WHITE PAPER

Evolving Your Data Lake with Qlik Compose™

Defining ETL Mappings for CDC Processing

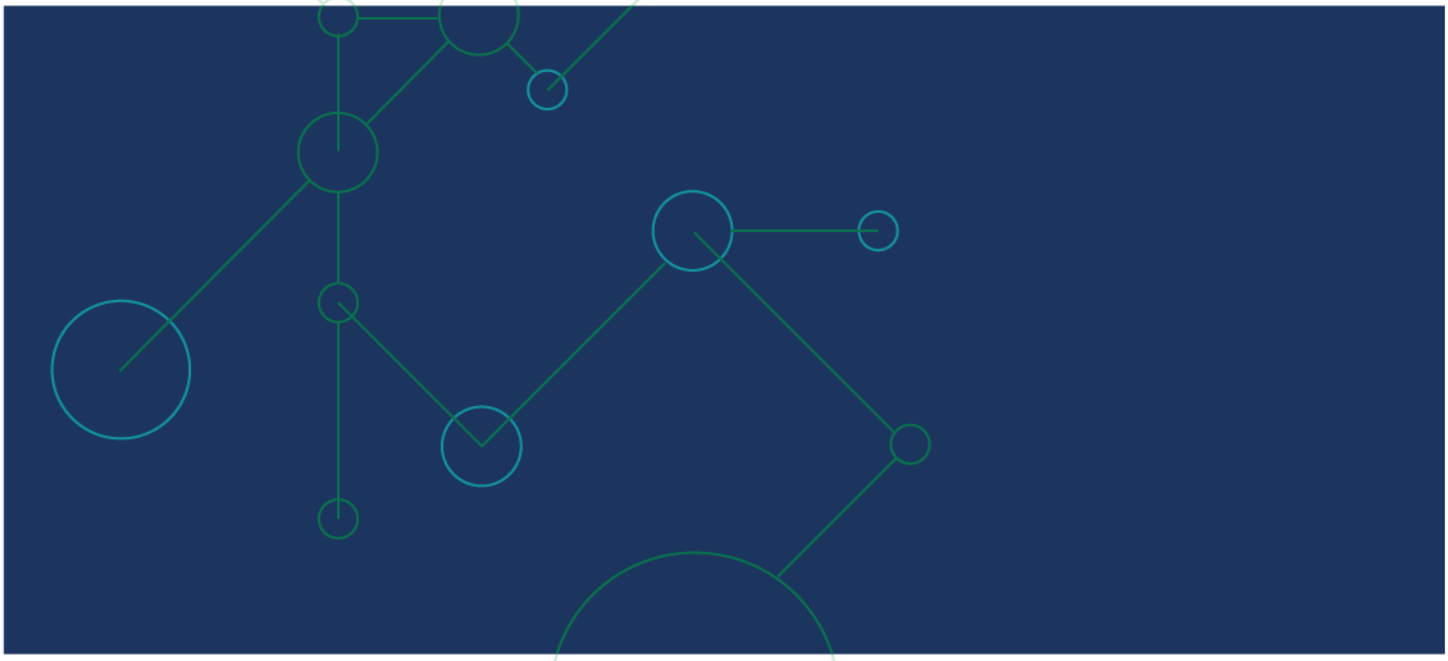


TABLE OF CONTENTS

SUMMARY

- Data Lake requirements may make it necessary to evolve your existing data lake.
- Qlik Replicate™ will automatically create new tables on your data lake target and Qlik Compose™ for Data Lakes can utilize schema evolution to add the new tables to existing data sets in your data lake
- Qlik Compose for Data Lakes can discover new replicated tables from Qlik Replicate and auto-adjust your data workflow to include the new tables without schema evolution.

INTRODUCTION

Data Lakes can store unstructured and structured data for analytic consumption. Since data is stored in files within data lake storage, it can be difficult to evolve your data lake. Over time source data systems can grow and there may be a need to capture new tables from the source data into your data lake.

If you are currently consuming data from your data lake, you will want to make sure the new source tables are included in the consumption layer of your data lake. Also, you will need to capture new committed transactions that are applied to the new source tables.

The following will describe a methodology for Qlik Compose for Data Lakes to process new source tables in an existing Qlik Replicate™ task.

Consumers of this document should have a basic understanding of Qlik Replicate and Qlik Compose for Data Lakes.

Use Case Description

Source systems are often updated with new tables to handle new use cases. Even though an enterprise may be transitioning to a modern data lake platform, there could be a case that only some of the original source tables have been migrated to the data lake at any given time. The remaining source tables will eventually need to be ingested into the data lake storage. In this specific use case, you will have a current data workflow that captures the changes for some of the source tables. The data workflow will need to be updated with new tables from the source. So, for example- if my initial data implementation project were to move 50 of 100 source tables to the data lake. The next phase of my implementation would be to add "X" more tables to my data workflow in the data lake with minimal disruption to the current workflow. This use case can be solved easily using the Qlik Data Integration (QDI) platform.

This whitepaper will demonstrate the use case using a small Northwind Sales dataset but do note that Qlik Compose for Data Lakes can scale to support your largest source data sets as needed.

Current Data Workflow

An existing data workflow is just replicating customer tables. Going forward, we also want to replicate some of the sales tables to the data lake including capturing changes from those tables.

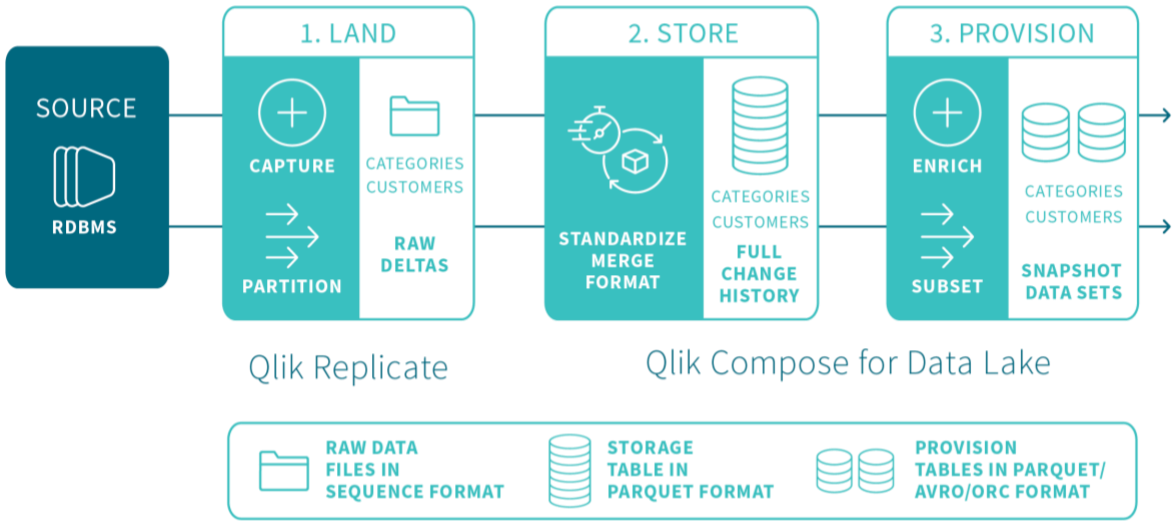


Figure 1 – Current Data Workflow with Landing Zone to Storage Zone

Create A Replication Task for the Source Tables

The Qlik Replicate task as shown in the Full Load tab below, automatically created the new categories and customers tables in the data lake and performs an initial load of the source data. In the Change Processing Tab (not selected) the Qlik Replicate task will capture changes from the source and process the changes into the data lake tables.

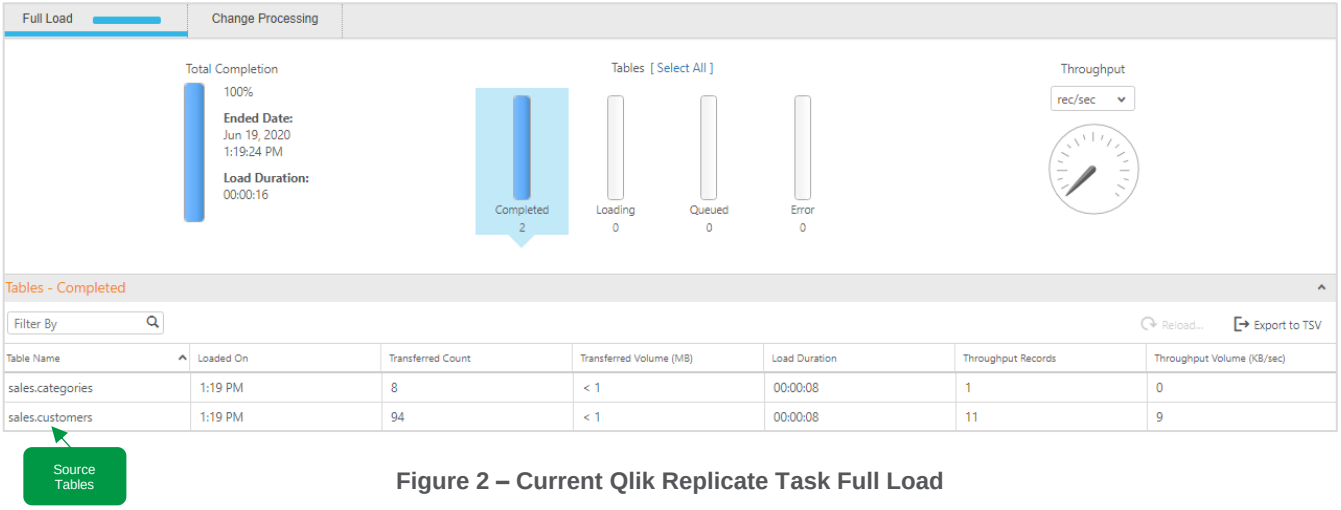


Figure 2 – Current Qlik Replicate Task Full Load

The existing tables will be processed in a Compose for Data Lake workflow, while the workflow is executing changes captured from the original Source Sales tables.

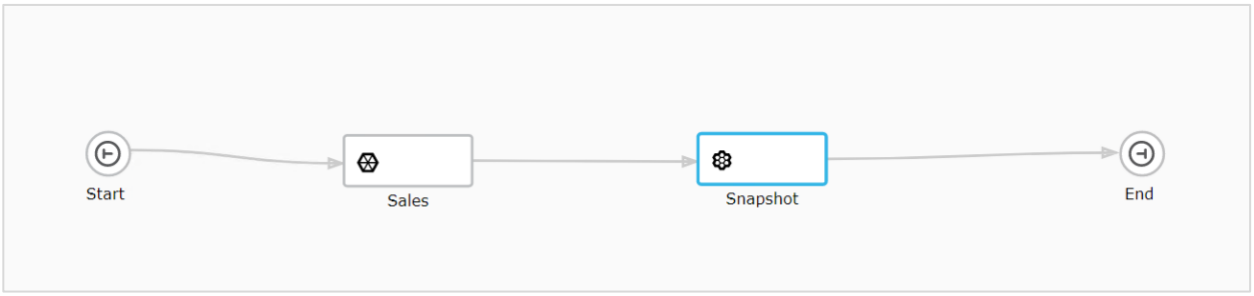


Figure 3 – Qlik Compose for Data Lakes Workflow

If more tables need to be added to the current workflow. Qlik Replicate will automatically create the new tables on your target. Using schema evolution Qlik Compose for Data Lakes will provide the ability to easily integrate the new tables into the existing full change history within your existing data storage. Qlik Compose for Data Lakes can also create a snapshot of the data in the new tables within your data lake consumption layer.

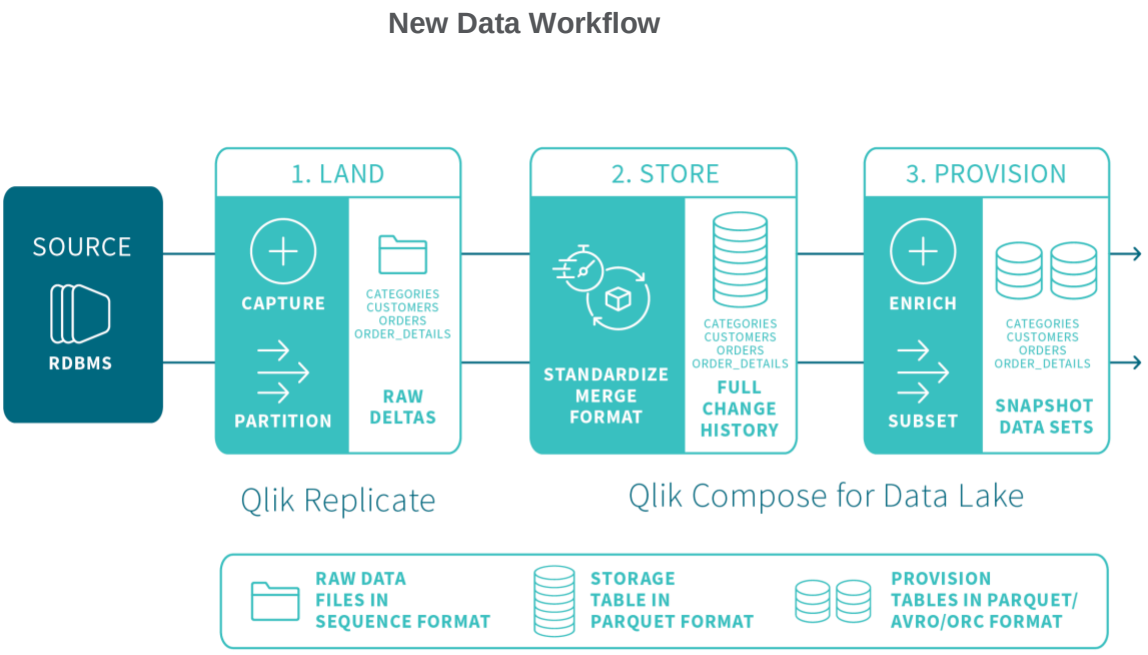


Figure 4 – Evolved Data Workflow with Landing Zone to Storage Zone

Qlik Compose for Data Lakes can also add the tables to the existing data workflow without schema evolution. The existing workflow can be either adjusted automatically or by generating a script.

Add Qlik Replicate Tables to a Compose Project

Once the tables are replicated to the data lake Landing layer. Qlik Compose for Data Lakes uses a project to define and manage the processing of these tables into the Storage and Provisioning layers of the data lake. Qlik Compose for Data Lakes offers three types of projects, each corresponding to the data lake processing engine supported (Apache Spark, Hive and Databricks) currently.

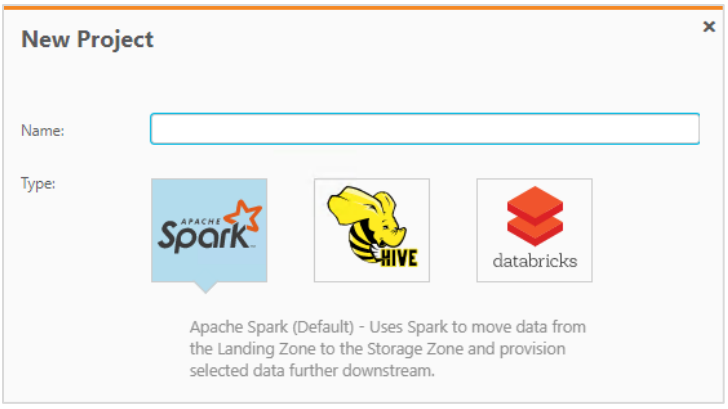


Figure 5 – New project menu in Qlik Compose™ for Data Lakes

In the Qlik Compose for Data Lakes project you can discover the data ingested from Qlik Replicate in the Landing layer and create a storage task that will load the data into the Data Lake Storage Layer. To load the data into the Data Lake Storage layer, the Landing and Storage Connections need to be configured.

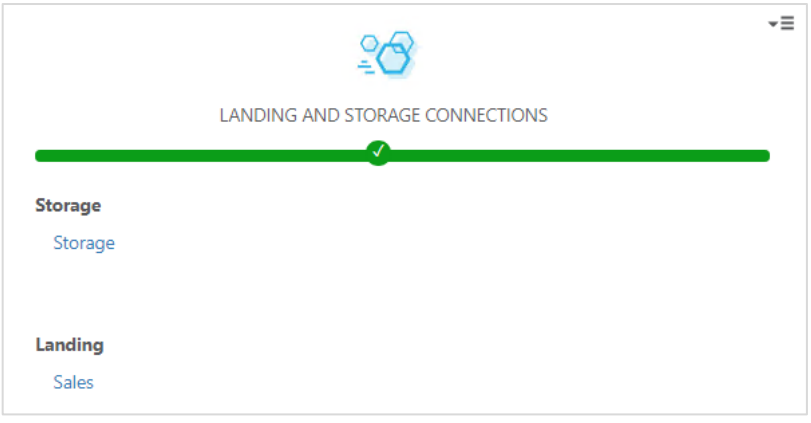


Figure 6 – Landing and Storage Connection menu in Qlik for Compose Data Lakes

Once connections are established, you can discover the tables that have been ingested from the replicate- task. These tables are used to populate the data storage tasks in the Qlik Compose for Data Lakes project.

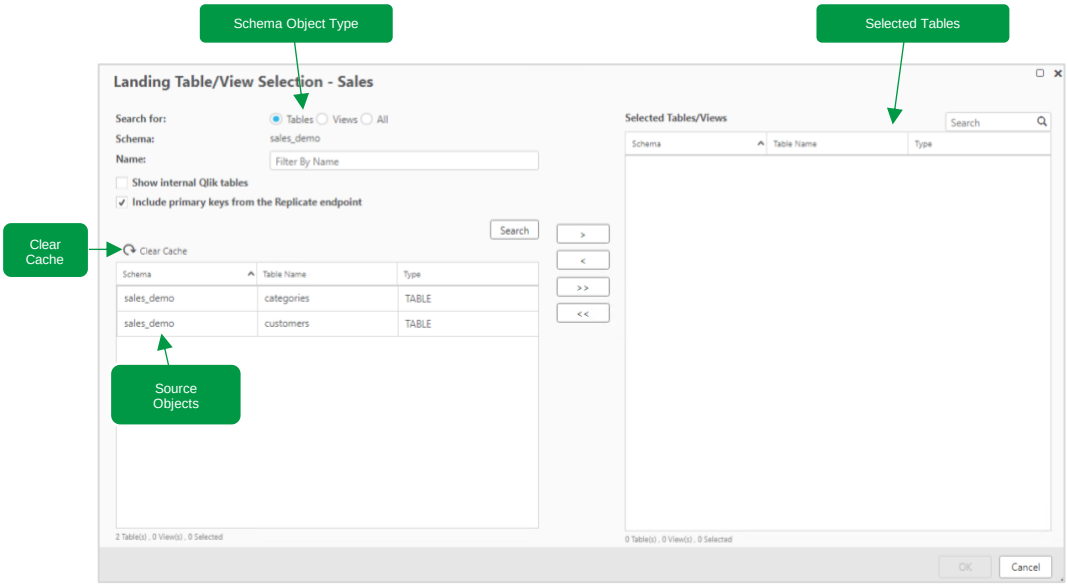


Figure 7 – Landing Table/View Discover Menu in Qlik Compose for Data Lakes

How does Compose discover metadata from a Qlik Replicate task?

Compose does a search against the Hive metastore to retrieve the metadata for the tables that have been ingested using Qlik Replicate.

With every Replicate task, a Hive database is created in the Data Lake Storage layer of the target endpoint.

Once the tables have been selected and metadata changes are completed, Qlik Compose for Data Lakes generates the mappings to ingest data into the Storage layer and create external Hive tables in the Storage layer database. In the generated mappings, the user can add expressions and lookups before executing the Task commands.

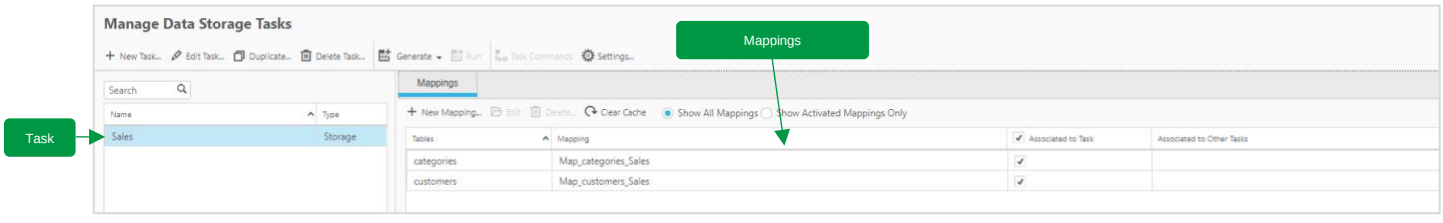


Figure 8 – Data Storage Tasks and Mappings in Compose for Data Lakes

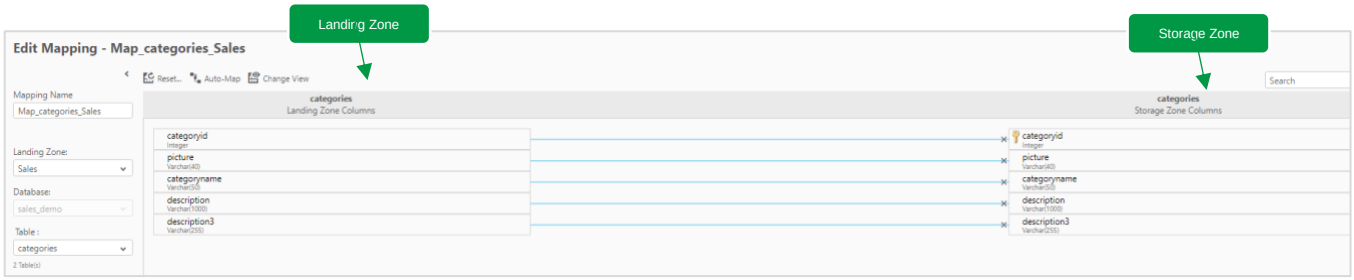
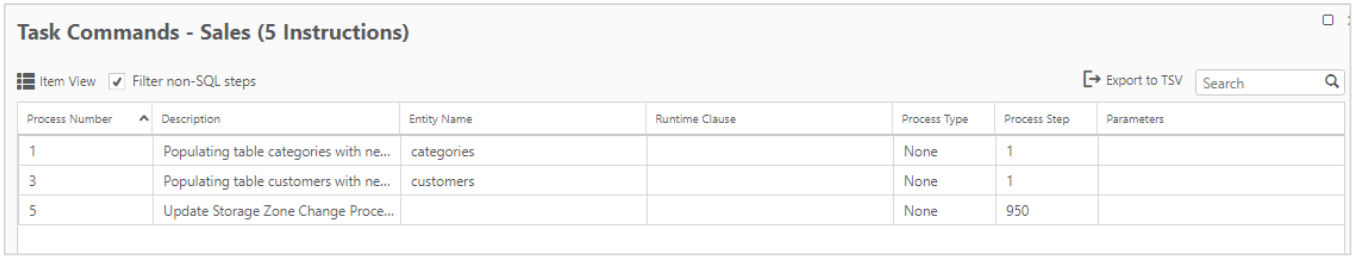


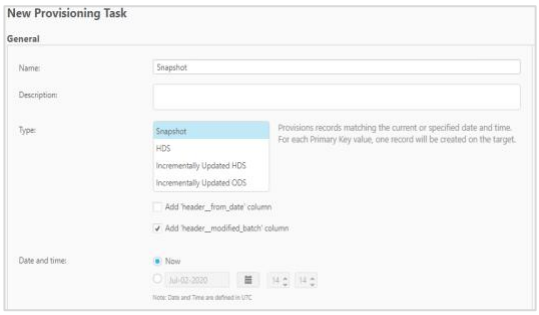
Figure 9 – Mapping Categories from Landing Zone to the Storage Zone

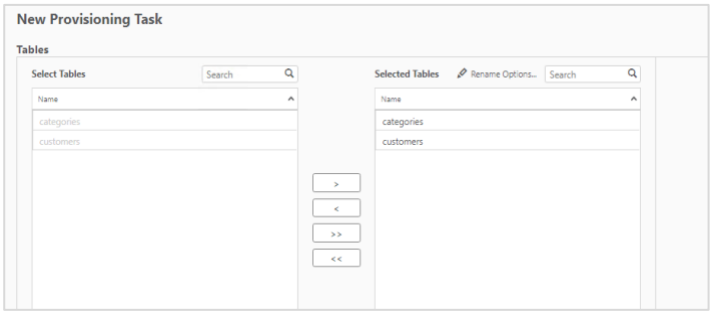


Process Number	Description	Entity Name	Runtime Clause	Process Type	Process Step	Parameters
1	Populating table categories with ne...	categories		None	1	
3	Populating table customers with ne...	customers		None	1	
5	Update Storage Zone Change Proce...			None	950	

Figure 10 – Task Commands Generated from the Mappings

After the execution of the Data Storage task, users can provision data to a target data storage location using a Compose for Data Lakes Spark project provisioning task.





Figures 11 and 12 – Provisioning Snapshot Task for the Categories and Customers tables

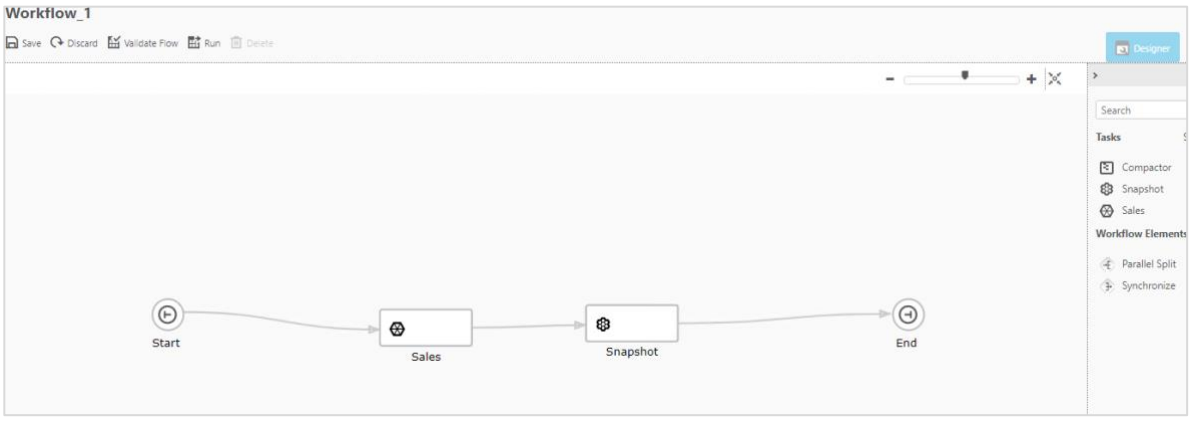


Figure 13 – Qlik Compose for Data Lakes Workflow with Storage Task and Provisioning Snapshot

Adding New Tables to an Existing Qlik Replicate Task

In Qlik Replicate define a Table Selection Pattern in the Select Tables Window. In Task Settings make sure the DDL History and Change Data Partition Tables are enabled. When the existing Qlik Replicate task is in change data capture mode, it will load any new data tables created at the source into the existing landing schema.

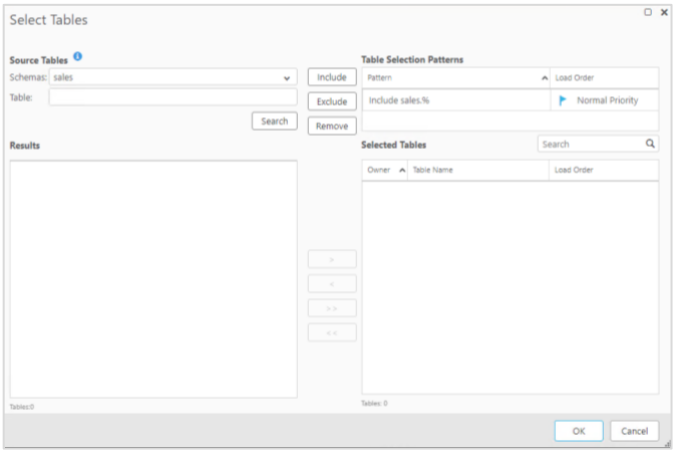


Figure 14 – Qlik Replicate Table Selection Schema Pattern defined

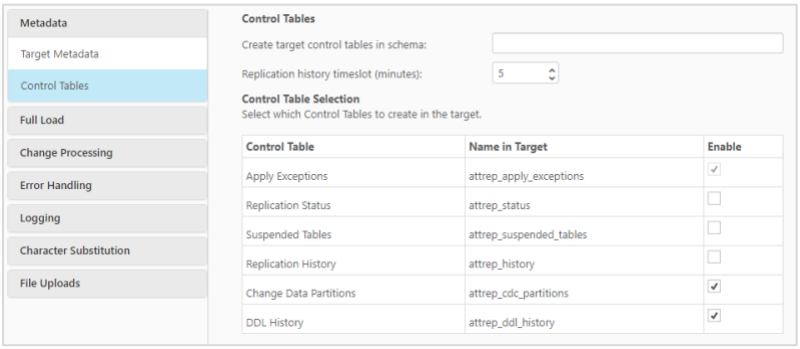


Figure 15 – Qlik Replicate Task Settings Control Tables

Without schema evolution, new tables can be loaded from the source database by adding them to the Qlik Replicate task. The new tables will be loaded into the existing landing schema. (Qlik Replicate will automatically create and load the tables in the existing landing schema. In addition to capturing committed changes once the task is in Change Data Capture mode.)

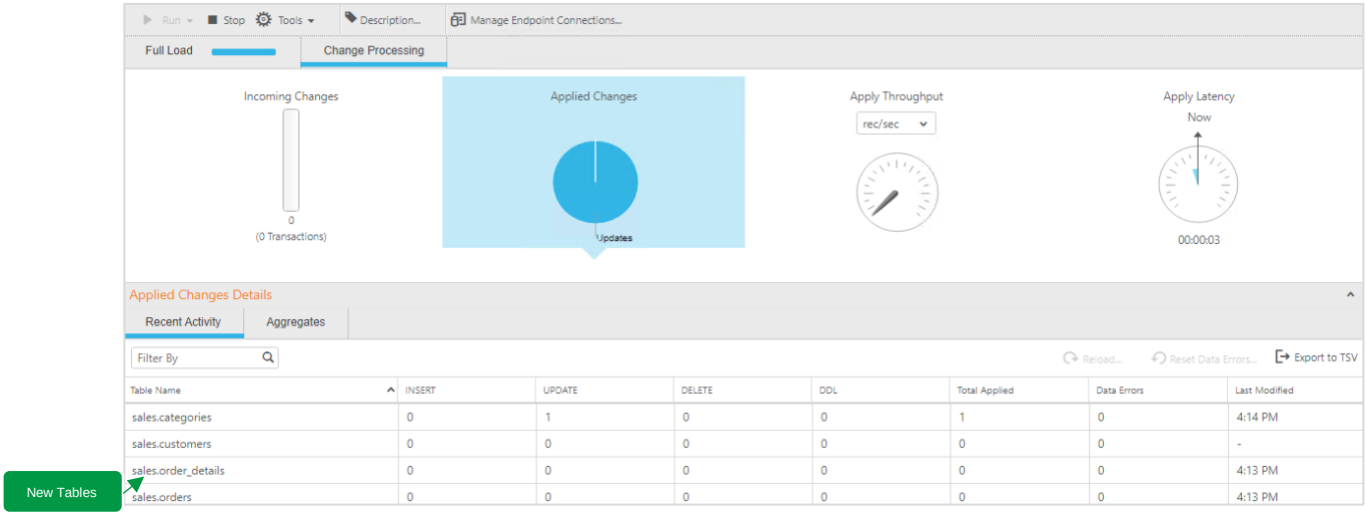


Figure 15– Qlik Replicate Task Change Processing Tab with new tables (orders and order_details)

Adding New Tables to an existing Workflow with Schema Evolution

To update a Qlik Compose for Data Lakes workflow with schema evolution, the Landing Connection needs to be associated with Replicate Task selected. Also, schema evolution needs to be enabled when creating the Compose Project Landing Connection. Automatic Schema Evolution applies only to

Change Processing Data Storage Tasks and not to the Full Load. (Association with Replicate Task allows Qlik Compose for Data Lakes to monitor and control Qlik Replicate tasks.)

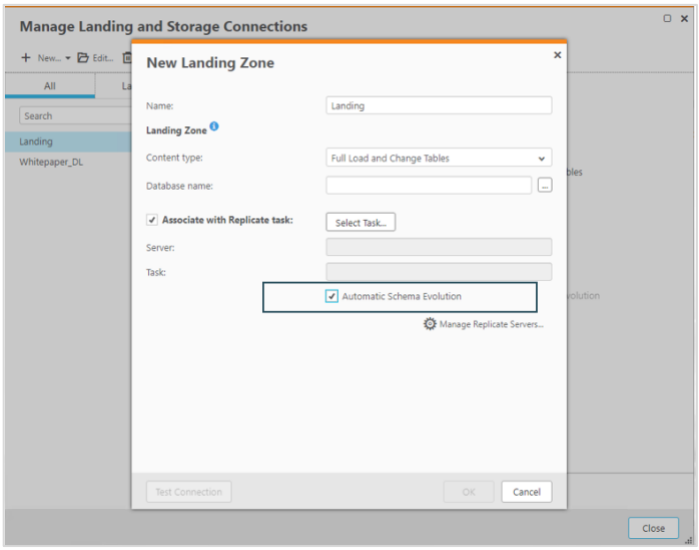


Figure 16– Qlik Compose™ for Data Lakes Landing Zone Connection with Schema Evolution

When you execute the data storage task in your Qlik Compose for Data Lakes Workflow, Compose for Data Lakes will query the DDL History table to identify the new tables before processing the captured data changes for the existing tables.

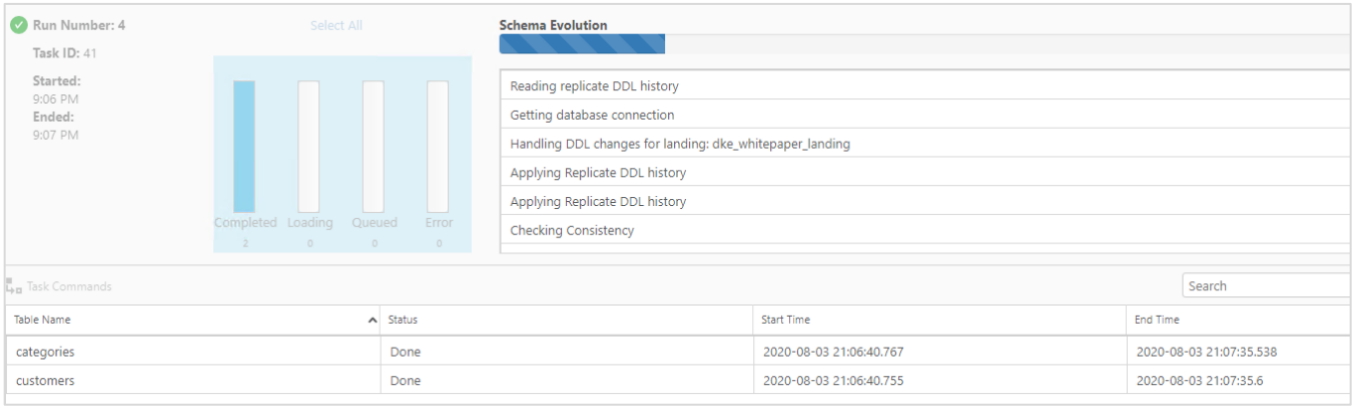


Figure 17– Qlik Compose for Data Lakes execution of Data Storage Task Schema evolution

The tables will be added to the Storage Data Layer database. You will need to update the provisioning Snapshot task with the new table before executing the workflow.

Adding New Tables to an existing Workflow without Schema Evolution

When setting up the landing connection make sure schema evolution is not enabled. Now in the existing Compose project select Discover from the Storage Zone panel. Click on the Clear Cache and the new tables should appear.

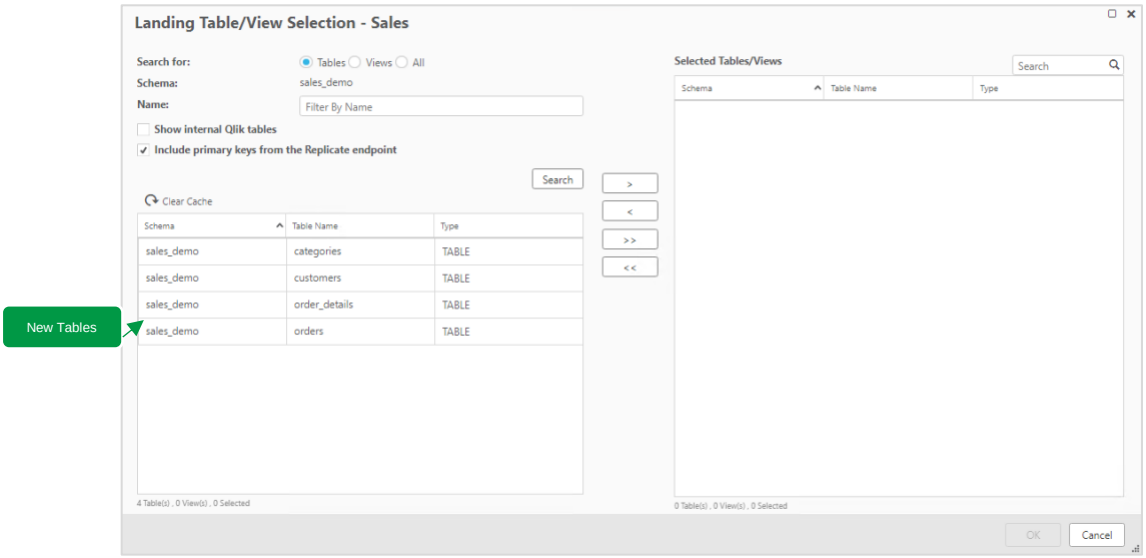


Figure 18– Qlik Compose for Data Lakes Landing Table Selection

Select the new tables and the metadata will be generated for the new tables. You can manage the logical metadata in the Storage Zone. From the Storage Zone menu select Validate External Tables. (for Hive projects, select Validate from Storage Zone menu.)

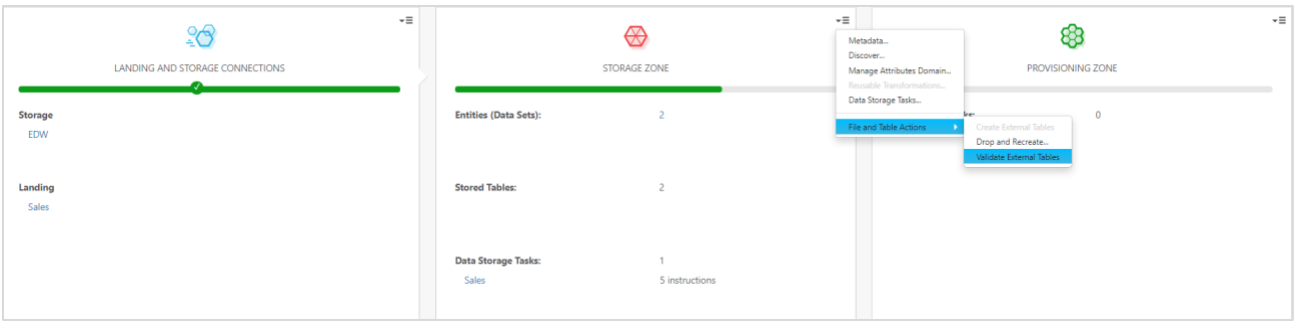


Figure 19– Qlik Compose for Data Lakes Data Storage Task Validate External Tables Menu

Compose will return a message that the Metadata is different. Click on More details.

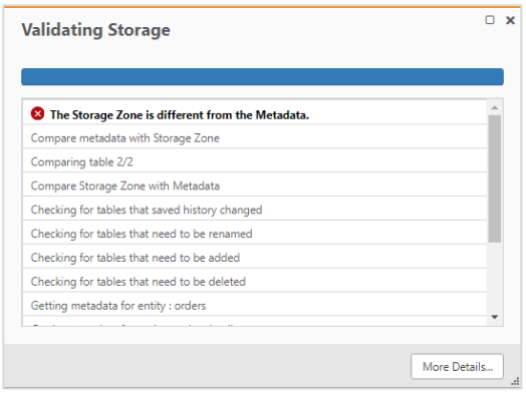


Figure 20– Qlik Compose for Data Lakes Storage Zone Validation Menu with More Details

Compose will give the option to adjust the storage task automatically or to generate a change script to be executed manually.

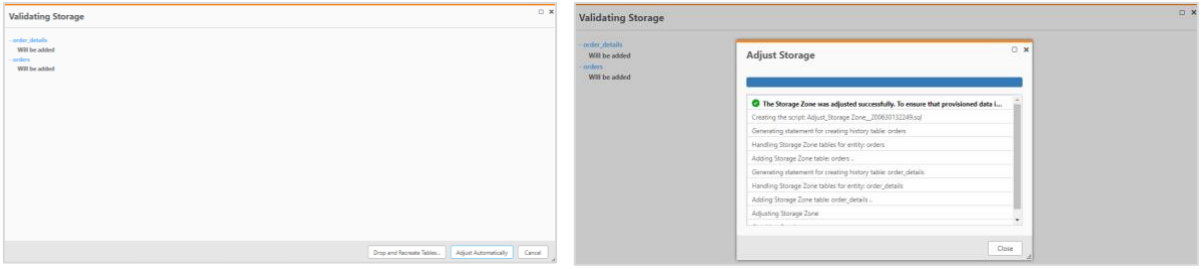


Figure 21– Qlik Compose for Data Lakes Adjust Storage for Metadata Change

When adjust automatically is selected, Qlik Compose for Data Lakes will automatically generate the mappings for the new tables.

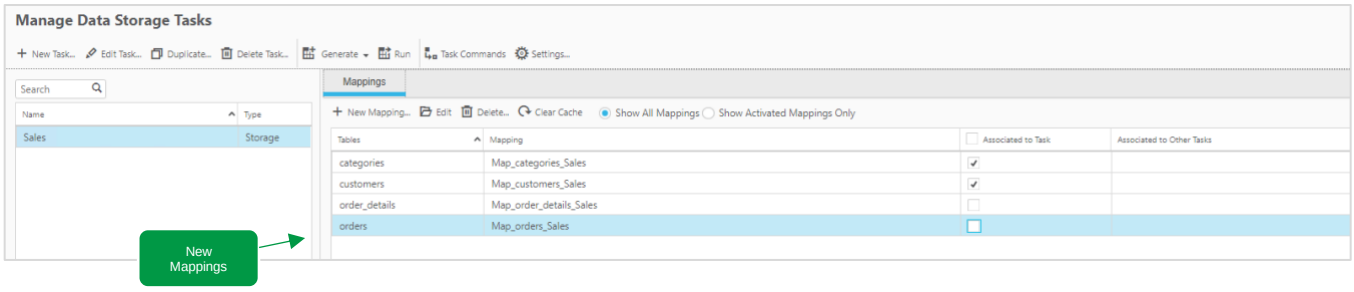


Figure 22– Qlik Compose for Data Lakes Data Storage Task New Mappings

If needed, expressions and lookups can be added to the new mappings. Just click on the mapping to edit the mapping. After you edit the mapping, associate the mappings to the Storage Task by adding the check marks to the task.

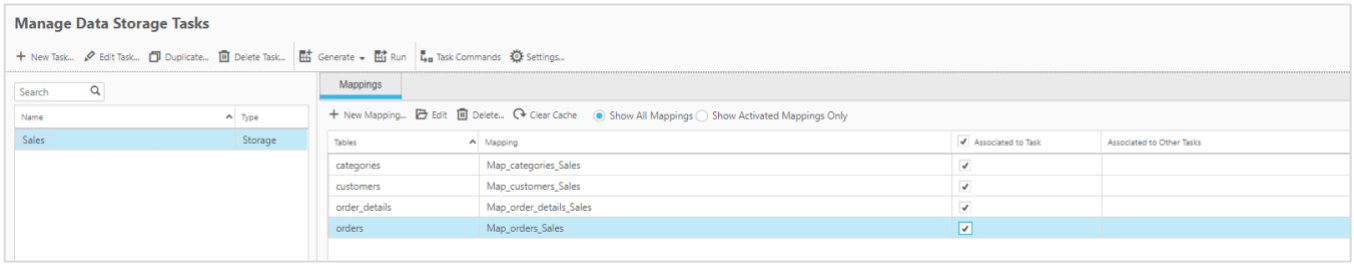


Figure 23– Qlik Compose for Data Lakes Landing Data Storage Task Association for New Mappings

Once the mappings are associated to the task. Click on Generate to recreate the Task Commands that will be executed with the updated code.

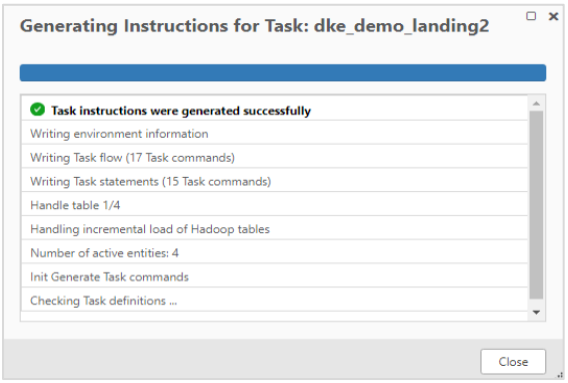


Figure 24– Qlik Compose for Data Lakes Generate ETL Instructions Task Menu

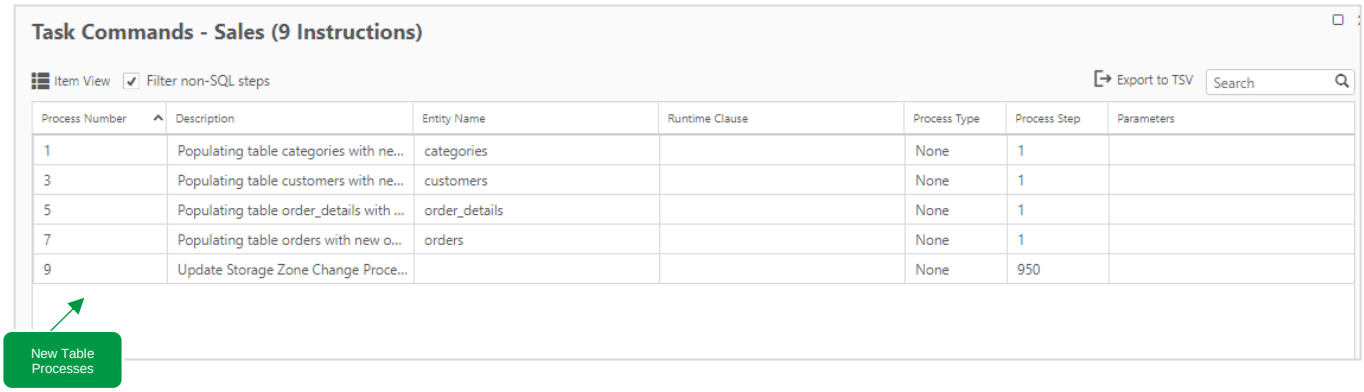


Figure 25– Qlik Compose for Data Lakes ETL Task Commands

After the task commands are generated, the newly generated code for the storage task will be applied in the next execution of the workflow. You can also edit the provisioning tasks in your workflow to add the new tables if those are needed in the provisioned data sets.

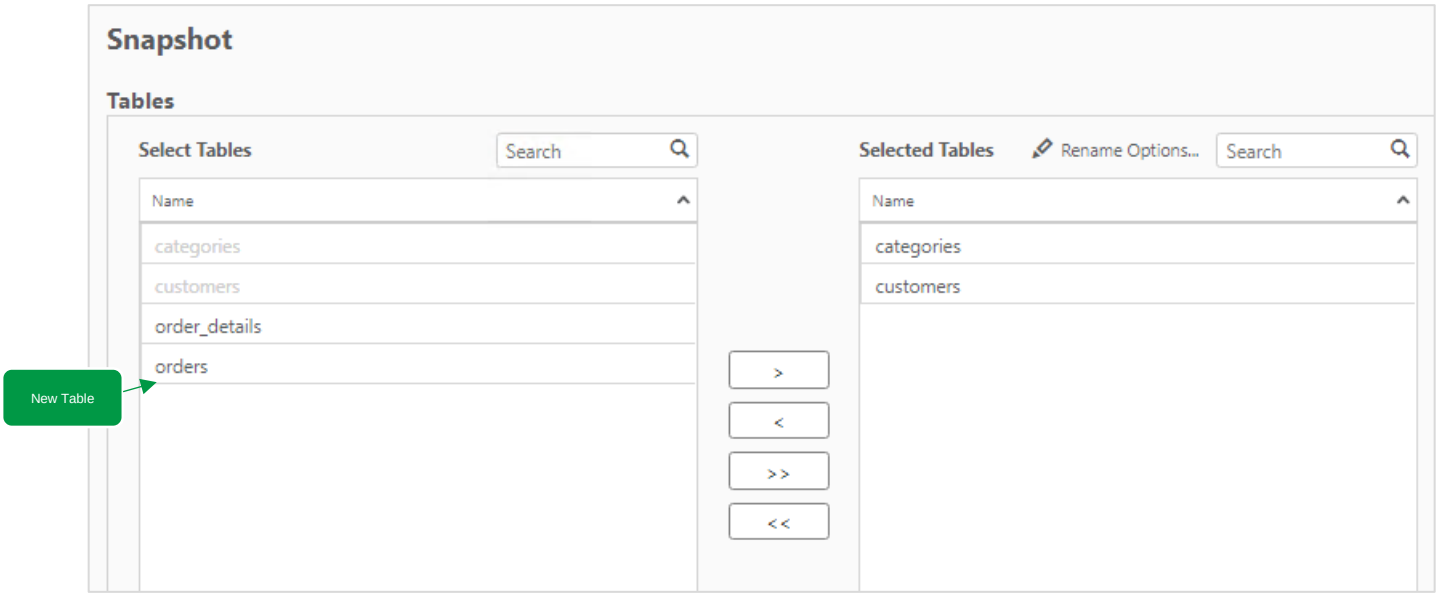


Figure 26– Qlik Compose for Data Lakes Provision Task Adding New Tables

Conclusion

Qlik Compose for Data Lakes provides capabilities to add new tables from a source to existing workflows through schema evolution and by auto-adjusting the tasks to create code. Thus, Qlik Compose for Data Lakes eliminates the need to add complex transformations or code manually if users want to evolve their data lake with new data.

Appendix: Script for Source Tables

SQL Statements used to create Northwind Sales source tables in your environment to test the solution. These statements may need to be modified for your source RDBMS (these are written for MySQL). The source tables will be added to your Qlik Replicate Task.

1. Create Source Tables

```
CREATE TABLE sales.categories(
		CategoryID int NOT NULL,
		Picture varchar(40) NULL,
		CategoryName varchar(50) NULL,
		Description varchar(1000) NULL,
		categoriescol varchar(45) NULL,
PRIMARY KEY
(
		CategoryID
)
);

CREATE TABLE sales.customers(
		CustomerID varchar(5) NOT NULL,
		CompanyName varchar(50) NULL,
		ContactName varchar(30) NULL,
		ContactTitle varchar(30) NULL,
		Address varchar(60) NULL,
		City varchar(15) NULL,
		Region varchar(15) NULL,
		PostalCode varchar(10) NULL,
		Country varchar(15) NULL,
		Phone varchar(24) NULL,
		Fax varchar(24) NULL,
PRIMARY KEY
(
		CustomerID
)
) ;
CREATE TABLE sales.order_details(
		odID int NOT NULL,
		OrderID int NULL,
		ProductID int NULL,
		UnitPrice decimal(10, 2) NULL,
		Quantity smallint NULL,
		Discount decimal(10, 2) NULL,
		Category int NULL,
PRIMARY KEY
(
		odID
)
) ;

CREATE TABLE sales.orders(
		OrderID int NOT NULL,
		CustomerID varchar(5) NULL,
		EmployeeID int NULL,
		OrderDate date NULL,
		RequiredDate date NULL,
		ShippedDate date NULL,
		ShipVia int NULL,
		Freight decimal(10, 2) NULL,
		ShipName varchar(5) NULL,
		ShipAddress varchar(5) NULL,
		ShipCity varchar(5) NULL,
		ShipRegion varchar(5) NULL,
		ShipPostalCode varchar(5) NULL,
		ShipCountry varchar(5) NULL,
PRIMARY KEY
(
		OrderID
)
) ;
```

2. Insert an Initial sample set of records

```
INSERT sales.categories (CategoryID, Picture, CategoryName, Description, categoriescol) VALUES (1, N'86051600_1344483927.jpg', N'Beverages', N'Soft drinks, coffees, teas, beers, and ales', NULL);
INSERT sales.categories (CategoryID, Picture, CategoryName, Description, categoriescol) VALUES (2, N'24242400_1344483908.jpg', N'Condiments', N'Sweet and savory sauces, relishes, spreads, and seasonings', NULL);
INSERT sales.categories (CategoryID, Picture, CategoryName, Description, categoriescol) VALUES (3, N'91786600_1344483888.jpg', N'Confections', N'Desserts, candies, and sweet breads', NULL);
INSERT sales.categories (CategoryID, Picture, CategoryName, Description, categoriescol) VALUES (4, N'16103200_1344483866.jpg', N'Dairy Products', N'Cheeses', NULL);
INSERT sales.customers (CustomerID, CompanyName, ContactName, ContactTitle, Address, City, Region, PostalCode, Country, Phone, Fax) VALUES (N'ALFKI', N'Alfreds Futterkiste', N'Maria Jones', N'Sales Representative', N'Obere Str. 57', N'Berlin', NULL, N'12209', N'Germany', N'030-0074321', N'030-0076545' );
INSERT sales.customers (CustomerID, CompanyName, ContactName, ContactTitle, Address, City, Region, PostalCode, Country, Phone, Fax) VALUES (N'ANATR', N'Ana Trujillo Emparedados y helados', N'Ms. Ana Trujillo', N'Owner', N'Avda. de la Constitucï¿½n 2222', N'Mï¿½xico D.F.', NULL, N'05021', N'Mexico', N'(5) 555-4729', N'(5) 555-3745' );

INSERT sales.customers (CustomerID, CompanyName, ContactName, ContactTitle, Address, City, Region, PostalCode, Country, Phone, Fax) VALUES (N'ANTON', N'Antonio Moreno Taquería', N'Antonio Moreno', N'Owner', N'Mataderos  2312', N'México D.F.', NULL, N'05023', N'Mexico', N'(5) 555-3932', NULL );
INSERT sales.customers (CustomerID, CompanyName, ContactName, ContactTitle, Address, City, Region, PostalCode, Country, Phone, Fax) VALUES (N'AROUT', N'Around the Horn', N'Thomas Hardy', N'Sales Representative', N'120 Hanover Sq.', N'London', NULL, N'WA1 1DP', N'United Kingdom', N'(171) 555-7788', N'(171) 555-6750' );

INSERT sales.order_details (odID, OrderID, ProductID, UnitPrice, Quantity, Discount, Category) VALUES (1, 10248, 11, CAST(14.00 AS Decimal(10, 2)), 12, CAST(0.00 AS Decimal(10, 2)), NULL );
INSERT sales.order_details (odID, OrderID, ProductID, UnitPrice, Quantity, Discount, Category) VALUES (2, 10248, 42, CAST(9.80 AS Decimal(10, 2)), 10, CAST(0.00 AS Decimal(10, 2)), NULL );
INSERT sales.order_details (odID, OrderID, ProductID, UnitPrice, Quantity, Discount, Category) VALUES (3, 10248, 72, CAST(34.80 AS Decimal(10, 2)), 5, CAST(0.00 AS Decimal(10, 2)), NULL );
INSERT sales.order_details (odID, OrderID, ProductID, UnitPrice, Quantity, Discount, Category) VALUES (4, 10249, 14, CAST(18.60 AS Decimal(10, 2)), 9, CAST(0.00 AS Decimal(10, 2)), NULL );
INSERT sales.order_details (odID, OrderID, ProductID, UnitPrice, Quantity, Discount, Category) VALUES (5, 10249, 51, CAST(42.40 AS Decimal(10, 2)), 40, CAST(0.00 AS Decimal(10, 2)), NULL );
INSERT sales.order_details (odID, OrderID, ProductID, UnitPrice, Quantity, Discount, Category) VALUES (6, 10250, 41, CAST(7.70 AS Decimal(10, 2)), 11, CAST(0.00 AS Decimal(10, 2)), NULL );

INSERT sales.orders (OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry) VALUES (10248, N'VINET', 5, CAST(N'2012-08-04' AS Date), CAST(N'2012-09-01' AS Date), CAST(N'2012-08-16' AS Date), 3, CAST(32.38 AS Decimal(10, 2)), N'VINET', N'VINET', N'VINET', N'VINET', N'VINET', N'VINET' );
INSERT sales.orders (OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry) VALUES (10249, N'TOMSP', 6, CAST(N'2012-08-05' AS Date), CAST(N'2012-09-16' AS Date), CAST(N'2012-08-10' AS Date), 1, CAST(11.61 AS Decimal(10, 2)), N'TOMSP', N'TOMSP', N'TOMSP', N'TOMSP', N'TOMSP', N'TOMSP' );
INSERT sales.orders (OrderID, CustomerID, EmployeeID, OrderDate, RequiredDate, ShippedDate, ShipVia, Freight, ShipName, ShipAddress, ShipCity, ShipRegion, ShipPostalCode, ShipCountry) VALUES (10250, N'HANAR', 4, CAST(N'2012-08-08' AS Date), CAST(N'2012-09-05' AS Date), CAST(N'2012-08-12' AS Date), 2, CAST(65.83 AS Decimal(10, 2)), N'HANAR', N'HANAR', N'HANAR', N'HANAR', N'HANAR', N'HANAR' );
```

Qlik LEAD WITH DATA

About Qlik

Qlik's vision is a data-literate world, one where everyone can use data to improve decision-making and solve their most challenging problems. Only Qlik offers end-to-end, real-time data integration and analytics solutions that help organizations access and transform all their data into value. Qlik helps companies lead with data to see more deeply into customer behavior, reinvent business processes, discover new revenue streams, and balance risk and reward. Qlik does business in more than 100 countries and serves over 50,000 customers around the world.

qlik.com

QLIKCOMPOSE102220-RJ