

Data 620 Assignment 6.1: ETL Team Project

Written by Group 1

Daanish Ahmed, Ryan Cloutier, and Bridget Van Gieson

Semester Spring 2017

Professor Majed Al-Ghandour

March 26, 2017

To: Executive Team  
From: Group 1  
Date: March 26, 2017  
Subject: Corporate ETL Process

---

Hello, thank you for reaching out to Group 1. We would like to dedicate this memo towards addressing our thoughts and answering any questions regarding the executive team's decision to merge the company's product order data into a new data mart. We will first describe our process for implementing the Extract, Transform, and Load (ETL) sequence to migrate our older data into the data mart. We will then address several questions that were directed towards our group so that we may provide the executive team a better understanding of our solution.

Before we go into detail about the ETL process, we would like to briefly address our mapping of the business situation. We have included an entity-relationship diagram (ERD) which shows the relationship between the tables currently existing in our database (see Figure 1). This diagram shows that each product must belong to one and only one business unit (BU) within the organization. Each business unit can own multiple products, though there are no requirements for a BU to own any products either. This model highlights one issue with our current data structure, and that is the lack of a strong foreign key within the product table. The two tables are linked through the BU name key, but it would be better if the product table relied on the BU ID rather than the name. This is because unlike the name, the ID is guaranteed to be unique and is therefore more reliable to use as a key between two tables. Due to data organization issues such as this, we at Group 1 believe that switching to the proposed data mart will undoubtedly improve the quality and efficiency of our company's analytical efforts.

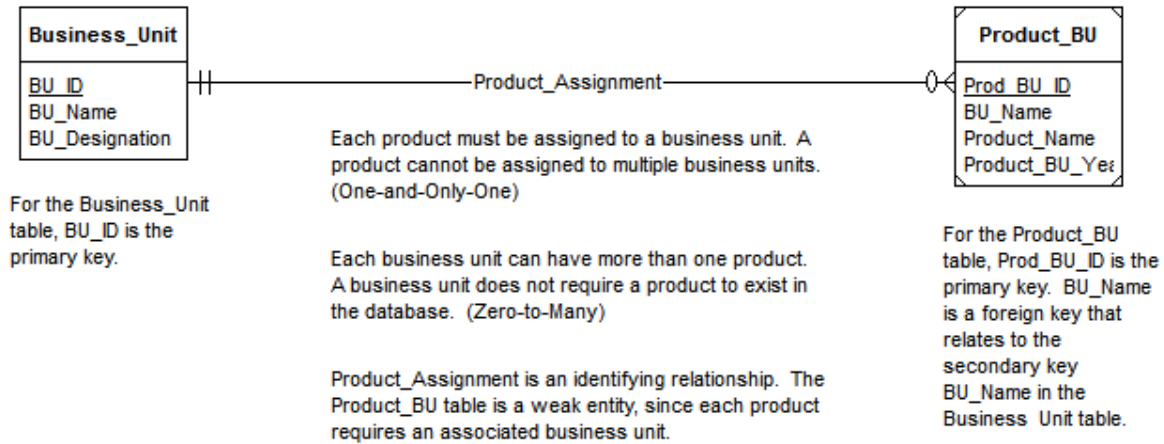


Figure 1. ERD Mapping Business Units and Products.

The process that we used to implement our ETL sequence is described in detail within the appendix (see Appendix B). We first create a new table that will store all of our data once the ETL process is complete. We need to make sure that we ran the “Week6\_business\_units.sql” file (see Appendix B) and have imported the three product data files into three separate tables using MySQL’s Table Data Import Wizard (see Appendix C) before running our code. The script then selects data from the eight requested columns in each of the three product data tables, using the “Union All” command to link the tables together. For all three of these tables, the query filters out all records with business unit designations of “Decline.” Our group also made the decision to filter out records with quantities and order totals of 0. We found that records with quantities of 0 also had order totals of 0 and vice versa, thus we deemed that these records added little to the analysis and could be deleted. Since there are only 10 such records out of 979, filtering them out will not negatively impact our data. To complete the process, we exported our new table to an output file.

To help analysts understand our results, we have included some metadata to describe the columns in the newly-created file (see Appendix A). This metadata is useful because it offers certain descriptions regarding the properties of the variables and their uses. For instance, we have

included information for each column's variable type, and whether that column accepts null values. In addition, the metadata also mentions some of the privileges given to the user regarding data entry or manipulation (including whether the user is allowed to insert or update records). Furthermore, we have manually inserted the column names into our output file to allow users to understand which variables are used in each column.

### **Granularity**

Per your question regarding whether or not the optimum level of granularity is achieved for our data mart, it is our recommendation that we extract more detailed information for our data mart. In our current presentation, we have achieved a foundational level of granularity that will provide reasonable and easily obtainable data over the course of several years. Our analyst team will be able to provide helpful analyses of product trends over longer periods of time. This data collection will assist in 5-year, 10-year, and even 15-year comparisons of our various products and successes.

The current granularity for our data mart is manageable in terms of storage. With 969 records over the course of three years (an average of 323 records per year), we have the storage capacity to expand as our product line expands while expanding our data mart. This granularity also provides essential information for the company to preserve an accurate outline of the product lines. As a team, we are confident that all of the information tracked in the data mart is essential, and there is no superfluous information being tracked.

However, we believe that data collection and analysis can provide unique insights into product trends, which will in turn assist the research and development team with identifying consumer needs and determining the best product to meet these needs. Moreover, we believe that

additional data could assist the marketing team with identifying strategies that are effective and strategies which could be abandoned.

Therefore, our recommendation is to track additional fields in the data mart. These fields should be determined with the help of the research and development team and the marketing team. The purpose of the additional fields is to track what factors are contributing to the success of products, which might include advertisements, a lack of competition, weather trends, and many other factors. A working group with members from the data analysis team, marketing team, and research and development team could formulate long-standing factors that allow tracking. These additional fields will contribute to the effective utilization of the data mart over the course of time.

### **Growth Trends (Ramon)**

Regarding the question of whether our data could show the growth trends requested by the company, we believe that our results could easily be analyzed to produce the desired information. We have executed queries to answer this question and have been able to produce the following table (see Figure 2). We have also included the SQL code required to generate this query (see Appendix B).

	BU_Designation	Year	Quantity	Order_Total
►	Growth	2012	3976	1441364
	Growth	2013	6008	434179
	Growth	2014	6287	2191977
	Mature	2012	1912	389920
	Mature	2013	4140	173882
	Mature	2014	4577	1340486

*Figure 2.* Quantity and Order Totals for 2012-2014.

By using some basic calculations, we could determine whether the growth figures for product quantity and order total match the expectations of the company for products designated as

either ‘growth’ or ‘mature.’ Unless our ETL process or the initial data is inaccurate, the results indicate that growth figures far exceed expectations. For BU designations listed as ‘growth,’ we found that product quantity sales increased by 51.1% from 2012 to 2013, and increased by another 4.6% from 2013 to 2014—resulting in a two-year average growth rate of 29.1%. This is significantly higher than the 10% yearly growth rate expected by the company. Regarding order totals for products with this designation, we found an average year-over-year growth rate of 26.0% (though with greater fluctuation in sales between individual years). Interestingly, for products designated as ‘mature,’ we found growth rates to be even higher than for those designated as ‘growth.’ For these products, quantity sales increased by 116.5% from 2012 to 2013 and 10.6% from 2013 to 2014, resulting in a two-year average of 69.7%. For order totals, the average growth rate over two years was 121.9%. These results differ drastically from the expectation that the sales figures for mature-designated products would remain roughly the same.

Although these figures sharply contrast with expectations, the format of our data file nevertheless makes it easy for Ramon to answer this question regarding yearly sales. Regarding why these answers differ from expectations, there are several possible explanations. Though we are confident in the quality of our ETL process, we acknowledge the possibility that our method may need improvement if the transferred data is not found to be consistently accurate. Of course, our output is a flat file that lacks key values, which may make analysis difficult (although we have shown that it is possible). However, it is quite likely that the issue lies with the initial data. As we have pointed out in our ERD, the formatting of the initial data sources had problems such as a lack of key values for certain tables which could make analytical efforts harder. The three input files also had columns that were inconsistent with one another, which may have complicated the ETL process. One other possibility is that there were no errors in either the initial data or the

output, but rather that these years did indeed see unusually high sales figures. In that case, we could provide a more accurate answer if we are given the data from the years preceding 2012 or following 2014 (assuming this data exists in our system).

### **Possible Other Formats**

With regards to the change in format for the ETL, we would strongly recommend remaining with the current layout. The proposed layout provides quick and helpful information but significantly limits the capacity for additional analysis. This structure inhibits other queries, as Susie warned, such as regional and seasonal trends. It also limits our ability to branch out our analysis in several years from now; it makes back-filling the data an arduous task. In a sense, the existing layout provides flexibility and the option to track many trends and run several queries, whereas the proposed layout tracks fewer trends and is simply a query that could have been run on the existing layout.

While the proposed layout seems to present “big picture” ideas, it actually limits the ability to accurately depict the big picture by failing to allow for enlightenment on the details. For example, suppose that we have the information for 2012, 2013, 2014, and 2015. In 2015, there may be record sales due to some advertisement or shortage by a competitor in January and February, but we might have experienced a sharp fall in sales for the rest of the year. However, the overall sales could still be more than the total sales for 2014. With the proposed layout, we would assume that things are gradually improving and there would be no indication that we have to change our approach in 2016. With the existing layout, discrepancies such as the 2015 example are identifiable and also provide additional information to solve them. While both layouts have

benefits and limitations, we recommend keeping the existing layout as it provides the most flexibility for current and future analysis.

### **Conclusion**

By integrating older data into the new data mart, we will be able to provide the company with essential information for determining the growth and success of the products over time. We hope that you will find our entity relationship diagram helpful in understanding the data process, especially in light of our extract, transform, and load process of the older data. We also included some information in the metadata which will assist the other analysts in understanding our work. In answering the questions regarding granularity, Ramon's queries, and the possibility of a different format, we feel confident in addressing any of your concerns with regard to the data mart. This approach will maximize the effectiveness of the data with regards to optimizing the growth and success of our products, and enable other branches, such as research and development and marketing, to benefit as well. Should you desire, we would be glad to discuss this recommendation further at your earliest convenience. Thank you.

### **Attached:**

Appendix A: Metadata

Appendix B: SQL Scripts

Appendix C: Technical Terms and Definitions



## References

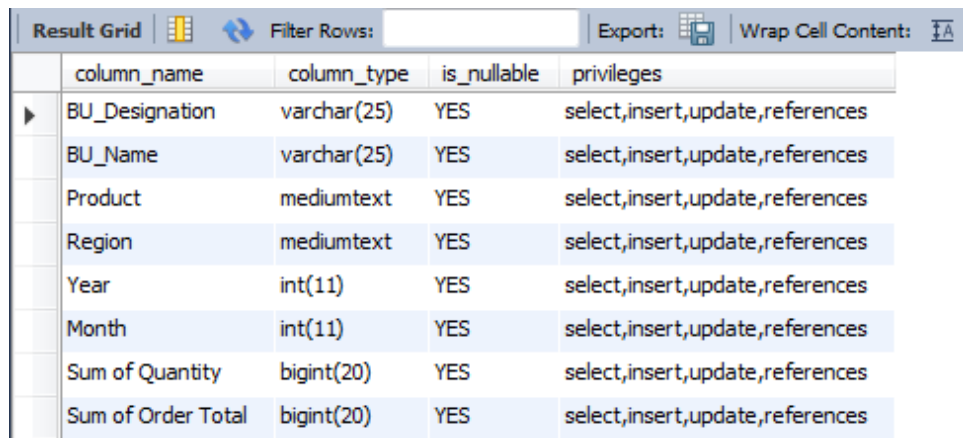
- Al-Ghandour, Majed. (2017). *Data 620-9080: Data Management and Visualization*. Online: University of Maryland University College.
- Dalvi, V. (2012, May 10). *SQL Server: Useful Metadata queries*. Retrieved March 23, 2017, from <http://www.tech-recipes.com/rx/24343/sql-server-useful-metadata-queries/>
- Inmon, W. H., & Linstedt, D. (2014). *Data Architecture: A Primer for the Data Scientist: Big Data, Data Warehouse and Data Vault*. Amsterdam: Morgan Kaufmann.
- Oracle Corporation. (2017). *MySQL 5.6 Reference Manual* [5.6.37]. Retrieved March 26, 2017, from <https://dev.mysql.com/doc/refman/5.6/en/>.

## Appendix A

### Metadata for Group 1's ETL Output File

The metadata that we included in this report was generated by typing the following code (Dalvi, 2012). Note that this process requires that the SQL code provided in our report has already been executed by the end user.

```
SELECT column_name, column_type, is_nullable, privileges  
  
FROM information_schema.columns  
  
WHERE table_name = 'etlcombine';
```



The screenshot shows a SQL query result grid with the following columns: column\_name, column\_type, is\_nullable, and privileges. The results are as follows:

	column_name	column_type	is_nullable	privileges
▶	BU_Designation	varchar(25)	YES	select,insert,update,references
	BU_Name	varchar(25)	YES	select,insert,update,references
	Product	mediumtext	YES	select,insert,update,references
	Region	mediumtext	YES	select,insert,update,references
	Year	int(11)	YES	select,insert,update,references
	Month	int(11)	YES	select,insert,update,references
	Sum of Quantity	bigint(20)	YES	select,insert,update,references
	Sum of Order Total	bigint(20)	YES	select,insert,update,references

*Table 1.* Metadata of ETL Output File.

## Appendix B

### Required SQL Scripts

#### SQL Code for ETL

# DATA 620 Assignment 6.1

# Written by Ryan Cloutier for Group 1

# Group 1 Members Ryan Cloutier, Daanish Ahmed, And Bridget Van Gieson

# Semester Spring 2017

# Professor Dr. Majed AL-Ghandour

#

# First, we create the grp1 database and select it. We drop the database if it already exists.

# This part of the script should only be run once at the very beginning, unless you are starting over.

Drop Database if exists grp1;

Create Database grp1;

Use grp1;

# Before the rest of this script can be run, the Week6\_business\_units.sql needs to be run.

# The three data files `2012\_product\_data\_students.csv`, `2013\_product\_data\_students.csv`, and

# `2014\_product\_data\_students.csv` need to be imported as tables to the grp1 database

# TO achieve this right click on the grp1 database and choose the Data Table Import Wizard

# Browse to the location of the csv and select one (will need to do same steps for all three)

# Select the csv and import the data by clicking next and accepting the pre-configured settings

```

#

# This script is designed to remove the ETLCombine table and repopulate it from the
# 2012, 2013, and 2014 Product Data Students tables that were created by importing
# CSV files to SQL Tables. Depending on the size of the table it would be more efficient
# to append the new data to the table instead of rebuilding it entirely

#

# First step is to drop the ETLCombine table so we can rebuild it

Drop Table if exists `ETLCombine`;

# Next is to create the table and creating with the 'as' statement will create the table with the
columns we selected

Create Table ETLCombine as

# The first select statement is pulling data from the `2012_product_data_students` Table that was
created from importing the csv

# of the same name

SELECT

    BU_Designation,

    product_bu.BU_Name,

    Product,

    Region,

# The Prod_BU_Year column needed to be formatted as 'Year' for the final output

    Prod_BU_Year as Year,

    Month,

```

# The Quantity column needed to be formatted as 'Sum of Order Total' for the final output

Quantity as 'Sum of Quantity',

'Order Total' as 'Sum of Order Total'

From '2012\_product\_data\_students' AS z

# The product\_bu table is joined to get the BU\_Name

JOIN

product\_bu ON product\_bu.Product\_Name = z.Product

# Since the first section was pulling from the 2012 data and no date was provided in the

'2012\_product\_data\_students' table

# the product\_bu table was joined also where Prod\_BU\_Year = 2012

AND Prod\_BU\_Year = 2012

# The business\_unit table is joined to get the BU\_Designation

JOIN

# Since it was determined we did not want any row where the BU\_Designation was = to 'Decline'

business\_unit ON business\_unit.BU\_Name = product\_bu.BU\_Name and BU\_Designation

!= 'Decline'

# Our group chose to not include when quantity was 0

Where Quantity > 0

# Union All is then used to append all of the second table to the first

Union All

# The Second select statement is pulling data from the '2013\_product\_data\_students' Table that was created from importing the csv

# of the same name

SELECT

BU\_Designation,

product\_bu.BU\_Name,

Product,

Region,

# The Prod\_BU\_Year column needed to be formatted as 'Year' for the final output

Prod\_BU\_Year as Year,

Month,

Quantity\_1+Quantity\_2 as `Sum of Quantity`,

Quantity\_1+Quantity\_2\*`Per-Unit Price` as `Sum of Order Total`

From `2013\_product\_data\_students` AS z

# The product\_bu table is joined to get the BU\_Name

JOIN

product\_bu ON product\_bu.Product\_Name = z.Product

# Since the SECOND section was pulling from the 2013 data and no date was provided in the

`2013\_product\_data\_students` table

# the product\_bu table was joined also where Prod\_BU\_Year = 2013

AND Prod\_BU\_Year = 2013

# The business\_unit table is joined to get the BU\_Designation

JOIN

# Since it was determined we did not want any row where the BU\_Designation was = to 'Decline'

I filtered it here

```

    business_unit ON business_unit.BU_Name = product_bu.BU_Name and BU_Designation !=
'Decline'

# Our group chose to not include when quantity was 0, In this case there were two quantity fields
that made the totals

    Where Quantity_1+Quantity_2 >0

# Union All is then used to append all of the third table to the first

Union All

# The Third select statement is pulling data from the '2014_product_data_students' Table that
was created from importing the csv

# of the same name

SELECT

    BU_Designation,

    product_bu.BU_Name,

    Product,

    Region,

# The Prod_BU_Year column needed to be formatted as 'Year' for the final output

    Prod_BU_Year as Year,

    Month,

    Quantity as `Sum of Quantity`,

    `Order Subtotal` - `Quantity Discount` as `Sum of Order Total`

```

```

From `2014_product_data_students` AS z

# The product_bu table is joined to get the BU_Name

JOIN

product_bu ON product_bu.Product_Name = z.Product

# Since the third section was pulling from the 2014 data and no date was provided in the
`2014_product_data_students` table

# the product_bu table was joined also where Prod_BU_Year = 2014

AND Prod_BU_Year = 2014

# The business_unit table is joined to get the BU_Designation

JOIN

# Since it was determined we did not want any row where the BU_Designation was = to 'Decline'
I filtered it here

business_unit ON business_unit.BU_Name = product_bu.BU_Name and BU_Designation !=
'Decline'

# Our group chose to not include when quantity was 0

Where Quantity >0

# The final output of the script is then sorted by each column

Order by BU_Designation,

BU_Name,

Product,

Region,

`Year`,

Month,

```



```
`Sum of Quantity`,  
`Sum of Order Total`;
```

# Then the new table can be used to output to a csv file

# The entire table was selected because the table is already formatted how we need

```
SELECT * FROM grp1.etlcombine
```

# The results are then saved to the Comma separated values file where the comma is added

between the fields using the 'Fields terminated by' command

into outfile 'C:/temp/G1\_output\_final.csv' fields terminated by ',';

### **SQL Code for Question #2 (Ramon):**

# This script is designed to return the business unit designation, year, and the sum of the quantity

# and order total for all products in the etlcombine table. It will group the results by BU

# designation (either 'Growth' or 'Mature') and year.

# The script requires that the user has already executed our ETL script to create, populate and

# export the etlcombine table.

```
SELECT BU_Designation, Year, SUM(`Sum of Quantity`) AS Quantity, SUM(`Sum of Order  
Total`) AS Order_Total  
FROM etlcombine  
GROUP BY BU_Designation, Year;
```

## Appendix C

### Technical Terms and Concepts

**Table Data Import Wizard** – this is a feature included in MySQL that we used to import our product data CSV files. Under the “schemas” tab in the left of the MySQL GUI, we click on the database we created and right click on “tables.” From there, we click “Table Data Import Wizard.” We browse to the file we are interested in (in this case, the product data CSV files for 2012, 2013, and 2014). We then click next and create a new table to store the data in all three cases. We leave all of the columns selected so that we may import all of the data in each file. We then click next until the process is complete and the data has been imported. This process must be done for all three CSV files.