Data Migrator for i
User's Guide
Part 1

IBM Db2 Web Query for i

DataMigrator ETL Extension

June 2021

Contents PART 1

1 Introduction

1.1 Product Description

IBM Db2 Web Query for i DataMigrator ETL Extension (DataMigrator for i) is a member of the *IBM Db2 Web Query* product family. It supports building, populating, and maintaining database tables from one or more data sources into target tables in another database. DataMigrator provides an Extract, Transform, and Load (ETL) solution that runs on IBM i. It is useful for IBM i customers interested in data replication, consolidating data for reporting and analytics, or for creating a more efficient reporting environment that does not interfere with an existing production environment. Customers can organize and consolidate data into a single environment that is *optimized* for Query Reporting, BI (Business Intelligence) and Analytics.

DataMigrator is *ordered* as product *5733WQM* from IBM. It is required to have one of the "Editions" as a prerequisite. In other words, DataMigrator must be paired with one of the four Editions – Express, Scheduler, RunTime User, or Standard Edition. Standard Edition is required for accessing non Db2 databases, and either Standard or Scheduler Edition is required for scheduling functions from an IBM i Job Scheduler or CL command). Db2Db2

> **NOTE:** While ordering is done for product 5733WQM, the actual DataMigrator software is *packaged* and distributed with Db2 Web Query for i product 5733WQX. It is enabled using (previously reserved) option 8 (#5108) of 5733WQX. It is highly recommended to install it using IBM's EZ-Install package and is available for a trial period of 70 days.

1.2 Product Overview

DataMigrator is a set of software components that automate the process of Extract, Transform and Load (ETL). DataMigrator provides the capability to:

- Access data, called data sources, from database files and flat (IFS) files.
- Integrate multiple data sources into a set of target tables such as a data warehouse.
- Apply transformation logic to 'clean up' and convert data into desirable types.
- Aggregate data to simplify reporting and decision making.
- Schedule updates to synchronize data sources.

DataMigrator has two main components:

- The *Data Management Console (DMC)* graphical user interface is used to turn data transport and transformation into a data flow through drag and drop operations. The interface lets you visualize how the data will flow and transform from the source to the target. The DMC is also used for scheduling flows, setting up email notifications for flow outcomes, and viewing logs and report statistics. The DMC is a Windows™ application that is available as part of the Developer Workbench option of Db2 Web Query.
- The *DataMigrator Server* stores and runs data flows. It also processes scheduled flows. Because of its integration with Db2 Web Query, the DataMigrator Server runs in the same instance as the Web Query server on the IBM i.

The following figure depicts the overall architecture of DataMigrator for i.

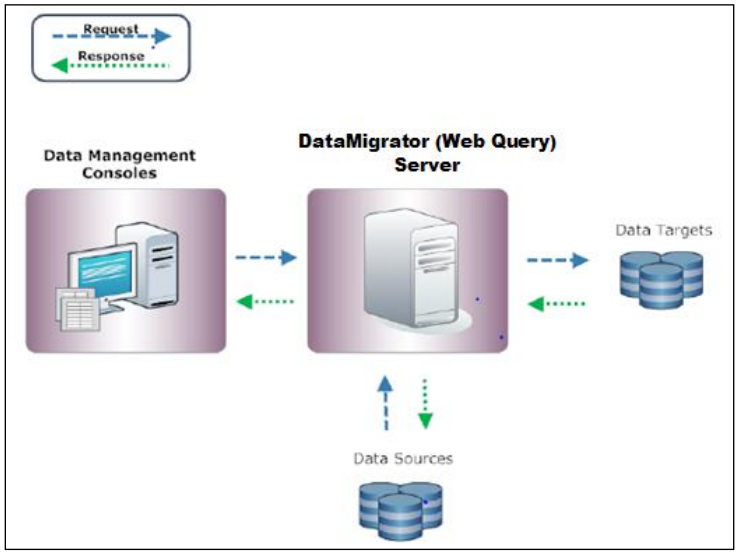


Figure 1 DataMigrator architecture

DataMigrator provides the ability to perform bulk loads and incremental updates of data. This makes it ideal for populating and maintaining a reporting database such as a data mart or data warehouse.

Defining the ETL process is done through the creation of a *Data Flow*. A Data Flow defines where the data comes from, how it should be transformed, and what files the resulting data should be loaded into. It can be run immediately, scheduled to run once, or scheduled to run on a recurring basis. There is no limit on how many data flows can be created. An example Data Flow can be seen in the figure below.

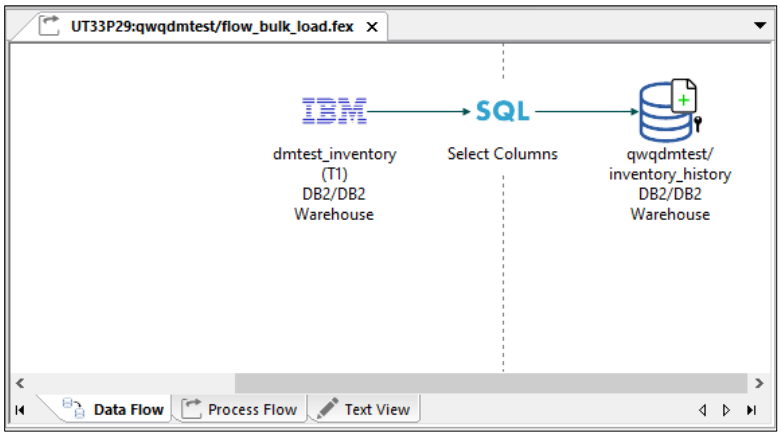


Figure 2 Graphical view of a data flow

Every Data Flow is driven by a *Process Flow*. A Process Flow controls how a Data Flow runs. Every Data Flow automatically has an associated Process Flow created by DataMigrator for i. When you 'run' a Data Flow, you are actually running the associated Process Flow, which in turn drives the Data Flow. You can also create explicit Process Flows. This is useful when you want to run multiple data flows at once.

Every Data Flow has at least one *Data Source* and at least one *Data Target*. If you are familiar with Db2 Web Query, a Data Source and Data Target are Synonyms (Metadata) representing actual data files or other data sources such as result sets returned from a Stored Procedure or a SQL View

Data Sources include any data available to Db2 Web Query for i. In fact, existing synonyms defined for Db2 Web Query can be used by DataMigrator for i. Database files/tables can be from the same system that DataMigrator is installed on, from one or more remote systems, or both. DataMigrator can also retrieve data from Db2 databases running in Linux/Unix/Windows or z/OS operating systems. Flat files in the IFS (Integrated File System) can also be used as a source of data, including using a file listener capability to help automate the process. DataMigrator can also use journals, including remote journals, as a data source. Journals are useful for incremental maintenance flows where changes to files/tables can be captured with low overhead.

When DataMigrator is installed with Db2 Web Query Standard Edition, adapters allow you to use non Db2 databases as a source or target. Microsoft SQL Server, Postgres, and MySQL

adapters are included as well as a generic JDBC driver that can be used for other relational databases (for example, Oracle).

A *Data Target* is a database file. In other words, it is the target repository for the collected data. A commonly recommended architecture is to have the target database on the same system where DataMigrator is installed such that you are pulling data from the data sources into that system or VM.

For more information about DataMigrator refer to the Db2 Web Query wiki at http://ibm.biz/db2wqwiki.

2 Setup and Configuration

2.1 Setup

2.1.1 Db2 Web Query for i Prerequisite

DataMigrator software is packaged as option 8 of the Db2 Web Query product 5733WQX. It is recommended to install Db2 Web Query products, including DataMigrator, using IBM's EZ-Install package. The Windows client required to set up and use DataMigrator is also available with EZ-Install. To request EZ-Install, send an email to QU2@us.ibm.com and provide your name, company name, serial number where you plan to install and what IBM i OS level you are running.

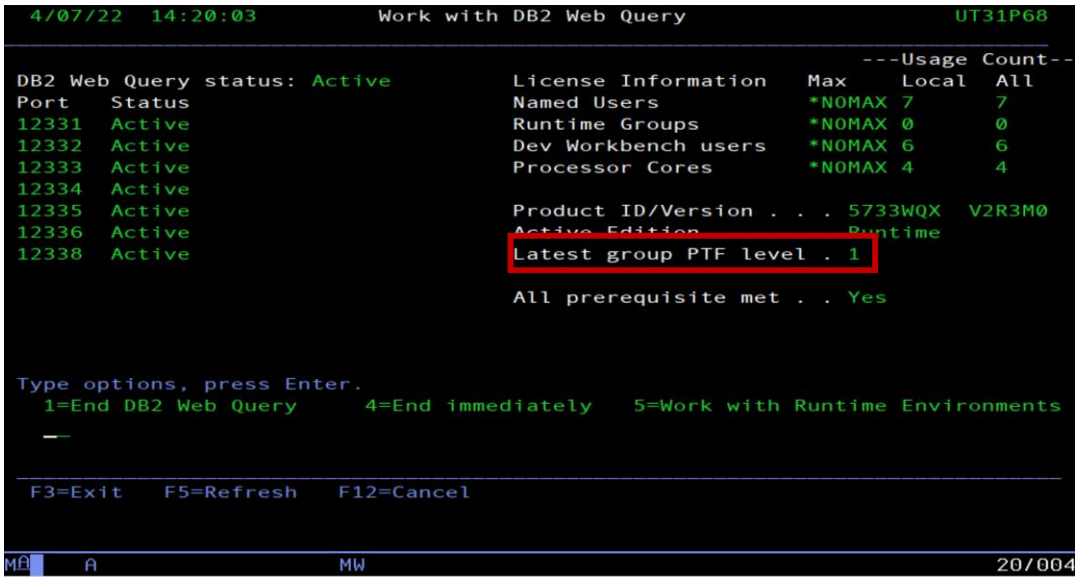


Figure 3 WRKWEBQRY command output to show PTF Group Level 1

2.1.2 Install DM License Key

When you first install DataMigrator and Developer Workbench you can run without needing license keys under the normal IBM i Licensed Program Products trial period, which is 70 days. If you purchased DataMigrator and Developer Workbench you should have a license key for options 5 and 8 from IBM. Install those keys using the `ADDLICKEY` CL command.

2.1.3 Restart Web Query

End and restart Db2 Web Query using the CL commands:

```
ENDWEBQRY
STRWEBQRY
WRKWEBQRY
```

Wait for Web Query to start. Using the `WRKWEBQRY` CL command, wait for all ports to become active:

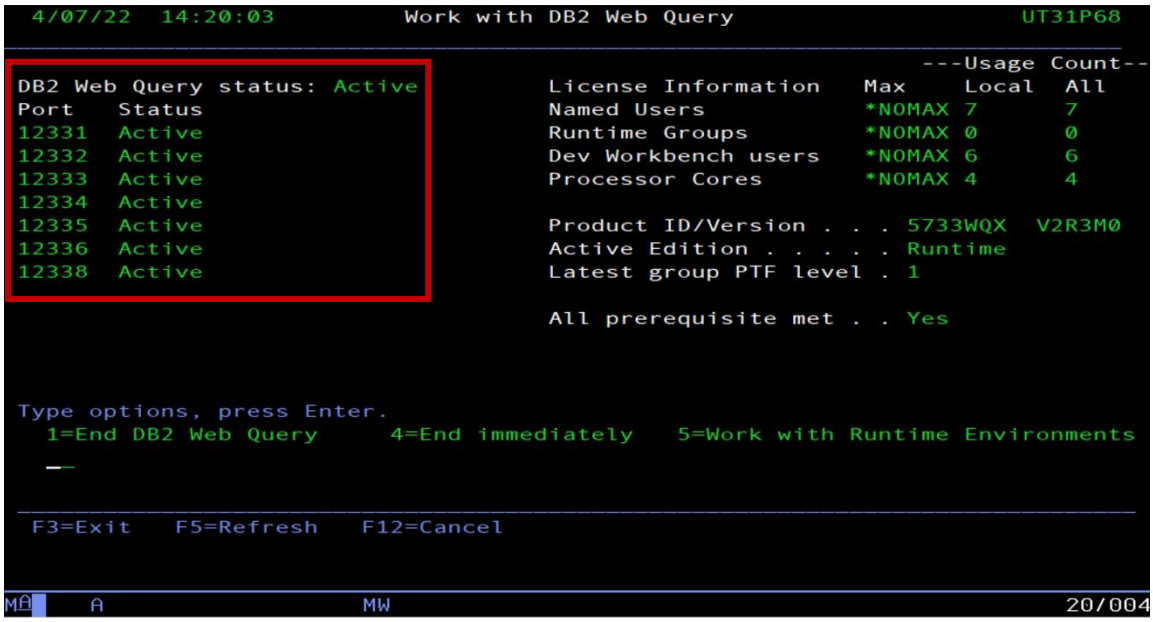


Figure 4 WRKWEBQRY output to show all ports 12331-1233x are active

2.2 Server Configuration

2.2.1 Overview

DataMigrator is integrated with Db2 Web Query for i. Therefore, many of its general management functions are controlled through Web Query interfaces. These include the `WRKWEBQRY` CL command and the Home Page web browser interface. The DataMigrator server component runs in the same backend server environment as Web Query. Consequently, the `STRWEBQRY`, `ENDWEBQRY` and `WRKWEBQRY` CL commands control starting, ending, and configuring the server.

The main interface for working with DataMigrator Data Flows and Process Flows is the Data Management Console (DMC), a component within Db2 Web Query Developer Workbench.

DataMigrator Flow objects are managed within Web Query workspaces, just like Web Query reports and synonyms. *Workspaces used for DataMigrator should be created using the Home Page or Developer Workbench. Do not use the DMC to create workspaces.*

> **NOTE:** The Home Page is the port 12331 interface. It can be accessed using a web browser pointed to address: `http://<yoursystem>:12331/webquery` where `<yoursystem>` is the IBM i where Web Query for i is installed.

2.2.2 Create a Workspace

Using the Home Page, click on `Workspaces` next to Db2 Web Query to view the navigation tree. Then click on `Workspaces` in the navigation tree. In the Action Bar, click the `Folder` button to create a new workspace. Enter the title for the workspace you are creating and Click `OK`.

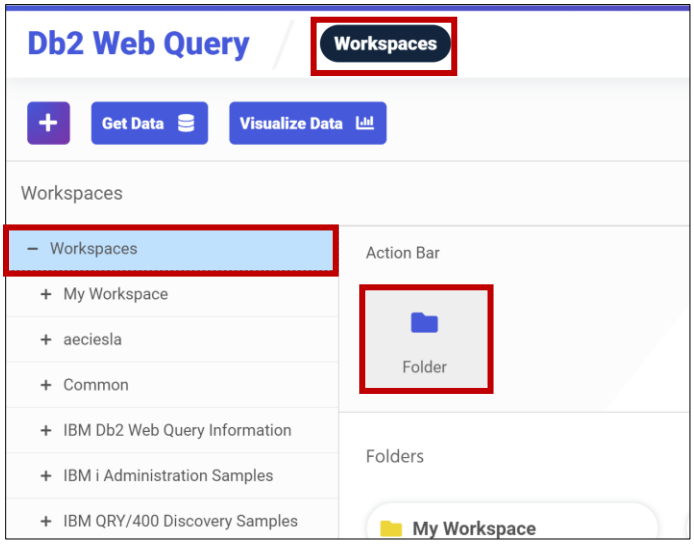


Figure 5 Creating a workspace on the home page

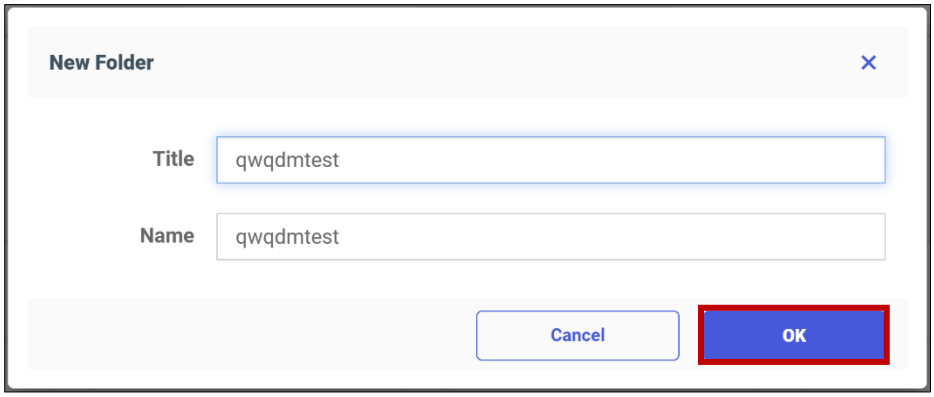


Figure 6 Entering the workspace title on the home page

2.2.3 DataMigrator Developer User

From a user management perspective, a DataMigrator user *is* a Developer Workbench user in Web Query. Any Developer Workbench user in Web Query is automatically allowed to be a DataMigrator user and use the DMC.

To authorize a user to access DataMigrator, you can either run the CL command REGWQUSR (specifying the user ID and selecting *YES for the "Developer Workbench User" prompt), or from the Home Page click the Gear icon in top right corner. Then Click the Security Center option in the drop down.

NOTE: You must be a Web Query administrator in order to access the Security Center.

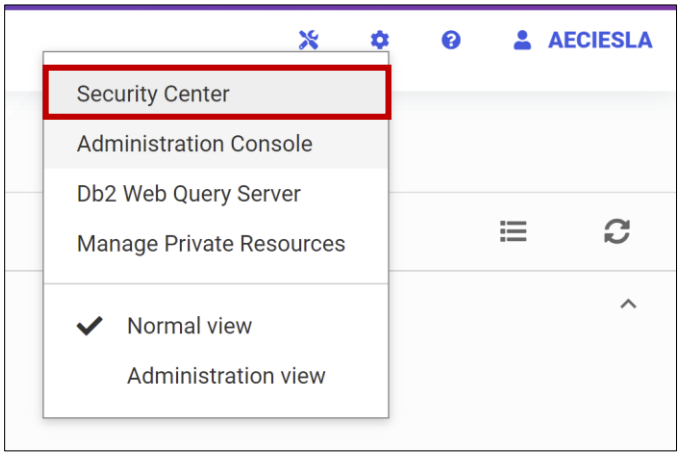


Figure 7 Security Center option on the home page

Highlight the `DevWorkBench` Group and the desired Web Query user and press the >> button. The user should be added to the User list in the lower right box.

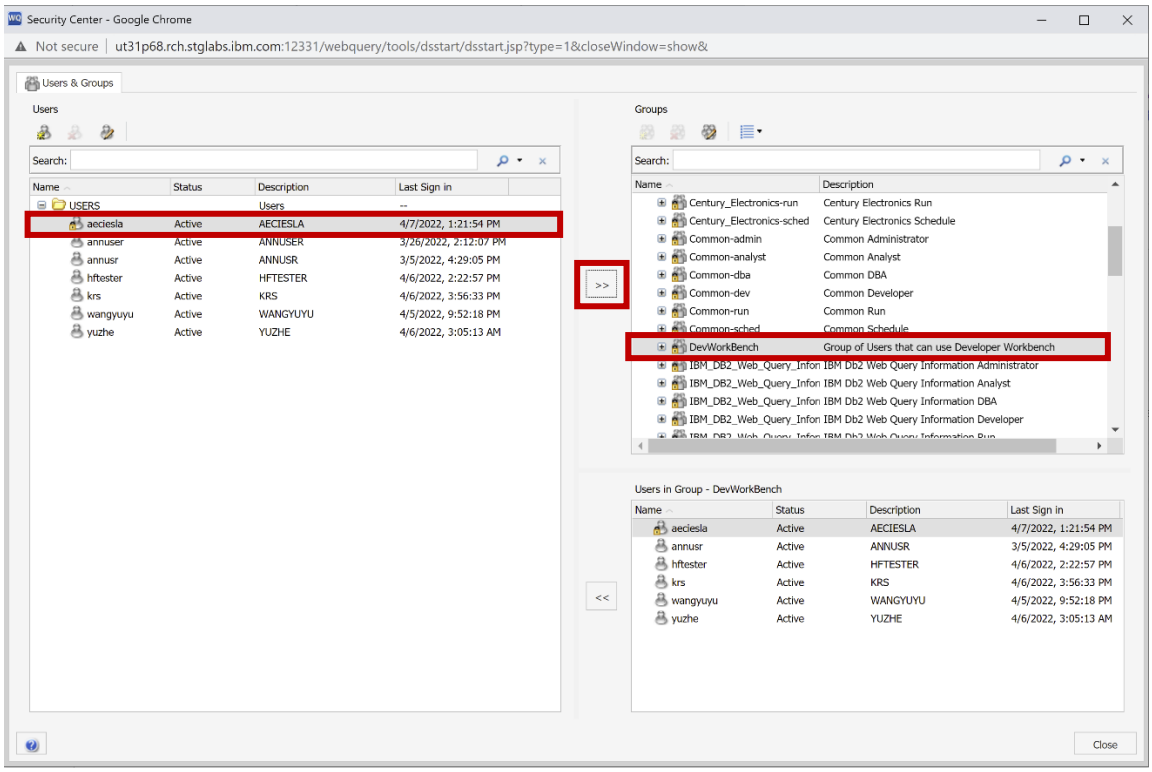


Figure 8 Enabling a user for Developer Workbench and the DMC

Next, provide the user with developer or admin access to the DataMigrator top level folder created earlier the same way as above.

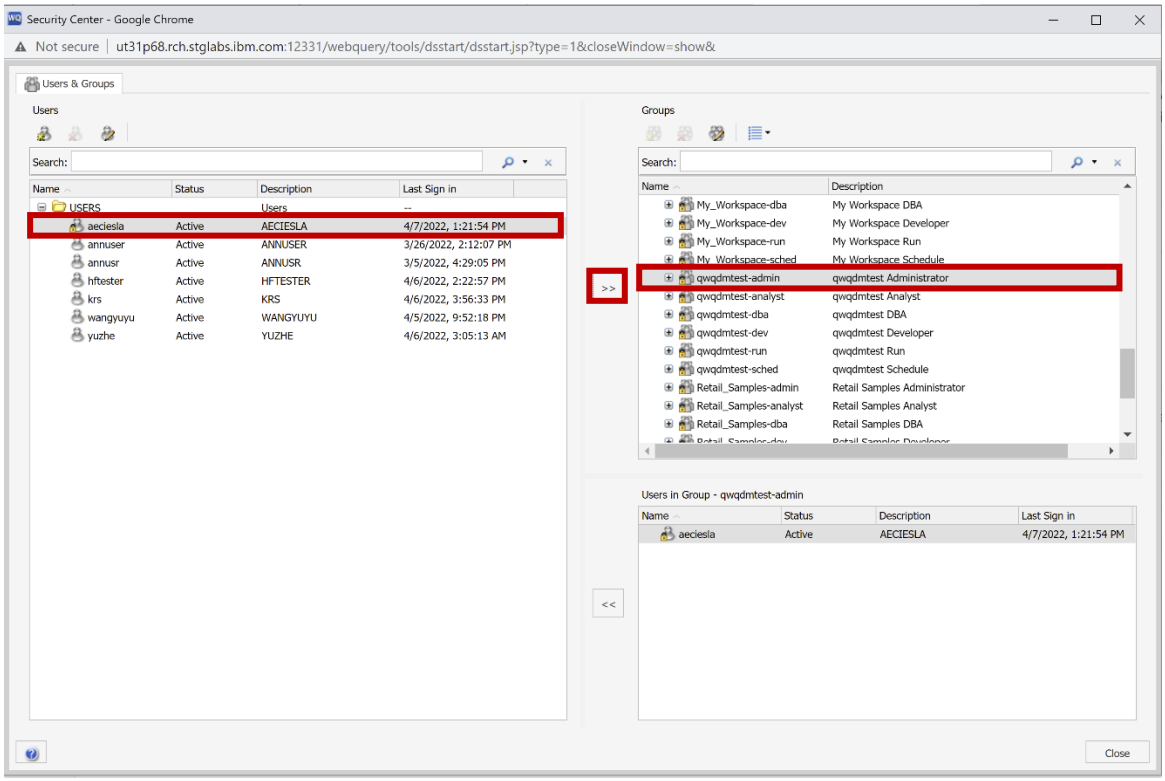


Figure 9 Giving admin or developer access to the DataMigrator folder

2.3 Install and Configure the DMC

If you installed using EZ-Install, the Developer Workbench installation image will be available in the location of the packaged provided to you. Download the Developer Workbench installation file to the user's PC. The Data Management Console (DMC) is part of the Developer Workbench install image. As you update your web query software by applying a PTF Group, the matching installation files for Developer Workbench are located in the IFS in this location: `/QIBM/ProdData/QWEBQRY/DeveloperWorkbench`.

Run the .exe file to install Workbench. This will also install the DMC. Once the installation completes, the DMC can be found as the Data Management Console in the Db2 Web Query for I Developer Workbench folder.

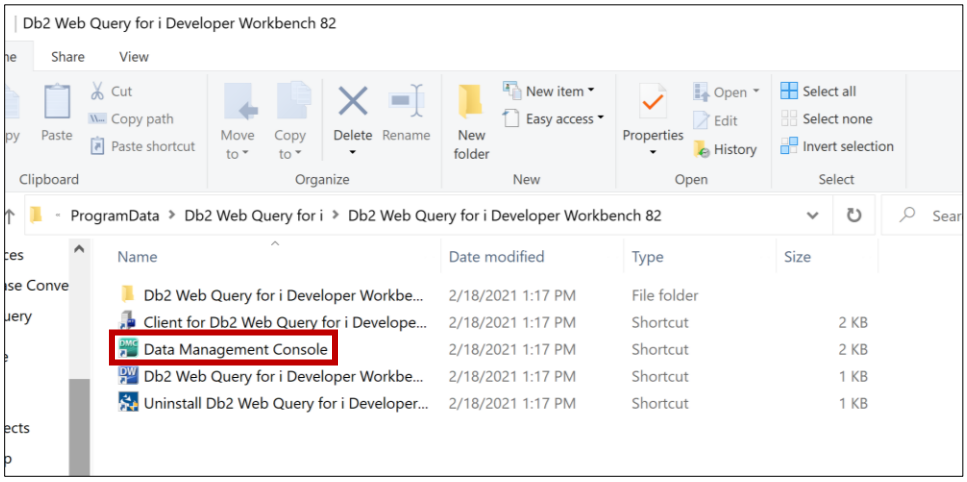


Figure 10 Finding the DMC

Open the DMC. Now the Web Query/DataMigrator server needs to be registered to the DMC. DataMigrator runs off ports `12333` and `12332` which are specified during the server configuration.

To add a server, either highlight the `Servers` folder and click the `New` button at the top or right click on the `Servers` folder and `Add Server Node`. For `Host`, enter the network name of the Web Query server. Specify port `12333` for the `HTTP Listener Port` and `12332` for the `TCP Listener Port`. The `Security Type` should be `explicit`. You may either provide the `User ID` and `Password` at this point or wait and be prompted during the server connection.

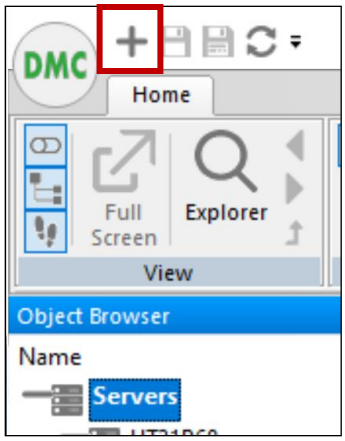


Figure 11 Adding a server

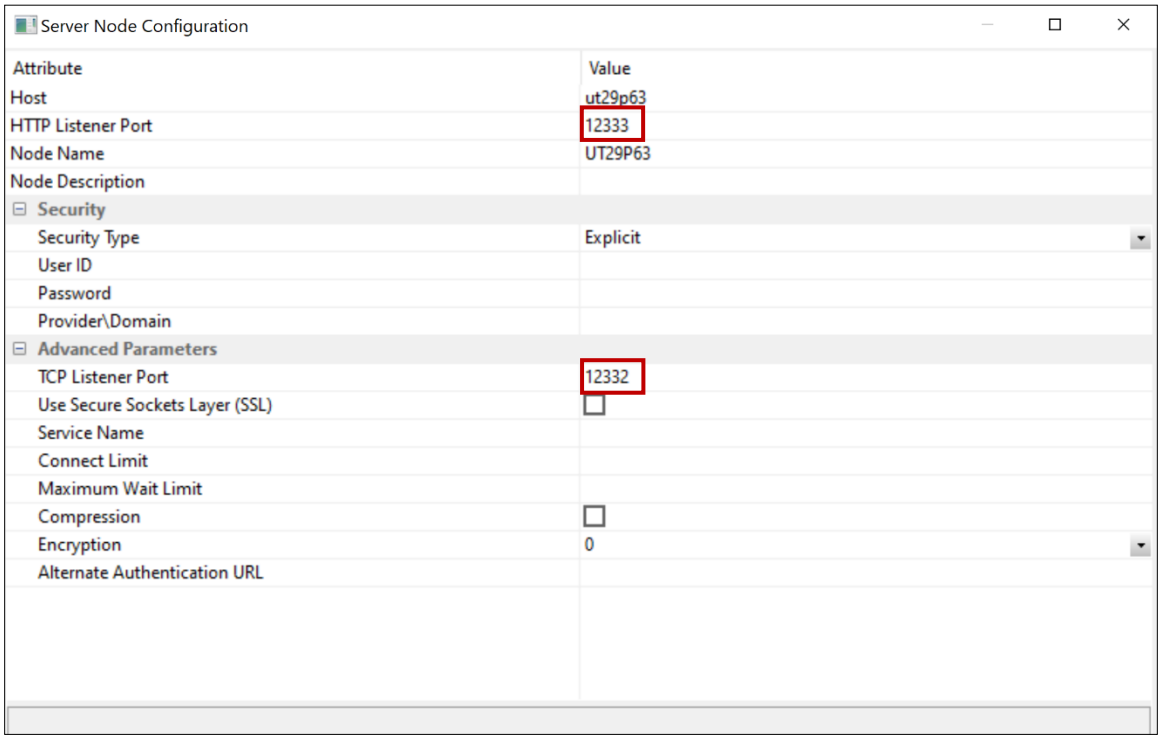


Figure 12 Configuring a server connection

Click `Save`.

Finally, register the workspace created earlier through the Home Page. Find and expand the newly added server in the left navigation tree in DMC. Right click on the `Application Directories` folder under the expanded server view, then click `Manage` and choose `Application Path`.

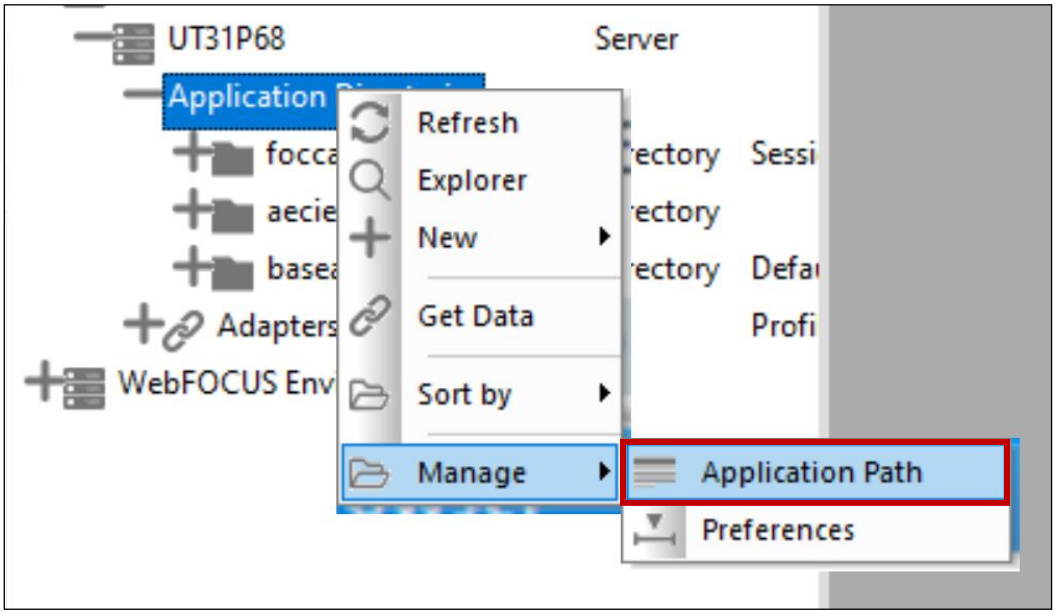


Figure 13 Choosing an application path

Click `Change Profile Precedence` and choose `Append to previously executed profiles`.

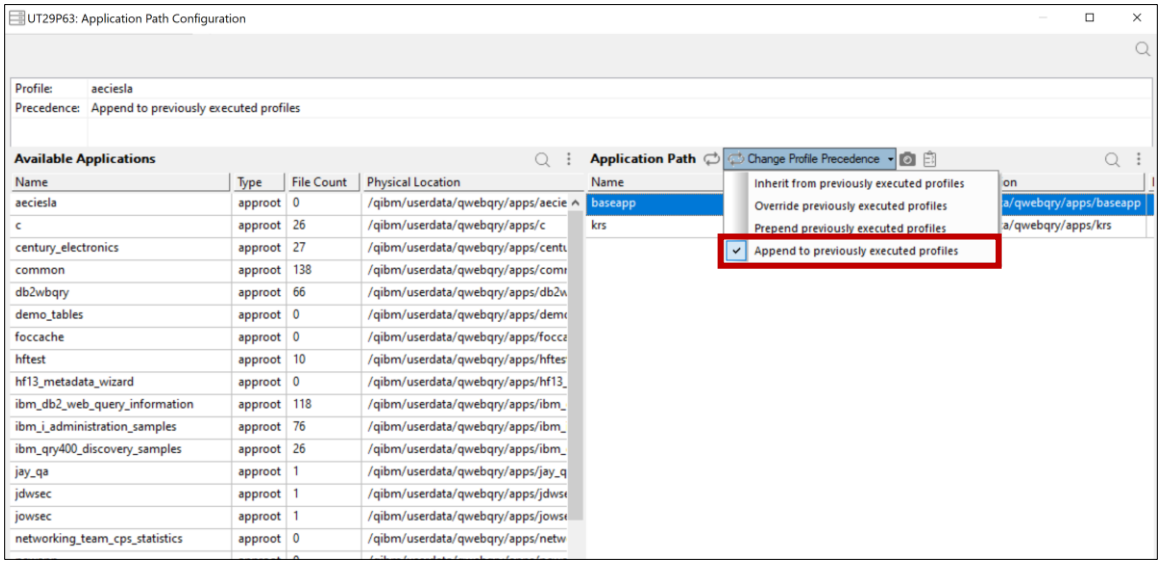


Figure 14 Changing profile precedence

Find and right click the workspace. Click `Add to Path`. Then click `Save` and `Ok` and close the dialog.

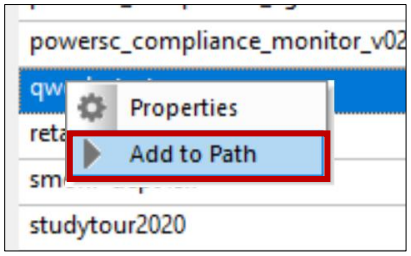


Figure 15 Select the workspace

Now the workspace should show in the navigation tree.

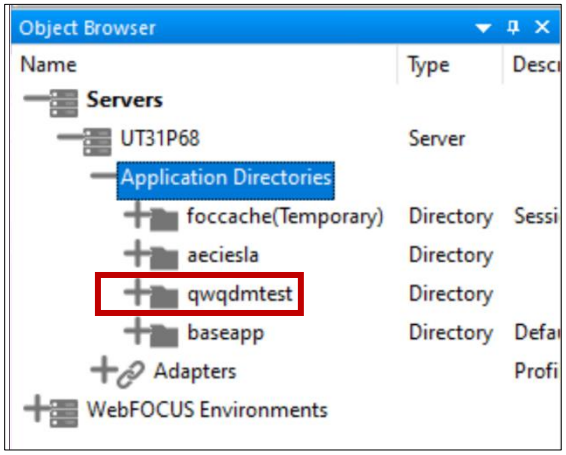


Figure 16 Workspace in the navigation tree

NOTE: The folder `baseapp` is included by default when adding folders to the Application Path. If you wish to remove it, for example to 'clean up' the list of folders under the Application Directories list, you may. To do so, right click `Application Directories`, click `Manage` then `Application Path`. Right click `baseapp` on the right side and click `Remove from Path`.

2.4 Verify the QWQCENT library

The examples in this guide use tables in the QWQCENT sample database library. This library is also referenced as the Century library. The QWQCENT library is shipped with the 5733WQX product.

The IBM Db2 Web Query development team makes occasional changes to the QWQCENT library. When this happens, a new save file of the library will be shipped as part of a group PTF. The save file will always be named QWEBQRY/QWQCENT.

This guide uses the version of QWQCENT found inversion 2.3.0, PTF Group Level 1. . If you have not made any customization of QWQCENT, it is highly suggested that you restore the latest version of the library from the shipped save file using this command:

```
RSTLIB SAVLIB(QWQCENT) DEV(*SAVF) SAVF(QWEBQRY/QWQCENT)
MBROPT(*ALL) ALWOBJDIF(*ALL)
```

If you already have a version of QWQCENT installed on your system, it is possible that certain table columns are missing or have different names than those used in this guide.

2.5 Conclusion

At this point setup and configuration are complete. You are ready to start creating Flows!

3 Creating A Simple Data Flow

3.1 Overview

The Data Management Console (DMC) is the primary interface for DataMigrator for i. We will use the DMC throughout this document for building, editing, and running flows. Find and start the DMC.

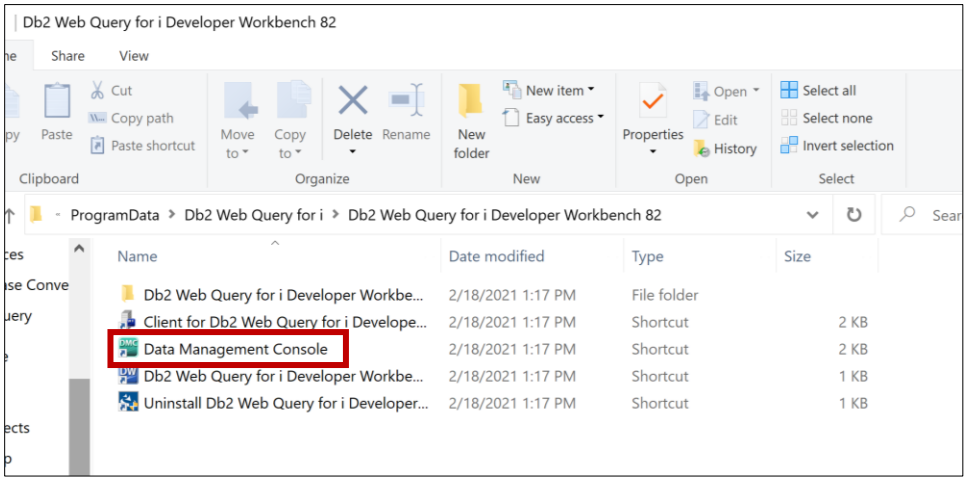


Figure 17 Find and start the DMC

You should already have your DataMigrator server added. Expand the server in the navigation tree and find the `qwqdmtest` folder created during setup.

The main development task of DataMigrator is creating Data Flows. As mentioned in the introduction, Data Flows define how data is gathered, processed, and stored in an *ETL* process.

Before creating a data flow, we will look at some basic DMC feature landmarks that will be useful to create and run flows.

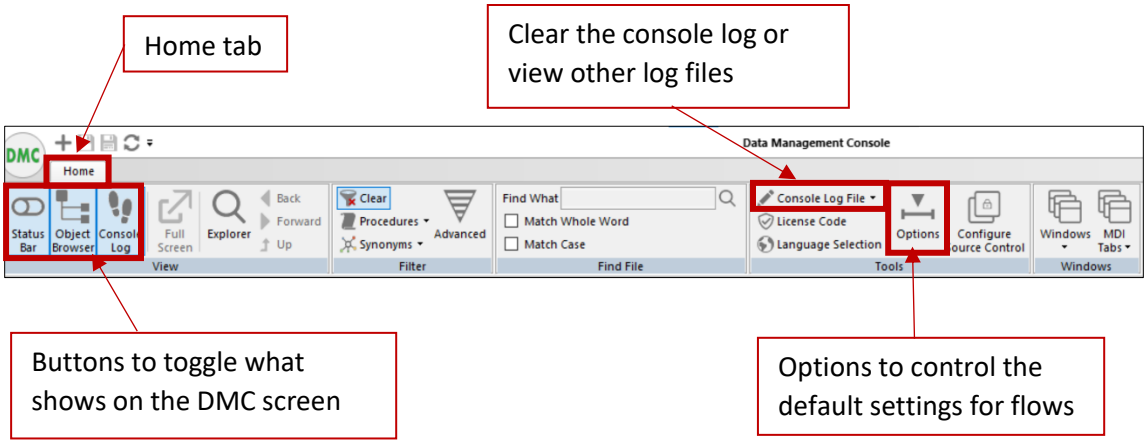


Figure 18 DMC ribbon interface

The DMC has a ribbon interface on the top of the window. It provides a set of options that change by context. Depending on what object you have highlighted in the DMC, the options in the ribbon can change.

The figure above shows the ribbon layout when viewing the Home tab. As you start to create flows and other objects, other tabs will show up, for example Flow, and the ribbon will change. At any point while working in the DMC you can click the Home tab and get back to this ribbon.

The Home ribbon has some important features that can affect how the DMC works.

- The buttons on the side of the ribbon toggle what is shown on the DMC screen.

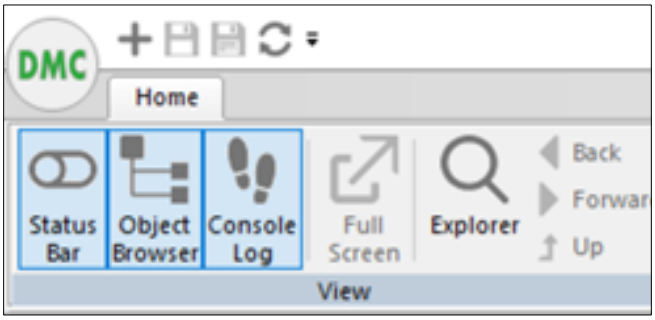


Figure 19 Display toggle buttons

For example, if you are working on a DMC flow and the console log is not showing, click on the Console Log button to bring it back. If you notice your DMC window is missing something shown in a figure in this guide, use the buttons on the Home ribbon screen to restore the 'missing' item.

- When working with log files, the Console Log File option on the ribbon is useful for clearing the log file or switching to a specific log file.

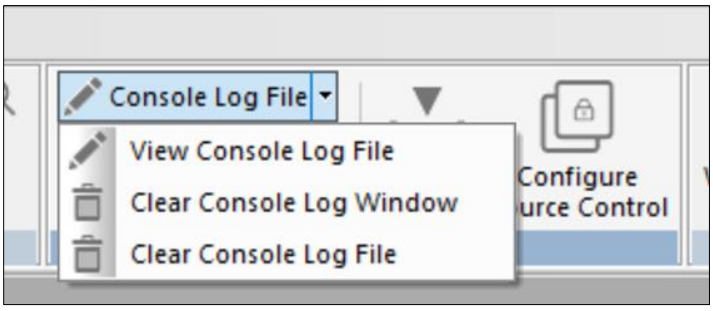


Figure 20 Console log file control

- The Options button on the ribbon brings up a window with a set of options that dictate how the DMC works. You can change the default settings to change the way the DMC looks or operates.

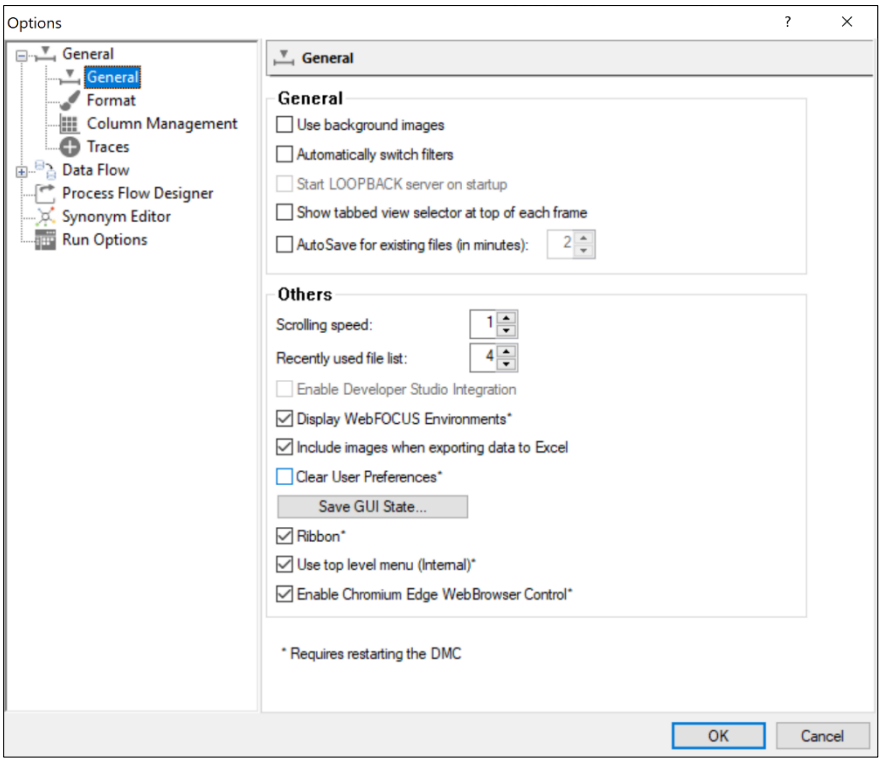


Figure 21 Options window

For example, if you want to change displays to show column names rather than titles when working with tables, click:

`Options, Column Management, Column name display strategy`

- Additional help text and the online guide can be found by clicking on the arrow next to the *i* information button in the upper right of the DMC window.

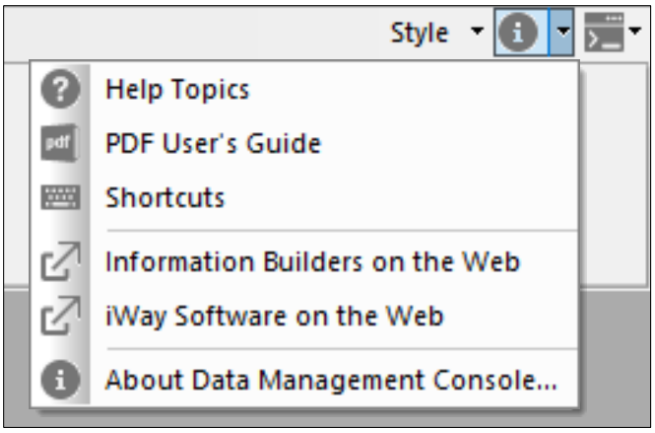


Figure 22 Information dropdown menu

`Help Topics` allows you to search for more details on a topic. The PDF User's Guide is a full user's guide.

Now, let us get back to the steps in creating a flow.

3.2 Defining synonyms (Data Sources)

To create a data flow, we must first identify data sources. Db2 Web Query, which DataMigrator is an extension of, comes with a sample database in library `QWQCENT`. We will use its files as our data sources.

As a reminder, data sources are represented by metadata objects referred to as synonyms. In Db2 Web Query, synonyms can be created through the Home Page ("Get Data" or "Prepare and Manage Data" options) or Developer Workbench. Synonyms created from those interfaces can be used for DataMigrator as well. However, the DMC also provides a way to create synonyms which we will use now. Right click on the test folder and click `New` then `Synonym`.

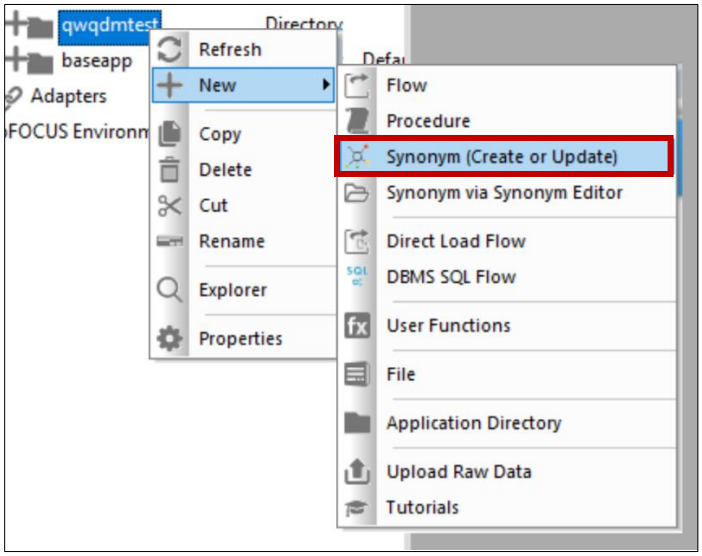


Figure 23 Creating a synonym

Double click the `*LOCAL` adapter.

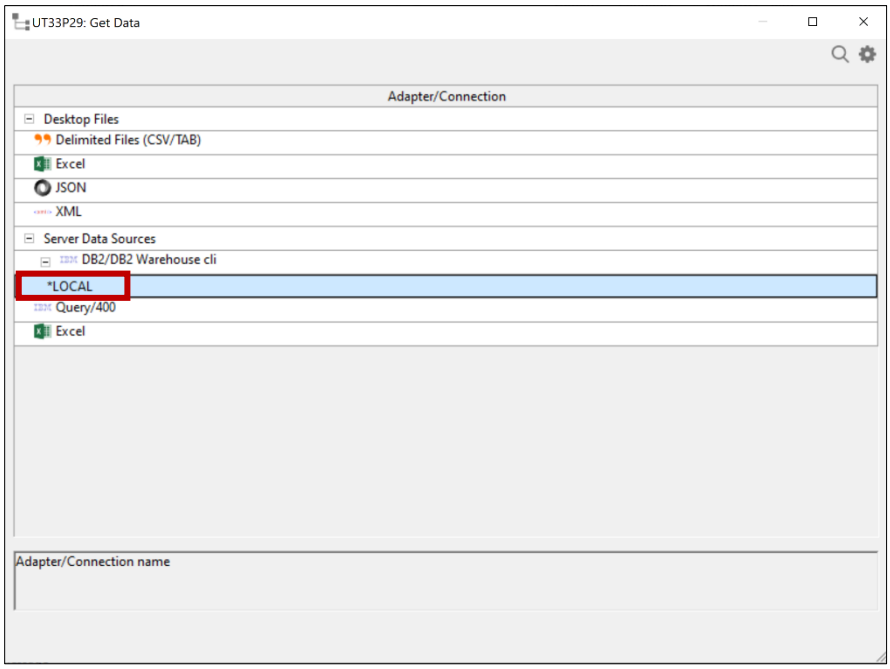


*Figure 24 Select the *LOCAL adapter*

In `Object Type`, choose `Tables`. In `Library`, type `QWQCENT`. Then click `Search`.

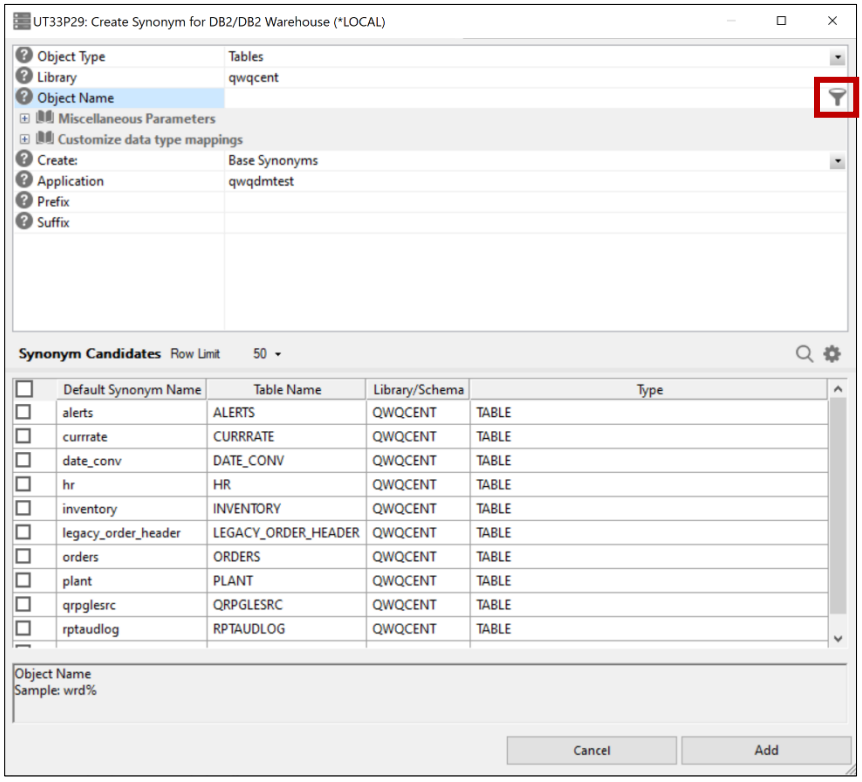


Figure 25 Creating synonyms for QWQCENT

Select all the tables shown *except* for *QRPGLESRC.* In Web Query, a best practice is to provide a prefix to synonyms. In this case, use the prefix `cen_`. Then click `Add`.

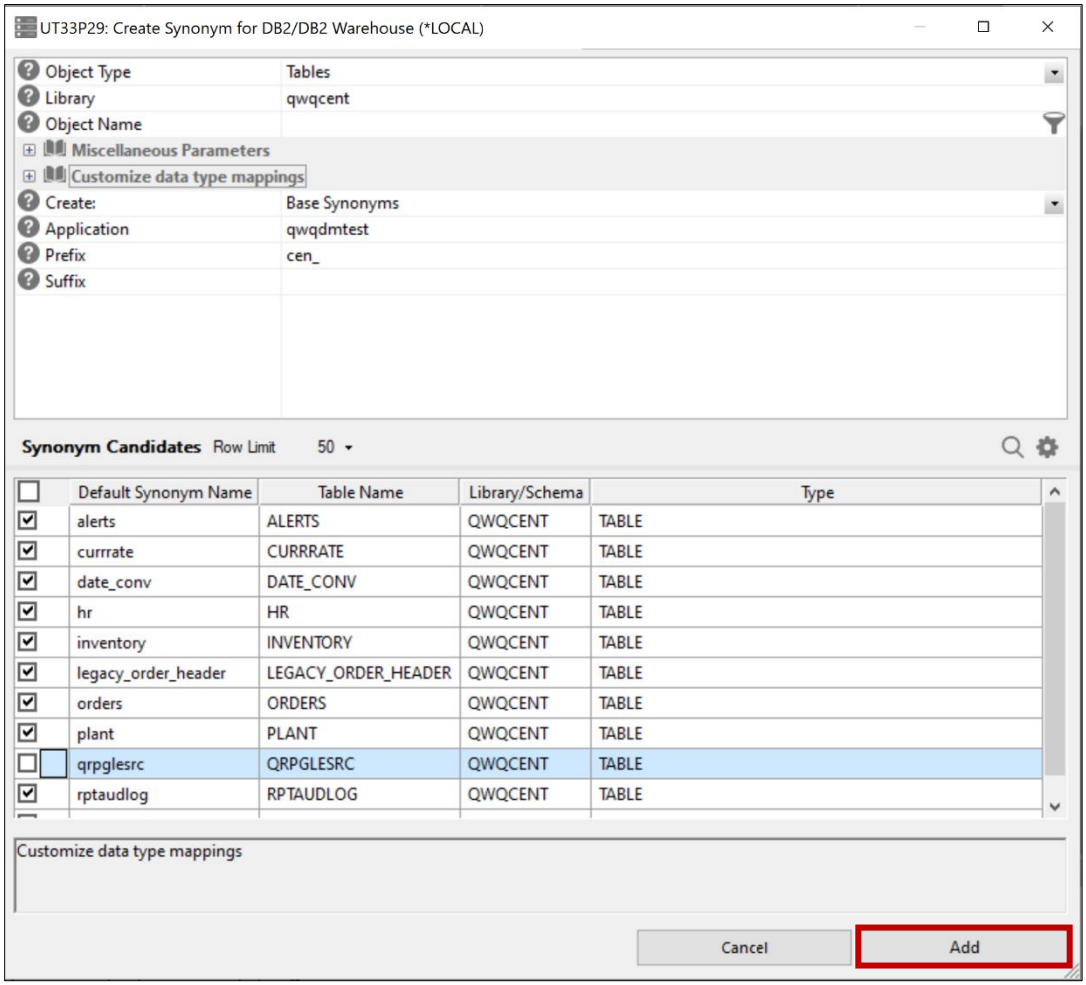


Figure 26 Choose tables to create synonyms for

The synonyms should be created successfully. Close the resulting status window.

We are now ready to create a data flow.

3.3 Creating a Flow

Right click on the folder and click `New`, then `Flow`.

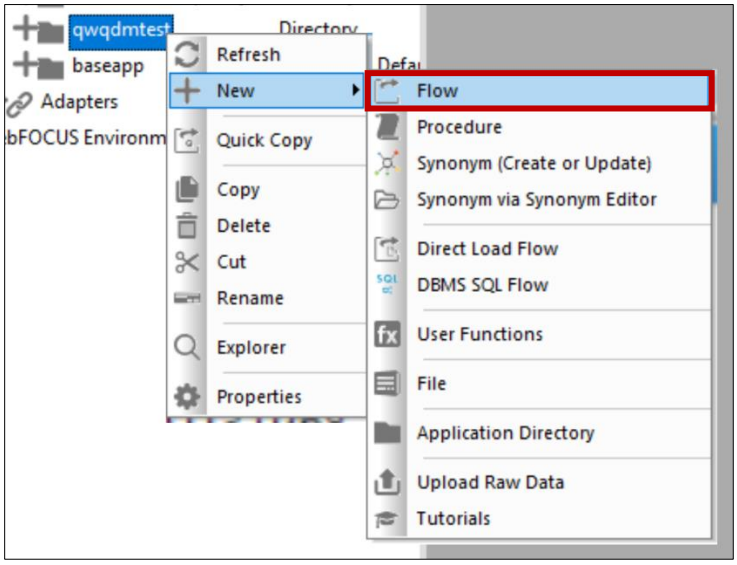


Figure 27 Creating a new data flow

> **NOTE**: If you do not see the `Flow` option on the context menu, it means the setup did not complete successfully. Either:
>
> 1. option 8 of the 5733WQX product is not installed,
> 2. the enablement PTF Group Level was not applied,
> 3. Web Query was not restarted or
> 4. the user you signed in as for DMC is not a Developer Workbench user.
>
> Go back to Chapter 2 Setup and Configuration and verify the steps.

The DMC workspace should now show the data flow in a Data Flow tab with the `SQL` icon showing on the palette.

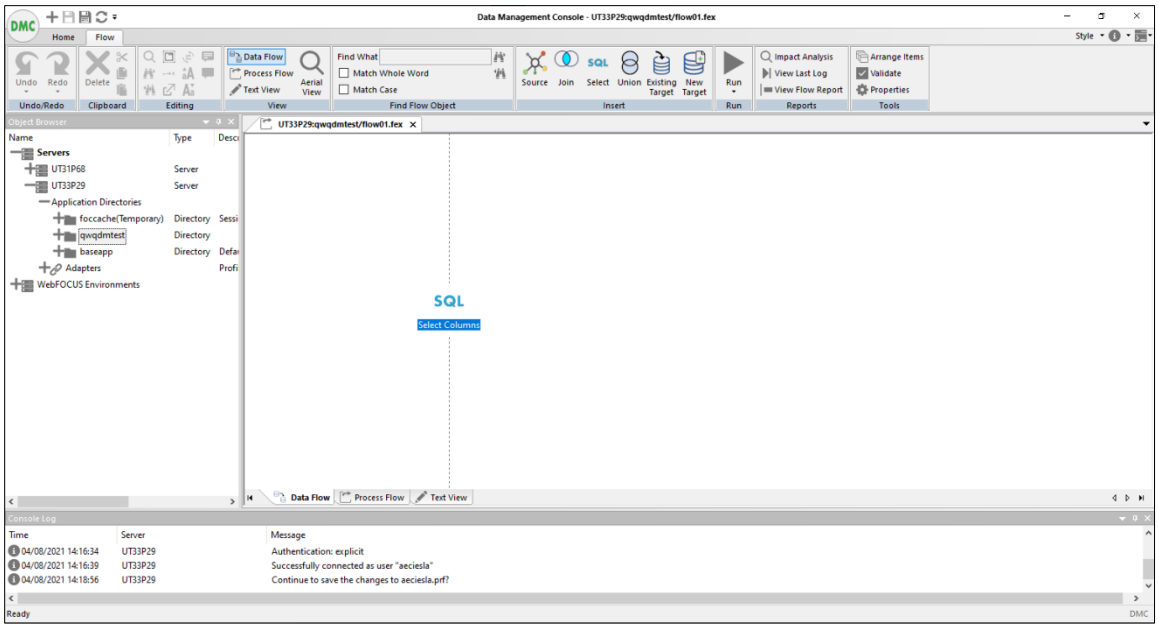


Figure 28 New data flow palette

Now we have data sources showing in the navigation tree on the left and the data flow on the right.

> **NOTE:** If the data sources do not show, expand the navigation folders by clicking on the + sign next to them.

The `SQL` icon and the dotted vertical line through it represent the major division in the data flow. Objects to the left of the line are data sources. Objects to the right are data targets.

To create the data flow, drag in a data source from the navigation onto the left side of the data flow palette. In this case choose `cen_inventory`.

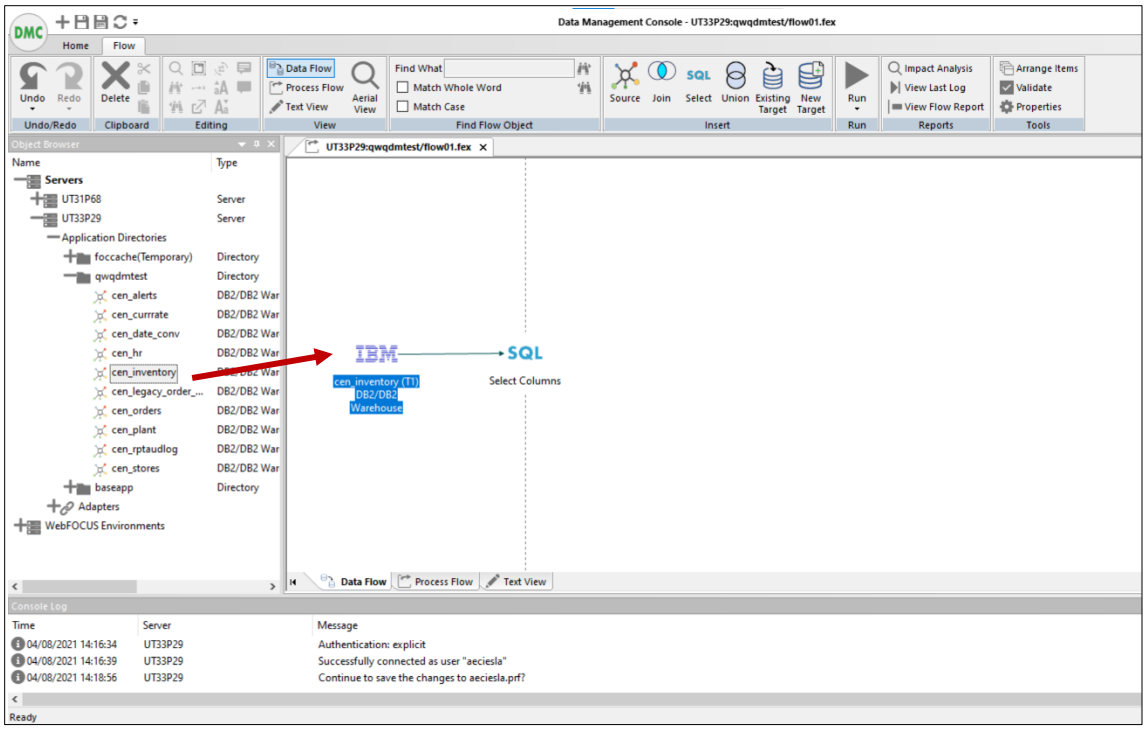


Figure 29 Choosing data source cen_inventory

When the data source (synonym) is dropped on the palette, DMC automatically connects it to the `SQL` icon with an arrow. This indicates the flow of data from the data source into the *SQL* operation.

> **NOTE:** If you drop the synonym to the right side of the `SQL` icon line it will become a data target, indicated by an arrow pointing into it. It is important to pay attention to which side you drop a synonym on as it could become the target, meaning data would be written into it if you ran the flow!

Alternatively, you can right click on the left side of the palette or on the `SQL` icon and click `Add`, then `Source` and select the synonym for the data source.

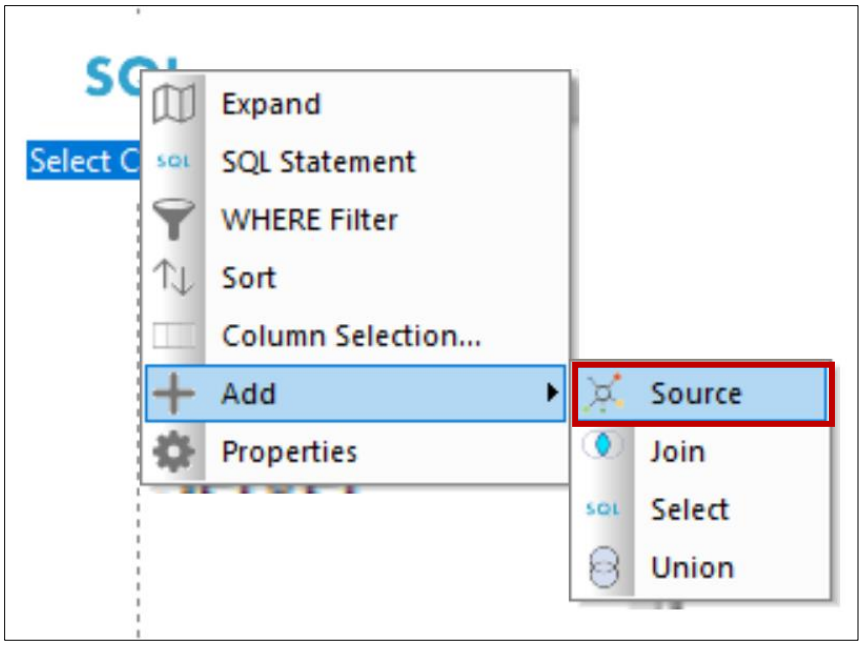


Figure 30 Alternate method for adding a data source

At any point you can sample data in the data source to verify its contents. Right click on the data source and click `Operations`, then `Sample Data`. This will bring up a tab with a data sample from the underlying file. You can close the tab with the `X` as shown.

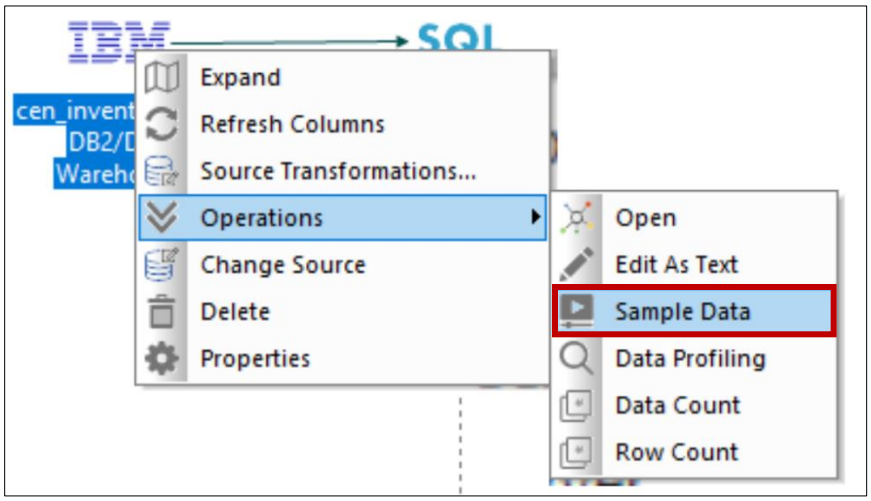


Figure 31 Sample data from a data source

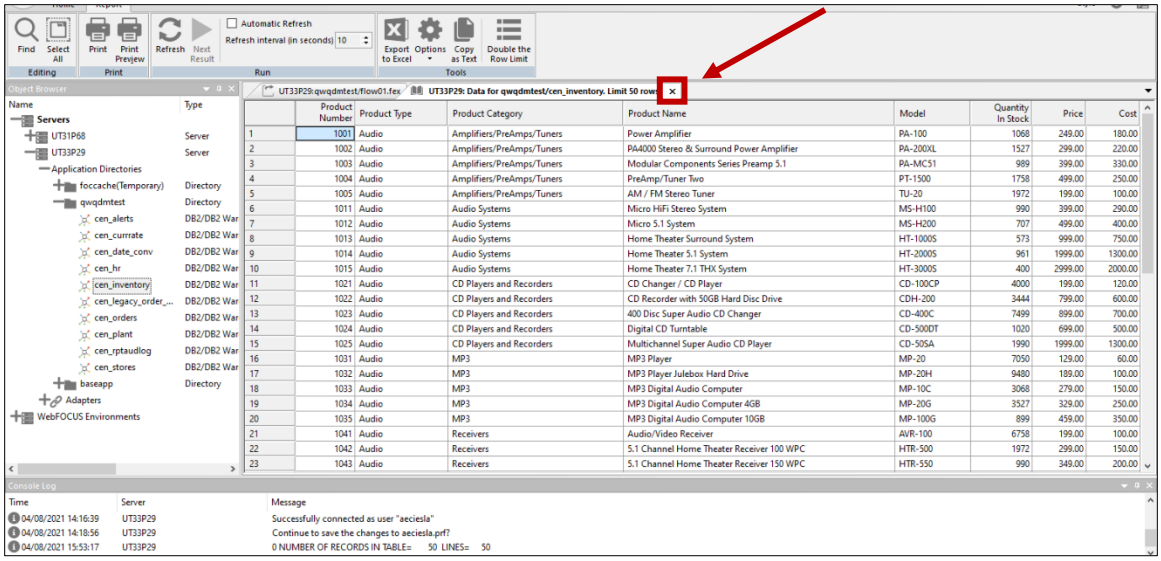


Figure 32 Sample data tab

For now, we will complete a simple flow by adding a data target.

3.4 Adding Data Target

First, we will create a library to contain the resulting database table. Using Access Client Solutions (ACS), create an SQL schema called QWQDMTEST. Alternatively, you can use the `STRSQL` CL command.

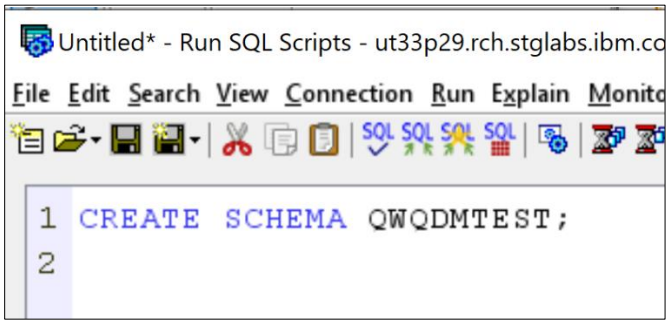


Figure 33 Creating schema QWQDMTEST

Now switch back to DMC.

> **NOTE:** DataMigrator can create a new data target table or insert into an existing one. Depending on your intended use, you may want anything from a simple copy of the source, frequently called an Operational Data Store (ODS) to something more involved where the target is a more optimized data model such as a Data Warehouse or Data Mart. A "Star Schema" or "Snow-flake" model is a commonly used data model for analytical applications. This document does not cover this topic but there are many resources available to learn more about these and proper data modeling.

Add a data target. Right click on the right side of the palette and click `Add Target` then `New`. Alternatively, you can click on `New Target` in the ribbon and drag it onto the workspace.

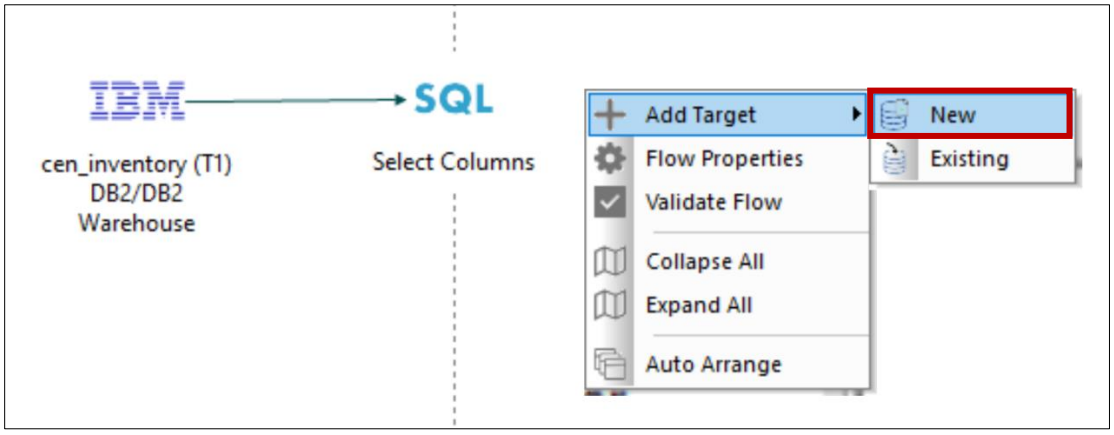


Figure 34 Adding a new data target

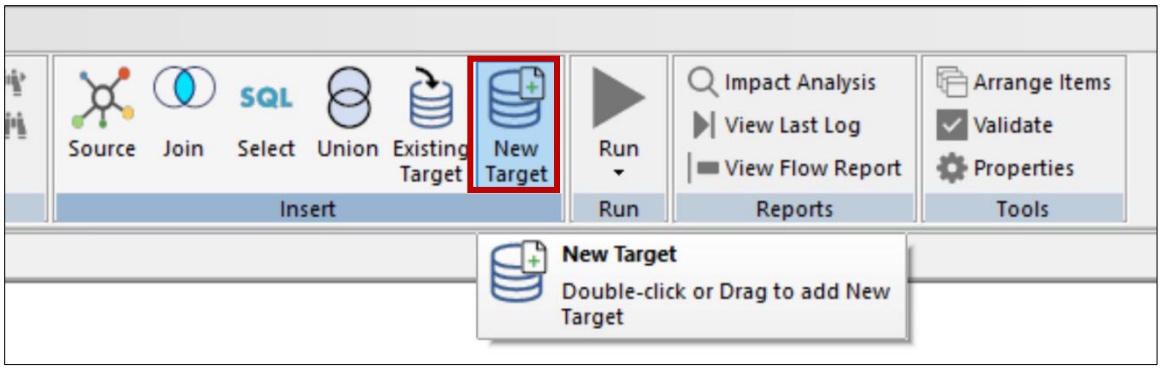


Figure 35 Alternative for adding a new data target

Once the data target is placed on the workspace, right click on the target icon and select `Properties`.

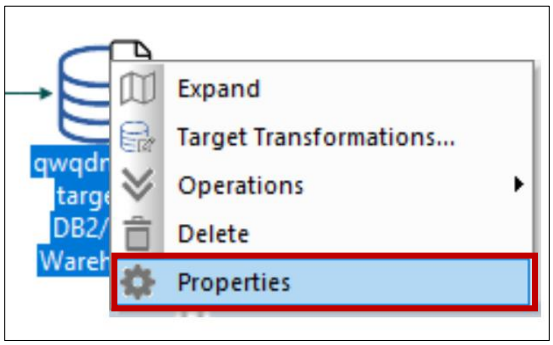


Figure 36 Selecting data target properties

The `Properties` dialog allows you to control aspects of the underlying database target table and how it is populated. There are three items of interest:

1. *Synonym name* - By default, a synonym will be created (during the run), named `targetxx` (where `xx` is an increasing number) and will reside in the folder where the flow was created. This can be change by choosing the ellipsis to the right of the text. We will leave it as is.

2. *Table* – The underlying database table name and location. By default, the database table will be the same name as the synonym, `targetxx` in this case. It will not be library qualified, which means the database table will be created in the first writeable library in your library list. The library can be explicitly specified using the lib/file IBM i notation. For this example, change the table library and name to `qwqdmtest/dmtarget01`.

3. *Load Type* - This determines how the records are read from the data source(s) and written to the table. By default, this is `Insert/Update`. However, the `Insert Records From Memory` option has better performance for the 'bulk load' we will be doing, since it allows for more record blocking. Choose this option.

NOTE: You may notice that the 'Bulk load utility disk file' is no longer shown for Load Type. This is because the flow property 'Optimize Load' is checked by default. This property allows for faster loading by combining insert and update operations. Unchecking the property allows you to choose the 'Bulk load utility disk file' for Load Type.

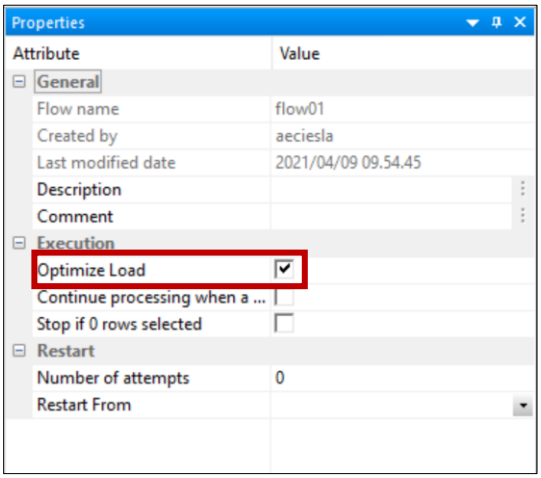


Figure 37 Optimize load property for a flow

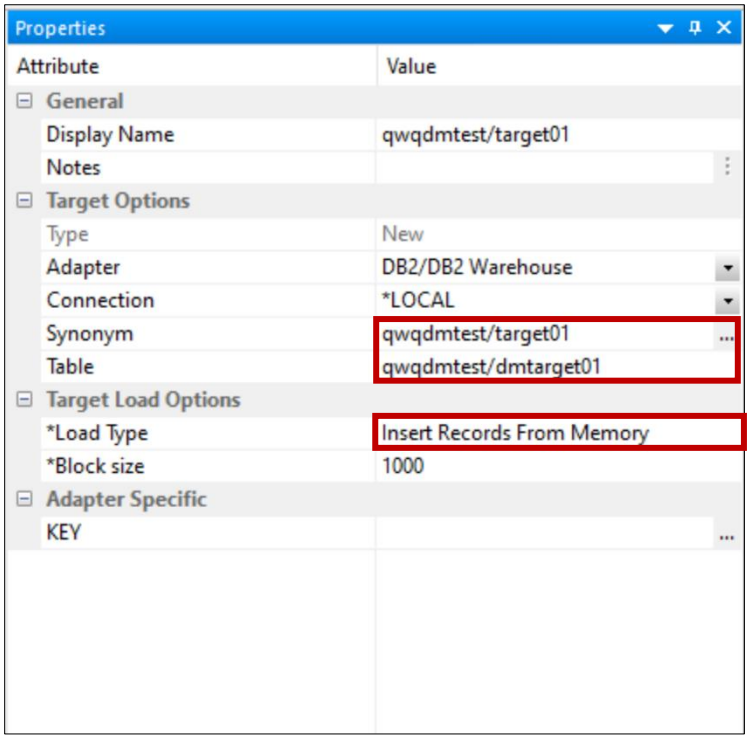


Figure 38 Setting properties for the data target

NOTE: The `Properties` dialog changes depending on what item or icon you have highlighted. If you find that the `Properties` do not look like above, (re)click on the data target icon to bring this `Properties` context back into focus.

3.5 Running a Flow

At this point, we will attempt to run this simple data flow. Click on the `Run` option in the ribbon.

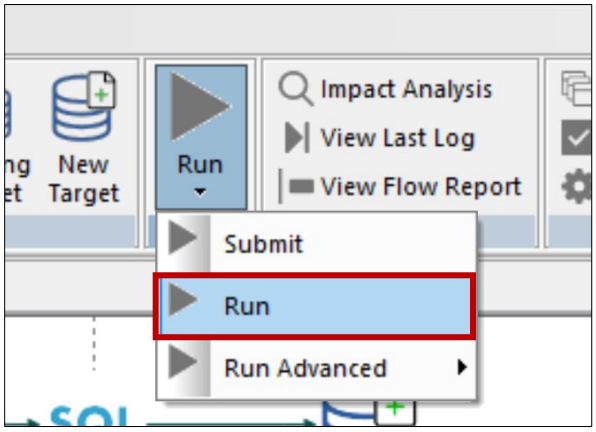


Figure 39 Run flow option

There is an error stating columns are missing.

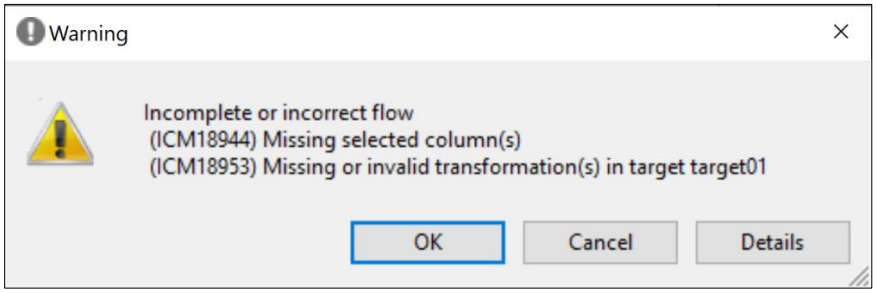


Figure 40 Error from running the flow

This error highlights an important distinction in DataMigrator compared to a simple file copy tool. Because it is an ETL product and not just a file copy tool, DataMigrator assumes you will be doing some transformation or selection in the data flow. In this case, we defined a source and a target, but no column mapping or data transformation was given. Even though DataMigrator *can* be used for simple file copies, it really is built to do more than that[1].

The `SQL` icon signifies the transition from the source to the target. It is a major point where mapping and transformation can be defined. We will utilize it now.

We will go back and add the list of columns we want mapped from the data source into the target. Right click on the `SQL` icon and choose `Column Selection`.

[1] That said, the default behavior for including columns can be overridden via 'Options'

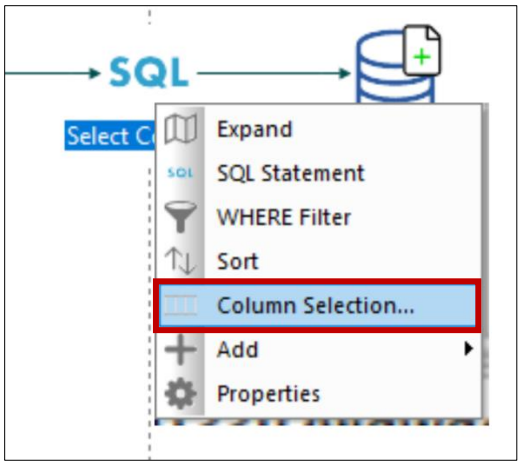


Figure 41 Column selection

The `Column Selection` window appears. The left side shows the columns available from the data source. Highlight all columns and select them by clicking the >> button in the middle of the window. Alternatively, select each column one at a time and select it with the >> button.

> **NOTE:** To highlight all columns, left click the first column. Then hold the `Shift` key and left click the last column.

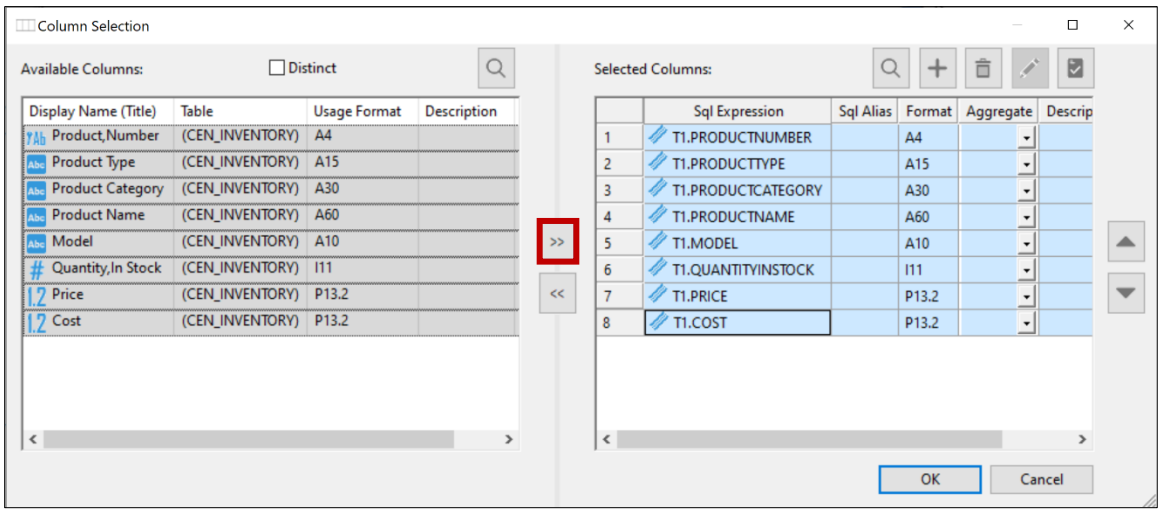


Figure 42 Selecting columns from the data source

Before continuing, examine the Column Selection window. Note that there are several powerful capabilities available. For example, duplicate rows can be eliminated during the run of the flow by checking the `Distinct` option at the top. Also, the order in which the columns appear in the target can be rearranged by highlighting a selected column and using the up/down arrows to move it around. More complex operations like expression and aggregate (group by) processing can also be specified. We will revisit these in more detail in a later section.

Once all columns have been selected, click `OK` on the Column Selection window. Click on the `Run` option again in the ribbon above the workspace. This time the flows should run successfully.

Look at the Console log at the bottom of the DMC screen to verify it worked.

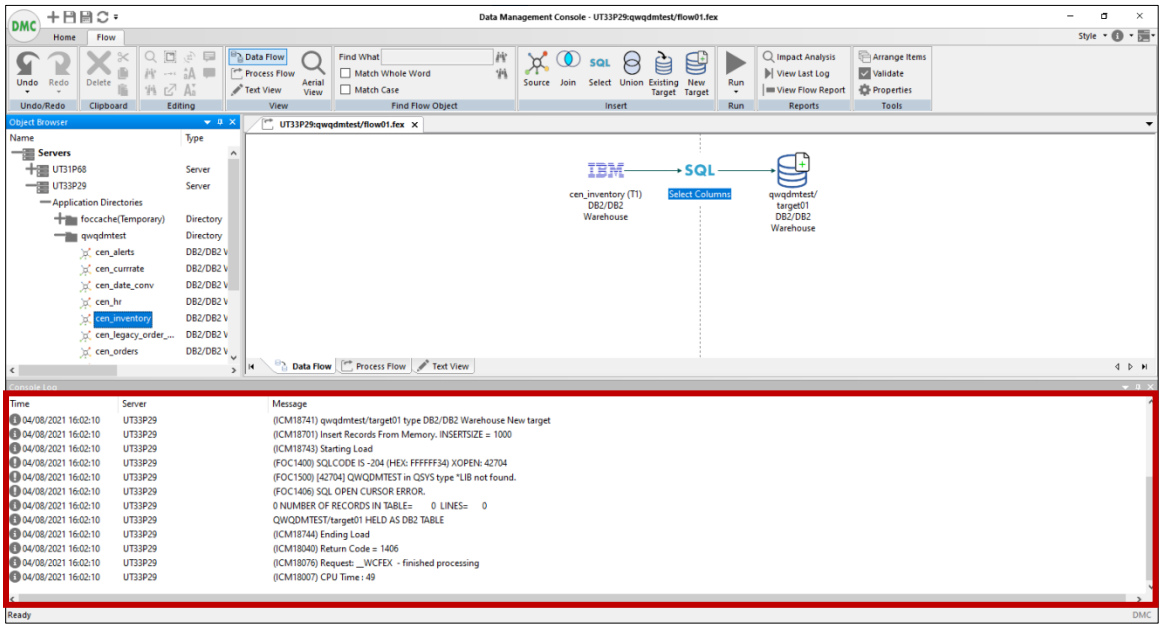


Figure 43 Console log showing the flow ran successfully

You can verify the target table was created and populated using three methods:

1. Look in the qwqdmtest library to verify the file was created.
2. Look at the target file contents from outside the DMC e.g. SQL.
3. Use the Sample Data option from the data target icon (right click).

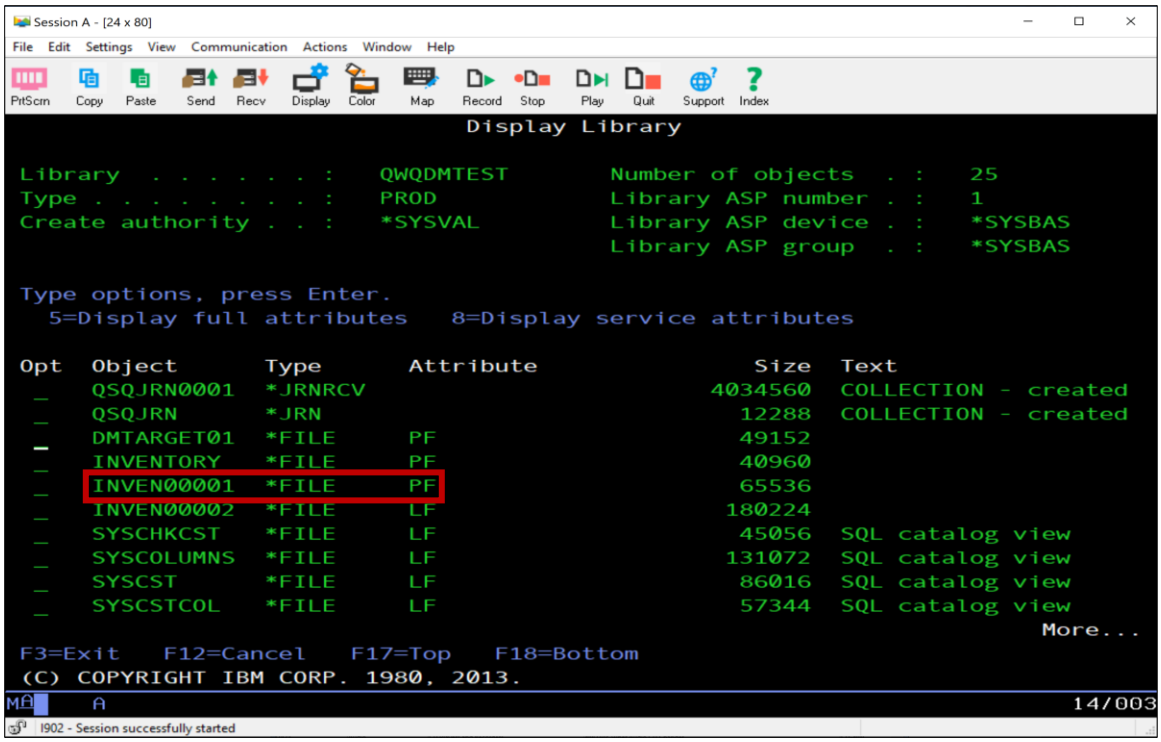


Figure 44 Viewing the data target using the DSPLIB CL command

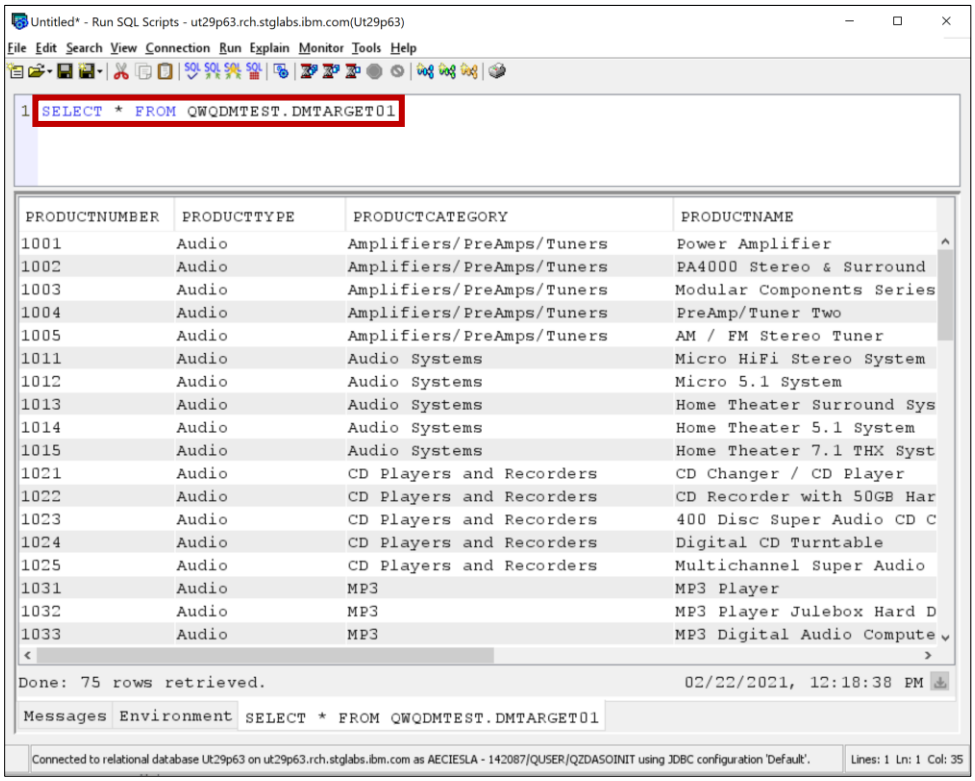


Figure 45 Viewing the data target using SQL in ACS

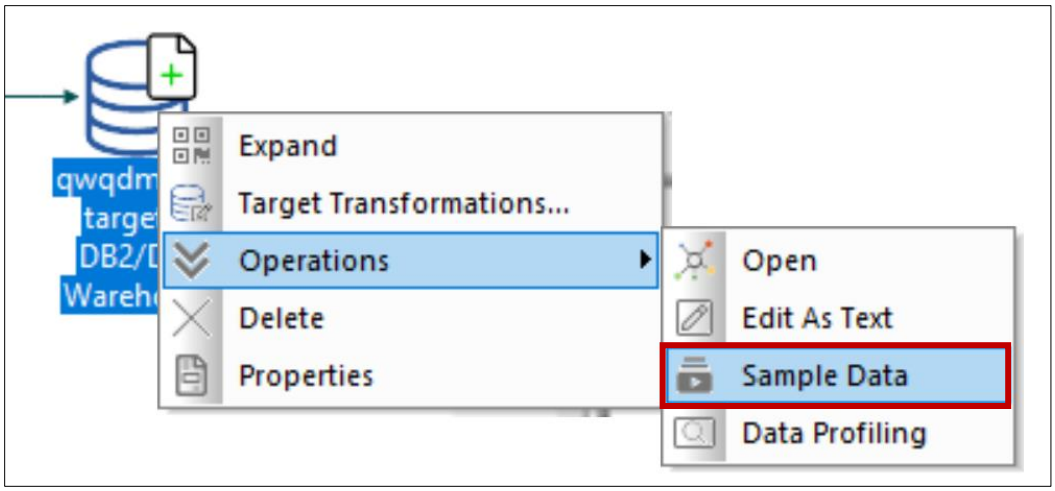


Figure 46 Viewing the data target using sample data

3.6 Saving a Flow

Save this data flow. Call it `flow01` and save it in the `qwqdmtest` folder.

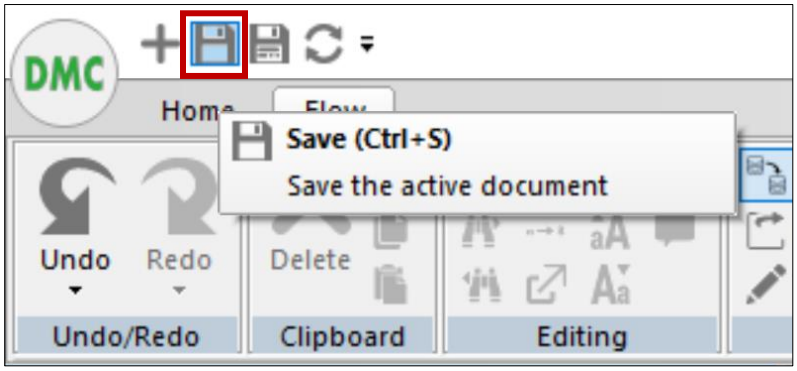


Figure 47 Saving the flow

You have successfully created your first flow! Now you can close the `flow01` tab.

4 A Deeper Look at Data Flows

In the previous chapter we created a simple data flow to illustrate the main components of a data flow. We will return to the data flow to create a more involved example.

4.1 Preparation

Before we get started on the flow, we must first create a table called `product_sold` in the library `qwqdmtest` that will be used later as a data target.

Using an SQL interface (Access Client Solutions, `STRSQL` CL command…) create the table as follows.

> **NOTE**: You may need to change to a system naming `qwqdmtest/product_sold` format instead of SQL naming `qwqdmtest.product_sold`.

```
CREATE TABLE qwqdmtest.product_sold
(PRODUCTNAME   CHAR(60),
 PRODUCTTYPE   CHAR(15),
 PRODUCTNUMBER CHAR(4),
 STORECODE     CHAR(6),
 SOLDDATE      DATE,
 QUANTITY      INT,
 REVENUE       DEC(12,2),
 COST          DEC(12,2));
```

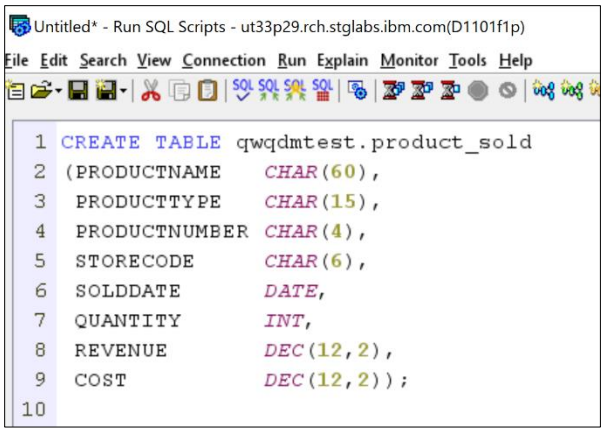


Figure 48 Creating a table using ACS

Once the table is created, create a synonym using the DMC. Right click the test folder and click `New`, then `Synonym`. Double click `*LOCAL` to specify the local adapter. Search only for the new table by typing `qwqdmtest` in the `Library` field and `product_sold` for the `Object Name` field. Use the prefix `flow_` . Select the table and click `Add`.

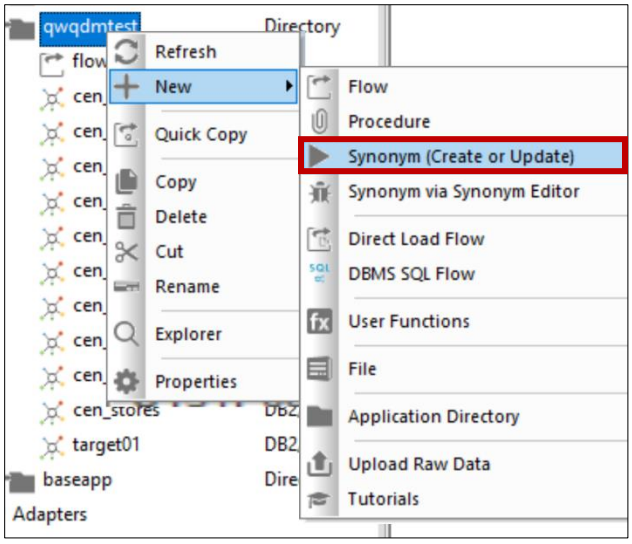


Figure 49 Creating a new synonym

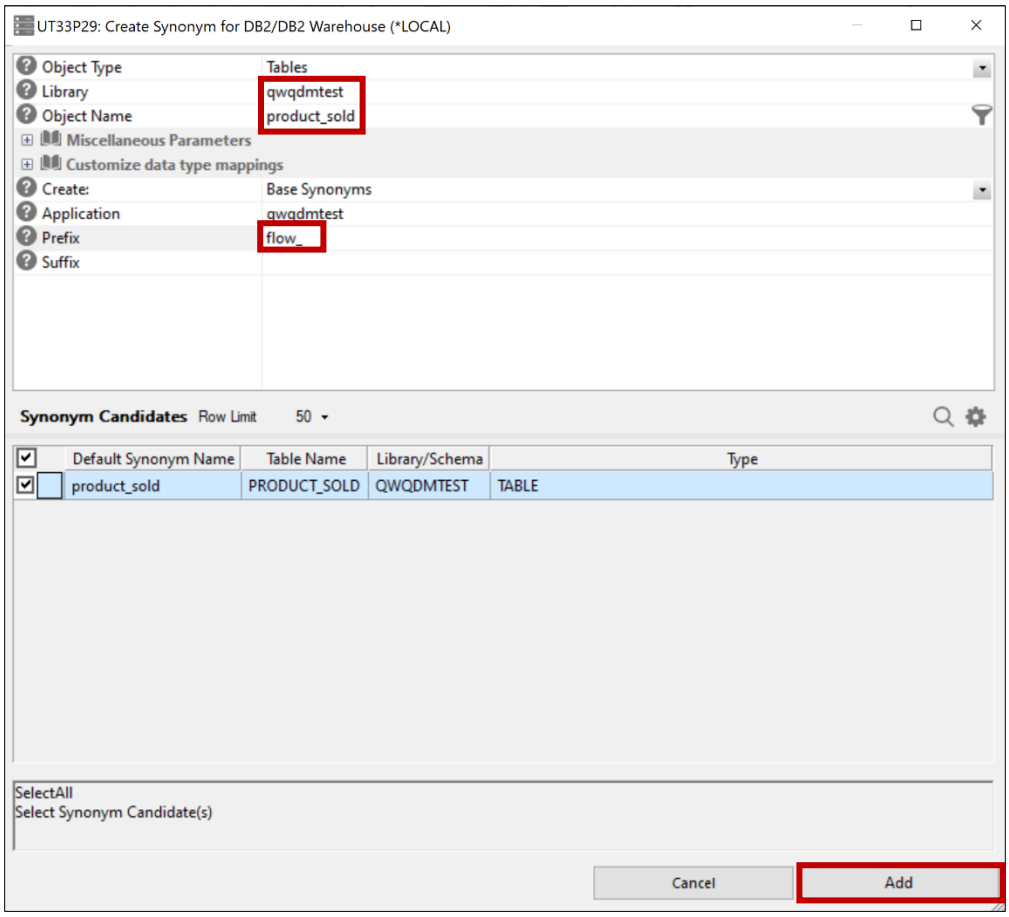


Figure 50 Creating the product_sold synonym

Now we will create the data flow.

4.2 Copying a Flow

While we could create a new flow, this is a good chance to try out the copy flow capability.

First, copy `flow01` by right clicking on the flow in the tree navigation and clicking `Copy`. To paste, right click on the folder and click `Paste`. Call the new flow `flow02`.

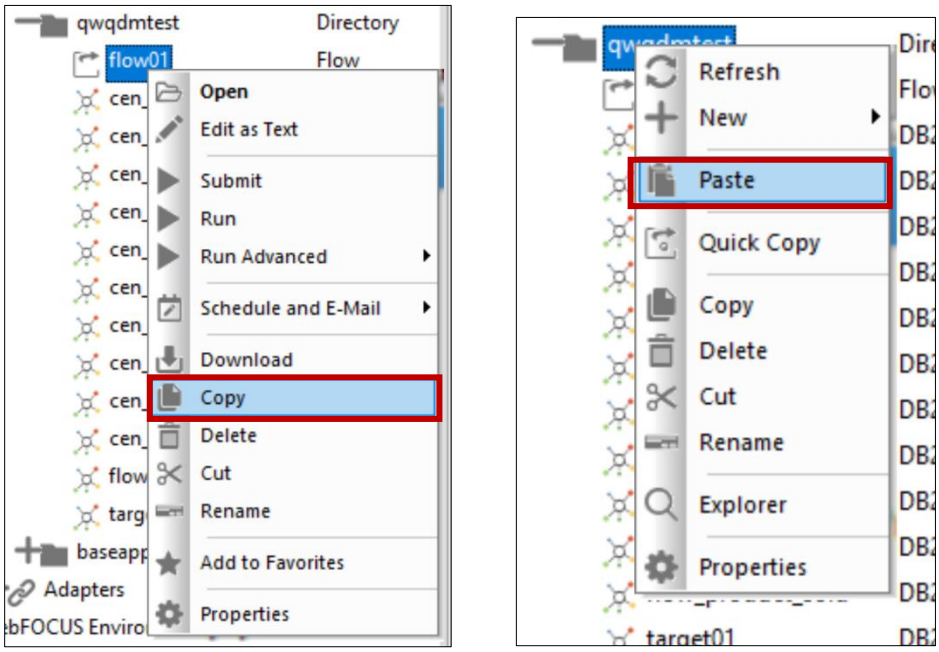


Figure 51 Copying and pasting to create a new flow

Open `flow02`. This flow starts from where we left off in the previous chapter. The `inventory` data source and target are shown.

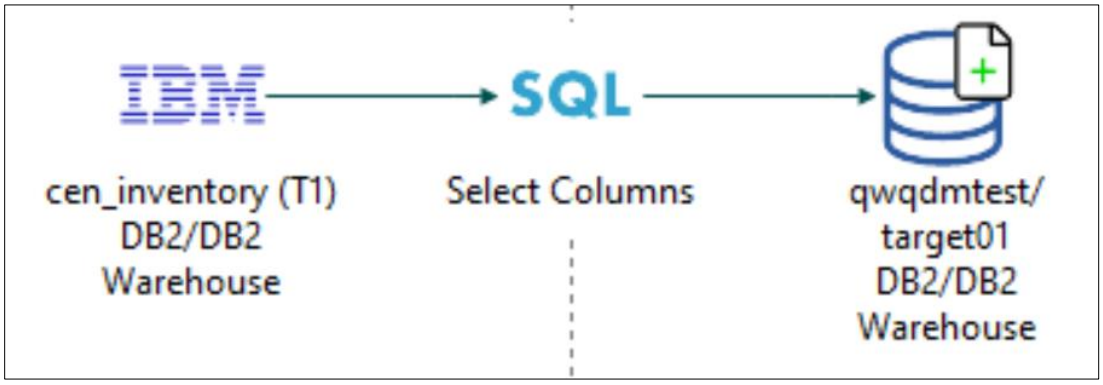


Figure 52 Starting point for flow02

4.3 Adding Joins and Transformations

Now drag in the data source `cen_orders`. Notice how DMC automatically applies a join connector between the inventory and orders sources.

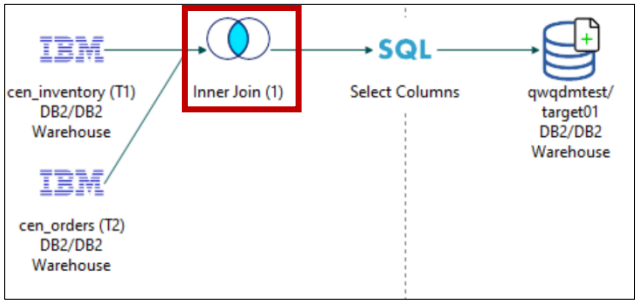


Figure 53 Adding the orders table

NOTE: If these connections are not appearing on your screen, go to the `Tools` menu and choose `Options`. Click the `Data Flow Designer` link and then click `Automatically add join conditions`.

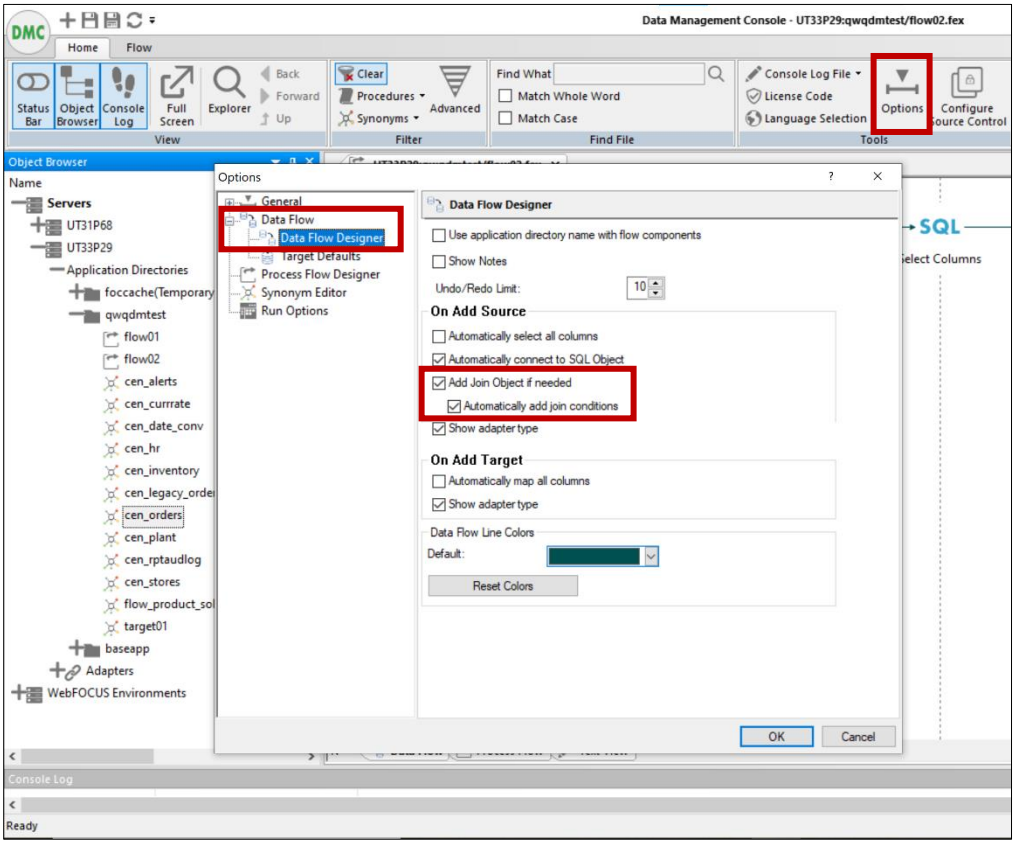
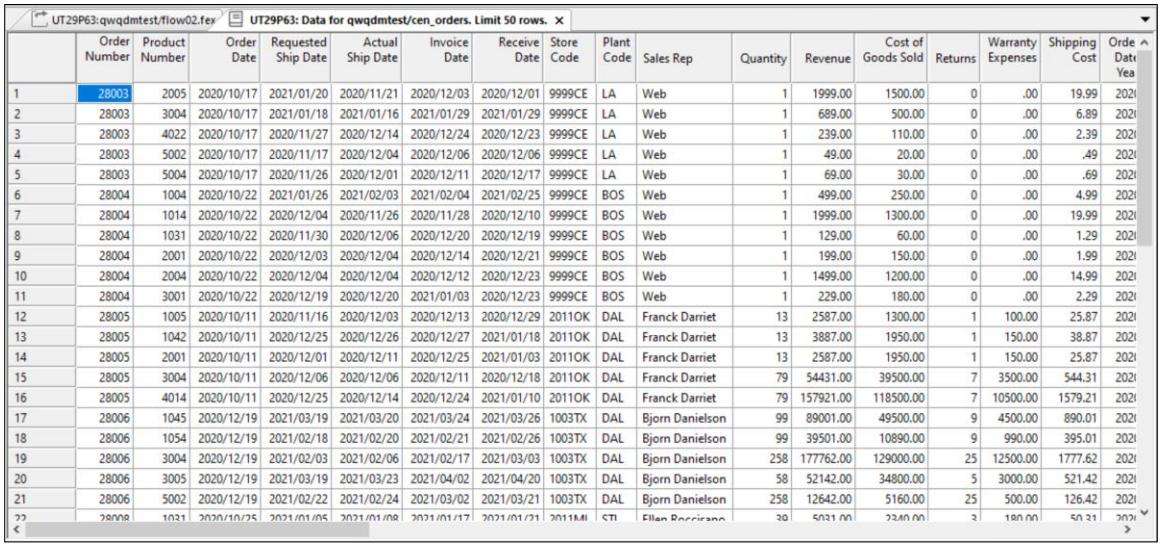


Figure 54 Data flow designer options from the Home tab

Verify the data in the `orders` table by right clicking the `cen_orders` source object and clicking `Operations`, then `Sample Data`.



UT29P63:qwqdmtest/flow02.fex | UT29P63: Data for qwqdmtest/cen_orders. Limit 50 rows.

	Order Number	Product Number	Order Date	Requested Ship Date	Actual Ship Date	Invoice Date	Receive Date	Store Code	Plant Code	Sales Rep	Quantity	Revenue	Cost of Goods Sold	Returns	Warranty Expenses	Shipping Cost	Orde Dat Yea
1	28003	2005	2020/10/17	2021/01/20	2020/11/21	2020/12/03	2020/12/01	9999CE	LA	Web	1	1999.00	1500.00	0	.00	19.99	202
2	28003	3004	2020/10/17	2021/01/18	2021/01/16	2021/01/29	2021/01/29	9999CE	LA	Web	1	689.00	500.00	0	.00	6.89	202
3	28003	4022	2020/10/17	2020/11/27	2020/12/14	2020/12/24	2020/12/23	9999CE	LA	Web	1	239.00	110.00	0	.00	2.39	202
4	28003	5002	2020/10/17	2020/11/17	2020/12/04	2020/12/06	2020/12/06	9999CE	LA	Web	1	49.00	20.00	0	.00	.49	202
5	28003	5004	2020/10/17	2020/11/26	2020/12/01	2020/12/11	2020/12/17	9999CE	LA	Web	1	69.00	30.00	0	.00	.69	202
6	28004	1004	2020/10/22	2021/01/26	2021/02/03	2021/02/04	2021/02/25	9999CE	BOS	Web	1	499.00	250.00	0	.00	4.99	202
7	28004	1014	2020/10/22	2020/12/04	2020/11/26	2020/11/28	2020/12/10	9999CE	BOS	Web	1	1999.00	1300.00	0	.00	19.99	202
8	28004	1031	2020/10/22	2020/11/30	2020/12/06	2020/12/20	2020/12/19	9999CE	BOS	Web	1	129.00	60.00	0	.00	1.29	202
9	28004	2001	2020/10/22	2020/12/03	2020/12/04	2020/12/14	2020/12/21	9999CE	BOS	Web	1	199.00	150.00	0	.00	1.99	202
10	28004	2004	2020/10/22	2020/12/04	2020/12/04	2020/12/12	2020/12/23	9999CE	BOS	Web	1	1499.00	1200.00	0	.00	14.99	202
11	28004	3001	2020/10/22	2020/12/19	2020/12/20	2021/01/03	2020/12/23	9999CE	BOS	Web	1	229.00	180.00	0	.00	2.29	202
12	28005	1005	2020/10/11	2020/11/16	2020/12/03	2020/12/13	2020/12/29	2011OK	DAL	Franck Darriet	13	2587.00	1300.00	1	100.00	25.87	202
13	28005	1042	2020/10/11	2020/12/25	2020/12/26	2020/12/27	2021/01/18	2011OK	DAL	Franck Darriet	13	3887.00	1950.00	1	150.00	38.87	202
14	28005	2001	2020/10/11	2020/12/01	2020/12/11	2020/12/25	2021/01/03	2011OK	DAL	Franck Darriet	13	2587.00	1950.00	1	150.00	25.87	202
15	28005	3004	2020/10/11	2020/12/06	2020/12/06	2020/12/11	2020/12/18	2011OK	DAL	Franck Darriet	79	54431.00	39500.00	7	3500.00	544.31	202
16	28005	4014	2020/10/11	2020/12/25	2020/12/14	2020/12/24	2021/01/10	2011OK	DAL	Franck Darriet	79	157921.00	118500.00	7	10500.00	1579.21	202
17	28006	1045	2020/12/19	2021/03/19	2021/03/20	2021/03/24	2021/03/26	1003TX	DAL	Bjorn Danielson	99	89001.00	49500.00	9	4500.00	890.01	202
18	28006	1054	2020/12/19	2021/02/18	2021/02/20	2021/02/21	2021/02/26	1003TX	DAL	Bjorn Danielson	99	39501.00	10890.00	9	990.00	395.01	202
19	28006	3004	2020/12/19	2021/02/03	2021/02/06	2021/02/17	2021/03/03	1003TX	DAL	Bjorn Danielson	258	177762.00	129000.00	25	12500.00	1777.62	202
20	28006	3005	2020/12/19	2021/03/19	2021/03/23	2021/04/02	2021/04/20	1003TX	DAL	Bjorn Danielson	58	52142.00	34800.00	5	3000.00	521.42	202
21	28006	5002	2020/12/19	2021/02/22	2021/02/24	2021/03/02	2021/03/21	1003TX	DAL	Bjorn Danielson	258	12642.00	5160.00	25	500.00	126.42	202
22	28008	1031	2020/10/25	2021/01/05	2021/01/08	2021/01/17	2021/01/21	2011MI	STL	Ellen Roccirano	39	5031.00	2340.00	3	180.00	50.31	202

Figure 55 Sample Data from cen_orders

> **NOTE:** You can change the number of rows to retrieve from the Tools group. On the `Home` tab, in the `Tools` group, click `Options`. From the Tools dialog box, click the `Run Options` link, and then change the number in `Maximum number of rows for test reports`.

Click the `X` in the upper-right corner to close the window and return to the object view.

Next, create a virtual column in the orders data source. This is also called a *Source Transformation*. Data transformations in a source object are performed when the records are read *before* any filtering or aggregation occurs.

The virtual column will compute the difference in days from when an order shipped to when it was requested to be shipped. Right click on the orders data source and choose `Source Transformations` to bring up the Source Transformations list. Choose `Insert Transforms`.

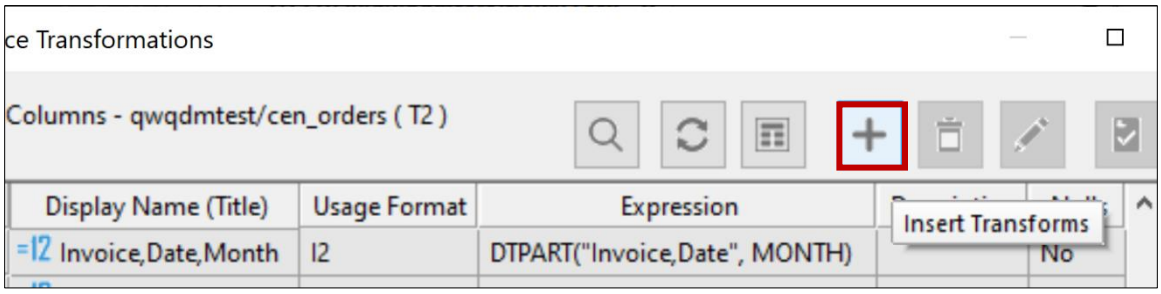


Figure 56 Creating a virtual column

The Transformation Calculator opens. In the `Name` box, type `SHIP_DIFFERENCE`. In the `Functions` tab, expand the `Date – Legacy` folder, double click on the `DATEDIF` function. This will bring up the Function Assist window. For the `start_date`, use the drop down to choose column `REQUESTED, SHIP DATE,Y-M-D (REQUESTEDSHIPDATE_YEAR_D)`. For the `end_date`, use the drop down to choose column `ACTUAL, SHIP DATE, Y-M-D (SHIPDATE_YEAR_D)`. Leave `component` as `'D'`. Click `OK`.

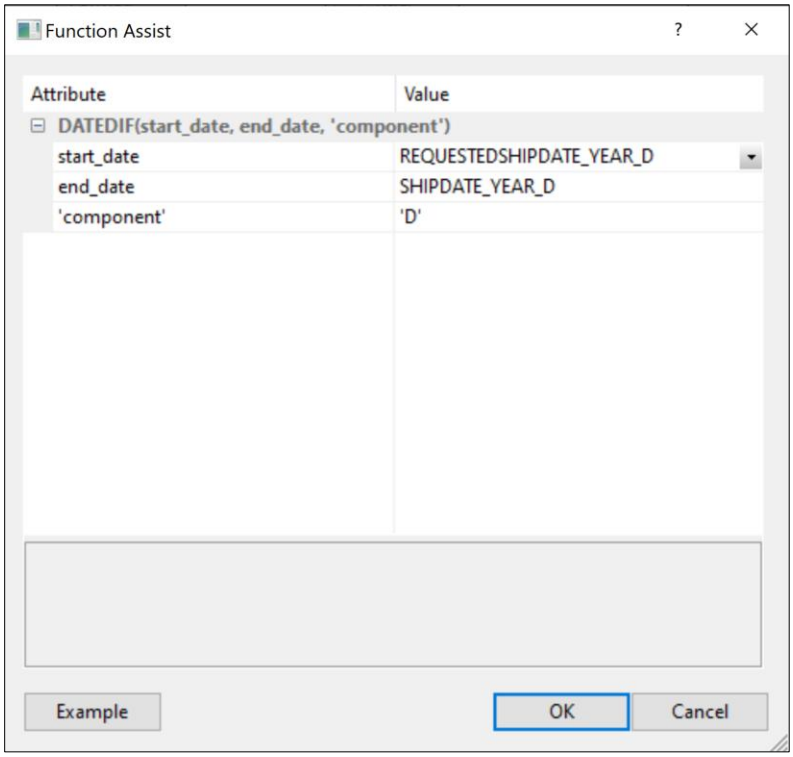


Figure 57 Function Assist window

The calculator shows the completed `DTDIFF` function.

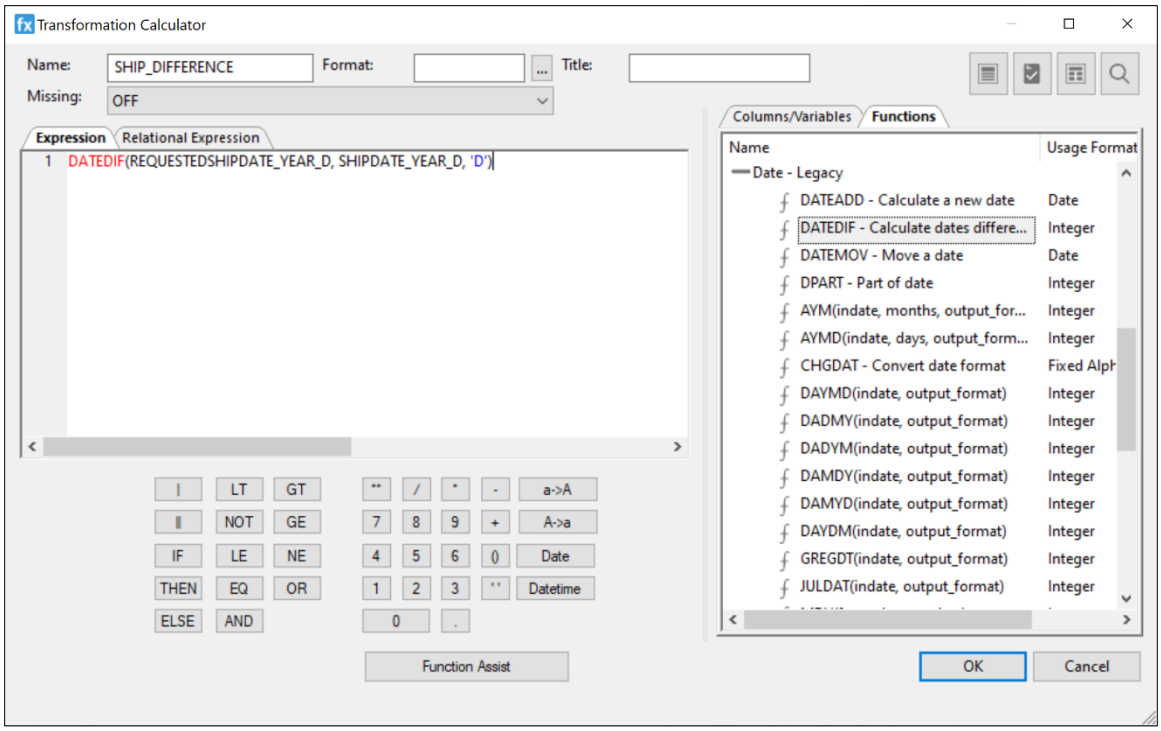


Figure 58 SHIP_DIFFERENCE virtual column definition

We are only interested in the absolute difference between the actual and the requested date. Add an `ABS` and a set of parentheses enclosing the `DATEDIF` function. Then click the `Sample Data` button on the upper right to make sure the expression is valid.

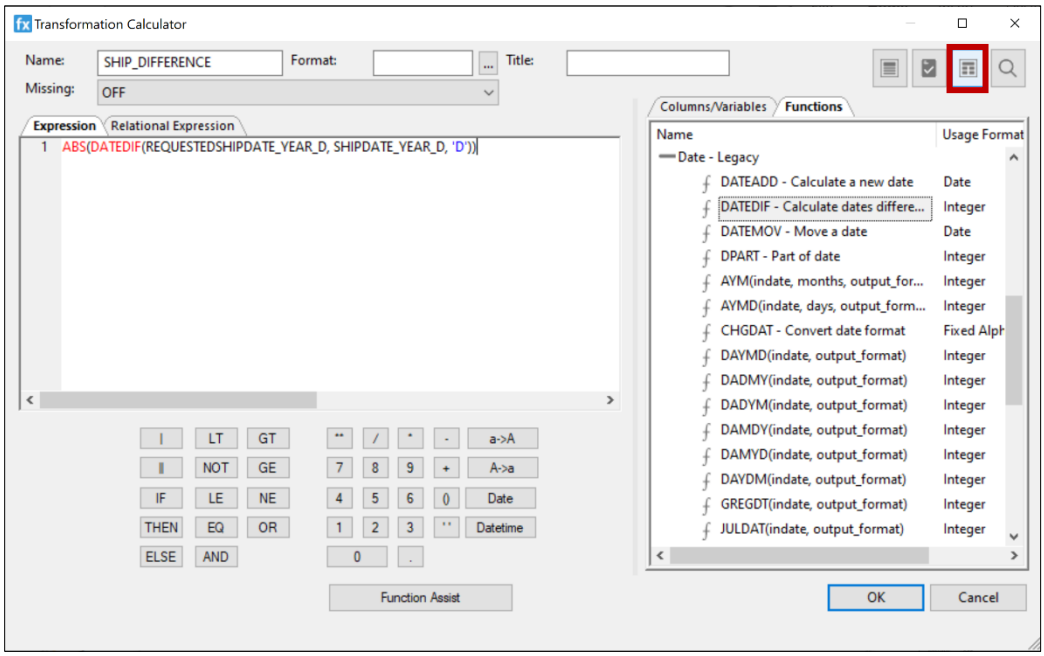
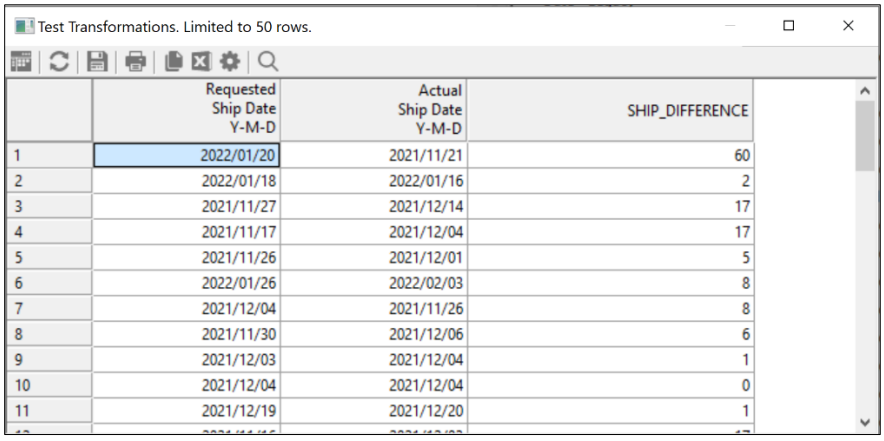


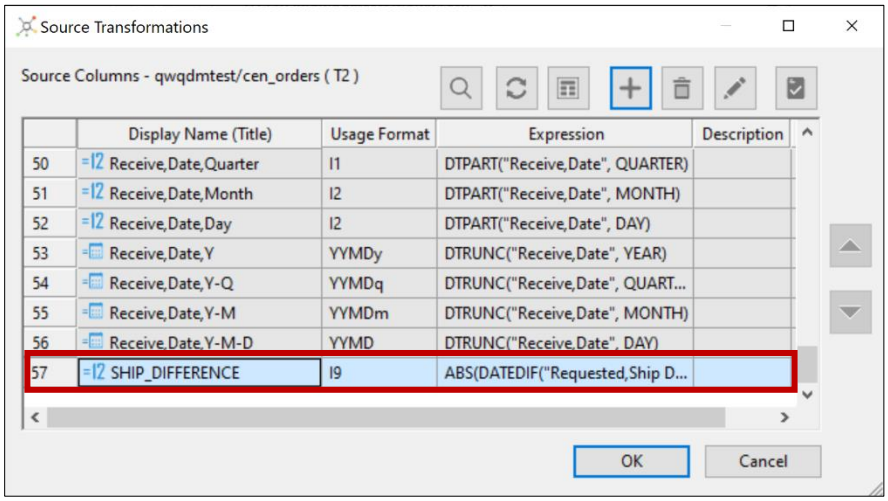
Figure 59 Adding the ABS function



	Requested Ship Date Y-M-D	Actual Ship Date Y-M-D	SHIP_DIFFERENCE
1	2022/01/20	2021/11/21	60
2	2022/01/18	2022/01/16	2
3	2021/11/27	2021/12/14	17
4	2021/11/17	2021/12/04	17
5	2021/11/26	2021/12/01	5
6	2022/01/26	2022/02/03	8
7	2021/12/04	2021/11/26	8
8	2021/11/30	2021/12/06	6
9	2021/12/03	2021/12/04	1
10	2021/12/04	2021/12/04	0
11	2021/12/19	2021/12/20	1

Figure 60 Sample data showing the calculated difference in ship dates

Close the sample data window, then click `OK` on the Transformation Calculator. `SHIP_DIFFERENCE` should now show up at the bottom of the Source Transformations list.



	Display Name (Title)	Usage Format	Expression	Description
50	Receive,Date,Quarter	I1	DTPART("Receive,Date", QUARTER)	
51	Receive,Date,Month	I2	DTPART("Receive,Date", MONTH)	
52	Receive,Date,Day	I2	DTPART("Receive,Date", DAY)	
53	Receive,Date,Y	YYMDy	DTRUNC("Receive,Date", YEAR)	
54	Receive,Date,Y-Q	YYMDq	DTRUNC("Receive,Date", QUART...	
55	Receive,Date,Y-M	YYMDm	DTRUNC("Receive,Date", MONTH)	
56	Receive,Date,Y-M-D	YYMD	DTRUNC("Receive,Date", DAY)	
57	SHIP_DIFFERENCE	I9	ABS(DATEDIF("Requested,Ship D...	

Figure 61 SHIP_DIFFERENCE in the source transformations list

> **NOTE**: Putting a virtual column in a synonym (rather than a flow) is a useful strategy when you expect to use the same synonym with more than one flow.

Click `OK` on the Source Transformation window.

Now consider the join object, which was added automatically when you selected the second data source. You will need to specify properties for the join.

By default, an inner join is created. An inner join extracts rows that appear in both tables. The join is based on an equality condition between two fields where one is in each data source. The use of an equality condition is also called an equi-join.

NOTE: DataMigrator supports multiple joins, joins based on conditions other than equalities, and joins that are modified by calculations, such as substrings or concatenations. A Join Calculator is available to assist you.

Right-click the join object and click `Join Editor`. The Join Editor window opens.

The join must be based on columns in each of the joined data sources. Notice that `Product Number` is in both Left and Right Source Columns lists. The join of `Product Number` between the data sources appears in the Expression field of the Join Conditions list. For our purposes, the default join on `Product Number` is sufficient.

NOTE: Again, if the join condition is not in effect, go to the `Tools` menu and choose `Options`. Click the `Data Flow Designer` link and then click `Automatically add join conditions`.

The inner join relationship is reflected in the Expression box. It is represented graphically by the overlapping area in the Join Type diagram, as shown in the following image.

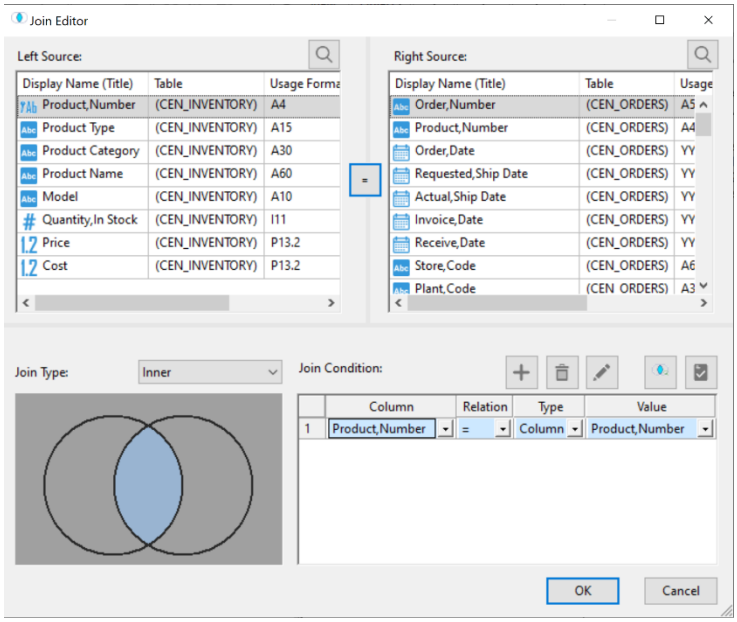


Figure 62 Inner join relationship on product number

NOTE: To change the type to a right or left outer join, click the left or right circle.

Click `OK` to close the Join Editor window.

4.4 Selecting Columns and Transformations

We are now ready to select the columns of data to load into the data targets. There are a variety of operations on the selected columns.

Right-click the `SQL` icon to see the options on the menu.

- *Expand* – Opens an information window.
- *SQL Statement* – Displays the SQL code. Right now, it reflects the join. It will be more interesting later.
- *WHERE Filter* – Provides a calculator where you can create an expression that limits record selection. For example, you might only want to retrieve records for a certain year.
- *HAVING Filter* – Provides a calculator where you can create an expression that limits retrieval based on aggregated values after a GROUP BY. This option only appears when the Column Selection includes a GROUP BY.
- *Sort* – *P*rovides a dialog box where you can control the order of data retrieval.
- *Column Selection* – Opens a window where you select the columns you want to include in your data target, and specify a variety of data retrieval requirements, which you will do in the following steps.
- *Add* – Allows you to add additional sources, selects, join, and union objects.
- *Properties* – Opens a property panel that shows statistics for the select statement.
- *Error Details* – This option only appears if there is an error in the columns or filters. When selected, it opens to explain the error.

Click the `Save` button to save the data flow up to this point.

> **NOTE:** The DMC options control whether the title, name, or description is shown for columns. If your screen does not show the column attribute you need or that is shown throughout this guide, change the option using the `Options` button from the main DMC screen under the `Home` tab.

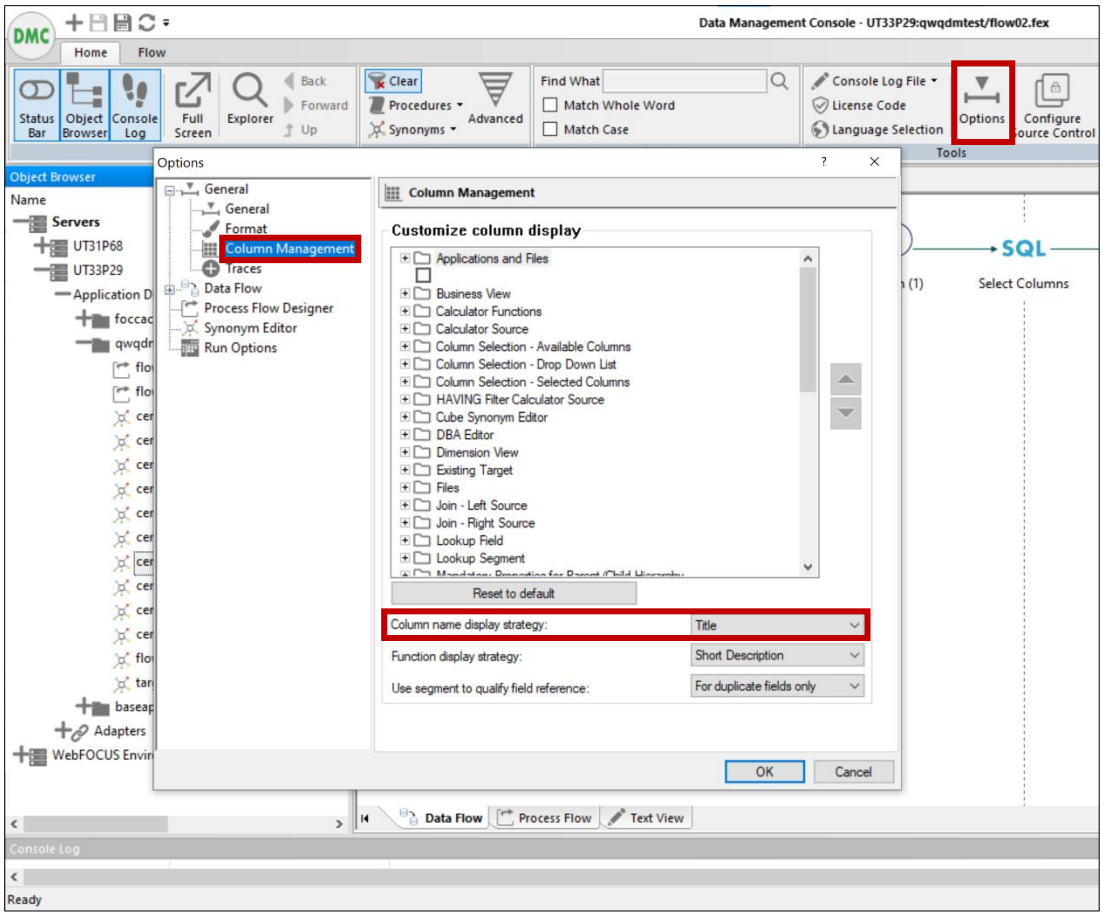


Figure 63 Changing column naming

Right-click the `SQL` icon and click `Column Selection`. Because this was a copy of the first data flow, the columns from inventory are already selected.

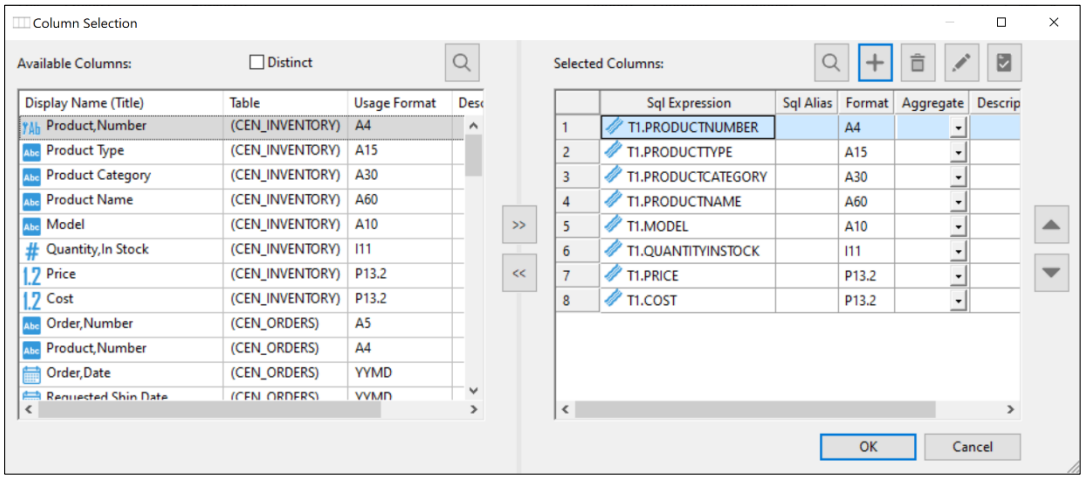


Figure 64 Initial column selections

We want to create a flow to track product and store information by month. We also want to track the average number of days that the actual ship date differs from the requested ship date.

We will select the following columns:

- PRODUCTNAME
- PRODUCTTYPE
- PRODUCTNUMBER
- STORECODE
- SHIPDATE
- QUANTITY
- REVENUE
- COSTOFGOODSSOLD

Using the >> and << buttons in the middle of the window, move columns into and out of the Selected Columns list. To move a column into the Selected Columns, highlight the column in the `Available Columns` list and click the >> button. To remove a `Selected Column`, highlight it and click the << button. Once the columns are added, move them around with the up and down arrows on the right until they are ordered as shown.

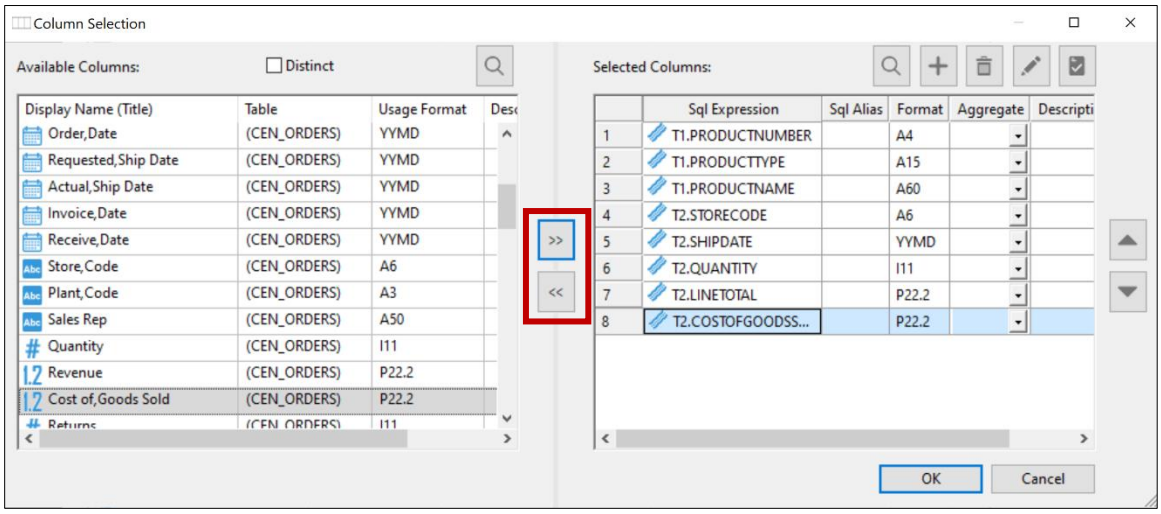


Figure 65 Selected columns

NOTE: Make sure to select *Quantity*, not Quantity in Stock!

Next, we want to calculate the profit.

This transformation can be done using a SQL calculation to create the column. To open the SQL Calculator, click the `Insert Columns` button above the Selected Columns list.

- In the Alias box, enter `PROFIT`.
- Verify that the Columns/Variables tab is selected. Then, under `CEN_ORDERS`, double click `Revenue` in the tree. The column appears in the Expression box.
- Click the subtraction sign (-) on the calculator keypad.
- Double-click `Cost Of, Goods Sold` in the tree to complete the expression.

The SQL Calculator should look like the figure below:

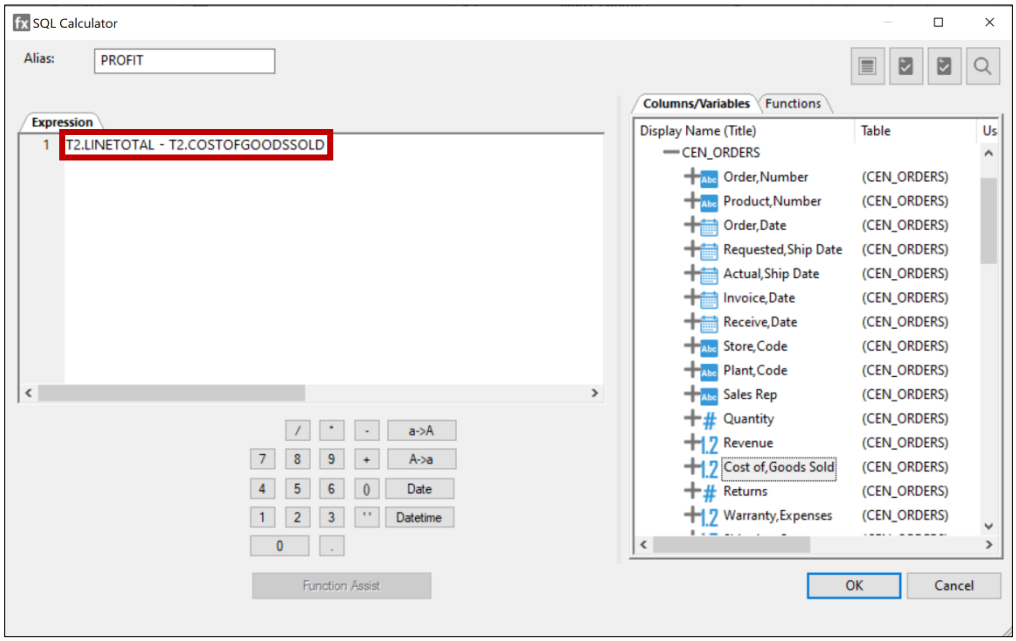


Figure 66 Computing the profit column

Click `OK` to close the SQL Calculator and return to the Column Selection window. Notice that the expression has been added to the bottom of the Selected Columns list. The expression is in the first column and the alias you assigned is in the second column.

Now we want to aggregate orders by date. We want to group by dimension columns and summarize measures. We would like total revenue, total cost, total profit, and average ship date difference by date. To do this, we must first aggregate on the key columns, and then on the year/month column. This is done by adding the `Group` attribute to each column.

Under Selected Columns, click `T1.PRODUCTNAME`, then under the Aggregate column select `Group By` from the dropdown menu. Repeat this step for `T1.PRODUCTTYPE`, `T1.PRODUCTNUMBER`, `T2.STORE_CODE`, and for `T2.SHIPDATE`.

Now multi-select the columns `T2.QUANTITY, T2.LINETOTAL, T2.COSTOFGOODSSOLD, and PROFIT` by pressing the Ctrl key and clicking each measure. Under the Aggregate column, select `Sum` from the dropdown menu.

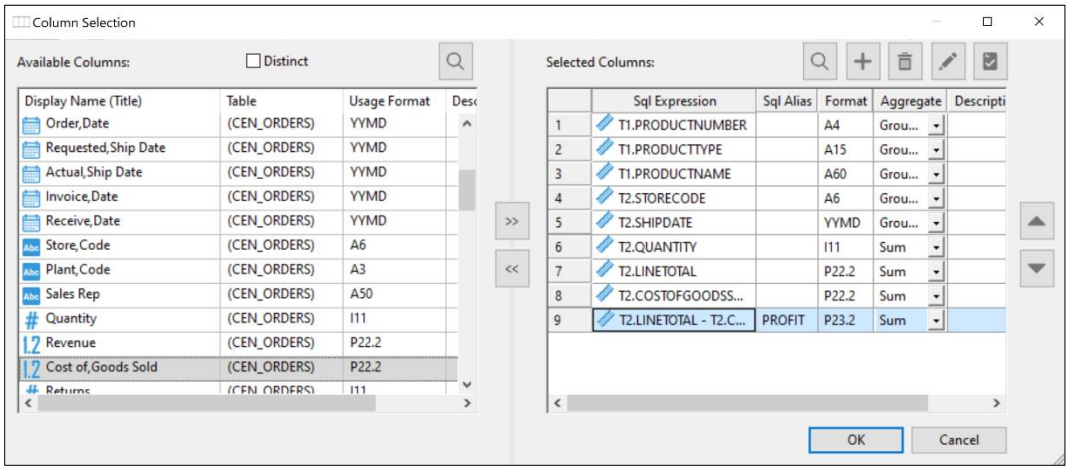


Figure 67 Aggregating the columns

As you clicked and aggregated each column down the list, you probably noticed that the remaining columns turned red. This was an indication that, at that point, the combination of selected columns was not valid. If one selected column is aggregated, all selected columns must be aggregated. Once all columns were aggregated, the red disappeared.

Notice that we missed one selected column, the average ship date difference. Go back and pull that in now. Unlike the other measures, we want the average, not sum. Pull in the `T2.SHIP_DIFFERENCE` column and select `Avg` from the dropdown. Now we have all the columns we want.

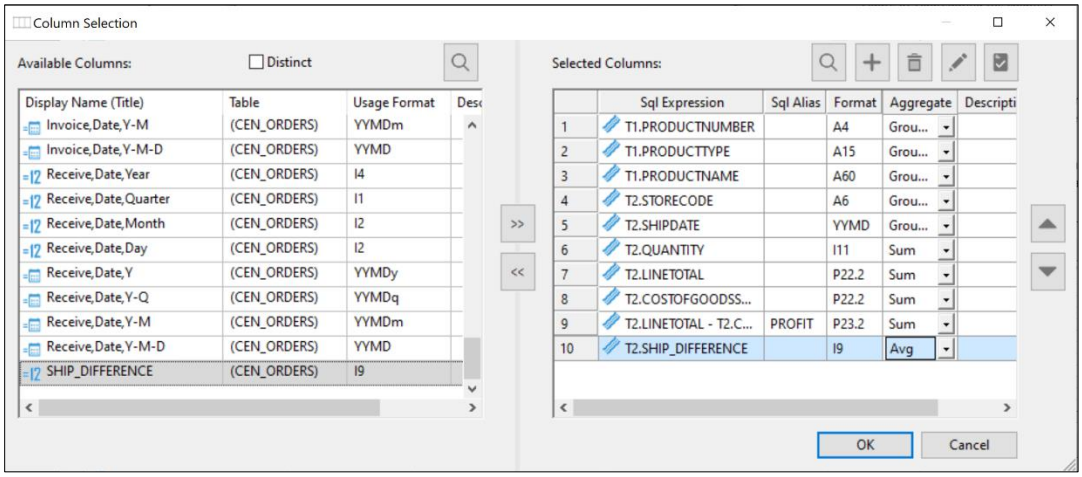


Figure 68 Aggregated columns and SHIP_DIFFERENCE

Click `OK` to close the Column Selection window and return to the object view.

We have identified the columns to extract from the source data, but the data in the source goes back several years and only the last few years are interesting. Therefore, we need to define a selection criterion to limit the retrieval to the appropriate years.

- Right-click the `SQL` object again, and this time click `WHERE Filter`. The WHERE Filter Calculator opens to help you construct the expression. Notice that this calculator is suitable for creating a wide range of selection criteria.
- As you can see, the columns here are represented by the same icons used in the Selected Columns dialog box. You can filter on real or virtual columns.
- To limit record retrieval to the time period beginning in the year 2013:
    a. Double-click `Actual Ship Date` under `cen_orders` in the Column list. `Actual Ship Date` is displayed in the Expression box.
    b. Click >= (greater than or equal to) in the calculator pad below the Expression window.
    c. Under the Type column, choose `Value` .
    d. Type in a date, that is type '2013-01-01' including the quotes.
- The expression in the WHERE Filter Calculator should look like the following figure:

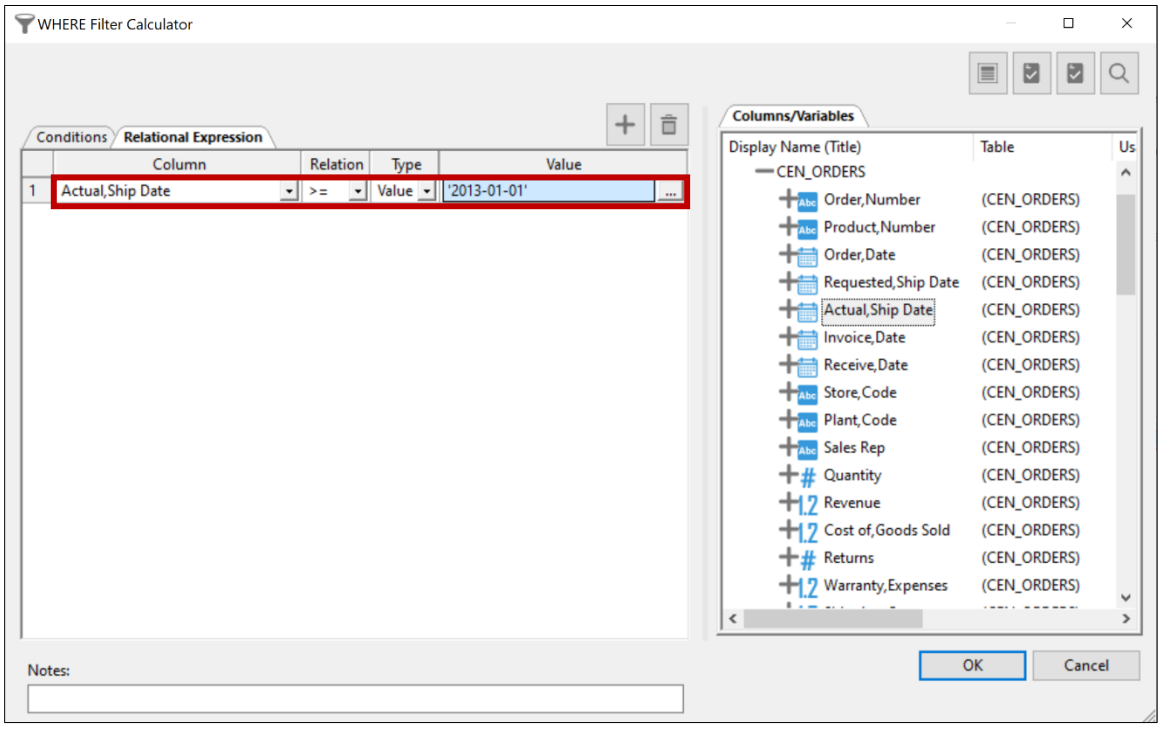


Figure 69 Creating a WHERE filter

NOTE: Though it is not required for this guide, you can build a SQL calculation using any ANSI SQL function. Click the `Functions` tab to see the available functions and arguments.

Click `OK` to complete the filtering expression. Right-click the `SQL` object and click `SQL Statement` to see the query that will execute.

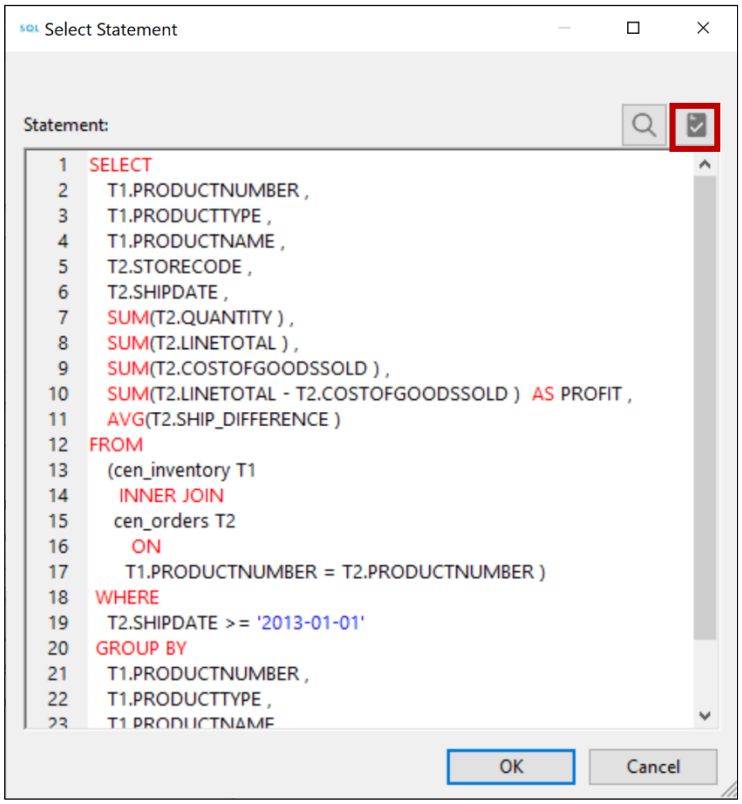
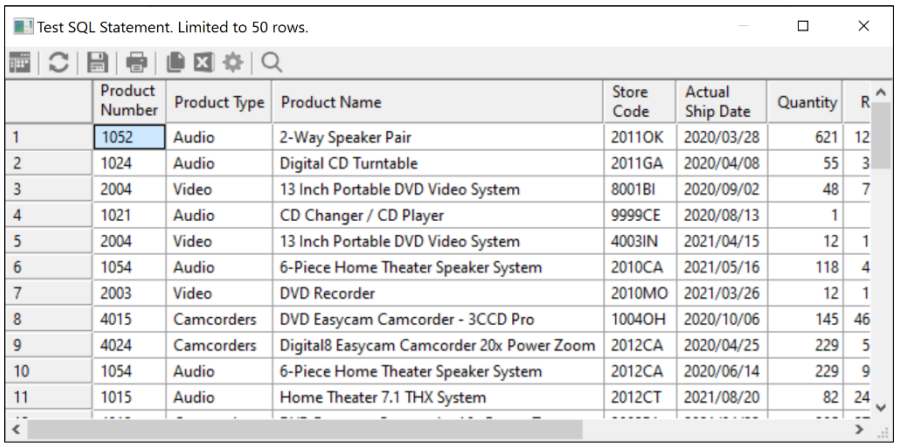


Figure 70 SQL statement

Click the `Test SQL Statement` button to see the results.



	Product Number	Product Type	Product Name	Store Code	Actual Ship Date	Quantity	R
1	1052	Audio	2-Way Speaker Pair	2011OK	2020/03/28	621	12
2	1024	Audio	Digital CD Turntable	2011GA	2020/04/08	55	3
3	2004	Video	13 Inch Portable DVD Video System	8001BI	2020/09/02	48	7
4	1021	Audio	CD Changer / CD Player	9999CE	2020/08/13	1	
5	2004	Video	13 Inch Portable DVD Video System	4003IN	2021/04/15	12	1
6	1054	Audio	6-Piece Home Theater Speaker System	2010CA	2021/05/16	118	4
7	2003	Video	DVD Recorder	2010MO	2021/03/26	12	1
8	4015	Camcorders	DVD Easycam Camcorder - 3CCD Pro	1004OH	2020/10/06	145	46
9	4024	Camcorders	Digital8 Easycam Camcorder 20x Power Zoom	2012CA	2020/04/25	229	5
10	1054	Audio	6-Piece Home Theater Speaker System	2012CA	2020/06/14	229	9
11	1015	Audio	Home Theater 7.1 THX System	2012CT	2021/08/20	82	24

Figure 71 Testing the SQL statement

> **NOTE:** The number of records retrieved will depend on the `Run Options` set in `Options` on the `Home` tab.

Click the `X` to close the Test SQL Statement window and click `OK` to close the Select Statement window. Click the `Save` button to save the data flow up to this point.

4.5 Adding Data Targets

We are ready to create the data targets where the source data will be copied, based on the mapping and rules defined in the SQL columns object.

First, we will get rid of the data target brought over when we copied the flow. Highlight the data target and press delete. Click `Yes` on the confirmation screen.

> **NOTE:** A very handy feature of the editor is that you can undo changes. If you delete something you wanted to keep, click the `Undo` button on the top left of the DMC window or hit the `Ctrl-z` key sequence.

At the beginning of this chapter, we created a data target table that we will use now. We are going to add two data targets and specify the options to use when loading data into them.

- The first data target is a pre-existing table that we will add data to. This represents a common situation where a data flow will add data into an existing data warehouse or data mart.
- The second is a target table that we will create as part of the data flow. It represents a transient table used for a fixed period of time and then ultimately discarded.

To specify the first data target:

1. From the navigation pane, drag the synonym `flow_product_sold` into the workspace, to the right of the SQL object. (The position to the right of the SQL object makes it a data target).
2. Once you have added the data target to the data flow, you can specify how incoming data should be handled during the loading process.

Right-click the `flow_product_sold` target object and click `Properties`. For most of the properties we can stay with the defaults. However, for `Load Type` specify `Insert Records From Memory`.

> **NOTE**: Using the Load Type `Insert Records From Memory` is a good habit to get into for DataMigrator for i, as it generally has the best performance characteristics for 'bulk' load scenarios such as this.

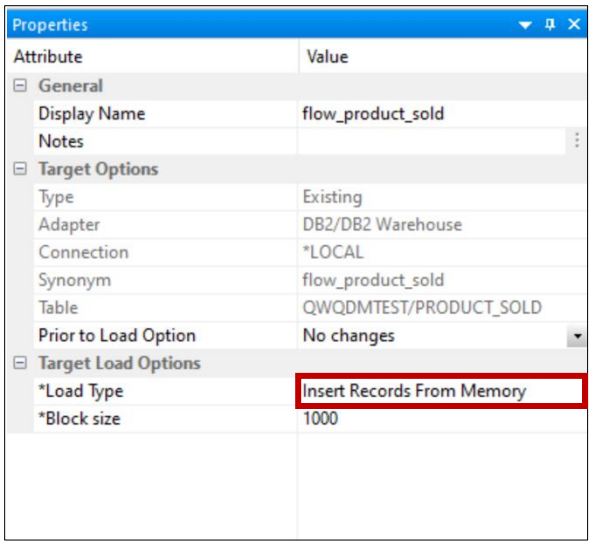


Figure 72 Properties for the existing data target

NOTE: The properties allow you to clear the target table before each run. In the `Prior to Load Option`, choose `Delete all rows from table`.

Click `X` to close the Properties window. The next step is to map the data source columns to the data target columns.

1. Right-click `flow_product_sold` and click `Target Transformations`. The Transformations window opens.
2. Click the `Automap` button.

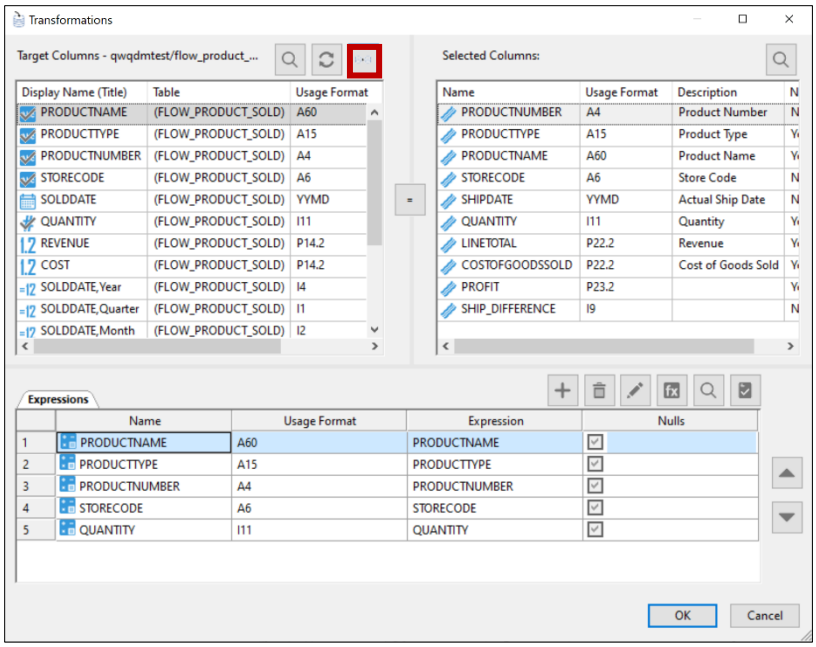


Figure 73 Target transformation using automap

The five columns with identical names and data types are mapped and moved to the Expressions tab.

Three target column names are deselected, so you need to create mappings or transformations for them. In the Target Columns list, `SOLDDATE` contains sold date. You will map it to the `SHIPDATE` column in the Selected Columns list.

1. Under Target Columns, click `SOLDDATE`.
2. Under Selected Columns, click `SHIPDATE`.
3. Click the equal sign (=) to move the mapping into the grid in the Expressions tab.

Repeat these steps for `REVENUE` to `LINE TOTAL` and `COST` to `COSTOFGOODSSOLD`.

> **NOTE:** If we needed to do any calculations or expressions, we would use the `Insert Transformations` button to create the transformation(s), like what we did when we created the `PROFIT` source transformation earlier. If we did the calculation here it would be called a Target Transformation.

4.6 Adding Validation

In 2.3.0, the validates tab is unavailable when the flow property 'optimize load' is checked. You can add validation to a column if you uncheck the optimize load property, however this will likely degrade the performance of your flow. Since we are not working with a large dataset in this guide, we will go ahead and uncheck the property.

Click `OK` to exit from the Transformation window. Then right click anywhere on the palette and click `Flow Properties`.

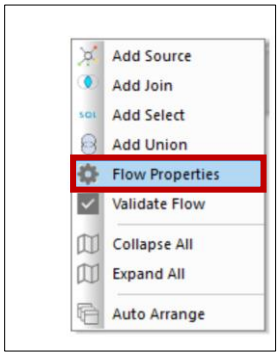


Figure 74 Accessing flow properties

Now uncheck the box next to `Optimize Load`. You will see more flow properties become available.

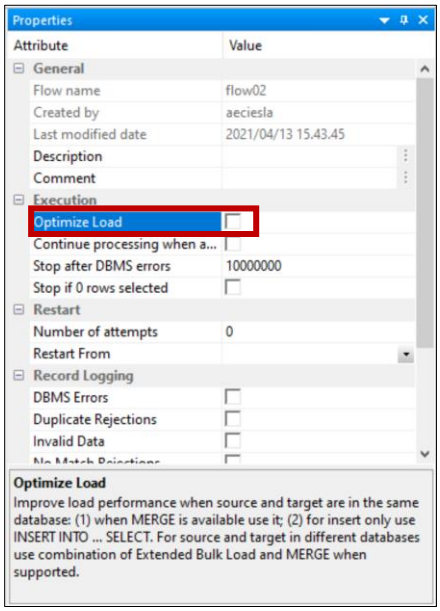


Figure 75 Unchecking optimize load

Now we will add some validation to the incoming data. To do that, we want to load only those records with a quantity greater than or equal to ten. Records that do not meet this validation criterion can be logged to a file, for additional processing, or for review later.

Right-click `flow_product_sold` and click `Target Transformations` to reopen the Transformation window. Click the `Validates` tab in the Transformation window. Then click the `Insert Intermediate Transformations` button to open the Transformation Calculator. To build the validation expression, double-click `QUANTITY` in the tree, and then select `GE` from the Relation drop-down menu. Input `10` into the Value field. Click `OK`.

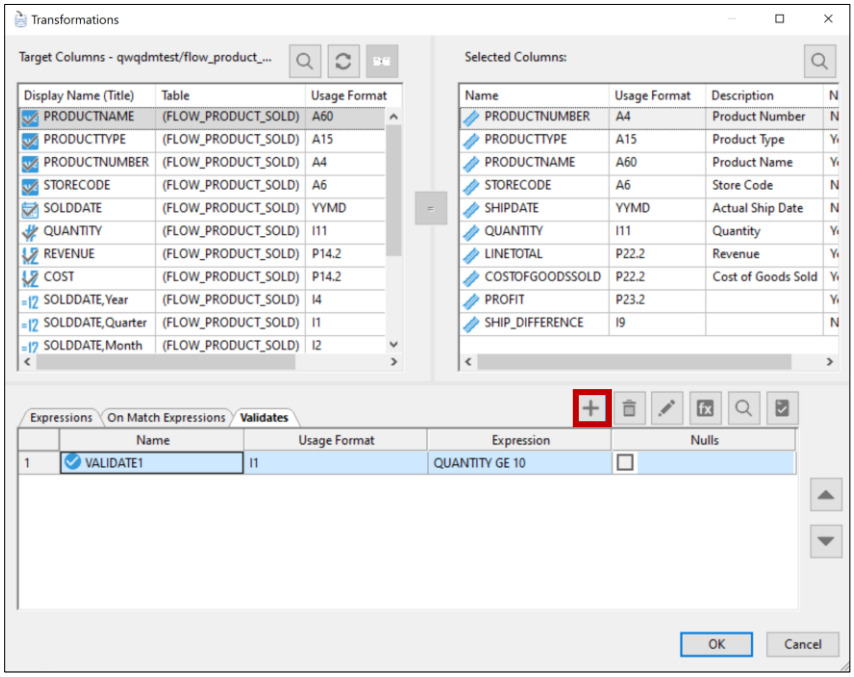
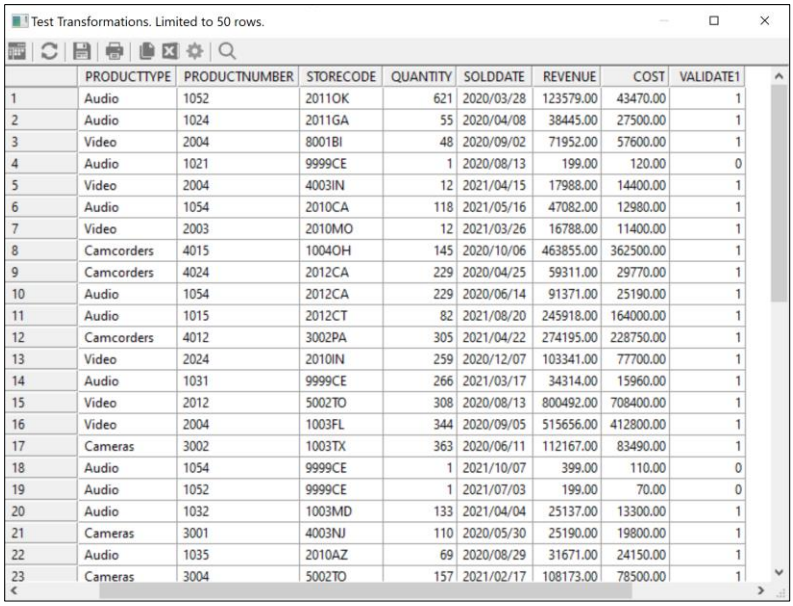


Figure 76 Adding validation

We can test the validation and transformations to ensure that they are syntactically correct and performing the desired calculations. The test retrieves some rows from the server, applies the transformations, and displays the results. Click the `Test Transforms` button in the upper-right corner.



	PRODUCTTYPE	PRODUCTNUMBER	STORECODE	QUANTITY	SOLDDATE	REVENUE	COST	VALIDATE1
1	Audio	1052	2011OK	621	2020/03/28	123579.00	43470.00	1
2	Audio	1024	2011GA	55	2020/04/08	38445.00	27500.00	1
3	Video	2004	8001BI	48	2020/09/02	71952.00	57600.00	1
4	Audio	1021	9999CE	1	2020/08/13	199.00	120.00	0
5	Video	2004	4003IN	12	2021/04/15	17988.00	14400.00	1
6	Audio	1054	2010CA	118	2021/05/16	47082.00	12980.00	1
7	Video	2003	2010MO	12	2021/03/26	16788.00	11400.00	1
8	Camcorders	4015	1004OH	145	2020/10/06	463855.00	362500.00	1
9	Camcorders	4024	2012CA	229	2020/04/25	59311.00	29770.00	1
10	Audio	1054	2012CA	229	2020/06/14	91371.00	25190.00	1
11	Audio	1015	2012CT	82	2021/08/20	245918.00	164000.00	1
12	Camcorders	4012	3002PA	305	2021/04/22	274195.00	228750.00	1
13	Video	2024	2010IN	259	2020/12/07	103341.00	77700.00	1
14	Audio	1031	9999CE	266	2021/03/17	34314.00	15960.00	1
15	Video	2012	5002TO	308	2020/08/13	800492.00	708400.00	1
16	Video	2004	1003FL	344	2020/09/05	515656.00	412800.00	1
17	Cameras	3002	1003TX	363	2020/06/11	112167.00	83490.00	1
18	Audio	1054	9999CE	1	2021/10/07	399.00	110.00	0
19	Audio	1052	9999CE	1	2021/07/03	199.00	70.00	0
20	Audio	1032	1003MD	133	2021/04/04	25137.00	13300.00	1
21	Cameras	3001	4003NJ	110	2020/05/30	25190.00	19800.00	1
22	Audio	1035	2010AZ	69	2020/08/29	31671.00	24150.00	1
23	Cameras	3004	5002TO	157	2021/02/17	108173.00	78500.00	1

Figure 77 Test transformation results

In the `VALIDATE1` column, the 1s represent rows that will be accepted and the 0s represent rows that will be rejected based on whether `QUANTITY` is greater than or equal to 10.

Close the Test Transformation window and click `OK` to close the Transformations window. Click the `Save` button to save the data flow up to this point.

4.7 Adding a Second Data Target

DataMigrator can load multiple data targets in a single data flow. We will add another target object into the data flow. This time, the data target does not exist, so we will create it using the columns in the `SQL Select` statement that were defined for the `SQL Select Columns` object.

> **NOTE:** This flow will create the base table, but other flows can update it with additional information.

Right click in the workspace to the right of the SQL object, select `Add target`, then click `New`. A new data target appears to the right of the SQL object.

Right-click the new target and click `Properties`. The Target Properties window opens.

Some of the properties are prefilled with the defaults we want. The adapter is Db2 (for Db2 for i). The connection is *LOCAL, which means the resulting database table will be put back on the same system where DataMigrator is running.

For synonym specify `qwqdmtest/flow_new_prod`, which will give the name of the synonym `flow_new_prod` and put it the `qwqdmtest` folder. For the table specify `qwqdmtest/flow_new_prod` to make sure the table, named `flow_new_prod`, goes into library `qwqdmtest`. For load type specify `Insert Records From Memory`.

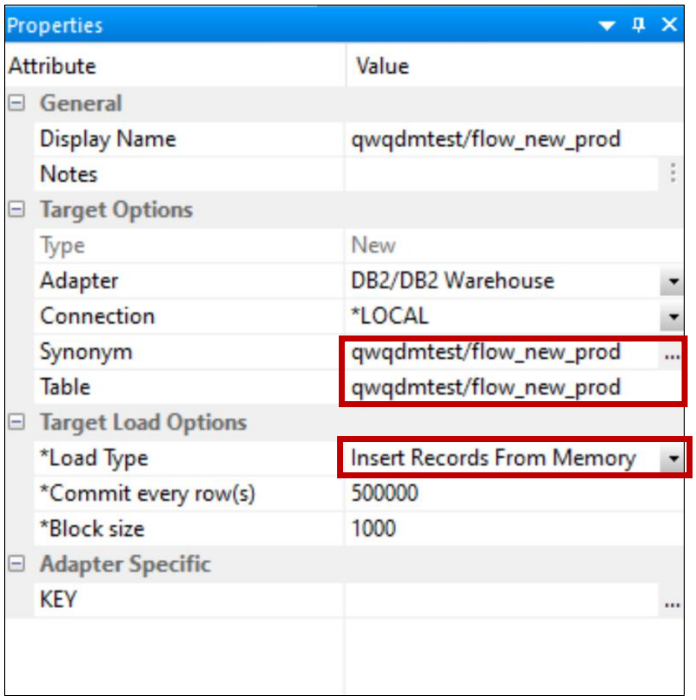


Figure 78 Properties for the new data target

The properties are now set correctly.

> **NOTE**: The properties options identified with an asterisk (*) are flow wide properties. That means they must be the same for all data targets. If the options are not consistent you will get a warning when you attempt to save or run the flow as seen below.

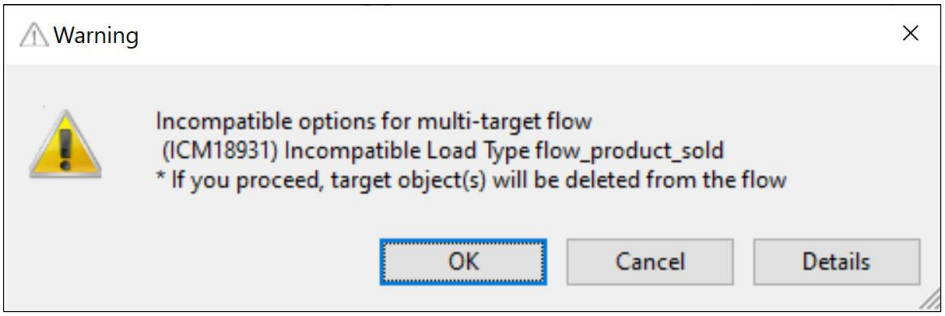


Figure 79 Warning message displayed when flow wide attributes are not in sync

Close the properties by clicking the `X` on the properties window. Then right-click the new data target and click `Target Transformations`.

In the table you are creating, `PRODUCTNAME`, `PRODUCTTYPE`, `PRODUCTNUMBER`, `STORECODE` and `SHIPDATE` are all keys. This means that each resulting row has a unique combination of those columns. Identifying the keys allows DataMigrator to add database enforcement (unique index) to the underlying table and to utilize uniqueness in subsequent change data processing. While not required, identifying keys of the target helps to ensure data integrity. Click the checkbox next to each of these columns to identify them as part of the key.

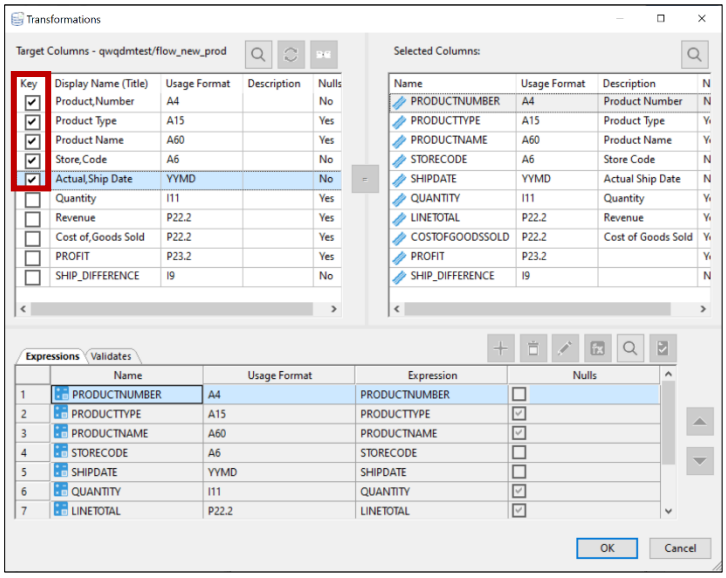


Figure 80 Specifying keys on target transformations

Click `OK` to return to the Data Flow tab, which now contains two data sources and two data targets.

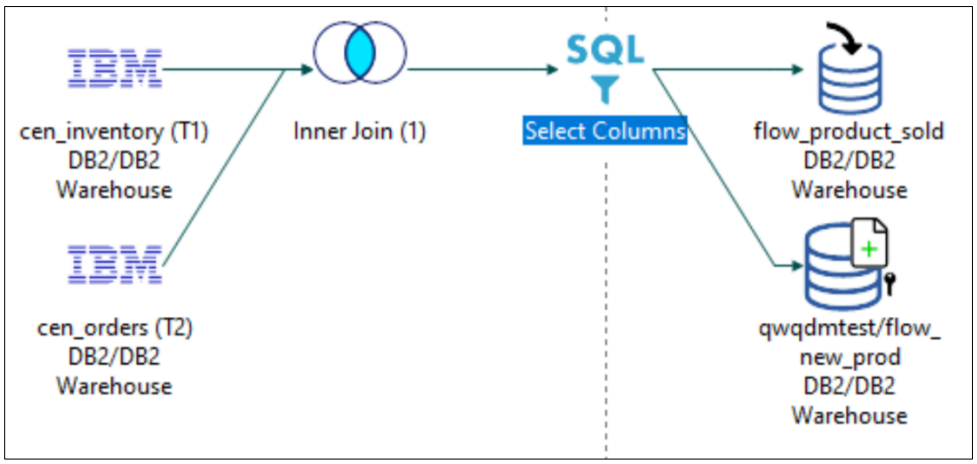


Figure 81 Completed data flow with two targets

Click the `Save` button to save the data flow.

4.8 Running the Flow

The data flow is ready to be run. Both data targets `flow_new_prod` and `flow_product_sold` will be loaded with data based on the data flow specifications.

A flow can be run in two ways:

1. In the left navigation tree, find `flow02` under the `qwqdmtest` folder. Right click on `flow02` and click `Submit`.
2. In the ribbon, click `Run` and then click `Submit` from the dropdown menu.

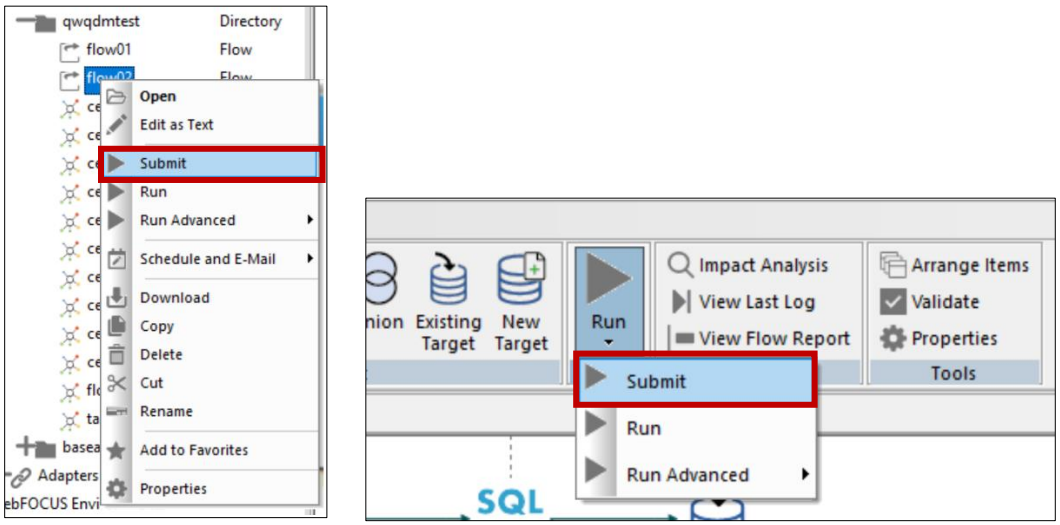


Figure 82 Submitting the flow

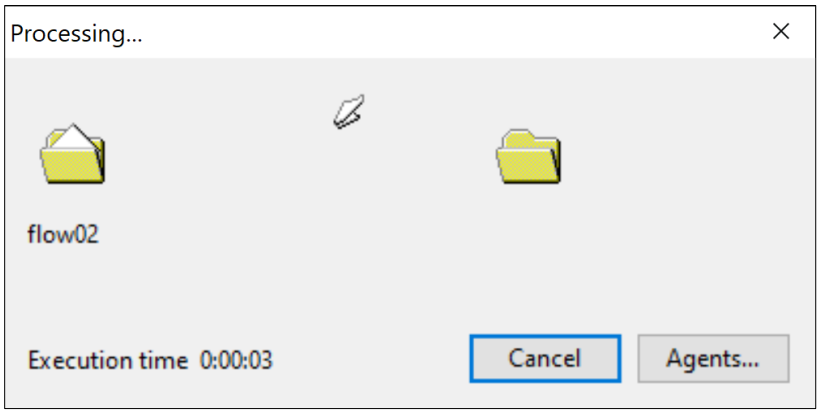


Figure 83 Flow execution

When the execution completes, you can review the console log at the bottom of the DMC window.

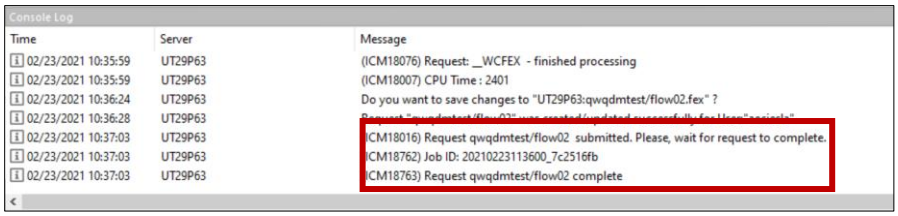


Figure 84 Console log output

When a flow is executed using `Submit` rather than `Run`, the flow is run in a background process. Consequently, the console log only shows that the request was submitted and that it was successful. To get more details on the process itself, we need to look at the run log from the process. The log is also important for analyzing any problems that may have occurred in the flow run.

> **NOTE:** In general, you should use `Run` when you are building and testing a flow. This will cause the flow to run in your DMC session and all log details will show in the console log. Once a flow is finished, you can use `Submit` to run it in the background and free the DMC session for other work.

To view the run log, click on the `View Last Log` option in the ribbon.

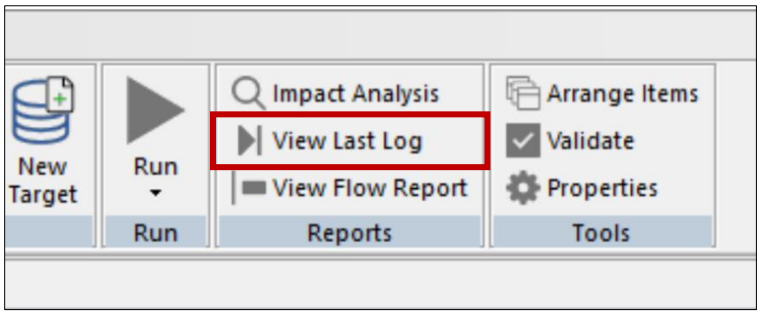
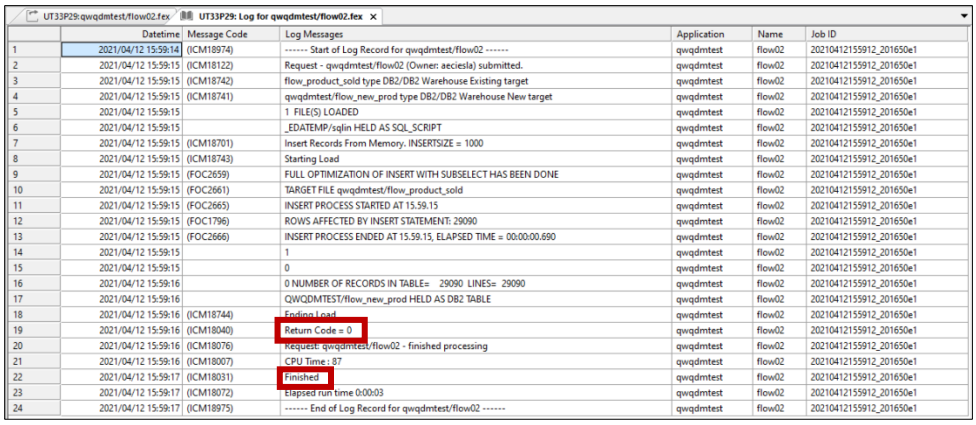


Figure 85 View the last log



	Datetime	Message Code	Log Messages	Application	Name	Job ID
1	2021/04/12 15:59:14	(ICM18974)	------ Start of Log Record for qwqdmtest/flow02 ------	qwqdmtest	flow02	20210412155912_201650e1
2	2021/04/12 15:59:15	(ICM18122)	Request - qwqdmtest/flow02 (Owner: aeciesla) submitted.	qwqdmtest	flow02	20210412155912_201650e1
3	2021/04/12 15:59:15	(ICM18742)	flow_product_sold type DB2/DB2 Warehouse Existing target	qwqdmtest	flow02	20210412155912_201650e1
4	2021/04/12 15:59:15	(ICM18741)	qwqdmtest/flow_new_prod type DB2/DB2 Warehouse New target	qwqdmtest	flow02	20210412155912_201650e1
5	2021/04/12 15:59:15		1 FILE(S) LOADED	qwqdmtest	flow02	20210412155912_201650e1
6	2021/04/12 15:59:15		_EDATEMP/sqlin HELD AS SQL_SCRIPT	qwqdmtest	flow02	20210412155912_201650e1
7	2021/04/12 15:59:15	(ICM18701)	Insert Records From Memory. INSERTSIZE = 1000	qwqdmtest	flow02	20210412155912_201650e1
8	2021/04/12 15:59:15	(ICM18743)	Starting Load	qwqdmtest	flow02	20210412155912_201650e1
9	2021/04/12 15:59:15	(FOC2659)	FULL OPTIMIZATION OF INSERT WITH SUBSELECT HAS BEEN DONE	qwqdmtest	flow02	20210412155912_201650e1
10	2021/04/12 15:59:15	(FOC2661)	TARGET FILE qwqdmtest/flow_product_sold	qwqdmtest	flow02	20210412155912_201650e1
11	2021/04/12 15:59:15	(FOC2665)	INSERT PROCESS STARTED AT 15.59.15	qwqdmtest	flow02	20210412155912_201650e1
12	2021/04/12 15:59:15	(FOC1796)	ROWS AFFECTED BY INSERT STATEMENT: 29090	qwqdmtest	flow02	20210412155912_201650e1
13	2021/04/12 15:59:15	(FOC2666)	INSERT PROCESS ENDED AT 15.59.15, ELAPSED TIME = 00:00:00.690	qwqdmtest	flow02	20210412155912_201650e1
14	2021/04/12 15:59:15		1	qwqdmtest	flow02	20210412155912_201650e1
15	2021/04/12 15:59:15		0	qwqdmtest	flow02	20210412155912_201650e1
16	2021/04/12 15:59:16		0 NUMBER OF RECORDS IN TABLE= 29090 LINES= 29090	qwqdmtest	flow02	20210412155912_201650e1
17	2021/04/12 15:59:16		QWQDMTEST/flow_new_prod HELD AS DB2 TABLE	qwqdmtest	flow02	20210412155912_201650e1
18	2021/04/12 15:59:16	(ICM18744)	Ending Load	qwqdmtest	flow02	20210412155912_201650e1
19	2021/04/12 15:59:16	(ICM18040)	Return Code = 0	qwqdmtest	flow02	20210412155912_201650e1
20	2021/04/12 15:59:16	(ICM18076)	Request: qwqdmtest/flow02 - finished processing	qwqdmtest	flow02	20210412155912_201650e1
21	2021/04/12 15:59:16	(ICM18007)	CPU Time : 87	qwqdmtest	flow02	20210412155912_201650e1
22	2021/04/12 15:59:17	(ICM18031)	Finished	qwqdmtest	flow02	20210412155912_201650e1
23	2021/04/12 15:59:17	(ICM18072)	Elapsed run time 0:00:03	qwqdmtest	flow02	20210412155912_201650e1
24	2021/04/12 15:59:17	(ICM18975)	------ End of Log Record for qwqdmtest/flow02 ------	qwqdmtest	flow02	20210412155912_201650e1

Figure 86 Viewing the run log

The contents of your log may not look the same as the log shown above. There are environmental factors that affect what is produced in the log. The entries `Return Code = 0` and `Finished` indicate that the flow completed successfully.

> **NOTE**: Notice that the ribbon options have changed to be specific to the log. This is another example of how the ribbon changes based on context.

5 Creating Process Flows

5.1 Overview

A *Process Flow* controls how one or more data flows will be processed. The order of the process is defined by positioning a set of objects in the workspace and defining their interactions. A process flow contains:

- A *Start* object that defines where to begin in the process flow.
- *Data Flow* objects that indicate what data to extract and copy from data sources to data targets.
- *E-mail* objects that notify users about the status of the process at specified points.
- *Stored Procedure* objects that perform a variety of supplementary tasks before or after the extraction, and load steps defined in a data flow.
    - **NOTE**: These are DataMigrator stored procedures, not database stored procedures.
- *Connector* objects, represented as arrows, that specify execution logic for the other objects included in the process flow. For example, it indicates what to do next when a step in the process succeeds or fails.
- *Group* objects that specify the flow of a subset of objects within the total flow. For example, two objects in a group that should be processed simultaneously, rather than sequentially, with the object that follows next. This could, be an email notification that is waiting until processing has been completed for both grouped objects.

As mentioned before, every data flow automatically has a process flow created for it. When you run or submit a data flow, it is the process flow that is being executed. The process flow drives the data flow.

We can see this in the flow created in the last chapter. There is a tab at the bottom of the data flow workspace called `Process Flow`. Click it to see the automatically generated process flow.

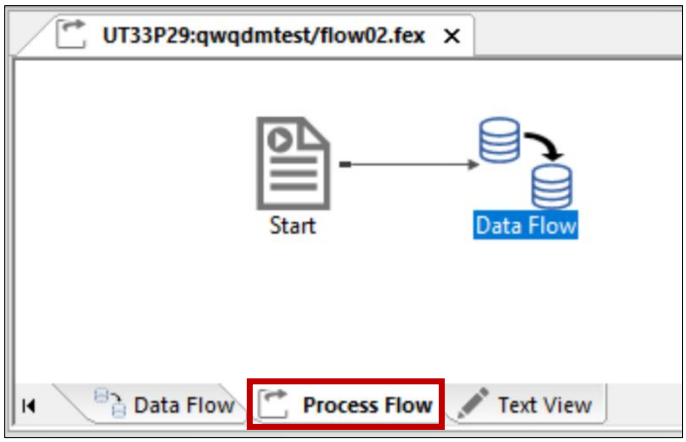


Figure 87 Process flow for a data flow

In the simplest case, you do not need to create a process flow since DataMigrator does it for you. However, there are many occasions where you will want to add more steps in the flow: combining multiple data flows into one process, generate email feedback, handle errors, etc…

5.2 Creating a Process Flow

Now we will create a process flow that uses the data flow created in the last chapter, then branches to one of two procedures based on the success or failure of the execution of that data flow. The results will be recorded in a log to communicate success or facilitate troubleshooting.

> **NOTE:** You can extend the success and failure branches of process flow to trigger the distribution of email messages, either to those in charge of correcting errors (upon failure), or to those who need the current data (upon success).

The simplest way to proceed would be to continue working with flow02's process flow and add other elements to enhance it. In some situations, that might be sufficient. However, this is not necessarily the best way to proceed. Instead, we will use a method that is more *modular* and *flexible*. We will start a new flow, then click the `Process Flow` tab and drag the Data Flow object into the Process Flow workspace in the correct position in the flow. With this method, each flow is saved separately; a Data Flow object can then be used in more than one Process Flow, and a Process Flow can be modified as needed and used to incorporate different data flow objects.

To create a process flow, we start the same way as we did for a data flow.

Right click the `qwqdmtest` folder and click `New`, then `Flow`. A workspace opens in the right pane. The Data Flow tab is active. Click the `Process Flow` tab to change design mode. To anchor the Process flow, the `Start` object is automatically added to the workspace.

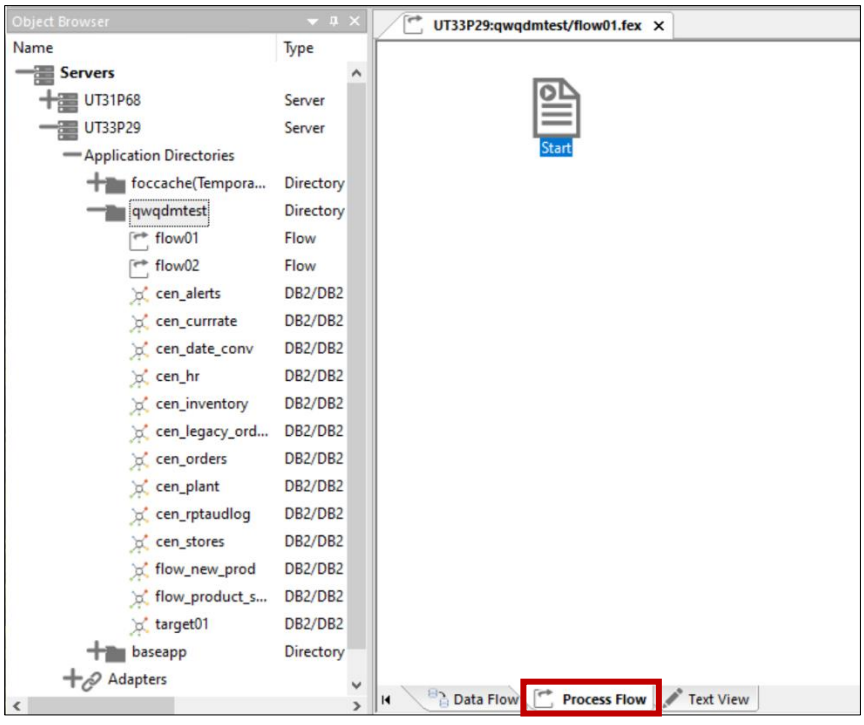


Figure 88 Creating a new process flow

Then add `flow02` to the process flow by dragging it from the navigation tree into the Process Flow workspace to the right of the `Start` object. Then right click the `Start` object and drag it towards the `flow02` object to form an arrow that connects the two objects.

> **NOTE**: Move your cursor well into the `flow02` icon so the arrow connection is made. Do not stop at the edge of the icon or the arrow will not connect.

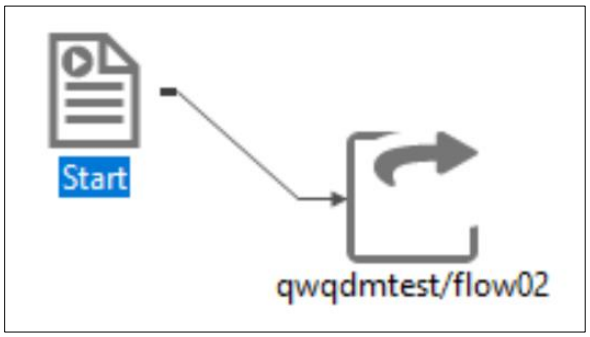


Figure 89 Inserting flow02 into the process flow

Click the `Save` button. The `Save Procedure As` window opens. Name the Process Flow `flow02_process`.

5.3 Adding Procedures and Conditions

Next, we will create two stored procedures. We will connect these procedures to the data flow object so that one will run when the data flow executes successfully, and the other one will run when the data flow fails. Following the same modular model, we will create each of the stored procedures as a separate file that can be used in this process flow and others.

First, create a procedure that will appear in the log when the data flow executes correctly. Right click the `qwqdmtest` folder and click `New`, then `Procedure`. A text editor window will open. Enter `-TYPE SUCCESS!` Click the `Save As` button and name this procedure `success`.

Next, create a procedure that will appear in the log when the data flow fails. Right click the `qwqdmtest` folder and click `New`, then `Procedure`. A text editor window will open. Enter `-TYPE FAILURE!` Click the `Save As` button and name this procedure `failure`.

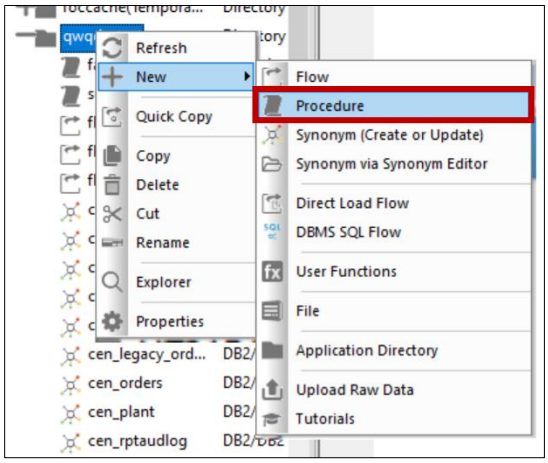


Figure 90 Creating a new procedure

Both procedures are now listed in the `qwqdmtest` folder.

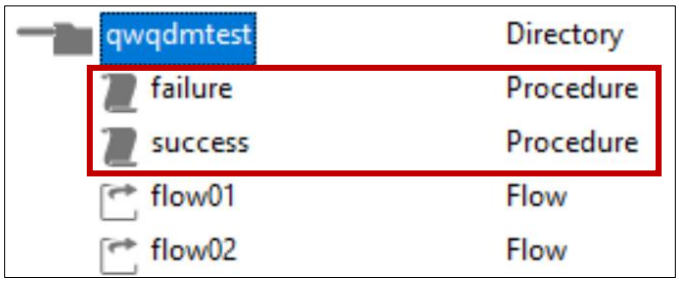


Figure 91 Viewing newly created procedures

Now drag the `success` and `failure` procedures onto the workspace to the right of the `flow02` object. Connect `flow02` to each procedure by right clicking and dragging the connector line to each procedure, as we did earlier to connect `Start` to `flow02`.

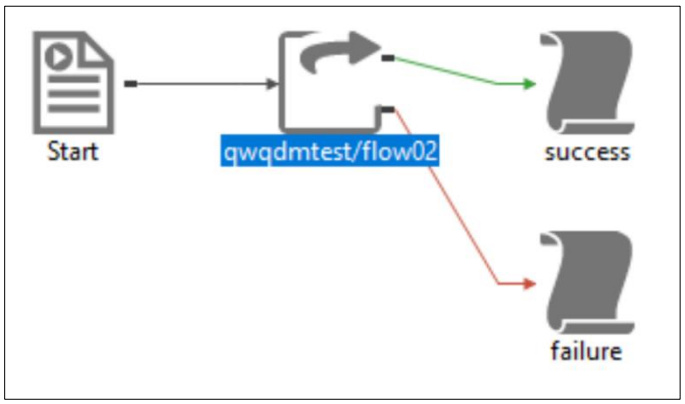


Figure 92 Proccess flow after adding procedures

NOTE: If the labels on the icons include the folder name `qwqdmtest`, which can make it hard to identify the icons, right click the icon, select `Properties`, then change the display name to something more readable.

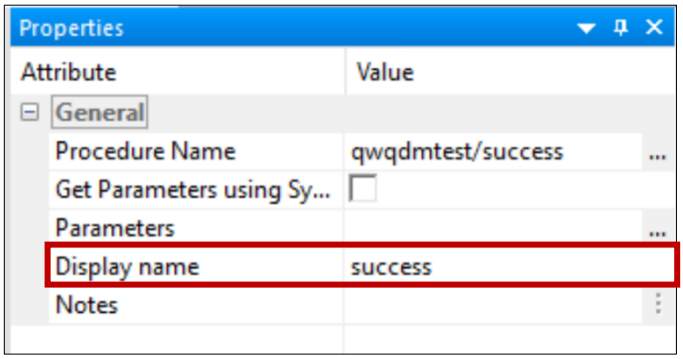


Figure 93 Changing the display name

We need to define the behavior associated with each branch of the flow, which can be done from the connector arrows. Double click the green arrow connecting the data flow object to `success`. The Condition window opens.

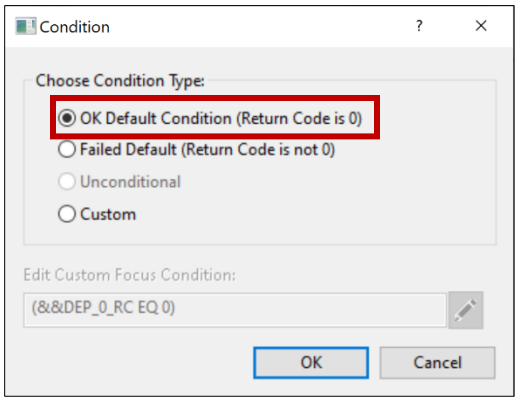


Figure 94 Success condition window

Select `OK Default Condition (Return Code is 0)` then click `OK`.

Now double click the red arrow that connects to the failed procedure. Select `Failed Default (Return Code is not 0)` then click `OK`.

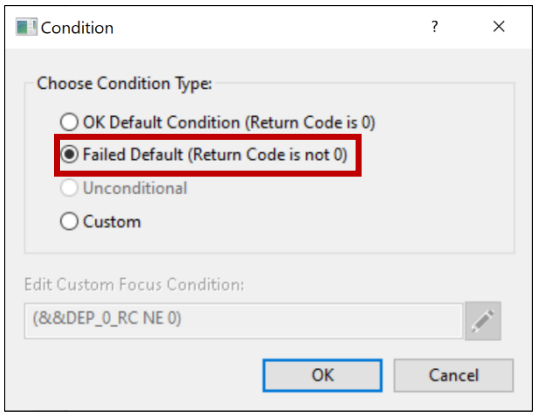
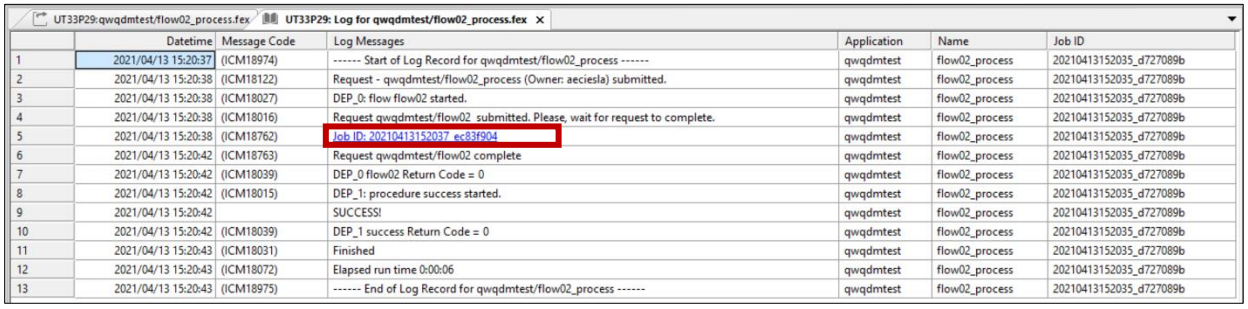


Figure 95 Failure condition window

Click the `Save` button to save the process flow.

5.4 Running the Process Flow

It is time to run the flow to see if it is successful and if the proper messages are displayed. Click `Run` from the ribbon and choose `Submit`. Once the run completes, click `View Last Log` from the ribbon.



UT33P29:qwqdmtest/flow02_process.fex | UT33P29: Log for qwqdmtest/flow02_process.fex

	Datetime	Message Code	Log Messages	Application	Name	Job ID
1	2021/04/13 15:20:37	(ICM18974)	------ Start of Log Record for qwqdmtest/flow02_process ------	qwqdmtest	flow02_process	20210413152035_d727089b
2	2021/04/13 15:20:38	(ICM18122)	Request - qwqdmtest/flow02_process (Owner: aeciesla) submitted.	qwqdmtest	flow02_process	20210413152035_d727089b
3	2021/04/13 15:20:38	(ICM18027)	DEP_0: flow flow02 started.	qwqdmtest	flow02_process	20210413152035_d727089b
4	2021/04/13 15:20:38	(ICM18016)	Request qwqdmtest/flow02 submitted. Please, wait for request to complete.	qwqdmtest	flow02_process	20210413152035_d727089b
5	2021/04/13 15:20:38	(ICM18762)	Job ID: 20210413152037 ec83f904	qwqdmtest	flow02_process	20210413152035_d727089b
6	2021/04/13 15:20:42	(ICM18763)	Request qwqdmtest/flow02 complete	qwqdmtest	flow02_process	20210413152035_d727089b
7	2021/04/13 15:20:42	(ICM18039)	DEP_0 flow02 Return Code = 0	qwqdmtest	flow02_process	20210413152035_d727089b
8	2021/04/13 15:20:42	(ICM18015)	DEP_1: procedure success started.	qwqdmtest	flow02_process	20210413152035_d727089b
9	2021/04/13 15:20:42		SUCCESS!	qwqdmtest	flow02_process	20210413152035_d727089b
10	2021/04/13 15:20:42	(ICM18039)	DEP_1 success Return Code = 0	qwqdmtest	flow02_process	20210413152035_d727089b
11	2021/04/13 15:20:43	(ICM18031)	Finished	qwqdmtest	flow02_process	20210413152035_d727089b
12	2021/04/13 15:20:43	(ICM18072)	Elapsed run time 0:00:06	qwqdmtest	flow02_process	20210413152035_d727089b
13	2021/04/13 15:20:43	(ICM18975)	------ End of Log Record for qwqdmtest/flow02_process ------	qwqdmtest	flow02_process	20210413152035_d727089b

Figure 96 Log results from running the process flow

The log shows the message `SUCCESS!` which means it was generated by the stored procedure called `success`.

> **NOTE**: If you double click the link you will see the detail log of that step.

This was a pretty simple process flow, but it demonstrates the potential for more complex flows. For example, the figure below shows a more complex flow.

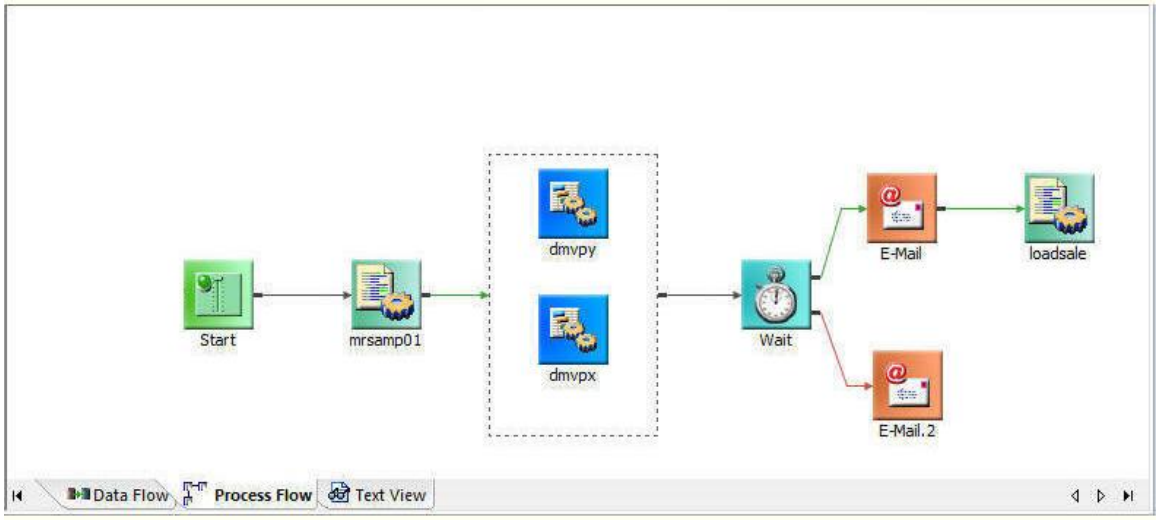


Figure 97 Example flow using email notifications