ETL Team Project

Group 5: Susmitha Karjala, Vidya Koduri, Tanushree Kumar, Hayat Wondimu, and Ian Khelil

Assignment 6.1: ETL Team Project

Data 620 - Spring 2024

Professor Majed Al-Ghandour

University of Maryland Global Campus

Due: February 20, 2024

TO: Director of Data Analytics

FROM: Group 5

DATE: February 20, 2024

SUBJECT: ETL Process for Business Units

Summary

We began by running the SQL script "week6_bu_grad.sql" to create two tables named "business_unit" and "Product_BU." Though we are met with the challenge of missing metadata descriptions, we're tasked with figuring out the schema, understanding the company's strategic business units like "On The Go" and "Snack" and how they fit under their designations like "Growth" or "Decline." We start by conducting an ETL process using SQL, pulling, transforming, and loading data from 2017, 2018, and 2019 product order records to create actionable insights.

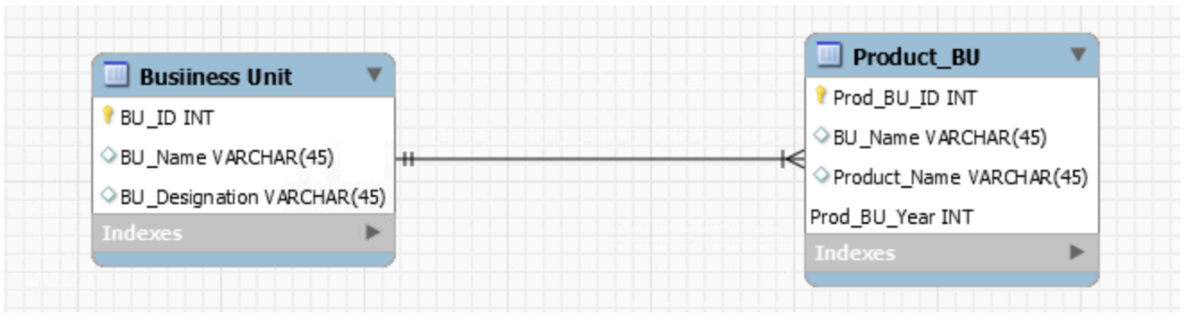
ERD, ETL Documentation, and Metadata

Entity Relationship Diagram (ERD) for Business Unit and Product Data:

Before a detailed exploration of the ETL process, it is crucial to establish a clear understanding of the entity-relationship dynamics between the tables, specifically the Business Units and Products tables, in our current database.

Figure 1

ERD for Business_Unit and Product_BU



The Entity-Relationship Diagram (ERD) (refer to Figure 1) visually represents these relationships, revealing the structure and connections within our data. This diagram highlights that each product is specifically linked to a singular business unit (BU) within the organization, while each business unit has the potential to be associated with multiple products.

In the Business_Unit table, BU_ID is designated as the primary key, ensuring uniqueness and mandatory values. Correspondingly, BU_Name and BU_Designation are linked, creating a unique identification for each Business Unit.

In the Product_BU table, Prod_BU_ID serves as the primary key, and BU_Name acts as a foreign key connected to Business_Unit.BU_Name. This relationship enforces referential integrity, mandating that values in Product_BU.BU_Name must exist in the Business_Unit table before being entered. This setup establishes a one-to-many relationship with Business_Unit, allowing a Business Unit to be associated with multiple products over different years.

Datafile Preparation:

This initiative encompasses three years' worth of order data from 2017_product_data_students-final.csv, 2018_product_data_students-final.csv, and 2019_product_data_students-final.csv, offering a comprehensive insight into our historical product orders. We've employed the "Table Data Import Wizard" (this is a functionality integrated into MySQL for importing data files) to seamlessly import these data files into the current database. Our goal is to successfully execute the ETL process, leveraging consolidated data to foster a culture of well-informed decision-making within our organization.

ETL Process:

The product data for each year exhibits unique structures. Initially, we standardized the format of each year's data to ensure a consistent structure, enhancing the usability of the output file for subsequent analysis. Each data file contains details on months, regions, and products, and our consolidation process revolves around grouping order totals and quantities based on these columns. Please refer to Appendix B for the complete SQL extraction process.

In the 2017 data files, both Quantity and Order Total were provided. Subsequently, we grouped the order total and quantity, considering the month, region, and product. Moving to the 2018 data file, the quantity was distributed across two columns. To obtain the total order quantity, we summed up Quantity1 and Quantity2. Additionally, considering the provided price per unit, we calculated the total order price by multiplying the price per unit with Quantity1 and Quantity2. The 2019 data file adopts a different structure, presenting information on Order Subtotal and discount. By subtracting the discount from the Order Subtotal, we obtained the total order. This consolidation was executed by grouping data based on month, region, and product, ensuring a standardized representation across all years for further analysis.

Subsequently, we correlated product information with our existing product and business data, extracting business Designation and name based on the respective years. Our organization specifically seeks information pertaining to business designations categorized as "Growth and Mature." Consequently, we have filtered the data to exclusively include information related to these designations. After the data refinement process, we have generated a consolidated table for future reference and utilization.

To aid analysts in comprehending our findings, we have incorporated metadata to reveal the columns in the newly generated file (refer to Figure 1 in Appendix A). The metadata provides insights into the data types, lengths, and purposes of each column in the dataset, facilitating a clear understanding of the information stored in the table. Excel was utilized to create the metadata based on the columns present in our new tables.

Granularity

After carefully considering the granularity of the data, we believe it's essential to assess whether the current level aligns with our objectives and if adjustments are necessary to enhance our decision-making. Increasing or decreasing the granularity should be done with a lot of thought because "If data is not granular enough, it cannot support a flexible analytical pattern. On the other hand, if the data is too granular, the data consumes too much space. In addition, when the data is too granular, it becomes unwieldy to handle" (Inmon et al., 2021, p.99-100).

Our current granularity level aggregates data based on the following values: Business Unit Designation, Business Unit Name, Product, Region, Year, Month, Sum of Quantity, and Sum of Order Total. This level of granularity provides a comprehensive overview of sales performance across different products, regions, and periods. It also allows for meaningful analysis of sales trends, product performance, and seasonal patterns.

Depending on our analytical requirements and how they could change in the future, we may benefit from extracting more detailed information to gain deeper insights into specific aspects of our business. For example, we could consider extracting transactional data which could capture potentially individual sale transactions. It could contain attributes such as customer demographics, promotional activities, and pricing variations. This level of granularity could allow us to perform more detailed analyses of the market, customer buying habits, and the effectiveness of promotions.

Alternatively, we could also consider adopting a coarser level of granularity by aggregating data at higher levels. For example, we could aggregate sales data at the product level instead of individuals, which would simplify our analysis while still providing valuable insights into the performance of the overall category. If we drop certain fields, such as the State or Country, it could also give us a coarser level of granularity without significantly sacrificing analytical depth, especially if our focus is primarily on regional and nationwide trends.

By considering both, a finer and coarser level of granularity, we ultimately aim to find a balance between analytical depth and data manageability. Extracting more detailed information can provide richer insights but may also increase the complexity and storage requirements of the data. On the other hand, adopting a coarser granularity level can streamline analysis and reduce data volume but may limit the depth of insights.

Our decision regarding granularity should align closely with our analytical objectives and the specific questions we seek to answer through our analysis. By assessing the trade-offs between detail and manageability, we can ensure that the granularity level chosen optimally supports our decision-making processes and strategic goals.

Ramon (Expert Analyst)

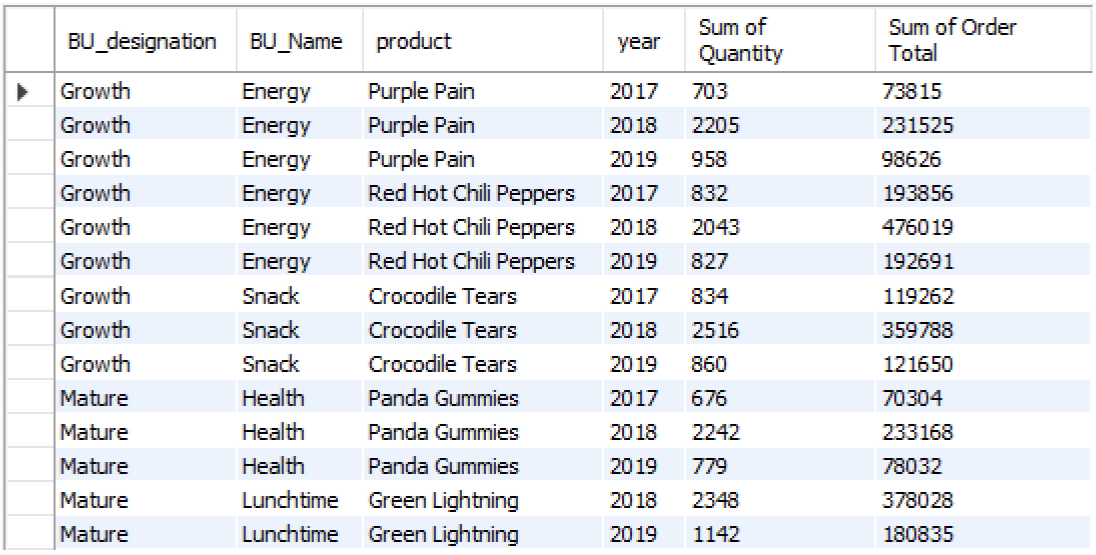
The layout of our data file enables complex analysis of various potential factors. In the example you provide, evaluating growth in the “Growth” and “Mature” business unit designations can be done with a query such as the one included here:

```
SELECT
    BU_designation, BU_Name, product, year, Sum(`Sum of Quantity`) AS 'Sum of
Quantity',
    Sum(`Sum of Order Total`) AS 'Sum of Order Total'
FROM
    g5_output_final
GROUP BY BU_designation , BU_Name , product , year
ORDER BY BU_designation, BU_Name, product, year;
```

This query creates a table that can easily be interpreted to demonstrate that we saw a huge increase in sales for many “Growth” and “Mature” designated products from 2017 to 2018 however that same jump was not preserved from 2018 to 2019.

Figure 2

Output Table Created from SQL Query



BU_designation	BU_Name	product	year	Sum of Quantity	Sum of Order Total
Growth	Energy	Purple Pain	2017	703	73815
Growth	Energy	Purple Pain	2018	2205	231525
Growth	Energy	Purple Pain	2019	958	98626
Growth	Energy	Red Hot Chili Peppers	2017	832	193856
Growth	Energy	Red Hot Chili Peppers	2018	2043	476019
Growth	Energy	Red Hot Chili Peppers	2019	827	192691
Growth	Snack	Crocodile Tears	2017	834	119262
Growth	Snack	Crocodile Tears	2018	2516	359788
Growth	Snack	Crocodile Tears	2019	860	121650
Mature	Health	Panda Gummies	2017	676	70304
Mature	Health	Panda Gummies	2018	2242	233168
Mature	Health	Panda Gummies	2019	779	78032
Mature	Lunchtime	Green Lightning	2018	2348	378028
Mature	Lunchtime	Green Lightning	2019	1142	180835

Our data file is striated by region also to allow further examination of region-specific growth for both of these designations and the products within. Another benefit of the layout of our data file is the possibility of evaluating monthly and yearly sales data, allowing Ramon to pinpoint the times of year when sales peak and times when we are not meeting our projections. The low level of granularity in our data set will allow Ramon to answer very precise questions and organize the data in ways that give a clear picture of sales for each product, in each region, and broken down by nearly any timeframe that is necessary.

Additionally, the variables captured by our data set allow us to easily evaluate multiple factors related to products and business unit designations. Ramon will have an easy time presenting information on any key variable under a sizable range of timeframes and regions to help establish strong paths forward for business strategy. Moving forward with the data layout currently captured in this file will benefit the organization by providing a high amount of clarity in evaluating the position of each product or the efficacy of the marketing approach for the “Growth” and “Mature” designated units, improving the capability of the leadership team to adjust business unit designation or strategies for each on a product, region, or temporal basis.

Different Format for the Data

The proposed layout, which Bobby supported, aligns data for 2017, 2018, and 2019 side by side and could simplify the visualization of growth trends over these years. This arrangement might make it easier to quickly identify how our sales quantities and order totals have evolved, providing a clear growth picture. However, Susie mentioned something important. This new setup might make it harder for us to look at other helpful info, like how a particular product is doing across all regions or if we have any monthly sales patterns. It could limit how we dig into our data for other insights.

Given the above considerations, we recommend maintaining our existing data layout. This approach ensures that we can conduct comprehensive analyses across various business dimensions.

Creating views or summary reports that focus on growth trends without altering the fundamental structure of your data mart involves designing SQL queries that aggregate and compare data across different periods. These SQL views or reports will enable you to analyze year-over-year growth for various business dimensions, such as sales, quantities, and order totals, for different products, regions, and business units. For example, we can create views summarising total sales and quantities by year for each product or region and then calculate the growth percentage from one year to the next. Summary reports can be generated from SQL views or directly from queries, formatted to highlight key growth metrics. These reports might include visualizations like charts or tables that show growth rates, making it easier for stakeholders to digest the information.

By keeping our current data organization and integrating specialized views that focus on growth, we're setting up our data management system in a way that can quickly grow and adjust over time. This approach means our data setup is prepared to handle more information as our business expands and is flexible enough to meet new or changing analysis needs without significant overhauls.

This allows us to preserve the comprehensive analytical capabilities of our data mart while also providing focused insights into our growth trends. This strategy ensures that our data infrastructure supports immediate decision-making needs and long-term strategic planning.

Tidy Data

Datasets have values that are collected at multiple levels and different types of observational units so they require careful organization. During tidying, each type of observational unit should be stored in its own table to avoid inconsistencies, which practices the principle of database normalization. A tidy data has a structured format where variables are columns, observations are rows, and values live in cells. This setup helps streamline data management and analysis in data marts and makes tasks like filtering, merging, and data manipulation performed by analysts more impactful.

The proposed layout demonstrates one of the five types of messy data, specifically “Column headers are values” since it includes the year in the column header for example, “Sum of Quantity for 2017”. In the existing layout (Question 6), each row represents a unique combination of "Designation" and "Year," with "Sum of Quantity" and "Sum of Order Total" as separate columns. This layout follows the principles of tidy data since it follows the one variable per column and one observation per row rule.

Conclusion

Overall, our ETL process effectively handles the provided data from the 2017, 2018, and 2019 "product_data_students-final.csv" files but also ensures consistency through standardization. Calculating total prices and aggregating order total and quantity gave us uniformity across the datasets while our decision on granularity showed selecting the appropriate level should be based on our analytical objectives. Ultimately, deciding on the use of the existing data layout reveals our dedication to accessing diverse insights without the limitations that the proposed layout may introduce.

Attachments:

1. G5_output_final.csv (final output table generated from SQL query)

2. G5_etl.sql (SQL code used to extract, transform, and load product data to produce one final output table)

References

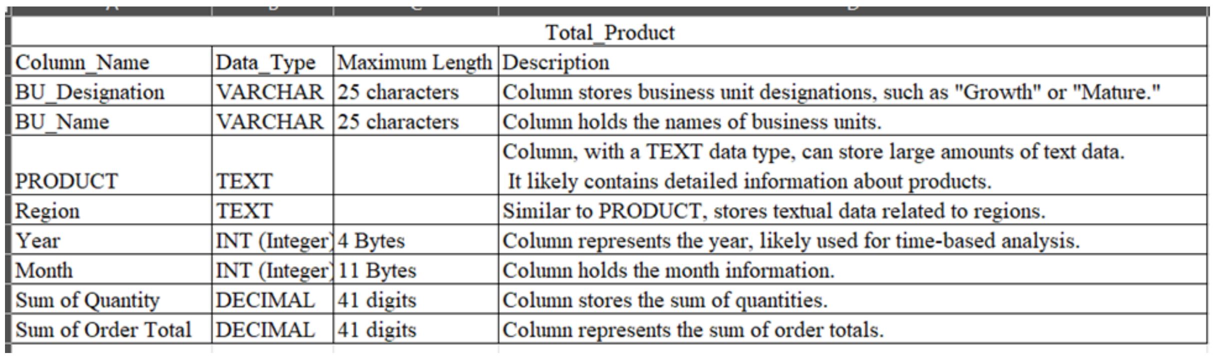
Hadley Wickham, Tidy Data, Journal of Statistical Software, August 2014, https://www.jstatsoft.org/index.php/jss/article/view/v059i10/v59i10.pdf

Inmon, B., Levins, M., & Srivastava, R. (2021). *Building The Data Lakehouse.* Technics Publications Llc.

Appendix A

Figure 1

Metadata details for the Output file



Total_Product			
Column_Name	Data_Type	Maximum Length	Description
BU_Designation	VARCHAR	25 characters	Column stores business unit designations, such as "Growth" or "Mature."
BU_Name	VARCHAR	25 characters	Column holds the names of business units.
PRODUCT	TEXT		Column, with a TEXT data type, can store large amounts of text data. It likely contains detailed information about products.
Region	TEXT		Similar to PRODUCT, stores textual data related to regions.
Year	INT (Integer)	4 Bytes	Column represents the year, likely used for time-based analysis.
Month	INT (Integer)	11 Bytes	Column holds the month information.
Sum of Quantity	DECIMAL	41 digits	Column stores the sum of quantities.
Sum of Order Total	DECIMAL	41 digits	Column represents the sum of order totals.

Appendix B

```
# DATA 620 Assignment 6
# Written by <GROUP-5> Susmitha Karjala, Vidya Koduri, Tanushree Kumar, Hayat
Wondimu, and Ian Khelil
# Semester <Spring>
# Professor <Majed AL-Ghandour>
#
#
#
/**
Before executing this script, follow these steps:
1. Execute the week6_bu_grad.sql script, which generates a schema named
"Candy" containing tables named business_unit and Product_BU.
2. Provide the data files: 2017_product_data_students-final.csv,
2018_product_data_students-final.csv, and
2019_product_data_students-final.csv.
      These files need to be imported into the Candy schema.
3. Utilize the "Table Data Import Wizard" option to import the data files
from the previous step into the Candy schema.
4. As a result of the import, three tables will be created:
      2017_product_data_students, 2018_product_data_students, and
2019_product_data_students.

**/
DROP TABLE if exists Total_Product;
CREATE TABLE Total_Product AS
SELECT
bu.BU_Designation,
bu.BU_Name,
Product,
Region,
Year,
Month,
SUM(Sum_quant) AS 'Sum of Quantity',
SUM(Sum_Order) AS 'Sum of Order Total'
FROMi
    (
        SELECT
           2017 as year,
            month,                                            -- Month
of the data from 2017_product_data_students-final
            region,                                                 --
Region of the product from 2017_product_data_students-final
            product,                                          --
Product Name from 2017_product_data_students-final
            SUM(Quantity) AS 'Sum_quant',             -- Total Quantity of
the product from 2017_product_data_students-final
            SUM(`Order Total`) AS 'Sum_Order'         -- Total Order Value
of the product from 2017_product_data_students-final
        FROM
            `2017_product_data_students-final`
        GROUP BY
                  year,
            month,
            region,
            product
        UNION ALL
        SELECT
           2018 as year,
            month,
                                 -- Month of the data from
2018_product_data_students-final
            region,
                                       -- Region of the product from
2018_product_data_students-final
            product,
                                 -- Product Name from
2018_product_data_students-final
            SUM(Quantity_1 + Quantity_2) AS 'Sum_quant',
                  -- Total Quantity of the product from
2018_product_data_students-final
            SUM((Quantity_1 + Quantity_2) * `Per-Unit Price`) AS 'Sum_Order'
      -- Total Order Value of the product from
2018_product_data_students-final
        FROM
            `2018_product_data_students-final`
        GROUP BY
            year,
            month,
            region,
            product
        UNION ALL
        SELECT
            2019 as year,
            month,
                                     -- Month of the data from
2019_product_data_students-final
            region,
                                           -- Region of the product from
2019_product_data_students-final
            product,
                                     -- Product Name from
2019_product_data_students-final
            SUM(Quantity) AS 'Sum_quant',
                         -- Total Quantity of the product from
2019_product_data_students-final
            SUM(`Order Subtotal` - `Quantity Discount`) AS 'Sum_Order'
            -- Total Order Value of the product from
2019_product_data_students-final
        FROM
            `2019_product_data_students-final`
        GROUP BY
            year,
            month,
            region,
            product
    ) AS tp
JOIN
    product_bu AS pb ON pb.Product_Name = tp.product AND pb.Prod_BU_Year =
tp.year           -- Joining our newly created table with the product_bu
table based on product name and year
JOIN
    business_unit AS bu ON bu.BU_Name = pb.BU_Name
                                     -- Joining the business_unit table with
our new table based on the business unit name
WHERE bu.BU_Designation IN ('Growth', 'Mature')
                                     -- Excluding results which do not have
the Business Unit Designation of "Growth" or "Mature"
GROUP BY
bu.BU_Designation,
bu.BU_Name,
Product,
Region,
Year,
Month;
# Arranging the data in ascending order based on the column headers below:
Select * from Total_Product
ORDER BY
      BU_Designation asc,
      BU_Name asc,
    Product asc,
    Region asc,
    Year asc,
    Month asc,
 `Sum of Quantity` asc,
 `Sum of Order Total` asc;
```