DataFlux Migration Guide

DATAFLUX
A SAS COMPANY

YOUR DATA.
YOUR BUSINESS.
ONE SOLUTION.

DataFlux Migration Guide

Version 2.1.2

November 23, 2010

This page is intentionally blank

Contact DataFlux

Corporate Headquarters

DataFlux Corporation

940 NW Cary Parkway, Suite 201
Cary, NC 27513-2792
Toll Free Phone: 877-846-FLUX (3589)
Toll Free Fax: 877-769-FLUX (3589)
Local Phone: 1-919-447-3000
Local Fax: 919-447-3100
Web: http://www.dataflux.com

DataFlux United Kingdom

Enterprise House
1-2 Hatfields
London
SE1 9PG
Phone: +44 (0) 20 3176 0025

DataFlux Germany

In der Neckarhelle 162
69118 Heidelberg
Germany
Phone: +49 (0) 6221 4150

DataFlux France

Immeuble Danica B
21, avenue Georges Pompidou
Lyon Cedex 03
69486 Lyon
France
Phone: +33 (0) 4 72 91 31 42

Technical Support

Phone: 1-919-531-9000
Email: techsupport@dataflux.com
Web: http://www.dataflux.com/MyDataFlux-Portal

Documentation Support

Email: docs@dataflux.com

Legal Information

DataFlux Legal Statements

DataFlux Solutions and Accelerators Legal Statements

DataFlux Legal Statements

Apache Portable Runtime License Disclosure

Apache/Xerces Copyright Disclosure

DataDirect Copyright Disclosure

Google Web Toolkit

JDOM Project

OpenSymphony

Sun Microsystems

Java Toolkit

Table of Contents

Introduction

Conventions Used In This Document

This document uses several conventions for special terms and actions.

Typographical Conventions

The following typographical conventions are used in this document:

Bold	Text in bold signifies a button or action
italic	Identifies document and topic titles
monospace	Typeface used to indicate examples of code

Syntax Conventions

The following syntax conventions are used in this document:

[]	Brackets [] are used to indicate variable text, such as version numbers
#	The pound # sign at the beginning of example code indicates a comment that is not part of the code

DataFlux Reference Publications

This document may reference other DataFlux® publications. Any referenced publications are available from the MyDataFlux customer portal on the DataFlux.com Web site, including the following:

DataFlux Authentication Server Administrator's Guide, v. 2.1.1

DataFlux Authentication Server User's Guide, v. 2.1.1

DataFlux Data Management Server Administrator's Guide, v. 2.1.1

DataFlux Data Management Server User's Guide, v. 2.1.1

DataFlux Data Management Studio Installation and Configuration Guide, v. 2.1.1

DataFlux Data Management Studio User's Guide, v. 2.1.1

DataFlux Expression Language Reference Guide, v. 2.1.1

DataFlux Federation Server Administrator's Guide, v. 2.1.1

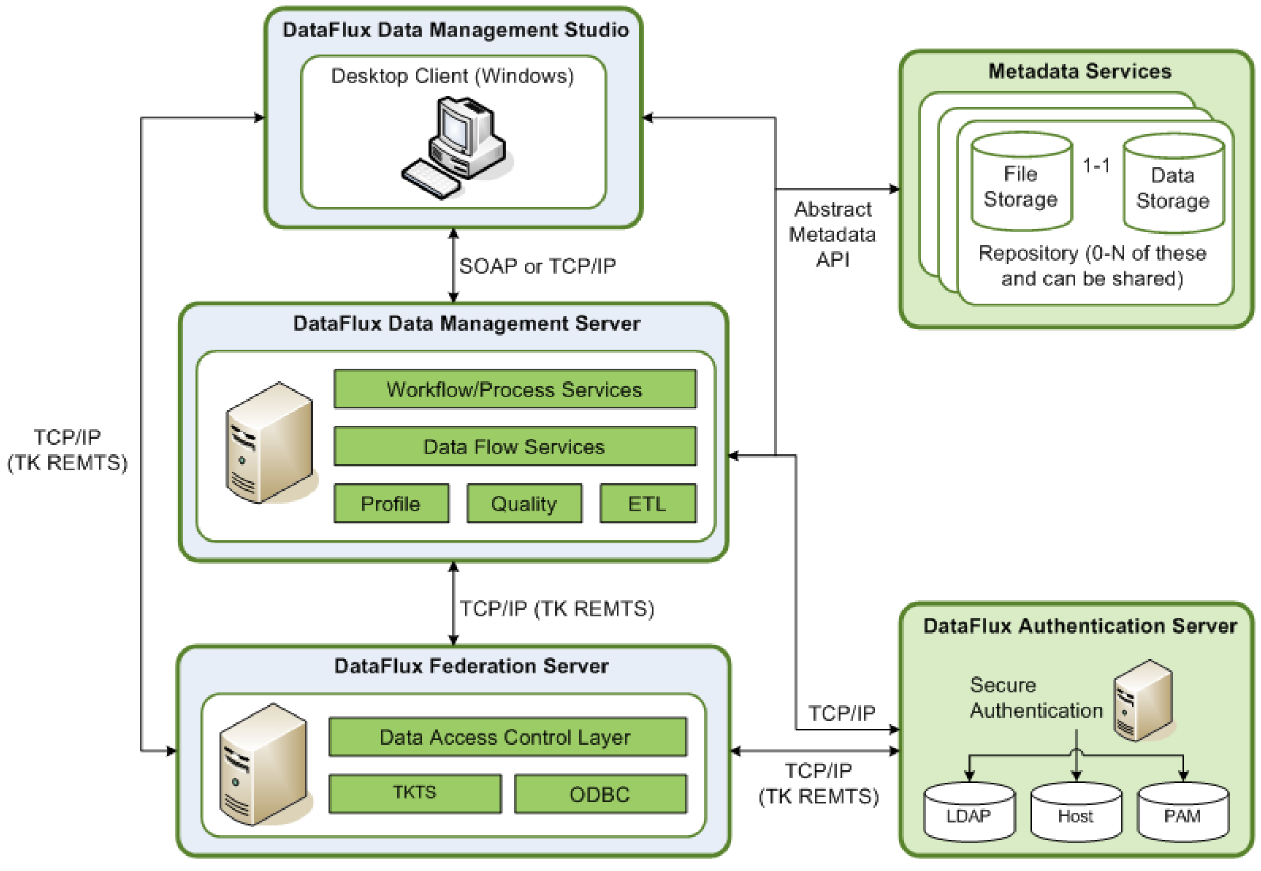
DataFlux Federation Server User's Guide, v. 2.1.1

DataFlux Quality Knowledge Base Online Help

Overview of Migration

Overview of the Data Management Platform

This guide describes how to migrate your content from DataFlux dfPower Studio to the new DataFlux Data Management Platform. The following diagram illustrates the components of the platform.



DataFlux Data Management Platform Architecture

Data Management Studio. DataFlux Data Management Studio is a data management suite that combines data quality, data integration, and Master Data Management (MDM). Compared to dfPower Studio, Data Management Studio has a more integrated interface, better management of data flows and process flows, enhanced data access through the Federation Server, and many other new features. The Migrating from dfPower Studio chapter in this guide describes how to migrate your content and manage the impact of differences between Data Management Studio and dfPower Studio.

Repositories. When you create jobs and other objects in Data Management Studio, they are stored in repositories that are similar to dfPower Studio repositories. Profiles, rules, tasks and some other objects in a repository are stored in database format. You can specify a separate storage location for objects that are stored as files, such as data jobs, process jobs, and queries. You can create a private repository for your own use, or you can create a shared repository that a number of people can use. The Migrating Repositories topic in this guide describes how to migrate a dfPower Studio repository to a Data Management Studio repository.

Servers. Data Management Studio can be used by itself or in combination with one or more of the following DataFlux servers:

- The DataFlux Data Management Server is a new version of the DataFlux Data Integration Server. It provides a scalable server environment for large Data Management Studio jobs. Jobs can be uploaded from Data Management Studio to a Data Management Server, where the jobs are executed.
- The DataFlux Federation Server is a new server. It provides central management of the data connections using ODBC or native drivers as well as access privileges for these connections.
- The DataFlux Authentication Server is a new server. It centralizes the management of users, groups, and database credentials.

The Federation Server and the Authentication Server will typically have little impact on migration from dfPower Studio because they are new to the Data Management Platform. They could have an impact after migration, if you decide to use the features that these servers provide.

Overview of Migrating from dfPower Studio

What is Migration?

Migration is the process in which dfPower Studio content is upgraded to run in Data Management Studio, or Data Integration Server content is upgraded to run on a Data Management Server.

What Versions Can Be Migrated?

Migrate From	Migrate To
DataFlux dfPower Studio 8.1 and DataFlux Data Integration Server 8.1	DataFlux Data Management Studio 2.1 and DataFlux Data Management Server 2.1

Migrations from version 8.1 to version 2.1 are supported regardless of fix patch level.

Summary of the Migration Process

A typical approach is as follows:

- migrate a dfPower Studio repository
- migrate Architect jobs
- migrate Profile jobs
- make any required post-migration adjustments

For details, see the Migrating from dfPower Studio chapter in this guide.

Content That Can Be Automatically Migrated

The following content can be automatically migrated from dfPower Studio to Data Management Studio. Some migrated items will require manual updates before you can use them in Data Management Studio.

- **Architect jobs**. You will use the upgrade job utilities in Data Management Studio to convert dfPower Studio Architect jobs to process jobs or data jobs. For more information, see Architect Jobs.
- **Business rules, custom metrics, and other items in the Business Rules Manager**. These items are migrated only when their repository is migrated. You cannot migrate individual rules, custom metrics, and other items in the Business Rules Manager. For more information, see Business Rules, Tasks, and Related Items.
- **Profile jobs**. You can use the **Import Profile Jobs and Link Reports** wizard in Data Management Studio to convert dfPower Studio profile jobs. For more information, see Profile Jobs.
- **Profile reports that are stored in a repository**. You can use the **Import Profile Jobs and Link Reports** wizard in Data Management Studio to link a dfPower Studio profile job with a related report and combine the two in a Data Management Studio profile. For more information, see Profile Jobs.
- **Queries**. Saved queries that were created with the Query Builder are migrated only when their repository is migrated. You cannot migrate individual queries.
- **Repositories**. The Add Repository dialog in Data Management Studio can do an in-place migration of a dfPower Studio repository. For more information, see Repositories and Management Resources .

Content That Must Be Modified or Recreated

The following content cannot be automatically migrated from dfPower Studio to Data Management Studio. You must modify it or recreate it.

- **Architect jobs that include certain nodes**. Architect jobs that include certain nodes require special handling. See Post-Migration Tasks for Architect Jobs.
- **Architect jobs that pass macro variable values dynamically between pages in the job**. For more information about the updates that are required for these jobs, see Architect Jobs That Pass Macro Variable Values Dynamically Between Pages in the Job.
- **Batch scripts**. The Data Management Platform has a new command line interface, so you must modify any batch scripts that execute **profexec.exe** (Profile command line interface for dfPower Studio); **archbatch.exe** (Architect command line interface for dfPower Studio), or **dfexec.exe** (command line interface for DataFlux Integration Server). For more information, see "Running Jobs from the Command Line" in the "Data Jobs" chapter of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.
- **Configuration files and global options**. You cannot convert configuration files or global options that were created in dfPower Studio. To review the global options for Data Management Studio, select **Tools** > **Data Management Studio Options** from the main menu. Some options can be set with checkboxes and similar controls. Other options must be set in configuration files. To review or change the settings in Data Management Studio configuration files, see the "Using Configuration Files" topic in the "Global Options" chapter of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.
- **Data Explorations**. You cannot convert Explorations that were created in dfPower Studio. You must recreate them in Data Management Studio. See the "Data Explorations" chapter in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*. Currently, the migration software will move dfPower Data Explorations into the **Folders** tree in Data Management Studio. You cannot use these items in Data Management Studio, however, so you can select and delete them from the **Folders** tree .
- **Data Management Server**. The DataFlux Data Management Server is a new version of the DataFlux Data Integration Server. The move to the new server will have some impacts on migration. For more information, see Changes Related to the Data Management Server.
- **Macro variables**. You will probably have to make some updates in order for dfPower Studio macro variables to work in Data Management Studio. For more information, see Macro Variables.
- **Profile reports that are stored in a file (.pfo file)**. You cannot convert dfPower Studio profile reports that are stored as files (.pfo files). For more information, see the Other Changes section in the "Profile Jobs" topic.
- **Redundant Data Analysis reports**. The way that Redundant Data Analysis is calculated has changed. Accordingly, after a Redundant Data Analysis profile job is converted, you must re-run the profile in Data Management Studio to recreate the report. For more information, see the Other Changes section in the "Profile Jobs" topic.

- **Repositories and Management Resources (some configurations)**. Differences in metadata architecture and other changes might require you to make manual adjustments before or after you migrate dfPower Studio repositories and Management Resources to Data Management Studio. For more information, see Changes to Repositories and Management Resources.

- **SAS connections**. You cannot convert SAS connections that were created in dfPower Studio. You must recreate them in Data Management Studio. See the "Adding SAS Data Set Connections" topic in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

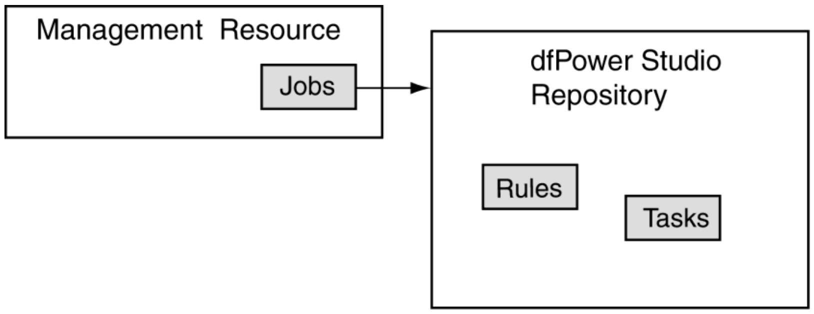
Migrating from dfPower Studio

Repositories and Management Resources

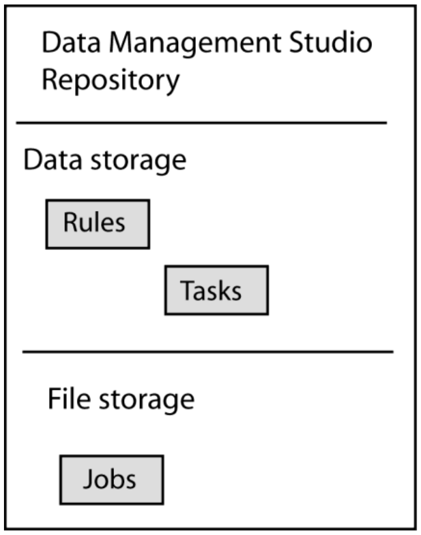
Changes to Repositories and Management Resources

The Meaning of "Repository" Has Changed

In dfPower Studio, you can have two independent storage areas for metadata: a database storage area called a repository, and a file-based storage area called a Management Resource. Some of the main items that are associated with each area are shown in the next figure.



In Data Management Platform, you can have a database storage area and a file storage area, but these two areas are not independent. The database storage area (called “data storage”) and the file storage area are bound together under a single ID, in a single entity called a repository. Some of the main items that are associated with each area are shown in the next figure.



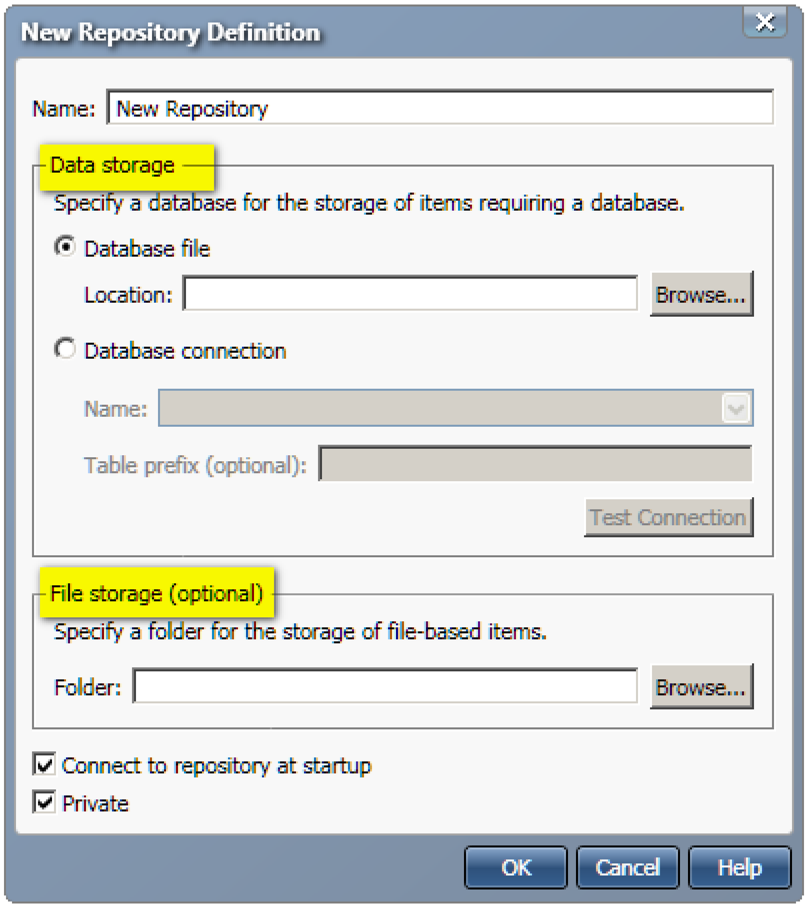
Storing metadata in a single entity enables Data Management Studio to present metadata in a single user interface. For example, business rules and tasks are displayed in the same interface that displays jobs and reports. However, the new architecture puts new constraints on where items can be stored. In Data Management Platform:

- Each Data Management Platform repository must have a data storage area. The file storage area is optional.
- Each Data Management Platform repository can have at most one data storage area and one file storage area.
- All related items must be stored in the same Data Management Platform repository. This means, for example, that a job or profile that is stored in a repository can reference only those business rules, tasks, or similar items that are stored in the same repository.

Differences in metadata architecture might require you to make some manual adjustments when you migrate dfPower Studio repositories and Management Resources to Data Management Studio. For more information, see Plan the Migration of Each Repository and Management Resource.

New Repository Interfaces

In Data Management Studio, the **Add New Repository Definition** dialog is used to add repositories.


New Repository Definition

Name: New Repository

Data storage

Specify a database for the storage of items requiring a database.

(●) Database file

Location:

( ) Database connection

Name:

Table prefix (optional):

File storage (optional)

Specify a folder for the storage of file-based items.

Folder:

[✓] Connect to repository at startup

[✓] Private

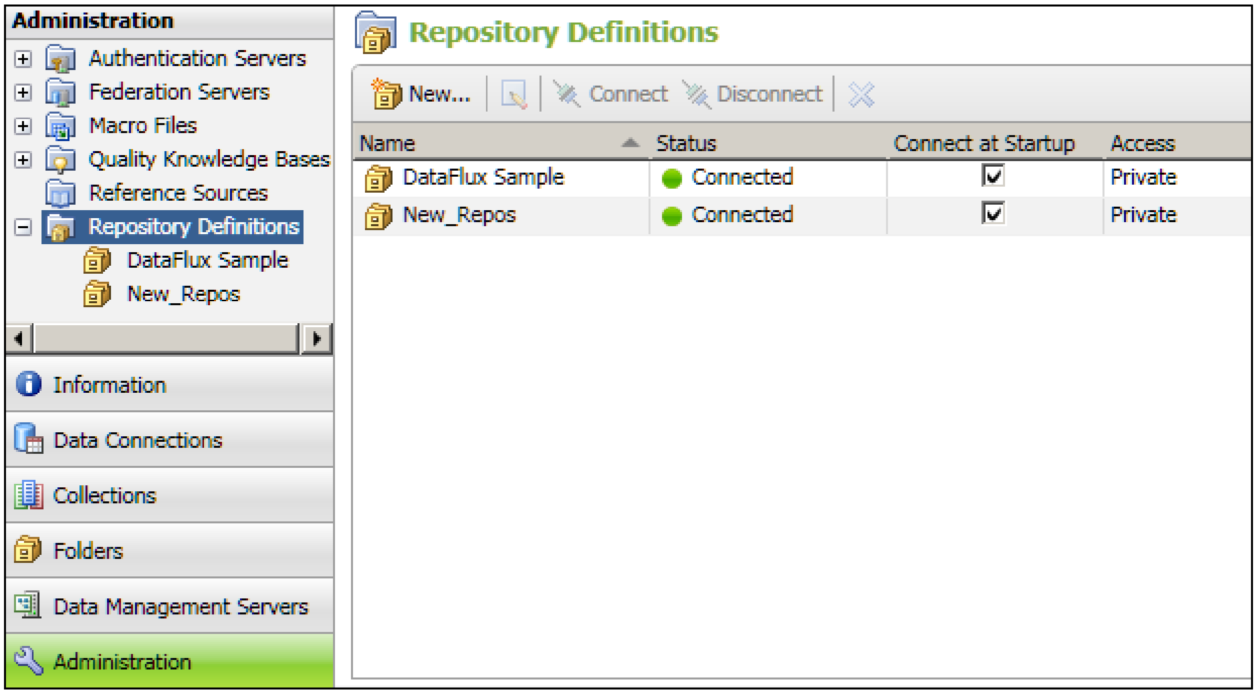
The **Data storage** area of the dialog is similar to the location of a dfPower Studio repository. The **File storage** area of the dialog is similar to the location of a dfPower Studio Management Resource. In Data Management Platform, however, the **Data storage** area and the **File storage** area are bound together under a single ID, in a single entity called a repository.

In Data Management Platform, it is possible to have a repository that specifies:

- a **Data storage** area only
- a **Data storage** area and a **File storage** area

It is not possible for a Data Management Platform repository to specify a **File storage** area only.

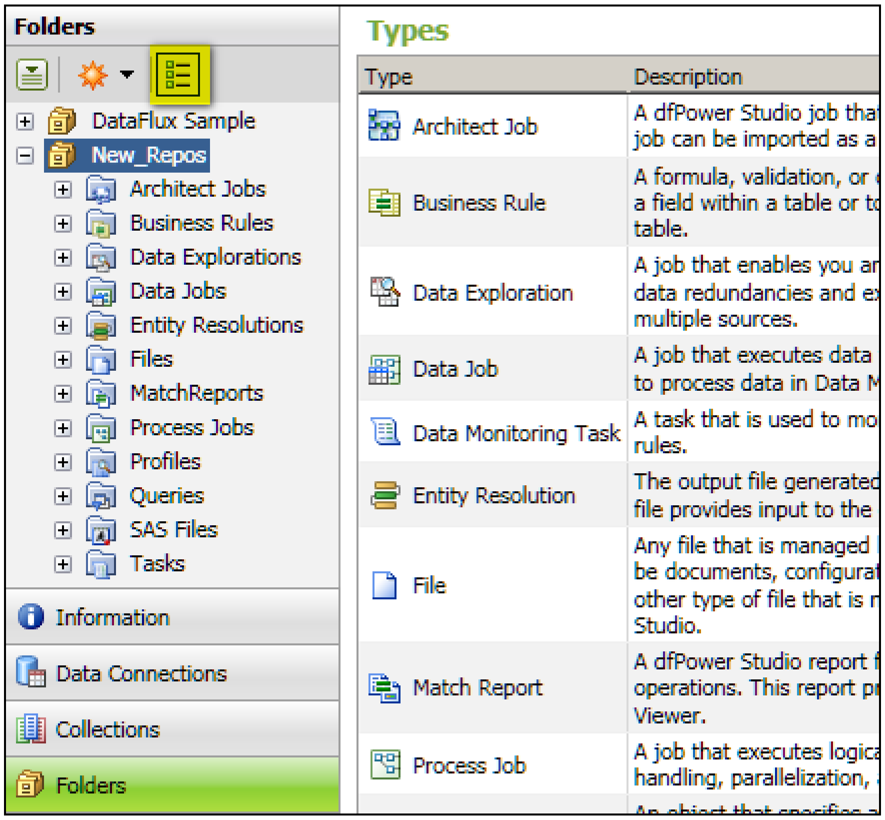
The **Repository Definitions** folder in the **Administration** riser is used to manage the list of available repositories.



Name	Status	Connect at Startup	Access
DataFlux Sample	● Connected	✓	Private
New_Repos	● Connected	✓	Private

In the previous display, there are two repository definitions: **DataFlux Sample** and **New_Repos**.

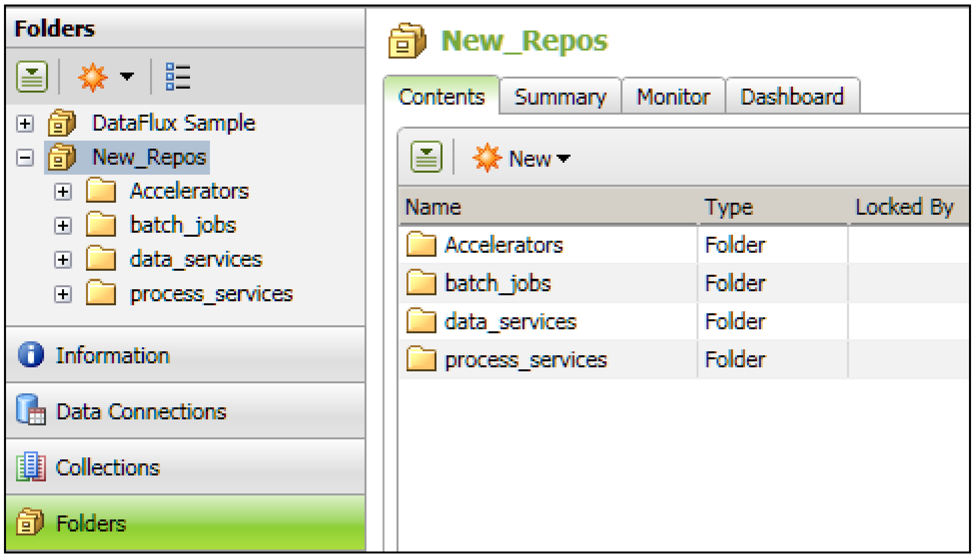
The **Folders** riser is where you manage the items in a repository. In the **Folders** tree shown in the next display, the **Group Items by Type** control at the top of the tree is selected.



Type	Description
Architect Job	A dfPower Studio job tha job can be imported as a
Business Rule	A formula, validation, or a field within a table or t table.
Data Exploration	A job that enables you ar data redundancies and e multiple sources.
Data Job	A job that executes data to process data in Data M
Data Monitoring Task	A task that is used to mo rules.
Entity Resolution	The output file generated file provides input to the
File	Any file that is managed be documents, configurat other type of file that is r Studio.
Match Report	A dfPower Studio report f operations. This report p Viewer.
Process Job	A job that executes logica handling, parallelization,

The **Group Items by Type** control groups items into system-defined folders that are named after general types, such as **Architect Jobs**, **Business Rules**, and so on. A description of all of the standard object types is displayed on the right.

If you deselect the **Group Items by Type** control, items are grouped into user-defined folders, as shown in the next display.



Name	Type	Locked By
Accelerators	Folder	
batch_jobs	Folder	
data_services	Folder	
process_services	Folder	

In the previous display, the standard folders for deployable jobs are shown: **batch_jobs**, **data_services**, and **process_services**. For more information about these folders, see Changes Related to the Data Management Server. For more information about the **Administration** riser and the **Folders** riser, see the "Overview of the Interface" chapter and the "Repositories" chapter of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Changes in Repository Storage

There are some changes in the databases and operating systems that can be used for Data Management Studio repository storage. For more information, see "Repository Storage" in the *DataFlux Data Management Studio Installation and Configuration Guide, v. 2.1.1*.

Changes Related to the Data Management Server

The DataFlux Data Management Server is a new version of the DataFlux Data Integration Server. Jobs can be uploaded from Data Management Studio to a Data Management Server, where the jobs are executed.

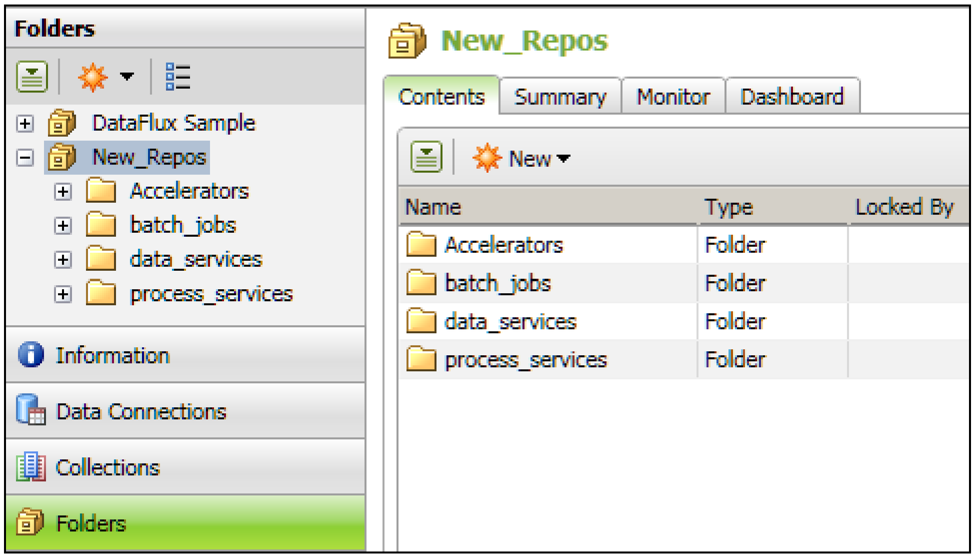
The DataFlux Data Management Server can execute both Data Management Studio jobs and dfPower Studio jobs. You can upload jobs from dfPower Studio to the Data Management Server for execution.

If you used a DataFlux Integration Server with security enabled, and you plan to continue to do so when moving to Data Management Studio, then you must regenerate and reconfigure all security ACL files. Enhancements have been made to Data Management Server that preclude the use of existing ACL files. For more information, see the *DataFlux Data Management Server Administrator's Guide, v. 2.1.1*.

Multiple repositories can be active on a Data Integration Server, but only one repository can be active on a Data Management Server. If your site requires multiple repositories, then you must install multiple Data Management Servers.

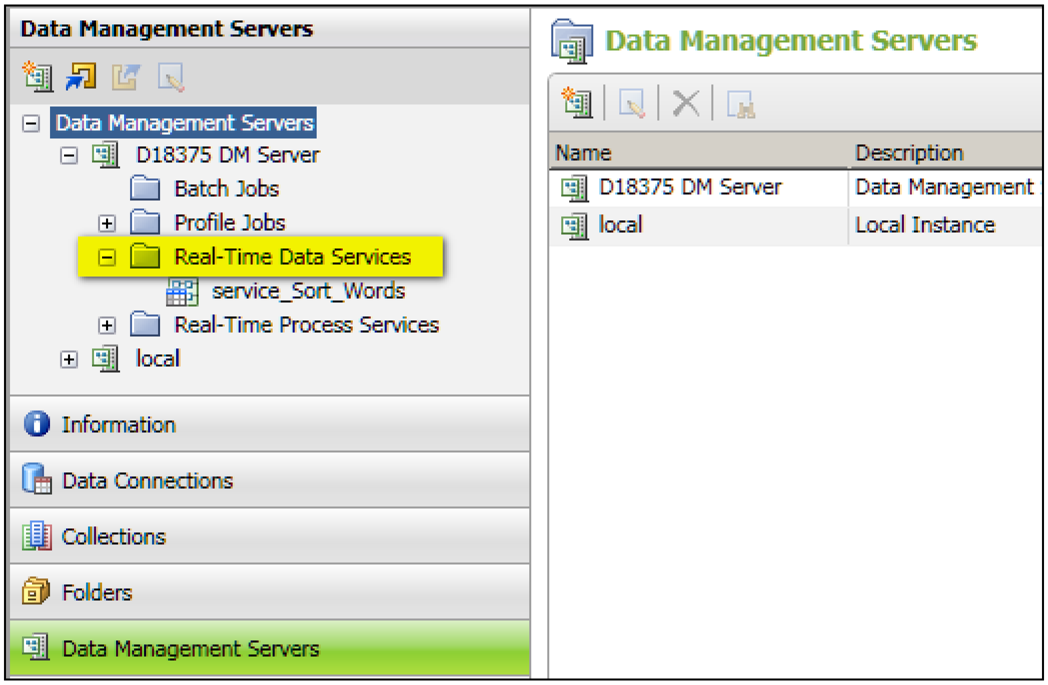
The standard folders for deployable jobs (**batch_jobs**, **data_services**, and **process_services**) should always exist in a Data Management Studio repository and in a Data Management Server repository. Mirroring these folders will help preserve any references where one Data Management Studio job references another.

A set of standard folders for deployable jobs are usually added automatically when you add a new repository in Data Management Studio and you specify a **File storage** location. To display the standard folders in the **Folders** tree, put the tree in user-defined folders mode by de-selecting the **Group Items by Type** control at the top of the tree. Then select and expand the new repository, as shown in the next display.



Name	Type	Locked By
Accelerators	Folder	
batch_jobs	Folder	
data_services	Folder	
process_services	Folder	

In the previous display, the standard folders for deployable jobs are shown: **batch_jobs**, **data_services**, and **process_services**. If you were to create a data service job in Data Management Studio, you would save it to the **data_services** folder. When you deploy the job on a Data Management Server, you would deploy it to the **data_services** folder on the server.



Name	Description
D18375 DM Server	Data Management
local	Local Instance

In the previous display, a data service job called **service_Sort_Words** has been deployed in a folder named **Real-Time Data Services**. The default name of this folder on the server does not happen to be **data_services**, but the physical path for this folder is the path to the **data_services** folder on the server's file system ([*SERVER_HOME*]\var\data_services).

Note: The standard folders for deployable jobs are not added automatically when you migrate a dfPower repository. Accordingly, you must manually create the standard job folders in the migrated repository, if you will ever deploy Data Management Studio jobs to a Data Management Server.

One way to add the standard folders for deployable jobs is to use operating system tools, as described in General Planning for Repositories. Another way is to add the folders in the user-defined folders view of the **Folders** tree. Perform the following steps.

1. Click on the **Folders** riser bar in Data Management Studio.
2. If the **Folders** tree is not in user-defined folders mode, deselect the **Group Items by Type** control at the top of the tree.
3. Select and expand the migrated repository in the **Folders** tree.
4. Right-click the repository icon and select **New** > **Folder**. A new folder is added to the tree.
5. Name the new folder after one of the standard folders for deployable jobs, such as **batch_jobs**. Be sure that the name is an exact match for the standard folder.
6. Use the same method to add the other standard folders.

Plan the Migration of Each Repository and Management Resource

General Planning for Repositories

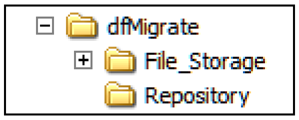
Review Overview of Migrating from dfPower Studio. Try to anticipate the main items that you will have to update, replace, move or delete in Data Management Studio.

Note: Data Management Studio does an in-place migration of a dfPower repository. dfPower Studio cannot access a repository that has been converted to Data Management Studio format.

Accordingly, make a copy of the dfPower repository to be migrated, then migrate the copy rather than the original. Migrating a copy of your repository enables you to preserve your original dfPower Studio content.

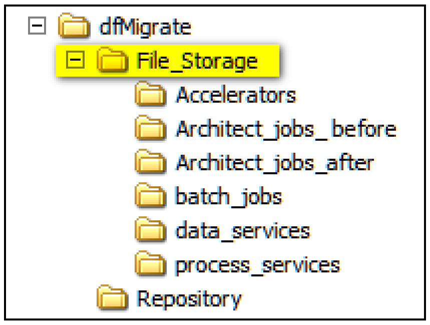
If the dfPower repository is stored on a database server, use DBMS tools to copy the repository. Then, create a new ODBC connection that points to the copy. Later, you will use that new connection to specify the dfPower repository to be migrated.

If the dfPower repository is stored as a database file (.RPS file) on the file system, create a new, empty folder structure, such as the one shown in the next display.



In the previous display, **dfMigrate** is a top-level folder. The **Repository** sub-folder is where the dfPower repository file will be copied and later converted to a Data Management Studio repository. The names of these folders are simply examples.

The **File_Storage** sub-folder corresponds to the file storage area in a Data Management Studio repository. Under the folder for the file storage area, you should at least create the standard folders for deployable jobs, as described in Changes Related to the Data Management Server. These standard folder names are not examples. They must literally be **batch_jobs**, **data_services**, and **process_services**, as shown in the next display.



Note: The standard folders for deployable jobs are not added automatically when you migrate a dfPower repository. Accordingly, you must manually create the standard job folders in the migrated repository, if you will ever deploy Data Management Studio jobs to a Data Management Server.

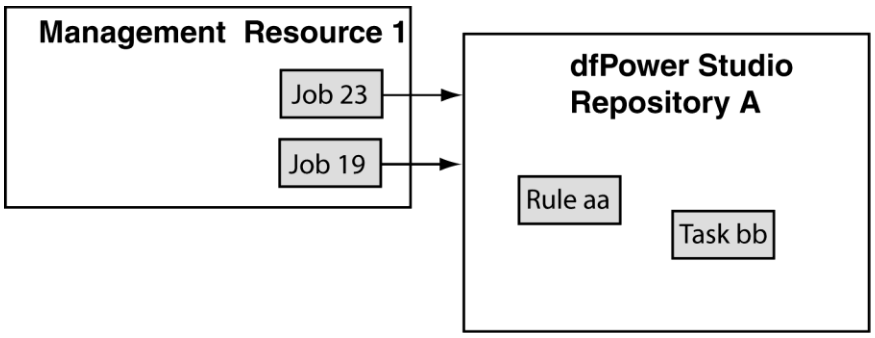
In this migration guide, the folders in the previous display are used as follows:

- Unconverted Architect jobs are copied to the **Architect_jobs_before** folder, as described in Copy Architect Jobs to the File Storage Area of the Repository.
- Architect jobs that have been converted to Data Management Studio format are saved to a number of different folders. The **Architect_jobs_after** folder is one example.
- The standard folders for deployable jobs (**batch_jobs**, **data_services**, and **process_services**) are mainly intended for Data Management Studio jobs that will be deployed to a Data Management Server.

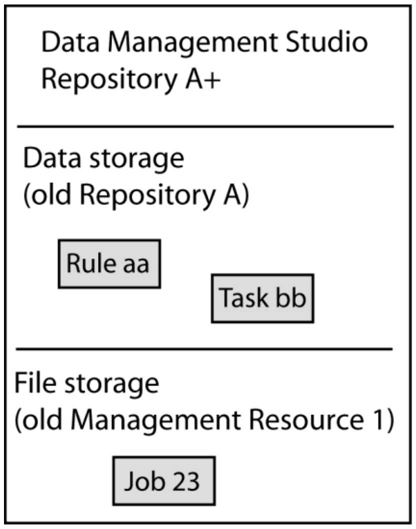
The **Accelerators** folder is used for jobs that are associated with DataFlux Accelerators. This guide does not cover the migration of accelerators.

Migration Scenario: One dfPower Repository, One Management Resource

The next figure illustrates the easiest metadata configuration to migrate one dfPower repository and one Management Resource. Assume that Management Resource 1 contains Architect jobs and profiles that refer to rules and tasks in Repository A only.



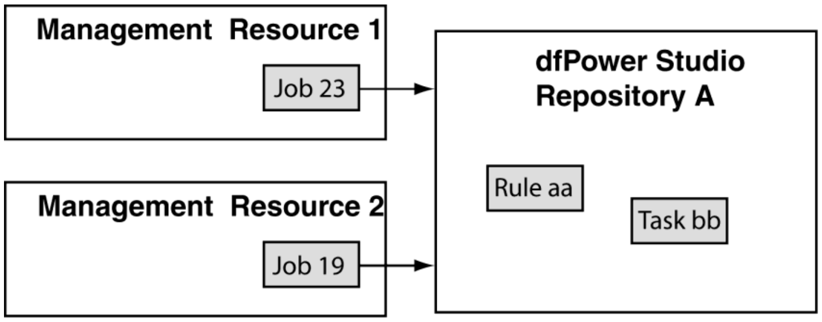
To migrate this configuration, you would run the New Repository Definition wizard in Data Management Studio. You would specify a data location, which would be the physical path to a copy of Repository A. You would specify an empty file storage area, such as the **File_Storage** folder that is described in General Planning for Repositories. Then you would copy the Architect jobs from Management Resource 1 to that file storage area. The end result would be a repository where the rules and tasks that are required by the jobs are in the same repository, as shown in the next figure.


Data Management Studio Repository A+
Data storage (old Repository A)
Rule aa
Task bb
File storage (old Management Resource 1)
Job 23

For detailed steps, see Migrate Repositories and Management Resources.

Migration Scenario: One dfPower Repository, Multiple Management Resources

In dfPower Studio, it is possible for multiple Management Resources to contain Architect jobs and profiles that refer to rules and tasks in a single repository, as shown in the next figure. Assume that both Management Resource 1 and Management Resource 2 contain Architect jobs and profiles that refer to rules and tasks in Repository A only.

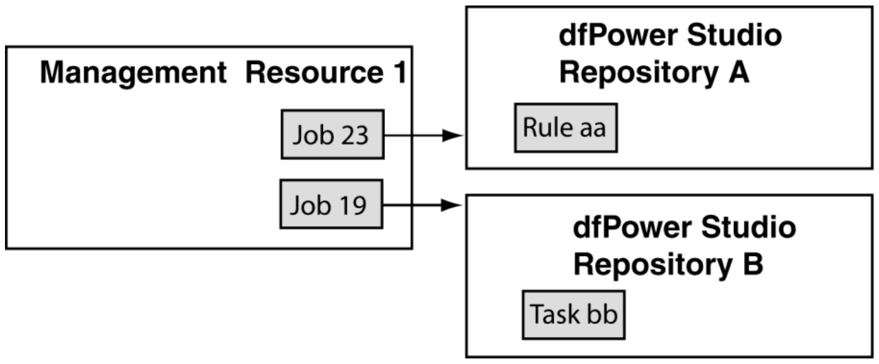


In Data Management Studio, however, each repository can have at most one data storage area and one file storage area. You cannot create a Data Management Studio repository that has two file storage areas that correspond to Management Resources 1 and 2. Given a dfPower Studio configuration similar to the one shown above, you must perform additional tasks so that all related items can be stored in the same Data Management Studio repository.

To migrate this configuration, you would run the New Repository Definition wizard in Data Management Studio. You would specify a data location, which would be the physical path to a copy of Repository A. You would specify an empty file storage area, such as the **File_Storage** folder that is described in General Planning for Repositories. Then you would copy the Architect jobs from Management Resource 1 and Management Resource 2 to that file storage area. The end result would be a repository where the rules and tasks that are required by the jobs are in the same repository. For detailed steps, see Migrate Repositories and Management Resources.

Migration Scenario: Multiple dfPower Repositories, One Management Resource

In dfPower Studio, it is possible for Architect jobs and profiles in one Management Resource to refer to rules and tasks in a multiple repositories, as shown in the next figure.

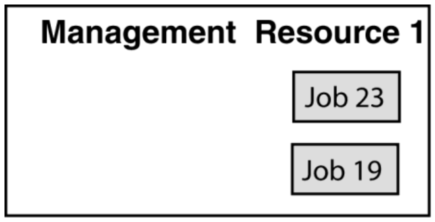


In Data Management Studio, however, each repository can have at most one data storage area and one file storage area. You cannot create a Data Management Studio repository that has two data storage areas that correspond to Repositories A and B. Given a dfPower Studio configuration similar to the one shown above, you must perform additional tasks so that all related items can be stored in the same Data Management Studio repository. For example, you could perform the following steps.

1. Migrate one dfPower repository as described in Migrate Repositories and Management Resources. In the current example, you could migrate Repository A. You would specify an empty file storage area for this repository, as described in General Planning for Repositories.
2. Copy the relevant Architect jobs from Management Resource 1 to the file storage area for Repository A, as described in Copy Architect Jobs to the File Storage Area of the Repository.
3. Migrate the other repository or repositories that have rules or tasks that are consumed by the jobs in the repository that you migrated in Step 1. In the current example, you would migrate Repository B.
4. In Data Management Studio, open the repository that you migrated in Step 3, such as Repository B.
5. Use the Business Rule Manager to export the relevant rules or tasks from the repository. For the current example, you would export Task bb from Repository B. For more information about this step, see Export Rules and Related Items.
6. Open the repository that you migrated in Step 1, such as Repository A.
7. Use the Business Rule Manager to import the rules or tasks that you exported in Step 5, such as Task bb. For more information about this step, see Import Rules and Related Items.

Migration Scenario: No dfPower Repository, One or More Management Resources

In dfPower Studio, it is possible for one or more Management Resources to contain Architect jobs and profiles that do not refer to rules or tasks in any dfPower repository, as shown in the next figure.



In Data Management Studio, however, each repository must have one data storage area. You cannot create a Data Management Studio repository that has a file storage area only.

To migrate this configuration, you would create a new Data Management Studio repository with a new (empty) data storage area, such as the **Repository** folder that is described in General Planning for Repositories. You would specify an empty file storage area as well, such as the **File_Storage** folder that is described in General Planning for Repositories. Then you would copy the relevant Architect jobs to that file storage area. The end result would be a repository where the jobs from one or more Management Resources were stored in the same repository. For detailed steps, see Migrate a Management Resource With No dfPower Repository.

Migrate a dfPower Repositories and Management Resources

Migrate a File-Based Repository That Will Not Be Managed By a Data Management Server

This topic is appropriate under the following conditions:

- the dfPower repository is in SQLite format (an .RPS file)
- jobs in a Management Resource refer to rules and tasks in the dfPower repository
- the dfPower repository is not managed by a DataFlux Data Integration Server
- the migrated repository will not be managed by a DataFlux Data Management Server

In general, you will create an empty folder structure such as the structure described in General Planning for Repositories. Creating such a structure ensures that you have the standard folders for deployable jobs. You will copy the dfPower repository to an appropriate folder in that structure. When you run the New Repository Definition wizard, you will specify the location of the copied dfPower repository and the location of an empty set of folders. Later, you will copy Architect jobs into these folders, as described in Copy Architect Jobs to the File Storage Area of the Repository.

Perform the following steps.

1. Create an empty folder structure to hold the new Data Management Studio repository and related assets, as described in General Planning for Repositories.

2. Use operating system tools to copy the dfPower repository to an appropriate folder from Step 1. For example, you could copy the dfPower repository to a folder such as C:\dfMigrate\Repository.

3. Run Data Management Studio.

4. Click on the **Administration** riser bar. Then select the **Repository Definitions** folder in the **Administration** tree. The Repository Definitions pane displays on the right.

5. In the Repository Definitions pane, click on the **New** button to create the new repository. The New Repository Definition dialog appears.

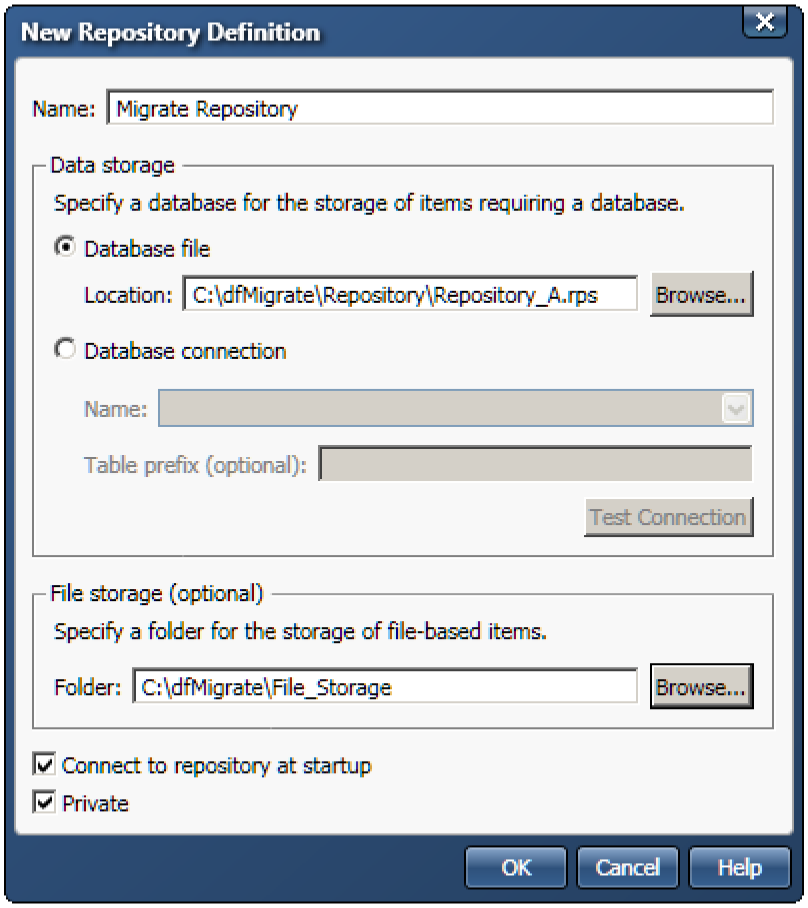
6. Enter the name of your repository in the **Name** field.

7. In the **Data storage** section of the dialog, specify the **Database file** for the dfPower repository that you want to migrate. For example, you could specify a path such as C:\dfMigrate\Repository\Repository_A.rps.

   Note: Do not create a repository database file (.RPS) in the file storage location that you will specify in Step 8. This practice prevents manipulating the file through Data Management Studio, and it triggers unneeded update events in the file storage area every time that the database file is updated.

8. In the **File storage** section of the dialog, specify an appropriate, empty folder from Step 1. For example, you could specify a path such as C:\dfMigrate\File_Storage. (Later, you will copy unconverted dfPower Architect jobs to this area and save converted copies of these jobs to the same area.)

9. Select or deselect the **Connect to repository at startup** checkbox or the **Private** checkbox as appropriate. The New Repository Definition dialog would look similar to the next display.


New Repository Definition

Name: Migrate Repository

Data storage

Specify a database for the storage of items requiring a database.

(●) Database file

Location: C:\dfMigrate\Repository\Repository_A.rps

( ) Database connection

Name:

Table prefix (optional):

File storage (optional)

Specify a folder for the storage of file-based items.

Folder: C:\dfMigrate\File_Storage

[✓] Connect to repository at startup

[✓] Private

10. Click on the **OK** button. The conversion process begins. An error displays, saying that a connection cannot be established until the data storage area (the dfPower repository) is upgraded.

11. Click on **OK** to dismiss the message.

12. Right-click the new repository in the **Administration** tree. Then select **Upgrade** from the context menu. A warning displays, saying that "older versions of the client" cannot connect to the repository after the repository is upgraded. This means that dfPower Studio will not be able to connect to this repository after it is upgraded to Data Management Studio format. That is why you are converting a copy of the repository, as described in Step 1.

13. Click **Yes** to finish the conversion. A status dialog displays, showing the progress of the conversion. When the conversion is finished, you should see a message that says that the repository was upgraded successfully. The next step is to connect to the new repository.

14. Right-click the new repository in the **Administration** tree. Then select **Connect**. The Status pane on the right should indicate that you are connected to the repository.

15. Verify that the new repository is working properly by browsing its contents. See Browsing the Results After Migration.

16. A typical next task is to migrate Architect Jobs.

Migrate a File-Based Repository That Will Be Managed By a Data Management Server

This topic is appropriate under the following conditions:

- the dfPower repository is in SQLite format (an .RPS file)
- jobs in a Management Resource refer to rules and tasks in the dfPower repository
- the dfPower repository is managed by a DataFlux Data Integration Server
- the migrated repository will be managed by a DataFlux Data Management Server

In general, you will copy the dfPower repository from the DataFlux Data Integration Server to a folder structure that is accessible to Data Management Studio. You will convert the local dfPower repository and create a new repository configuration file (.RCF file) . Then you will upload the new repository configuration file to the Data Management Server.

Assume that both Data Management Studio and the DataFlux Data Management Server have been installed. Perform the following steps.

1. Create an empty folder structure that is accessible to Data Management Studio, such as the folders that are described in General Planning for Repositories. Create the **File_Storage** folders as well as the **Repository** folders.

2. Copy the dfPower repository from the Data Integration Server to an appropriate folder from Step 1. For example, you could copy the repository to a folder such as C:\dfMigrate\Repository.

3. Perform Steps 3-15 as described in Migrate a File-Based Repository That Will Not Be Managed By a Data Management Server. The goal is to migrate the local repository and create a repository configuration file (.RCF file). Be sure to specify a **File storage** location, using the file storage folders you created in Step 1. Repositories that are managed by DataFlux servers do not use this area to store file-based items like jobs. However, you can use this local path at a later stage for other purposes. For example, you can use it to store jobs that were originally deployed on a Data Integration Server and that require modification before deploying them to a Data Management Server.

4. Copy the repository configuration file from the Data Management Studio location to the Data Management Server location for .RCF files.

   Data Management Studio stores .RCF files in the document settings area: [*drive:*]\Documents and Settings\[*username*]\Application Data\DataFlux\DataManagement\[*version]*\repositories\[*Migrated Repository*].RCF.

   Data Management Server stores .RCF files in the program files area: [*SERVER_HOME*]\var\repositories\[*Migrated Repository*].RCF.

5. You should now have a fully-functional Data Management Studio repository on the Data Management Server.

A typical next task would be to migrate Architect jobs and upload them to the Data Management Server. In general, you will use Data Management Studio to migrate the jobs as described in Architect Jobs. Then you will upload the converted jobs to the Data Management Server, as described in Deploy a Job to a Data Management Server.

Migrate a DBMS-Based Repository That Will Not Be Managed by a Data Management Server

This topic is appropriate under the following conditions:

- the dfPower repository is in DBMS format (not SQLite)
- jobs in a Management Resource refer to rules and tasks in the dfPower repository
- the dfPower repository is not managed by a DataFlux Data Integration Server
- the migrated repository will not be managed by a DataFlux Data Management Server

In general, you will use DBMS commands to create a copy of the dfPower repository and create a new DBMS connection to the copy. Then you will convert the dfPower repository. Perform the following steps:

1. Create an empty folder structure that is accessible to Data Management Studio, such as the folders that are described in General Planning for Repositories. You can omit the **Repository** folder, but create the **File_Storage** folders.
2. Use DBMS tools to copy the repository.
3. Create a new ODBC connection that points to the copy.
4. Run Data Management Studio.
5. Click on the **Administration** riser bar. Then select the **Repository Definitions** folder in the **Administration** tree. The Repository Definitions pane displays on the right
6. In the Repository Definitions pane, click on the **New** button to create the new repository. The Add Repository Definition dialog appears.
7. Enter the name of your repository in the **Name** field.
8. In the **Data storage** section of the dialog, specify the **Database connection** for the dfPower repository from Step 3.

9. In the **File storage** section of the dialog, specify the file storage folders you created in Step 1. Repositories that are managed by DataFlux servers do not use this area to store file-based items like jobs. However, you can use this local path at a later stage for other purposes. For example, you can use it to store jobs that were originally deployed on a Data Integration Server and that require modification before deploying them to a Data Management Server.
10. Select or deselect the **Connect to repository at startup** checkbox or the **Private** checkbox as appropriate.
11. Click on the **OK** button. The conversion process begins. An error displays, saying that a connection cannot be established until the data storage area (the metadata database) is upgraded.
12. Click on **OK** to dismiss the message.
13. Right-click the new repository in the **Administration** tree. Then select **Upgrade** from the context menu. A warning displays, saying that "older versions of the client" cannot connect to the repository after the repository is upgraded. This means that dfPower Studio will not be able to connect to this repository after it is upgraded to Data Management Studio format. That is why you are migrating a copy of the repository, as described in Steps 2 and 3.
14. Click **Yes** to finish the conversion. A status dialog displays, showing the progress of the conversion. When the conversion is finished, you should see a message that says that the repository was upgraded successfully. The next step is to connect to the new repository.
15. Right-click the new repository in the **Administration** tree. Then select **Connect**. The Status pane on the right should indicate that you are connected to the repository.
16. Verify that the new repository is working properly by browsing its contents. See Browsing the Results After Migration.
17. A typical next task is to migrate Architect jobs as described in Architect Jobs.

Migrate a DBMS-Based Repository That Will Be Managed by a Data Management Server

This topic is appropriate under the following conditions:

- the dfPower repository is in DBMS format (not SQLite)
- jobs in a Management Resource refer to rules and tasks in the dfPower repository
- the dfPower repository is managed by a DataFlux Data Integration Server
- the migrated repository will be managed by a DataFlux Data Management Server

In general, you will use DBMS commands to create a copy of the dfPower repository and create a new DBMS connection to the copy. You will convert the dfPower repository and create a new repository configuration file (.RCF file). Then you will upload the new repository configuration file to the Data Management Server.

1. Perform Steps 1-16 in the previous section, Migrate a DBMS-Based Repository That Will Not Be Managed by a Data Management Server.

2. Copy the repository configuration file from the Data Management Studio location to the Data Management Server location for .RCF files.

   Data Management Studio stores .RCF files in the document settings area: [*drive:*]\Documents and Settings\[*username*]\Application Data\DataFlux\DataManagement\[*version]*\repositories\[*Migrated Repository*].RCF.

   Data Management Server stores .RCF files in the program files area: [*SERVER_HOME*]\var\repositories\[*Migrated Repository*].RCF.

3. You should now have a fully-functional Data Management Studio repository on the Data Management Server.

A typical next task is to Architect jobs and upload them to the Data Management Server. In general, you will use Data Management Studio to migrate the jobs as described in Architect Jobs. Then you will upload the converted jobs to the Data Management Server, as described in Deploy a Job to a Data Management Server.

Migrate a Management Resource With No dfPower Repository

This topic is appropriate when you want to migrate a Management Resource that does do not refer to rules and tasks in a dfPower repository, as described in Migration Scenario: No dfPower Repository, One or More Management Resources.

In general, you will create an empty folder structure such as the structure described in General Planning for Repositories. Creating such a structure ensures that you have the standard folders for deployable jobs. When you run the New Repository Definition wizard, you will specify the location of an empty folder where the new Data Management Studio repository will be added. You would also specify the location of an empty set of folders. Later, you will copy Architect jobs into these folders, as described in Copy Architect Jobs to the File Storage Area of the Repository.

Perform the following steps.

1. Create an empty folder structure to hold the new Data Management Studio repository and related assets, as described in General Planning for Repositories.

2. Run Data Management Studio.

3. Click on the **Administration** riser bar. Then select the **Repository Definitions** folder in the **Administration** tree. The Repository Definitions pane displays on the right.

4. In the Repository Definitions pane, click on the **New** button to create the new repository. The New Repository Definition dialog appears.

5. Enter the name of your repository in the **Name** field.

6. In the **Data storage** section of the dialog, specify a new **Database file** for the Data Management Studio repository that you want to create. For example, you could specify a path such as C:\New_Repos_C\Repository\Repository_New.rps.

   Note: Do not create a repository database file (.RPS) in the file storage location that you will specify in Step 7. This practice prevents manipulating the file through Data Management Studio, and it triggers unneeded update events in the file storage area every time that the database file is updated.

7. In the **File storage** section of the dialog, specify an appropriate, empty folder from Step 1. For example, you could specify a path such a C:\New_Repos_C\File_Storage. (Later, you will copy unconverted dfPower Architect jobs to this area and save converted copies of these jobs to the same area.)

8. Click on **OK** to save the new repository. You will get a message that asks if you want to create a new repository.

9. Click **Yes**. The repository is created.

10. Verify that the new repository is working properly by browsing its contents. See Browsing the Results After Migration.

11. Copy the relevant Architect jobs from one or more Management Resources to an appropriate folder in the file storage area of the new repository, as described in Copy Architect Jobs to the File Storage Area of the Repository.

Post Migration Tasks for Repositories

Browsing the Results After Migration

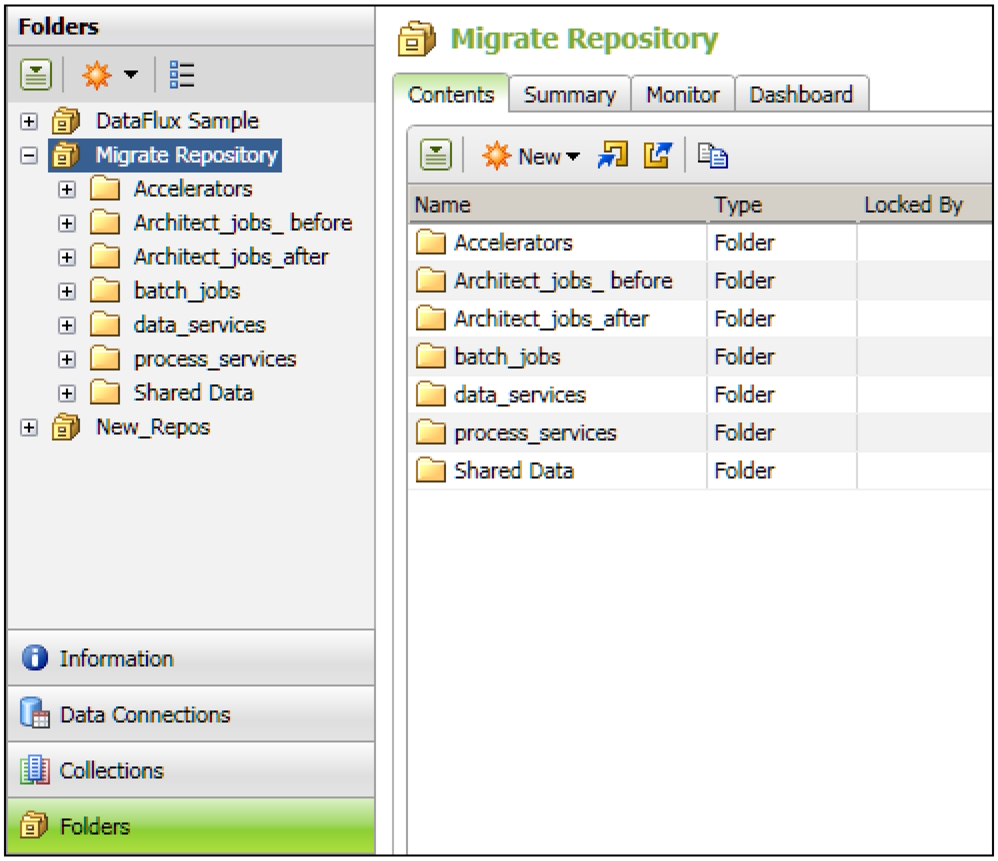
The following steps demonstrate one way to browse a migrated repository in Data Management Studio. It is assumed that you are in Data Management Studio and are connected to the migrated repository.

1. To explore the contents of the migrated repository, click on the **Folders** riser.

2. Select the new repository in the **Folders** tree.

3. Toggle between the two main views in the **Folders** tree by selecting and deselecting the **Group Items by Type** control at the top of the tree.

   When the **Group Items by Type** control is not selected, items are grouped into user-defined folders, as shown in the next display.

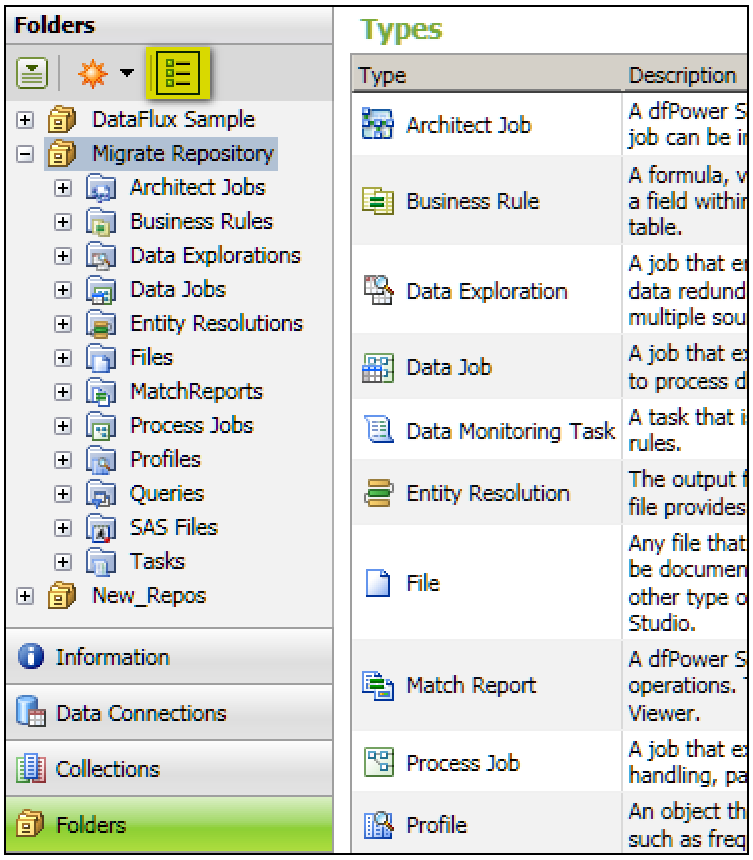


Name	Type	Locked By
Accelerators	Folder	
Architect_jobs_ before	Folder	
Architect_jobs_after	Folder	
batch_jobs	Folder	
data_services	Folder	
process_services	Folder	
Shared Data	Folder	

   The migration process will create some user-defined folders automatically, based on any sub-folders in the **File storage** area. Most of the folders in the display above were created based on the folder structure that is described in General Planning for Repositories. These folders are empty at this point because we have migrated only the items that were in the dfPower repository (metadata database), not the file-based items, such as Architect jobs, that are stored in any dfPower Studio Management Resource.

   The **Shared Data** folder is different. It is automatically created for a migrated dfPower repository. It displays any business rules or tasks that were defined in the dfPower repository. These items are stored in the **Data storage** area (metadata database area) of the migrated repository, but references to these items are displayed in the **Shared Data** folder for convenience.

4. When the **Group Items by Type** control is selected, items are grouped into system-defined folders that are named after general types, such as **Architect Jobs**, **Business Rules**, as shown in the next display.



Folders

- DataFlux Sample
- Migrate Repository
  - Architect Jobs
  - Business Rules
  - Data Explorations
  - Data Jobs
  - Entity Resolutions
  - Files
  - MatchReports
  - Process Jobs
  - Profiles
  - Queries
  - SAS Files
  - Tasks
- New_Repos

Types

Type	Description
Architect Job	A dfPower S job can be i
Business Rule	A formula, v a field withi table.
Data Exploration	A job that e data redund multiple sou
Data Job	A job that e to process d
Data Monitoring Task	A task that i rules.
Entity Resolution	The output f file provides
File	Any file that be documen other type o Studio.
Match Report	A dfPower S operations. Viewer.
Process Job	A job that e handling, pa
Profile	An object th such as freq

5. With the **Group Items by Type** control selected, expand the migrated repository and expand the folders for each type of object. Most of these folders will be empty at this point because we have migrated only the items that were in the dfPower repository (metadata database), not the file-based items, such as Architect jobs, that are stored in any dfPower Studio Management Resource.

   The **Business Rules** folder contains Business Rules that have been converted to Data Management Studio format and will often be ready to use. Similarly the **Tasks** folder contains any Monitor Tasks (tasks created in the Business Rule Manager) that have been converted and will often be ready to use. For more information, see Business Rules, Tasks, and Related Items.

Miscellaneous Post-Migration Tasks

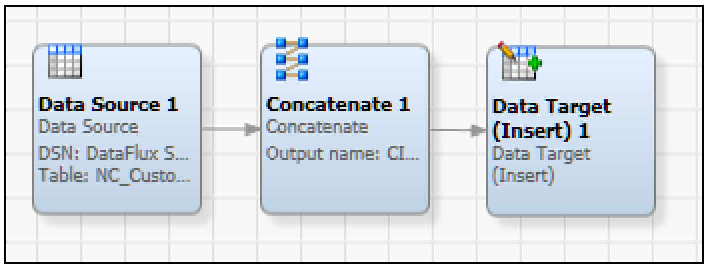
Delete any Data Explorations migrated from dfPower Studio. The migration software will move **dfPower Studio** Data Explorations into the Folders tree in Data Management Studio. You cannot use these items in Data Management Studio, however, so you can delete them. To delete these items after migration, click the **Folders** riser in Data Management Studio. Then click the **Group Items by Type** icon at the top of the Folders tree so that items are grouped in folders with type names, such as Architect Jobs, Business Rules, and so on, Select the folder for Data Explorations. A list of Data Explorations displays on the right. Select all of the Data Explorations that you want to delete, then click the **X** icon in the tool bar.

Architect Jobs

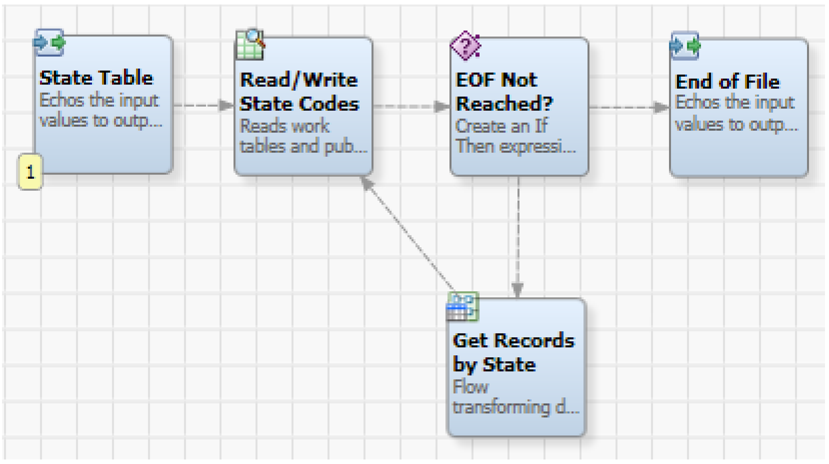
Changes to Architect Jobs

New Job Architecture

Data Management Studio has two kinds of jobs that are used for data integration data jobs and process jobs. **Data jobs** are the main way to process data in Data Management Studio. Each data job specifies a set of data-processing operations that flow from source to target, as illustrated in the next display.



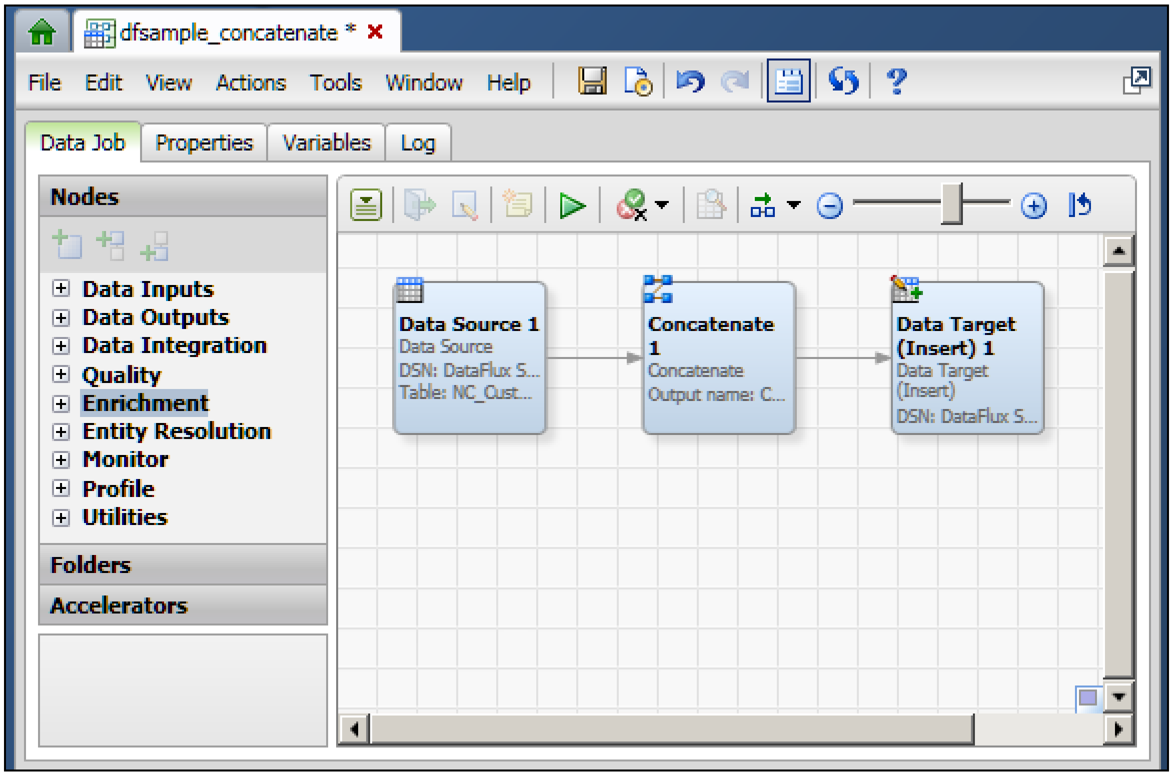
Process jobs combine data processing with conditional processing. The process flow in the job supports logical decisions, looping, events and other features that are not available in a data job flow. Data Job nodes can be added to a process flow to encapsulate all of the data processing power of a data job into a node in the process flow.


State Table
Echos the input values to outp...
1
Read/Write State Codes
Reads work tables and pub...
EOF Not Reached?
Create an If Then expressi...
End of File
Echos the input values to outp...
Get Records by State
Flow transforming d...

Some Architect jobs can be converted to either a data job or a process job. Other Architect jobs must be converted to one of these job types. In either case, the upgrade job wizards enable you to perform a correct conversion, as described later in this topic.

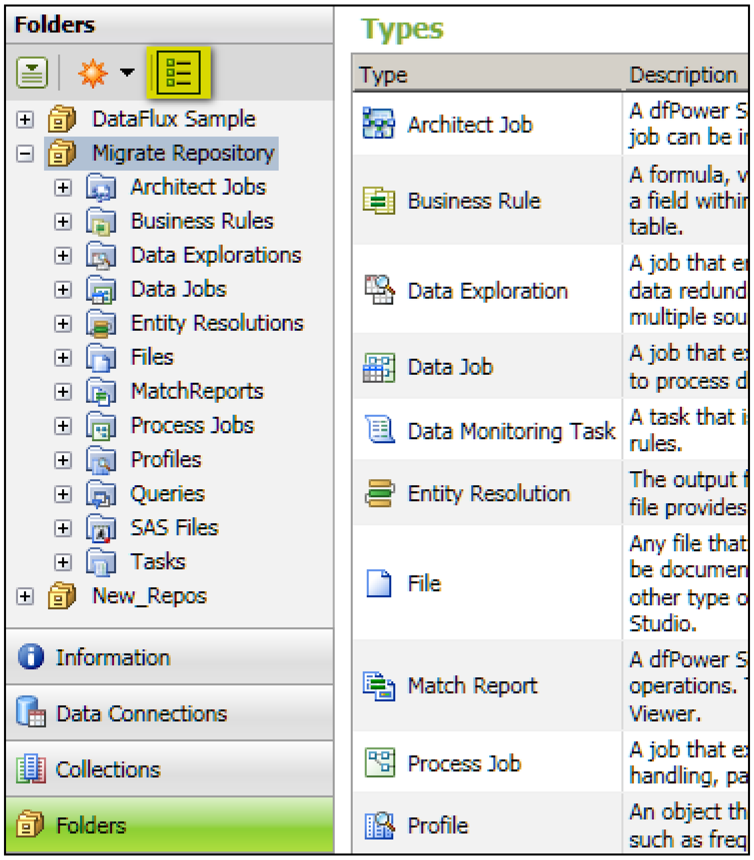
New Job Interfaces

Data jobs and process jobs have a similar editor. The next display shows the data job editor.



For more information about using the data job editor and the process job editor, see the "Data Jobs" and "Process Jobs" chapters of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

The **Folders** riser is where you manage Architect jobs before and after they have been converted to Data Management Studio jobs. In the **Folders** tree shown in the next display, the **Group Items by Type** control is selected.


Folders

- DataFlux Sample
- Migrate Repository
  - Architect Jobs
  - Business Rules
  - Data Explorations
  - Data Jobs
  - Entity Resolutions
  - Files
  - MatchReports
  - Process Jobs
  - Profiles
  - Queries
  - SAS Files
  - Tasks
- New_Repos

Information

Data Connections

Collections

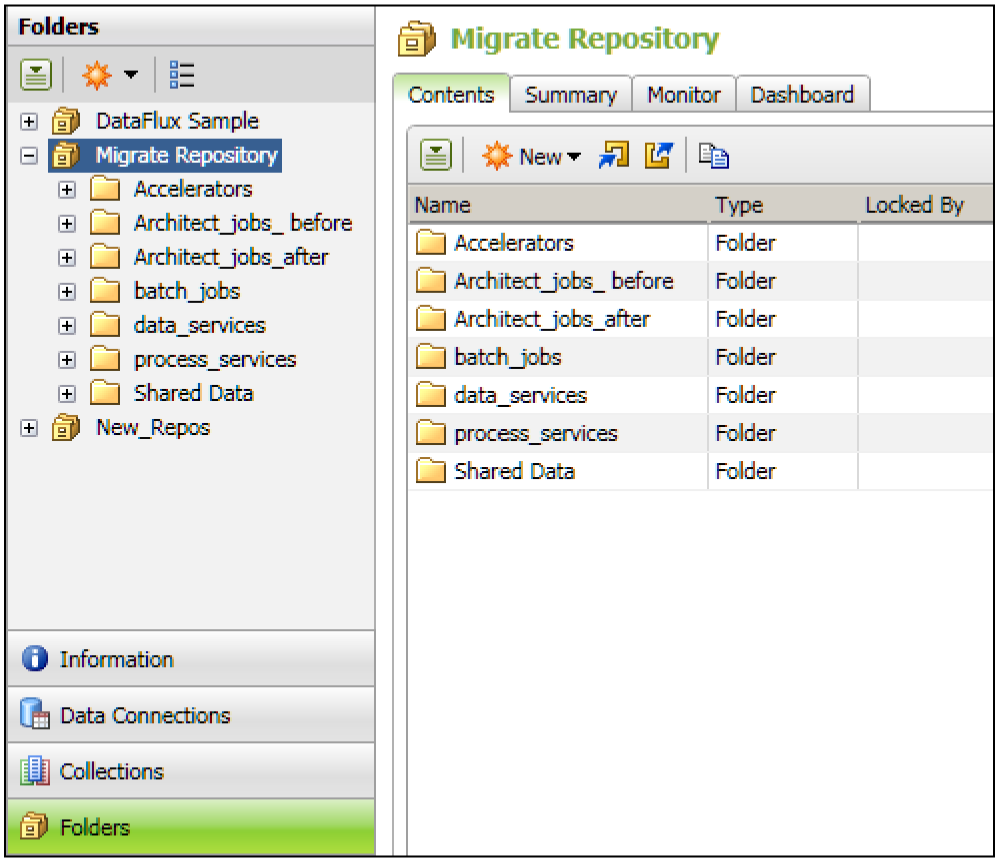
Folders

Types

Type	Description
Architect Job	A dfPower S job can be i
Business Rule	A formula, v a field withi table.
Data Exploration	A job that e data redund multiple sou
Data Job	A job that e to process d
Data Monitoring Task	A task that i rules.
Entity Resolution	The output f file provides
File	Any file that be documen other type o Studio.
Match Report	A dfPower S operations. Viewer.
Process Job	A job that e handling, pa
Profile	An object th such as freq

The **Group Items by Type** control groups items into system-defined folders that are named after general types, such as **Architect Jobs**, **Business Rules**, and so on. The **Architect Jobs** folder contains Architect jobs in their original, dfPower Studio format, after you have copied them to the **File storage** area of the migrated repository, as described in Copy Architect Jobs to the File Storage Area of the Repository. You cannot open Architect jobs that have not been converted to Data Management Studio format.

If you deselect the **Group Items by Type** control, items are grouped into user-defined folders, as shown in the next display.



- DataFlux Sample
- Migrate Repository
  - Accelerators
  - Architect_jobs_ before
  - Architect_jobs_after
  - batch_jobs
  - data_services
  - process_services
  - Shared Data
- New_Repos

Name	Type	Locked By
Accelerators	Folder	
Architect_jobs_ before	Folder	
Architect_jobs_after	Folder	
batch_jobs	Folder	
data_services	Folder	
process_services	Folder	
Shared Data	Folder	

The migration process will create some user-defined folders automatically, based on any sub-folders in the **File storage** area. Most of the folders in the display above were created based on the folder structure that is described in General Planning for Repositories. Most of these folders will be empty until you copy or save appropriate file-based objects into them, as will be explained later.

New Storage Constraints

As explained in New Repository Architecture, Data Management Platform puts new constraints on where jobs can be stored. In Data Management Platform:

- Each Data Management Platform repository can have at most one data storage area and one file storage area.
- All related items must be stored in the same Data Management Platform repository. This means, for example, that a job that is stored in a repository can reference only those business rules, tasks, or similar items that are stored in the same repository.

Differences in metadata architecture might require you to make some manual adjustments when you migrate dfPower Studio repositories and Management Resources to Data Management Studio. For more information, see Plan the Migration of Each Repository and Management Resource.

Other Changes

See Post-Migration Tasks for Architect Jobs.

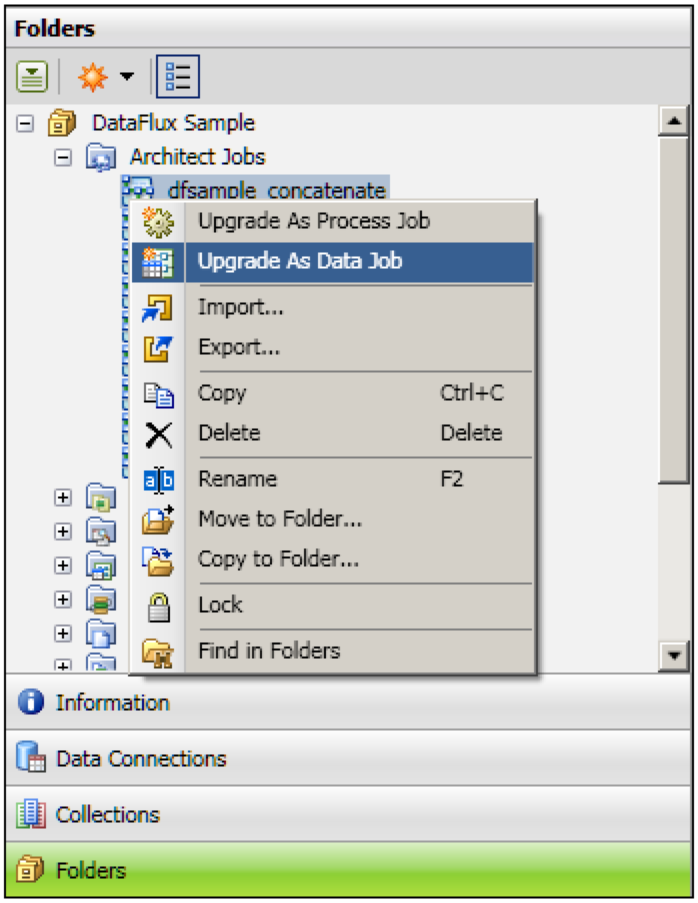
Plan the Migration of Architect Jobs

Review Overview of Migrating from dfPower Studio. Try to anticipate the main issues that could affect the conversion of your Architect jobs. It is always best to know as much as possible about the Architect jobs that you are migrating. That way you can identify changes and possible problems after migration. Also, it will be easier to test the migrated jobs if you have a basic understanding of Data Management Studio data jobs and process jobs, as described in the "Data Jobs" and "Process Jobs" chapters of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Migrating Architect Jobs

Copy Architect Jobs to the File Storage Area of the Repository

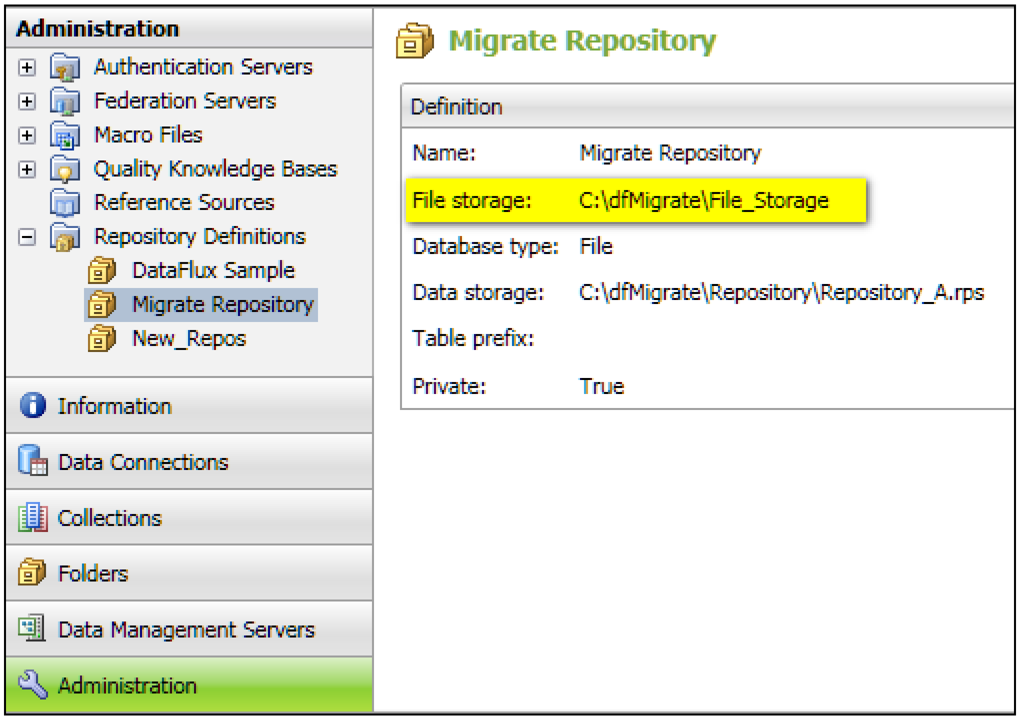
You will use the upgrade job wizards in Data Management Studio to convert Architect jobs to process jobs or data jobs. The upgrade job wizards will work only on Architect jobs that are visible in the **Folders** tree in Data Management Studio, as shown in the next display.



In order for an unconverted Architect job to appear in the **Folders** tree, you must copy it from a dfPower Studio Management Resource to an appropriate folder within the **File storage** area of the migrated repository. You can use operating system tools to accomplish this task.

The first task is to identify the path to the **File storage** area of the migrated repository.

1. Click on the **Administration** riser bar.
2. Expand the **Repository Definitions** folder and select the migrated repository. The physical path to the top-level folder for the **File storage** area will be displayed on the right. For example, the physical path to the top-level folder for the **File storage** area might be might be as shown in the next display.

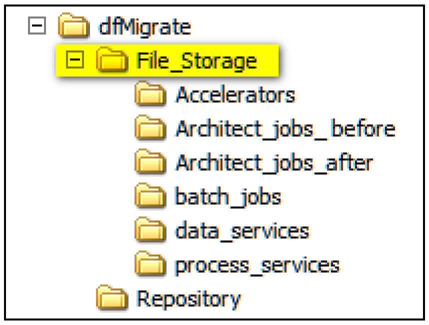


Migrate Repository

Definition

Name:	Migrate Repository
File storage:	C:\dfMigrate\File_Storage
Database type:	File
Data storage:	C:\dfMigrate\Repository\Repository_A.rps
Table prefix:	
Private:	True

Typically, you will not copy unconverted Architect jobs to the top-level folder for the **File storage** area. You will copy them to a sub-folder within that area. For example, the **File storage** area for the **Migrate Repository** might have the sub-folders that are shown in the next display.



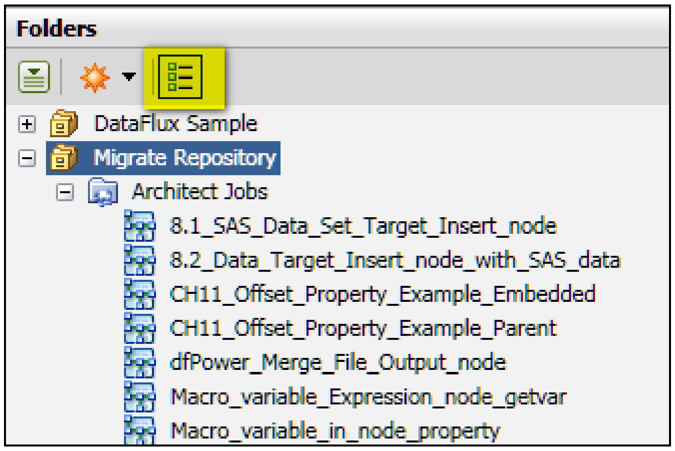
3. Use operating system tools to copy unconverted Architect jobs from a dfPower Studio Management Resource to an appropriate sub-folder within the **File storage** area. For example, given the folder structure shown above, you could use operating system tools to copy unconverted Architect jobs to the folder C:\dfMigrate\File_Storage\Architect_jobs_ before.

4. The next task is to verify that you can see the unconverted Architect jobs in the **Folders** tree. Click on the **Folders** riser.

5. Select the migrated repository in the **Folders** tree.

6. Select the **Group Items by Type** control at the top of the tree. Items will grouped into system-defined folders that are named after general types, such as **Architect Jobs** and **Business Rules**.

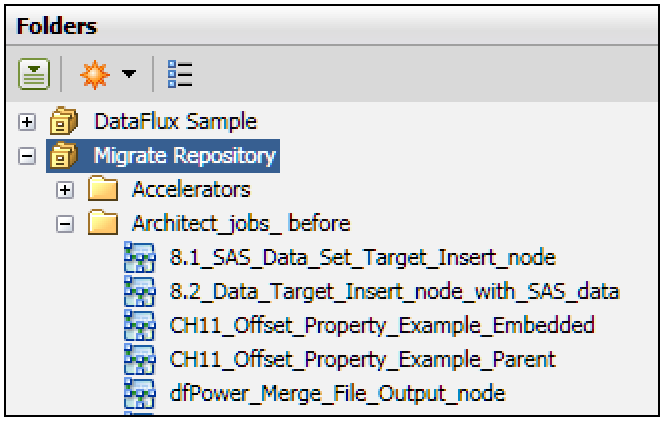
7. Expand the **Architect Jobs** folder. You should see the unconverted Architect jobs that you copied into the **File storage** area, as shown in the next display.



8.1_SAS_Data_Set_Target_Insert_node
8.2_Data_Target_Insert_node_with_SAS_data
CH11_Offset_Property_Example_Embedded
CH11_Offset_Property_Example_Parent
dfPower_Merge_File_Output_node
Macro_variable_Expression_node_getvar
Macro_variable_in_node_property

8. You can also view these jobs in the user-defined folders view. De-select the **Group Items by Type** control at the top of the tree. Items will grouped into user-defined folders.

9. Expand the user-defined folder where you copied the unconverted Architect jobs, as shown in the next display.



8.1_SAS_Data_Set_Target_Insert_node
8.2_Data_Target_Insert_node_with_SAS_data
CH11_Offset_Property_Example_Embedded
CH11_Offset_Property_Example_Parent
dfPower_Merge_File_Output_node

You can now access the unconverted Architect jobs in the Data Management Studio **Folders** tree. Before you convert any jobs, review the conversion characteristics that are described in the next section.

Migrate to a Data Job or a Process Job?

You will use the upgrade job wizards to convert an Architect job to a process job or a data job, as permitted by the wizards. The wizards evaluate the following conditions.

- Architect job with a single page that does not have an **External Data Provider** node in the job flow.
  You can select **Upgrade as a Data Job** or **Upgrade as a Process Job**. You might want to select **Upgrade as a Data Job** until you become familiar with process jobs.

- Architect job with a single page that has an **External Data Provider** node in the job flow.
  Select **Upgrade as a Data Job**. If you want to use the data provider job in multiple jobs, then it is preferable to convert it to a data job. This allows you to link to it from many process jobs and only maintain one copy of the data provider job that is used in multiple process jobs.

- Architect job with multiple pages.
  Select **Upgrade as a Process Job**. If you choose **Upgrade as a Data Job** you will get an error.

You can convert a set of similar jobs at the same time. The jobs should have the same conversion characteristics. To select multiple jobs, select the folder that contains the jobs in the **Folders** tree. The jobs in that folder will display on the right. Use standard selection methods to select multiple jobs. Then right-click a selected job and select the appropriate upgrade job wizard.

Convert an Architect Job to a Data Job or a Process Job

This topic is appropriate under the following conditions:

- You want to convert an Architect job to a Data Management Studio data job or a process job.

- You have migrated the dfPower repository that contains rules, tasks, or similar items that are used in the job, as described in Repositories and Management Resources.

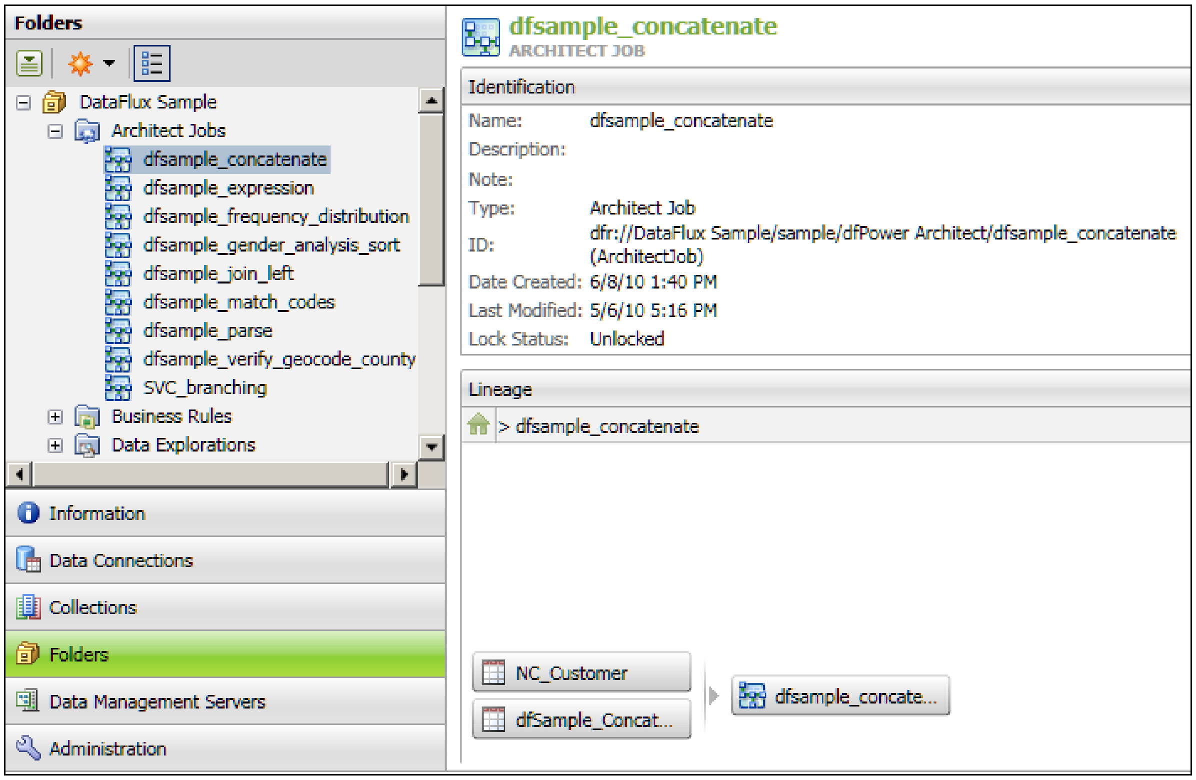
- You have copied Architect jobs into the **File storage** area of the migrated repository, as described in Copy Architect Jobs to the File Storage Area of the Repository.

- You have a general idea whether the Architect job should be converted to a data job or a process job, as described in Migrate to a Data Job or a Process Job?

The steps for converting an Architect job to a Data Management Studio data job are similar to the steps for converting an Architect job to a process job. The differences between data jobs and process jobs become more important after the conversion from Architect format. Some of the converted jobs will work immediately after conversion. Others will require manual updates, as described in Post-Migration Tasks for Architect Jobs.

The first task to find the Architect job that you want to convert to a data job or process job. One way to do that is to open the appropriate repository in the **Folders** tree, then put the tree in **Group Items by Type** mode.

1. Click on the **Folders** riser bar.
2. Select the **Group Items by Type** control at the top of the tree. This puts the tree in **Group Items by Type** mode.
3. Expand the repository that contains the Architect job to be converted.
4. Expand the **Architect Jobs** folder.
5. Select the job that you want to convert. The **Folders** tree will look similar to the next display.



dfsample_concatenate
ARCHITECT JOB

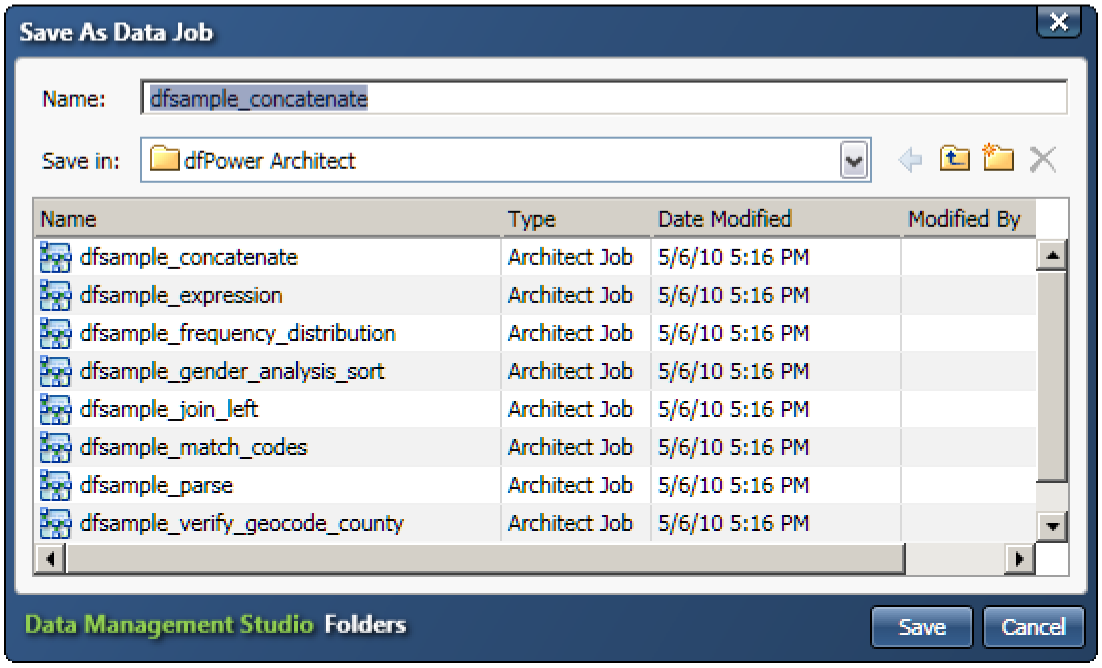
Identification

Name:	dfsample_concatenate
Description:	
Note:	
Type:	Architect Job
ID:	dfr://DataFlux Sample/sample/dfPower Architect/dfsample_concatenate (ArchitectJob)
Date Created:	6/8/10 1:40 PM
Last Modified:	5/6/10 5:16 PM
Lock Status:	Unlocked

In the previous display, the Architect job **dfsample_concatenate** is selected in the **Folders** tree. Information about that job is displayed on the right. Note the **Lineage** pane on the right, which displays the inputs to the job: the (**NC_Customer** table and the **dfSample_Contatenate** table.

If an input table has an invalid path, a red **X** decoration would be displayed on the icon for the table. If the selected Architect job has any invalid inputs, you cannot correct them now. You will have to convert the job to Data Management Studio format and correct that version.

6. Right-click the selected Architect job and select **Upgrade as a Data Job** or **Upgrade as a Process Job**, as appropriate. A **Save As** dialog appears. The next display shows an example of the **Save As Data Job** dialog. The **Save As Process Job** dialog is similar.



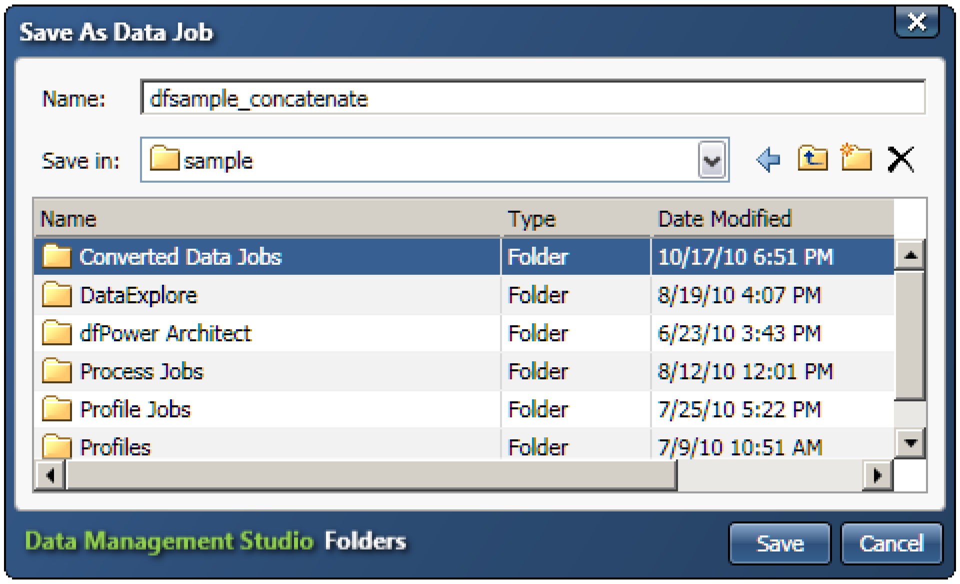
Name: dfsample_concatenate

Save in: dfPower Architect

Name	Type	Date Modified	Modified By
dfsample_concatenate	Architect Job	5/6/10 5:16 PM	
dfsample_expression	Architect Job	5/6/10 5:16 PM	
dfsample_frequency_distribution	Architect Job	5/6/10 5:16 PM	
dfsample_gender_analysis_sort	Architect Job	5/6/10 5:16 PM	
dfsample_join_left	Architect Job	5/6/10 5:16 PM	
dfsample_match_codes	Architect Job	5/6/10 5:16 PM	
dfsample_parse	Architect Job	5/6/10 5:16 PM	
dfsample_verify_geocode_county	Architect Job	5/6/10 5:16 PM	

7. Verify that the **Save As** dialog will save the converted Architect job to the appropriate user-defined folder in the **Folders** tree. For example, in the previous display, the **Save as a Data Job** dialog will save the converted Architect job to the **dfPower Architect** folder where the unconverted Architect jobs are. This is probably not what you want.

8. To specify a different user-defined folder, click on the folder icon with the up arrow and select a different folder in the current repository, such as the **Converted Data Jobs** folder shown in the next display.



Name: dfsample_concatenate

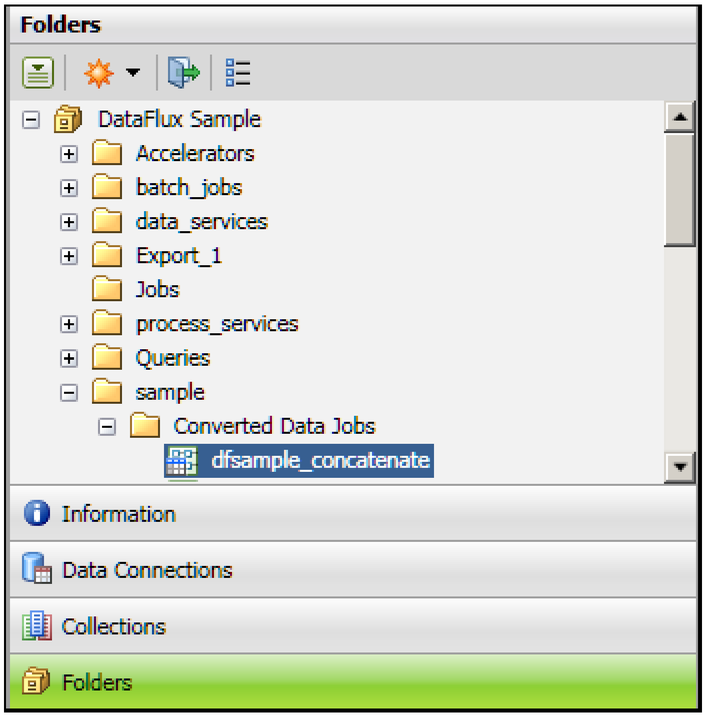
Save in: sample

Name	Type	Date Modified
Converted Data Jobs	Folder	10/17/10 6:51 PM
DataExplore	Folder	8/19/10 4:07 PM
dfPower Architect	Folder	6/23/10 3:43 PM
Process Jobs	Folder	8/12/10 12:01 PM
Profile Jobs	Folder	7/25/10 5:22 PM
Profiles	Folder	7/9/10 10:51 AM

9. When the appropriate target folder is selected, click **Save**. The Architect job will be converted to Data Management Studio format and saved to the selected user-defined folder in the **Folders** tree.

10. To verify that the converted Architect job was saved to the correct folder, put the **Folders** tree in user-defined folders mode: de-select the **Group Items by Type** control at the top of the tree.

11. Navigate to the folder where the job should have been saved. For example, in the following display, the converted job **dfsample_concatenate** has been saved to the **Converted Data Jobs** folder.



You are now ready to inspect the new job and perform any other post-migration tasks. See Post-Migration Tasks for Architect Jobs.

Post-Migration Tasks for Architect Jobs

Inspect Architect Jobs After Migration

After you convert an Architect job to a Data Management Studio job, perform the following steps to verify that the job is working properly. You will need a basic understanding of Data Management Studio jobs, as described in the "Data Jobs" and "Process Jobs" chapters of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

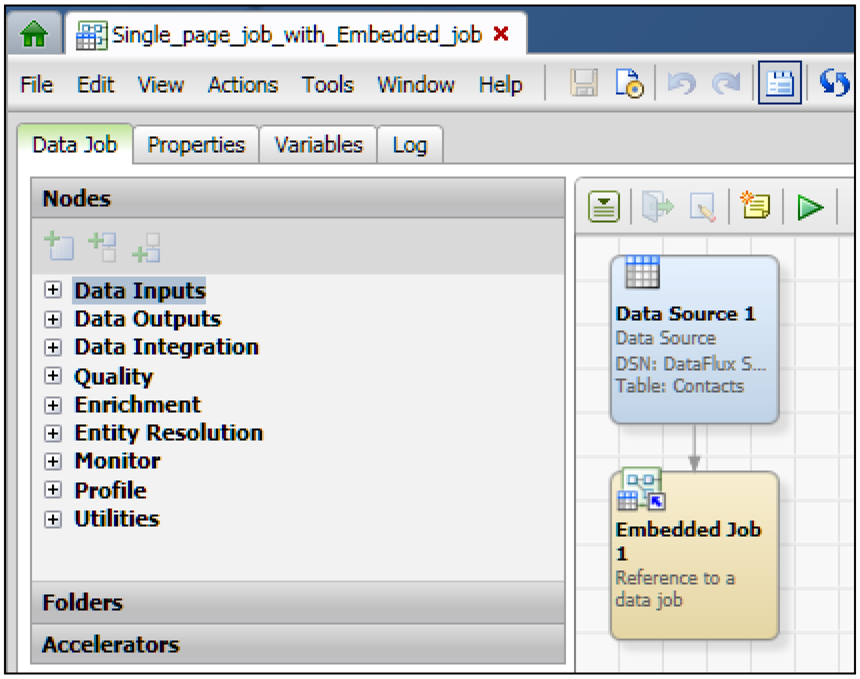
1. Right-click the job in the **Folders** tree and select **Open**. The job will be opened in a job editor.
2. If appropriate, try to run the job to see if it works as it did before migration. Select **Actions** > **Run Data Job** or **Run Process Job** from the main menu.
3. The output, status messages, the log, and other indicators will tell you whether the job ran properly and will help you troubleshoot any errors.

Architect Jobs That Use Macro Variables

You will probably have to make some updates in order for dfPower Studio macro variables to work in Data Management Studio data jobs or process jobs. General updates for macros are described in Macro Variables.

Macro Variables and Multi-Page Jobs. See Architect Jobs That Pass Macro Variable Values Dynamically Between Pages in the Job.

Macro Variables and Embedded Jobs. Architect jobs with Embedded Job nodes will be converted to data jobs with **Data Job (reference)** nodes, as shown in the next display.



In the previous figure, **Embedded Job 1** is a **Data Job (reference)** node. To pass macro variable values down to the embedded job, you would right-click the **Data Job (reference)** node and select **Advanced Properties**. Then you would specify the macro variable in the KEY_VALUES property.

To get macro variable values out of the embedded job, you would right-click the **Data Job (reference)** node and select **Advanced Properties**. Then specify the macro variable in the OUT_KEY_VALUES property.

Deploy a Job to a Data Management Server

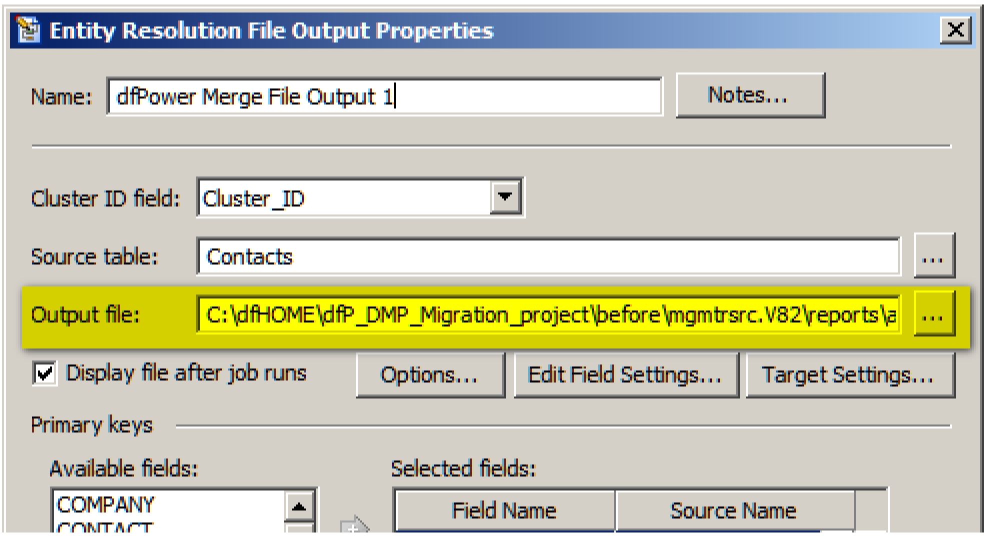
After you have converted an Architect job to DataFlux Data Management Studio format, and you have completed any other post-migration tasks for that job, you can deploy it to a Data Management Server. For information about deploying jobs to a Data Management Server, see the following sections in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*:

- "Deploying a Data Job as a Real-Time Service" topic in the "Data Jobs" chapter
- "Deploying a Process Job as a Real-Time Service" topic in the "Process Jobs" chapter

Architect Jobs That Contain a Merge File Output Node

When you migrate a job that contains the **dfPower Merge File Output** node, the merge file node is upgraded to an **Entity Resolution File Output** node. After migration, perform these steps to ensure that the job will work properly:

1. In the **Folders** tree, right-click the Data Management Studio job that contains the **Entity Resolution File Output** node and select **Open**. The job opens in an editing window.
2. In the job flow, right-click the **Entity Resolution File Output** node and select **Edit**. The properties dialog for the node displays.
3. Inspect the path in the **Output file** field, as shown in the next display.



Entity Resolution File Output Properties

Name: dfPower Merge File Output 1

Cluster ID field: Cluster_ID

Source table: Contacts

Output file: C:\dfHOME\dfP_DMP_Migration_project\before\mgmtrsrc.V82\reports\a

☑ Display file after job runs

4. Verify that the file that is specified in the **Output File** field has a path within the **File storage** area of the current repository. Otherwise, you will not be able to select and view the entity resolution output file in the **Folders** tree. Use the file selection control to specify a new path within the **File storage** area, if needed. For example, you might specify a path such as: C:\dfMigrate\MANAGEMENT_RESOURCE\reports\entity_resolution\dfPower_Merge_File_Output_node.sri
5. Verify that the file that is specified in the **Output File** field has a .SRI extension. If the file does not have an .SRI extension, the extension will be automatically changed when you click OK to save your changes.

   The **Entity Resolution File Output** node has a new output file format and a new file extension, the .SRI file extension. The entity resolution viewer can only display output with the .SRI extension. However, the migration software does not automatically update the old extension to .SRI. Accordingly, you must update the extension as described in this step or otherwise update the extension.
6. Click OK to save your changes.
7. Test the job by running it and viewing the entity resolution output file.

Architect Jobs That Contain a Match Report Node

A job that contains a **Match Report** node can be migrated like other Architect jobs. To view **Match Report** output in Data Management Studio, select **Tools** > **Match Report Viewer**, then navigate to the **Match Report** output file (.MDF file).

Architect Jobs That Contain Deprecated Nodes

The following dfPower Studio nodes have been deprecated in Data Management Studio. They should be deleted and replaced as described below.

COM Plugin. Replace this node with the new **Java™ Plugin** node.

SAS Data Set Source. To replace this node, define a SAS data set connection in the **Data Connections** riser, then use the **Data Source** node to point to the SAS data sets in that connection.

SAS Data Set Target (insert). To replace this node, define a SAS data set connection in the **Data Connections** riser, then use the **Data Target (Insert)** node to point to the SAS data sets in that connection.

SAS SQL Query. To replace this node, define a SAS data set connection in the **Data Connections** riser, then use the **SQL Query** node to specify a query against a SAS data set.

Architect Jobs That Pass Macro Variable Values Dynamically Between Pages in the Job

Overview

This topic is appropriate under the following conditions:

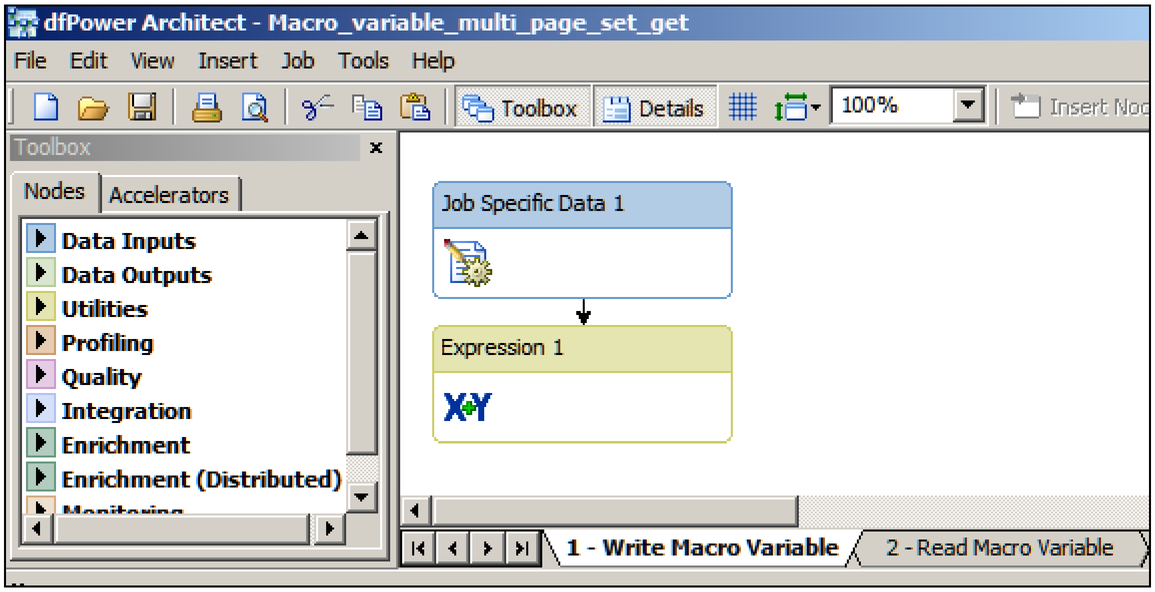
- You had a multi-page Architect job that used the setvar() and getvar() functions to dynamically pass macro variable values between pages in the job.
- You converted this Architect job to a Data Management Studio process job.

A multi-page Architect job can use the setvar() and getvar() functions to dynamically pass macro variable values between pages in the job. A Data Management Studio process job can use input and output variables to do something similar. The two methods are different enough, however, that some manual updates are required when you migrate a multi-page Architect job that dynamically passes values between pages in the job. Without these updates, the new process job will not produce the correct output.

In order to perform these updates, you must understand how the setvar() and getvar() functions were used in the original Architect job. Then you must recreate the same functionality, using input and output variables. This task will be easier if you understand how input and output variables are used in process jobs, as described in the "Create a Process Job" topic in the "Process Jobs" chapter of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Understanding an Architect Job That Passes Macro Variable Values Dynamically

For the current example, assume that the original Architect job is similar to the **Macro_variable_multi_page_set_get** job that is shown in the next display.

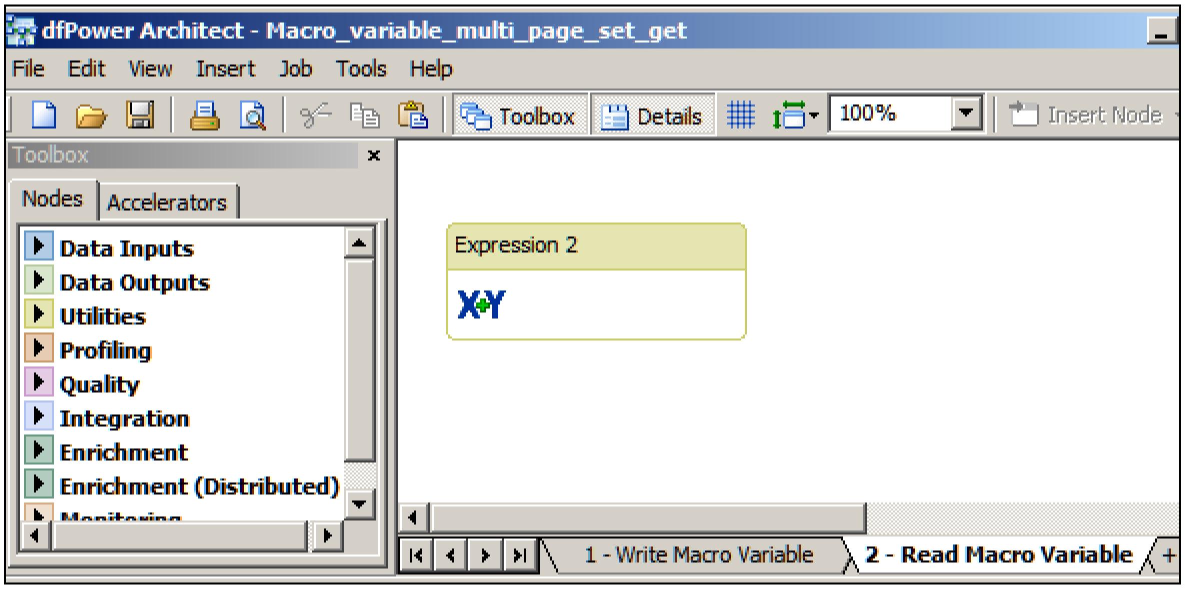


Note that the job has two pages: **Write a Macro Variable** and **Read a Macro Variable**.

On the **Write a Macro Variable** page, the flow reads data values from the **Job Specific Data 1** node, one value at a time. Each read operation increments a macro variable (**_nrecords_snlwih**) in the **Expression 1** node. The **Expression 1** node specifies the following expressions.

```
hidden integer _recordnr
_recordnr=0
_recordnr=_recordnr+1
setvar("_nrecords_snlwih", _recordnr)
```

The **Read a Macro Variable** page consists of a single **Expression** node, as shown in the next display.



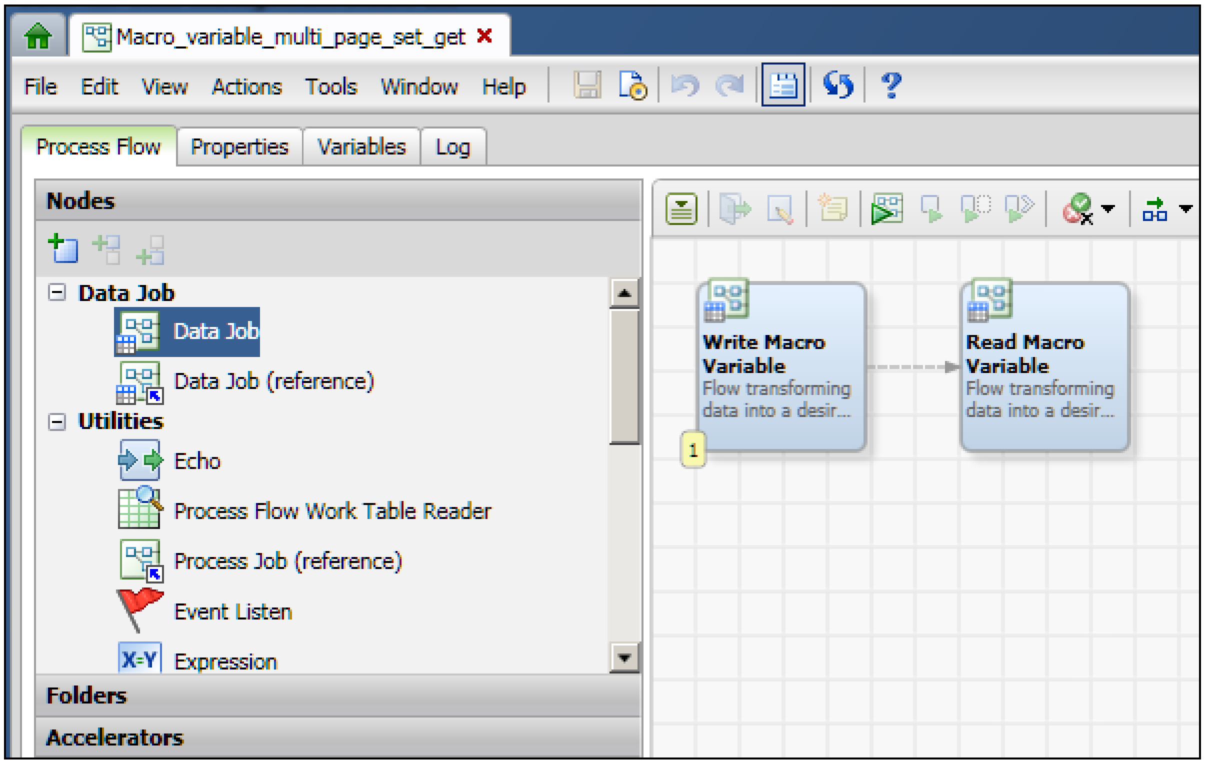
The **Expression 2** node specifies the following expressions:

```
string _mytext
_mytext=getvar("_nrecords_snlwih")
file f
f.open("c:\temp\stats.txt","a")
f.writeline("number of records processed: " & _mytext)
f.close()
seteof()
```

This expression reads the macro variable that was set on the first page and uses it to write a line of text to a file: "number of records processed: [*_nrecords_snlwih*]".

Specify an Output Variable in a Process Job

After conversion to a Data Management Studio process job, the **Macro_variable_multi_page_set_get** job would look similar to next display.



Note that the converted job has two Data Job nodes: **Write a Macro Variable** and **Read a Macro Variable**. These two nodes play the same role as the pages of the same name in the Architect job. Each of these nodes contains a data flow that is similar to the corresponding page in the Architect job.

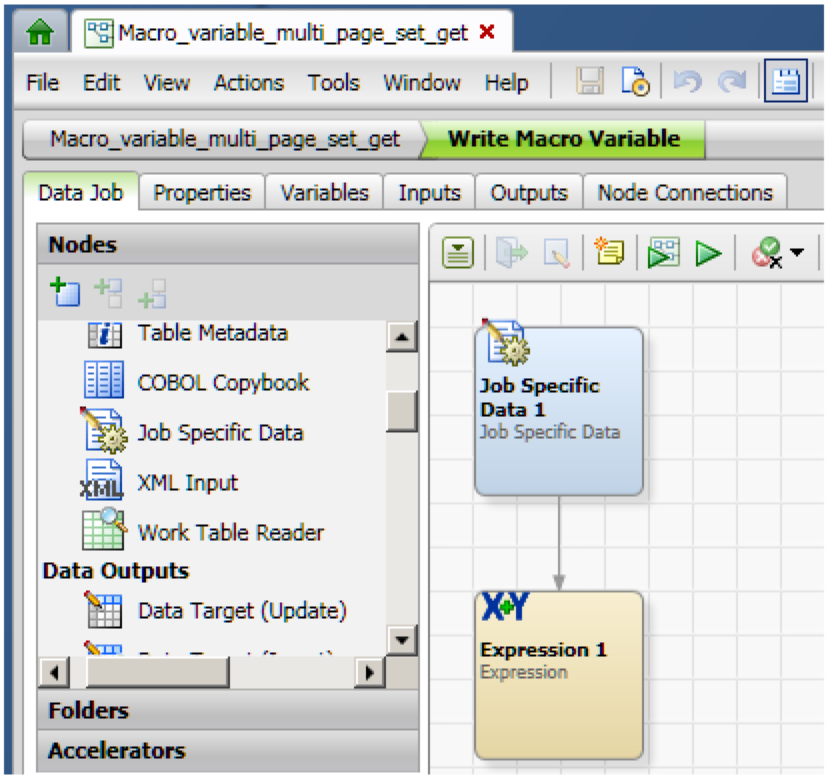
If you were to run the process job, however, the output would not be correct. For the current example, the log would show that some number of data values were read, but the text in the output file would display a null value for the number of records processed. This error is caused by a change in the way that macro variables are handled in the context of Data Management Studio jobs. Macro variables alone cannot be used to pass values between two data flows, such as the flows in the two Data Job nodes, **Write a Macro Variable** and **Read a Macro Variable**. You must use input and output variables to do this.

To correct the converted **Macro_variable_multi_page_set_get** job, you could:

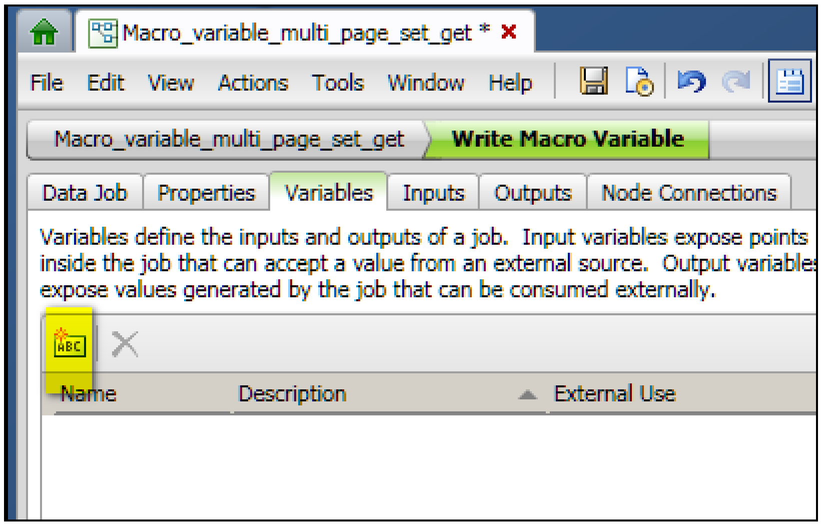
- Add an output variable to the **Write Marco Variable** node to represent the macro variable that was created with setvar() function in the original Architect job. Give the output variable the same name as the macro variable.
- Add an input variable to the **Read Macro Variable** node to represent the macro variable that was created with getvar() function in the original Architect job. Give the input variable the same name as the macro variable.
- Define a source binding for the input variable that binds it to the output variable.

You could perform the following steps to add an output variable to the **Write Marco Variable** node.

1. Right-click the **Write Marco Variable** node and select **Edit**. The data flow for this node opens, as shown in the next display.



2. Click on the **Variables** tab, one of the tabs above the **Nodes** tree. The **Variables** tab displays.


Macro_variable_multi_page_set_get *
File Edit View Actions Tools Window Help
Macro_variable_multi_page_set_get
Write Macro Variable
Data Job Properties Variables Inputs Outputs Node Connections
Name
Description
External Use

Variables define the inputs and outputs of a job. Input variables expose points inside the job that can accept a value from an external source. Output variable expose values generated by the job that can be consumed externally.

3. Put your cursor on the **ABC** icon at upper left of the **Variables** tab. The hover text should say: **Insert New Variable**.

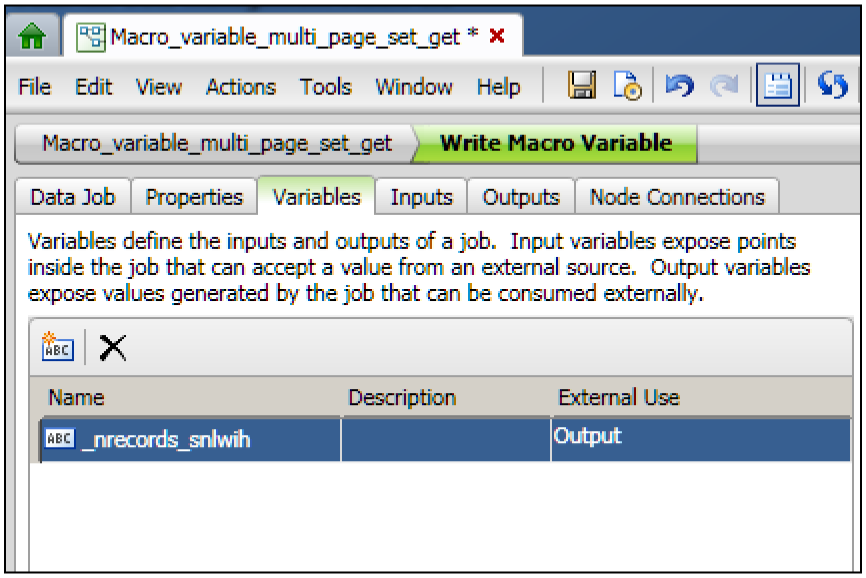
4. Click the **ABC** icon. The **New Variable** dialog appears.

5. Enter an appropriate name for the variable. In the current example, you would specify the output variable **_nrecords_snlwih** to match the macro variable that was set with the setvar() function in the original Architect job.

6. Click OK to save the new variable.

7. The new variable is an Input variable by default, but we want this variable to an Output variable. Click the value in the **External Use** column (Input, in this case).

8. User the selector in this column to select **Output**. The **Variables** tab would look similar to the following display.

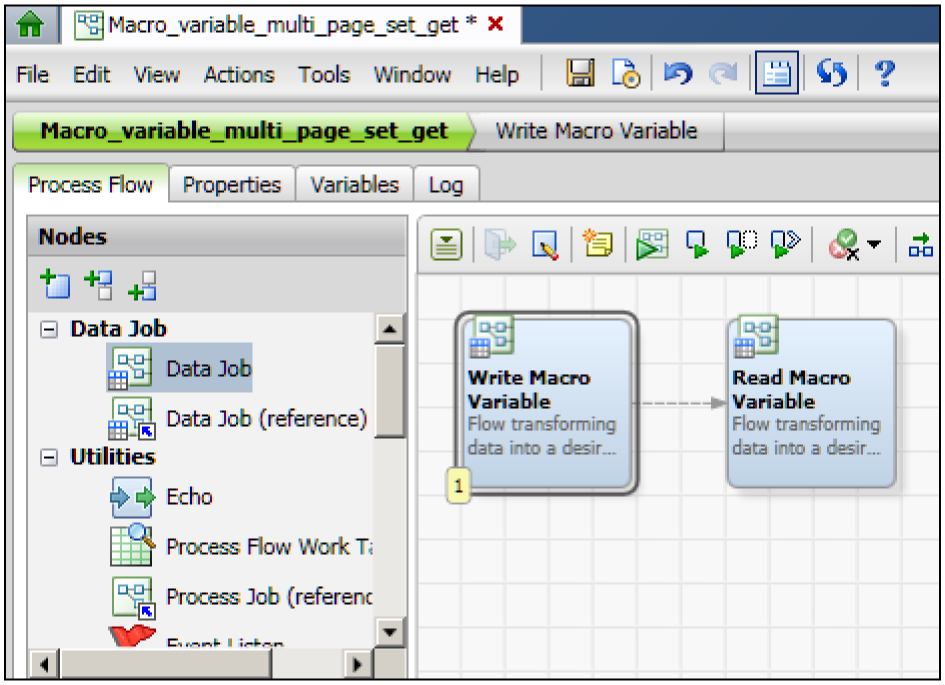

Macro_variable_multi_page_set_get *
File Edit View Actions Tools Window Help
Macro_variable_multi_page_set_get
Write Macro Variable
Data Job
Properties
Variables
Inputs
Outputs
Node Connections

Variables define the inputs and outputs of a job. Input variables expose points inside the job that can accept a value from an external source. Output variables expose values generated by the job that can be consumed externally.

Name	Description	External Use
_nrecords_snlwih		Output

9. Save your changes. You can click the disk icon in the tool bar or select **File** > **Save Process Job** from the main menu.

10. Click the first tab in the flow navigation bar in the process editor dialog to go back to the main flow for the process job. This is the tab with the name of the job (such as **Macro_variable_multi_page_set_get**), as shown in the next display.

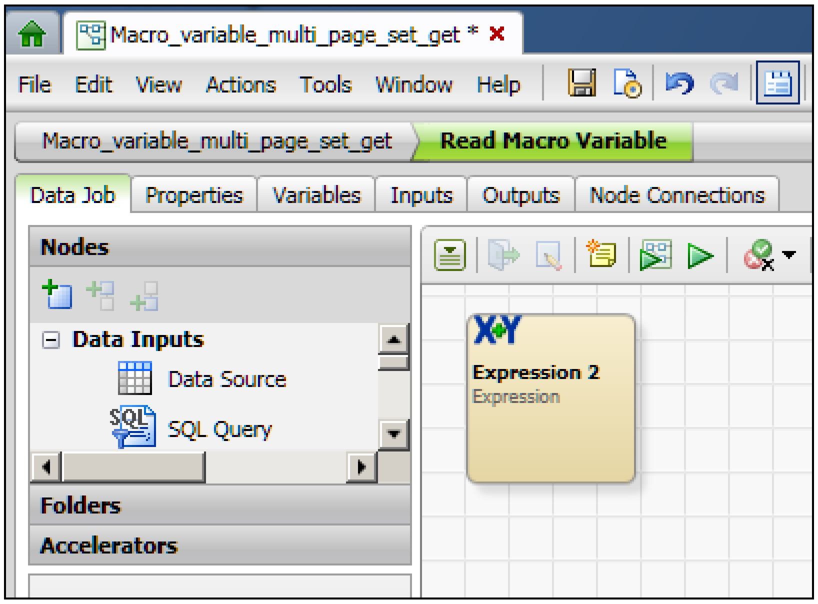


The goal of this step was to display a view where you can access the next node in the process job, so that you can update it.

Specify an Input Variable in a Process Job

The next task is to add an input variable to the second node (such as **Read Macro Variable**) and bind the input variable to the output variable of the first node. For the current example, you could perform the following steps.

1. Right-click the **Read Marco Variable** node and select **Edit**. The data flow for this node opens, as shown in the next display.


Macro_variable_multi_page_set_get *
File Edit View Actions Tools Window Help
Macro_variable_multi_page_set_get
Read Macro Variable
Data Job Properties Variables Inputs Outputs Node Connections
Nodes
Data Inputs
Data Source
SQL Query
Folders
Accelerators
Expression 2
Expression

2. Click on the **Variables** tab, one of the tabs above the Nodes tree. The **Variables** tab displays.

3. Put your cursor on the **ABC** icon at upper left of the **Variables** tab. The hover text should say: **Insert New Variable**.

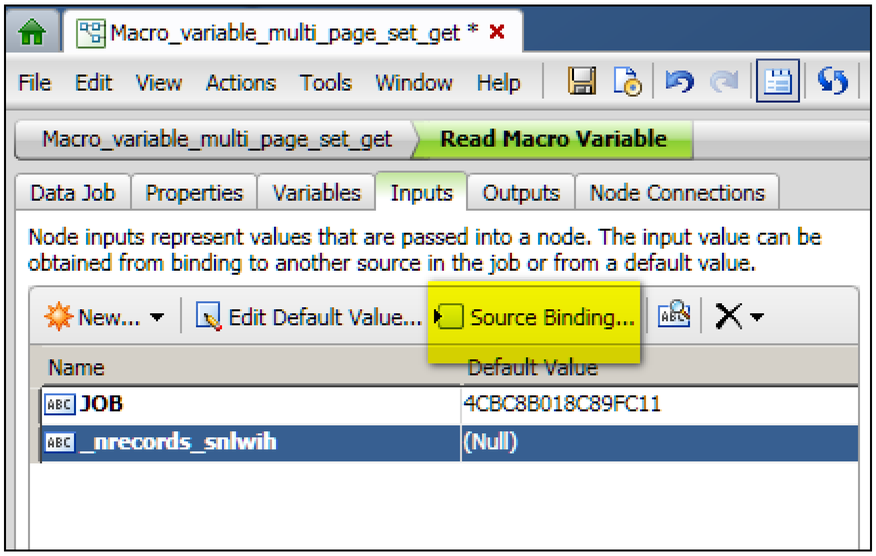
4. Click the **ABC** icon. The **New Variable** dialog appears.

5. Enter an appropriate name for the variable. In the current example, you could specify the input variable **_nrecords_snlwih** to match the output variable that you just specified for the **Write Macro Variable** node.

6. Click **OK** to save the new variable. It is an Input variable by default, which is appropriate. The next step is to bind this input variable to the output variable of the first node.

7. Click on the **Inputs** tab above the **Nodes** tree.

8. Select the input variable that you just created. The **Inputs** tab would look similar to the next display.

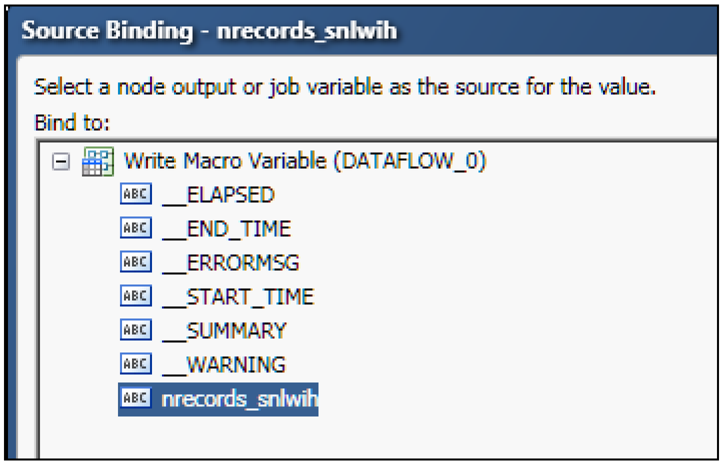

Macro_variable_multi_page_set_get *
File Edit View Actions Tools Window Help
Macro_variable_multi_page_set_get
Read Macro Variable
Data Job Properties Variables Inputs Outputs Node Connections

Node inputs represent values that are passed into a node. The input value can be obtained from binding to another source in the job or from a default value.

New...
Edit Default Value...
Source Binding...

Name	Default Value
JOB	4CBC8B018C89FC11
_nrecords_snlwih	(Null)

9. Click on the **Source Binding** icon. It is above the **Default Value** column for the selected variable. The **Source Binding** dialog appears, as shown in the next display.

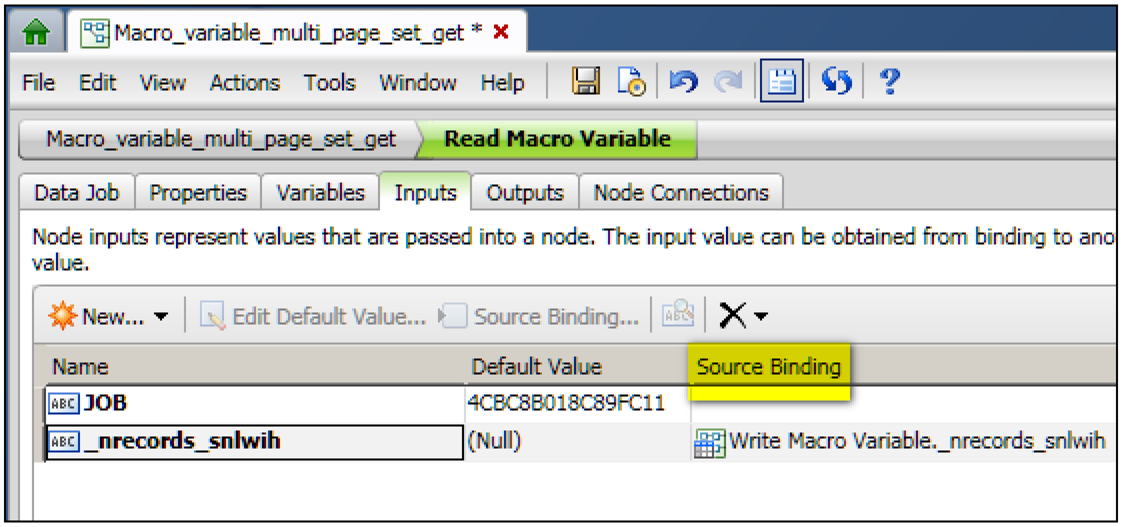


Select a node output or job variable as the source for the value.

Bind to:

10. Select the output variable that should populate the input variable that you just defined. For the current example, this would be the **_nrecords_snlwih** variable for the **Write Macro Variable** node.

11. Click **OK** to save the selected binding. The binding appears in the **Source Binding** column for the variable, as shown in the next display.



Node inputs represent values that are passed into a node. The input value can be obtained from binding to ano value.

Name	Default Value	Source Binding
JOB	4CBC8B018C89FC11	
_nrecords_snlwih	(Null)	Write Macro Variable._nrecords_snlwih

12. Save your changes. You can click the disk icon in the tool bar or select **File** > **Save Process Job**.

    At this point, you have created input and output variables that perform the same function as the setvar() and getvar() functions did in the Architect job. The next task is to run the job and verify the output.

13. Click the first tab in the flow navigation bar in the process editor dialog to go back to the main flow for the process job. This is the tab with the name of the job (such as **Macro_variable_multi_page_set_get**).

14. Run the entire process job. You can click the **Run Process Job** icon in the tool bar, or select **Actions** > **Run Process Job** from the main menu.

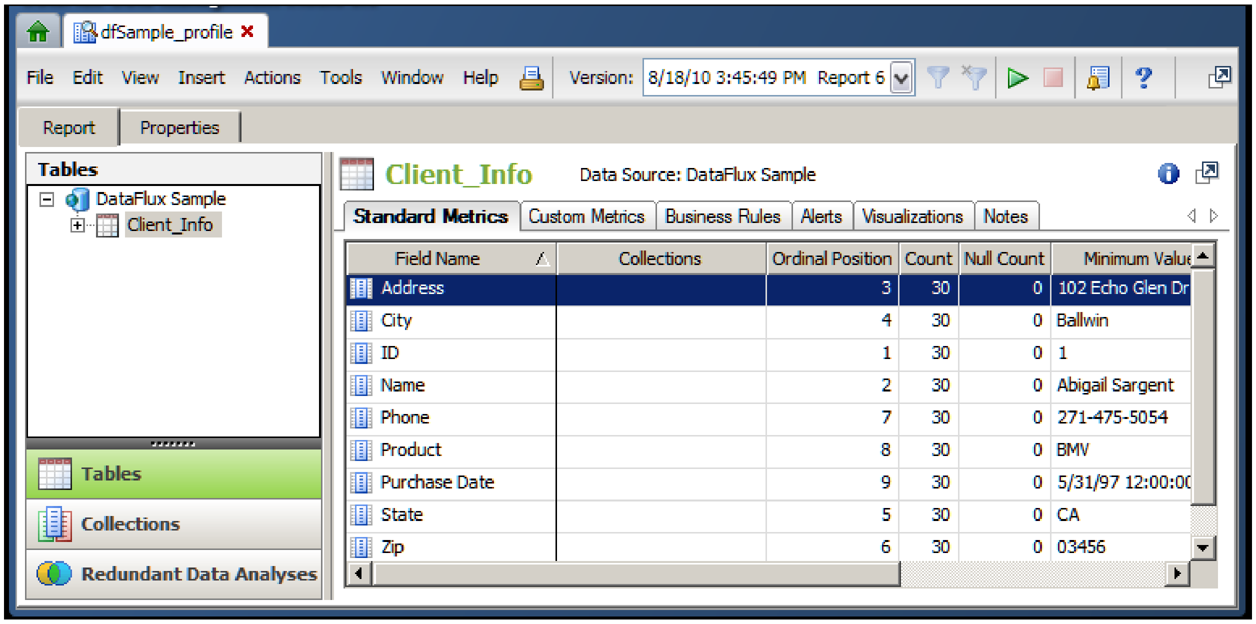
15. Verify that the output is what you expect. For the current example, the output would be something like: `number of records processed: 4`.

Profile Jobs

Changes to Profile Jobs

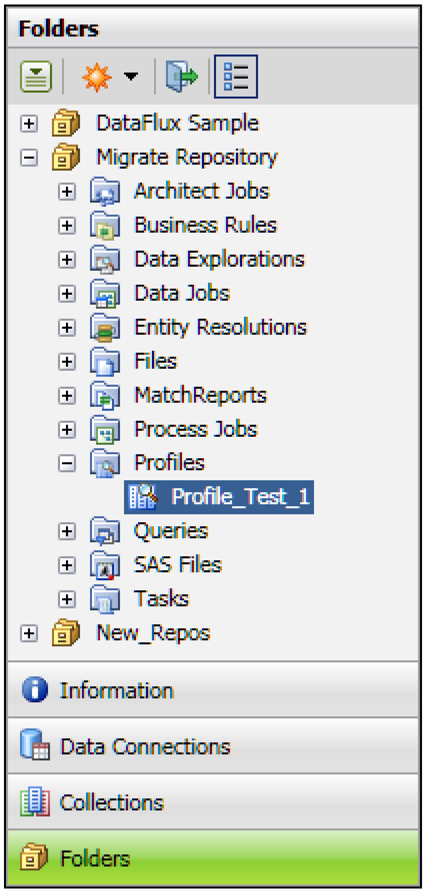
The Meaning of "Profile" Has Changed

In dfPower Studio, a profile job and its report are separate objects. In Data Management Studio, a profile job and its report have been merged into a single object, and this object is simply called a *profile*. The profile dialog in Data Management Studio has separate tabs for the base properties of the profile and the properties of its report, as shown in the next display.

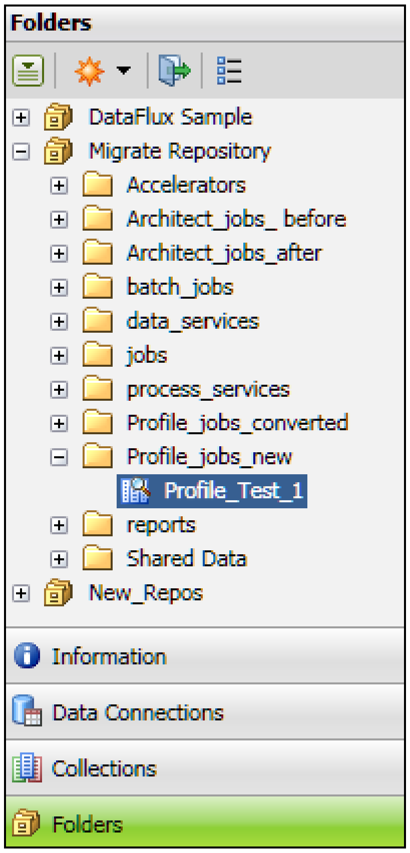


Field Name	Collections	Ordinal Position	Count	Null Count	Minimum Value
Address		3	30	0	102 Echo Glen Dr
City		4	30	0	Ballwin
ID		1	30	0	1
Name		2	30	0	Abigail Sargent
Phone		7	30	0	271-475-5054
Product		8	30	0	BMV
Purchase Date		9	30	0	5/31/97 12:00:00
State		5	30	0	CA
Zip		6	30	0	03456

Profiles can be viewed from the **Profiles** folder in the **Folders** tree, when the **Group Items by Type** control is selected at the top of the tree, as shown in the next display.



Profiles can also be viewed from user-defined folders in the **Folders** tree, when the **Group Items by Type** control is de-selected, as shown in the next display.



For more information about user-defined folders for profiles, see Overview. For more information about using profiles, see the "Profiles" chapter in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

New Storage Constraints

As explained in New Repository Architecture, Data Management Platform puts new constraints on where profiles can be stored. In Data Management Platform:

- Each Data Management Platform repository can have at most one data storage area and one file storage area.
- All related items must be stored in the same Data Management Platform repository. This means, for example, that a profile that is stored in a repository can reference only those business rules, tasks, or similar items that are stored in the same repository.

Differences in metadata architecture might require you to make some manual adjustments when you migrate dfPower Studio repositories and Management Resources to Data Management Studio. For more information, see Plan the Migration of Each Repository and Management Resource.

Other Changes

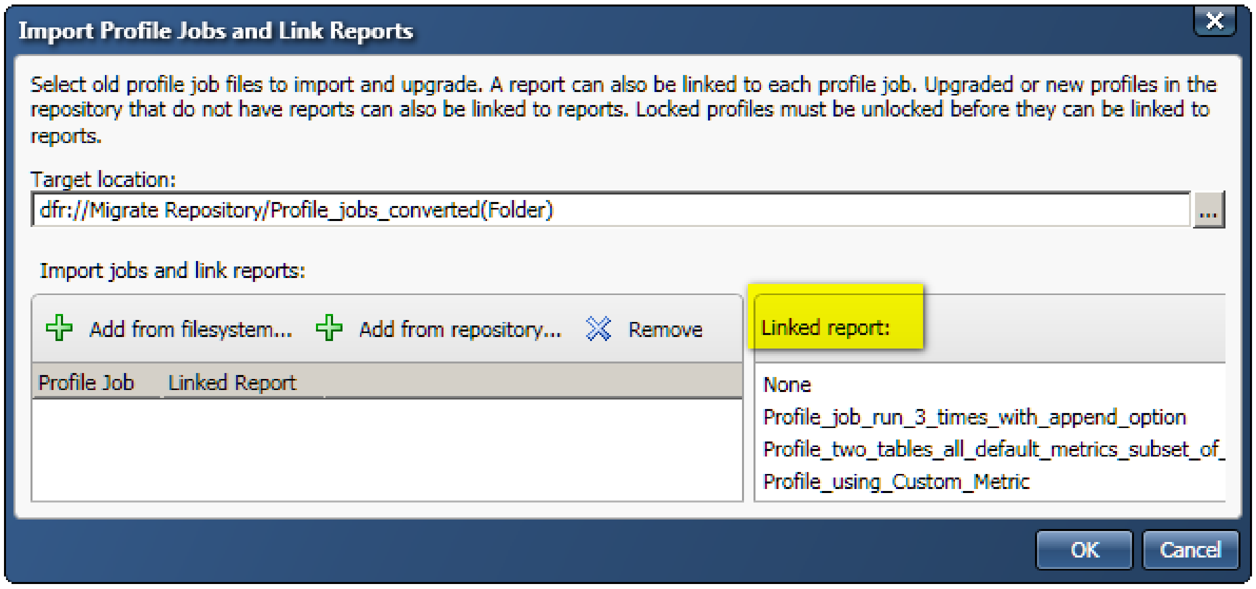
You cannot convert dfPower Studio profile reports that are stored as files (.pfo files). You must either recreate the report for the relevant profile in Data Management Studio or re-run the profile job in dfPower Studio and store the report in a dfPower Studio repository. Then you can use the **Import Profile Jobs and Link Reports** wizard in Data Management Studio to link a dfPower Studio profile job with the related report and combine the two in a Data Management Studio profile. For more information, see the next section.

The way that Redundant Data Analysis is calculated has changed. In dfPower Studio, you set up a Redundant Data Analysis profile for a primary field and add secondary fields that you want to compare to the primary field. The secondary fields were not compared to each other. In Data Management Studio, you set up Redundant Data Analysis profile and all fields that are added to the profile are compared to each other. Accordingly, after a Redundant Data Analysis profile job is converted, you must re-run the profile in Data Management Studio to recreate the report. For more information, see the redundant analysis topic in the "Profiles" chapter of the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Migrating Profile Jobs

Overview

You will use a wizard to copy dfPower Studio profile jobs and convert the copies to Data Management Studio profiles. If the current repository was migrated from dfPower Studio, and profile reports were saved to the dfPower Studio repository before migration, then these reports will be listed in the **Linked report** section of the wizard, as shown in the next display.



Select old profile job files to import and upgrade. A report can also be linked to each profile job. Upgraded or new profiles in the repository that do not have reports can also be linked to reports. Locked profiles must be unlocked before they can be linked to reports.

Target location:
dfr://Migrate Repository/Profile_jobs_converted(Folder)

Import jobs and link reports:

Profile Job	Linked Report

None
Profile_job_run_3_times_with_append_option
Profile_two_tables_all_default_metrics_subset_of_
Profile_using_Custom_Metric

If you do not see any reports in the **Linked report** section of the wizard, verify that you are connected to the correct repository. If you are connected to the correct repository, but you do not see any reports, then you will not be able to include the dfPower Studio reports in the conversion. You will have to re-create the reports as described in the "Profiles" chapter in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Data Management Studio profiles are metadata objects. They are not stored as files on the file system. For convenience, however, references to these profiles are saved to a user-defined folder in the **Folders** tree, so you can view profiles from there. Accordingly, you should identify a user-defined folder where you will save the dfPower Studio profile jobs after they have been converted.

Perform the following steps to create a new user-defined folder, if needed.

1. Click on the **Folders** riser bar in Data Management Studio.
2. If the **Folders** tree is not in user-defined folders mode, deselect the **Group Items by Type** control at the top of the tree.
3. Select and expand the migrated repository in the **Folders** tree.
4. Right-click the repository icon and select **New** > **Folder**. A new folder is added to the tree.
5. Give the new folder a convenient name. For example, you might create a folder named **Profile_jobs_converted**.

Convert dfPower Studio Profile Jobs to Data Management Studio Profiles

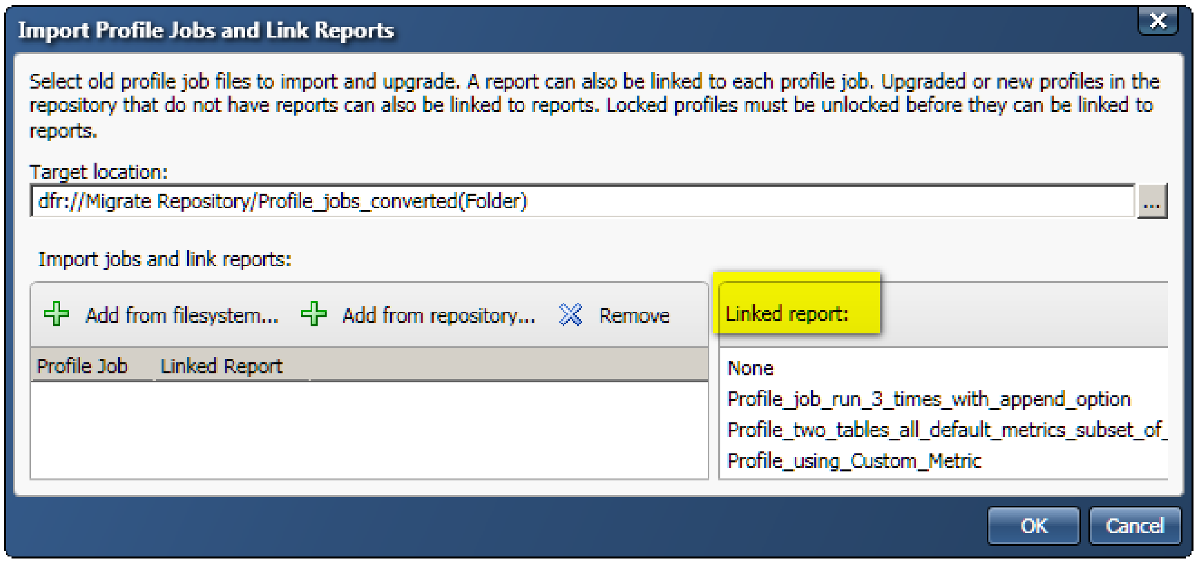
This topic is appropriate when you want to convert dfPower Studio profile jobs to Data Management Studio profiles. Assume that you have identified a user-defined folder where you will save the dfPower Studio profile jobs after they have been converted, as described in Overview. If you want to combine dfPower Studio profile jobs with their reports during the conversion, the following conditions must be met:

- The current repository must have been migrated from dfPower Studio.
- Profile reports must have been saved to the dfPower Studio repository before the repository was migrated.
- You must know the physical path to the profile jobs that correspond to the reports.

This task is easier if you start by locating the user-defined folder where you will save the dfPower Studio profile jobs after they have been converted. Perform the following steps:

1. Click on the **Folders** riser bar in Data Management Studio.
2. If the **Folders** tree is not in user-defined folders mode, deselect the **Group Items by Type** control at the top of the tree.
3. Select and expand the migrated repository in the **Folders** tree.
4. Select the user-defined folder where you will save the dfPower Studio profile jobs after they have been converted. For example, you could select a folder named **Profile_jobs_converted**.

5. Select **Tools** > **Migrate Profile Jobs** > **Import Profile Jobs and Link Reports** from the main menu. A wizard displays.



Select old profile job files to import and upgrade. A report can also be linked to each profile job. Upgraded or new profiles in the repository that do not have reports can also be linked to reports. Locked profiles must be unlocked before they can be linked to reports.

Target location:
dfr://Migrate Repository/Profile_jobs_converted(Folder)

Import jobs and link reports:

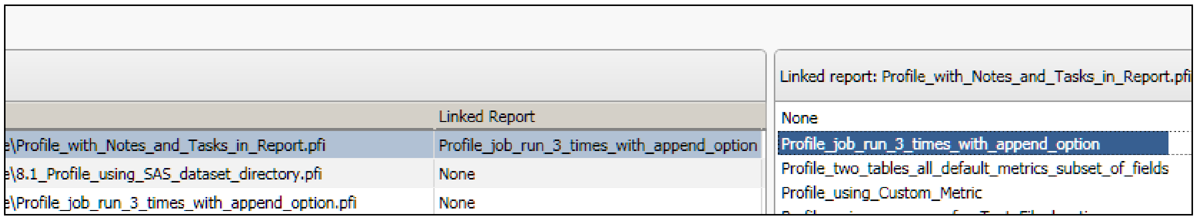
Profile Job	Linked Report

Linked report:

None
Profile_job_run_3_times_with_append_option
Profile_two_tables_all_default_metrics_subset_of_
Profile_using_Custom_Metric

If the current repository was migrated from dfPower Studio, and profile reports were saved to that repository, then these reports will be listed in the **Linked report** section of the wizard. . The wizard does not display reports that have already been linked to migrated profile jobs using the wizard. If you have made a mistake during linking of the reports and only realized after saving the modifications, then you can select **Tools** > **Migrate Profile Jobs** > **Unlink Profile Reports**, to undo these mistakes.

6. Click on the **Add from file system** option. A file selection dialog displays.
7. Navigate to the folder where the profile jobs that you want to convert are stored.
8. Select one or more of these jobs, then click **Open**. The jobs will populate the **Profile Job** section of the wizard.
9. Expand the wizard so that the **Linked Report** column for the profile jobs is clearly visible on the left side of the wizard.
10. To link a report to a profile job, select the profile job on the left, then select the corresponding report on the right. The linked report should appear in the **Linked Report** column for the profile job, as shown in the next display.



	Linked Report
e\Profile_with_Notes_and_Tasks_in_Report.pfi	Profile_job_run_3_times_with_append_option
e\8.1_Profile_using_SAS_dataset_directory.pfi	None
e\Profile_job_run_3_times_with_append_option.pfi	None

Linked report: Profile_with_Notes_and_Tasks_in_Report.pfi

None
Profile_job_run_3_times_with_append_option
Profile_two_tables_all_default_metrics_subset_of_fields
Profile_using_Custom_Metric

11. Repeat Step 10 as needed.
12. When finished, click Ok to convert the profile jobs that are listed in the wizard. The jobs will be saved to the user-defined folder that you selected in Step 4.

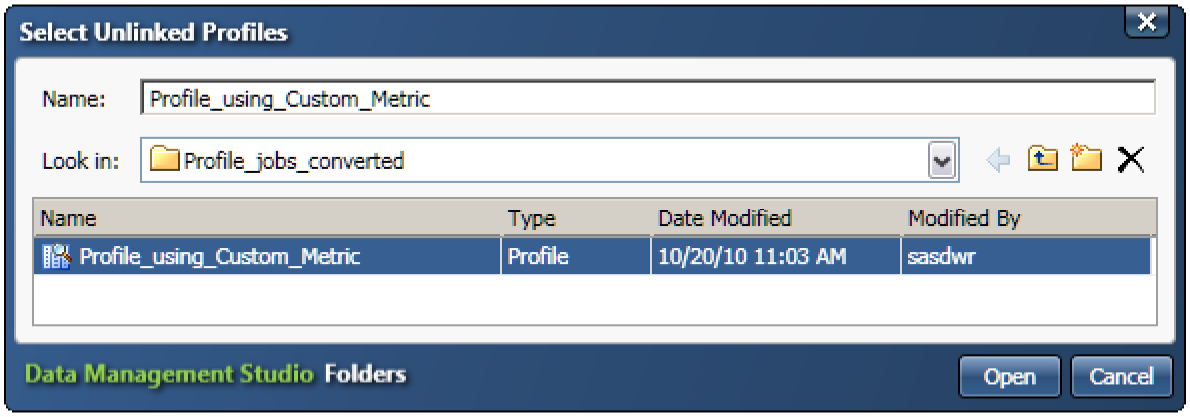
Associate Data Management Studio Profiles with dfPower Studio Reports

This topic is appropriate when you have already converted dfPower Studio profile jobs to Data Management Studio profiles, but when the profile jobs were converted, the appropriate reports were not associated with the profile jobs. You want to associate them now. Assume that the following conditions have been met:

- The current repository was migrated from dfPower Studio.
- Profile reports must have been saved to the dfPower Studio repository before the repository was migrated.
- You know which Data Management Studio profiles correspond to the dfPower Studio reports .

Perform the following steps:

1. Click on the **Folders** riser bar in Data Management Studio.
2. If the **Folders** tree is not in user-defined folders mode, deselect the **Group Items by Type** control at the top of the tree.
3. Select and expand the migrated repository in the **Folders** tree.
4. Select the user-defined folder where you have save the converted dfPower Studio profile jobs (now Data Management Studio profiles).
5. Select **Tools** > **Migrate Profile Jobs** > **Import Profile Jobs and Link Reports** from the main menu. A wizard displays.
6. Click on the **Add from repository** option. The **Select Unlinked Profiles** dialog displays.



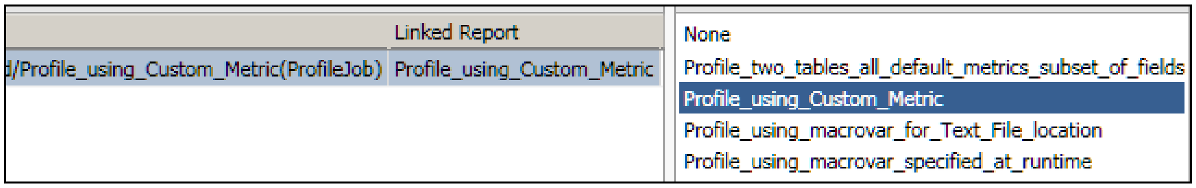
Name: Profile_using_Custom_Metric

Look in: Profile_jobs_converted

Name	Type	Date Modified	Modified By
Profile_using_Custom_Metric	Profile	10/20/10 11:03 AM	sasdwr

7. Select one or more of these profiles, then click **Open**. The selected jobs will populate the **Profile Job** section of the wizard.
8. Expand the wizard so that the **Linked Report** column for the profiles is clearly visible on the left side of the wizard.

9. To link a report to a profile, select the profile on the left, then select the corresponding report on the right. The linked report should appear in the **Linked Report** column for the profile, as shown in the next display.



	Linked Report
d/Profile_using_Custom_Metric(ProfileJob)	Profile_using_Custom_Metric

None
Profile_two_tables_all_default_metrics_subset_of_fields
Profile_using_Custom_Metric
Profile_using_macrovar_for_Text_File_location
Profile_using_macrovar_specified_at_runtime

10. Repeat Step 9 as needed.

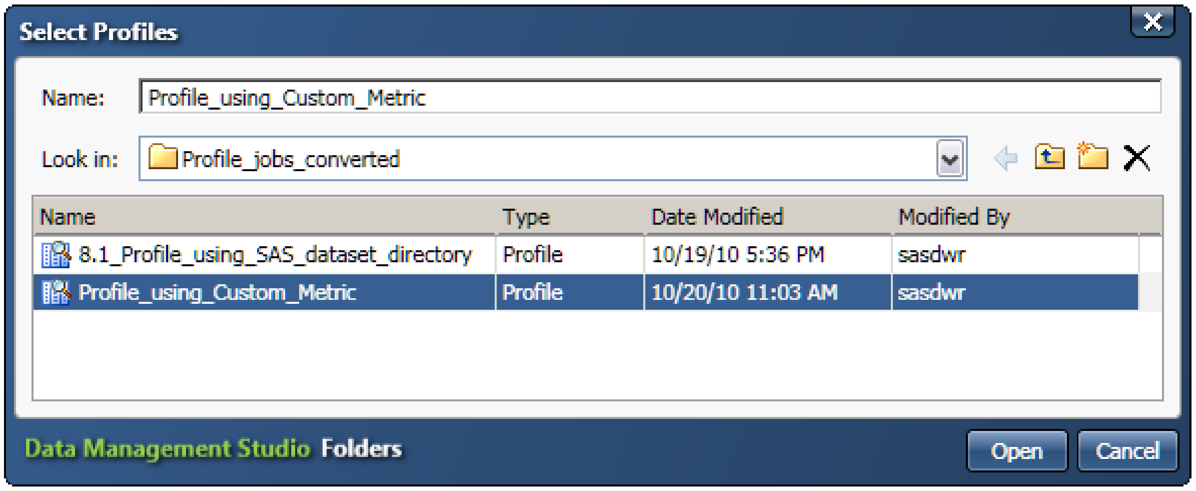
11. When finished, click Ok to link the profile reports and save the profile definitions. The profiles will be saved to the user-defined folder that you selected in Step 4.

Unlink a dfPower Studio Report from a Data Management Studio Profile

This topic is appropriate when you have already converted dfPower Studio profile jobs to Data Management Studio profiles, and you linked a report to the profile job in the process, but the link is incorrect. You want to remove the dfPower Studio report that is incorrectly associated with a Data Management Studio profile.

1. Select **Tools** > **Migrate Profile Jobs** > **Unlink Link Reports** from the main menu. A dialog displays.

2. Click the **Add Profile** option. The **Select Profile** dialog displays.



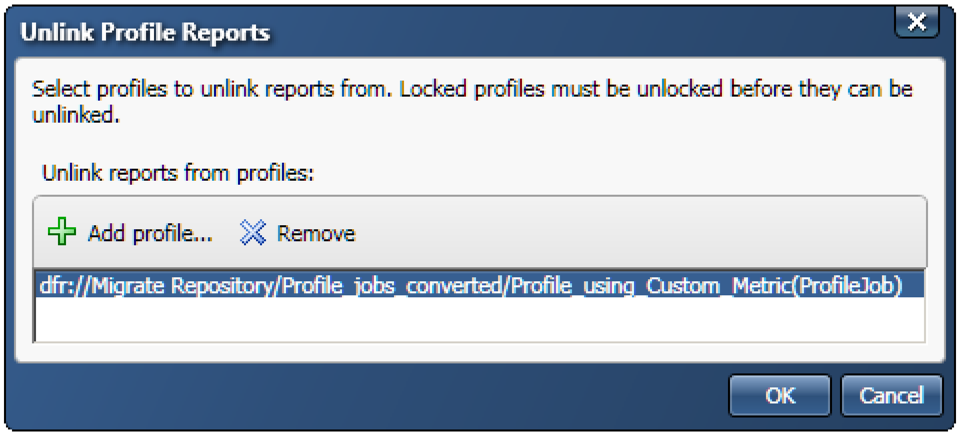
Select Profiles

Name: Profile_using_Custom_Metric

Look in: Profile_jobs_converted

Name	Type	Date Modified	Modified By
8.1_Profile_using_SAS_dataset_directory	Profile	10/19/10 5:36 PM	sasdwr
Profile_using_Custom_Metric	Profile	10/20/10 11:03 AM	sasdwr

3. Select one or more profiles and click **Open**. The selected profiles will be listed in the **Unlink Profile Reports** dialog.


Unlink Profile Reports

Select profiles to unlink reports from. Locked profiles must be unlocked before they can be unlinked.

Unlink reports from profiles:

dfr://Migrate Repository/Profile_jobs_converted/Profile_using_Custom_Metric(ProfileJob)

4. Select one or more profiles from which you will remove the linked report and click OK.
5. The reports are unlinked from the profiles.

Post-Migration Tasks for Profile Jobs

Profile Jobs That Contain Macro Variables

Contact Technical Support if you want to use macro variables to pass values to Data Management Studio profiles.

Macro Variables

Changes to Macros

New macro configuration files. The files and folders where macro variable definitions can be stored have changed. The new macro configuration files are described in the "Macro Variables" chapter in the *DataFlux Data Management Studio User's Guide*. If you move your dfPower Studio macro variable definitions into one or more Data Management Studio configuration files, and you use these macro variables in the properties of a node, or in expressions using %% notation, or in any instance of the getvar() function, then your dfPower Studio macro variables will often work in Data Management Studio without further modification.

Macro syntax changes. The syntax for macro variable definitions has changed. This change could affect any item that uses a macro variable, such as an Architect job, profile, business rule, or task. In Data Management Studio, all characters after the equal sign (=) and before the new line character become part of the definition. This means, for example, that you should not use spaces, quotation marks, or other characters after the equal sign unless you want them to be part of the text that is retrieved when the macro variable is called. For more information, see the "Macro Variables" chapter in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Macros can have other impacts as well. For more information, see the topics related to macros in the following sections of this guide:

- Architect Jobs That Use Macro Variables
- Architect Jobs That Pass Macro Variable Values Dynamically Between Pages in the Job
- Rules and Related Items That Use Macro Variables
- Profile Jobs That Contain Macro Variables

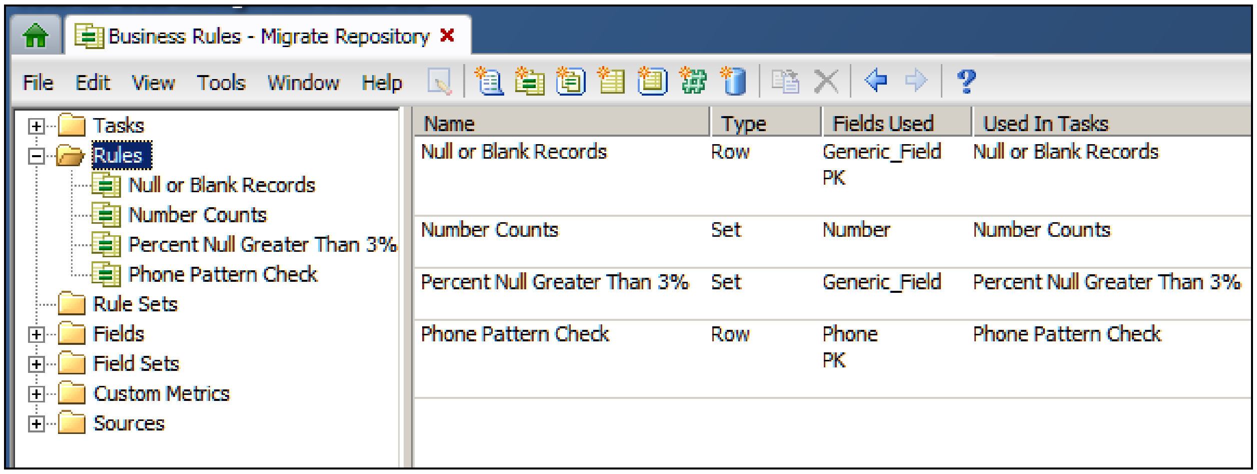
Business Rules, Tasks, and Related Items

Changes to Rules and Related Items

Main Interfaces for Rules

There have not been many changes to most interfaces for business rules, tasks, and related objects. Like dfPower Studio, Data Management Studio has a Business Rule Manager dialog. You can use the Business Rule Manager to manage business rules, tasks, and related objects. You can also use this dialog to manage custom metrics (user-defined DataFlux expressions) that can be selected in one or more business rules or profiles.

If you select **Tools** > **Business Rule Manager** > [*repository_name*] from the main menu, the Business Rules Manager displays.



Name	Type	Fields Used	Used In Tasks
Null or Blank Records	Row	Generic_Field PK	Null or Blank Records
Number Counts	Set	Number	Number Counts
Percent Null Greater Than 3%	Set	Generic_Field	Percent Null Greater Than 3%
Phone Pattern Check	Row	Phone PK	Phone Pattern Check

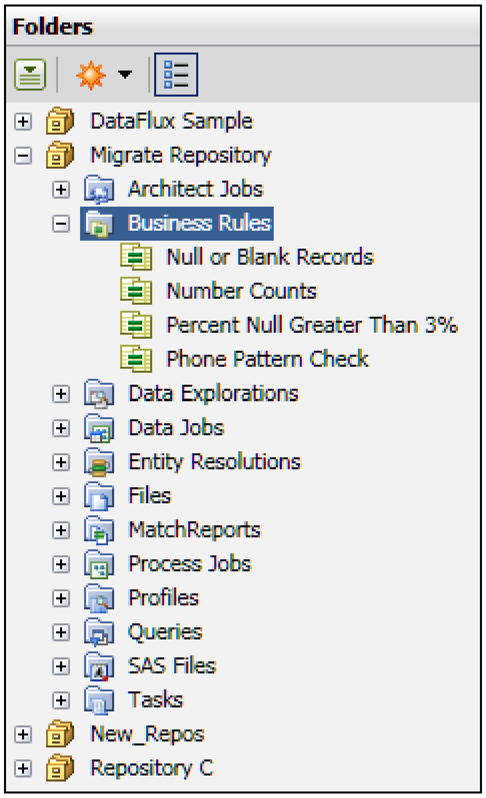
The tree on the left of the Business Rule Manager displays folders for tasks, rules, and other objects that are associated with rules in the current repository (Migrate Repository). The panel on the right displays information about the items that are selected in the tree.

The property dialogs for individual rules and other items have few changes from dfPower Studio.

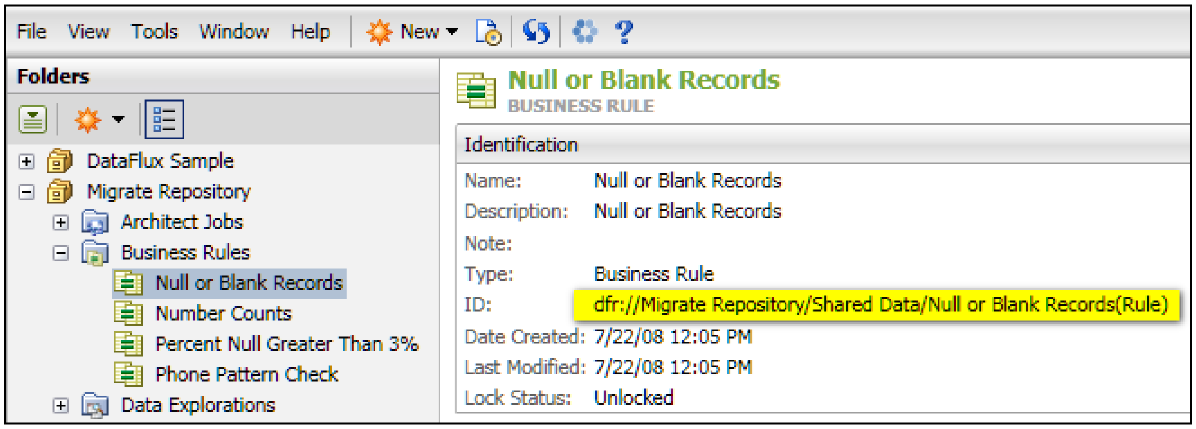
Business rules and other items in the Business Rule Manager are stored in metadata, so they are mainly accessible through the Business Rule Manager. For convenience, however, business rules and tasks can now be viewed in the **Folders** tree, along with jobs, profiles, and other top-level objects.

To display the system-defined folders for business rules and tasks, perform the following steps:

1. Click on the **Folders** riser bar in Data Management Studio.
2. If the **Folders** tree is not in system-defined folders mode, select the **Group Items by Type** control at the top of the tree.
3. Select and expand the appropriate repository in the **Folders** tree, such as a migrated repository. The folders for business rules and tasks will be visible in the tree as shown in the next display.



4. If you select an item in the tree on the left, the path to the user-defined folder for that item is displayed on the right, as shown in the next display.



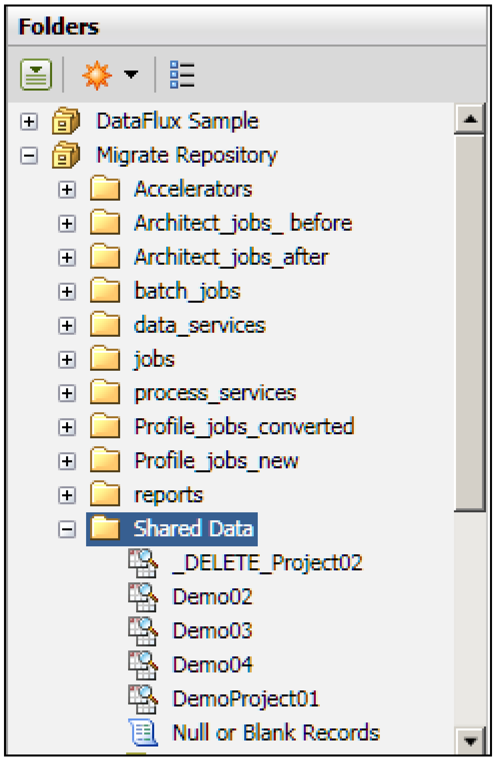
Name:	Null or Blank Records
Description:	Null or Blank Records
Note:	
Type:	Business Rule
ID:	dfr://Migrate Repository/Shared Data/Null or Blank Records(Rule)
Date Created:	7/22/08 12:05 PM
Last Modified:	7/22/08 12:05 PM
Lock Status:	Unlocked

Note that the selected business rule **Null or Blank Records** is stored in the **Shared Data** folder, which is a user-defined folder that is automatically created during migration. This folder is used to store references to items stored in metadata, such as business rules and tasks, so that they can be viewed in the **Folders** tree.

5. To display business rules and tasks in user-defined folders, deselect the **Group Items by Type** control at the top of the tree.

6. Select and expand the appropriate repository, such as a migrated repository.

7. Locate the folder that you identified in Step 4 and open it, as shown in the next display.



You will see a number of references to items stored in metadata, including business rules and tasks.

As in dfPower Studio, you can export and import selected items in the Business Rule Manager. In Data Management Studio, you can also export and import packages of rules and related items from the **Folders** tree. Both of these methods are described in Post-Migration Tasks for Rules and Related Items.

For more information about rules and related items in Data Management Studio, see the "Business Rules and Tasks" chapter in the *DataFlux Data Management Studio User's Guide, v. 2.1.1*.

Migrating Rules and Related Items

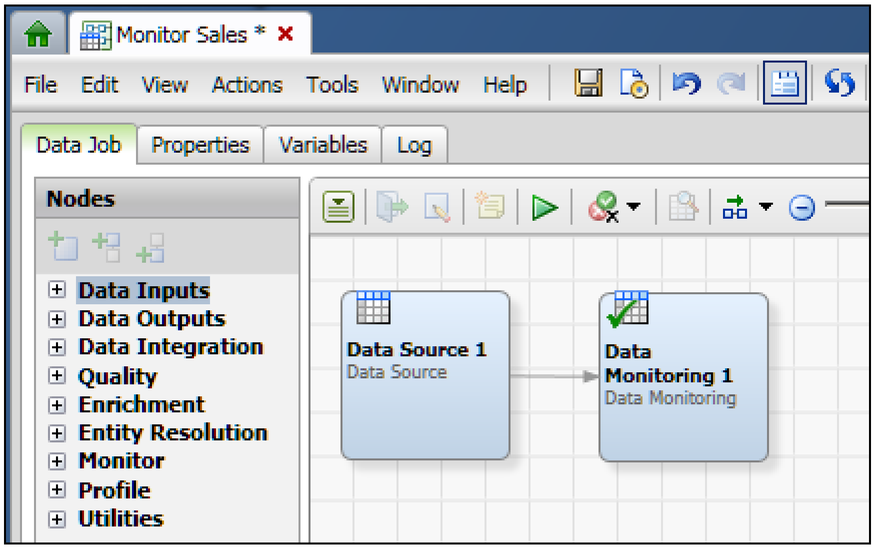
dfPower Studio business rules, tasks, custom metrics, and other items in the Business Rules Manager are migrated only when their repository is migrated. You cannot migrate individual rules and related items from dfPower Studio.

Post-Migration Tasks for Rules and Related Items

Rules and Related Items That Use Macro Variables

When a macro variable was used in a dfPower Studio task, the Data Monitoring node that is referencing this task will probably have to be updated before it will work in Data Management Studio. General updates for macros are described in Macro Variables.

Macro Variables, Rules and Tasks. If an Architect job references a rules or task, and the rule or task dynamically retrieves or sets the value of a macro variable, then the Data Monitoring node that references such a task must be updated. Such a job is shown in the next display.



To pass macro variable values to a rule that is included in the **Data Monitoring** node shown in the previous display, you would right-click the **Data Monitoring** node and select

Advanced Properties. Then you would specify the macro variable in the KEY_VALUES property.

To get macro variable values out of a rule that is included in the **Data Monitoring** node shown in the previous display, you would right-click the **Data Monitoring** node and select **Advanced Properties**. Then you would specify the macro variable in the OUT_KEY_VALUES property.

Moving Rules and Related Items Where They Are Needed

After dfPower Studio rules and related items have been migrated to Data Management Studio, you can export them from one repository and import them in to another. In this way, you can move existing rules and related items to the repository where they are needed. For example, you could use the export and import features to address the scenario described in Migration Scenario: Multiple Repositories, One Management Resource.

Export and Import from the Folders Tree

This topic is appropriate when you want to export and import a package of related rules, tasks, and other items. For example, if you export a rule will this feature, any components that are required by the rule, such as fields or custom metrics, will be included in the package.

Note: There is one scenario where you might not want to use the package tool. If you import a package that includes custom metrics and fields, and the target repository happens to have custom metrics and fields with the same names, then the identically-named metrics and fields in the package will be ignored. The corresponding metrics and fields in the target repository will be retained. This might not be what you want.

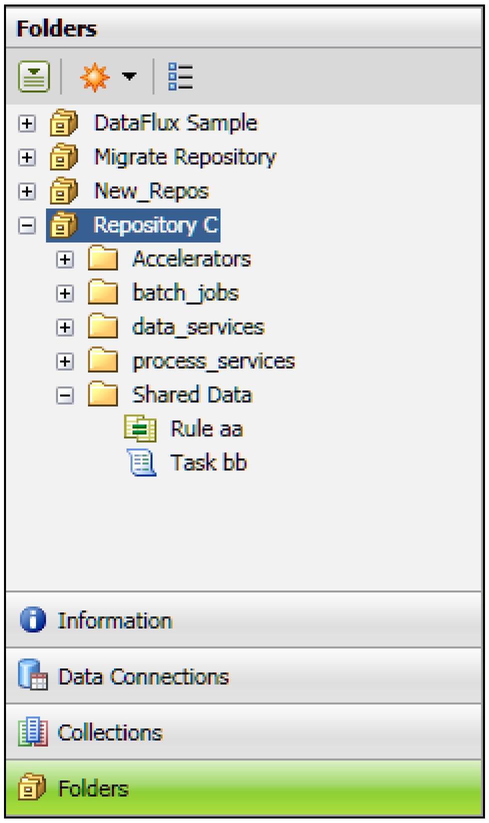
It you want to overwrite identically-named custom metrics and fields in a target repository, use the **Import** feature in the Business Rule Manager to do so, as described in Import Sources, Fields, and Custom Metrics from the Business Rule Manager. After the correct version of the metrics and fields are in the target repository, you can use the package tool, if desired.

Note: The export and import feature in the **Folders** tree does not support rule names with special characters in them. If you get an error in the export log about this issue, you can remove the special characters before export and update all references to the affected rules.

The following steps demonstrate how to export a set of rules and related items as a package. Then the package is imported into another repository. It is assumed that you are connected to the source repository and the target repository. The first task is to identify a set of rules and related items in a source repository that you want to copy into a target repository. For the current example, assume that you want to export **Rule aa** and **Task bb** from Repository C to the Migrate Repository.

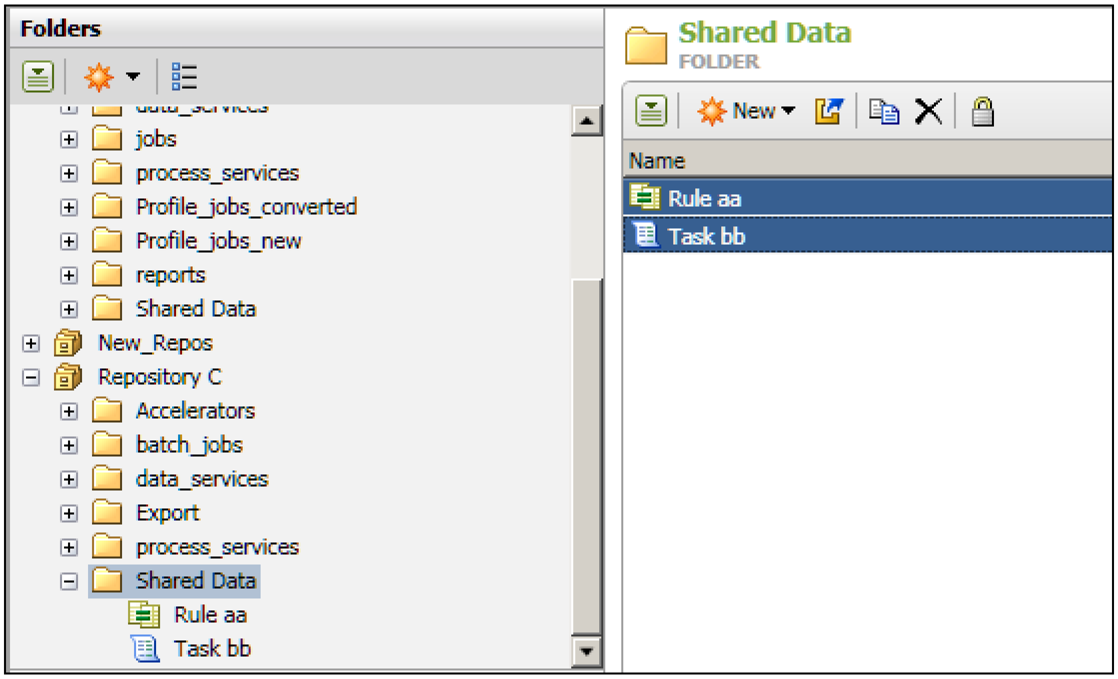
1. Select the source repository in the **Folders** tree, such as Repository C.
2. De-select the **Group Items by Type** control at the top of the tree. Items will grouped into user-defined folders.

3. Expand the user-defined folder where the appropriate rules and tasks are stored, as shown in the next display.

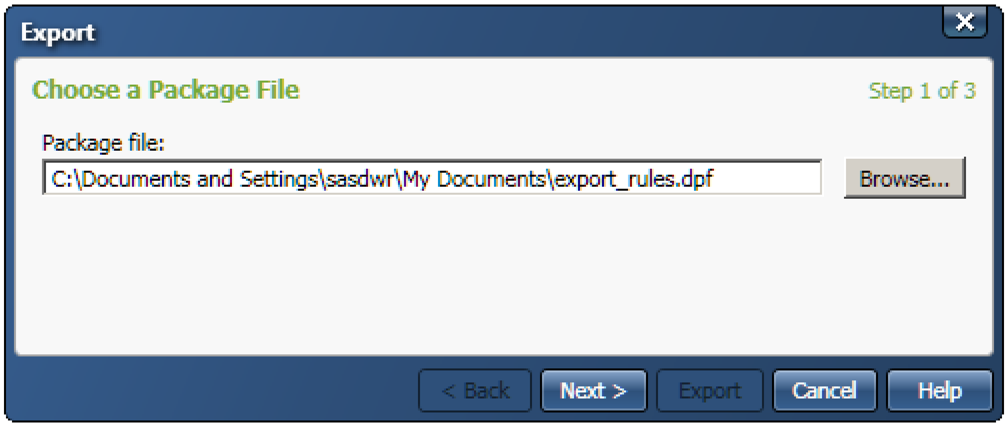


4. Typically, you will want to export the items within a folder, rather than the folder itself and the items in the folder. If that is the case, select the folder in the tree on the left, and the items in the folder will be displayed on the right.

5. Select multiple items on the right, as shown in the next display.



6. Right-click the selected items on the right, then select **Export**. The first page of the export wizard displays.

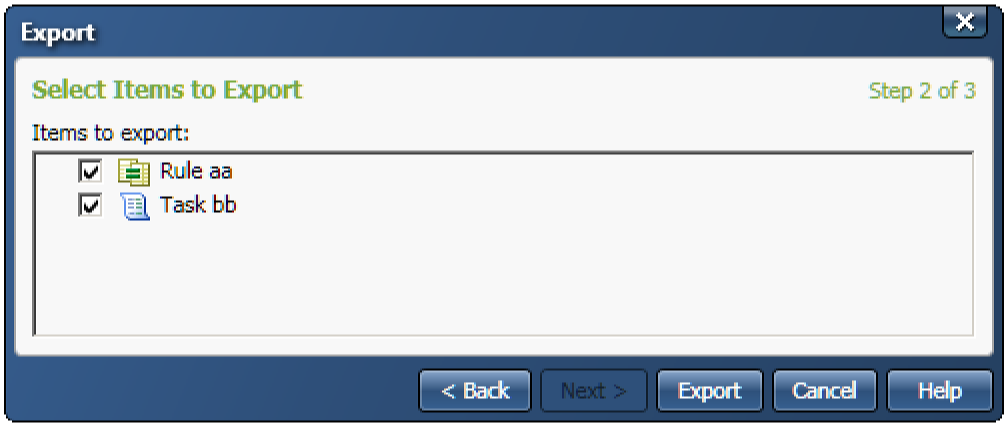


Package file: C:\Documents and Settings\sasdwr\My Documents\export_rules.dpf

The wizard will display the name of the last package that was either imported or exported.

7. If you want to change the name of the export file to match the content of the current export operation, do so. Do not change the extension of the file. For the current example, the file export file name could be: **export_rules.dpf**.

8. When the file name if correct, click **Next**. The second page of the export wizard displays.



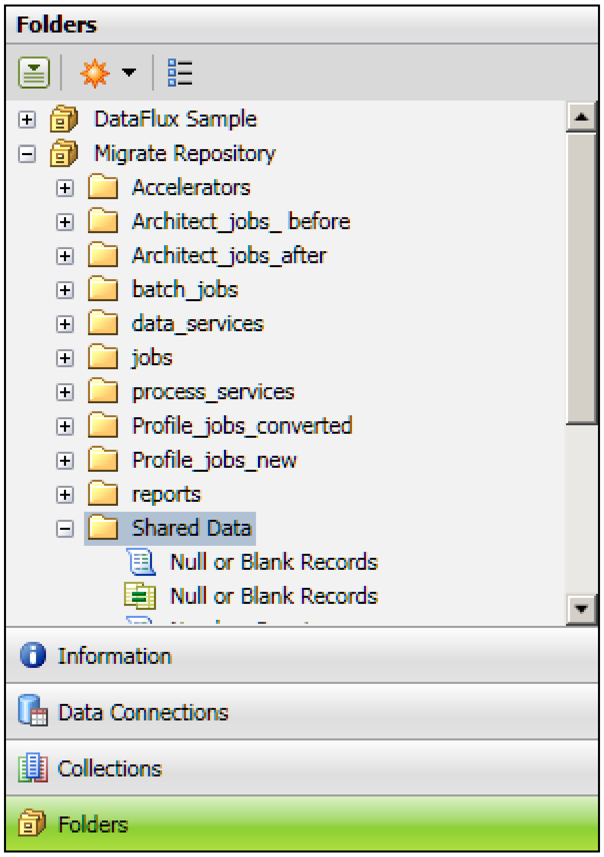
The items that you selected for export are displayed. You can de-select some items if you wish.

9. If the list of items is correct, click **Next**. The items will be exported. A dialog is displayed.

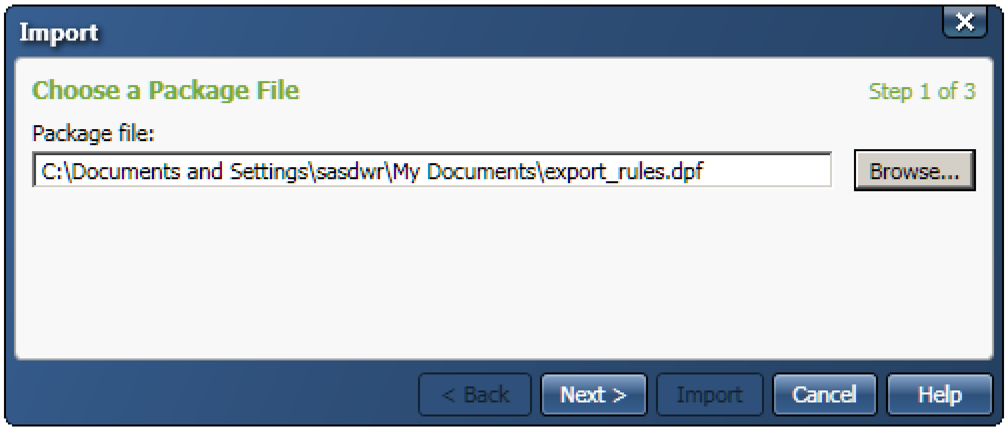
10. You can view the export log or simply click **Close**. The next task is to import the exported items into the target repository.

11. In the **Folders** tree, expand the target repository.

12. Expand the user-defined folder where you import the exported items, as shown in the following display.



13. Right-click the folder where you import the exported items, then select **Import**. The first page of the import wizard displays.

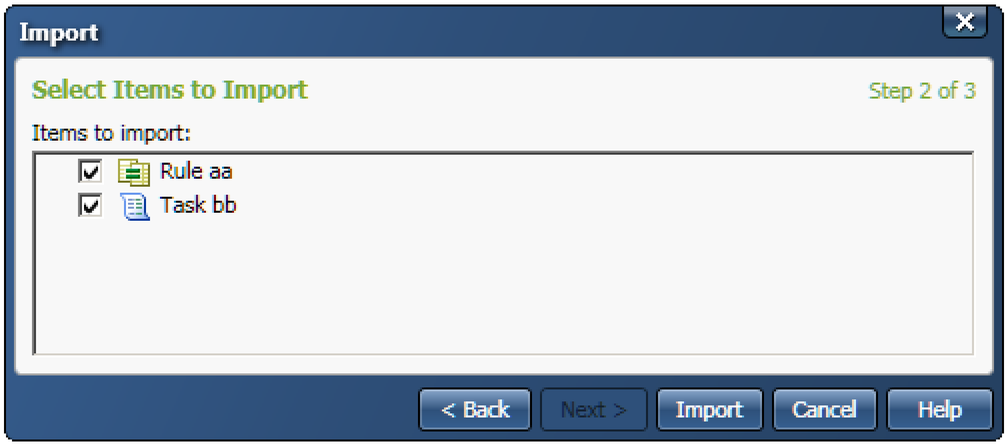


Package file:

C:\Documents and Settings\sasdwr\My Documents\export_rules.dpf

The wizard will display the name of the last package that was either imported or exported.

14. If the file name if correct, click **Next**. The second page of the import wizard displays.



The items in the package are displayed. You can de-select some items if you wish.

15. If the list of items is correct, click **Next**. The items will be imported. A dialog is displayed.

16. You can view the import log or simply click **Close**.

17. Verify that the rules are in the target folder.

18. You might want to open the imported items and verify that any components that are required by the item, such as sources, fields, or custom metrics, have been imported. If not, you can import these items individually in the Business Rule Manager, as described in the next section.

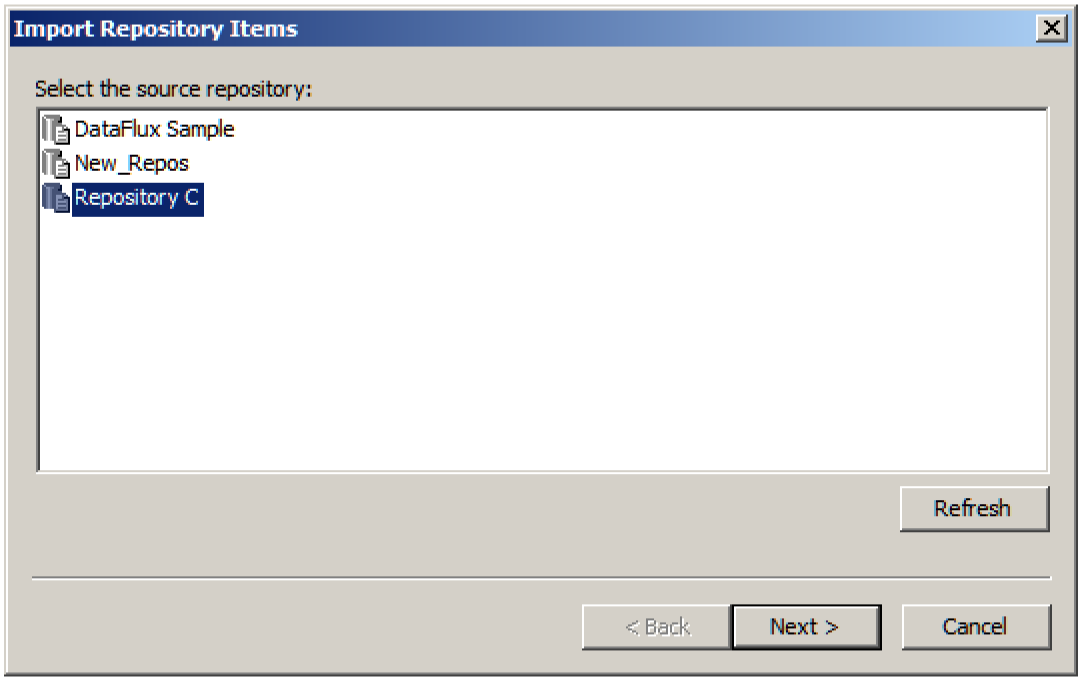
Import Sources, Fields, and Custom Metrics from the Business Rule Manager

This topic is appropriate when you want to import *only* sources, fields, or custom metrics from the Business Rule Manager. It is assumed that you are connected to the source repository and the target repository.

For the current example, assume that you want to import only the custom metric **Approx Value** from Repository C into the Migrate Repository. Perform the following steps.

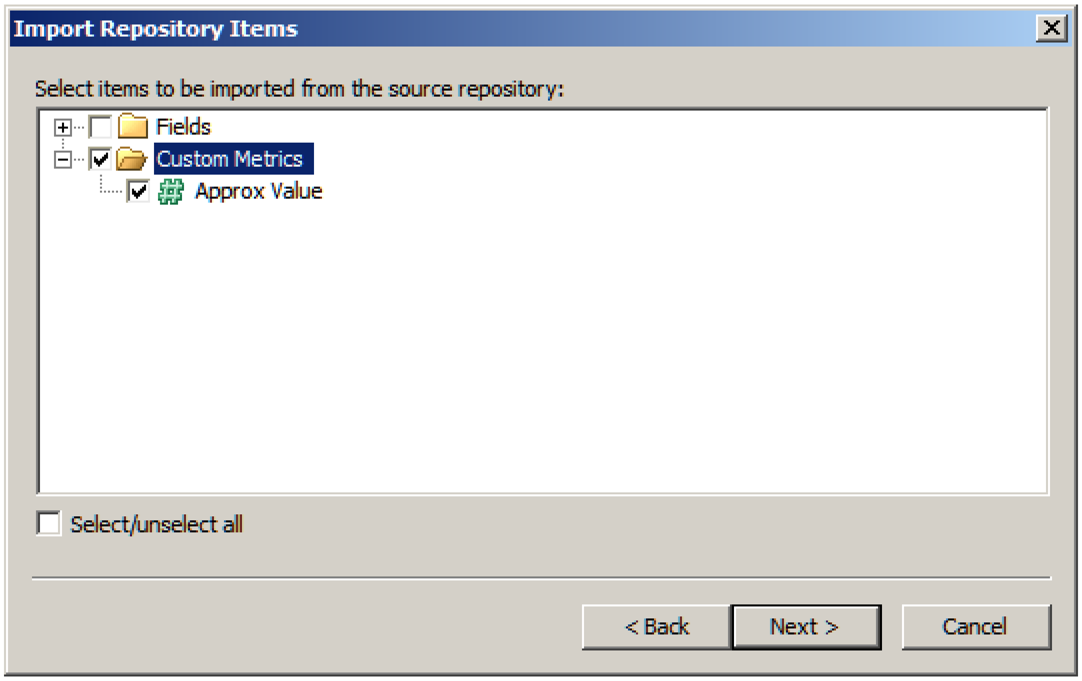
1. Select **Tools** > **Business Rule Manager** > [*target_repository*] from the main menu. The Business Rule Manager displays any rules or related objects for the selected repository, such as Migrate Repository.

2. Select **File** > **Items from Another Repository** from the Business Rule Manager menu. The first page of the import wizard displays.



3. Select the repository that contains the item that you want to import, such as Repository C.

4. Click **Next**. The second page of the import wizard displays.
5. Select the item that you want to import, such as the custom metric **Approx Value**.



6. Click **Next**. The third page of the import wizard displays.
7. Resolve any conflicts (name collisions) between the imported items and existing items in the target repository.
8. When finished click **Finish**.
9. You might want to open the imported items and verify that they are correct.

Glossary

C

case definition

A set of logic used to accurately change the case of an input value, accounting for unique values that need to be cased sensitively, such as abbreviations and business names.

chop table

A proprietary file type used by DataFlux as a lex table to separate characters in a subject value into more usable segments.

D

data type

The semantic nature of a data string. The data type provides a context determining which set of logic is applied to a data string when performing data cleansing operations. Example data types are: Name, Address, and Phone Number.

definition

Another name for an algorithm. The definitions in the Quality Knowledge Base are the data management processes that are available for use in other SAS or DataFlux applications like dfPower Studio or SAS Data Quality Server.

I

identification definition

A set of logic used to identify an input string as a member of a redefined or user-defined value group or category.

L

locale guessing

A process that attempts to identify the country of origin of a particular piece of data based on an address, a country code, or some other field.

M

match code

The end result of passing data through a match definition. A normalized, encrypted string that represents portions of a data string that are considered to be significant with regard to the semantic identity of the data. Two data strings are said to "match" if the same match code is generated for each of them.

match definition

A set of logic used to generate a match code for a data string of a specific data type.

match value

A string representing the value of a single token after match processing.

P

parse

The process of dividing a data string into a set of token values. For example: Mr. Bob Brauer [Mr. = Prefix , Bob = Given Name, Brauer = Family Name]

parse definition

A name for a context-specific parsing algorithm. A parse definition determines the names and contents of the sub-strings that will hold the results of a parse operation.

pattern analysis definition

A regular expression library that forms the basis of a pattern recognition algorithm.

phonetics

An algorithm applied to a data string to reduce it to a value that will match other data strings with similar pronunciations.

Q

QKB

The QKB, or Quality Knowledge Base, is a collection of files and configuration settings that contain all DataFlux data management algorithms. The Quality Knowledge Base is directly editable using dfPower Studio.

Quality Knowledge Base

The collection of files and configuration settings that contain all DataFlux data management algorithms. The Quality Knowledge Base is directly editable using dfPower Studio.

R

regular expression

A mini-language composed of symbols and operators that enables you to express how a computer application should search for a specified pattern in text. A pattern may then be replaced with another pattern, also described using the regular expression language.

S

sensitivity

Regarding matching procedures, sensitivity refers to the relative tightness or looseness of the expected match results. A higher sensitivity indicates you want the values in your match results to be very similar to each other. A lower sensitivity setting indicates that you would like the match results to be "fuzzier" in nature.

standardization definition

A set of logic used to standardize a string.

standardization scheme

A collection of transformation rules that typically apply to one subject area such as company name standardization or province code standardization.

T

token

The output string of a parse process. These are words or atomic groups of words with semantic meaning in a data string. A set of expected tokens is defined for each data type.

V

vocabulary

A proprietary file type used for categorizing data look-ups pertinent to a specific subject area.