



Data Warehouse Automation Workshop



Qlik Data Integration Workshop

Data Warehouse Automation

Deliver analytical ready data in the context of real-time Data Integration

Workshop Introduction

Qlik's Data Integration (QDI) solution helps organizations automate and simplify data pipelines to deliver analytic-ready data in the context of real-time data integration. This high-level workshop will help you see the many ways it could benefit your organization.

Our use case is that of a team looking to reduce data pipeline complexity while responding more quickly to the analytical needs of their business. The business is not only asking for access to new and diverse data sets in a timelier manner, but they'd like to see KPI's that are updated within an hour or so. This is accomplished with Qlik's near real-time data delivery and Data Warehouse Automation solution.

The team's first step was to move to a more agile and progressive architecture. They've decided to migrate their decades-old on-premise Data Warehouse to Snowflake, and to leverage a new cloud-based BI solution for better business insights.

Note: in our hands-on exercises we'll use Microsoft SQL Server as our EDW, but QDI's Data Warehouse Automation solution integrates with Snowflake, Azure Synapse, AWS Redshift, Microsoft SQL Server, and Oracle EDW's. Regardless of which EDW you use, QDI delivers real-time analytic data to any BI tool.

When we began this workshop, we briefly discussed what a more progressive data pipeline might look like for you. It began with, as does any data pipeline, getting data out of transactional systems. Traditionally, that's meant performing burdensome batch extracts that, quite often, occupy unacceptable windows and delivers data that ages until the next batch cycle.

A better approach is to stream data out of transactional systems – as it changes – to an EDW where it can then be transformed into an analytic-ready state (more on that later). This allows business data to be delivered to the analytics layer in a timely manner using a low impact, agentless streaming engine.

So, let's begin our workshop with an exercise to help you understand just how simple it is to configure streaming with QDI. Our use case involves a food distribution application running on top of a MySQL OLTP database. Our goal is to continuously move data out of that system, in real-time, as business is conducted, without burdening it.

In the workshop, you will learn how to implement real-time data replication from the MySQL source to the SQL Server data warehouse using Qlik's streaming engine, how to model your data warehouse and data marts, and how to generate and monitor ETL to load the warehouse and marts in near-real time using Qlik's Data Warehouse Automation.

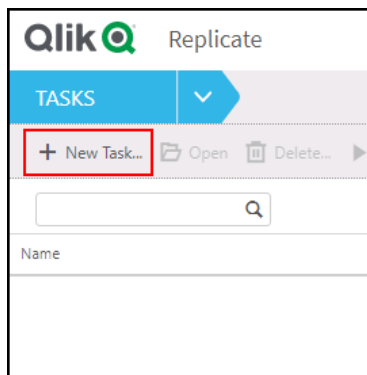
Lab 1: Replicate source data: streaming from MySQL to SQL Server EDW

In this exercise, we'll create a replication task that streams data into a small SQL Server Data Warehouse landing area.

Several things will happen when we run this streaming task:

- Source and target connectivity will be tested and established.
 - The data we'd like to stream will be identified, data type mappings are auto configured, and target objects will be created.
 - An initial target Full Load will be done.
 - After the Full Load begins, the streaming engine will turn its attention to mining the source's transaction logs to capture change data and some DDL it happens, streaming it in near real time to our SQL Server EDW staging area.
1. On your workshop environment screen, you'll see a Chrome browser displaying Qlik's streaming solution's administration console.

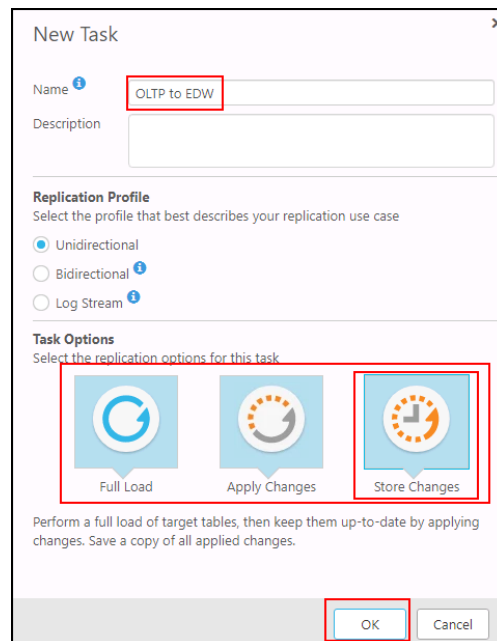
Create a new replication task by **selecting the New Task** button at the top left:




2. There are choices available under Task Options:

- The task can instantiate and do a *Full Load* to the target.
- The task can Full Load, as above, and will also continuously *Apply Changes* to stream data into the target (i.e. perform CDC) and ...
- The task can utilize *Store Changes* to create target 'sister' tables that track and audit each applied DML and DDL operation. In the case of Data Warehouse Automation, these 'Store Changes' tables are critical to creating and managing Type 2 attributes to handle history data.

3. Name the task **OLTP to EDW** and select all three Task Options as you see below. The Option will highlight itself in blue once selected:





New Task


Name  OLTP to EDW

Description

Replication Profile
Select the profile that best describes your replication use case

☒ Unidirectional 

☐ Bidirectional 

☐ Log Stream 

Task Options
Select the replication options for this task

☒ Full Load ☒ Apply Changes ☒ Store Changes

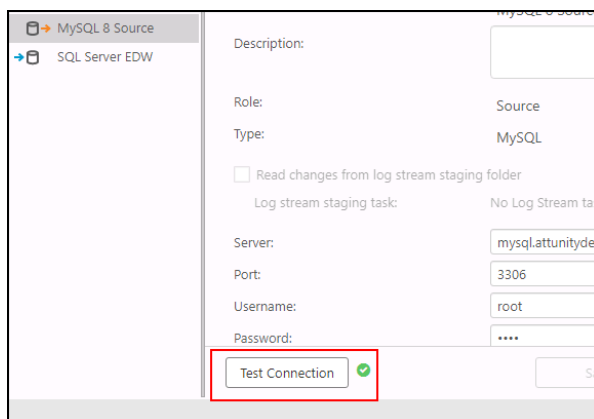
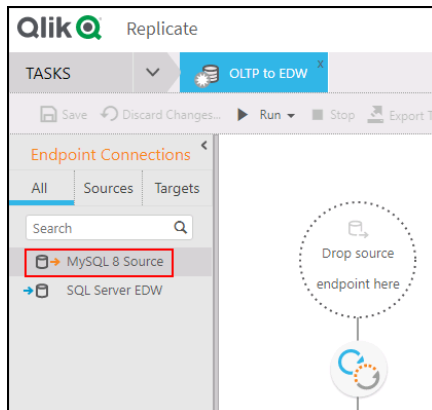
Perform a full load of target tables, then keep them up-to-date by applying changes. Save a copy of all applied changes.

OK Cancel

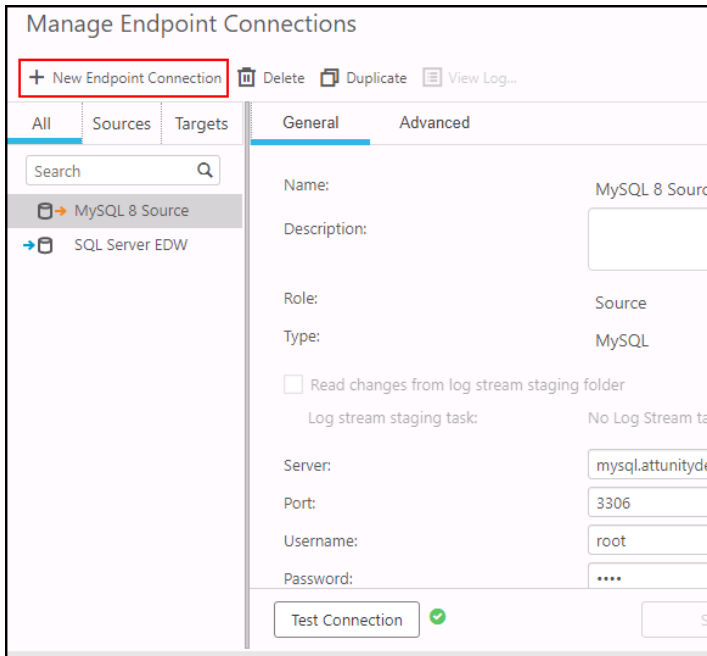
Select **OK** to further configure our streaming task.

4. In the next screen, in the left pane, you should see our planned stream source and target.

Before you select them, **double click the MySQL** endpoint to see the connectivity information required and test the connection to check whether the replication server can connect to it:



- Review all the possible sources and targets available for streaming by selecting **New Endpoint Connection**:



Manage Endpoint Connections

+ New Endpoint Connection Delete Duplicate View Log...

All Sources Targets

Search

MySQL 8 Source

SQL Server EDW

Name: MySQL 8 Source

Description:

Role: Source

Type: MySQL

☐ Read changes from log stream staging folder

Log stream staging task: No Log Stream task

Server: mysql.attunitydev

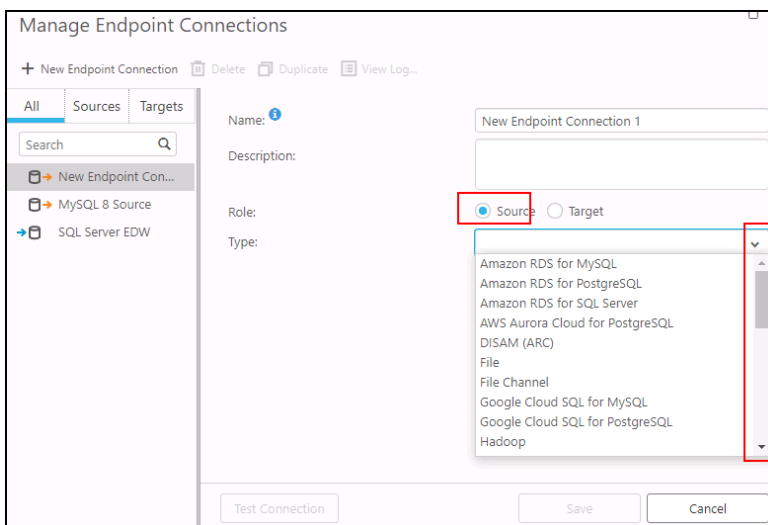
Port: 3306

Username: root

Password: ****

Test Connection

- Explore the available sources and targets (by enabling the radio buttons) and then **click Cancel** and then **Close** to return to our streaming task. Do not create a new endpoint:



Manage Endpoint Connections

+ New Endpoint Connection Delete Duplicate View Log...

All Sources Targets

Search

New Endpoint Con...

MySQL 8 Source

SQL Server EDW

Name: New Endpoint Connection 1

Description:

Role: Source Target

Type:

Amazon RDS for MySQL

Amazon RDS for PostgreSQL

Amazon RDS for SQL Server

AWS Aurora Cloud for PostgreSQL

DISAM (ARC)

File

File Channel

Google Cloud SQL for MySQL

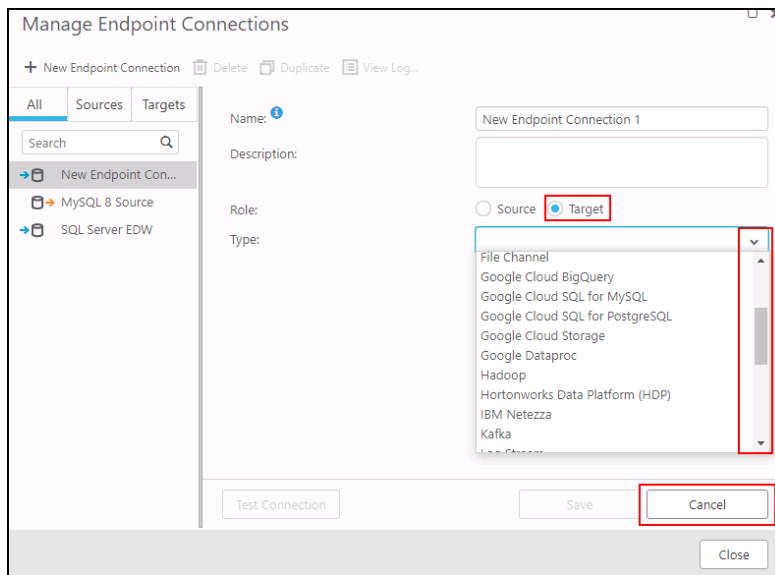
Google Cloud SQL for PostgreSQL

Hadoop

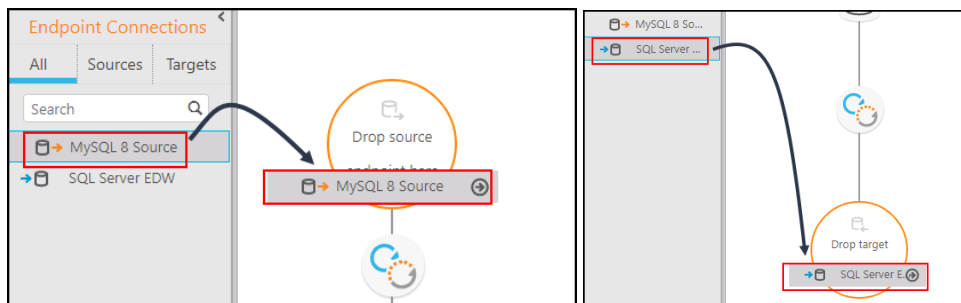
Test Connection

Save

Cancel

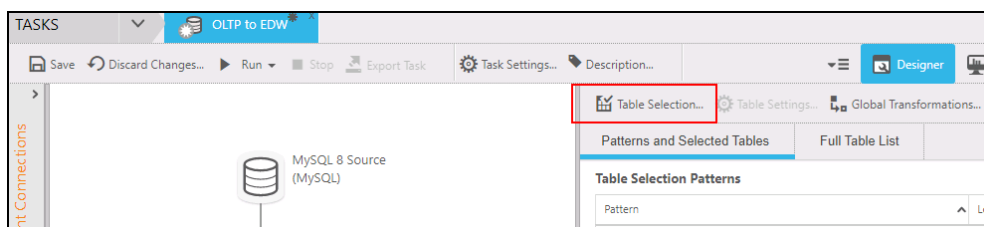


- Once returned to the task, **drag and drop the MySQL source** onto the top of the diagram labeled 'Drop source endpoint here', then **drag and drop the MS SQL Server target** where it is labeled 'Drop target endpoint here':

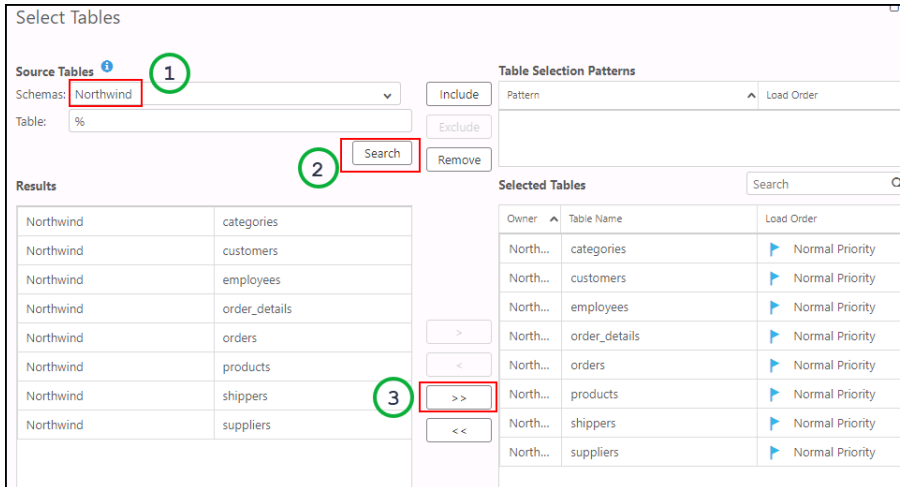


Note: you will know when to 'drop' the source and target when the circle you're hovering over changes from dashes to a solid orange.

- Now that you've selected the source and target, you'll select the source tables to include in your task by **clicking Table Selection**:



9. Select the **Northwind source schema**, then click '**Search**' (% is the wildcard character) and then **move all** the tables to the pane on the right:



Select Tables

Source Tables

Schemas: Northwind

Table: %

Table Selection Patterns

Pattern: Load Order

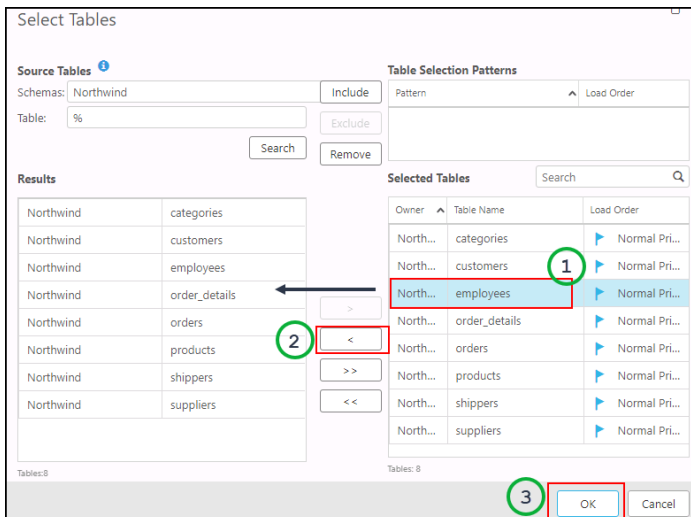
Results

Owner	Table Name
Northwind	categories
Northwind	customers
Northwind	employees
Northwind	order_details
Northwind	orders
Northwind	products
Northwind	shippers
Northwind	suppliers

Selected Tables

Owner	Table Name	Load Order
North...	categories	Normal Priority
North...	customers	Normal Priority
North...	employees	Normal Priority
North...	order_details	Normal Priority
North...	orders	Normal Priority
North...	products	Normal Priority
North...	shippers	Normal Priority
North...	suppliers	Normal Priority

10. **Remove the Employees table** as it will not be part of our analytic model and **click OK** to complete the selection:



Select Tables

Source Tables

Schemas: Northwind

Table: %

Table Selection Patterns

Pattern: Load Order

Results

Owner	Table Name
Northwind	categories
Northwind	customers
Northwind	employees
Northwind	order_details
Northwind	orders
Northwind	products
Northwind	shippers
Northwind	suppliers

Selected Tables

Owner	Table Name	Load Order
North...	categories	Normal Pri...
North...	customers	Normal Pri...
North...	employees	Normal Pri...
North...	order_details	Normal Pri...
North...	orders	Normal Pri...
North...	products	Normal Pri...
North...	shippers	Normal Pri...
North...	suppliers	Normal Pri...

Tables: 8

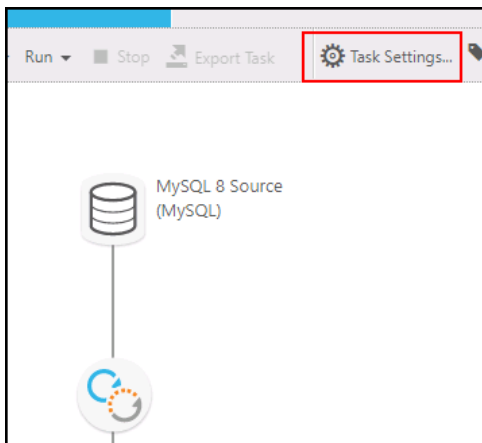
Tables: 8

OK **Cancel**

11. The last step before we run the task and start streaming is to make a metadata change. The source data schema name is 'Northwind' and, if we do nothing more, when the task is run it will create a target EDW schema of the same name.

In this step we'll specify a more meaningful name.

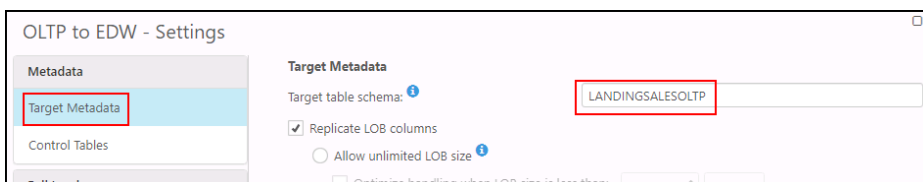
In our main screen **select the Task Settings** button:



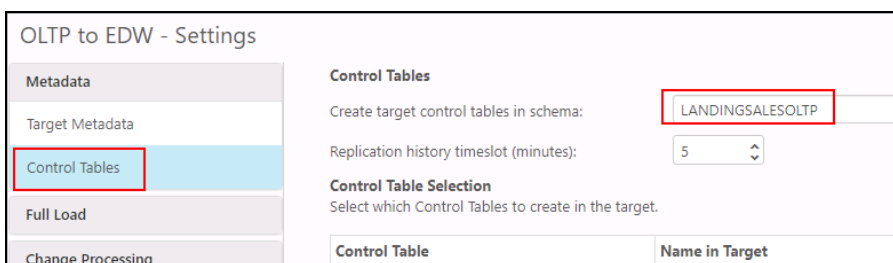
12. On the 'Target Metadata' tab (default) on the resulting screen **type LANDINGSALESOLTP** into the *Target table schema* field.

Ensure that you do so in ALL CAPS and precisely named as:

L A N D I N G S A L E S O L T P

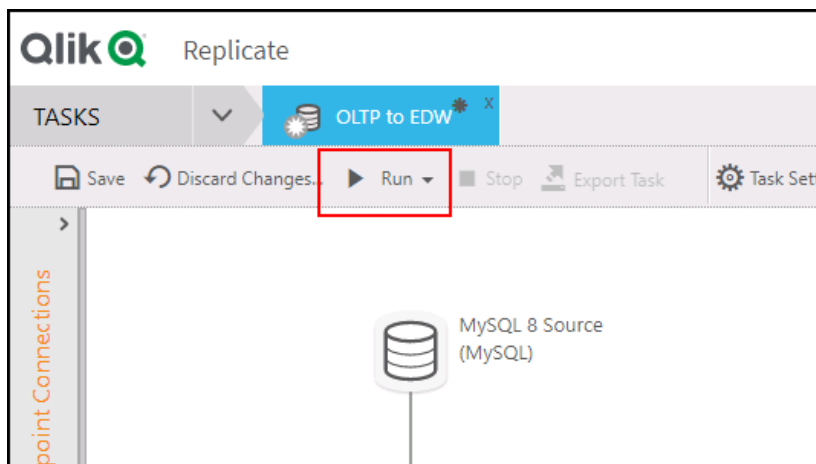


13. **Copy this value into your clipboard** and **paste it** into the *Create target control tables in schema* field on the 'Control Tables' tab:

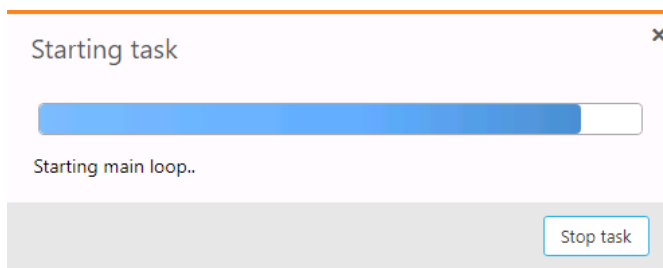


14. **Click OK** to complete the configuration of your task.

15. **Click the Run button** to perform the initial load and ongoing CDC streaming:



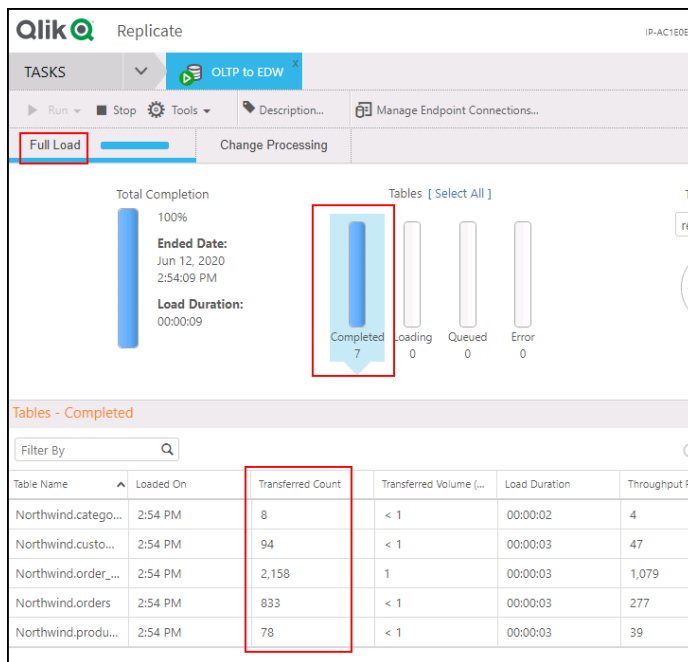
The GUI will display that it's starting the task and will automatically take you to the Monitor screen for the task (toggle from Designer to Monitor on top right of GUI).



16. Because we're dealing only with a small number of source rows, your initial load should complete in under a minute.

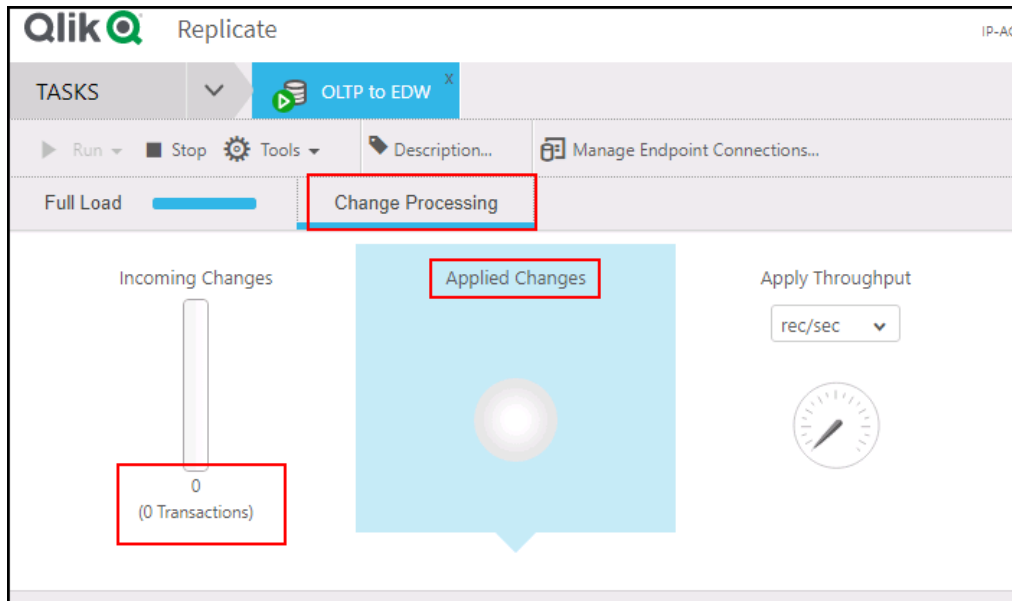
In the resulting screen you'll see a few 'Full Load' job stats on the Monitor screen.

Select the "Completed" bar for more information about the data loaded via this Full Load batch into the SQL Server EDW landing area.



17. Now that the Full Load (select * from source table) is completed, the streaming engine is now monitoring the OLTP transaction logs to quickly stream committed transactions and DDL to your SQL Server EDW landing area.

Click on the “Change Processing” tab to see what data it is now streaming:



18. No data is streaming because we have yet to transact any business (we'll do that later).

But note that Qlik's streaming engine immediately transitioned to CDC mode once finished with the Full Load.

Lab 1 Outcome: Data is extracted from source and ready to load into the Data Warehouse!

Qlik's Data Warehouse Automation Solution

Now that you're prepared for streaming, we can turn our attention to the SQL Server EDW.

Analytics cannot be accomplished simply by landing data onto a Data Warehouse server, regardless of whether we do so in batch or real-time. This data is in a raw transactional state and must be transformed from that raw form to a "modeled" state for analytic consumption.

At this point, Qlik's Data Warehouse Automation solution picks up our streamed data and shepherds it through that "last mile" to the visualization layer. You'll see in the upcoming labs just how easily and quickly Qlik's Data Warehouse Automation accelerates your efforts to:

- More rapidly design your EDW
- Stand up the Data Warehouse
- Automate GUI-driven transformation logic to populate the Data Warehouse
- Load the DW
- Incrementally update the DW to ensure a current view of your business
- Provision on-demand, real-time, Data Marts for deeper insights and simpler delivery
- Evolve and refactor your pipeline more rapidly when business requirements change

Unlike Qlik's streaming engine, Qlik's Data Warehouse Automation solution does not access, process, or otherwise transform your data on a separate server. Rather, it generates, orchestrates, and executes – pushes down – the ETL code to the Data Warehouse database. Landed data is thus processed and transformed by leveraging the power of the EDW you use (e.g. SQL Server, Redshift, Snowflake, Synapse, Oracle, or, in this case, SQL Server).

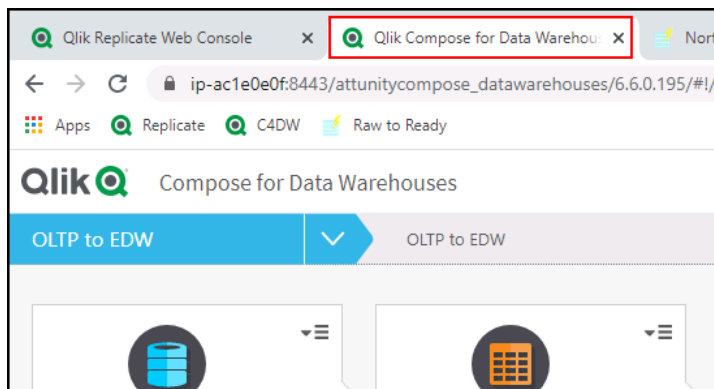
Qlik's Data Warehouse Automation solution is used to design and coordinate the Analytic component in your data pipeline; that is, to design the Data Warehouse, automate the transformation logic, and provision on-demand, real-time Data Marts.

Lab 2: Data Warehouse Automation – Model Creation

One of the first steps in building out your Data Warehouse is to establish your Data Model from the data landed by the streaming CDC and organizing your business and reference data in such a way as to optimize its use in analytics.

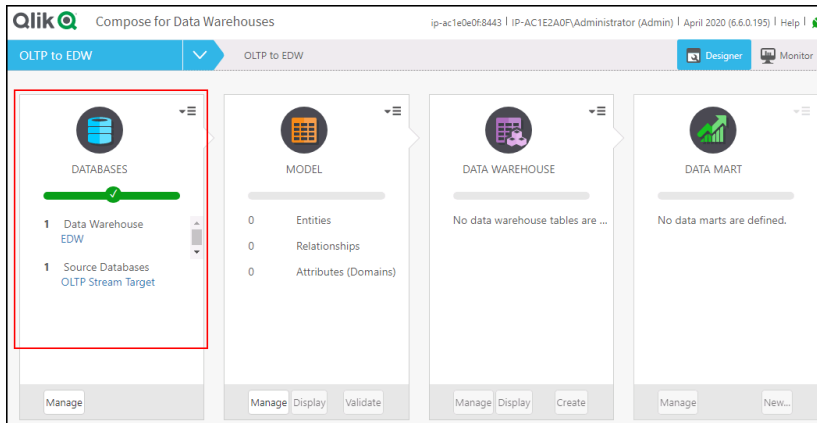
In this exercise you'll explore the advantage that Qlik's solution affords you in accelerating your Data Warehouse Modelling process.

1. Navigate from the streaming engine console and **select the second Chrome tab** labeled 'Qlik Compose for Data Warehouses' at the top of your screen.



2. Here you'll find the Data Warehouse Automation's Console (Designer tab at the right) where Data Engineers and Architects build out and implement their Data Warehouse and various downstream Data Marts.
3. In the Design console you'll see four panes, each with a specific purpose:

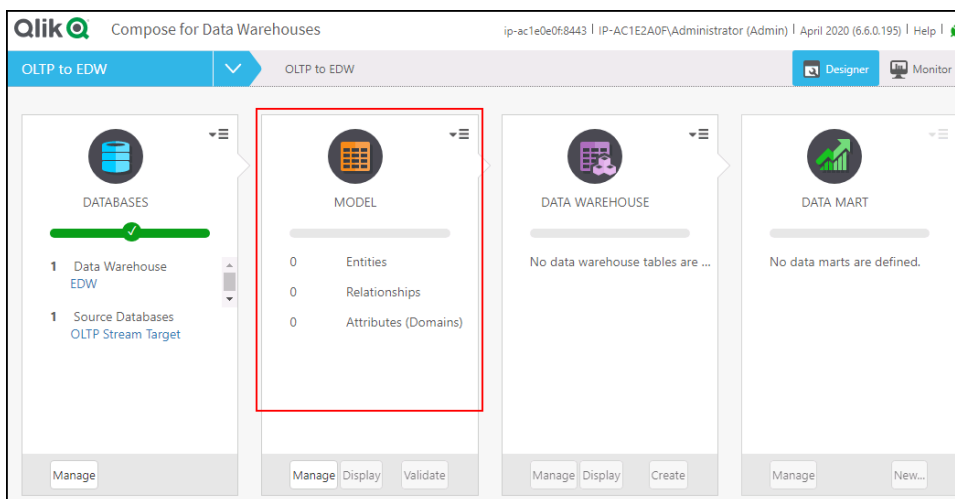
The Databases section is where we establish connectivity to the data stores/sources to be used in the data warehouse, and specify where the Data Warehouse, Error Mart, and Data Marts themselves will be created.



The Model pane is where we manage designing and building our Model.

The connectivity defined in the Databases section helps us reverse engineer or “Discover” the metadata relationships behind the data in our landing areas (usually streamed from multiple sources) so we can get a significant jump-start in our model design.

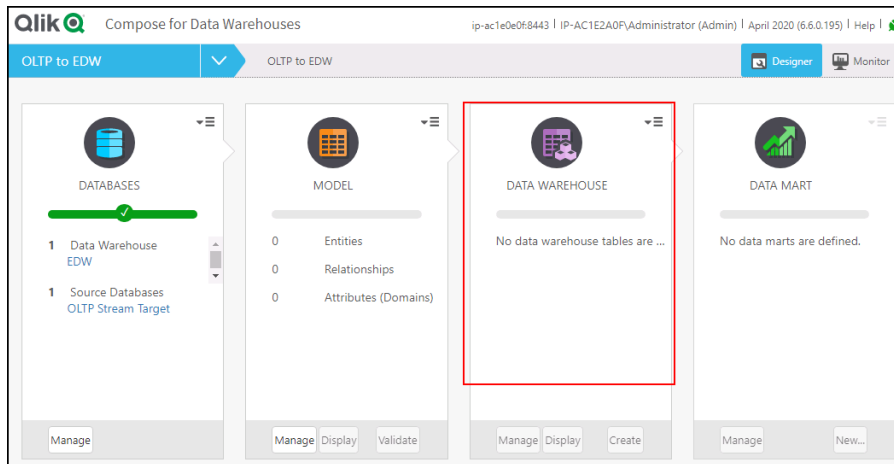
Models can also be imported from ErWin or specially formatted spreadsheets, but we will not be covering that in this workshop.



4. Once we're finished designing the Model, we manage and implement it using the Data Warehouses panel.

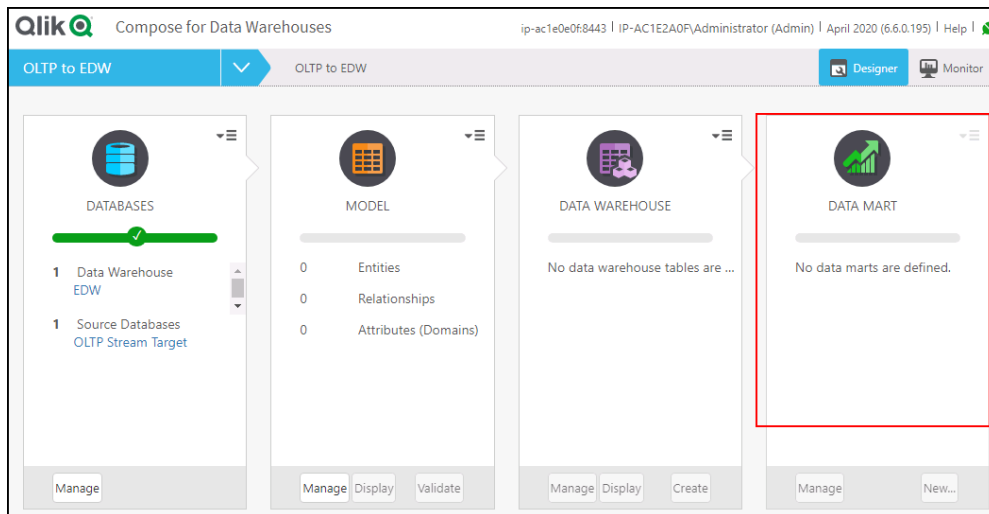
Here we can customize data mappings and create transformations.

We then auto-generate the DDL and ETL code necessary to create and load the Data Warehouse based on our customizations. We also create ETL to update the Data Warehouse with change data in defined intervals throughout the day.



5. Finally, once we've instantiated and populated the Data Warehouse, we make it available to Data Engineers and Analysts to provision real-time Data Marts.

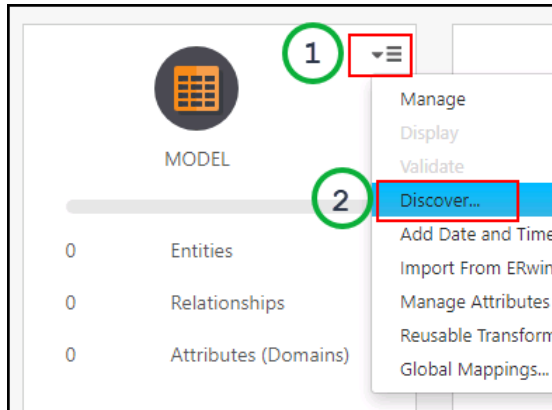
You'll see later how this is done using a very intuitive 'Star Schema Wizard'.



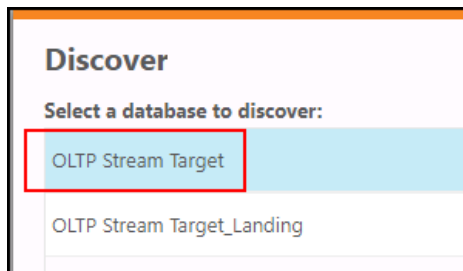
6. Not shown here, but something we'll also visit later, is Qlik's workflow engine. This is located within the Monitor tab, where we can create workflows for operations designed and created in the Designer tab.

As an example, we might orchestrate the EDW ELT jobs designed to ingest data that Qlik's streaming engine has landed in the last 30 minutes, transform it to conform to our model, and then load that change data into our EDW, wait 30 minutes and then repeat.

We'll now explore Qlik's Data Modelling capabilities. In the 'Model' panel, select the drop-down menu and **select "Discover"**:



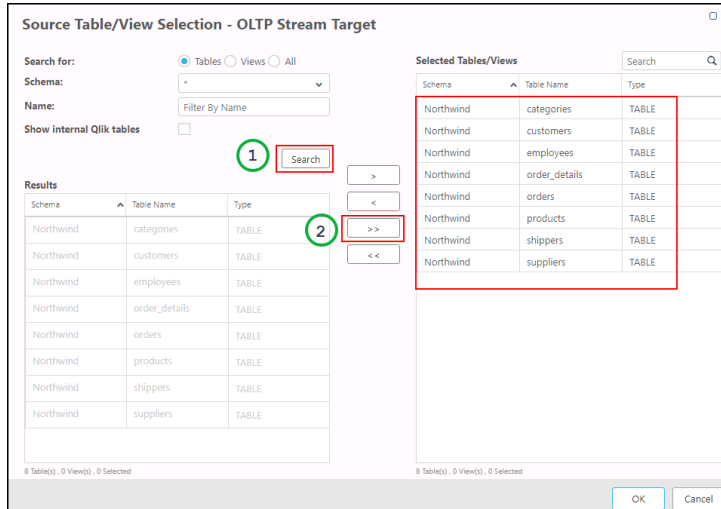
In the resulting screen, **choose the top entry** and **click OK**



Despite what its name implies, the 'OLTP Stream Target' is a connection to our source OLTP database. In the Discovery process, what we're looking to do is reverse engineer the batch and streaming data landed into our EDW so we can determine the relationships between their tables. We do this by analyzing the metadata from the source to help us start the data modelling process.

- On the next screen **click the 'Search' button** to pull in this metadata from our source. This will examine the metadata from our OLTP source so we can begin the modelling process.

Move all the source tables to the right pane:



Source Table/View Selection - OLTP Stream Target

Search for: ☒ Tables ☐ Views ☐ All

Schema:

Name:

Show internal Qlik tables ☐

Results

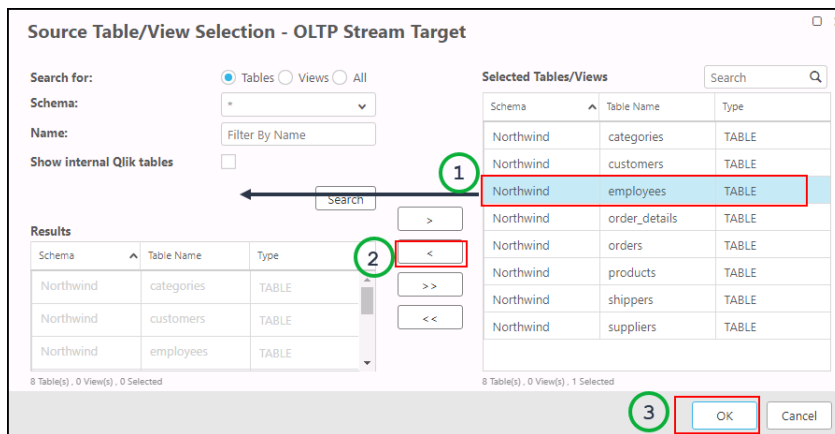
Schema	Table Name	Type
Northwind	categories	TABLE
Northwind	customers	TABLE
Northwind	employees	TABLE
Northwind	order_details	TABLE
Northwind	orders	TABLE
Northwind	products	TABLE
Northwind	shippers	TABLE
Northwind	suppliers	TABLE

Selected Tables/Views

Schema	Table Name	Type
Northwind	categories	TABLE
Northwind	customers	TABLE
Northwind	employees	TABLE
Northwind	order_details	TABLE
Northwind	orders	TABLE
Northwind	products	TABLE
Northwind	shippers	TABLE
Northwind	suppliers	TABLE

OK Cancel

- Remove the employees table** since it is not part of our analytics requirements:



Source Table/View Selection - OLTP Stream Target

Search for: ☒ Tables ☐ Views ☐ All

Schema:

Name:

Show internal Qlik tables ☐

Results

Schema	Table Name	Type
Northwind	categories	TABLE
Northwind	customers	TABLE
Northwind	employees	TABLE

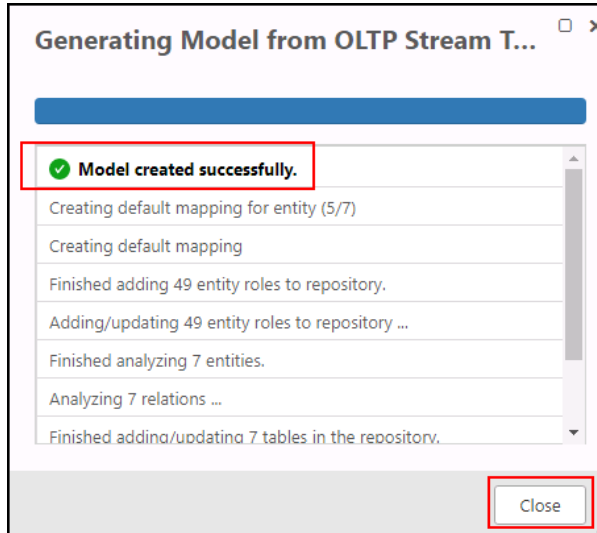
Selected Tables/Views

Schema	Table Name	Type
Northwind	categories	TABLE
Northwind	customers	TABLE
Northwind	employees	TABLE
Northwind	order_details	TABLE
Northwind	orders	TABLE
Northwind	products	TABLE
Northwind	shippers	TABLE
Northwind	suppliers	TABLE

OK Cancel

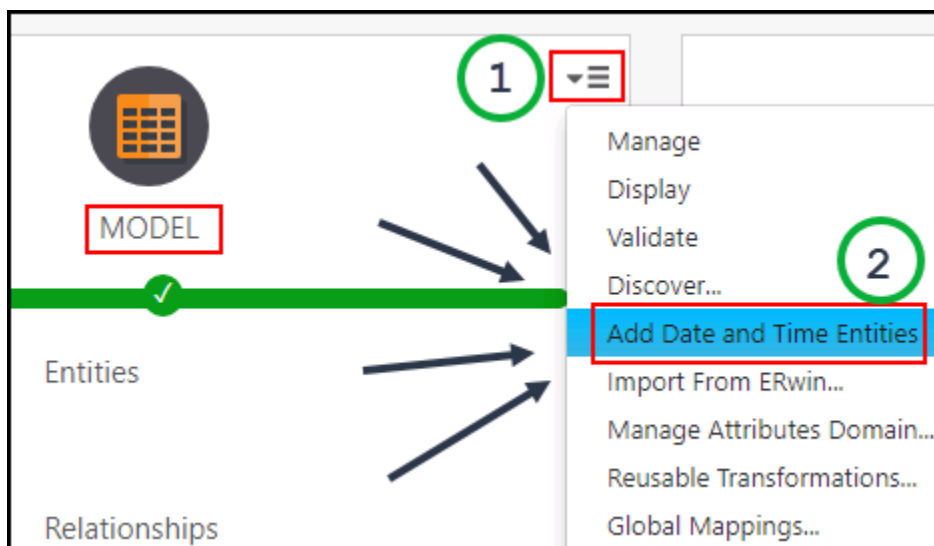
9. **Click OK**

The discovery process will complete successfully, then **click Close**:

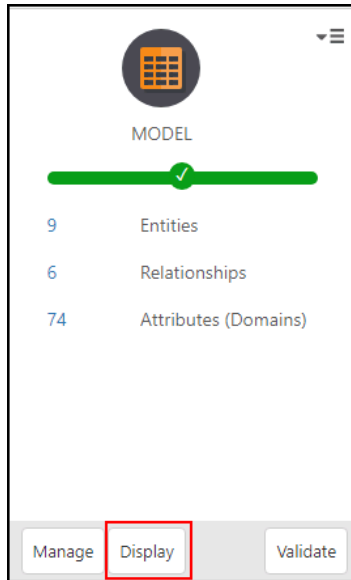


10. Because Data Warehouses typically include Date/Time dimensions, we'll generate them now.

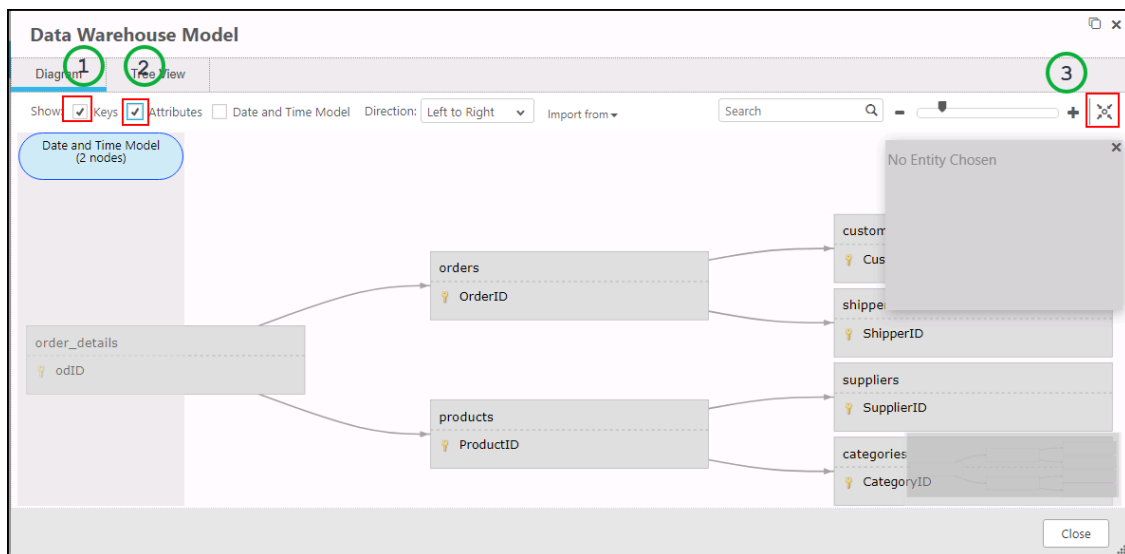
Return to the drop-down menu in the MODEL pane and **select 'Add Date and Time Entities'** and **click OK**:



11. To see our new model with Date and Time entities included, select the 'Display' button at the bottom of the Model pane:



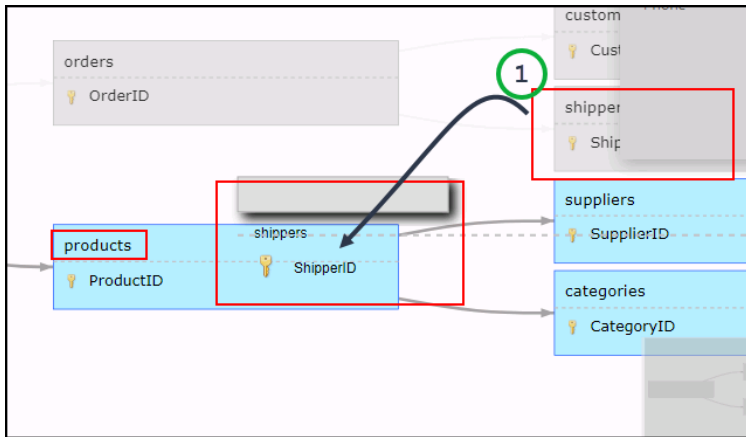
12. On the resulting screen, you'll see a simple graphical depiction of our model showing Entities and Relationships. In order to more closely inspect it, select the 'Keys' and 'Attributes' options at the top left and fit the newly rendered diagram to your screen if necessary, by clicking the auto scale button on the top right portion of your screen:



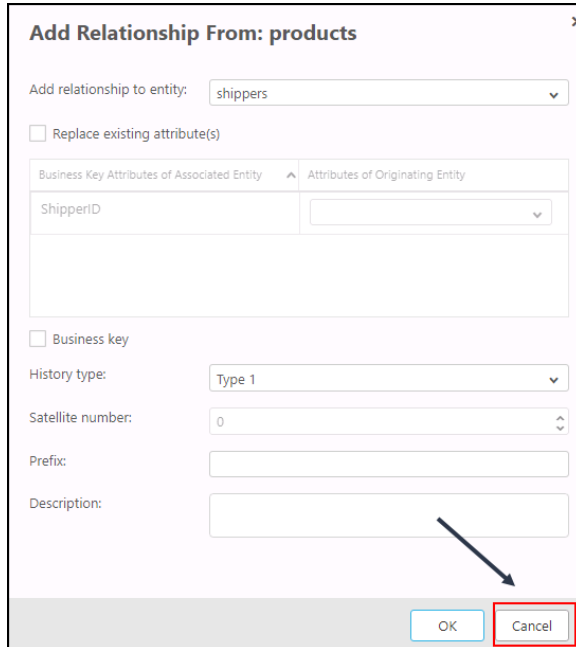
13. Click on few of the entities (e.g. orders, products, etc.) and take note of the display of key relationships and attributes brought in from the Discover process and modelled.
14. Suppose that there's a relationship between 'products' and 'shippers' that was not defined on the source but needs to be defined for analytical purposes.

We can manually define relationships in the model.

Drag and drop the 'shippers' entity onto the 'products' entity.



15. The resulting dialogue box allows us to customize the relationship in the likely event that not all our source OLTP structure are fully comported with how that data needs be structured in our Model:



In real world scenarios, there'd most certainly be more than one source and the modelling process would be more challenging. In cases like that, we'd repeat the 'Discovery' exercise for each source, adding more entities to the candidate Model. We'd then use this visualization capability to establish our relationships.

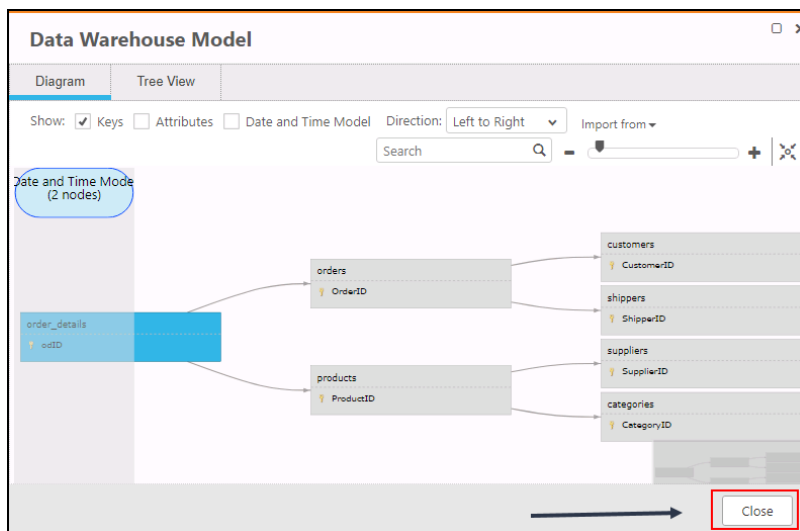
Cancel out of this dialogue.

16. The workshop leader will now go through a brief demo of how this model can be customized and then how the autogenerated ETL code can be customized.

Please stop here and wait for that demo to begin.



17. After the demo has completed, **close the Model display screen** to return to our main console:



18. At this point, your model is ready for implementation.

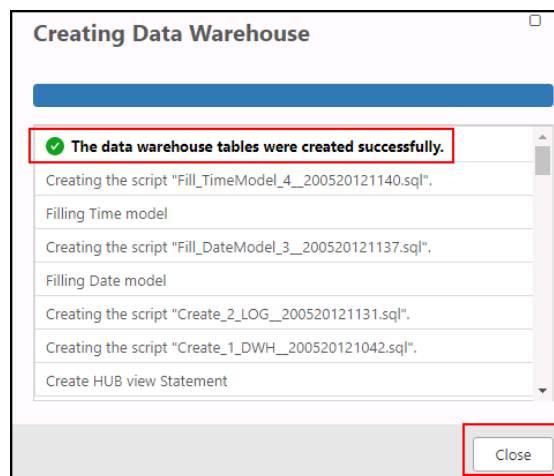
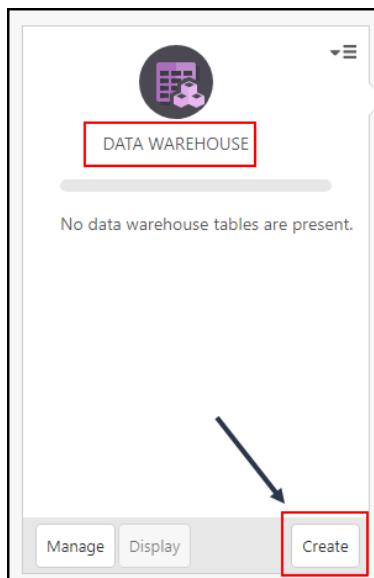
Lab 2 Outcome: Project is connected to source and target; model is generated and customized to suit the business needs.

Lab 3: Data Warehouse Automation – Build and Load the Warehouse

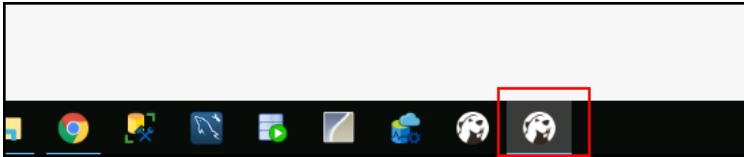
Once our Model is complete, we can generate and execute the SQL Server DDL necessary to create our Data Warehouse from this Model.

1. To create the Data Warehouse **click the 'Create' button** in the Data Warehouse pane and wait for completion. This should take about 15 seconds

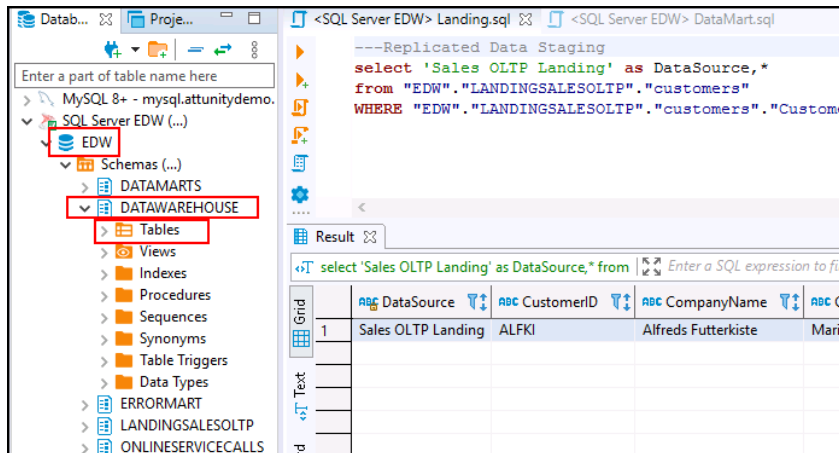
Once complete **click Close**:



- To see these (empty) Data Warehouse tables, open the running SQL Editor in your Windows taskbar (at the bottom of your screen) by clicking the icon seen here:



In the resulting screen expand the DATAWAREHOUSE schema and then review its tables:



- When complete, return to your browser.

Autogenerate ELT Code

One of the difficulties in building a data pipeline is the task of transforming raw transactional data into an analytic-ready state. The traditional ETL coding process is often complex, error prone, repetitive, and not easily nor quickly adapted to meet the ever-changing needs of the business.

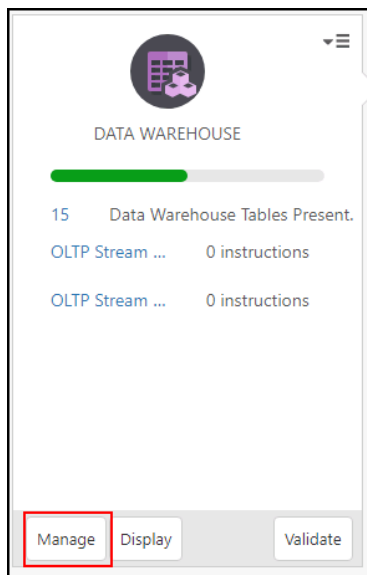
Qlik's Data Warehouse Automation solution automates much of this transformation logic; all implemented as ELT code.

Specifically, what this means is that SQL code (i.e. ELT code) is generated, scheduled, and then executed throughout the day via workflow jobs, leveraging the compute power of your EDW (whatever that is) instead of proprietary ETL middleware performing these tasks. Data is periodically pulled from the EDW landing area which, in turn, is continuously fed by the low-impact streaming capabilities of Qlik's streaming engine.

The next step of this lab is to autogenerate the ETL.

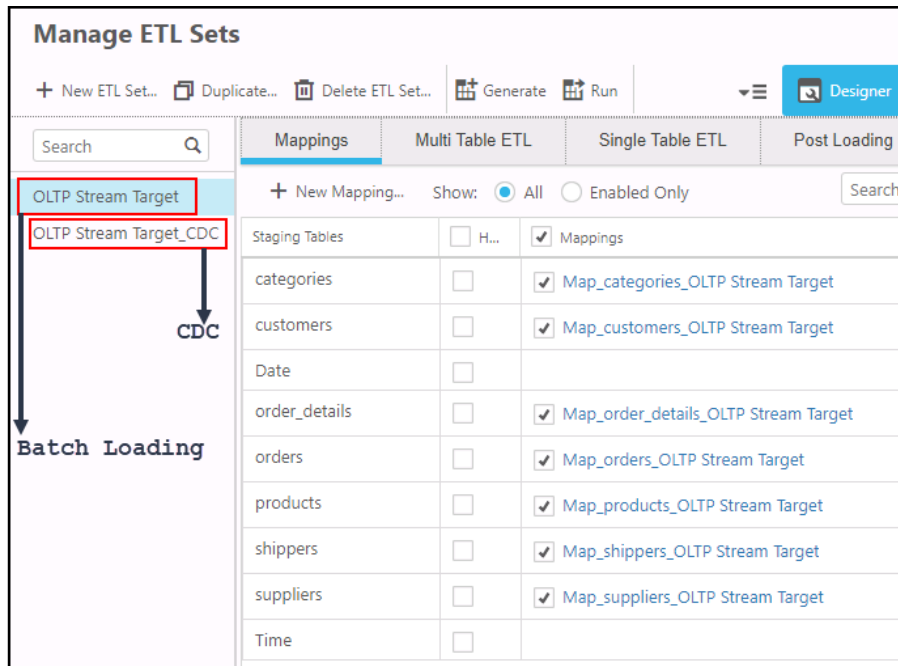
4. **Close the 'Creating Data Warehouse' dialogue** if it is still open.

In the 'Data Warehouse' pane **click the 'Manage' button** to see the auto-generated mappings:



- On the left pane of the resulting window there should be two sets of ETL mappings:

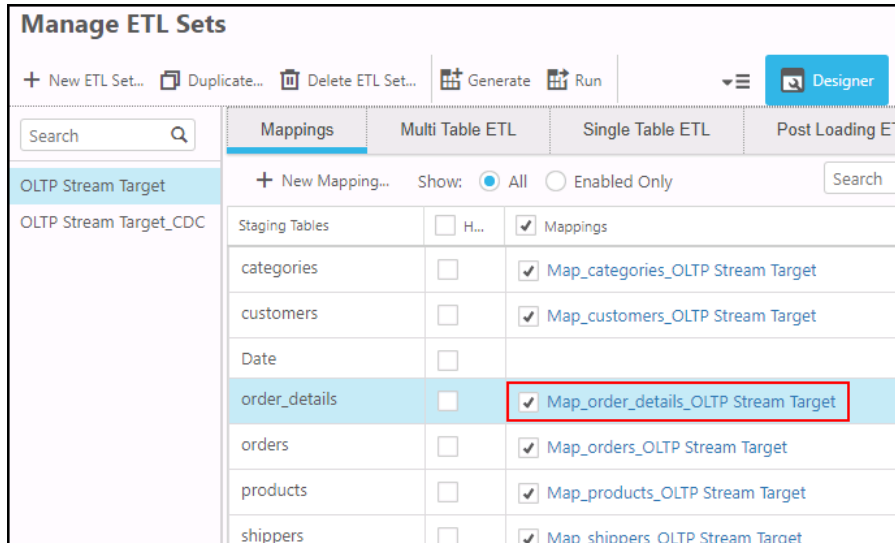
One code-set (the top entry) is for the initial batch load of the Warehouse, and the second code-set is run in user-determined intervals throughout the day to update the EDW with data delivered by the streaming engine (CDC):



It's important to note that as difficult as it is to write batch ETL code, it is often orders of magnitude more difficult to write code to process change data. What's most valuable about Qlik's solution is that it completely abstracts this complexity so you can customize your transformations logically, visually and, thus, much more easily.

6. Let's explore an autogenerated ETL mapping

Click to open the “Map_order_details_OLTP Stream Target” mapping for the order_details table:

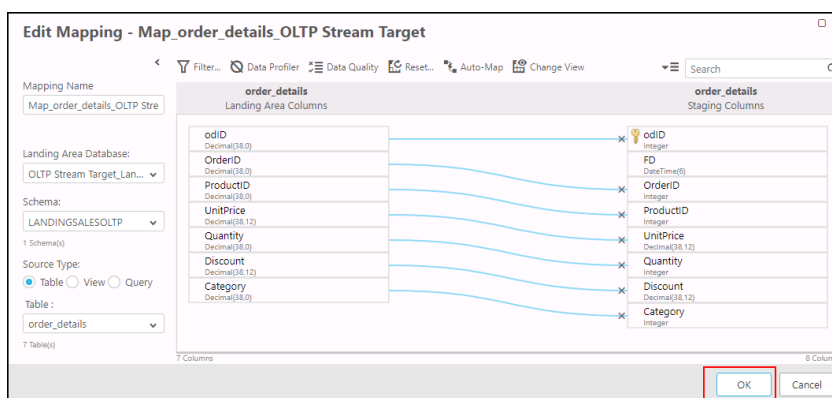


Manage ETL Sets			
+ New ETL Set... Duplicate... Delete ETL Set... Generate Run			
Search	Mappings	Multi Table ETL	Single Table ETL
OLTP Stream Target	+ New Mapping... Show: <input checked="" type="radio"/> All <input type="radio"/> Enabled Only Search		
OLTP Stream Target_CDC	Staging Tables	<input type="checkbox"/> H...	<input checked="" type="checkbox"/> Mappings
	categories	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_categories_OLTP Stream Target
	customers	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_customers_OLTP Stream Target
	Date	<input type="checkbox"/>	
	order_details	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_order_details_OLTP Stream Target
	orders	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_orders_OLTP Stream Target
	products	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_products_OLTP Stream Target
	shippers	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_shippers_OLTP Stream Target

7. The blue lines indicate auto mapping of landing area columns to the EDW staging columns. These are very straightforward, one-to-one mappings; very common in ETL coding but also very error prone. In typical business scenarios Qlik's solution would automate anywhere from 80 to 90 percent of your ELT code with the remainder easily customized as shown here.

In the brief demo earlier, you saw how we could further customize our transformation logic.

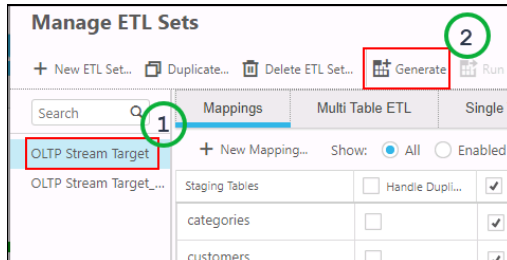
No changes need to be made here so simply click OK



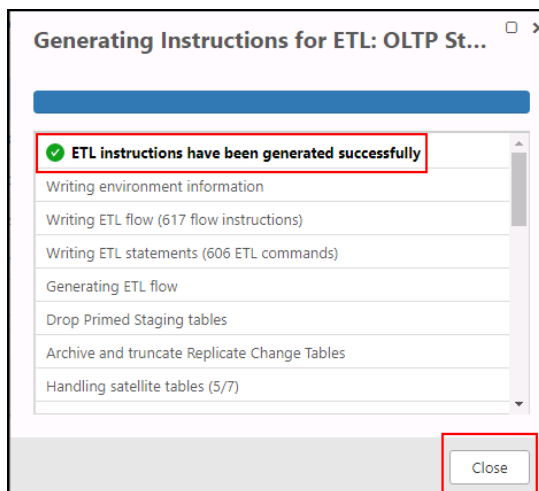
Edit Mapping - Map_order_details_OLTP Stream Target			
Mapping Name		order_details	order_details
Map_order_details_OLTP Stre		Landing Area Columns	Staging Columns
Landing Area Database: OLTP Stream Target_Lan... Schema: LANDINGSALESOLTP Source Type: <input checked="" type="radio"/> Table <input type="radio"/> View <input type="radio"/> Query Table: order_details	odiID	odiID	odiID
	OrderID	FD	OrderID
	ProductID	ProductID	ProductID
	UnitPrice	UnitPrice	UnitPrice
	Quantity	Quantity	Quantity
	Discount	Discount	Discount
	Category	Category	Category

8. Auto-generate the ETL code that will be used for the initial batch load, which will hydrate our EDW.

Select the OLTP Stream Target batch code set (the top entry) and then click **Generate**:



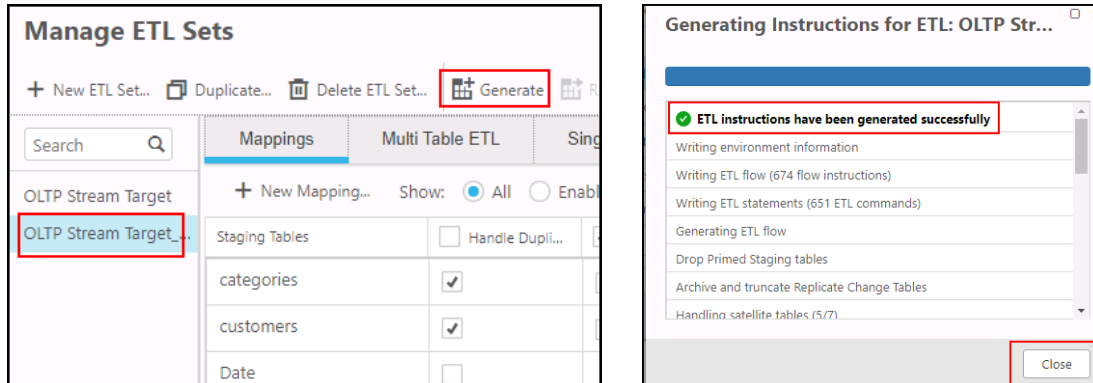
9. Once finished, **click Close**:



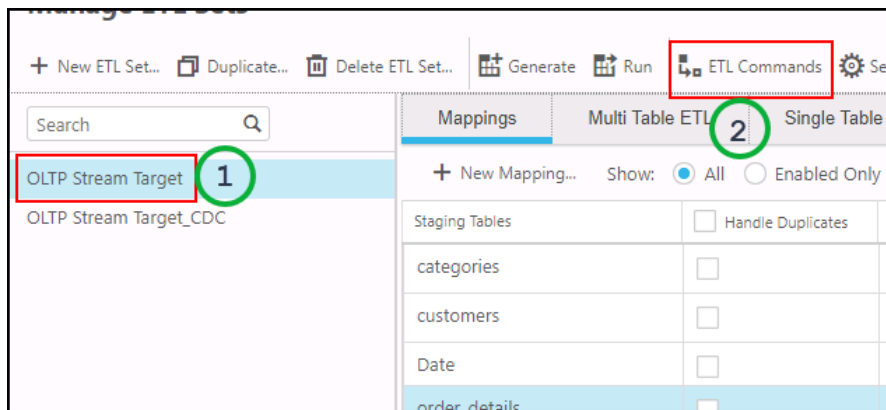
10. **Select** the CDC code-set (**the second entry**) and generate its ETL.

This is the code that will be run frequently throughout the day to keep the EDW up to date.

Once complete (about a minute), select **Close**:



11. Select the top code set again, and review the auto-generated (but not run yet) ELT SQL Server SQL code by selecting the **ETL Commands** button:



12. Double click on one of the entries to view the code and then **Close** to return to the 'Manage ETL Sets' screen.

ETL Commands - OLTP Stream Target (605 Instructions)

Item View

Process Number	Description	Entity Name	Runtime
1	Deleting open run numbers		
2	Adding open run number		
3	Flow Anchor - Handling Run numb...		
4	Creating table TTMP_MSNG_ID_32...	orders	
5	Populating table TTMP_MSNG_ID_...	orders	
6	Correcting table TDWH_orders_S01...	orders	&&Ms
7	Dropping temporary table: TTMP_...	orders	
8	Creating table TTMP_MSNG_ID_45...	products	
9	Populating table TTMP_MSNG_ID_...	products	
10	Correcting table TDWH_products_S...	products	&&Ms
11	Dropping temporary table: TTMP_...	products	
12	Creating table TTMP_MSNG_ID_6...	categories	

13. Finally, load the EDW by running the initial batch load ETL set.

(Since we have not processed any source changes, running the CDC ETL will have no effect. We'll run that later after making a source change).

Make sure the **first (batch) code set** is still highlighted and **select Run**:

This will push the ELT code down to SQL Server, which will execute it and will batch-load the EDW.

Manage ETL Sets

+ New ETL Set... Duplicate... Delete ETL Set... Generate **Run** 2

Search Q

OLTP Stream Target 1

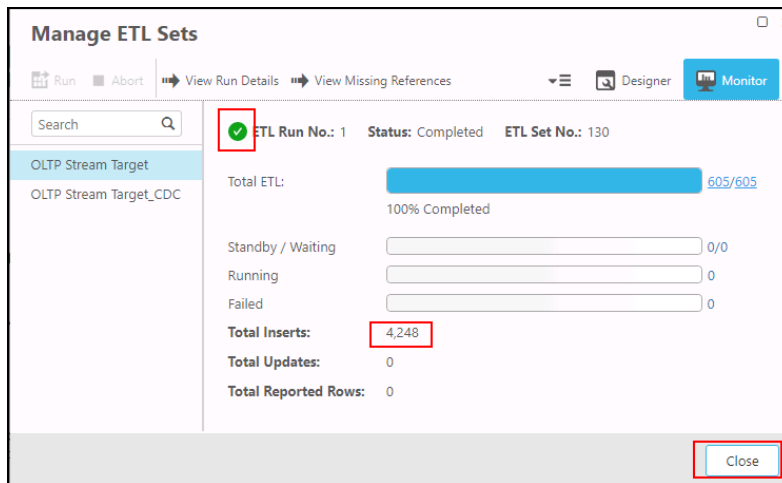
OLTP Stream Target_CDC

Mappings Multi Table ETL Single Table ETL

+ New Mapping... Show: ☒ All ☐ Enabled Only

Staging Tables	Hand...	Mappings
categories	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_categories_OLTP Str
customers	<input type="checkbox"/>	<input checked="" type="checkbox"/> Map_customers_OLTP Str

14. Wait about 15 seconds to complete and then click Close:



4,248 rows should have been inserted into our SQL Server EDW.

You can return to your SQL Editor application if you'd like to see the data that is now in the EDW.

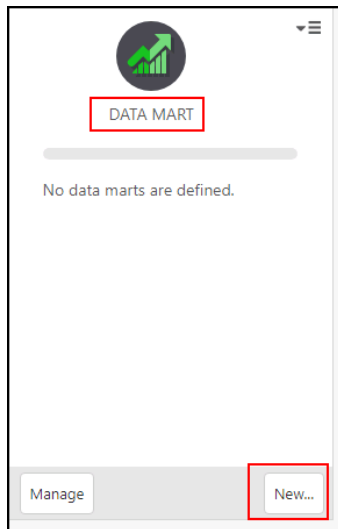
Lab 3 Outcome: Physical Data Warehouse deployed and loaded in SQL Server Data Warehouse.

Lab 4: Data Warehouse Automation – Generate Data Marts

With a populated Data Warehouse, we can now turn our attention to provisioning real-time Data Marts using Qlik's Star Schema Wizard.

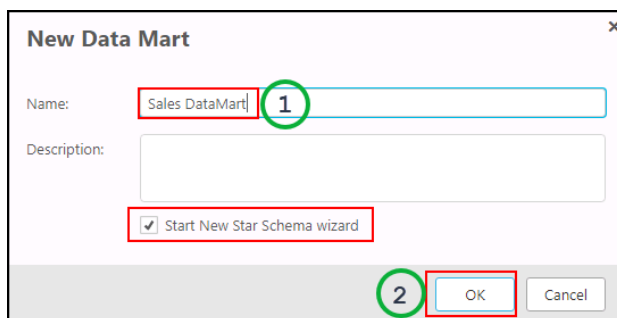
From this fourth pane of our main console, Data Mart, a Data Engineer or Analyst can provision on-demand Data Marts.

1. Begin the process by selecting the **New...** button:



2. Name this Data Mart **Sales DataMart** and **click OK**

This will start the Star Schema Wizard:

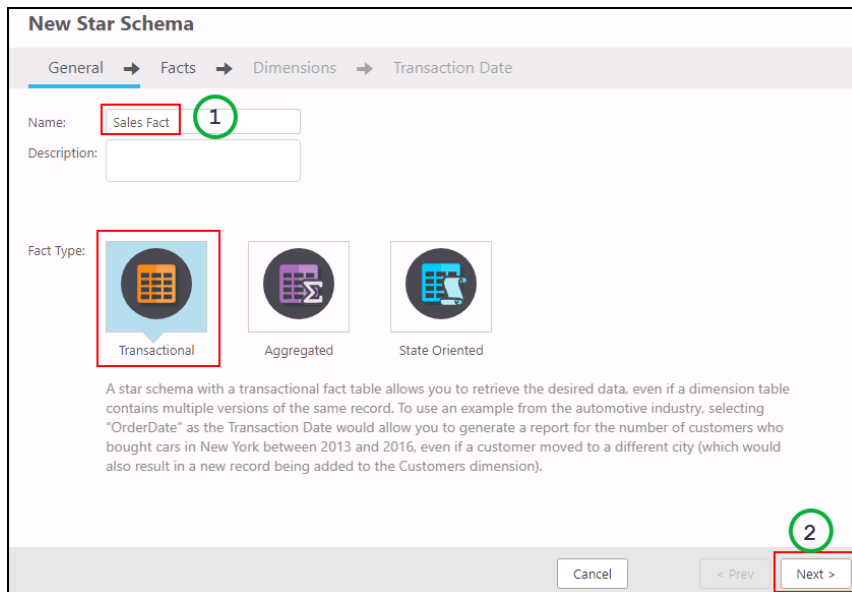


3. You'll see three variants of Star Schema that can be generated.

At the finest grain, we will choose a 'Transactional' Star Schema, however you can also create an 'Aggregated' (i.e. roll-up) schema or a 'State Oriented' (temporal) mart one which is designed for business events that complete in phases over time and include historical data.

Ensure that the **Transactional** option remains selected

Name this **Sales Fact**, and choose **Next**.



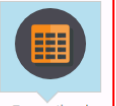
New Star Schema

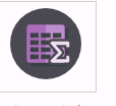
General → **Facts** → Dimensions → Transaction Date

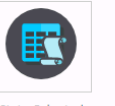
Name: 1

Description:

Fact Type:


Transactional

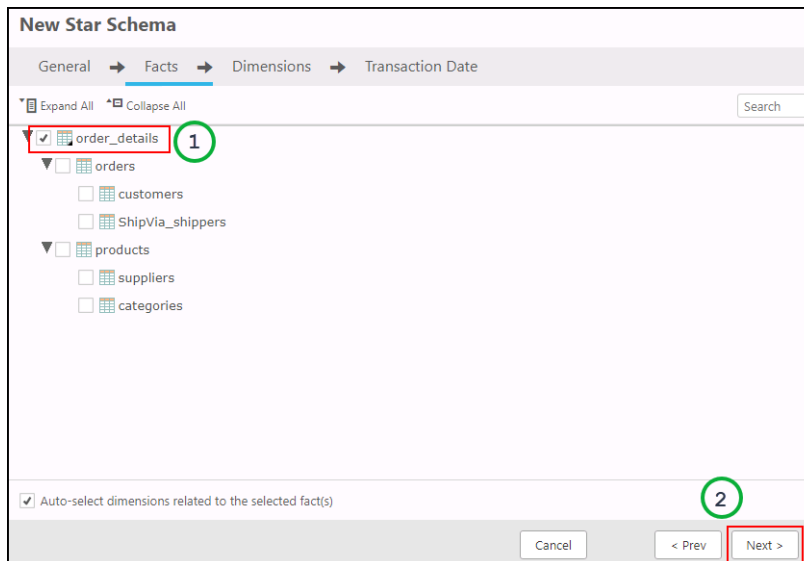

Aggregated


State Oriented

A star schema with a transactional fact table allows you to retrieve the desired data, even if a dimension table contains multiple versions of the same record. To use an example from the automotive industry, selecting "OrderDate" as the Transaction Date would allow you to generate a report for the number of customers who bought cars in New York between 2013 and 2016, even if a customer moved to a different city (which would also result in a new record being added to the Customers dimension).

2

- On the next screen, check the box next to **order_details** as your Fact table and click **Next**:



New Star Schema

General → **Facts** → Dimensions → Transaction Date

Expand All Collapse All Search

☒ **order_details** 1

▼ ☐ orders

☐ customers

☐ ShipVia_shippers

▼ ☐ products

☐ suppliers

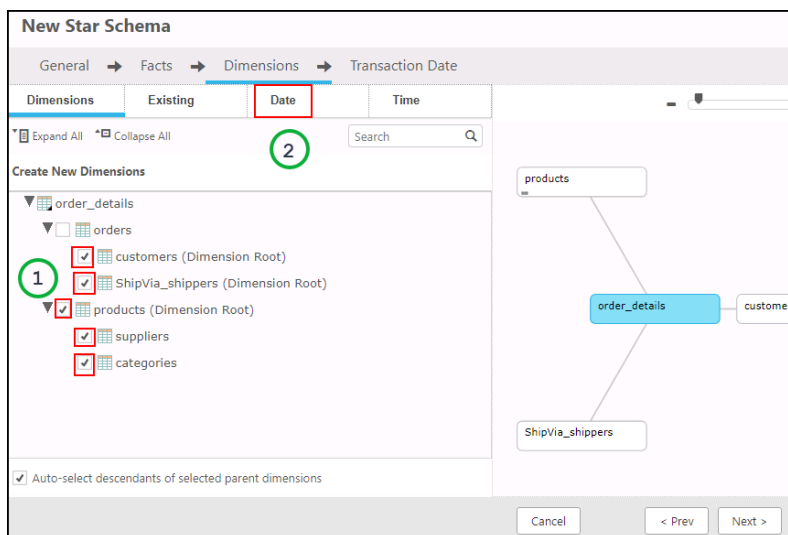
☐ categories

☒ Auto-select dimensions related to the selected fact(s) 2

Cancel < Prev **Next >**

- In the resulting screen make sure that all options *except* the **orders** check boxes are selected.

Then select the **Date** tab to choose what date attributes to include in the Fact table:



New Star Schema

General → Facts → **Dimensions** → Transaction Date

Dimensions Existing **Date** Time

Expand All Collapse All Search

Create New Dimensions

1 ☐ **orders**

☒ customers (Dimension Root)

☒ ShipVia_shippers (Dimension Root)

☒ products (Dimension Root)

☒ suppliers

☒ categories

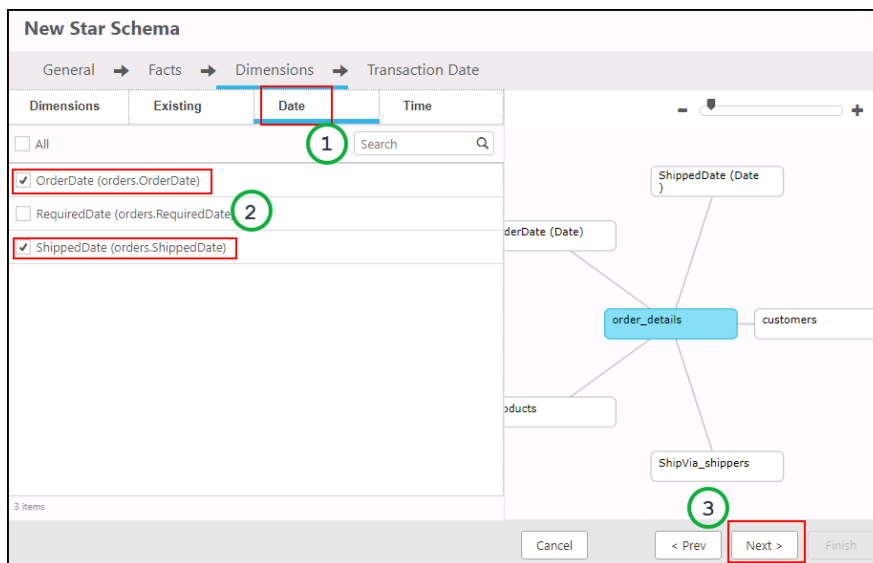
☒ Auto-select descendants of selected parent dimensions

Cancel < Prev **Next >**

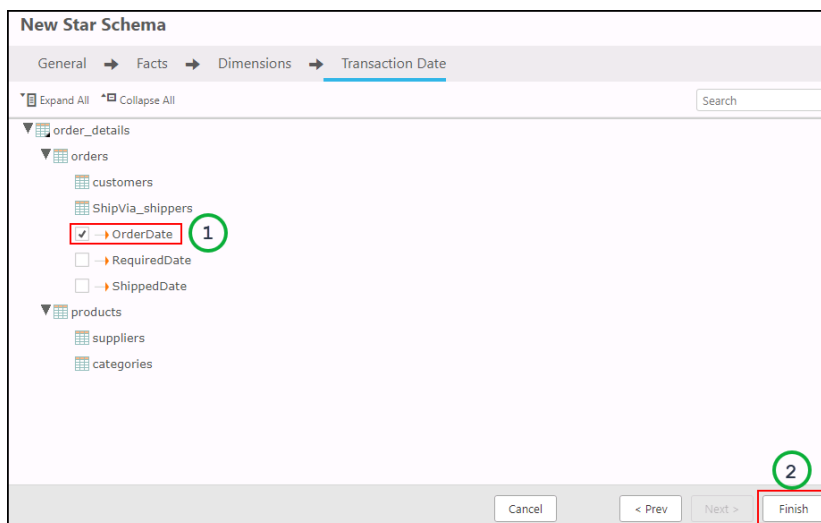
Diagram: products — order_details — customers

Diagram: ShipVia_shippers — order_details

6. In the resulting screen choose **OrderDate** and **ShippedDate** and click **Next**:

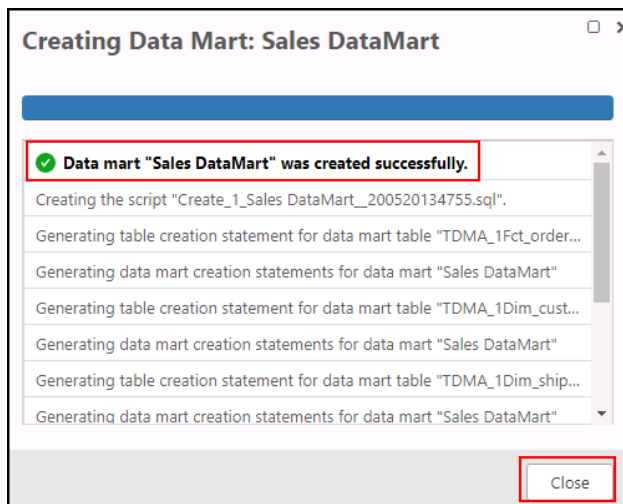
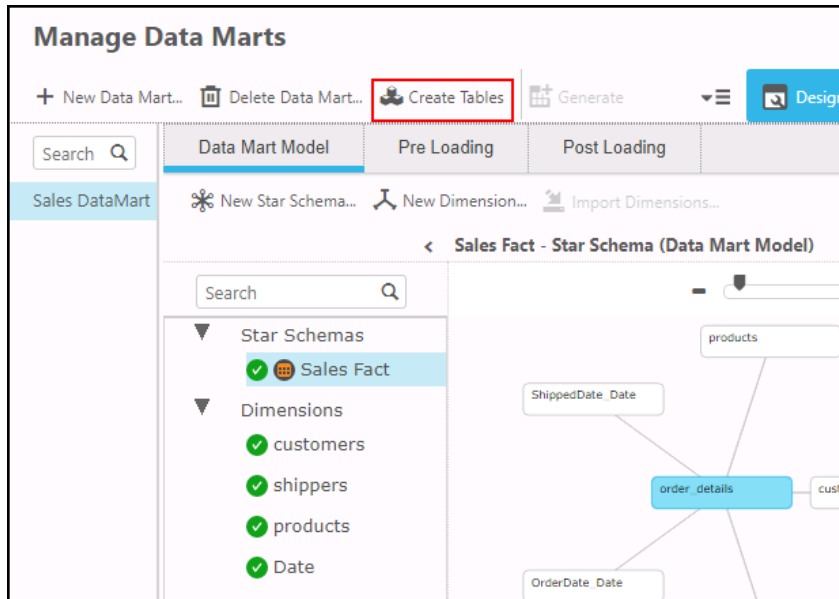


7. Finally, choose **OrderDate** as the transaction date for your Star Schema and then click **Finish**:

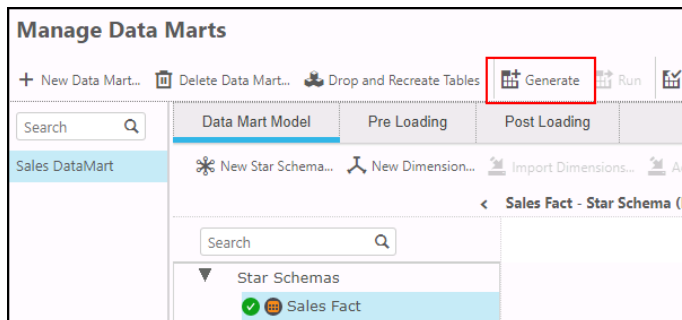


8. You can see here where your Data Engineer or Analyst can customize their Star Schema if necessary or desired. Just as with the Data Warehouse, this console could be used to add measures and LoB-specific transformations, modify Type1 vs. Type2, add or remove attributes and so on.

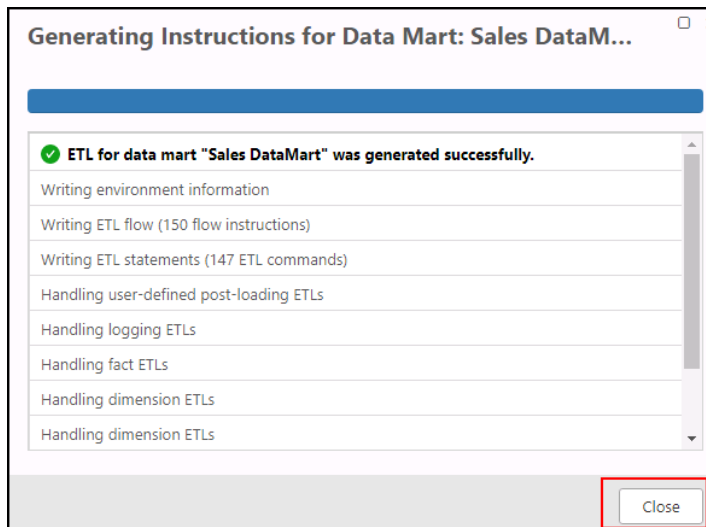
For now, select the **Create Tables** button to create the DataMart and then click **Close** from the Creating Mart list box when it creates successfully:



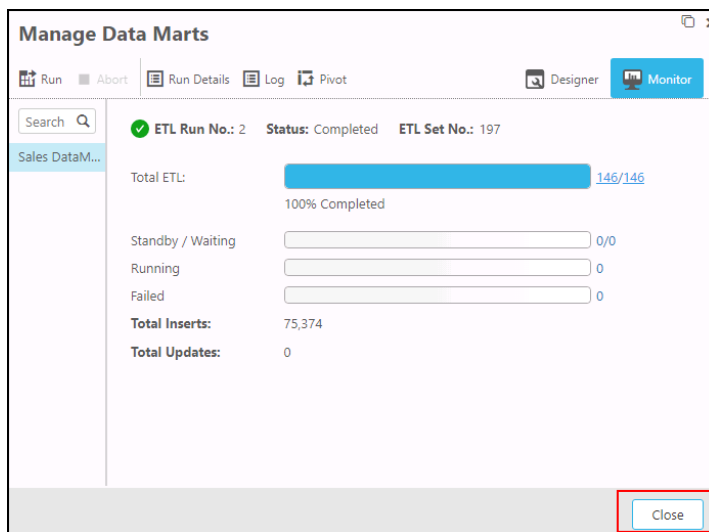
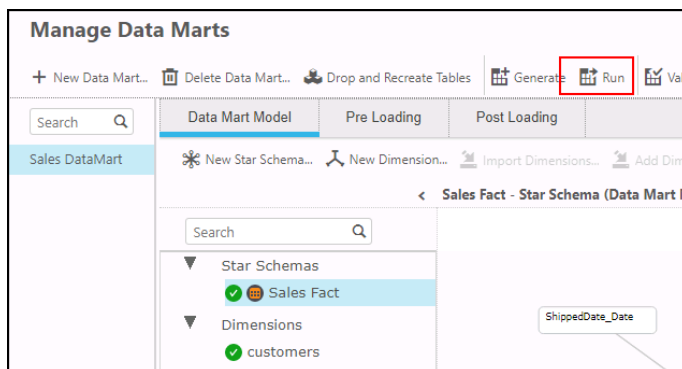
9. Next, click the **Generate** button to generate the ETL needed to eventually hydrate the Data Mart from the Data Warehouse we built:



10. Once complete click **Close**:



11. Then click **Run** to execute the code to populate the new Data Mart:



75,374 rows should be inserted into our new Data Mart.

Click **Close**.

Lab 4 Outcome – Star schema created and loaded with Sales Data.

Lab 5: Data Warehouse Automation – Real-time Data Marts

Thus far we've only dealt with design-time aspects of our real-time data pipeline. We'll now focus on Qlik's orchestration capabilities to operationalize what you've just designed.

The Data Warehouse and one Data Mart have just been created and loaded with batch data, but we now need to process changes as they stream into our landing area.

Since, in a previous step, you generated the CDC ETL Code Set, the logic is now present to process your OLTP change data into your modelled warehouse in small increments throughout the day.

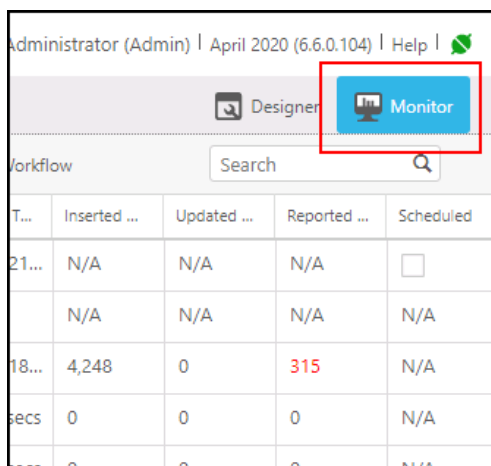
The streaming task you created earlier is still running, constantly monitoring our source OLTP database's transaction logs, and streaming new transactions to our EDW landing area.

To keep things clear and simple you will execute a simple update on the source database and watch the change, within a second or two, land in the SQL Server EDW, ready to be processed with the ETL code.

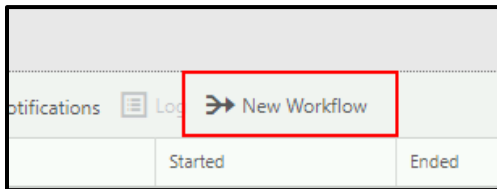
You'll configure the Data Warehouse Automation solution to consume all changes landed in the last one minute (an unrealistic cycle time but appropriate for this workshop), run the CDC ETL code to conform that data to the model, update the EDW and Data Marts, and then stop and repeat after another minute has passed.

To configure that workflow, we'll open the Monitor console:

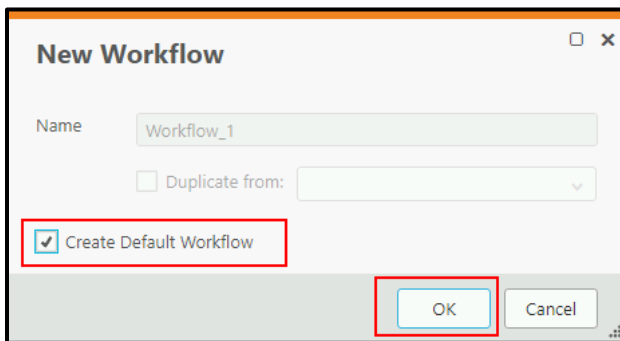
1. In the top right portion of your Design console, **select the Monitor** tab:



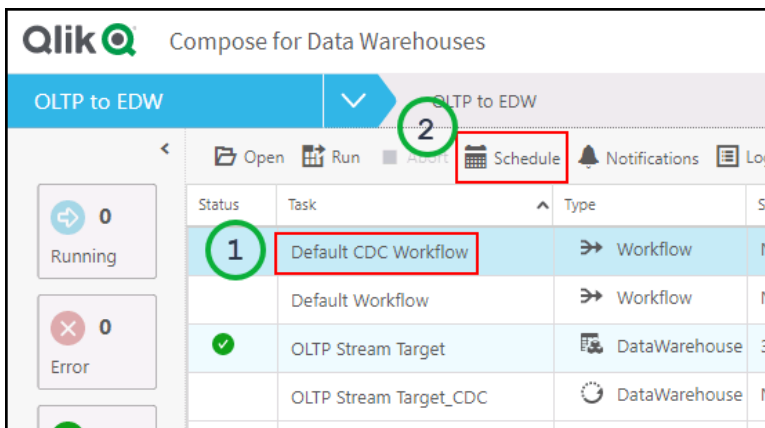
2. Create the “default” workflows for our Compose Project by clicking “New Workflow”



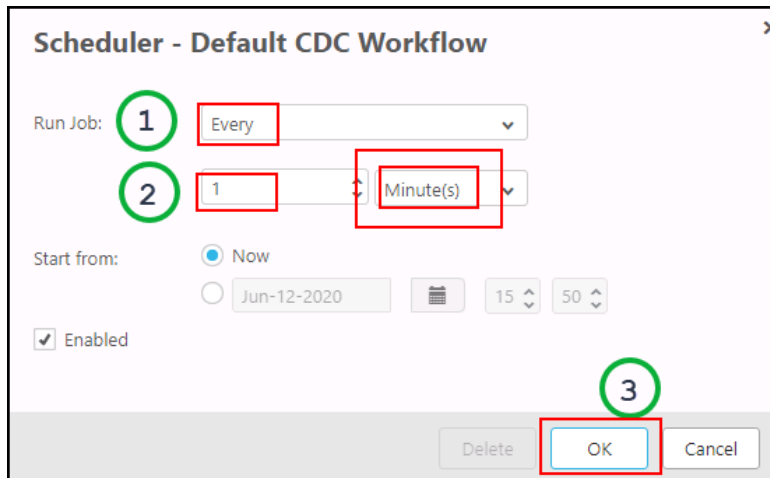
3. Check the “Create Default Workflow” checkbox and then “OK”



4. **Select the Default CDC Workflow** and then the **Schedule** button
(note: we’ve already run the initial Full Load manually so there is no need to schedule the default full load workflow here again):



On the resulting pop-up, select **Every**, and then enter **1** and **Minute(s)** from the **Run Job** drop downs. Click **OK**.

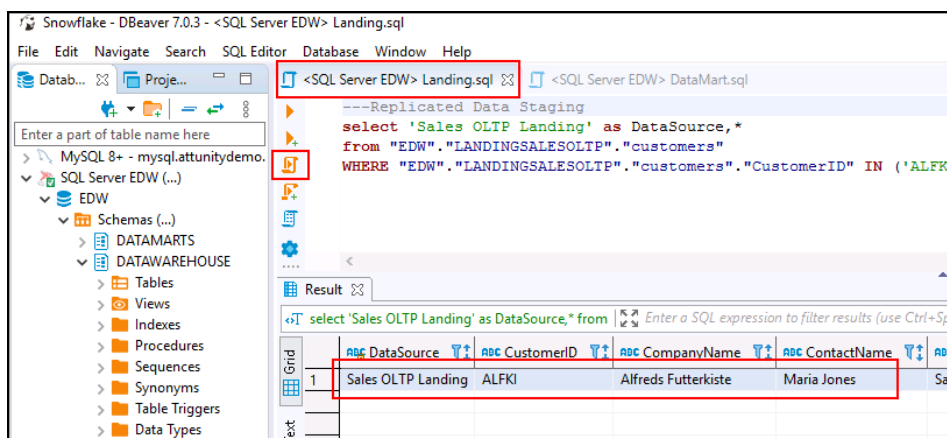


- To execute a change in the OLTP database we'll use a very simple Web App to update a single customer record.

At the bottom of your screen, on the Windows task bar, return to the SQL Editor so we can review the data that you landed in SQL Server when you started the stream task earlier:

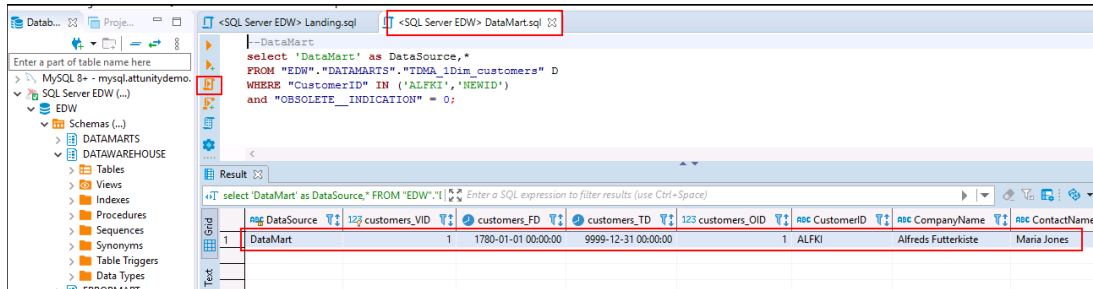


Execute the Landing.sql SQL Server query in the first pane to display a single record in the EDW Landing area where our streaming task landed our data from our OLTP system:



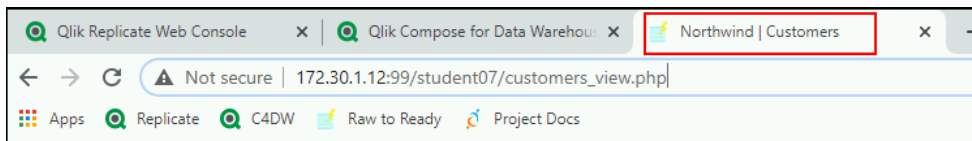
You should see a record for a customer named Maria Jones.

- Click the second query tab and **run the DataMart.sql** query to see that same record in the Data Mart:



We'll return to these queries later to see how the change we make to Maria Jones' customer record gets processed through to the Data Mart.

- Back in your browser, click on the **Northwind Customers** tab at the top to open our Web App.



The application will be open to a customer entry for Maria Jones

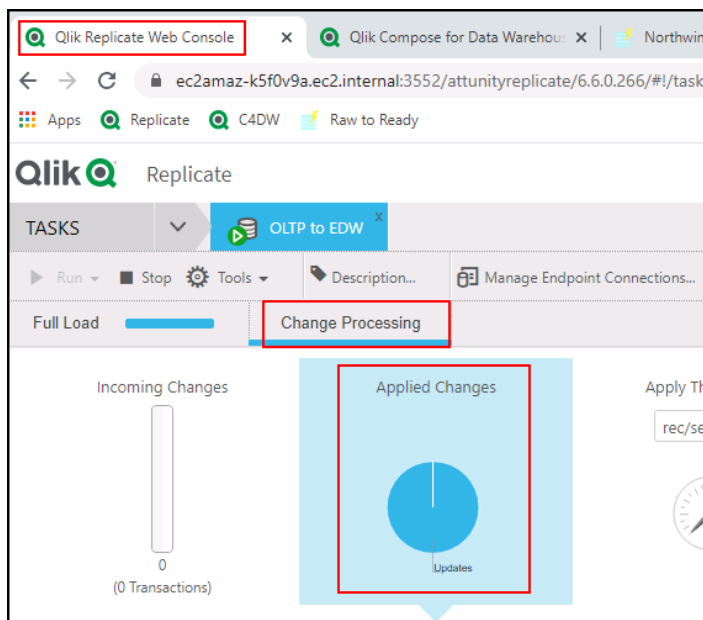
- Change Maria's country of residence from Germany to **Georgia** and click **Save Changes**:



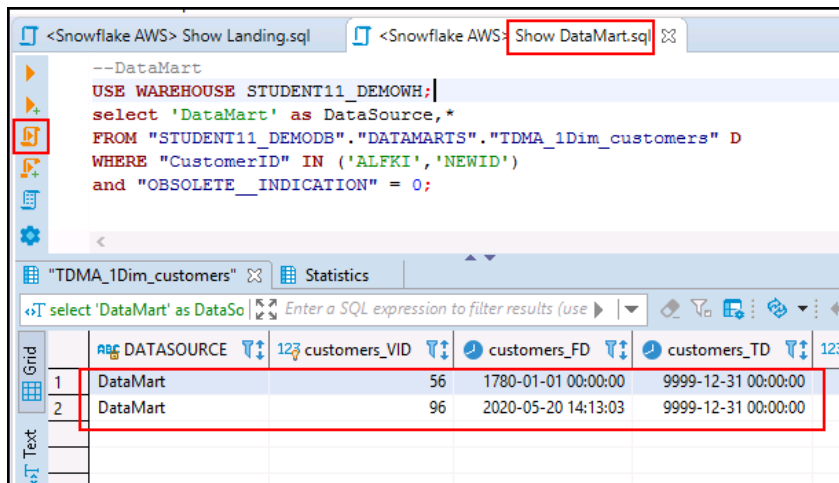
Customer ID: ALFKI
 Company Name: Alfreds Futterkiste
 Contact Name: Maria Jones
 Contact Title:
 Address:
 City:
 Region:
 Postal Code:
 Country: Georgia
 Phone: 030-0074321

Save Changes
 Back
 Print Preview
 Delete

- Next, **go back to the first Chrome tab** to the streaming console from Lab 1, navigate to the **Change Processing** tab, and observe that the transaction has been streamed:



- Return to the SQL editor and run the DataMart query again several times (up to 3 minutes) until you see two records:



The screenshot shows the Qlik SQL editor with a query window titled '<Snowflake AWS> Show DataMart.sql'. The query is as follows:

```
--DataMart
USE WAREHOUSE STUDENT11_DEMOWH;
select 'DataMart' as DataSource, *
FROM "STUDENT11_DEMODB"."DATAMARTS"."TDMA_1Dim_customers" D
WHERE "CustomerID" IN ('ALFKI', 'NEWID')
and "OBSOLETE_INDICATION" = 0;
```

Below the query editor, the results are displayed in a table. The table has columns: DATA SOURCE, customers_VID, customers_FD, and customers_TD. The results show two records for 'DataMart'.

	DATA SOURCE	customers_VID	customers_FD	customers_TD
1	DataMart	56	1780-01-01 00:00:00	9999-12-31 00:00:00
2	DataMart	96	2020-05-20 14:13:03	9999-12-31 00:00:00

The fact that we see two records is a reflection that the country of origin was set in our model (by default) as a Type2 attribute and you are seeing the before and after values for Maria Jones' county of origin. As such, these two records are time bound and would properly affect any time-based analysis on 'Sales by Country'

Note: When you set up your streaming task at the beginning of this workshop, you selected the option to maintain **Store Change** tables in our target (i.e. our SQL Server Data warehousing Landing Area). The store changes table for the Customers table captured the exact time the customer's country of residence was changed and what it was changed to... This is what enabled the tracking as a Type2 attribute in the DataMart.

Lab 5 Outcome: Perform and review real-time updates to the data warehouse and data marts created in previous labs.