Module 8: Portfolio Project, Option # 1

Eric Stiever

MIS 541: Data Warehousing in Enterprise Environments

Colorado State University - Global Campus

Professor Raquel Hicks

April 5, 2020

Option #1: Data Loading – Northwind Data

The purpose of this assignment is to create a data warehouse using the data loaded into PostgreSQL via Structured Query Language (SQL) code for the Northwind Trading Company, which is a fictitious company that does business on an international level. The company is basically set up like a grocery store that buys products from suppliers and then resells the products to their customers. The initial SQL code created the tables and loaded the transactions over the sales order period of July 4, 1996 to May 6, 1998. However, since the source data only contains the online transaction processing data (OLTP), Northwind’s senior management requested a data warehouse be created to run summary calculations to answer key performance questions. It was requested that the data warehouse perform the calculations in lieu of the business users in order to save time.

The primary business process used to develop the dimensional model is processing customer orders or sales. This process was selected as it is the impetus that ignites the product delivery process via the Northwind Trading Company’s supply chain. Without customer sales orders, there is no need for Northwind employees to order products from suppliers, place them in inventory, and then resell to customers. Key business questions that can be answered by queries against the data warehouse are as follows:

* What is the average transaction dollar amount?
* Does the average revenue per order vary by region?
* Which products sell the most?
* What is the optimal inventory level for each product?
* Which products should we discount in order to sell more to reduce our inventory?
* Which employees in each region are selling the most products? Which is our profit margin by product and by employee?

The atomic grain level that I used to create the Northwind data warehouse is one row per scan of a single product on a customer’s sales transaction as it is the finest grain for the Northwind Trading Company. This grain is also easy to understand and query. Consequently, the fact table in the Northwind schema is the Sales (fact\_sales) table, which contains foreign keys to its dimension tables’ primary keys. This linkage between foreign and primary keys establishes referential integrity (Kimball, & Ross, 2013). The Sales fact table contains summarized numerical values of its dimension tables (Kimball, & Ross, 2013). The dimension tables that explain the Sales table are Employee (dim\_employee), Product (dim\_product), Time (dim\_time), and Customer (dim\_customer). The Sales fact table supports the business process of customer order transaction as it contains granular data for each product sold in order to give senior management the requisite information to answer the aforementioned key business questions.

The first dimension table is the Employee (dim\_employee) table. Previously, this table contained the following labeled attributes: employee\_id (primary key), last\_name, first\_name, region, city, and country. After engaging with Northwind management, it was decided that two new attributes be added to the Employee dimension: hire\_date, title, and reports\_to. Next, the original Customer (dim\_customer) dimension table contained the following named attributes: customer\_id (primary key), company\_name, contact\_name, contact\_title, region, city, and country. I added the postal\_code attribute in case the marketing department wants to pull reports on customers who are located in certain postal codes. Furthermore, the Time (dim\_time) dimension table contains these specific attribute names: date\_id (primary key), date\_, day\_of\_year, month\_num, quarter\_num, and year\_. Of note, the date\_ column references the order date versus the required date and/or shipped date. Finally, the original Product (dim\_product) dimension table contained these attributes: product\_id (primary key), product\_name, category\_name, supplier\_id, category\_id, description, and quantity\_per\_unit. The reorder\_level and discontinued attributes were later added after meeting with upper management.

The fact table is labeled fact\_sales in the PostgreSQL data warehouse. The original fact table contained the following attributes: order\_id (primary key), customer\_id (foreign key), employee\_id (foreign key), product\_id (foreign key), date\_id (foreign key), required\_date, shipped\_date, unit\_price, units\_in\_stock, units\_on\_order, quantity, discount, and extended\_price. Upon further review, I added the order\_date attribute in order to track the date the product was ordered. Here is how the star schema of the Northwind Trading Company is currently constructed:

A close up of text on a white background

Description automatically generated

*Figure 1: Revised star schema for the Northwind Trading Company.*

A screenshot of a cell phone

Description automatically generated

*Figure 2: Added dimension tables and fact table in the PostgreSQL Northwind database.*

A screenshot of a social media post

Description automatically generated

*Figure 3: SQL code to create the Customer, Employee, and Product dimension tables.*

A screenshot of a social media post

Description automatically generated

*Figure 4: SQL code to create the Time dimension and Fact tables.*

*A screenshot of a cell phone

Description automatically generated*

*Figure 5: SQL code to populate the Customer dimension table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 6: SQL code to populate the Employee dimension table.*

*A screenshot of a cell phone

Description automatically generated*

*Figure 7: SQL code to populate the Product dimension table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 8: SQL code to populate the Time dimension table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 9: SQL code to populate the Sales fact table.*

*A screenshot of a cell phone

Description automatically generated*

*Figure 10: SQL code to count the number of rows in the Customer dimension table.*

*A screenshot of a cell phone

Description automatically generated*

*Figure 11: SQL code to count the number of rows in the Employee dimension table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 12: SQL code to count the number of rows in the Time dimension table.*

*A screenshot of a cell phone

Description automatically generated*

*Figure 13: SQL code to count the number of rows in the Product dimension table.*

*A screenshot of a cell phone

Description automatically generated*

*Figure 14: SQL code to count the number of rows in the Sales Fact table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 15: SQL code to show the first ten rows in the Customer dimension table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 16: SQL code to show the first ten rows in the Employee dimension table.*

*A screenshot of a social media post

Description automatically generated*

*Figure 17: SQL code to show the first ten rows in the Product dimension table.*

A screenshot of a social media post

Description automatically generated

*Figure 18: SQL code to show the first ten rows in the Time dimension table.*

A screenshot of a social media post

Description automatically generated

*Figure 19 : SQL code to show the first ten rows in the Sales Fact table.*

Lessons Learned

Writing SQL code to populate the Customer and Employee dimension tables were very simple as I just needed to reference the column names from the Customers and Employees OLTP tables. When loading the Time dimension table, I decided to use the order date as the reference date for the date\_ column as the sales fact table references the order or sales date versus the required shipping date and the shipped date. In case the data warehouse users want to group by those dates, the shipped\_date and required date columns are referenced in the sales fact table. Next, the populating the Product dimension table required creating an SQL JOIN statement to connect the Products and Categories OLTP tables where the category\_id field = product\_id field. Finally, for the Sales fact table, the one column that I had to calculate was the extended\_price as follows: quantity \* unit price \* (1-discount). Filling the Sales fact table with the necessary data proved to be a challenge as I had to create JOIN statements to link the Orders, Order Details, and Products OLTP tables to the Sales fact table. The final JOIN statement linked the date\_id field from the Time dimension table to the Sales Fact table.

Conclusion

Based on what I learned, I have advice for organizations that would like to embark on building a data warehouse system. First, I recommend the organization determine which business process(es) that the senior management team would like to analyze. Associated with the business process, it is also a good idea to ask how often the data needs to be loaded or refreshed in the data warehouse. In addition, compiling a team of Subject Matter Experts (SMEs) is very important as the people from departments such as Sales, Marketing, Finance, and IT need to be involved. However, keep in mind that too many voices may disrupt the process. Next, using clean data from existing and even new data sources is very important. Clean data means agreeing on table names, column names, data types for each column, and deciding to allow null values for certain attributes.

Furthermore, database security is also important to administer as some data will likely need to be encrypted and decrypted due to data privacy laws and other regulations. Moreover, if there is a need to purchase an ETL application in order to make sure the organization’s data is clean, secure, and uploaded correctly into the data warehouse, then make sure to get at least three quotes from ETL software vendors that will do exactly what you need them to do as referenced in the scope of work (SOW) document. This is paramount as some services are priced as add-ons that may require substantial implementation time and cost. If the organization generates a huge volume, variety and velocity of unstructured data such as social media posts and video files, then a data warehouse solution that includes unstructured data processing like Hadoop or MapReduce is needed.

References:

Dyer, J. N., & Rogers, C. (2015). Teaching Case Adapting the Access Northwind Database to Support a Database Course. *Journal of Information Systems Education*, 26(2), 85–101.

Kimball, R., & Ross, M. (2013). The data warehouse toolkit: The complete guide to dimensional modeling (3rd ed.). Hoboken, NJ: Wiley.