleted from the destination model.

Special Case Examples: Processes and Dependencies

In the following situations, Process A is a process that is directly or indirectly selected to be migrated and Processes B and C are other processes:

- Dependency Copied
  - A Dependency exists between Process A and Process B in the source model.
  - Process B exists in the destination model.
  - The Dependency does not exist in the destination model.
  - Result: The Dependency will be copied to the destination model.
- Dependency Replaced
  - A Dependency exists between Process A and Process B in the source model.
  - Process B exists in the destination model.
  - The Dependency exists in the destination model.
  - Result: The Dependency will be replaced in the destination model.
- Mutually Exclusive or Parallel Dependency Copied
  - A Mutually Exclusive or Parallel Dependency exists between Process A and Processes B and C in the source model.
  - Processes B and C exist in the destination model.
  - The Dependency does not exist in the destination model.
  - Result: The Dependency will be copied to the destination model.
- Mutually Exclusive or Parallel Dependency Replaced
  - Parallel Dependency Replaced
  - A Mutually Exclusive or Parallel Dependency exists between Process A and Processes B and C in the source model.
  - Processes B and C exist in the destination model.
  - The Dependency exists in the destination model.
  - Result: The Dependency will be replaced in the destination model.
- Mutually Exclusive or Parallel Dependency Not Copied
  - A Mutually Exclusive or Parallel Dependency exists between Process A and Processes B and C in the source model.
  - Process B does not exist in the destination model *OR*
  - Process C does not exist in the destination model *OR*
  - Processes B and C do not exist in the destination model.
  - Result: The Dependency will not be copied in the destination model.

- Mutually Exclusive or Parallel Dependency Deleted
  - A Mutually Exclusive or Parallel Dependency exists between:
  - Process A and Processes B and C in the destination model.
  - Process B does not exist in the source model *OR*
  - Process C does not exist in the source model *OR*
  - Processes B and C do not exist in the source model *OR*
  - The Dependency does not exist in the source model.
  - Result: The Dependency will be deleted in the destination model.
- Closure Dependency Copied
  - A Closure Dependency exists between Processes A and B and Process C in the source model OR a Closure Dependency exists between Processes B and C and Process A in the source model.
  - Processes B and C exist in the destination model.
  - The Mutually Exclusive Dependency that is closed by the Closure Dependency exists in the destination model.
  - The Closure Dependency does not exist in the destination model.
  - Result: The Dependency will be copied to the destination model.
- Closure Dependency Replaced
  - A Closure Dependency exists between Processes A and B and Process C in the source model OR a Closure Dependency exists between Processes B and C and Process A in the source model.
  - Processes B and C exist in the destination model.
  - The Mutually Exclusive Dependency that is closed by the Closure Dependency exists in the destination model.
  - The Closure Dependency exists in the destination model.
  - Result: The Dependency will be replaced in the destination model.
- Closure Dependency Not Copied
  - A Closure Dependency exists between Processes A and B and Process C in the source model OR a Closure Dependency exists between Processes B and C and Process A in the source model.
  - Process B does not exist in the destination model *OR*
  - Process C does not exist in the destination model *OR*
  - The Mutually Exclusive Dependency that is closed by the Closure Dependency does not exist in the destination model.
  - Result: The Dependency will not be copied to the destination model.

- Closure Dependency Deleted
  - A Closure Dependency exists between Processes A and B and Process C in the destination model OR a Closure Dependency exists between Processes B and C and Process A in the destination model. The Closure does not exist in the source model.
  - Result: The Dependency will be deleted in the destination model.

Relationship Membership

The following table describes the relationship membership:

Component Objects	Companion Objects	Enabling Objects
None	None	Source and destination entity type or subtype

Special Cases

Mutually exclusives are migrated if at least two of the relationships participating in the mutually exclusive exist in the destination model.

Special Case Examples: Relationships and Mutually Exclusives

In the following situation, Relationship A is a relationship that is selected to be migrated:

- Mutually Exclusive Copied
  - Relationship A participates in Mutually Exclusive A with Relationship B.
  - Relationship A and Relationship B exist in the destination model.
  - Result: Mutually Exclusive A is copied to the destination model.
- Mutually Exclusive Not Copied
  - Relationship A participates in Mutually Exclusive A with Relationship B.
  - Relationship A exists in the destination model.
  - Relationship B does not exist in the destination model.
  - Result: Mutually Exclusive A is not copied to the destination model.

Screen

The following table describes the screen:

Component Objects	Companion Objects	Enabling Objects
Literals, fields, special fields, and custom edit patterns	Referenced templates, dialects, default edit patterns, and prompts.	Parent procedure step, views referenced by screen fields

For examples of special cases, see Special Case Examples: Attributes and Prompts.

Scroll Amount Value

The following table describes the scroll amount value:

Component Objects	Companion Objects	Enabling Objects
None	Referenced dialects	None

Server Manager

The following table describes the server manager:

Component Objects	Companion Objects	Enabling Objects
Package list entries	None	None

Specification Type

The following table describes the specification type:

Component Objects	Companion Objects	Enabling Objects
Aliases, attributes and their components, child subtypes and their components, classifiers, identifiers, partitionings, entity state changes and transitions, entity triggers, and contents.	None	Parent subject area, identifying relationships, entity types and work attribute sets used in component contents.

Storage Group

The following table describes the storage group:

Component Objects	Companion Objects	Enabling Objects
DASD volume	Technical design	None

Subject Area

The following table describes the subject area:

Component Objects	Companion Objects	Enabling Objects
None	None	Parent subject area (unless aggregate object is root subject area)

System PF Key

The following table describes the system PF key:

Component Objects	Companion Objects	Enabling Objects
None	Referenced commands	Parent business system

System Work Attribute Set

The following table describes the system work attribute set:

Component Objects	Companion Objects	Enabling Objects
System attributes	None	None

Tablespace

The following table describes the tablespace:

Component Objects	Companion Objects	Enabling Objects
Data sets, extended tablespaces, and extended data sets	Storage groups	Parent database

Technical Design Default

The following table describes the technical default design:

Component Objects	Companion Objects	Enabling Objects
Extended technical design defaults, bind package defaults and package lists	None	None

Template

The following table describes the template:

Component Objects	Companion Objects	Enabling Objects
Literals and special fields	Referenced templates, dialects, prompts, and default edit patterns	Parent business system

Transaction Operation

The following table describes the transaction orientation:

Component Objects	Companion Objects	Enabling Objects
Constraints and external parameters	Referenced commands	Owning entity type or work attribute set, procedure step or delegated to transaction and views referenced by component constraints

Typemap

The following table describes the typemap:

Component Objects	Companion Objects	Enabling Objects
Correspondences	None	Action block

User Defined Object

The following table describes the user defined object:

Component Objects	Companion Objects	Enabling Objects
None	User defined object class	None

Web Service Definition

The following table describes the web service definition:

Component Objects	Companion Objects	Enabling Objects
Web service	None	Business system

Note: The association is recreated in the destination model between the Web Service and the Web Operation if the Web Operation already exists in the destination model.

Window Load Module

The following table describes the window load module:

Component Objects	Companion Objects	Enabling Objects
Package list entries	None	None

Special Cases

The migration of a window load module or the migration of packaging for a window load module can create an invalid situation in the destination model, where a load module would not contain any packaging for procedure steps. If this situation is detected, migration will delete the load module to prevent the corruption.

Work Attribute

The following table describes the work attribute:

Component Objects	Companion Objects	Enabling Objects
Permitted values	Referenced dialects	Parent work attribute set

Special Cases

The following is a list of special cases:

- Prompts are created if they exist in the source model and do not already exist in the destination model. Prompts are replaced if they exist in both the source and destination models. Prompts are unchanged if they exist in the destination model but not in the source model.
- Prompt values are created if they exist in the source model and do not already exist in the destination model. Prompt values are replaced if they exist in both the source and destination models. Prompt values are unchanged if they exist in the destination model but not in the source model.

Work Attribute Set

The following table describes the work attribute set:

Component Objects	Companion Objects	Enabling Objects
Work attributes	None	None

z/OS Library

The following table describes the z/OS library:

Component Objects	Companion Objects	Enabling Objects
None	None	Technical System

Special Equivalency Rules

Special Equivalency Rules establish equivalence between aggregate objects that exist in different models and are meant to be equivalent but do not share common ancestry.

Occurrences Aggregate Objects	Equivalent When
ACTION BLOCK (BAA)	Are associated to the same elementary process
BATCH JOB	Implement the same procedure
BATCH JOB STEP	Implement the same procedure step and belong to the same batch job
BUSINESS SYSTEM IMPLEMENTATION	Are associated to the same business system
CONSTRAINT	Implement the same relationship and have the same source and target tables
DATA COLUMN	Implement the same attribute
DATA TABLE	Implement the same entity type
DEFAULT DIALOG	Are primary for the same procedure step A dialog can be equivalent to a primary window if each is primary for the pstep.
DENORMALIZED COLUMN	Implement the same attribute and are denormalized along the same relationship
DFLT PRIMARY WINDOW	Are both primary for the same procedure step A primary window can be equivalent to a dialog if each is primary for the pstep.
DIALECT	Are both the default dialect for the model
FOREIGN KEY COLUMN	Are associated to the same parent table, same source column and same constraint
FUNCTION	Are both the root function for the model

Occurrences Aggregate Objects	Equivalent When
INDEX	One of the following: Are associated to the same identifier and neither index implements a relationship Are associated to the same identifier and implement the same relationship Implement the same relationship and neither index is associated to an identifier
LINK TABLE	Implement the same relationship
ORGANIZATIONAL UNIT	Are both the root organizational unit for the model
PKG FOR PSTEP	Are associated to the same procedure step and the same type of load module
SCREEN	Are associated to the same procedure step
SCROLL AMOUNT VALUE	Have the same name
TECH DESIGN DEFAULT	Exist in both models

Appendix C: Adoption Rules

This appendix contains the rules used to determine whether the source model can adopt the aggregate objects that you select for adoption. Adoption of an object depends on the source model containing an object that is deemed to be equivalent to an object in the destination model, based on the adoption rules for that object type.

How Adoption Rules Are Used

When a selected object is adopted, all of its components are evaluated for adoption, based on adoption criteria for their respective object type. Then components of components are evaluated for adoption. When an object in the destination model is adopted, it is assigned the Original Encyclopedia ID and Original Object ID of the object in the source model that is logically the same. Matching IDs establishes equivalence between two objects.

Global Rules for Non-Adoption

An object in the destination model cannot be adopted if:

- It already has common ancestry with an object in the source model. (No action is required)
- Another object in the destination model already has common ancestry with the object in the source model that meets the adoption rule. (Two objects in the same model cannot have the same Original Encyclopedia ID and Original Object ID.)

Adoption Rules Tables

The Adoption Rules appear in two tables, one for selectable objects and the other for non-selectable objects.

- Selectable objects are the aggregate objects on the adoption selection list.
- Non-selectable objects can be adopted only by adopting their parent aggregate object.

Component objects in adoption are selectable if they are aggregate objects for adoption. Otherwise, they are non-selectable. Aggregate objects for adoption appear on the Adoption selection list. Object types that are not aggregates for adoption do not. Attribute, a component of Entity Type is selectable; Batch Job, a component of Procedure, is not. To adopt an Attribute, you would select it directly; to adopt a Batch Job, you would select the parent Procedure.

Adoption Rules for Selectable Objects

This set of tables contains the adoption criteria for aggregate objects on the Adoption selection list. Adoption rules, or criteria, for components of selected objects are listed in this set of tables if the component object is selectable; if not selectable, see Adoption Rules for Non-Selectable Objects.

Selected Object	Adoption Rule
ACTION BLOCK (BAA) See Adoption Rule for: Data view group Group view Entity view Implementation unit	Source model contains BAA common Action Block with same name. (BAA common Action Block also refers to default and derivation algorithms)
ACTION BLOCK (BSD) See Adoption Rule for: Data view group Group view Entity view Implementation unit	Source model contains BSD common Action Block with same name.
ACTIVITY CLUSTER	Source model contains Activity Cluster with same name.
ATTRIBUTE See Adoption Rule for: Permitted value	Source model contains Attribute with same name. Parent Entity Types (or subtypes) share common ancestry.
BUSINESS AREA	Source model contains Business Area with same name.
BUSINESS SYSTEM See Adoption Rule for: Default Edit Pattern System PF key Technical system	Source model contains Business System with same name.
COMMAND	Source model contains Command with same name. Parent Business Systems share common ancestry.
COMPONENT MODEL	Source model contains Component Model with same name.
CONFIGURATION INSTANCE	Source model contains Configuration Instance with same name.

Selected Object	Adoption Rule
CONSTRAINT See Adoption Rule for: TD Action Block	Source model contains Constraint with "From" table, "To" table, and associated relationship that share common ancestry with the corresponding tables and associated relationship belonging to this Constraint.
CRIT SUCCESS FACTOR	Source model contains Crit Success Factor with same name.
CURRENT DATA STORE	Source model contains Current Data Store with same name.
CURRENT INFO SYSTEM	Source model contains Current Info System with same name.
CUSTOM PROXIES	Source model contains Custom Proxy with same name (first 32 characters only). Associated Business Systems share common ancestry.
CUSTOM VIDEO PROPERTY	Source model contains Custom Video Property with same name. Parent Business Systems share common ancestry.
DATA CLUSTER	Source model contains Data Cluster with same name.
DATA COLUMN	Source model contains Data Column with a parent table and associated Attributes that share common ancestry with this Data Column's parent Data Table and associated Attributes.
DATA TABLE See Adoption Rule for: Link table Index Data Column Foreign Key Column Denormalized Column "From" Constraint	Source model contains Data Table with associated Entity Type that shares common ancestry with this Data Table's Entity Type.
DATABASE	Source model contains Database with same name.
DEFAULT EDIT PATTERN	Source model contains Default Edit Pattern with same name and type. Parent Business Systems share common ancestry.

Selected Object	Adoption Rule
DIALECT (non-default)	Source model contains Non-default Dialect with same name.
DIALOG FLOW	Source model contains Dialog Flow with source and destination procedure step that shares common ancestry with this Dialog Flow's source and destination procedure step. At least one associated exit state shares common ancestry with an associated exit state of the Dialog Flow on the source model.
COMPONENT IMPLEMENTATION See Adoption Rule for: Attribute Identifier Subtype TD Action Block	Source model contains Component Implementation with same name.
COMPONENT MODEL	Source model contains Component Model with same name. Scoping subject areas share common ancestry.
COMPONENT SPECIFICATION See Adoption Rule for: Attribute Identifier Subtype TD Action Block	Source model contains Component Specification with same name.
ENTITY TYPE See Adoption Rule for: Attribute Identifier Subtype TD Action Block	Source model contains Entity type with same name.
ENVIRONMENT	Source model contains Environment with same name.
EVENT	Source model contains Event with same name.
EXIT STATE	Source model contains Exit State with same name.
EXTERNAL OBJECT	Source model contains External Object with same name.

Selected Object	Adoption Rule
FACILITY	Source model contains Facility with same name.
FOREIGN KEY ATTRIBUTE	Source model contains Foreign Key Attribute with parent Entity Types (or subtypes), derived from attribute (or foreign key attribute) and all relationships referenced by associated attribute relationship usages that share common ancestry with this attribute's parent Entity Types or subtypes, derived from attribute (or foreign key attribute) and all relationships referenced by associated attribute relationship usages
FOREIGN KEY COLUMN	Source model contains Foreign Key Column with parent Data Table, source column, and Constraint that share common ancestry with this Foreign Key Column's parent Data Table, source column, and Constraint.
FUNCTION (non-root) See Adoption Rule for: Data view group Group view Entity view	Source model contains Non-root function with same name.
GOAL	Source model contains Goal with same name.
INDEX (identifier-based)	Source model contains Identifier-based Index with an associated table and associated identifiers that share common ancestry with this Index's associated table and associated identifiers. Both Indexes either implement a relationship with common ancestry or do not implement a relationship.
INFORMATION NEED	Source model contains Information Need with same name.
INTERFACE TYPE See Adoption Rule for: Attribute Identifier Subtype TD Action Block Component Model	Source model contains Interface Type with same name.

Selected Object	Adoption Rule
LINK TABLE See Adoption Rule for: Foreign Key Column Constraint Index	Source model contains Link Table with associated data tables and relationship that share common ancestry with this Link Table 's associated Data Tables and relationship.
LOCATION	Source model contains Location with same name.
NAVIGATION DIAGRAM	Source model contains Navigation Diagram with same name.
OBJECTIVE	Source model contains Objective with same name.
ONLINE LOAD MODULE	Source model contains Online Load Module with same member name and type (cooperative or non-cooperative). Associated Business Systems share common ancestry.
OPERATIONS LIBRARY	Source model contains Operations Library with same member name. Associated Business Systems share common ancestry.
ORGANIZATIONAL UNIT (non-root)	Source model contains Non-root Organizational Unit with same name.
PERFORMANCE MEASURE	Source model contains Performance Measure with same name.
PROCEDURE See Adoption Rule for: Procedure step Batch Job	Source model contains Procedure with same name. Parent Business Systems share common ancestry.
PROCEDURE STEP See Adoption Rule for: Procedure Step Action Block Source Dialog Flow Screen Default primary window Default dialog Batch Job Step Procedure step usage	Source model contains Procedure Step with same name. Parent Procedures share common ancestry.

Selected Object	Adoption Rule
PROCESS See Adoption Rule for: Data view group Group view Entity view Process Action Block	Source model contains Process with same name.
RELATIONSHIP MEMBERSHIP See Adoption Rule for: Inverse relationship membership	Source model contains Relationship Membership with same name and the inverse relationship with the same name. Source and destination Entity Types (or subtypes) share common ancestry.
SERVER MANAGER	Source model contains Server Manager with same member name. Associated Business Systems share common ancestry.
SPECIFICATION TYPE See Adoption Rule for: Attribute Identifier Subtype TD Action Block	Source model contains Specification Type with same name.
STORAGE GROUP	Source model contains Storage Group with same name.
STRATEGY	Source model contains Strategy with same name.
TABLESPACE	Source model contains Tablespace with same name. Parent Databases share common ancestry.
TACTIC	Source model contains Tactic with same name.
TEMPLATE	Source model contains Template with same name. Parent Business Systems share common ancestry.

Selected Object	Adoption Rule
TRANS OPERATION	For delegating transaction objects: Source model contains delegating transaction objects with the same name. Parent Entity Type (or Work Attribute Sets) share common ancestry. Delegated to transaction operations with common ancestry For non-delegating transaction objects: Source model contains non-delegating transaction objects with the same name. Parent Entity Types (or Work Attribute Sets) share common ancestry. Associated procedure steps with common ancestry
TYPEMAP	Source model contains Type Map with the same name.
USER DEF MATRIX	Source model contains User-Defined Matrix with the same name. Associated object classes on the x and y-axis share common ancestry.
USER DEF OBJ CLASS See Adoption Rule for: User-defined object	Source model contains User Defined Object Class with same name.
USER DEFINED OBJECT	Source model contains User Defined Object with the same name. Parent User Defined Object Classes share common ancestry.
USER SUBJECT AREA	Source model contains User Subject Area with same name.
WEB SERVICE DEFINITION	Source model contains Web Service Definition with same name (first 32 characters only). Associated Business Systems share common ancestry.
WINDOW LOAD MODULE	Source model contains Window Load Module with same member name and type (cooperative or non-cooperative). Associated Business Systems share common ancestry.

Selected Object	Adoption Rule
WORK ATTRIBUTE See Adoption Rule for: Permitted value	Source model contains Work Attribute with same name. Parent Work Attribute Sets share common ancestry.
WORK ATTRIBUTE SET See Adoption Rule for: Work Attribute	Source model contains Work Attribute Set with same name.
z/OS Library	Source model contains z/OS Library with same name. Associated Business Systems share common ancestry.

Adoption Rules for Non-Selectable Objects

This table contains the adoption rules for objects that are not on the Adoption selection list. These non-selectable objects are components of the selectable objects.

Adoption of non-selectable objects occurs when the parent object is adopted, if the non-selectable object meets the adoption criteria for its object type. Adoption rules for each object type that is not selectable appear on the alphabetically ordered tables that follow.

Component of Selected Object	Adoption Rule
BATCH JOB	Source model contains batch job with associated procedure that share common ancestry with this batch job's associated procedure.
BATCH JOB STEP	Source model contains batch job step with associated procedure step and associated batch job that share common ancestry with this batch job step's associated procedure step and associated batch job.
DATA VIEW GROUP	Source model contains data view group with same type (import, export, entity action, or local). Parent Action Blocks, functions, or processes share common ancestry.

Component of Selected Object	Adoption Rule
DIALOG (default primary) See Adoption Rule for: Non-default dialog Non-default window	Source model contains Default primary dialog or Default primary window with associated procedure step that share common ancestry with this default primary dialog's associated procedure step.
DIALOG (default non-primary) See Adoption Rule for: Non-default dialog Non-default window	Source model contains Default non-primary dialog with same name. Associated procedure steps share common ancestry.
DIALOG (non-default)	Source model contains Non-default dialog or Non-default window with associated procedure step and associated Dialect that share common ancestry with this dialog's associated procedure step and associated Dialect.
ENTITY VIEW	Source model contains Entity view with same name. Associated Entity Types share common ancestry. Parent group views or data view groups share common ancestry.
GROUP VIEW See Adoption Rule for: Group view Entity view	Source model contains Group view with same name. Parent group views or data view groups share common ancestry.
IDENTIFIER	Source model contains an Identifier with attributes and relationships that share common ancestry with attributes and relationships for identifier on the destination model. Parent Entity Types or subtypes share common ancestry.
IMPLEMENTATION UNIT (internal or external)	Source model contains Implementation unit with associated Action Block that shares common ancestry with this implementation unit's associated Action Block.
PACKAGING FOR PROCEDURE STEP	Source model contains Packaging for procedure step with associated load modules of same object type and same type (cooperative or non-cooperative). Associated procedure steps share common ancestry.
PERMITTED VALUE	Source model contains Permitted value with same low and high values. Associated attributes share common ancestry.

Component of Selected Object	Adoption Rule
PROCEDURE STEP ACTION BLOCK See Adoption Rule for: Data view group Group view Entity view Implementation unit	Source model contains procedure step action block with parent procedure step that shares common ancestry with parent procedure step on the destination model.
PROCESS ACTION BLOCK See Adoption Rule for: Implementation unit	Source model contains process action block with parent process that shares common ancestry with this process action block's parent process.
SCREEN See Adoption Rule for: Screen implementation	Source model contains screen with parent Procedure Step that shares common ancestry with this screen's parent Procedure Step.
SCREEN IMPLEMENTATION	Source model contains screen implementation where its associated screen shares common ancestry with this screen implementation's screen.
SUBTYPE See Adoption Rule for: Attributes of subtype Identifiers of subtype	Source model contains subtype with same name. Parent Entity Types share common ancestry.
SYSTEM–WIDE FUNCTION KEY	Source model contains system-wide function key with same number. Parent Business Systems share common ancestry.
TD ACTION BLOCK FOR ENTITY TYPE See Adoption Rule for: Implementation unit	Source model contains Entity Type's TD Action Block with common ancestry for associated Entity Types.
TD ACTION BLOCK FOR CONSTRAINT See Adoption Rule for: Implementation unit	Source model contains Constraint's TD Action Block with common ancestry for associated Constraints.
TECHNICAL SYSTEM	Source model contains technical system with parent Business System that shares common ancestry with this technical system's parent Business System.
WINDOW (default primary) See Adoption Rule: Non-default window Non-default dialog	Source model contains a default primary window or default primary dialog with a parent procedure step that shares common ancestry with this default primary window's parent procedure step.

Component of Selected Object	Adoption Rule
WINDOW (non-default)	Source model contains non-default window or non-default dialog with associated Procedure Step and associated Dialect that share common ancestry with this window's associated Procedure Step and associated Dialect.

Glossary

adoption

Adoption is the process that establishes common ancestry between objects in different models that are logically the same. Whether objects are adopted is determined by adoption rules, where criteria are often identical object name value and same aggregate object type, or common ancestry between parent objects. When a match is found, the Original Encyclopedia ID and Original Object ID of the object in the destination model is changed to have the same values as those of the object that is logically the same in the source model. After adoption, these two objects share common ancestry. When an aggregate object is adopted, its component objects are also adopted if they meet adoption criteria according to the adoption rules. When a parent object is adopted, but a component of that object is not adopted because it is not logically the same as any component in the source model, that component is reported as unadopted.

adoption rules

Adoption rules are the rules that determine whether an object in the destination model will have its Original Encyclopedia ID and Original Object ID replaced by those of the object that is logically the same in the source model. See Adoption Rules.

aggregate object

An aggregate object is an occurrence of an aggregate object type. It is also called aggregate object occurrence. An aggregate object represents a collection of related objects within a model. An aggregate object is the smallest unit of information that you can select to migrate, adopt, or compare. An aggregate object is the smallest unit of information that can be marked changed (see Change Capture). Entity types, Attributes, and Relationship Memberships are examples of aggregate objects.

aggregate object expansion

Aggregate object expansion is the process of determining which related objects are migrated along with a selected aggregate and whether the proper conditions exist in the destination model for the migration to succeed. See the section, Aggregate Object Expansions and Special Cases

aggregate object type

Aggregate object type is a category to which aggregate object occurrences belong. To display a selection list of aggregate objects for Migration, Adoption, or Compare Report, you first select the aggregate object type to which it belongs. Only certain object types are aggregates for Version Control. Aggregate object types for Adoption are different from aggregate object types for Migration and Compare Report.

aggregate set

Aggregate set is a collection of aggregate object occurrences that can be reused as is or modified for functions requiring the same or similar sets, such as such as Trial Migration and Migration.

ancestry

Ancestry of an object refers to the combination of its Original Encyclopedia ID and its Original Object ID. This composite identifier is used internally by Version Control to identify objects across models that are versions of the same object. Migration preserves ancestry when replacing an object; migration propagates ancestry when adding an object. See also common ancestry.

Change Capture

Change Capture is a subsystem that collects and saves information in the model on changes made to aggregate objects or their components. In any session where model objects can be changed, the Change Capture subsystem identifies each change by the CHNGED/CHNGDBY association between an aggregate object and a session object, which contains the date and time of the session and the user ID in effect for the session. An aggregate object is marked changed when it participates in an upload, load module packaging, member name specification, or code and DDL generation. See Change Capture for Version Control.

child model

Child model is a model that has been extracted from one encyclopedia and loaded to another encyclopedia. The model is designated as child in the encyclopedia where it is loaded. A child model can be used as the source model for Migration, Adoption, and Compare Report. Among Version Control functions, it can be used as the destination model only for Compare Report. A child model can be created with the Extract, Model command from the Encyclopedia Model Selection window on the Encyclopedia Client.

common ancestry

Two objects in different models share common ancestry if the objects have the same Original Encyclopedia ID and the same Original Object ID. When objects are migrated from the source model to the destination model, they replace objects with which they share common ancestry. Common ancestry between objects that are logically the same is created by Adoption according to adoption rules by assigning the ancestry of the source model object to the selected destination model object. Common ancestry between objects is created by Migration for objects meeting Special Equivalency Rules. Migration replaces the ancestry as well as the definition of the object being replaced. Objects that share common ancestry are equivalent; they may be the same version or different versions.

companion object

Companion objects maintain the context in which the aggregate object exists in the source model. In migrations, if an aggregate object's companion objects already exist in the destination model, they are not replaced. However, any companion objects missing from the destination model are migrated with the aggregate object. You do not need to specifically request migration of a companion. For example, referenced dialects are companion objects of exit states. If an exit state is migrated, its referenced dialects are copied with it if the dialects do not already exist in the destination model.

Compare Aggregate Object Report

Compare Aggregate Object Report, is the report that is produced when you run the Compare Report function that summarizes how aggregate objects differ between the two models being compared. Sections reported reflect sections you select. If you are comparing two models that you want to keep synchronized and you want to know what new objects were created in the source model that need to be added to the destination model, you would request the section Source only. If you want to know which of the objects that exist in both models have been changed in one model since being changed in the other, you would request the section Different. If you are managing models used for parallel development and you want to know what new components may have been added to an aggregate object in the destination model that are missing from that object in the source, you would request the section Destination only. By examining these report sections, you can determine which objects to select for migration. If you want to identify the objects that were last updated in the same change session, you would request the section Same.

Compare Report

Compare Report is the process that performs difference analysis by comparing all or selected aggregate objects within two selected models in the same C/S Encyclopedia. In addition, Compare Report follows down the chain of subordinate aggregate objects comparing each existing level until no more exist. The purpose of this analysis is to determine which of the models' aggregate objects differ or do not exist. Objects may differ based on changes made to the model. Object changes are collected by the Change Capture subsystem. Differences between equivalent objects are determined by comparing timestamps of their respective session objects or their last change date and time information. An object that is said not to exist is one for which there is no equivalent object in the other model.

component object in adoption

Component objects are objects adopted when an aggregate object is adopted. Component objects in Adoption may be either another aggregate object (attribute), used in the adoption rules of another aggregate (identifier), or referenced across other aggregate objects (view).

component object in migration

Component objects are part of the basic definition of the aggregate object. Component objects always migrate with the aggregate object so that the aggregate object will have the same definition and context in the destination model as in the source model. Conversely, an aggregate replaced by migration will have all component objects replaced or removed. In some instances, a component object of an aggregate object is also an aggregate object in its own right For example, an attribute is a component object of an entity type and also an aggregate object in its own right.

destination model

Destination model is the model that you copy objects to in migration. It is the model in which you update objects' ancestry in adoption. When you select objects to adopt, the selectable objects reside in the destination model. See also source model.

enabling object

Enabling objects are objects that either must be in the destination model when the aggregate object is migrated or must be included in the same migration as the related aggregate object for the migration to succeed. An enabling object, or enabler, may have a parental relationship to the aggregate object, as a parent subject area has to an entity type. The parent subject area must be in the destination model when you migrate an entity type or you must migrate the subject area when you migrate the entity type. An enabling object may be a referenced object, as an entity type is referenced by a view of an action block. The entity type must be in the destination model when you migrate the action block or you must migrate the entity type when you migrate the action block. If an enabler of an aggregate object is not an aggregate object, you must determine what aggregate object the enabler is a component of and migrate that aggregate object along with the other selected objects.

Encyclopedia ID

Encyclopedia ID is the numeric identifier assigned to each C/S Encyclopedia during installation. An encyclopedia's Encyclopedia ID is unique across encyclopedias. The Encyclopedia ID is automatically assigned to each object created within the encyclopedia as one part of the object's unique identification. The combination of an object's Encyclopedia ID and Object ID uniquely identifies the object across all encyclopedias. See also Original Encyclopedia ID.

equivalence

Equivalence between two objects in different models exists when the two objects share common ancestry or the objects meet one of the Special Equivalency Rules. Migration replaces equivalent objects. Compare Report lists equivalent objects as either the same or different, depending on the comparison of their respective session objects that indicate the date and time of the last change session.

equivalent objects

Equivalent objects are objects that either share common ancestry or meet one of the Special Equivalency Rules. Equivalent objects are said to be versions of the same object. Equivalent objects are compared as being the same or different, based on whether their respective session objects are the same or different. See also equivalence.

Host Encyclopedia Version Control

Host Encyclopedia Version Control is the implementation of Version Control available on the Host Encyclopedia. For a summary of differences between C/S Encyclopedia Version Control and Host Encyclopedia Version Control, see Differences from Host Encyclopedia Functions.

language code

Language code is the encyclopedia code page assigned to a model.

logically the same

Logically the same describes objects in different models that are meant to be equivalent, but do not share common ancestry. Objects that are logically the same can be objects that are alike but did not originate from the same source or objects that had their common ancestry severed. Objects that are logically the same can be adopted if they meet adoption criteria specified by the adoption rules for their object type. After adoption, objects that were logically the same become equivalent objects.

migration

Migration is the process that creates or replaces an aggregate object and its components with its definition from a source model within the same encyclopedia. If the destination model contains the aggregate object equivalent to the object being migrated plus any required enabling objects, migration replaces that object and all of its components and adds any missing companion objects. If a component that exists in the destination model does not exist in the source, migration deletes that component. If the destination model does not contain the aggregate object equivalent to the object being migrated, migration adds the aggregate object to the destination model. See also special cases.

Object ID

Object ID is the numeric identifier assigned to each object when it is created. By itself, the Object ID uniquely identifies the object within one encyclopedia. The combination of an object's Encyclopedia ID and Object ID uniquely identifies the object across all encyclopedias. See also Original Object ID.

Original Encyclopedia ID

Original Encyclopedia ID is an object property that is part of an object's ancestry. When a new object is initially uploaded from the CA Gen Toolset, the object's Encyclopedia ID becomes its Original Encyclopedia ID. When a new object is created on an encyclopedia through migration or any of the following commands issued from the Encyclopedia Client, the object inherits the Original Encyclopedia ID from the object on which it is based: Copy From This Encyclopedia, Copy From Another Encyclopedia, or Create Model from Subset. This object property along with the Original Object ID determines common ancestry between objects.

Original Object ID

Original Object ID is an object property that is part of an object's ancestry. When a new object is initially uploaded from the toolset, it is assigned an Original Object ID that is equal to its Object ID. When a new object is created on an encyclopedia through migration or any of the following commands issued from the Encyclopedia Client, the object inherits the Original Object ID from the object on which it is based: Copy From This Encyclopedia, Copy From Another Encyclopedia, or Create Model from Subset. This object property, along with the Original Encyclopedia ID, determines common ancestry between objects.

related objects

Related objects are objects related to an aggregate object. For Migration, an aggregate object's related objects include component objects, companion objects, and enabling objects. For Compare Report, an aggregate object's related objects include subordinate aggregate objects. For Adoption, an aggregate object's related objects include component objects in adoption.

session object

Session object is the object created during a process that changes model objects, which indicates the date and time of the session and the user who initiated the session. Session objects are associated to aggregate objects via CHNGDBY when any of the following C/S Encyclopedia processes change model objects: upload, load module packaging, member name specification, code and DDL generation, and conversion. When an object changes, the object or its parent aggregate object is marked as changed, and a new session object is created. The changed aggregate object is associated with the new session object and disassociated from the old session object. See also Change Capture.

source model

Source model is the model from which you select objects to migrate or compare. It is the model whose objects' ancestry is propagated during adoption. It is the model whose objects' definitions are copied during migration. See also destination model.

special cases

Special cases for migration include objects that are migrated only if certain other objects exist in the destination model, but the migration can succeed even if these objects are absent.

Special Equivalency Rules

Special Equivalency Rules is a table of rules for establishing equivalence between two aggregate objects usually of the same aggregate object type that exist in different models and do not share common ancestry. One such rule is: Business System Implementation objects are equivalent if they are associated to Business Systems that share common ancestry. The exception to the rule of being the same aggregate object type is: a Default Dialog is equivalent to a Default Primary Window if each is primary for equivalent Procedure Steps. Objects that are deemed equivalent based on one of these rules are treated as though they share common ancestry, even though they do not. Compare Report and Migration use the Special Equivalency Rules table. When objects that are equivalent based on Special Equivalency Rules are migrated, Migration establishes common ancestry between the objects. See the table, Special Equivalency Rules subordinate aggregate object

Subordinate aggregate objects

Subordinate aggregate objects are, in some cases, component objects. For example, an entity type's attributes are subordinate aggregate objects and also components of that entity type. In other cases, subordinate aggregate objects are not component objects but have a well-known child-parent relationship to the aggregate object in question. For example, Procedures are subordinate aggregate objects to Business Systems even though Procedures are not defined as a component of a Business System. Compare Report processing follows down the chain of subordinate aggregate objects, comparing each existing level. If you select a subject area to be compared within two models, all subordinate aggregate objects of that subject area are compared, including its child subject areas, entity types, attributes, relationship memberships, and so on.

Trial Adopt Model Report

Trial Adopt Model Report is the report produced by Trial Adoption when you select Adopt All Objects as the Object Range.

Trial Adopt Objects Report

Trial Adopt Objects Report is the report produced by Trial Adoption when you select Adopt Selected Objects as the Object Range.

Trial Adoption

Trial Adoption is a process that simulates an actual Adoption and reports what the results of the adoption would be. Unlike Adoption, Trial Adoption does not change the ancestry of objects in the destination model and does not perform or report component unadoption. Trial Adoption produces either the Trial Adopt Model Report or the Trial Adopt Objects Report.

Trial Migrate Aggregate Object Report

Trial Migrate Aggregate Object Report is the report produced by Trial Migration.

Trial Migration

Trial Migration is a process that simulates an actual Migration and reports what the results of an actual migration would be. Unlike Migration, Trial Migration does not change the destination model. Trial Migration produces the Trial Migrate Aggregate Object Report.

unadopted object

Unadopted object is a component of an aggregate object for Adoption, where the aggregate object was adopted but the component was not adopted because it did not conform to the adoption rules for its object type. When an object is unadopted, it is given a new Original Object ID that has never been used and its Original Encyclopedia ID is replaced with the Encyclopedia ID of the current encyclopedia.

unadoption

Unadoption is a phase of Adoption processing where components of adopted objects that cannot be adopted with the parent have their ancestry replaced to sever any common ancestry the objects may have with objects in another model. (If unadoptions were not performed, subsequent migration could place component objects within multiple parents.)

Index

F

G

H

I

L

M

N

O

P

R

S

T

U

V