

Ecomedate Quick-Start Guide

Ecomedate Implementation Overview

The objective of this guide is to help you to generate the necessary initial load and incremental Warehouse scripts with which to convert, normalize, transfer, and move data from Ecometry tables to the Ecomedate Data Foundation. It is not meant to replace the Jump Start Training but to act as a refresher after the trainer has left to augment the notes you took during the training sessions.

Ecomedate uses of 80+ Ecometry source tables to populate 300+ target tables. These tables are organized into 13 Queues: SETUP, ITEM, OFFER, CUST, CSTX, ORD, ORDS, OA, ORDO, PO, INV, SUM and FIN. You will develop scripts for tables on a queue-by-queue basis. From customers that have completed their implementations, they shared with us that that if they could do it over again they would:

- 1) Have a data source expert at the ready to work through the numerous data issues that were uncovered through the implementation.
- 2) Have a dedicated resource to implement. It was very difficult doing the work as a 30 minute a day effort. Having some contiguous time devoted to this implementation project ultimately made the project proceed more quickly.
- 3) Implement in pieces. Bring up queue by queue. Finish the entire process for one queue **and** put into production before moving on to the next queue.
- 4) Implement in the order that would bring the most value to the business. If having sales information would save the company \$90 and having a report of obsolete parts would save the company \$50, then do the sales information first.

You will be using the Warehouse's interactive development environment, DataBridger Studio, to manipulate the structures and the data and to generate the Warehouse scripts. Studio is a programmer's workbench. Warehouse is the tool you will use to execute the scripts that move the data in production. Warehouse can either be invoked from within Studio or run outside the workbench.

Ecomedate Quick-Start Guide

In order to establish the environment, you will need to install `Studio` on your workstation and `Warehouse` on the platforms where the source and target databases are located. Please see <http://www.taurus.com/support/StudioQuickStart.htm> for installation of these products and the creation of a `Studio` repository on your workstation. We recommend creating a non-shared blank `MicrosoftAccess` database that is local to your workstation as your `Studio` repository.

Installation of the software and testing of the connectivity to the data sources and targets is part of the planning stages as outlined in the *Ecomedate Implementation: What to Expect Guide*. All planning stages must be complete before beginning your data movement training or data movement work. Completing this work up front assures a more successful implementation. The data movement work will be outlined in the Ecomedate Roadmap document which was developed by Taurus and the data movement team. The Roadmap is a `Microsoft Excel` spreadsheet detailing all the mappings, their source tables, the target tables created, and the mappings involved in the data movement. This document is essential for navigating your development effort in your Ecomedate implementation.

In addition to the `Warehouse` software installed on both source and target and `Studio` installed on your PC, you will need the mappings. These mappings can be downloaded on the Taurus download site. You should set up two directories on your machine: one for mappings delivered by Taurus and one for the mappings which you have modified. Within those directories, you should set up a folder for each queue. This will allow you to get back to the original mapping if need be and to have an easy list of those mappings which were modified for your use.

Ecomedate Quick-Start Guide

Prior to beginning any work with `studio` to generate the data movement routines, you need to:

- 1) review the mappings in `Excel` and ensure that you are NOT using any fields in Ecometry differently than they are documented in the mappings
- 2) **If you use Ecometry on MPE**, you need to ensure that the unique identifiers indicated in the mappings will work for your data. The unique identifier work is the single most important part of the project. If you can find unique identifiers that work for your data, the finding of records for updates and deletes will be faster. The UI combination should yield one and one only record. Check your data against the suggested unique identifiers prior to going any further in the development process. If the suggested UI doesn't yield a unique entry, review the duplicates and ensure that the data is good and see if you can find another combination of fields which will work. This can be done using the data profiling feature of the `studio IDE` product. See *section 14* for more information about using this feature or the use the `studio` help for a small tutorial.

Ecomedate Quick-Start Guide

The Mechanics

Once these steps are complete, you are ready to move forward and develop the data movement routines. The remaining parts of the *Ecomedate Quick-Start Guide* detail the process for developing these routines. For our purposes, we will assume our Ecometry application is running on SQLServer and we have decided to create our Ecomedate data foundation on SQLServer.

The required steps in the script generation process are:

1. Creating a repository (usually in Microsoft Access) to store your work.
2. Opening the repository
3. Creating the data sources
4. Importing the macros file into Studio
5. Importing the mappings into Studio (either via an entire folder or individually)
6. Creating the target tables and indexes either using DDL created in Studio or by creating them directly using Studio
7. Creating a Load Mapping
8. Creating a Load Script
9. Testing the Load Script
10. Validating that the queue is in sync
11. Creating a Propagation Queue
12. Testing the Propagation Script
13. Go live

In addition to the mechanics, we have added at the end of the document a special topics section, *section 14*, to discuss in more depth, some of the special data modeling situations you may encounter in your Ecomedate implementation.



Ecomedate Quick-Start Guide

1. Creating a repository

In this step you will:

- Create blank Microsoft Access database
- Name repository for queue
- Create ODBC Datasource to connect to repository

Open Microsoft Access and select New from the File menu. Select blank database. Select a location to save your new repository and give it a name that identifies the queue that you're working on. For instance, if you're working on the item queue, call the repository Item. If you do not own Microsoft Access, contact support and they will email you a blank Microsoft Access database that you can use. It not necessary to own Microsoft Access to use a database created by Microsoft Access. You will want a different repository for each queue that you implement. Think of a repository as temporary storage for a data movement project in the development cycle.

Once your repository has been created, you'll need to create an ODBC Data Source that connects to that repository. To do this, go into the Control Panel, select Administrative Tools. Select Data Sources (ODBC). Select the System DSN tab and click on Add. From the list of drivers, choose the driver for Microsoft Access and click FINISH. Give the data source a name that identifies the queue, such as Item, and then click SELECT to locate the Microsoft Access database that you just created. Click OK until you have exited Data Sources. It makes it easier to manage, if you name the repository and data source by the name of the queue.

2. Opening the Repository

In this step you will:

- Open Studio
- Select the repository
- Update General Preferences

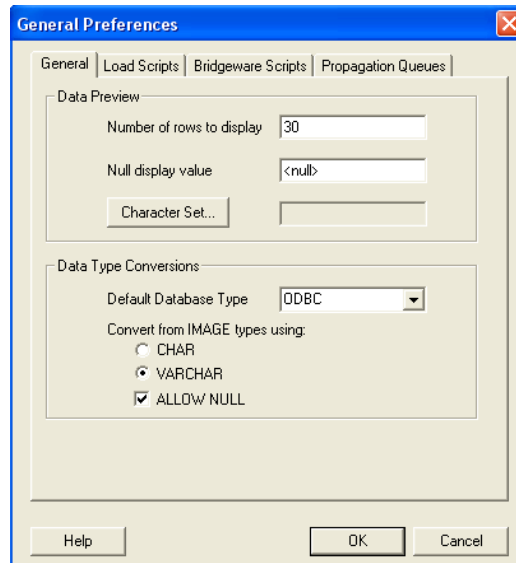
Ecomedate Quick-Start Guide

- Update Load Script Preferences
- Update Propagation Queue Preferences

Start up Studio, and if this is the first time accessing this repository, select `Connect to new repository...` When the `Select Data Source` window appears, choose the Studio repository you have created from the list of ODBC DSNs under the `Machine Data Source` tab and click `OK`. Next it will ask you to `Login` in the repository, just click `OK`. When prompted whether to `Create New DataBridger repository tables`, reply `YES`. Studio will create the necessary `Repository tables` for your use.

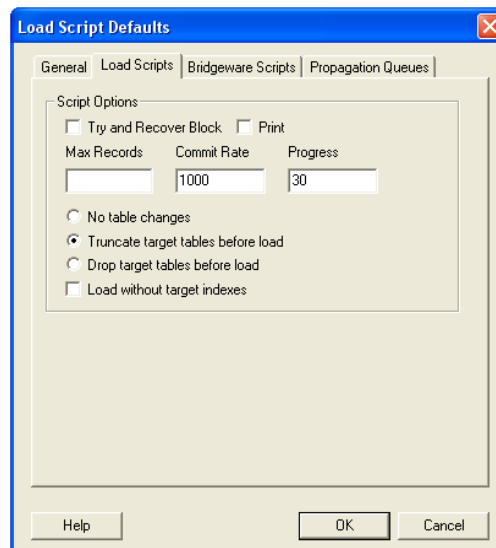
If you have accessed this repository before, you can just select the repository from the list of repositories listed. If you can't find it on the list, just select `Connect to a new Repository`. When prompted, select your old repository.

To modify the general preferences, select menu option `View | Preferences`. From the dialog box, `General Tab` we can change the number of data rows to display and set what text to use to denote `NULL` values. The default is to show 30 rows of data and to show blanks for `NULL` values. Change the `NULL` value entry to `<null>`. Your screen should look similar to this:



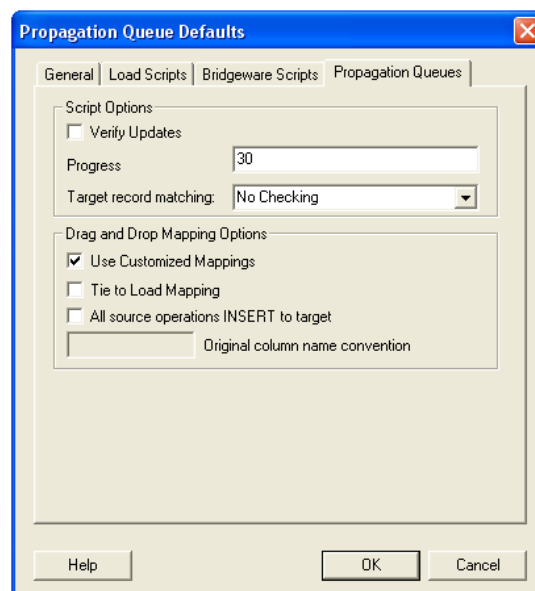
Ecomedate Quick-Start Guide

To update the Load Script preferences, click on the tab: Load Scripts. We are going to change the Commit Rate and Progress fields to optimize performance and provide feedback at a regular interval and set the indicator to delete table data before performing a load. Enter a Commit Rate of 1000, and a Progress rate of 30 seconds. Select the option to Truncate target tables before load. When finished, your screen should look similar to this:



Ecomedate Quick-Start Guide

To update the Propagation Queue preferences, click on the tab: `Propagation Queues`. We are going to change the `Progress` field to provide feedback at a regular interval and set the indicator to use custom mappings. Enter a `Progress` rate of 30 seconds and select the option to `Use Customized Mappings`. When finished, your screen should look similar to this:



Similar changes can be made to the `BridgeWare Scripts` tab. Click `OK` to save the changes.

3. Creating the data sources

In this step you will:

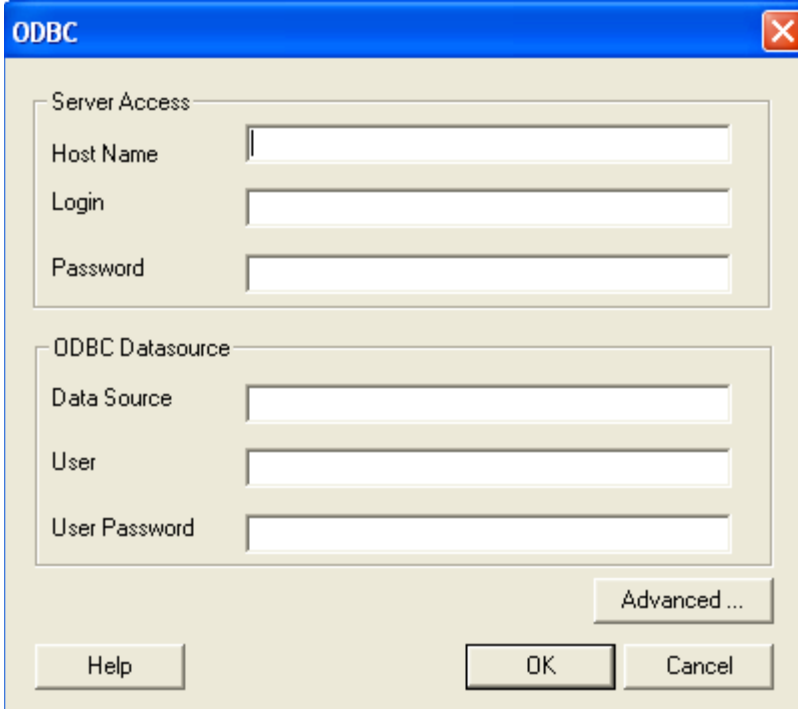
- Create the Ecometry data source for your source environment
- Create the target data source (your Ecomedate database)
- Change advanced settings on Ecometry data source
- Connect to both the target and source databases

The next step is to connect to your source and target databases. This is done by creating a data source for each environment.

Ecomedate Quick-Start Guide

To connect to a data source, right-click on `Data Sources` and select `Open Data Source`. Choose a meaningful name for these data sources as this name will be used within the scripts. Normally we use the data source name of `Ecometry` for the source environment. We use `Ecomedate` for our data source name for our target.

Each connection should be setup as a **remote** data source. As we have a `SQLServer` source and target, we will be using a `REMOTE ODBC` database type for both. Press `OK`. You will see the following screen:



The information you will need for this screen is as follows:

- `Host Name`: This is the name or IP address of the server you are connecting to.
- `Login`: This is the Windows server login (keep in mind that this user name must have the right to log on as a service). Instructions for providing a Windows user for the right to log on as a service is

Ecomedate Quick-Start Guide

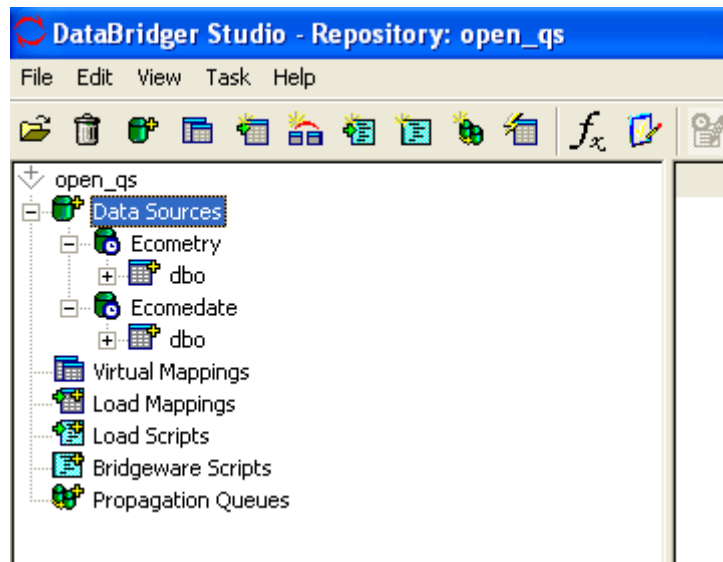
- covered in the *Warehouse Reference Manual* in *Chapter 7, Installation*.
- **Password:** Enter the Windows server login password.
 - **Data Source:** This is the data source name for the database you are attempting to connect to. For example, in Ecometry, the Data Source would most likely be `ECOMLIVE`, `ECOMTEST` or `ECOMVER`. In Ecomedate, it would probably be called `ECOMEDATE` or `ECOMEDATETEST`. If you're uncertain, you can log on to the server, go into Data Sources (ODBC) and find out.
 - **User:** This is the user to log in to the database. This is only used if you are using SQL Server Authentication in that environment
 - **User password:** Supply a password if using SQL Server Authentication.

 - For **Ecometry** data source **only:** select the `Advanced` button and make sure the `make arrays from sequential columns` box is checked.

Once the connection is established to the data source, you will see the list of tables in that database. You can view the table's structure by selecting `View Attributes` from the `View` menu. Or, alternatively, you can view data in the table by selecting `View Data`, which allows you to browse the actual data in a table. The number of rows of data which are returned is controlled by `Preferences` on the `View` menu.

Ecomedate Quick-Start Guide

After your data sources have been connected, your repository should look similar to this:



4. Importing the Macros into Studio

In this step you will:

- Obtain `macros.csv` from Taurus Support or the download site
- Import macros into repository

Before importing any mappings into Studio, you will want to import the macros which are used throughout the mappings. In order to import the macros file, click on **Tasks** from the task bar. Under **Tasks**, select **Macro Expressions**. A dialog box will appear. Click on **Import**. Find the macros files that was sent to you by your support representative, this file is normally named `macros.csv`. As the import process completes, you will get a status of how many macros were imported. Click **OK**. Click on **Done** to exit the import process. The macro file must be imported into every repository that is created.

Ecomedate Quick-Start Guide

5. Importing the Mappings into Studio

In this step you will:

- Import mappings by folder
- Select Ecometry data source
- Chose appropriate data type
- Remove mappings that won't be used
- Edit mappings to select correct key fields to create indexes

Each source table is represented by one or more mappings in CSV format that will be used by Studio to construct each target table. Each Queue represents multiple source tables and multiple target tables.

As previously stated, you will likely be creating scripts on a Queue-by-Queue basis, so we will only be working with the mappings for the queue that you are working on. The Roadmap will be your guide to what source tables belong in what queue, what level of normalization you want to implement, the mappings to be used and what release is supported by those mappings. Using our normal Roadmap conventions you will be implementing those tables which are NOT shaded.

It is important to double check that you are using the correct version of the mappings for your release of Ecometry. In the excerpt of the Roadmap shown below, we can see that the STYLEHEADER table is part of the SETUP queue. If I am on version 5 of Ecometry, I would use the 01B_STYLE_HEADER mapping and only write to one target table, SV_MACITM_STYLE_HEADER. If I am on version 6 of Ecometry, I would use the 01B_601_STYLE_HEADER, 01B_IT_601_STYLE_LVL_INDEX_1, 01B_IT_601_STYLE_LVL_INDEX_2, 01B_IT_601_STYLE_LVL_INDEX_3 mappings and write to two target tables: SV_IT_601_STYLE_HEADER and SV_IT_601_STYLE_LVL_INDEX. Normally in the Roadmap, only those mappings which are appropriate for my version of Ecometry should be "white" or not shaded. If you find a discrepancy or have questions, contact your support representative.

DataSet Name	Queue	Target Table Name(s)	Description	Mapping Name	5x ?	6x ?	7x ?
SYTLEHEA	SETUP	SV_MACITM_S	Table	01B_STYLE_HEADER	Y	N	N

Ecomedate Quick-Start Guide

DER		TYLE_HEADER	Description:STYLE-HEADER contains an entry for each "style" defined in the Ecometry product set-up. Information includes: style ID and description, and a flag to indicate whether this information is based on an actual entry in the Ecometry STYLE-HEADER table, or if it is generated by "backfilling" while processing ITEM-MAST entries.				
SYTLEHEA DER	SETUP	SV_IT_601_S TYLE_HEADER	Version 6 of style-header	01B_601_STYLE_HEAD ER	N	Y	Y
SYTLEHEA DER	SETUP	sv_it_601_s tyle_lvl_in dex	Table Description:STYLE_LVL_INDEX contains an entry for each variant of each style in the STYLE-HEADER table. Information includes the style ID, the variant level (level 3, level 2, or level 1), and the highest value of the INDEX for the variant level for individual items that are part of the style.	01B_IT_601_STYLE_L VL_INDEX_1, 01B_IT_601_STYLE_L VL_INDEX_2, 01B_IT_601_STYLE_L VL_INDEX_3	N	Y	Y

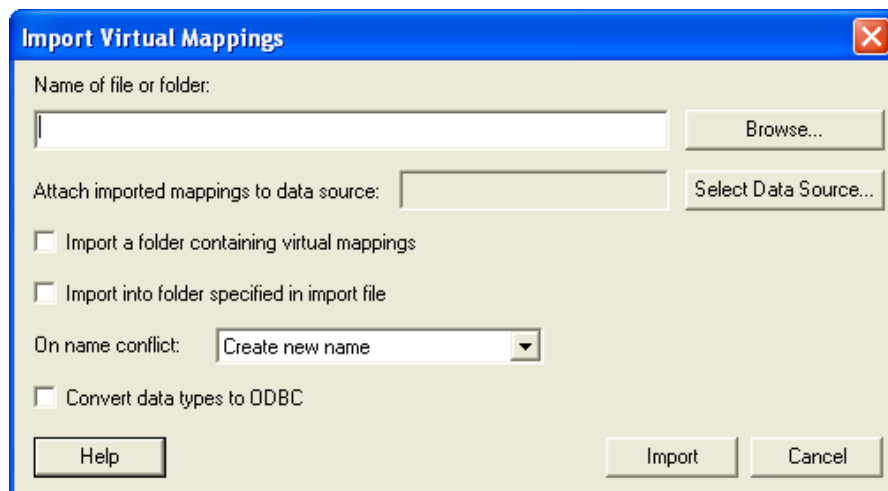
There are two types of tables that are created in the Ecomedate architecture: base tables and breakout tables. A base table will have one row of data for every row of source data. Base tables will have three groups of columns: header columns, original columns and breakout columns. Header columns contain data about who wrote the transaction,

Ecomedate Quick-Start Guide

when it was written, and how the person was logged on when they wrote it. The original Ecometry columns are converted into corresponding SQLServer data types with no transformation. Breakout columns are clean, usable columns that contain Ecometry data including any coded or hidden Ecometry source fields. Breakout tables are for selected information only and are used to normalize the Ecometry data into more usable structures. Assuming that we are on version 6.x of Ecometry, the base table for style-header is `sv_it_601_style_header`. The breakout table is `sv_it_601_style_lvl_index`. In the base table, I would expect the same number of rows as are in the `styleheader` table. For the breakout table, there will be rows written for each of the three style levels that are not blank. Each mapping and each column within the mapping is documented.

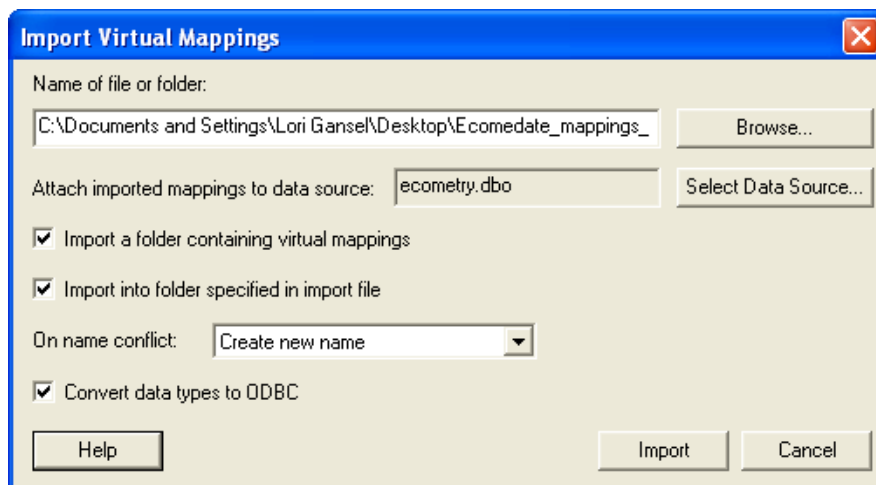
When importing mappings, it is helpful to import entire folders of mappings. Ecomedate mappings include a "folder name" column in the csv file so that the folder hierarchy is maintained once imported. This not only helps to keep everything organized, but also allows you to easily drag and drop the entire folder (as opposed to individual mappings) when creating tables and load mappings.

To import mappings, right-click on `Virtual Mappings` and select `Import Virtual Mappings`. A dialog box will be displayed:



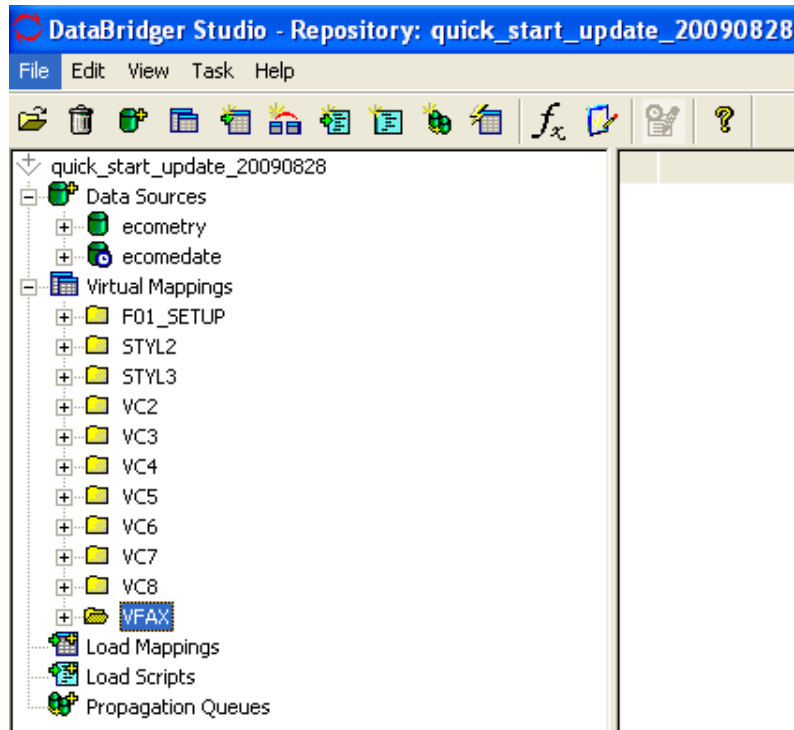
Ecomedate Quick-Start Guide

You have two options for importing the mappings: importing them one at a time, or importing a folder of mappings. For this example, we will import the mappings for the entire queue, and we want to maintain the folder names contained in the mappings. To do so, check the boxes next to `Import a folder containing virtual mappings` and `Import into folder specified in import file`, and then click on `Browse` to select the folder. Select a folder at the queue level. For example, if you are implementing the Setup queue, you will select the folder labeled "Setup". Once the folder is selected, click on the button labeled `Select Data Source` and choose your source environment. All mappings default to Oracle database type for the target database. If your target environment is SQLServer, as ours is, check the box labeled `Convert data types to ODBC`. Click on the `Import` button. All mappings within that queue will be imported into Studio. The screen should look like this prior to clicking on the `Import` button:

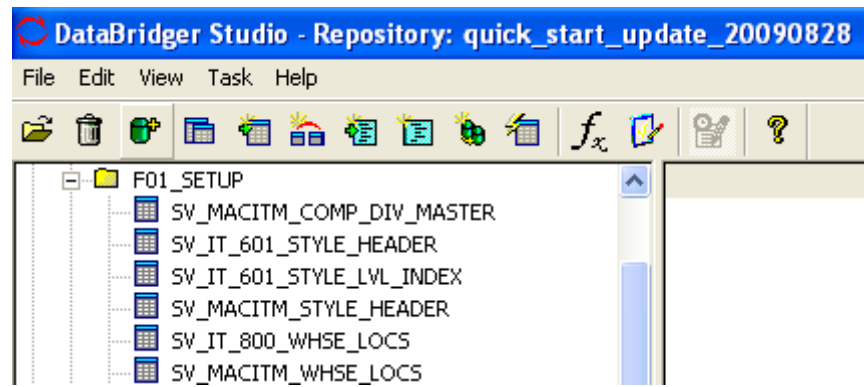


Ecomedate Quick-Start Guide

Once the mappings have been imported, Studio's tree should look similar to this:



Any virtual mappings that will not be used should be deleted before continuing. For example, after importing the Setup queue mappings, your screen looks like this:



Ecomedate Quick-Start Guide

If, for example, you are on version 8.x of Ecometry, you would want to use version 6.01 of Style Header since there is no mapping for version 8.x. You would also want version 8.00 of Whse Locs. In this example, you would remove the mappings called SV_MACITM_STYLE_HEADER and SV_MACITM_WHSE_LOCS, leaving the mappings for the correct versions.

A virtual mapping can be edited to add or drop columns, change column attributes, perform transformations or specify conditional transformations based on data content. To edit a mapping, right-click on the virtual mapping and select `Edit Virtual Mapping`.

If you are an MPE Ecometry user, don't forget to design your unique indexes to the table before creating your table. Also make sure the unique identifier you have chosen is listed as the `KEY` in the mapping and the attributes for the field are `NOT NULL`. If the column is a character data type, make sure we will be able to match the source and target correctly by either choosing `CHAR` as your data type for strings or `TRIMR` (remove trailing space) of the string so that the compare of the string works. Most mappings that you will receive will have default keys and indexes set, so review to ensure that these will work for your business. Be careful of the `CHECKCHAR` function as it may return `NULLS` depending on how you chose to set it up. If you are unsure, please talk with your support representative.

You may find it helpful to use the Data Profiling functionality of Studio to identify a unique identifier for your table. Please see the *Extras and special circumstances* section of this documentation for information on Data Profiling.

Before moving to the next step:

- you should have **NO** errors being reported either in the mapping itself or the preview of data using the mapping
- you have indicated your unique index in the mapping.

6. Creating the target tables and Indexes

In this step you will:

Ecomedate Quick-Start Guide

- Create target table in one of three ways
- Create Indexes on target tables

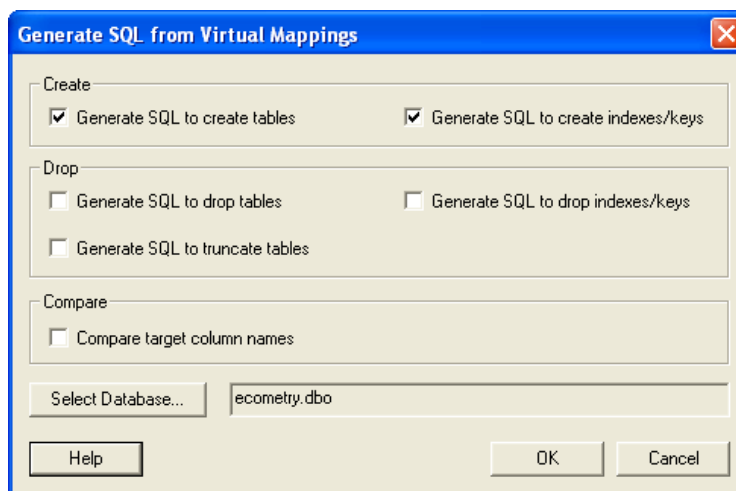
There are several ways to create a table in our Ecomedate data foundation:

- Right-click on an individual mapping and choose *Generate SQL* and then select the database where you'd like for the table to reside
- Drag the mapping up to the Ecomedate data foundation data source
- Right-click on the mapping and choose *Save DDL*, which could then be executed within *Query Analyzer*

You also have the option of creating all tables within the queue at the same time, by:

- Dragging the entire folder up to the Ecomedate data foundation
- Right-clicking on the folder and choosing *Generate SQL*
- Right-clicking on the folder and choosing *Save DDL* and then running the result DDL through *Query Analyzer*

When you choose to create the tables and indexes using Studio, the following dialog box will appear:



Ecomedate Quick-Start Guide

Choose `Select Database` to choose your target database and then press `OK`. You may also use this functionality when you wish to drop tables or indexes, or to truncate tables.

Special cases for handling virtual mappings

As you already know, sometimes there is a single mapping for a target table. An example of this is the target table `SV_MACITM_COMP_DIV_MASTER`. Sometimes, we create multiple target tables using more than one mapping. This is done when the data is sourced from the source table from multiple fields as we do with telephone numbers. An example of this model is the target table mapping `SV_MACITM_VEND_ORD_TEL`. The source table `VENDORFROM` can create up to 2 target records depending on the source values in the column `dayphone` and the column `faxno`. We have two mappings, one which sources the data from the column `dayphone` and one mapping which sources the phone number from the column `faxno`.

There are certain situations where there is no base table in Ecomedate for a particular table in Ecometry. For example, the `PROCESSING` table in Ecometry writes to several different tables in Ecomedate based on processing record type. You can select however many of the breakout tables that you'd like to report on. For example, you could decide you want to have coupon information and invoice information in their own breakout tables. In this scenario, you would have three target tables in Ecomedate written to by the single `processing` table in Ecometry: `SV_MACORD_PROCESS_COUPON`, `SV_MACORD_PROCESS_INVOICE` and `SV_MACORD_PROCESS_OTH`. The mappings for the `_OTH` tables are released expecting that all breakout tables were implemented. If this is not your case, the `_OTH` will need to be modified in order to include the data that would have resided in the breakout tables that you excluded in your implementation. So in this example, the `_OTH` table will need include all `Processing` records that are NOT coupon or invoice records; that way, when you add the coupon, invoice and `_OTH` transactions, they should equal the number of transactions in the `Ecometry Processing` table.

The mechanics for handling these special situations are discussed in *section 14*.

Ecomedate Quick-Start Guide

7. Creating a Load Mapping

In this step you will:

- Create your load mappings from the virtual mappings
- Tie your load mappings to target tables

The difference between a `Virtual Mapping` and a `Load Mapping` is that a `Load Mapping` connects a physical data source and a physical target. A `Load Mapping` is the basis for script generation.

There are two ways in which to create load mappings: 1) drag an individual virtual mapping to the load mappings node or 2) you can drag the entire folder of virtual mappings to the load mappings node. When you drag the individual mapping to load mappings, you are asked to specify a target table. If you drag the whole folder to load mappings, you are only asked to specify the database in which your target tables reside.

8. Creating a Load Script

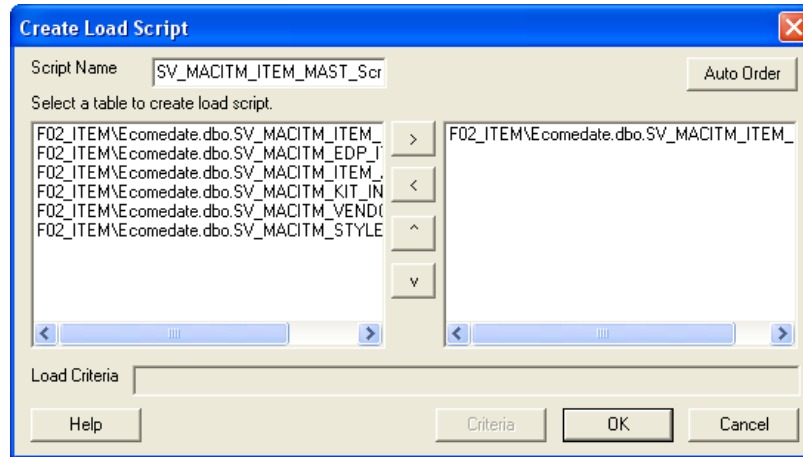
In this step you will:

- Select all load mappings from the same source table
- Create load scripts for each source table
- Save the load script

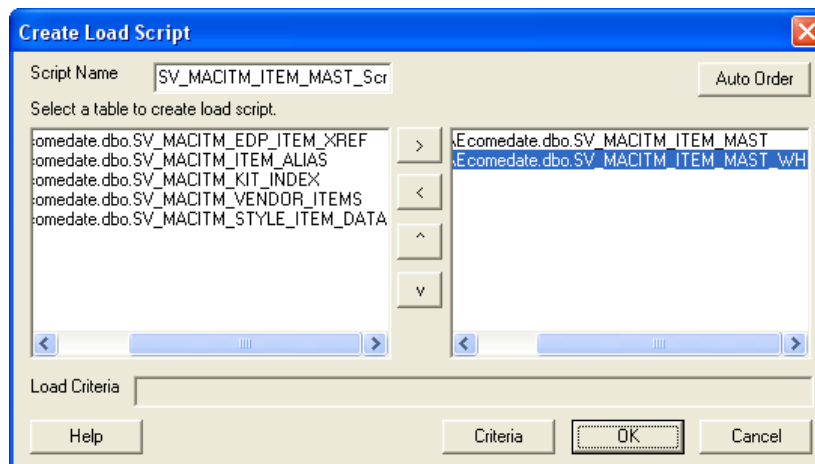
When creating load scripts, remember that all loads from the same source table or dataset should be in one load script. For example, since the `itemmast` table in Ecometry writes to both `SV_MACITM_ITEM_MAST` and `SV_MACITM_ITEM_MAST_WH`, they should be included in the same load script.

Ecomedate Quick-Start Guide

Now you are ready to turn the studio references into Warehouse script code. Drag a Load Mapping to the Load Script node on the tree. You will get a dialog box that looks like this:

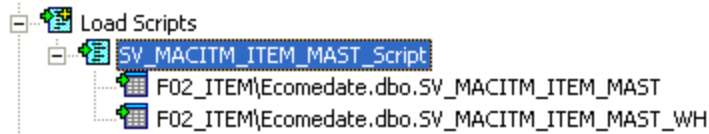


In our example, we want to create a load script for source table ITEMMAST, which writes to two target tables, SV_MACITM_ITEM_MAST and SV_MACITM_ITEM_MAST_WH. After dragging the load mapping for SV_MACITM_ITEM_MAST to the load scripts node, we receive the dialog box allowing us to add more target tables to our script. To include SV_MACITM_ITEM_MAST_WH, click on SV_MACITM_ITEM_MAST_WH mapping on the left side of the dialog box, and then click the right-pointing arrow to add it. The result is:



Ecomedate Quick-Start Guide

Which results in a load script that looks like:



You can then see the generated Warehouse load script in the view window:

```

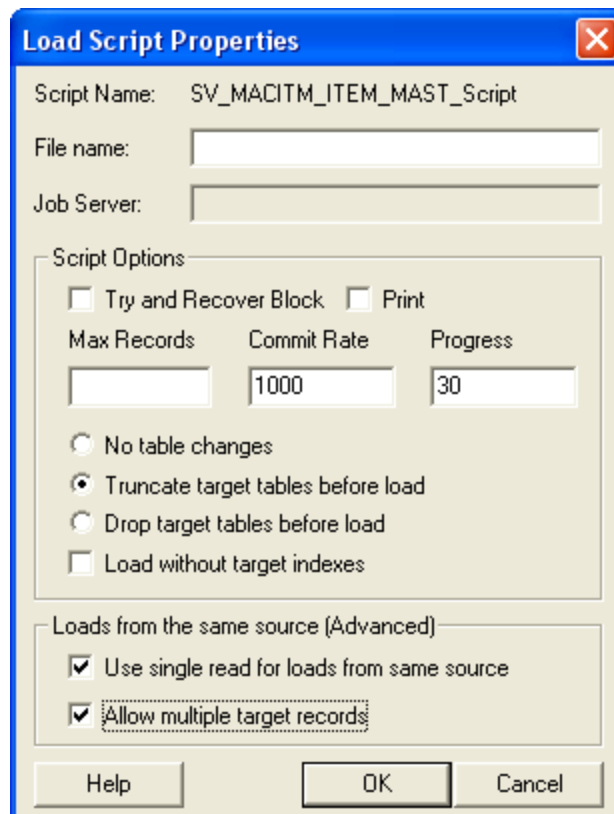
Load Script SV_MACITM_ITEM_MAST_Script
*****
* Taurus/DataBridger Studio 4.8.599500 - Load Script *
*
* Script Name : SV_MACITM_ITEM_MAST_Script *
* Created : 09/02/09 16:49:35 *
* Repository : ItemQueueRepository *
* System : green *
*****
*
* Base table: this mapping maps the item-mast records into the base table, sv_
* macitm_item_mast.
* Table Definition: ITEM-MAST is the repository for all base item information.
* A single entry exists for each item.
* In addition to the base table there are 3 breakout tables: sv_macitm_item_
* mast_alt, sv_macitm_mast_act, sv_macimt_item_mast_wh. For each row within i
* tem-mast it is possible to write 12 records to the breakout tables.
* Table description: ITEM-MAST contains all product data for each individual i
* tem in Ecometry. Includes item number, description, status, standard selling
* price, category codes, customization options, and various other codes and i
* ndicators.
* Modification History for the base table:
* 10-15-03 - vas - Initial mapping for 5.x
* 11-03-03 - css - Add comments for table
* 01-19-04 - vas - Case 3622 - modified mapping to check for NOT space
* 03-24-04 - vas - Case 3874 - modified mapping for add_ph_amt and wght from
* checkz(xxx divf 100) to checkz(xxx) divf 100
* 03-24-06 - swj - Case 7445 - add TRIMR() to CUSTOM_LGCV_CD mapping
* 08-31-06 - vas - 6831 - Add meta data columns to mapping
* 01-21-08 - 12697 - vas - add indexes to mappings
* 01-21-08 - 12698 - vas - remove b_ columns
* 07-21-09 - 16754 - vas - add folder names to mappings
* Copyright 2003, Taurus Software Inc.

```

To optimize our load script, we want to alter the properties of this script to adjust how Warehouse reads our source table.

Ecomedate Quick-Start Guide

To change the properties, right-click on the specific load script, in our case, `SV_MACITM_ITEM_MAST_Script`, select Properties. Check the boxes that read Use Single Reads from Same Source and Allow multiple target records. Your dialog box should look similar to this:



9. Testing the Load Script

In this step you will:

- Run your load scripts
- Visually verify that the resulting data is accurate

The focus of the initial load testing is to ensure that the column by column mapping has occurred correctly. You can run the script either by exporting the script to a DOS directory and running it with the Warehouse client `wh.exe` (right click Save Script), running it directly from Studio

Ecomedate Quick-Start Guide

(right click `Save` and `Run Script Locally`), or by adding the script to `Job Control` and running it from there. Please see *section 14* for instructions on how to create the job control database. Remember that if you modify the generated script outside of Studio, the changes to the script cannot be re-imported.

Regardless of how you run the script, you can verify the results AND the original data easily in Studio. On the tool bar, under `View`, select `Show Two Views`. Click in the first view (top right half) and then go to the source table in the data sources and click on it. The first 30 records of the source table will be shown in top half. Now click on the bottom half of the split screen. Go to your target data source and click on one of the target tables in your load script. Studio will show these records on the bottom half. You can visually verify the results. If you want to view a larger sample of records on the screen, click the `View` menu and select `Preferences`. Put the number of records to view in the `Number of rows to display` field.

10. Validating that the queue is in sync

In this step you will:

- Develop or modify validation scripts for the queue
- Run the scripts and verify the source tables and target base tables are in sync

You will want to develop and use an automated validation set of routines. This can be done by using an architectural feature of the Ecomedate environment, namely the base tables. The base table counts should match the source table contents at all times. Knowing this fact means that at a chosen time every day, you should be able to run a set of checks which will ensure that your environments are in sync.

You can either use `warehouse` scripts or you could write SQL routines to verify that your environments are in sync. If you are interested in using `warehouse` scripts, your Taurus support representative can provide you with some sample templates and help you modify them to match your implementation of Ecomedate.

Ecomedate Quick-Start Guide

The templates provided by Taurus support will contain a set of three reconciliation (or validation) scripts for each. These scripts are named by queue, but the naming convention is the same on each set. For instance, the `ITEM` queue reconciliation scripts are:

`Create_item.txt` – Creates a table for count of both Ecometry and Ecomedate records for that queue.

`Refresh_table_item_cnts.txt` – Writes the number of records in each data source to the table that has been created.

`Chkcounts_item.txt` – Reads the number of records from both data sources and compares them to verify that they are in sync.

These scripts should be run in the order listed above and should contain only the base tables that you are implementing.

Do NOT move to the development of the `Propagation Queue` scripts until all data is validated and you have determined that you are in sync.

11. Creating a Propagation Queue and Script

In this step you will:

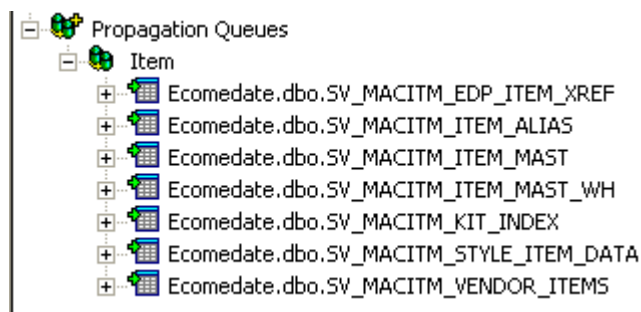
- Create Propagation queue
- Name the propagation queue appropriately
- Select the source database for the triggers to be created in
- Modify the properties of the queue to name the file and use customized mappings
- Drag load mappings for that data source into the queue
- Create queue objects
- Save your propagation script

In order to create a propagation Script, right click the `Propagation Queues` node on the Studio tree. Select `Create Propagation Queue`. Enter a propagation queue name as it appears in the Roadmap. Select

Ecomedate Quick-Start Guide

the data source for the source database in order that triggers can be built to capture changes to the source tables. Right click the Propagation Queue Name and select Properties. Put a name in the Script File box in the Propagation Queue Properties Screen. Check the Use Customized Mappings box and click OK to save the changes.

A propagation queue can be set up to process changes from one or more source tables. The Roadmap has queue groupings already setup for you. For example, the `SETUP` queue has the source tables: `compdivmaster`, `styleheader`, `whselocs`, etc. The `SETUP` propagation queue should have these source tables in it. Once you have created the Setup propagation queue, you will drag all of the load mappings for the above source tables to the propagation queue, one at a time. Studio will add these load mappings to your propagation script. When we completed adding all of the load mappings for our implementation of the `ITEM` queue, our propagation script, `Item`, looked like this in the Studio tree:



For target tables which have multiple mappings, Studio will need to distinguish between each mapping. The first mapping is accepted by Studio in the same way as the other mappings. The second and subsequent mappings will need a unique tag so Studio can distinguish it from the first. When Studio determines that a target table is already part of a Propagation Script, the Script Data Source Tag dialog box will appear. Enter a unique tag name in the box for each load mapping with the same name that is dragged to the propagation queue.

After all of the load mappings have been dragged to the propagation queue, the triggers need to be created for the queue. The triggers are

Ecomedate Quick-Start Guide

created to capture changes to the source tables and need to be set up after the initial loads are done and before any additional changes have been made to the records. To create the triggers and capture objects, right click the `Propagation Queue Name` and select `Create Queue Objects`. If you need to add another load mapping to the propagation queue later, first delete the queue objects by right clicking on the `Propagation Queue Name` and selecting `Drop Queue Objects`. Add your new mapping, then right click again and select `Create Queue Objects`. Just make sure that all of the data in the queue has been processed before the objects are dropped. Right click the propagation queue and select `Save Script`. Save the script to the directory of your choice.

When you create queue objects, Studio will create a table in your source database `queue_name_q` (where `queue_name` is the name of the propagation queue). For each source table that has been linked to the queue, Studio will create a `table_c1` table (where `table` is the name of the source table). The `queue_name_q` table has the type of change, the table changed, and the date of the change. The `table_c1` table has the before and after picture of the data that was updated, or the inserted or deleted data.

12. Testing the Propagation Script

In this step you will:

- Create insert, update and delete transactions in the source database
- Verify those transactions have been captured in the `_q` and `_c1` tables
- Run the propagation script to move the data to the target data base
- Verify the target database changes are correct
- Run the validation scripts to ensure that the source tables and base target tables are still in sync.

The focus of propagation script testing is to: 1) ensure that data is being captured correctly and 2) each captured transaction changes the data

Ecomedate Quick-Start Guide

in the target Ecomedate data foundation appropriately. The propagation script can be run from a command prompt using the `Warehouse` executable (`wh.exe`), or within the `Job Control Panel` by right-clicking on the script and adding to job control. For instructions on creating a job control database, see *section 14*.

To test a propagation script, you will need to make changes to the data in your Ecometry source environment. These changes will result in those changes being captured in the `_q` and `_c1` tables. No data will be moved into the target environment until you run your propagation script.

Validating the data will be a little more difficult. You will need to look at the `queue_q` table to find out what tables were changed and what the changes are. Then you need to look at the `table_c1` to see how the records were changed. You can then use this information to find the changed data in the target tables and verify the results.

To make this process easier, you can develop a set of `makap` scripts to insert, update and delete controlled data into your source database. When these scripts are run, a record is inserted into your source database. That transaction is then updated and then the same transaction is deleted. These changes should be replicated to your target database after you run your propagation script. This will give you the opportunity to run multiple tests that should generate the same results. These scripts will also cause no net change to either your source or target databases while providing you with an opportunity to have controlled test transactions. Please contact your Taurus support representative for a sample script.

It is important to make sure that you test all of the standard transactions, e.g. an insert, update, and delete for each base table and an insert, update, update which turns into an insert, an update which turns into a delete, and delete for each breakout table.

Once you have completed your testing, you should run the count scripts for the queue to make sure the queue is still in sync. If the tables are all still in sync then you are ready to implement the queue in production.

Ecomedate Quick-Start Guide

13. Going live with a queue

In order to bring a queue live, the most important thing is to have a perfect sync point. Without this, your go-live will not be correct and your queue will not be in sync. The steps to bringing a queue live are as follows:

- Make sure both source and target environments have the Warehouse service installed and running
- At a quiet time, with no users logged in or listeners running, back up your production Ecometry environment and create queue objects for the queues that are going live. This is your sync point; this means that all data prior to the back up can be used for initial loads, while all new transactions will be captured by the triggers that have been installed
- Copy your backup of production Ecometry into a static test environment
- Create tables and indexes in your production Ecomedate environment
- Modify open statements in initial load scripts to point to the Ecometry test environment and the production Ecomedate environment
- Run and validate initial loads for the queue
- Modify open statements for propagation scripts to point to your production Ecometry environment and your production Ecomedate environment
- Run propagation script and verify that inserts, updates and deletes for base tables is equal to the number of transactions read in the `table_q` table
- Use BAT files or Windows scheduler to automate the running of the propagation scripts

14. Extras and special circumstances

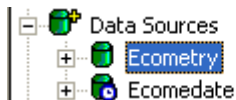
- Creating the job control database
- Handling 1-M-1 models
- Modifying selection criteria for `_OTH` tables

Ecomedate Quick-Start Guide

- Adding new or updated tables after initial implementation
- Data Profiling
- Using Configurations
- BAT files

Creating the job control database:

In order to run jobs in job control within Studio, you must first create your job control database. To do so, choose the environment where you would like your job control database to reside, e.g. the machine you want to run your scripts on. Select that data source, in the example below we selected `Ecomedate`, right-click, and select `Create Control Database`. After clicking `OK`, the data source should then look like this:



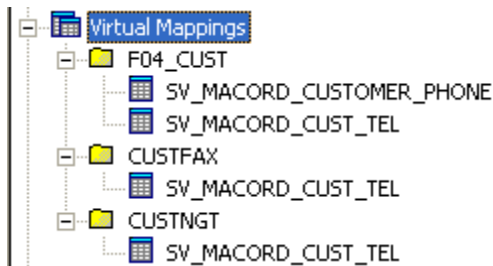
Notice the `Ecomedate` data source now has a symbol that resembles a clock; this indicates that a job control database exists within that data source. The `Ecometry` data source does not have a job control database associated with it.

Handling 1-M-1 models:

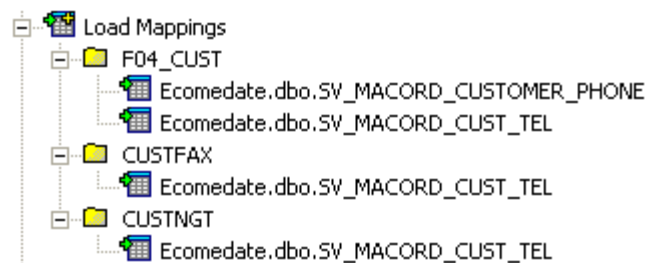
When you have a 1-M-1 situation, such as the `CUST_TEL` table, you'll need to handle it a bit differently in Studio. The `CUST_TEL` table is a normalized table that is written from the `CUSTOMERPHONE` table in `Ecometry`. Three mappings are used to load the table in `Ecomedate`, one each for `day phone`, `night phone` and `fax phone`. Each mapping has the same target table name of `CUST_TEL`. Studio doesn't permit mappings with the same target table name in the same grouping; therefore, we use folders to accomplish the creation of the scripts. We allow the `day phone` mapping to go in the `F04_CUST` folder, but must create separate folders for the `fax` and `night phones`; those mappings

Ecomedate Quick-Start Guide

are then imported into their respective folders. After the virtual mappings have been imported, it should look something like this:



The folders must be created in load mappings as well, and then the virtual mapping for CUSTNGT should be dragged to the folder in load mappings for CUSTNGT; same should occur for CUSTFAX phone. When selecting the target table, CUST_TEL should be selected for both. Once the load mappings are created, they should look like this:

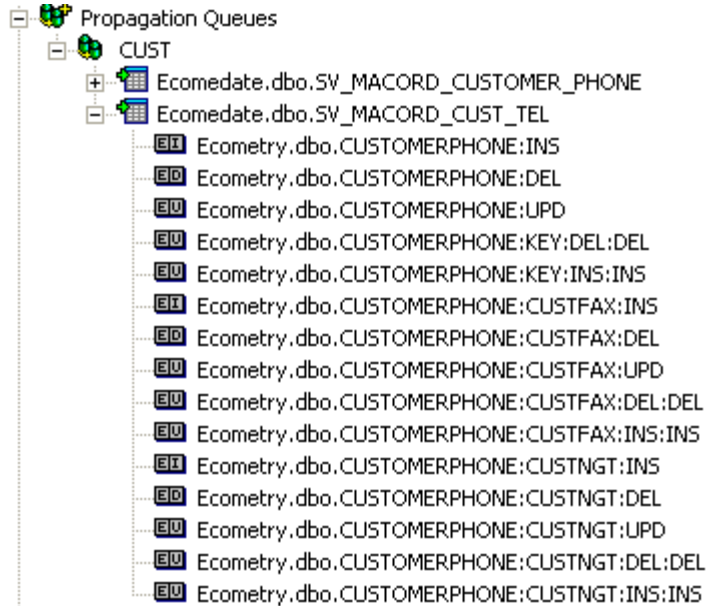


Generating the load script and propagation scripts are done as they normally would be; you just need to make certain that all mappings for CUST_TEL are dragged into the load and propagation scripts. They should look like the following:



And:

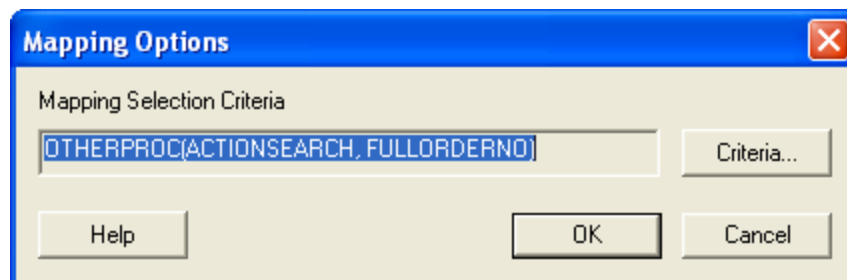
Ecomedate Quick-Start Guide



Modifying selection criteria for `_OTH` tables:

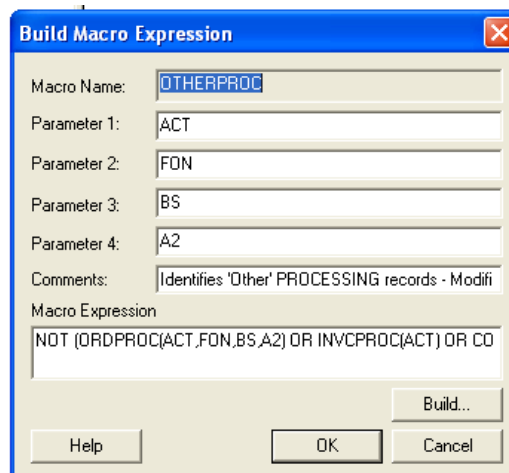
For this example, we will use the previously discussed `Processing` table and assume that we want to have 2 breakout tables, coupons and invoices, and the `_OTH` table in Ecomedate. The virtual mappings should still be edited as usual; verifying keys and data types, then the tables and indexes should be created in Ecomedate. Prior to creating the load mapping is where we'll want to specify the criteria for the `_OTH` table.

To specify the criteria, you'll first want to right-click on the `_OTH` table and select `Mapping Options`. A dialog box should resemble the following:

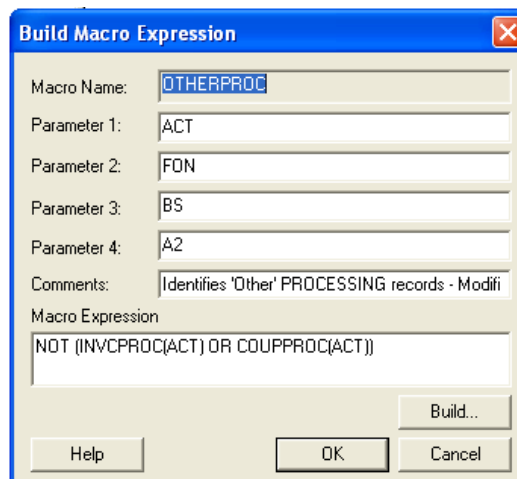


Ecomedate Quick-Start Guide

This tells us that we are using a macro called `OTHERPROC` to determine which types of transactions should be included in the `_OTH` table. This macro needs to be modified. To do this, we go to the `Task` menu and select `Macro Expressions`. When the dialog box with the macro list appears, scroll down to `OTHERPROC`, select it and choose `Edit`. The following dialog box will appear:



Click on `Build...` We want to remove all references to the macros which are **NOT** coupons or invoices. When we are finished modifying the expression, it should look like:



Click `OK` and continue building scripts as you normally would.

Ecomedate Quick-Start Guide

Adding new or updated tables after initial implementation:

There will be circumstances where you will want to either add new tables that are not currently part of your Ecomedate data foundation, or upgrade to new versions of tables that already exist.

For upgrades to new versions, you'll need to do the following:

- At a quiet time, back up your production Ecometry database and truncate the `_q` and `_c1` tables for the affected queue(s).
- Copy the back-up into a static test environment
- Delete the old version of the mapping(s) from the propagation script, load script, load mappings and virtual mappings
- Delete the table(s) in Ecomedate
- Import the new mapping(s) and create the table(s) index(es)
- Drag the virtual mapping(s) into load mapping(s)
- Create the new load script(s)
- Drag the load mapping(s) into the existing propagation script
- Save and run initial load script(s) using your test Ecometry environment and production Ecomedate environment
- Save your new propagation script, making sure to change the open statements to reflect your production Ecometry and Ecomedate environments
- Re-start propagation

If you're adding new tables to Ecomedate, you can either create a new queue with the new tables, or add the tables to the queues they belong to according to the Roadmap. If the new tables are base tables and that particular source table is not used in any of the other queues, you can implement the new tables together in the same newly-created queue. This would be done following the normal instructions for implementing a queue. If, however, your new tables are breakout tables of tables already existing in a live queue or use the same data source as a table in a live queue, you will need to follow the same steps listed above for implementing new versions.

Ecomedate Quick-Start Guide

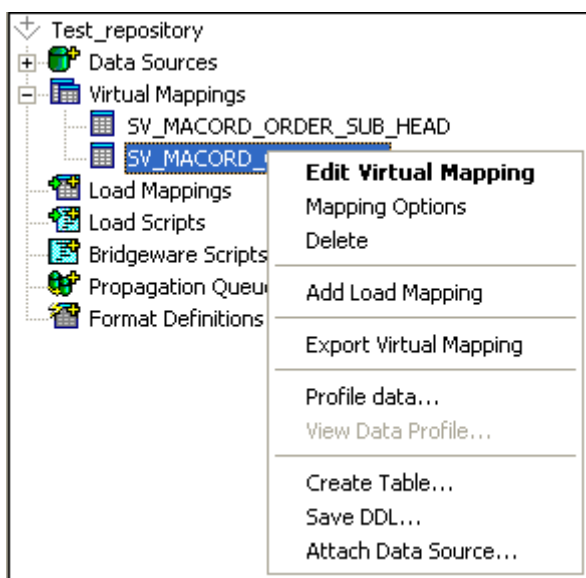
You will also need to modify your reconciliation scripts to add any base tables that you have added to the queue and change any table names that may have changed with a new release.

Data Profiling:

Data profiling allows you to profile your data in order to determine its characteristics. This process will read through the data in a table and gather statistics on the data. This can help you establish unique identifiers as well as indentify fields that may contain bad data.

To use data profiling to determine if a certain combination of fields yields a unique record, you would add an additional field to your mapping that is a concatenation of the fields you would like to check. For instance if you are checking to see if the combination of customer name and zip code produce a unique record in your database, you could add a field called NAME_ADDRESS to your mapping. Then modify the criteria for the field to FNAME || LNAME || ZIP. Save your mapping and then right click on the mapping to create the data profile.

To create a data profile, right click on your virtual mapping, then select Profile Data... from the menu.

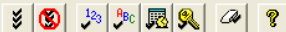


Ecomedate Quick-Start Guide

The dialog box to create a data profile is then presented. Each column in the table has a check box on the far left of the grid. Check the NAME_ADDRESS box.

Create Data Profile

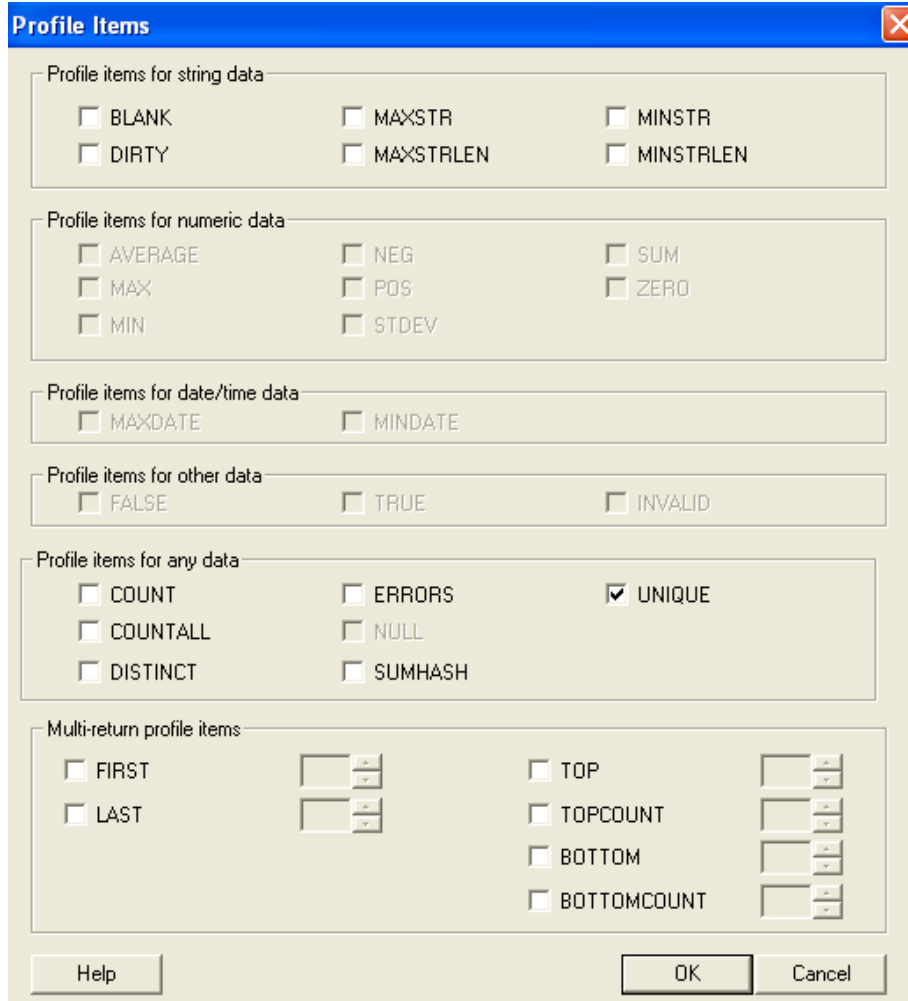
Profile Name: Do Profile Online...
 Profile Options...: Do Profile in Job...

 Profile Items...

Ch	Field Name	Field Type	Field Wic	Null	Key	Expression
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input type="checkbox"/>						
<input checked="" type="checkbox"/>	NAME_ADDRESS	VARCHAR	48			FNAME LNAME ZIP
<input type="checkbox"/>	CHG_LOG_ID	DECIMAL	28			0
<input type="checkbox"/>	SRC_SYS_CHG_LOG_ID	DECIMAL	28	Y		IF(\$SRCOP <> "NEW", \$MSG.SEQ,\$NULL)
<input type="checkbox"/>	SRC_SYS_TRANS_DT	DATE		Y		IF(\$SRCOP = "NEW", \$NOWO, (\$MSG.TXDATE \$MSG.TXTIME))
<input type="checkbox"/>	SRC_SYS_PROC_NM	VARCHAR	30	Y		IF(\$SRCOP <> "NEW", \$MSG.PROGRAM, \$NULL)
<input type="checkbox"/>	SRC_SYS_SESSION_NM	VARCHAR	30	Y		IF(\$SRCOP <> "NEW", \$MSG.SESSION, \$NULL)
<input type="checkbox"/>	SRC_SYS_USER_NM	VARCHAR	30	Y		IF(\$SRCOP <> "NEW", \$MSG.USER, \$NULL)
<input type="checkbox"/>	SRC_SYS_GRP_NM	VARCHAR	30	Y		IF(\$SRCOP <> "NEW", \$MSG.GROUP, \$NULL)
<input type="checkbox"/>	SRC_SYS_ACCT_NM	VARCHAR	30	Y		IF(\$SRCOP <> "NEW", \$MSG.ACCOUNT, \$NULL)

To add profile items, click on the Profile Items... button.

Ecomedate Quick-Start Guide



Profile Items

Profile items for string data

BLANK MAXSTR MINSTR
 DIRTY MAXSTRLEN MINSTRLEN

Profile items for numeric data

AVERAGE NEG SUM
 MAX POS ZERO
 MIN STDEV

Profile items for date/time data

MAXDATE MINDATE

Profile items for other data

FALSE TRUE INVALID

Profile items for any data

COUNT ERRORS UNIQUE
 COUNTALL NULL
 DISTINCT SUMHASH

Multi-return profile items

FIRST
 LAST
 TOP
 TOPCOUNT
 BOTTOM
 BOTTOMCOUNT

Help OK Cancel

The profile items dialog box will be displayed. Items on this dialog box will be available or unavailable based on the data types of checked columns. For example, `BLANK`, which counts the number of blank columns, is only available for character type columns. Likewise, `SUM`, which sums all the values in a column, is only available for numeric items. In order to check for uniqueness, you would check the `UNIQUE` box as well as any other valid boxes. After selecting the desired, profile items, click `OK` to return to the create profile dialog box where the select profile items are displayed on the right of the grid. More columns may then be checked and profile items selected until all columns have the desired profile items. You can then either run the profile online or run it as a job.

Ecomedate Quick-Start Guide

Please see the [DataBridger Application help](#) for further details.

Using Configurations

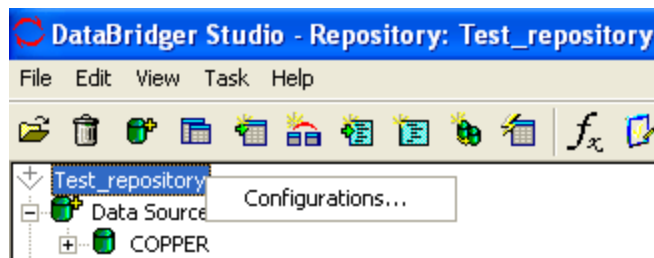
You can easily modify the open statements in your scripts by using the Configurations feature of DataBridger. This allows you to use the same repository with multiple sets of data sources. We suggest you create three different configurations:

Test Ecometry to test Ecomedate for testing – DEFAULT_CFG

Test Ecometry to Live Ecomedate for initial loads – INITIAL_LOAD_CFG

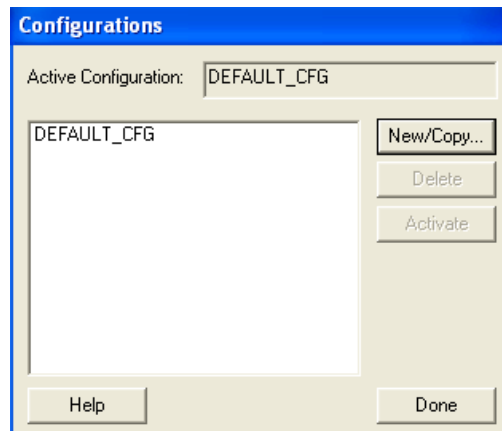
Live Ecometry to Live Ecomedate for propagation – PRODUCTION_CFG

The DEFAULT_CFG is set up automatically with the test data sources you set up when you create your repository. To access the configurations, right click on the name of your repository and click on configurations.

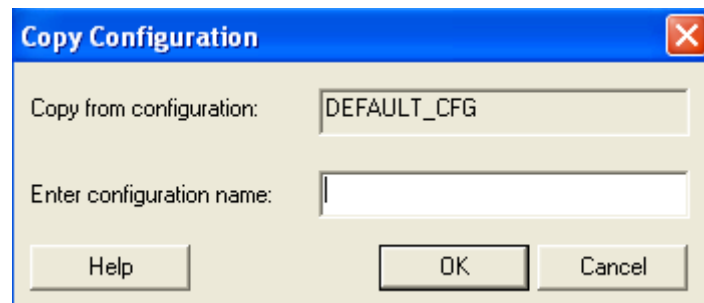


Ecomedate Quick-Start Guide

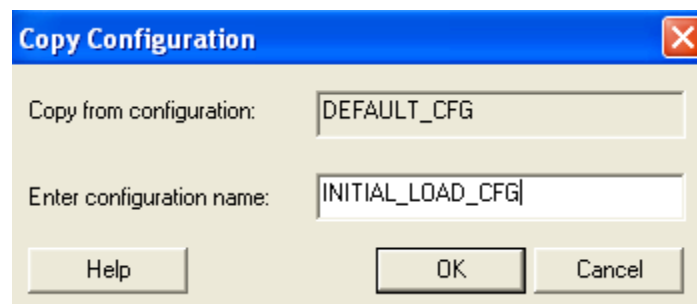
You will see the configurations window with the default configuration already created.



Click on *New/Copy* to create a new configuration to set up for your initial load configuration. This is going to read from the test Ecometry database (to which you should have stored your live account backup prior to initial load) and write to the live Ecomedate database.

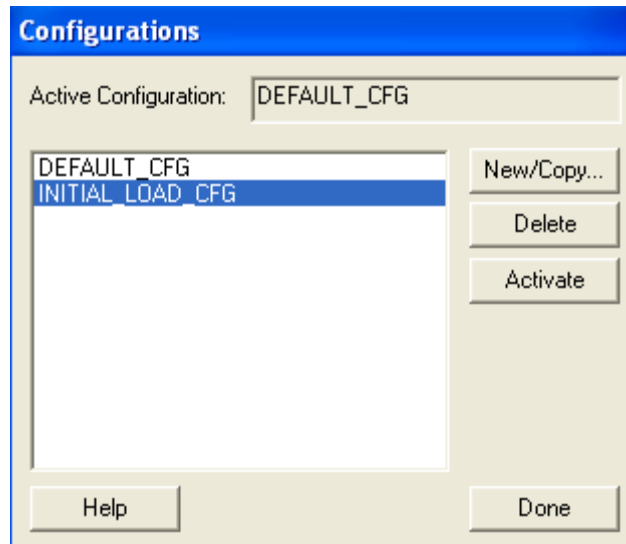


Enter the name of the new configuration (`INITIAL_LOAD_CFG`)

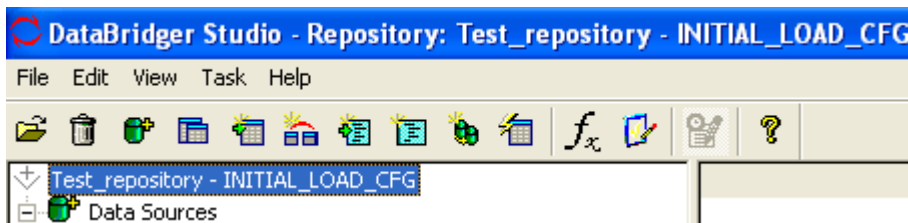


Ecomedate Quick-Start Guide

Click **OK**. Then click on the name of the new configuration to highlight it and click **Activate**.



Next to the name of your repository you should now see the name of the current configuration.



This configuration is simply a copy of the `DEFAULT_CFG`. You will need to modify your data sources to contain connection information for the test Ecometry system and the Live Ecomedate system. Those data source changes will only affect the current configuration.

Repeat the process to create the `PRODUCTION_CFG` which should contain your live Ecometry and your live Ecomedate data sources.

Ecomedate Quick-Start Guide

Bat files:

BAT files to automate the running of the propagation script will be provided to you to help automate running the propagation scripts. A separate set of bat files should be created for each queue. You will need to modify the locations of the files contained within each file as well as the file names to point to the appropriate queues. Use the `_start` file to start the propagation script. There is a `sleep` file that you can modify to include any desired sleep time. To stop the script, use the `_stop` file. Log files are also created log files for the propagation script runs. The log files are named with the time and date the script is run. You should check the log files periodically to make sure the scripts are running correctly

Ecomedate Quick-Start Guide

Conclusion

Perfect reliable data delivery is the cornerstone for accurate operational reporting. If data is not delivered correctly, the Ecometry and Ecomedate environment will not contain the same data. That means any report or down stream data use will be using incorrect data. So, please take the time to ensure that your data delivery is done correctly and that you remain in sync after "go live".

Support is here to support throughout the process. Please do NOT hesitate to open a case with support if you are unclear as what you should do next. To contact support either use the self-serve case management at: <http://www.taurus.com/support/selfserve.htm> or send an email to support@taurus.com.

Good luck!