WHITE PAPER

Modeling Real-time Data Warehouses

Understanding Data Warehouse Automation

LEAD WITH DATA™

TABLE OF CONTENTS

EXECUTIVE SUMMARY

- Modern data warehouses must balance complex transformation requirements with the need for low latency / near real-time data
- Understand how Qlik graphical modeling techniques express complex business concepts while enabling automated data warehouse processing
- Recognize how Qlik's data mart automation supports and simplifies complex transformation requirements

INTRODUCTION

In a modern analytics environment data warehouses must deliver more data than ever before to support right-time analytics and insights. For some enterprises this means reducing the end of day processing cycle, while for others it means delivering data in near real-time streams. However, the key to achieving this agility is to quickly harness complex data transformations that rapidly deliver the right data sets, at the right latencies, to the right people, at the right time.

Qlik Data Integration helps data architects simplify their transformation requirements to provide the flexibility to support multiple data velocities in near real-time for various data warehouse architectures.

This technical paper is a guide to understanding Qlik Data Integration warehouse modeling and transformation techniques and their impact on near real-time Change Data Capture (CDC) processing in an agile data warehouse environment.

The Modern Data Warehouse Architecture

To understand why data modeling is important to transformations, we must first understand the end-goal of a modern data warehouse. Historically, data warehouse solutions provided structured data for consumers and Business Intelligence (BI) tools to support reporting, dashboards and ad-hoc analysis across various subject areas, Key Performance Indicators (KPIs) and conformed dimensions. More recently modern data warehouses may also be used to provide data sets that support machine learning, predictive model training and other data science use cases.

Traditionally data warehouses were designed to produce highly denormalized data artifacts (dimension and fact tables) which were typically loaded from source systems or operational data stores (ODS). This practice worked well when the only business requirement was to provide end of day reporting, and projects delivery timelines were measured in months or years. In addition, the Extract, Transform and Load (ETL) tools that performed the processing didn't understand the data models and were built to work in a batch-oriented manner. As organizations looked to modernize investments in their data platforms many organizations realized these batch-oriented solutions could not support real-time data delivery and decision making.

Qlik Data Integration addresses these shortcomings to effectively manage and automate the end-to-end lifecycle of your data warehouse – from initial data models, to transformation processing, to final data marts provisioning and delivery. The complete data architecture is detailed in the diagram below. Qlik Replicate delivers data to the landing area in near real-time, while Qlik Compose for Data Warehouses picks up changed data and processes it through a staging area to a central data warehouse. The data warehouse is defined by the data model which in turn can feed 1:N star-schema data marts. This approach improves the data engineer's productivity because it automates the complex join transformations traditionally hand-coded by batch-oriented ETL tools for end of day processing.

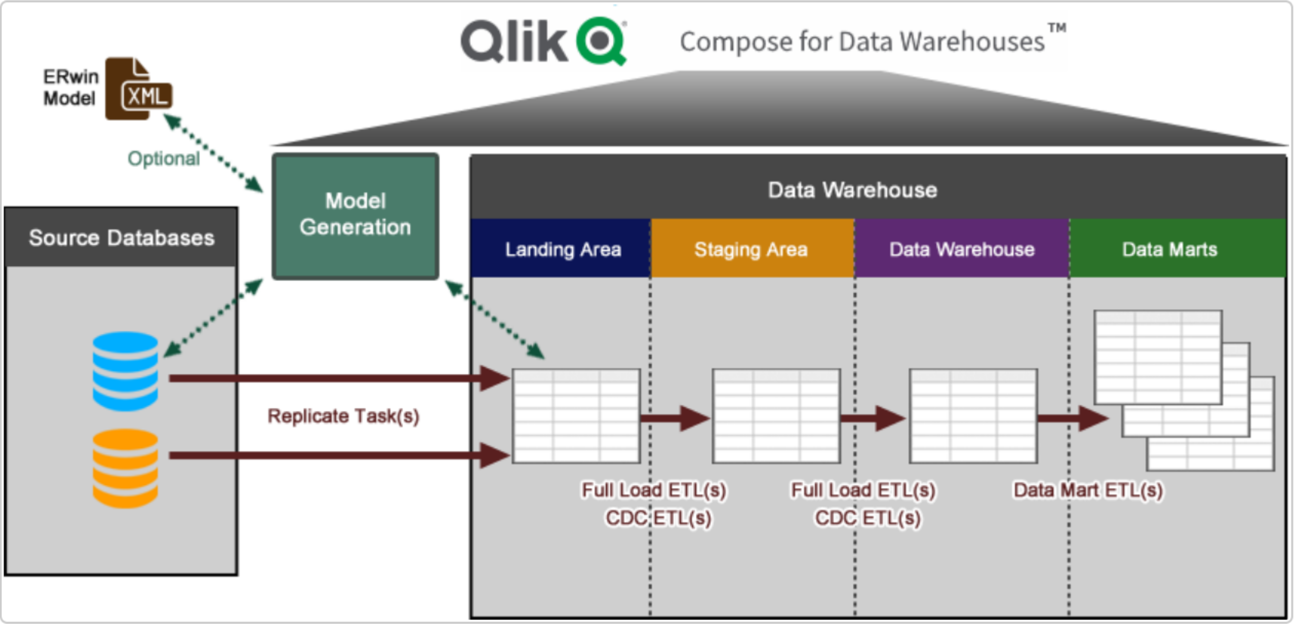


Figure 1: Qlik Data Architecture

Understanding Qlik Compose Model

Qlik Compose model generation provides the foundational layer for your data warehouse solution. It not only serves to structure the data in an efficient, business centric fashion (leveraging data vault modeling techniques such as hubs and satellites), it also drives much of the intelligence in the ETL automation and data mart creation. Therefore, it's important to understand modelling best practices and how your certain architectural decisions can impact your data warehouse projects and timelines.

Good modeling practices can:

- Reduce complexity of ETL developed by the data warehouse team
- Reduce data processing SLAs with CDC integration
- Increase the amount of data warehouse automation
- Increase the centralization data processing logic and reduce complexity of data mart design

Qlik provides four out-of-the-box methodologies for implementing the model:

1. Intelligent discovery
2. ErWin model import
3. CSV import
4. Manual-create data model with visual tools.

Note: The tips provided in this paper should be followed regardless of the chosen modeling method.

Qlik Modeling Concepts

Explaining generic database modelling concepts is out of scope for this paper, but it is important to understand that the Qlik model is defined and designed using similar modeling definitions

- **Entities**
- **Attributes**
- **Relationships**

These logical constructs are then physical implemented in the data warehouse as

- **Tables –** Entities are physically created as 1: N tables using Hubs and Satellites
- **Columns –** Attributes are physically created as columns in Hubs and Satellites
- **Relationships** – Although not physically implemented as foreign keys (FK's), but have a direct impact on the ETL automation functionality within Qlik.

Qlik's modeling philosophy are based on the data vault methodology. However, modeling in Qlik does not require in-depth knowledge of data vault specifics. Rather, we provide logical modeling features while automatically managing the underlying physical data vault. Note Qlik can also handle advanced data vault management scenarios such as multiple satellites to manage fast and slow-moving data. The Qlik model provides the structure for the data warehouse from which data marts are generated.

Qlik Data Marts

While modern data platforms such as Amazon Redshift, Azure Synapse and Snowflake continue to rollout more scalability features, data warehouses still need to structure their tables for efficient queries to support granular and summary analytics. Those tables are commonly referred to as data marts. Qlik provides a data mart wizard that helps you rapidly design the tables and generates all of the necessary automation code to fit your business requirements.

Qlik's notion of data marts are star-schema oriented materialized (physical) structures that are built from the central data warehouse model. The central model has a direct impact to the data mart structures that can be built and automated.

In addition, fact tables can also be built from any data warehouse table or tables related to each other in the data warehouse model. Furthermore, dimensions are selected from tables related to the data warehouse fact tables. Note that dimensions do not need to be modeled in a de-normalized fashion since Qlik automatically de-normalizes tables related to the dimension "root" (or grain of the dimension). Processing of this data is done in an automated, incremental fashion.

Model Normalization with CDC

As mentioned, data warehouses have historically been modeled in a denormalized manner to support BI / visualization and reporting performance. This leads to complex ETL particularly related to change data capture (CDC) based processing. The better practice is to design and manage a more normalized structure for the central warehouse model while leveraging Qlik's data mart automation to handle incremental denormalization requirements. Let's examine a common use case that requires denormalization of multiple tables and near real-time processing in detail.

Normalizing the Data Model

In the example scenario we highlight a few tables from Microsoft SQL Server's AdventureWorks database, specifically tables that represent Products. Specifically let's look at a 2-table example - ***ProductModel*** and ***Product***. In the diagram below we can see that ***ProductModel*** is a 1:M ***Product*** relationship.

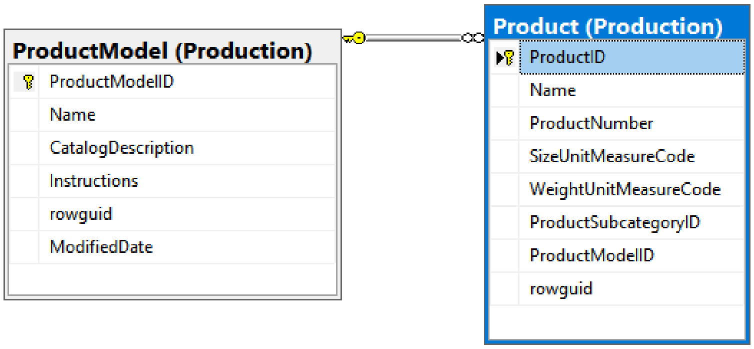


Figure 2: ProductModel and Product Table Relationship

A typical product dimension may appear as follows:

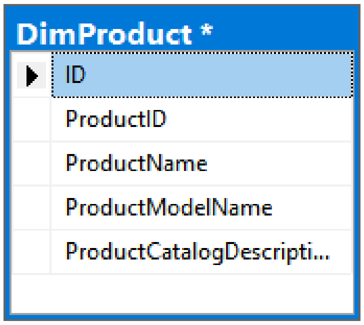


Figure 3: Product Dimension Table

A common pattern to load this structure would be:

Product <Lookup> ProductModel Attributes → Dimension Processing

This simple **bulk** processing ETL task queries all the ***Product*** data, performs a ***lookup*** to retrieve ***ProductModel*** attributes and the ***Dimension Processing*** determines new and changed records. However, processing this in near real-time from a CDC stream adds complexity because changes to either ***Product*** or ***ProductModel*** need to be considered.

The table below highlights a scenario where there were just 2 changes highlighted in green. Note that we do not show the before values, but just indicate that the record has changed. In the ***Product*** table we see a change highlighted where ProductID=1 changed. In the ***ProductModel*** table we see that the ProductModelID = 66 changed.

Product. ProductID	Product. ProductModelID
1	99
2	66
3	67
4	66

ProductModel. ProductModelID	
99	Model A
66	Model B
67	Model C

Therefore, the warehouse ETL must account for changes from both tables to process and load data correctly. The ETL must account for changes to rows ***ProductID 1***, ***ProductID 2*** and ***ProductID 4*** because there was a change in the ***ProductModelID 66***. Similarly, the ETL must also process ***ProductModelID 66*** too. Of course, there would be 1000's of changes to both tables in a real-world scenario, impacting many more ***Product*** and ***ProductModel*** records.

The ETL required to process these changes in near real-time simply can't join both tables or perform a modest lookup from the ***Product*** to ***ProductModel*** table. However, designing a normalized structure (see below) allows for more streamlined CDC processing where the ***Product*** and ***ProductModel*** changes are managed independently.

This structure accomplishes two things:

1. It enables CDC-based processing in a streamlined fashion
2. Reduces storage requirements for changes

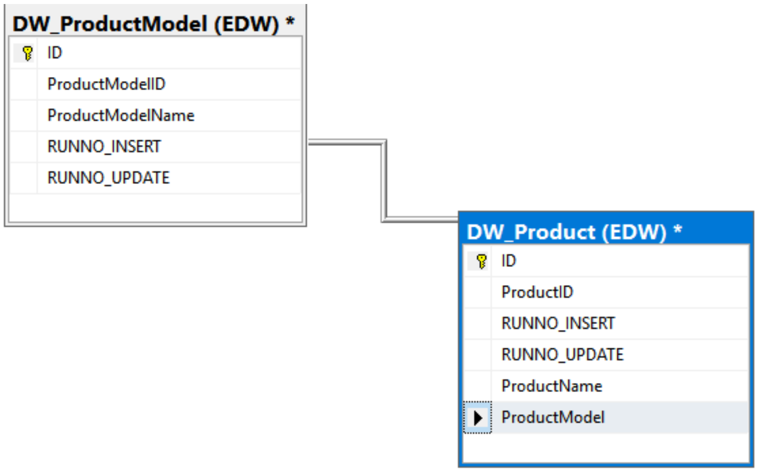


Figure 4: A Normalized Structure

The diagram below depicts Qlik’s evolution of this concept where ***Product**, **ProductModel**,* ***ProductSubcategory***, ***ProductCategory*** and ***UnitMeasure*** are managed as independent entities with relationships rather than as a single denormalized entity.

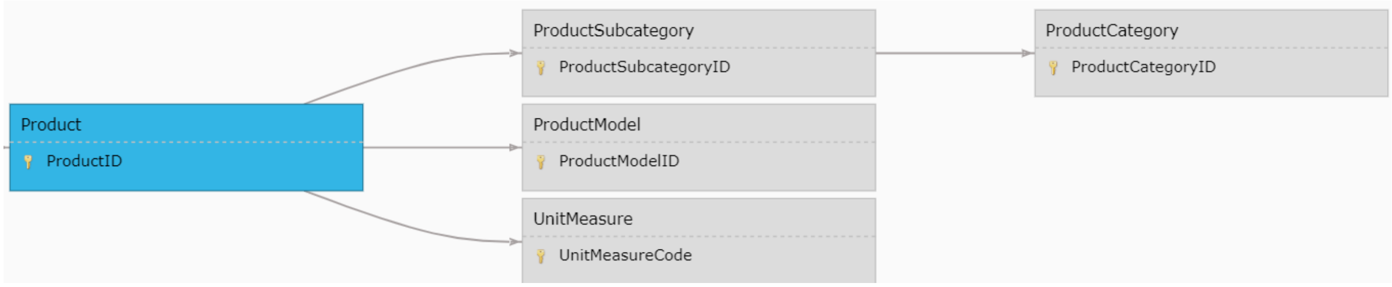


Figure 5: Qlik’s Model

Impact of Normalization on CDC Processing

Qlik provides two methods of ETL execution:

1. **Bulk** - Bulk processes the entire data set and is useful for initial data loading, or when data sources can’t provide change data.

2. **CDC** - The CDC method is efficient and required for near real-time processing. The CDC method is integrated with Qlik Replicate and reads it’s **Store Changes** tables (*__ct tables*) to process only change data. This method of processing is effective since the __ct tables notify Qlik Compose of the data that has changed since the last time the data was processed.

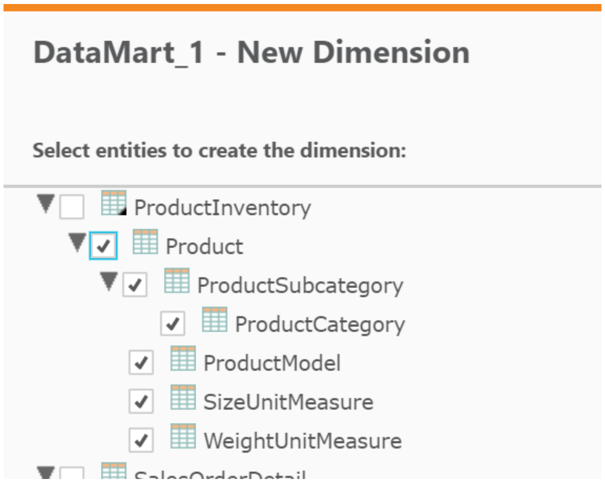
The normalized model allows Qlik Compose to process data in the __**ct** tables, detect changes and relate the normalized entities with automated ETL. If the structure were de-normalized, then the mapping would need to be manually coded to account for changes across all source tables for the ***Product*** dimension.

Normalized design will scale to N tables for 100% automated delivery of data into the central data warehouse layer and provide agility to add in additional related attributes / concepts. The work of joining these tables together to process incrementally into a denormalized dimensional concept is automated by the data mart features of Qlik Compose.

Data Marts – Automated De-Normalization and Processing

Qlik Compose data mart functionality automates the processing of data from the defined model into dimensional structures. The framework supports incremental loading of the data marts from the data warehouse and simplifies near real-time delivery to the final consumable dimensions.

The data mart dimension design automatically generates the SQL necessary to de-normalize the data warehouse tables. The illustration below highlights this instance where the data warehouse table ***UnitMeasure*** has been used in a role-play scenario with two relationships to ***Product*** for size and weight.



DataMart_1 - New Dimension

Select entities to create the dimension:

- □ ProductInventory
  - ☑ Product
    - ☑ ProductSubcategory
      - ☑ ProductCategory
    - ☑ ProductModel
    - ☑ SizeUnitMeasure
    - ☑ WeightUnitMeasure

Figure 6: Data Mart Modeling

We join ***Product***, ***ProductSubcategory***, ***ProductCategory***, ***ProductModel*** and the two ***UnitMeasure*** concepts into a single **Product** dimension as below.

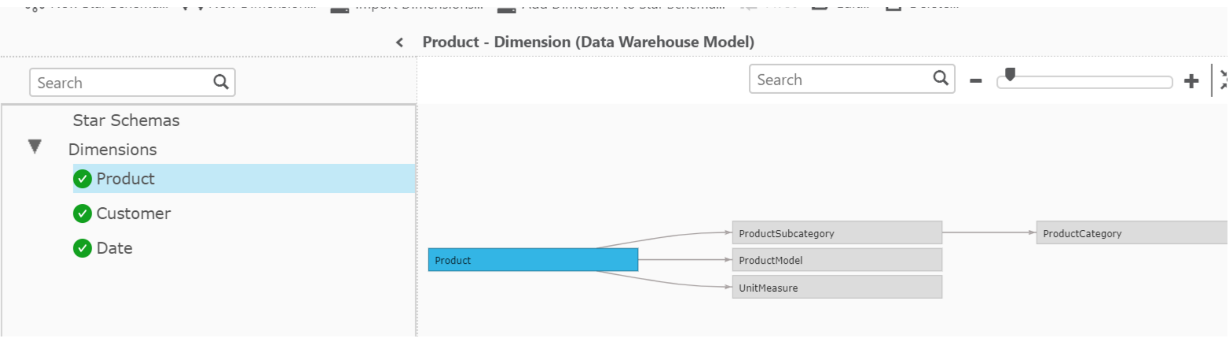
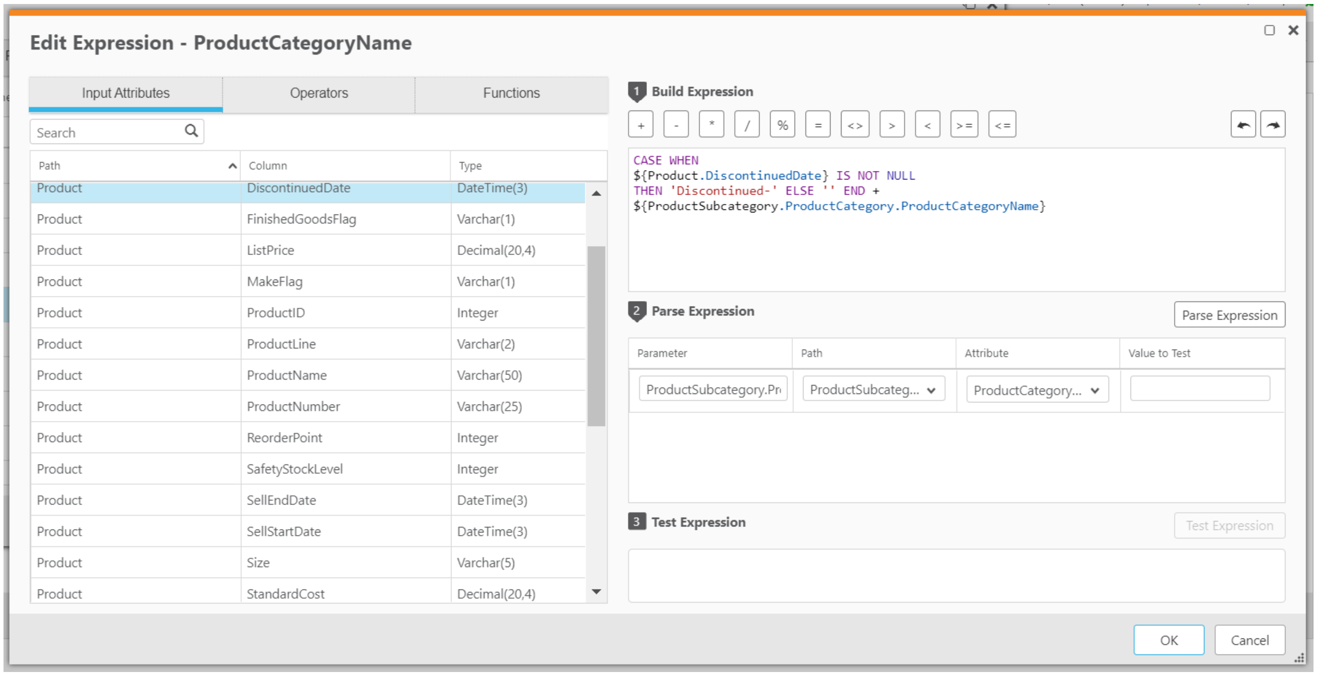


Figure 7: Qlik Compose Dimensional Modeling

The **Product** dimension can then be further edited to provide specific calculations or transformations across the data warehouse source tables. Qlik Compose provides a framework from data warehouse to data mart transformation that automatically understands data changes since the last load of the data mart.



Edit Expression - ProductCategoryName

Path	Column	Type
Product	DiscontinuedDate	DateTime(3)
Product	FinishedGoodsFlag	Varchar(1)
Product	ListPrice	Decimal(20,4)
Product	MakeFlag	Varchar(1)
Product	ProductID	Integer
Product	ProductLine	Varchar(2)
Product	ProductName	Varchar(50)
Product	ProductNumber	Varchar(25)
Product	ReorderPoint	Integer
Product	SafetyStockLevel	Integer
Product	SellEndDate	DateTime(3)
Product	SellStartDate	DateTime(3)
Product	Size	Varchar(5)
Product	StandardCost	Decimal(20,4)

1 Build Expression

```
CASE WHEN
${Product.DiscontinuedDate} IS NOT NULL
THEN 'Discontinued-' ELSE '' END +
${ProductSubcategory.ProductCategory.ProductCategoryName}
```

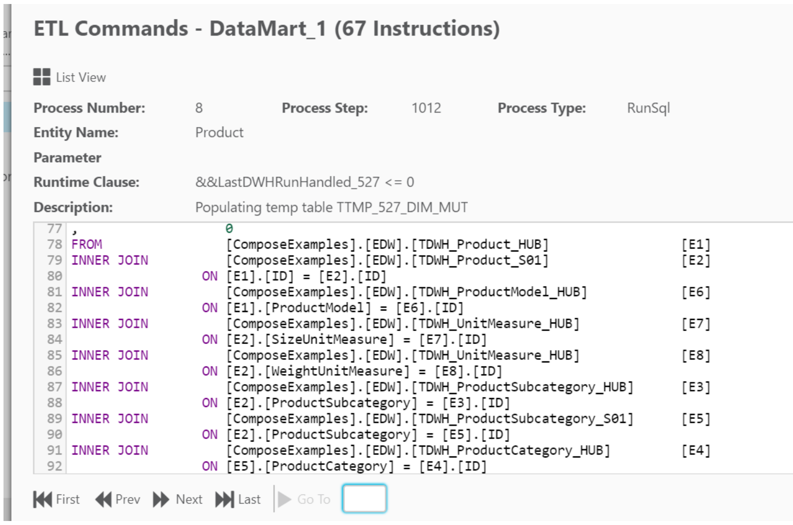
2 Parse Expression

Parameter	Path	Attribute	Value to Test
ProductSubcategory.Pr	ProductSubcateg...	ProductCategory...	

3 Test Expression

Figure 8: Expression Editor

Compose takes the above dimension design and generates appropriate code to load the dimension incrementally.



ETL Commands - DataMart_1 (67 Instructions)

Process Number: 8 **Process Step:** 1012 **Process Type:** RunSql

Entity Name: Product

Parameter

Runtime Clause: &&LastDWHRunHandled_527 <= 0

Description: Populating temp table TTMP_527_DIM_MUT

```
,                   0
FROM                [ComposeExamples].[EDW].[TDWH_Product_HUB]                  [E1]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_Product_S01]                  [E2]
                 ON [E1].[ID] = [E2].[ID]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_ProductModel_HUB]             [E6]
                 ON [E1].[ProductModel] = [E6].[ID]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_UnitMeasure_HUB]              [E7]
                 ON [E2].[SizeUnitMeasure] = [E7].[ID]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_UnitMeasure_HUB]              [E8]
                 ON [E2].[WeightUnitMeasure] = [E8].[ID]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_ProductSubcategory_HUB]       [E3]
                 ON [E2].[ProductSubcategory] = [E3].[ID]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_ProductSubcategory_S01]       [E5]
                 ON [E2].[ProductSubcategory] = [E5].[ID]
INNER JOIN          [ComposeExamples].[EDW].[TDWH_ProductCategory_HUB]          [E4]
                 ON [E5].[ProductCategory] = [E4].[ID]
```

Figure 9: The Generated SQL

Conclusion

Qlik’s data warehouse automation is architected to support near real-time data delivery and transformation for modern data warehouse implementations. The modeling techniques and data structure normalization simplify data warehouse CDC processing and the data mart functionality reduce the developer burden by automating complex ETL routines. While the approach described in this paper highlighted a simple normalization for a single source model, the same methodology can be applied for many common data warehousing scenarios. From conforming data from multiple sources for a single data domain, to handling simple lookup logic. The Qlik Compose for Data Warehouses data model can be used to abstract and automate ETL complexity in many enterprise data warehouse environments.

.

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**