Data Warehousing

Design and Implementation of a Data Warehouse for Dominick's Finer Foods

Prepared By:
Mithilesh Menakuru

TABLE OF CONTENTS

Table of Contents

Section 1: Introduction

Dominick's Finer Foods (DFF) founded in 1918 by a Sicilian immigrant Dominick DiMatteo was a major grocery store chain in the Chicago area. The company evolved from a small deli to become Chicago's second-largest supermarket chain. Over the decades, DFF expanded significantly and they underwent several changes in ownership which impacted DFF's performance.

They were one of the first to feature in-store delicatessens and frozen food sections and pioneered online grocery ordering well before e-commerce became the norm. They partnered with Starbucks to install coffee bars in stores to enhance the customer's in-store experience. DFF's success was largely credited to its ability to stay ahead of the market trends. The company also employed the latest technologies to consistently improve efficiency and customer service, staying committed to meeting customers' needs and requirements.

However, the company faced significant challenges after it was acquired by Safeway Inc. as the changes implemented were unfamiliar to the long-time local customers and impacted their shopping experience. They replaced the familiar local products with Safeway's in-house brands disrupting the shopping experience of long-term customers. Additionally, the entry of competitors such as Target, Walmart, Kroger, etc. resulted in a drop in DFF's market share stressing the impact of integrating local stores into national operations [1].

The primary objective of this project is to design and implement a data warehouse for DFF using the store-level data collected from 1989-1994. This involves integrating the data from various sources ensuring data consistency to provide a consolidated view of the company's operations. The data warehouse will facilitate historical data analysis to understand the patterns in customer behavior, sales trends, and marketing strategies. This information is essential to understand the factors that contributed to DFF's success and downfall.

Furthermore, The data warehouse will enable the creation of reports tailored to the organization and its stakeholders. These reports will provide valuable insights into various aspects of the business enabling targeted marketing and strategic decision-making. Successful implementation of this project will allow us to understand and optimize the operations.

Understanding of the data

The DFF's historical data is divided into General Files and Category Specific Files.

A) General Files:

1) Customer Count File: This dataset provides daily and store-specific customer traffic information, sales data, and coupon usage (if any) data for different product categories at Dominick's Fine Foods. Each row or record in the file represents a single day's data related to the number of customers and a summary of their purchases for a specific store. In addition to customer count, the dataset tracks coupon usage and sales revenue for various product categories, from groceries to pharmacy, cosmetics, alcohol, bakery, frozen items, and more.

The data in the Customer count file is mainly related to 3 things -

1. **Customer Traffic:**
    - There is a column (field) called CUSTCOUNT available in the Customer count file. This field indicates the total number of customers who made purchases on a particular given day.
    - The other fields DATE and STORE represent the particular date on which sales happened and the store number (at which store sales occurred) respectively
2. **Sales Data:**
    - This file contains multiple fields or columns that capture sales in dollars for different departments such as bakery, beer, cosmetic, pharmacy frozen foods, etc. For instance, the columns in the data file are - BAKERY, BEER, COSMETIC, PHARMACY, and FROZEN which capture related sales information per day respectively.
    - Each department-specific column indicates the total sales in dollars for that day and specific store.
3. **Coupon Usage:**
    - Not all departments are offering coupons but most of the departments are offering coupons for customers and in the dataset there are multiple fields to capture the coupon usage for various departments.
    - For instance, columns such as BAKCOUP, COSMCOUP, LIQCOUP, and PHARCOUP indicate how many coupons were redeemed in those departments.

- MANCOUP tracks the number of manufacturer coupons redeemed.

This dataset contains sales and coupon data broken down into categories such as bakery, frozen items etc. Time-series data is recorded per store and day, allowing for temporal analysis.

2) Demographics file: Dominick's Fine Foods contains detailed store-specific demographic data derived from the **1990 U.S. Census** for the **Chicago metropolitan area**. This data provides insight into the socio-economic characteristics of the population in the trading area of each store, helping to understand the customer base and tailor store offerings accordingly.
The data includes:

- **Age distribution** (e.g., % population under 9 or over 60)
- **Ethnic composition** (e.g., % Blacks and Hispanics)
- **Education levels** (e.g., % College Graduates)
- **Income distribution** (e.g., Median income, income variance)
- **Household characteristics** (e.g., household size, % single-person households)
- **Employment data** (e.g., % working women, % unemployed)
- **Homeownership and home value** (e.g., % households with mortgages, mean household value)
- **Shopping behavior** (e.g., shopper types like "Avid Shoppers", "Hurried Shoppers")

This data was processed by **Market Metrics** to provide a demographic profile for each store's trading area, which can be leveraged for various analytical purposes, such as pricing strategies, shelf management, product placement, and understanding customer behavior.

B) Category Specific Files:

1) UPC Files: The UPC datasets contain detailed information related to the products sold at Dominick's Fine Foods, identified by their UPC codes. The data includes product-specific details such as category (commodity code), item code, product description, size and the number of items per case. This information helps to track individual products across the store's inventory system and facilitates better organization and sales tracking at the item level.
Some important columns in the dataset are:

1. **UPC Code:** The last 5 digits of the code identify the specific product, and the preceding digits represent the manufacturer. This allows for easy and fast identification of both the product and manufacturer within the dataset.
2. **Commodity Code (COM_CODE):** A classification system used by Dominick's to categorize products in different sections and departments. While multiple commodity codes may exist within a single dataset (file), a single commodity code will not be repeated across different files.
3. **Item Code (NITEM):** This is a product tracking number across UPCs. If 2 different UPCs have the same item code, it indicates the one is a newer version of the other (though this system is not foolproof).
4. **Description:** This product description field includes various special characters to denote specific attributes:
    - # - Indicates that the product is available in Combo stores (Stores with a pharmacy)
    - < - Denotes trial size products (though this is not always accurate)
    - ~ - Marks discontinued products
    - $ and * - Have no specific meaning
5. **Case:** The number of items in a case delivered by the manufacturer. This field is mainly used for inventory purposes but not visible to customers.

2) Understanding of the Movement Data by UPC:

The movement data files (named wxxx, where xxx is a category acronym) provide weekly sales data at the store level for each unique product (UPC) across various categories. This data includes sales, price, quantity, and other relevant metrics that can be used to analyze sales performance and retail strategy over time.

Key aspects to understand about this data include:

1. **UPC Code as a Key**: The **UPC** is a unique identifier for products, which is crucial when merging data with other files (e.g., product descriptions or categories).
2. **Price, Quantity, and Movement**:

- Sometimes products are sold in bundles (e.g., "3 for $2"). The **qty** variable represents the bundle size, and the **move** variable represents the actual number of items sold (not the number of bundles).
- To calculate total sales dollars, the formula is: Sales=Price×MoveQty\text{Sales} = \frac{\text{Price} \times \text{Move}}{\text{Qty}}Sales=QtyPrice×Move

3. **Profit**: This variable represents the gross margin for each product sold. It does not directly reflect replacement or last transaction costs but an average acquisition cost (AAC), which adjusts based on historical inventory and current purchases.
4. **Sales**: This is a promotional flag that tracks whether the product was sold with a promotion (Bonus Buy, Coupon, or Price Reduction). However, it's noted that this field may not always be set consistently.
5. **Data Validation**: The **OK** variable is a flag indicating if the data for a particular week is valid. If the flag is set to 0, that data should not be used in analysis.

Metadata information

1. Customer Count File:

Column Name	Description	Datatype	Length
DATE	Date of the Observation	Character	6
Week	Week Number	Numeric	8
Store	Store Code	Numeric	8
BAKCOUP	Bakery Coupons Redeemed	Numeric	8
BAKERY	Bakery Sales in Dollars	Numeric	8
BEER	Beer Sales in Dollars	Numeric	8
BOTTLE	Bottle Sales in Dollars	Numeric	8
BULK	Bulk Sales in Dollars	Numeric	8
BULKCOUP	Bulk Coupons Redeemed	Numeric	8
CAMERA	Camera Sales in Dollars	Numeric	8

CHEESE	Cheese Sales in Dollars	Numeric	8
CONVFOOD	Conventional Foods Sales in Dollars	Numeric	8
COSMCOUP	Cosmetics Coupons Redeemed	Numeric	8
COSMETIC	Cosmetics Sales in Dollars	Numeric	8
CUSTCOUN	Customer Count	Numeric	8
DAIRCOUP	Dairy Coupons Redeemed	Numeric	8
DAIRY	Dairy Sales in Dollars	Numeric	8
DELI	Deli Sales in Dollars	Numeric	8
DELICOUP	Deli Coupons Redeemed	Numeric	8
DELIEXPR	Deli Express Sales in Dollars	Numeric	8
DELISELF	Deli Self Service Sales in Dollars	Numeric	8
FISH	Fish Sales in Dollars	Numeric	8
FISHCOUP	Fish Coupons Redeemed	Numeric	8
FLORAL	Floral Sales in Dollars	Numeric	8
FLORCOUP	Floral Coupons Redeemed	Numeric	8
FROZCOUP	Frozen Items Coupons Redeemed	Numeric	8
FROZEN	Frozen Items Sales	Numeric	8
FTGCCOUP	Food-to-Go Coupons Redeemed	Numeric	8
FTGCHIN	Food-to-Go Chinese Sales in Dollars	Numeric	8
FTGICOUP	Food-to-Go Coupons Redeemed	Numeric	8
FTGITAL	Food-to-Go Italian Sales in Dollars	Numeric	8
GM	General Merchandise Sales in Dollars	Numeric	8
GMCOUP	General Coupons Redeemed	Numeric	8
GROCCOUP	Grocery Coupons Redeemed	Numeric	8
GROCERY	Grocery Sales in Dollars	Numeric	8
HABA	Health and Beauty Aids Sales in Dollars	Numeric	8

HABACOUP	Health and Beauty Aids Coupons Redeemed	Numeric	8
JEWELRY	Jewelry Sales in Dollars	Numeric	8
LIQCOUP	Liquor Coupons Redeemed	Numeric	8
MANCOUP	Manufacturer Coupons Redeemed	Numeric	8
MEAT	Meat Sales in Dollars	Numeric	8
MEATCOUP	Meat Coupons Redeemed	Numeric	8
MEATFROZ	Meat-Frozen Sales in Dollars	Numeric	8
MISCSCP	Misc. Coupons Redeemed	Numeric	8
MVPCLUB	MVP	Numeric	8
PHARCOUP	Pharmacy Coupons Redeemed	Numeric	8
PHARMACY	Pharmacy Sales in Dollars	Numeric	8
PHOTCOUP	Photo Coupons Redeemed	Numeric	8
PHOTOFIN	Photo	Numeric	8
PRODCOUP	Produce Coupons Redeemed	Numeric	8
PRODUCE	Produce Sales in Dollars	Numeric	8
PROMCOUP	Promotion Coupons Redeemed	Numeric	8
PROMO	Promotion Sales in Dollars	Numeric	8
SALADBAR	Salad Bar Sales in Dollars	Numeric	8
SALCOUP	Salad Coupons Redeemed	Numeric	8
SPIRITS	Spirits Sales in Dollars	Numeric	8
SSDELICP	Self Service Deli Sales in Dollars	Numeric	8
VIDCOUP	Video Coupons Redeemed	Numeric	8
VIDEO	Video Sales in Dollars	Numeric	8
VIDEOREN	Video Rentals (Dollar Amounts)	Numeric	8
WINE	Wine Sales in Dollars	Numeric	8

2) Demographic File:

Column Name	Description	Datatype
age9	% Population under age 9	Float
age60	% Population over age 60	Float
ethnic	% Blacks & Hispanics	Float
educ	% College Graduates	Float
nocar	% With No Vehicles	Float
income	Log of Median Income	Float
incsigma	Std dev of Income Distribution (Approximated)	Float
hsizeavg	Average Household Size	Float
hsize1	% of households with 1 person	Float
hsize2	% of households with 2 persons	Float
hsize34	% of households with 3 or 4 persons	Float
hsize567	% of households with 5 or more persons	Float
hh3plus	% of households with 3 or more persons	Float
hh4plus	% of households with 4 or more persons	Float
hhsingle	% of households with 1 person	Float
hhlarge	% of households with 5 or more persons	Float
workwom	% Working Women with full-time jobs	Float
sinhouse	% Detached Houses	Float
density	Trading Area in Sq Miles per Capita	Float
hval150	% of Households with Value over $150,000	Float
hval200	% of Households with Value over $200,000	Float
hvalmean	Mean Household Value (Approximated)	Float
single	% of Singles	Float
retired	% of Retired	Float

unemp	% of Unemployed	Float
wrkch5	% of working women with children under 5	Float
wrkch17	% of working women with children 6 - 17	Float
nwrkch5	% of non-working women with children under 5	Float
nwrkch17	% of non-working women with children 6 - 17	Float
wrkch	% of working women with children	Float
nwrkch	% of non-working women with children	Float
wrkwch	% of working women with children under 5	Float
wrkwnch	% of working women with no children	Float
telephn	% of households with telephones	Float
mortgage	% of households with mortgages	Float
nwhite	% of population that is non-white	Float
poverty	% of population with income under $15,000	Float
shopcons	% of Constrained Shoppers	Float
shophurr	% of Hurried Shoppers	Float
shopavid	% of Avid Shoppers	Float
shopstr	% of Shopping Stranges	Float
shopunft	% of Unfettered Shoppers	Float
shopbird	% of Shopper Birds	Float
shopindx	Ability to Shop (Car and Single Family House)	Float
shpindx	Ability to Shop (Car and Single Family House)	Float

3. UPC Files:

Column Name	Description	Type	Length
UPC	UPC number used to identify the product and the manufacturer.	Numeric	8

COM_CODE	Dominick's commodity code (category).	Numeric	8
NITEM	Item code used to track products.	Numeric	8
DESCRIP	Product Description (with special characters)	Character	20
SIZE	Product size (Quantity)	Character	6
CASE	Number of items in a case (for internal inventory use)	Numeric	8

4) Movement Files:

Column Name	Description	Type	Length
UPC	Unique Product Code (UPC) for each item	Numeric	8
STORE	Store identifier	Numeric	3
WEEK	Week number during which sales occurred	Numeric	3
MOVE	Number of units sold	Numeric	8
PRICE	Retail price for the product	Numeric	8
QTY	Number of items bundled together	Numeric	3
PROFIT	Gross profit margin (%)	Numeric	8
SALE	Sales promotion code ('B', 'C', 'S')	Character	8

OK	Data validity flag (1 = valid, 0 = invalid)	Numeric	3

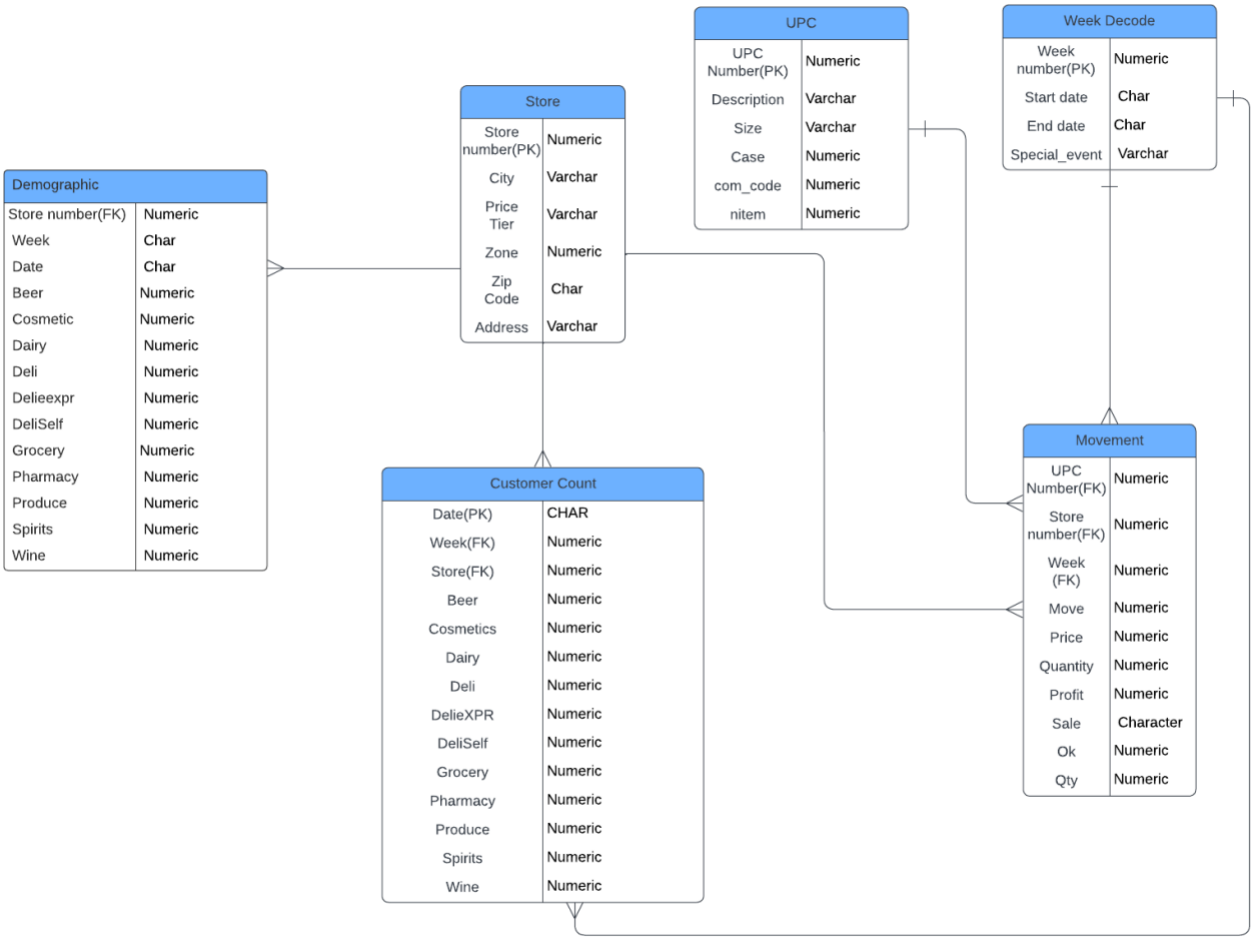
5) Store data:

Variable	Description	Type	Length
Store	Store Code	Numeric	3
City	Name of city	Character	20
Price Tier	Segmentation of prices	Character	10
Zone	Zone where the store is located	Numeric	6
Zip Code	Zip code of the store	Numeric	8
Address	Address of the store	Character	24

6) Week Decode Table:

Variable	Description	DataType	Length
Week#	Week Number	Numeric	3
Start	Start Date of that particular week	Date	20
End	End Date of that particular week	Date	20
Special Events	Holidays or Festival periods	Character	20

Entity Relationship Diagram



Section 2: Business Questions

Business Questions

1. How does the profit margin of toothpaste vary by brand?

   Impact:

   Toothpaste is a staple product in our everyday life and has constant demand. Understanding the profit margin based on the brand can help DFF make informed decisions on pricing, promotion, and shelf space allocation. This will also provide them leverage to negotiate better terms with the manufacturers. It will also allow DFF to renegotiate with brands with lower profit margins or replace them with better-performing brands.

   Support:

   For doing preliminary analysis, we have used wtpa.csv which contains movement data for toothpaste. Then I created a pivot table with upc as rows and calculated the average profit margin and total sales = [MOVE*PRICE]/QTY. I then sorted it in descending order of profit margin to show the UPCs with the highest profit margin. In the later stages, we can join the data from the UTC file for toothpaste to then do further analysis.

Row Labels	Sum of TOTAL SALES	Average of PROFIT
1111341101	₹3,26,582.49	3882.98%
1111319120	₹1,08,967.34	2923.70%
1851527243	₹3,837.18	2833.27%
1111323101	₹2,25,230.06	2758.12%
2260064631	₹43.25	2727.93%
1111381660	₹11,388.63	2501.19%
1111374102	₹4,01,777.65	2335.77%
1111320202	₹1,18,305.18	2335.21%
1851527239	₹1,698.37	2319.96%
1111329780	₹73,106.72	2285.85%
1111381680	₹6,343.64	2234.82%
1111381650	₹1,438.23	2104.13%
1111315120	₹9,365.67	2088.49%
1111374801	₹1,19,397.43	2085.45%
1851527235	₹4,825.92	2067.52%
1111363761	₹2,09,512.87	2037.47%
3320018180	₹6,145.95	2022.24%
1111363961	₹2,27,068.84	1992.14%
1111363861	₹1,63,336.63	1984.50%
2260062031	₹1,486.71	1830.03%
1111363701	₹1,56,990.97	1826.63%
1111325381	₹47,113.64	1825.87%
1111374242	₹39,191.01	1789.12%
1111379161	₹1,91,531.29	1785.78%
3320018190	₹2,396.47	1782.39%
1111349510	₹25,957.74	1699.79%
1111363801	₹1,20,946.36	1696.76%
1111363901	₹1,23,975.72	1691.38%
1111379152	₹24,200.36	1640.34%

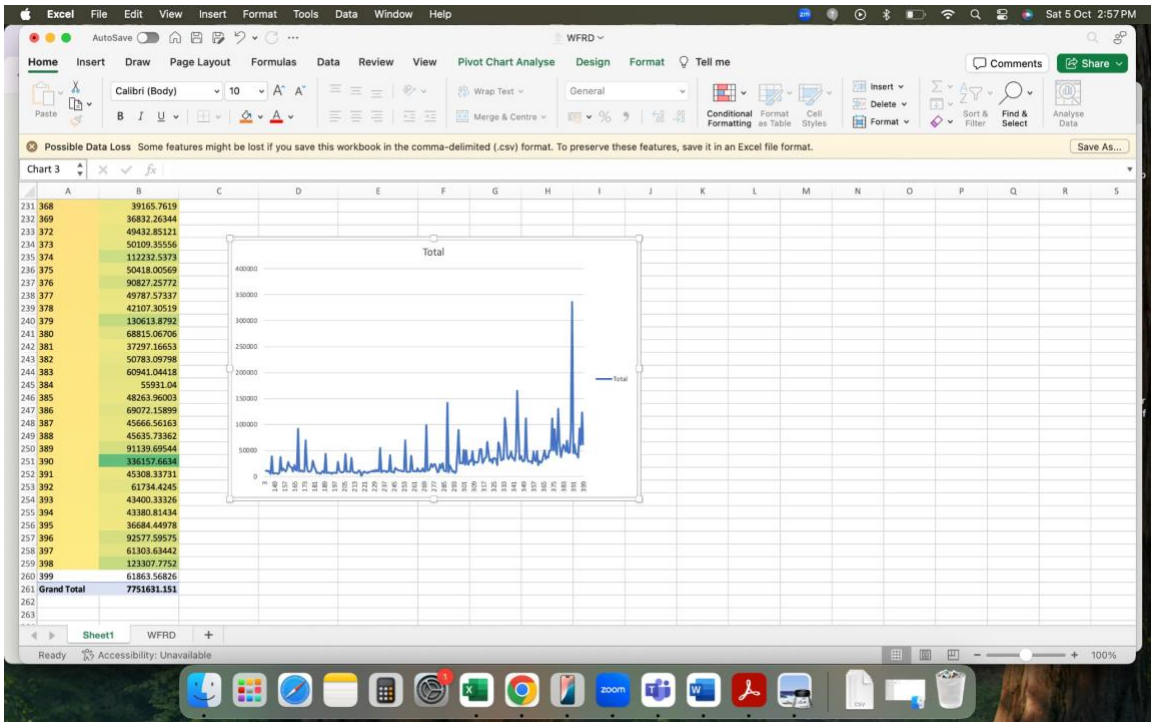
2. What is the trend in frozen dinner sales? How does it vary across different stores?

   Impact

   The frozen dinner market is growing as people are getting more and more busy and looking for convenient and hassle-free ways to get in their meals. Understanding the sales trend for this product category helps DFF to capitalize on the growing need for frozen dinners. This provides valuable insights that will help DFF manage its inventory, and plan product placement, promotions, and overall business strategy. It also helps to identify the regional preferences tailoring the product stocking specific to stores to increase the overall sales.

   Support:

   For this, I used WFRD.csv which has the data for frozen food. For preliminary analysis, I have created a pivot table with Week as rows and calculated the total sales = (MOVE * PRICE)/QTY. Further, I have added stores as filter to filter data as per stores for in-depth analysis. Finally we have highlighted the table using a heatmap to highlight the low and high sales and also displayed a pivot table to see the trend.



3. Which product is purchased most by college graduate students?

   Impact:

   Understanding what a key demographic is purchasing and their preference is invaluable. College graduates are an important set of customers even though they might not generate the most revenue, the insights from this analysis can help with targeted marketing and product selection in the stores near the universities. It can also be leveraged to capture a potential loyal customer base.

   Support:

   We need to join the movement data using the UPC to answer this question and get the product codes. Then we would join the store demographics to get the percentage of graduate students and then filter stores having above-average college students. Finally, we would aggregate the sales for these stores.

4. What is the average price and sales volume of women's shampoo products in stores located in areas with above-average percentages of working women?

Row Labels	Sum of TOTAL SALES	Average of PROFIT
102	₹74,932.02	1629.52%
128	₹72,978.30	1795.47%
122	₹59,618.61	1649.19%
12	₹58,820.51	1686.35%
100	₹57,160.72	1640.56%
130	₹55,334.77	1506.06%
98	₹54,150.80	1582.68%
75	₹54,102.32	1597.44%
126	₹53,643.65	1590.31%
121	₹53,244.91	1652.95%
132	₹50,902.41	1551.68%
114	₹49,604.09	1523.60%
86	₹48,984.22	1607.93%
112	₹48,447.61	1643.20%
109	₹47,085.92	1586.51%
74	₹46,877.80	1515.14%
8	₹45,649.49	1620.81%
124	₹45,011.11	1592.22%
32	₹44,453.21	1531.34%
133	₹44,401.17	1437.33%
71	₹44,252.74	1549.73%
101	₹43,702.75	1525.23%
123	₹43,556.36	1557.78%
131	₹42,861.49	1553.04%
107	₹42,177.07	1589.55%
18	₹42,052.70	1475.25%
73	₹41,534.20	1492.59%
80	₹40,771.60	1594.16%
84	₹40,593.97	1556.07%

Impact:

This analysis combines the demographic data with the product performance providing DFF with valuable insights to place targeted marketing. Working women are a significant consumer group, and understanding their purchase patterns can help DFF tailor their offering and adjust its pricing strategy accordingly. It also creates an opportunity to provide value-oriented options in stores with more working women traffic which can potentially increase sales.

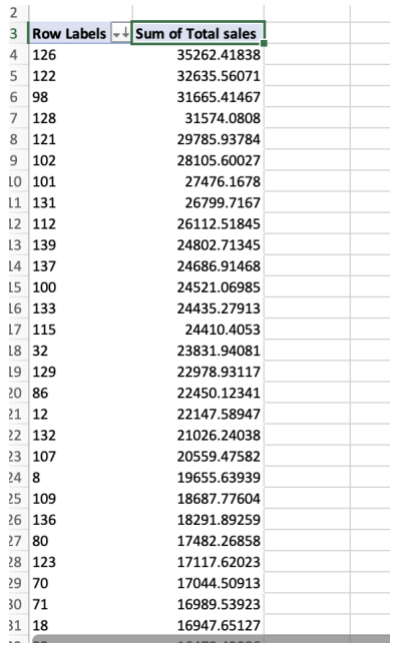
Support:

To answer this question, we need to join the movement, store, and store demographics data. Next, we need to calculate the average percentage of women across all the stores. Then find the stores with above average percentage of working women. Finally, calculate the average sales and sales volume.

5. What is the percentage contribution of cookies towards the total sales during the holiday season (i.e., Christmas and New Year) of the year 1994?

   Impact:

The seasonal sales patterns are important to focus on as it's the time of the year when people buy products for hosting events and gifting. We have chosen cookies as purchased

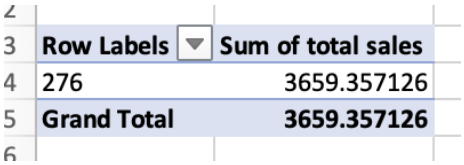


Row Labels	Sum of Total sales
126	35262.41838
122	32635.56071
98	31665.41467
128	31574.0808
121	29785.93784
102	28105.60027
101	27476.1678
131	26799.7167
112	26112.51845
139	24802.71345
137	24686.91468
100	24521.06985
133	24435.27913
115	24410.4053
32	23831.94081
129	22978.93117
86	22450.12341
12	22147.58947
132	21026.24038
107	20559.47582
8	19655.63939
109	18687.77604
136	18291.89259
80	17482.26858
123	17117.62023
70	17044.50913
71	16989.53923
18	16947.65127

to be consumed as well as gifts. Understanding a particular product's sales pattern can provide valuable insights into the overall holiday sales. DFF can leverage this information for planning holiday promotions, store layout planning, and inventory management during the holiday season.

Support:

To solve this question we need to join the movement and upc data for the weeks of holiday as per the manual. Calculate the total sales of cookies. Then calculate the total sums. Finally, calculate the percentage contribution.



Row Labels	Sum of total sales
276	3659.357126
Grand Total	**3659.357126**

6. Which store has the highest store traffic for the last quarter of 1994?

Impact

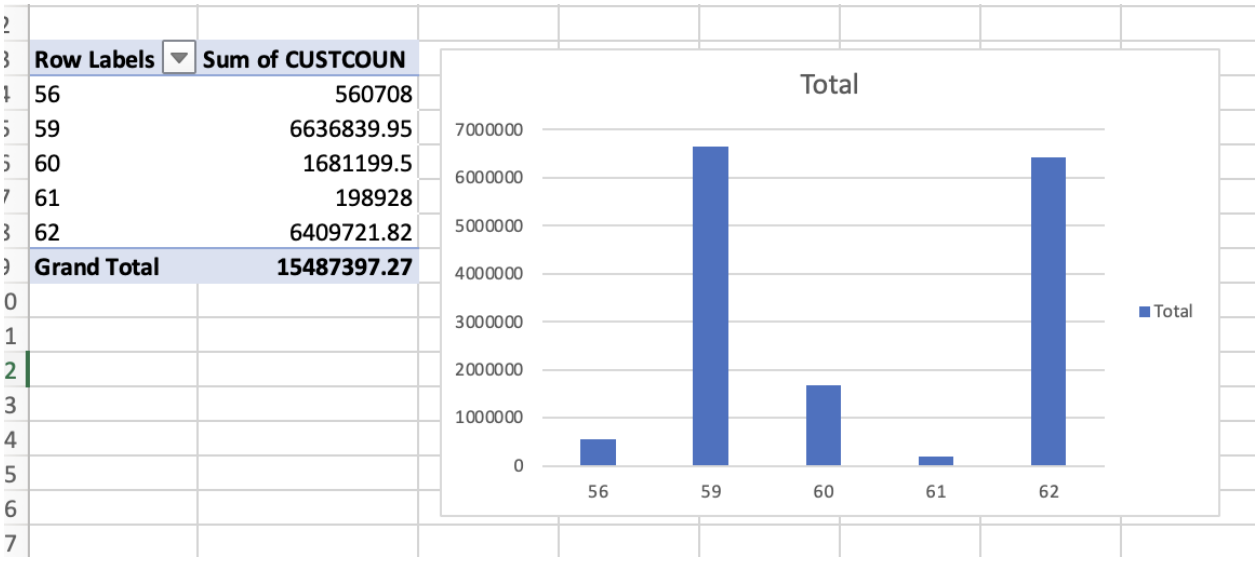
Store traffic is a key indicator of the overall business performance and sales. By identifying the highest-traffic store, DFF can analyze what contributed to its success and

replicate the same across other locations. The focus on the last quarter of the year is important as it's the holiday season, and the insights from this can be used to improve the performance across other stores as well.

Support

For this, I used the CCOUNT.csv and filtered out the data for weeks 264-276 [The week code for the last quarter of 1994]. I then created pivot tables with stores as a row number and sum of CustCount to calculate the store traffic. I finally displayed the bar chart showing the result.

As per this, store number 59 which is Crystal Lake [As per the Dominicks Manual] store had the highest store traffic in the last quarter.



Row Labels	Sum of CUSTCOUN
56	560708
59	6636839.95
60	1681199.5
61	198928
62	6409721.82
Grand Total	**15487397.27**

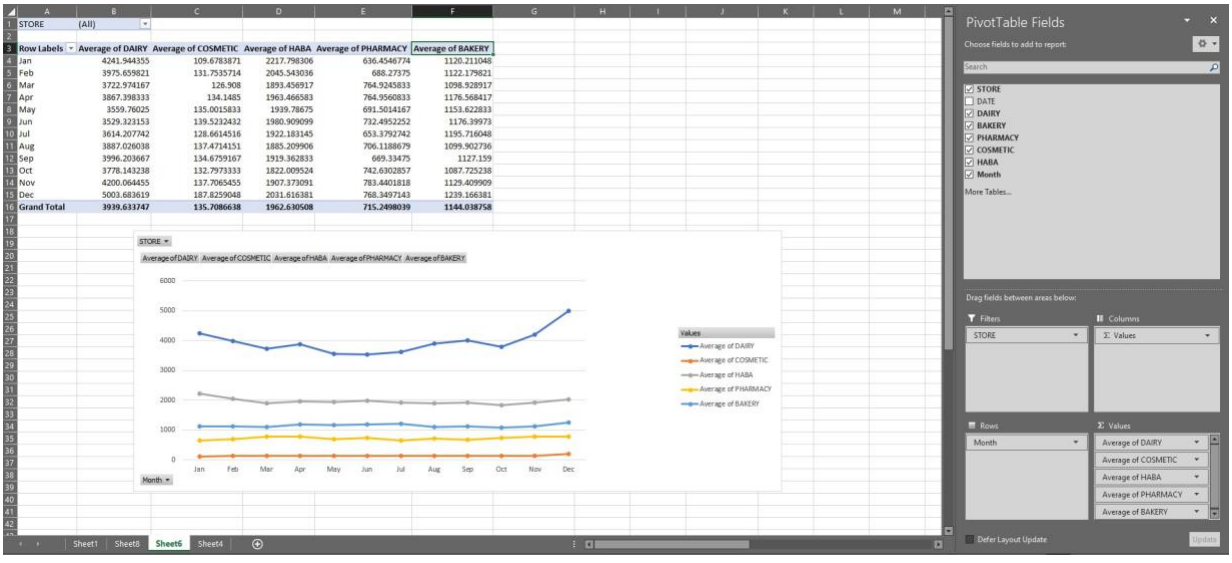
7. What are the monthly average sales amounts for the BAKERY, DAIRY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores located in Naperville and Schaumburg during the year 1994, and how do these averages compare among the different departments within these locations?

Impact:

Analyzing the monthly average sales amounts for the BAKERY, DIARY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores in Naperville and Schaumburg during 1994 is beneficial for DFF as it provides insights into consumer preferences and sales trends. Understanding which departments perform better can guide inventory management, marketing strategies, and resource allocation. This data-driven approach enables DFF to optimize promotions and improve customer satisfaction, ultimately enhancing profitability.

Support:

For this, I have used CCOUNT.csv file date. Naperville and Schaumburg store numbers are 54, 115 and 48, 117 respectively. As we are looking the data related to these stores for the year 1994, we initially fetched Store, Date, Diary, Bakery, Pharmacy, Cosmetic, and Hba categories data from CCOUNT.csv to different files. In the new file, we filtered records accordingly. Our new files consist of 4 stores of data related to those categories in the year 1994. We have added a new column called month which stores the month of each observation respectively. After that, we created a pivot table and a line chart as mentioned below.



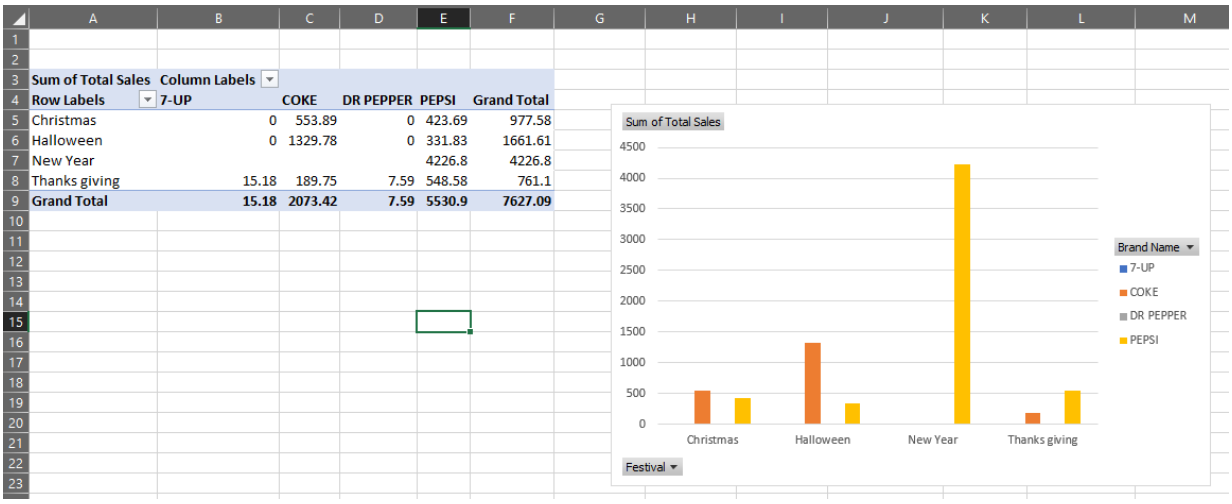
The line chart showcases different category's average sales amount monthly and how it varies month by month and how it is performing compared to the monthly average sales of other categories. We can drill down this further by using the store filter to know more about monthly average sales at each store level. From the chart we can see that, the Diary category is performing better than any other category as its average monthly sales amount is higher than others. We can see trends in the DIARY category month by month. In the month of May, the diary sales went down compared to all other months' performance of the diary category and its monthly sales were highest in Dec 1994.

8. Which soft drink brand (PEPSI, COKE, 7 UP, DR PEPPER) recorded the highest sales performance at the Orland Park store during the 1992 holiday season, specifically across Halloween, Thanksgiving, Christmas, and New Year? [6]

Impact:

Understanding which soft drink brand PEPSI, COKE, 7 UP, or DR PEPPER performed best during key holiday periods in 1992 at the Orland Park store provides valuable insights for Dominick's Fine Foods (DFF). This information helps in optimizing product placements, tailoring promotions, and aligning stock levels with consumer demand during high-traffic seasons. It also enables DFF to better negotiate with suppliers and brands based on performance metrics, ultimately improving profitability and customer satisfaction.

Support:



Sum of Total Sales	Column Labels				
Row Labels	7-UP	COKE	DR PEPPER	PEPSI	Grand Total
Christmas	0	553.89	0	423.69	977.58
Halloween	0	1329.78	0	331.83	1661.61
New Year				4226.8	4226.8
Thanks giving	15.18	189.75	7.59	548.58	761.1
Grand Total	15.18	2073.42	7.59	5530.9	7627.09

For this, I have used UPCSDR.csv and wsdr.csv files. First I have found out the PEPSI, COKE, 7-UP, and DR PEPPER product's UPC codes from the UPCSDR.csv file. After finding out the codes, I filtered the records in a wsdr.csv file. I selected store number 84 which is the store number of the Orland Park store. I fetched the sales of those brands exclusively during the holiday season. In 1992 holiday season weeks were - 120, 164, 168, and 172. Using the filters I have fetched sales of brands during the holiday season and stored the results in a separate sheet. From that data, I created the pivot table and the bar chart.

From the bar chart, we can understand that on New Year and Thanksgiving week - the PEPSI brand has more sales than all others, on Halloween and Christmas week, COKE has the highest sales recorded. In this way, we can analyze for any area or city or across all stores if needed.

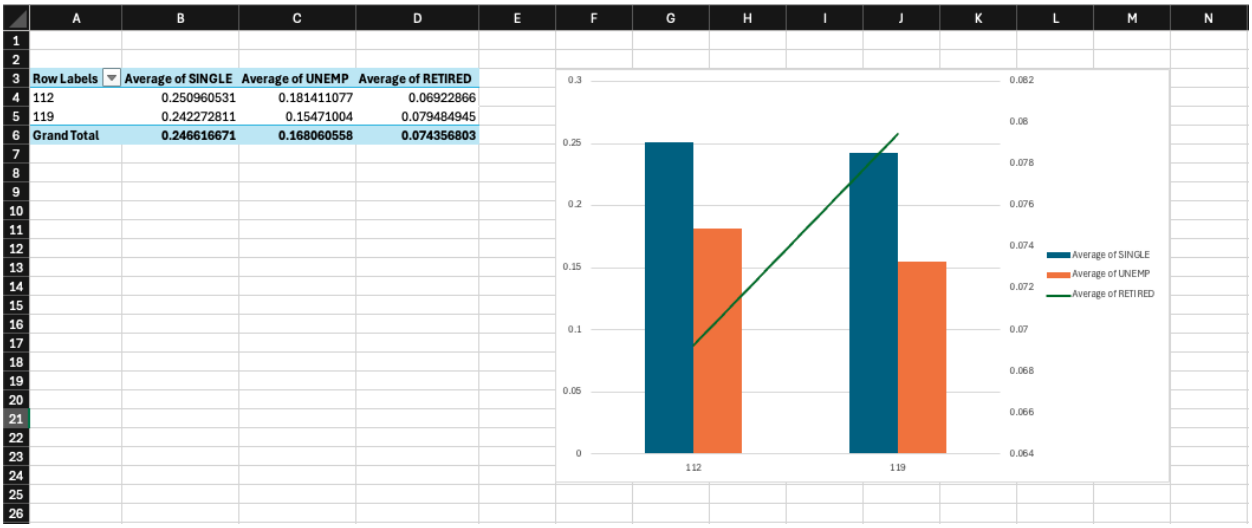
9. What were the highest sales contributions from single and retired individuals in the Buffalo Grove stores for BUDWEISER BEER N.R.B during the Thanksgiving week of 1993?

   This analysis helps **Dominick's Fine Foods** by revealing how **single** and **retired** shoppers contribute to sales during Thanksgiving week. Understanding demographic-driven sales for high-demand products like **BUDWEISER BEER N.R.B** enables DFF to optimize promotions and inventory for these customer segments. It allows for more targeted marketing and better stocking strategies, helping the retailer maximize sales and meet customer demand during key holiday periods.

   Support:

   To analyze the sales data, I used the demographic CSV file to filter the relevant week of sales, focusing on specific customer segments, namely single, unemployed, and retired individuals. I separated the necessary columns and rows to isolate the required data. Next, I worked with the UPC and Movement CSV files. From the UPC file, I extracted the product UPC codes and sizes. Then, using the Movement CSV file, I filtered the data for

stores located in Buffalo Grove, identifying the sales transactions for the target products during the specified week. After filtering the data, I created a pivot table to summarize the findings. Finally, I generated a pivot chart to visually depict the differences in sales during that particular week.

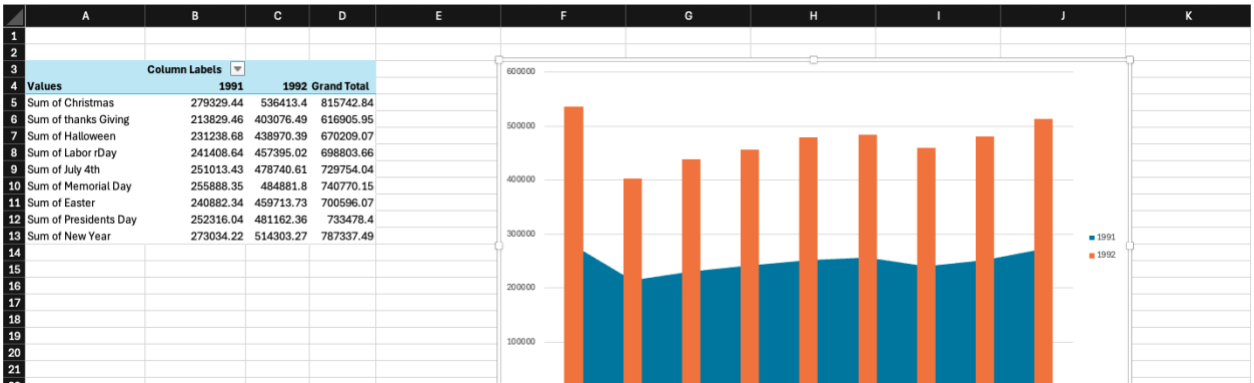


Row Labels	Average of SINGLE	Average of UNEMP	Average of RETIRED
112	0.250960531	0.181411077	0.06922866
119	0.242272811	0.15471004	0.079484945
Grand Total	**0.246616671**	**0.168060558**	**0.074356803**

10. Which holiday week in 1991 and 1992 saw the highest grocery sales in the low-tier Buffalo Grove stores? [3]

    This analysis is valuable for **Dominick's Fine Foods (DFF)** as it helps identify peak grocery sales periods in low-tier stores during key holidays. By pinpointing which holiday weeks in 1991 and 1992 had the highest sales, DFF can optimize future inventory management, staffing, and promotions for low-tier locations. This allows DFF to better align supply with demand, minimizing stockouts while maximizing revenue during high-traffic holiday periods.

    Support:



Values	Column Labels 1991	1992	Grand Total
Sum of Christmas	279329.44	536413.4	815742.84
Sum of thanks Giving	213829.46	403076.49	616905.95
Sum of Halloween	231238.68	438970.39	670209.07
Sum of Labor rDay	241408.64	457395.02	698803.66
Sum of July 4th	251013.43	478740.61	729754.04
Sum of Memorial Day	255888.35	484881.8	740770.15
Sum of Easter	240882.34	459713.73	700596.07
Sum of Presidents Day	252316.04	481162.36	733478.4
Sum of New Year	273034.22	514303.27	787337.49

To perform this analysis, I utilized the account CSV file alongside the DFF manual PDF. I began by collecting the grocery purchase data for the entire year of 1991, followed by repeating the same process for 1992. After gathering the data, I filtered the weekly grocery purchases for both years individually. Once the data was cleaned, I created a pivot table to summarize the findings. Using this pivot table, I then generated a pivot chart to visualize the results. The chart clearly shows that sales during holiday weeks in 1992 were higher compared to the same period in 1991, highlighting a notable increase in holiday-related sales.

Selected Business Questions

1. Which store has the highest store traffic for the last quarter of 1994?
2. Which holiday week in 1991 and 1992 saw the highest grocery sales in the low-tier Buffalo Grove stores?
3. How does the profit margin of toothpaste vary by brand?
4. What are the monthly average sales amounts for the BAKERY, DAIRY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores located in Naperville and Schaumburg during the year 1994, and how do these averages compare among the different departments within these locations?
5. What were the highest sales contributions from single and retired individuals in the Buffalo Grove stores for **BUDWEISER BEER N.R.B** during the Thanksgiving week of 1993?

Section 3: Independent Data Marts design using Kimball's approach

Overview of Kimball's Methodology

Kimball's dimensional modeling methodology is used largely in data warehouse designing, specifically for creating data marts to help us address specific business questions. This approach manages data in a way that is created for optimizing analytical queries using the star schema. In a star schema, there is a central fact table containing quantitive data which is surrounded by dimension tables that contain information related to the fact table.

Star schemas are used for analytics since they help users filter, examine, and slice data with minimal complexity, and complex joins are unnecessary.

Steps for creating Independent Data marts using Kimball's Methodology

1. **Selection of Business Process :** The first step is to choose a specific business process to make sure that the data model aligns with the queries that are applicable to the process.
2. **Decide the granularity of Fact Data:** Define the most atomic level of data in the fact table. For example, in a retail domain, the granularity can be at the line item level for each sale rather than the order level.
3. **Select Relevant Dimension for Fact table :** Select dimensions based on how the users interact and use data for the chosen process.
4. **Identify numeric facts for the Fact table:** Decide the numeric value for each fact table. This will be based on the business questions that have to be answered.

Why It's Important to Follow Kimball's Methodology to Create Independent Data Marts?

1. **Modular and Incremental Development**

   Kimball's methodology advocates creating data marts incrementally, with each one focusing on a specific business process (e.g., sales, finance, marketing). These data marts

are designed independently but use conformed dimensions (shared dimension tables across multiple data marts). This approach makes it easier to build the data warehouse in smaller, manageable pieces that deliver immediate business value while avoiding the complexity of trying to build an entire enterprise data warehouse in one go.

2. **Business-Centric Approach**

   By focusing on business processes and requirements from the start, Kimball's methodology ensures that the data warehouse is closely aligned with the organization's business needs. Each data mart is designed to answer specific business questions, making the data warehouse more valuable and user-friendly.

3. **Simplified Integration**

   One of the key aspects of Kimball's approach is the concept of conformed dimensions, which allows for easy integration of data marts into a larger enterprise data warehouse. Even though the data marts are built independently, they can be linked together because they share common dimensions (e.g., time, customer, product). This makes it possible to create a unified data warehouse over time without needing to redesign or rework the data marts.

4. **Reduced Complexity**

   Building data marts independently following Kimball's methodology reduces the initial complexity of building an enterprise-wide data warehouse. Each data mart is a self-contained, manageable project that focuses on a specific business process. Once several data marts are in place, they can be integrated to form the larger data warehouse. This approach prevents large-scale failures that could occur if you attempt to build everything at once.

5. **Scalability and Flexibility**

   Kimball's methodology is designed to be flexible. As business needs evolve, new data marts can be added to the warehouse with minimal disruption to the existing

infrastructure. The modular nature of the data marts, combined with the use of conformed dimensions, makes it easy to expand the data warehouse to accommodate new business processes or additional data sources.

6. **User Empowerment and Adoption**
   The business-driven nature of Kimball's methodology results in data marts that are highly relevant to end-users. When users can directly see the value of the data mart in answering their questions, they are more likely to adopt the system. Kimball's approach also emphasizes delivering solutions quickly, allowing users to benefit from data insights earlier, rather than waiting for an entire warehouse to be completed.

DW Logical Design

We are using Kimball's methodology to create independent data marts. Below are logical design steps in creating the data marts.

Requirement Analysis

The first step is to analyze the business questions that need to be answered. Below are the business questions that have been selected for this project.

1. Which store has the highest store traffic for the last quarter of 1994?
2. How does the profit margin of toothpaste vary by brand?
3. What were the highest sales contributions from single and retired individuals in the Buffalo Grove stores for BUDWEISER BEER N.R.B during the Thanksgiving week of 1993?
4. What are the monthly average sales amounts for the BAKERY, DAIRY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores located in Naperville and Schaumburg during the year 1994, and how do these averages compare among the different departments within these locations?
5. Which holiday week in 1991 and 1992 saw the highest grocery sales in the low-tier Buffalo Grove stores?

Identify Dimension and Fact Tables

The next step is to identify the dimension and fact tables necessary to address these questions

Dimension Tables

- Store_Dim
- Date_dim
- Product_Dimension
- Brand_Dimension
- Demography_Dim_BQ5
- Product_Dim_BQ5

Fact Tables

- Fact_BQ2
- Fact_Sales
- Store_Traffic_Fact
- Profit_Fact
- Customer_Demographics_Fact

Matrix Creation

	Store_Dim	**Date_Dim**	**Brand_Dimension**	**Product_Dimension**	**Demography_Dim_BQ5**	**Product_Dim_BQ5**
Store_Traffic_Fact	x	x				
Fact_BQ2	x	x				
Profit_Fact			x	x		
Fact_Sales	x	x				
Customer_Demographics_Fact		x			x	x

Conceptual Design

Fig: Conceptual Schema for Store Traffic Data Mart

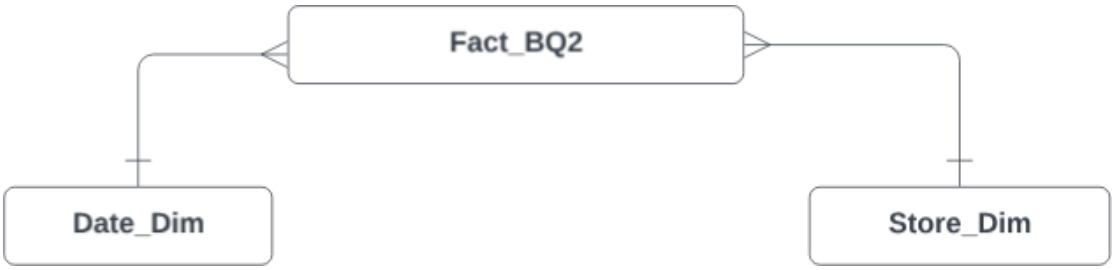


Fig: Conceptual Schema for Holiday Sales Data Mart

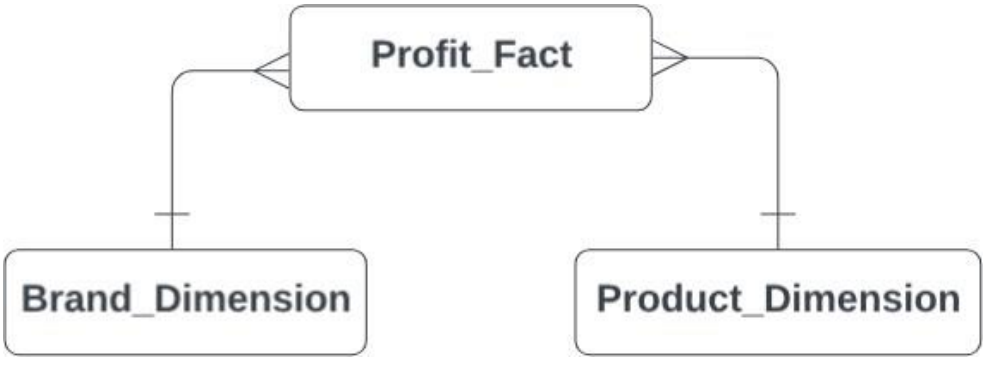


Fig: Conceptual Schema for Profit Margin Data Mart

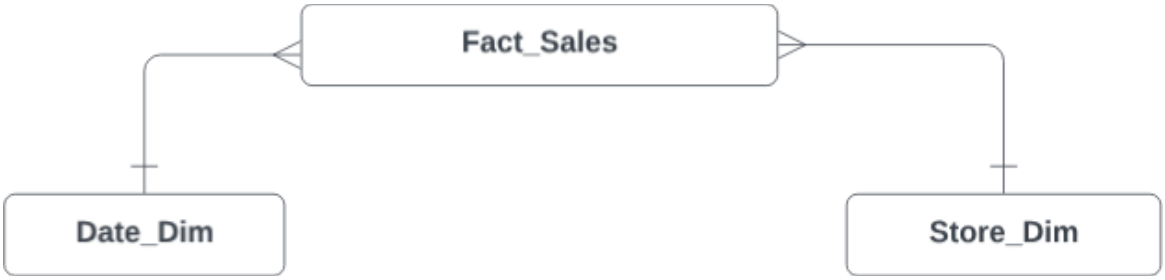


Fig: Conceptual Schema for Departmental Sales Data Mart

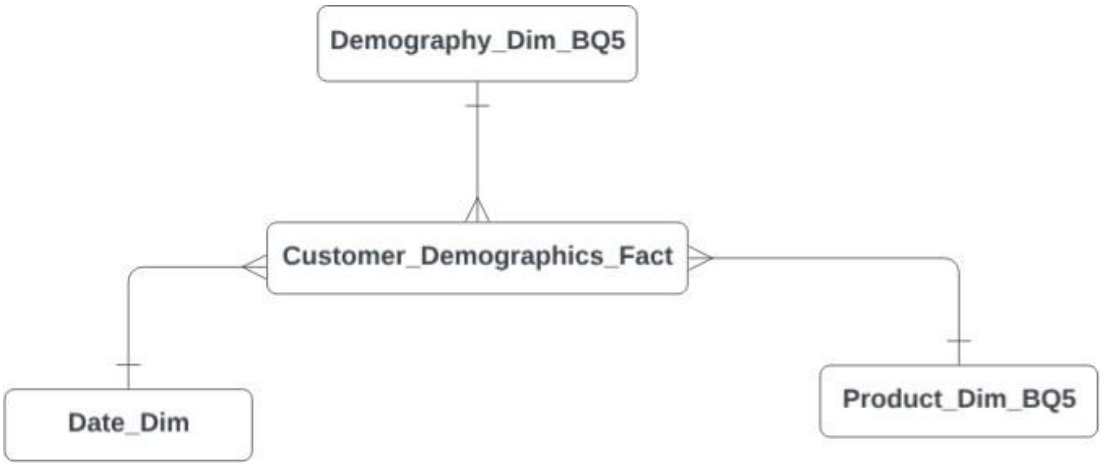


Fig: Conceptual Schema for Customer Demographics Data Mart

Dimension Tables

1. **Brand_Dimension**

   BRAND_ID: Unique Identifier of the Brand_Dimension table

   BRAND_NAME: Name of the product brand

Brand_Dimension	
BRAND_ID (PK)	INT
BRAND_NAME	VARCHAR

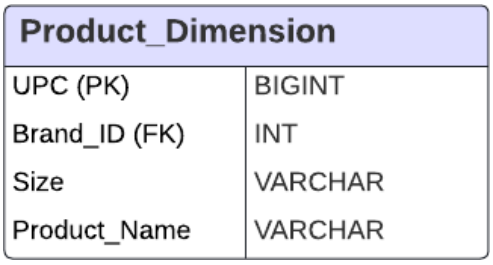
2. **Product_Dimension**

   UPC: Unique Identifier of the Product_dimension Table

   Brand_ID: Foreign Key from Brand_Dimension

   Size: Size of the product

   Product_Name: Name of the product

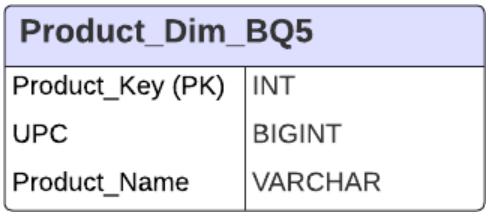


3. **Product_Dim_BQ5**

   Product_Key: Unique Identifier of the Product_Dim_BQ5 table

   UPC: Unique identifier of the product

   Product_Name: Product Name



4. **Demography_Dim_BQ5**

   Demography_Key: Unique Identifier of the Demography_Dim_BQ5

   Store: Unique Store ID

   Single: % of Single People

   Retired: % of Retired people

Demography_Dim_BQ5	
Demography_Key(PK)	INT
Store	INT
Single	FLOAT
Retired	FLOAT

5. **Store_Dim**

   Store_Key: Unique identifier for the Store_Dim table

   Store_ID: Store Code

Store_Dim	
Store_Key (PK)	INT
Store_ID	INT

6. **Date_dim**

   Date Key: Unique identifier for the Date_Dim table

   DATE: Date of the observation

   YEAR: Year of purchase

   MONTH: Month of purchase

   DAY: Day of purchase

   WEEK: Week Number of the purchase

   QUARTER: Quarter of the purchase

Date_Dim	
DateKey (PK)	INT
DATE	VARCHAR (64)
YEAR	INT
MONTH	INT
DAY	INT
WEEK	INT
QUARTER	INT

Fact Tables

1. **Fact_Sales**

   Store_Key: Unique identifier generated as surrogate key in Store_dim

   Date_Key: Unique identifier generated as surrogate key in Date_dim

   Bakery_Sales: Bakery sales in dollars

   Dairy_Sales: Dairy sales in dollars

   Pharmacy_Sales: Pharmacy sales in dollars

   Cosmetics_Sales: Cosmetics sales in dollars

   HABA_Sales: HABA sales in dollars

Fact_Sales	
Store_Key (FK)	INT
Date_Key (FK)	INT
Bakery_Sales	DECIMAL(10,2)
Dairy_Sales	DECIMAL(10,2)
Pharmacy_Sales	DECIMAL(10,2)
Cosmetic_Sales	DECIMAL(10,2)
HABA_Sales	DECIMAL(10,2)

2. **Profit_Fact**

   Profit_ID: Unique identifier for Profit_Fact Table

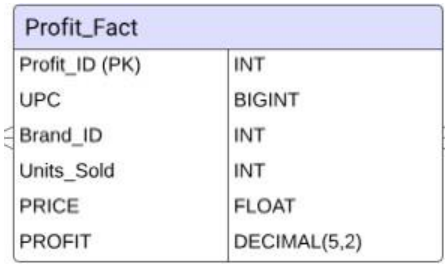
   UPC: Unique identifier generated as surrogate key in Product_Dimension

   Brand_ID: Unique identifier generated as surrogate key in Brand_ID

   Units_Sold: Number of units sold

   Price: Price of the product

   Profit: Gross margin

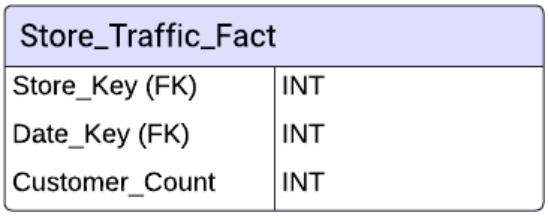


3. **Store_Traffic_Fact**

   Store_Key: Unique identifier generated as surrogate key in Store_dim

   Date_Key: Unique identifier generated as surrogate key in Date_dim

   Customer_Count: Number of customers



4. **Fact_BQ2**

   Store_Key: Unique identifier generated as surrogate key in Store_dim

   Date_Key: Unique identifier generated as surrogate key in Date_dim

   Total_Grocery_Sales: Total sales made from Grocery

   Week_Name: Day of the week

Fact_BQ2	
Store_Key (FK)	INT
Date_Key (FK)	INT
Total_Grocery_Sales	DECIMAL(10,2)
Week_Name	VARCHAR

5. **Customer_Demographics_Fact**

Fact_Key: Unique identifier for the customer_demographics_fact table

Product_Key: Unique identifier generated as surrogate key in Product_Dim_BQ5

Demography_Key: Unique identifier generated as surrogate key in Demography_Dim_BQ5

Date_Key: Unique identifier generated as surrogate key in Date_dim

Sales: Move * Price/Qty

Customer_Demographics_Fact	
Fact_Key (PK)	INT
Product_Key(FK)	INT
Demography_Key (FK)	INT
Date_Key (FK)	INT
Sales	DECIMAL(10,2)

Star Schema

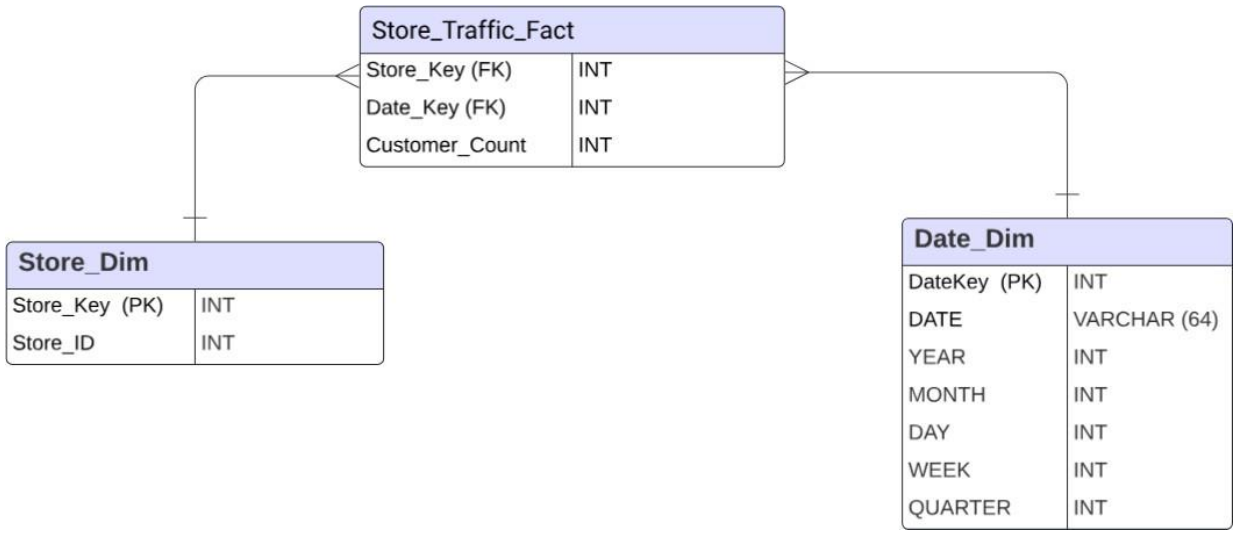


Fig: Star Schema Store Traffic Data Mart

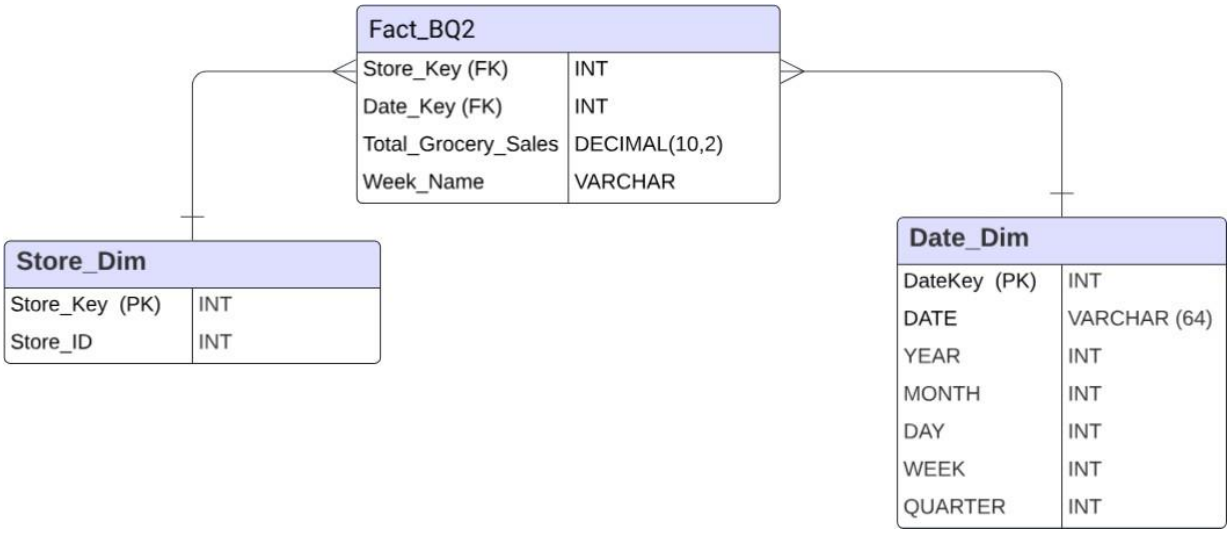


Fig: Star Schema Holiday Sales Data Mart

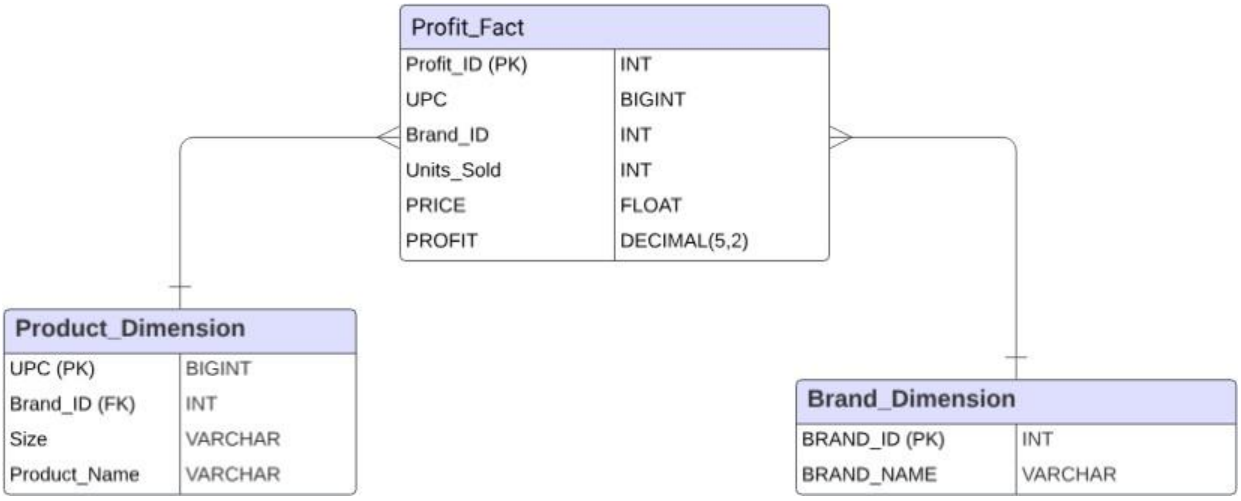


Fig: Star Schema Profit Margin Data Mart

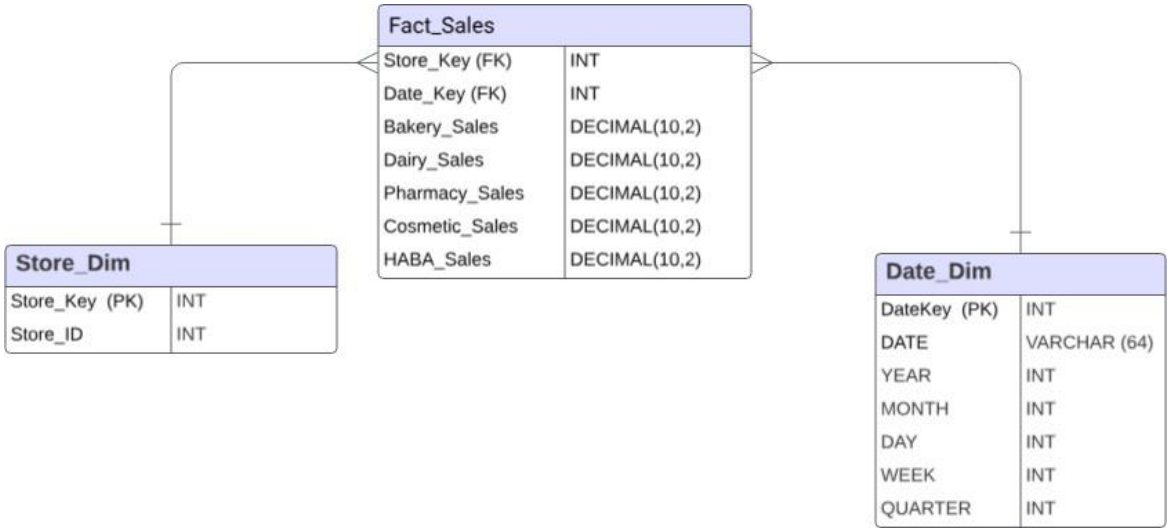


Fig: Star Schema Departmental Sales Data Mart

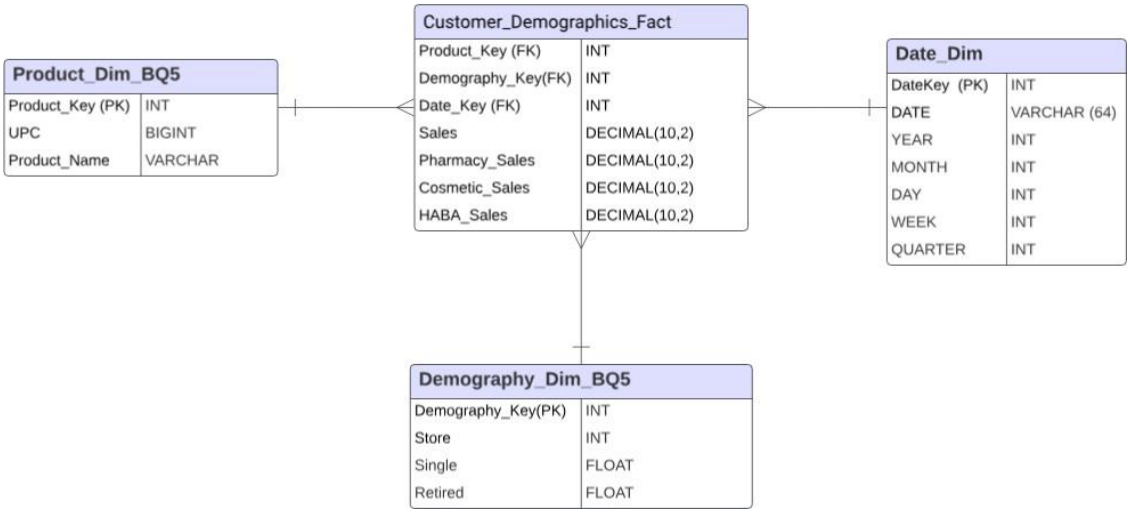


Fig: Star Schema Customer Demographics Data Mart

Section 4: Data Cleaning and Integration

Data Quality Issues in DFF Data set

When we evaluated data quality issues in the Dominick Finer Foods, we looked at the CCount, Stores and WeekTables.

Store Values Missing: The column "Store" in CCount had missing values in some reports. Without this information, it was difficult to analyse the data accurately.

Null Values: CCount had null values in the Store and Date columns, which could lead in inaccurate analysis.

Date column format: CCount stores DATE in varchar instead of standard Date format. This adds the need to extract the date from the varchar and store it for accurate analysis.

Junk Values in CCount : There were some records which had junk value like "aaa" in the Sales column which should be dollars. This junk values have to cleaned before analysis.

Data containing extra spaces: There are extra spaces either before or after some data in various columns. These extra spaces had to be removed before using the data.

Invalid Store Mapping: Stores were mapped based on the store_name instead of store_id which is more accurate. This can lead to incorrect mappings.

In conclusion, the DFF has various data quality issues like null values, missing data, incorrect data type, incorrect mapping etc that have to be resolved before using the data for analysis.

ETL Plan

1. **Identify Target Data**

   Below are the target data details

   a. Store_Dim

Target Column	Target Data Type
Store_Key	int
Store_ID	int

   b. Date_dim

Target Column	Target Data Type
Date_Key	int
DATE	DATE
YEAR	int
MONTH	int
DAY	int
WEEK	int
QUARTER	int

   c. Product_Dimension

Target Column	Target Data Type
UPC	BIGINT
Brand_ID	INT
Size	VARCHAR
Product_Name	VARCHAR

d. Brand_Dimension

Target Column	Target Data Type
Brand_ID	INT
Brand_Name	VARCHAR

e. Demography_Dim_BQ5

Target Column	Target Data Type
Demography_Key	INT
Store	INT
Single	FLOAT
Retired	FLOAT

f. Product_Dim_BQ5

Target Column	Target Data Type
Product_Key	INT
UPC	BIGINT
Product_Name	VARCHAR

g. Fact_BQ2

Target Column	Target Data Type
Store_Key	INT
Date_Key	INT
Total_Grocery_Sales	DECIMAL(10,2)
Week_Name	VARCHAR

h. Fact_Sales

Target Column	Target Data Type
Store_Key	INT
Date_Key	INT
Bakery_Sales	DECIMAL(10,2)
Diary_Sales	DECIMAL(10,2)
Cosmetic_Sales	DECIMAL(10,2)
HABA_Sales	DECIMAL(10,2)

i. Store_Traffic_Fact

Target Column	Target Data Type
Store_Key	INT
Date_Key	INT
Customer_Count	INT

j. Profit_Fact

Target Column	Target Data Type
Profit_ID	INT
UPC	INT
Brand_ID	INT
Units Sold	INT
Price	FLOAT
Profit	DECIMAL(10,2)

k. Customer_Demographics_Fact

Target Column	Target Data Type
Store_Key	INT
Date_Key	INT
Demography_Key	INT
Sales	DECIMAL(10,2)
Pharmacy_Sales	DECIMAL(10,2)
Cosmetic_Sales	DECIMAL(10,2)
HABA_Sales	DECIMAL(10,2)

2. Identify Data Sources

Below is the list of the files that were used for the project

Data Source	Source File Name	Staging Area Table Name	Data Mart Table Name
Customer Count Data	CCOUNT.csv	Cccount_stg, ccount_clean	Store_Traffic_Fact, Fact_BQ2
Demographics Data	DEMO.csv	BQ5_Demo_Cleaned	Demography_Dim_BQ5
Product_Data	UPCTPA.csv	UPCTPA_clean	Product_Dim, Product_Dim_BQ5
Weekly Product Data	WTPA.csv	WTPA_done	Profit_Fact
Beer Product Data	UPCBER.csv	UPCBER	Product_Dim
Weekly Beer Data	WBER.csv	DONE-WBER	Customer_Demographics_Fact

3. Data Mapping

Mapping Table 1 :Source to Staging

BQ 1					
Source data	Source data field	Mapping	Staging Table type	Staging Table name	Attribute
CCOUNT.csv	WEEK	Copy	Relation	ccount_clean	WEEK
CCOUNT.csv	STORE	Copy	Relation	ccount_clean	STORE
CCOUNT.csv	CUSTCOUN	Copy	Relation	BQ1_data	CUSTCOUN
BQ 2					
Source data	Source data field	Mapping	Staging Table type	Staging Table name	Attribute
CCOUNT.csv	GROCERY	Copy	Relation	BQ2_Agg	TOTAL_GROCERY_SALES
CCOUNT.csv	STORE	Copy	Relation	ccount_clean	STORE
		Derived Year from DATE	Relation	ccount_clean	YEAR
		Derived based on festival WEEK	Relation	BQ2_Agg	WEEK_NAME
BQ 3					
Source data	Source data field	Mapping	Staging Table type	Staging Table name	Attribute
WPTA.csv	PROFIT	Copy	Relation	WTPA_done	PROFIT
WPTA.csv	MOVE	Copy	Relation	WTPA_done	MOVE
WPTA.csv	PRICE	Copy	Relation	WTPA_done	PRICE

UPCTPA.csv	UPC	Copy	Relation	UPCTPA_clean	UPC
UPCTPA.csv	DESCRIP	Copy	Relation	UPCTPA_clean	DESCRIP
UPCTPA.csv	SIZE	Copy	Relation	UPCTPA_clean	SIZE
		Derived from DESCRIP	Relation	UPCTPA_clean	BRAND
BQ 4					
Source data	Source data field	Mapping	Staging Table type	Staging Table name	Attribute
CCOUNT.csv	DIARY	Copy	Relation	BQ4_data	DIARY
CCOUNT.csv	STORE	Copy	Relation	ccount_clean	STORE
CCOUNT.csv	DATE	Copy	Relation	ccount_clean	DATE
CCOUNT.csv	BAKERY	Copy	Relation	BQ4_data	BAKERY
CCOUNT.csv	PHARMACY	Copy	Relation	BQ4_data	PHARMACY
CCOUNT.csv	COSMETIC	Copy	Relation	BQ4_data	COSMETIC
CCOUNT.csv	HBA	Copy	Relation	BQ4_data	HBA
		Derived from DATE Column of CCOUNT.csv	Relation	ccount_clean	MONTH
BQ 5					
Source data	Source data field	Mapping	Staging Table type	Staging Table name	Attribute
CCOUNT.csv	WEEK	Copy	Relation	ccount_clean	WEEK
CCOUNT.csv	STORE	Copy	Relation	ccount_clean	STORE
DEMO.csv	SINGLE	Copy	Relation	BQ5_Demo_Cleaned	Single_perct

DEMO.csv	RETIRED	Copy	Relation	BQ5_Demo_Cleaned	Retired_perct
UPCBER.csv	UPC	Copy	Relation	BQ5_Cleaned	UPC
WBER.csv	MOVE	Copy	Relation	BQ5_Cleaned	MOVE
WBER.csv	PRICE	Copy	Relation	BQ5_Cleaned	PRICE
WBER.csv	QTY	Copy	Relation	BQ5_Cleaned	QTY
UPCBER.csv	DESCRIP	Copy	Relation	BQ5_Cleaned	DESCRIP
		Derived from (PRICE * MOVE) / QTY	Relation	BQ5_Cleaned	Sales
Deriving Date dimension columns					
Source data	Source data field	Mapping	Staging Table type	Staging Table name	Attribute
CCOUNT.csv	DATE	DATE	Relation	ccount_clean	DATE
		Derived From DATE	Relation	ccount_clean	DAY
		Derived From DATE	Relation	ccount_clean	MONTH
		Derived From DATE	Relation	ccount_clean	QUARTER
		Derived From DATE	Relation	ccount_clean	YEAR

Mapping Table 2: Staging to Data marts

Data Mart 1					
Source data in staging	Staging table data field	Mapping	Data Mart Table type	Table name	Attribute
ccount_clean	WEEK	Copy	Dimension Table	Date_Dim	WEEK
ccount_clean	STORE	Copy	Dimension Table	Store_Dim	Store_ID
BQ1_data	CUSTCOUN	Copy	Fact Table	Store_Traffic _Fact	Customer_Co unt
ccount_clean	DATE	Copy	Dimension Table	Date_Dim	DATE
ccount_clean	MONTH	Copy	Dimension Table	Date_Dim	MONTH
ccount_clean	DAY	Copy	Dimension Table	Date_Dim	DAY
ccount_clean	YEAR	Copy	Dimension Table	Date_Dim	YEAR
ccount_clean	QUARTER	Copy	Dimension Table	Date_Dim	QUARTER
		Surrogate Key of Date Dimension Table.	Dimension Table	Date_Dim	Date_Key
		Surrogate Key of Store Dimension Table.	Dimension Table	Store_Dim	Store_Key
		Foreign Key of Fact table referencing Store_Key of Store_Dim	Fact Table	Store_Traffic _Fact	Store_Key

		dimension table			
		Foreign Key of Fact table referencing Date_Key of Date_Dim dimension table	Fact Table	Store_Traffic_Fact	Date_Key
Data Mart 2					
Source data in staging	Staging table data field	Mapping	Data Mart Table type	Table name	Attribute
BQ2_Agg	TOTAL_GROCERY_SALES	Copy	Fact Table	Fact_BQ2	Total_Grocery_Sales
ccount_clean	STORE	Copy	Dimension Table	Store_Dim	Store_ID
ccount_clean	YEAR	Copy	Dimension Table	Date_Dim	YEAR
BQ2_Agg	WEEK_NAME	Copy	Dimension Table	Fact_BQ2	Week_Name
ccount_clean	DATE	Copy	Dimension Table	Date_Dim	DATE
ccount_clean	MONTH	Copy	Dimension Table	Date_Dim	MONTH
ccount_clean	DAY	Copy	Dimension Table	Date_Dim	DAY
ccount_clean	WEEK	Copy	Dimension Table	Date_Dim	WEEK
ccount_clean	QUARTER	Copy	Dimension Table	Date_Dim	QUARTER
		Surrogate Key of Date Dimension Table.	Dimension Table	Date_Dim	Date_Key

		Surrogate Key of Store Dimension Table.	Dimension Table	Store_Dim	Store_Key
		Foreign Key of Fact table referencing Store_Key of Store_Dim dimension table	Fact Table	Fact_BQ2	Store_Key
		Foreign Key of Fact table referencing Date_Key of Date_Dim dimension table	Fact Table	Fact_BQ2	Date_Key
Data Mart 3					
Source data in staging	Staging table data field	Mapping	Data Mart Table type	Table name	Attribute
UPCTPA_clean	BRAND	Copy	Dimension Table	Brand_Dimension	Brand_Name
UPCTPA_clean	UPC	Copy (Primary Key)	Dimension Table	Product_Dimension	UPC
UPCTPA_clean	DESCRIP	Copy	Dimension Table	Product_Dimension	Product_Name
UPCTPA_clean	SIZE	Copy	Dimension Table	Product_Dimension	Size
WTPA_done	MOVE	Copy	Fact Table	Profit_Fact	Units_Sold
WTPA_done	PRICE	Copy	Fact Table	Profit_Fact	PRICE
WTPA_done	PROFIT	Copy	Fact Table	Profit_Fact	PROFIT

		Surrogate key of the Brand Dimension Table	Dimension Table	Brand_Dimension	Brand_ID
		Foreign Key referencing UPC of the Product_Dimension Table	Fact Table	Profit_Fact	UPC
		Foreign Key referencing Brand_ID of the Brand Dimension Table	Fact Table	Profit_Fact	Brand_ID
		Surrogate key of Fact table. [Auto increment]	Fact Table	Profit_Fact	Profit_ID
Data Mart 4					
Source data in staging	Staging table data field	Mapping	Data Mart Table type	Table name	Attribute
BQ4_data	DIARY	Copy	Fact Table	Fact_Sales	Diary_Sales
BQ4_data	BAKERY	Copy	Fact Table	Fact_Sales	Bakery_Sales
BQ4_data	PHARMACY	Copy	Fact Table	Fact_Sales	Pharmacy_Sales
BQ4_data	COSMETIC	Copy	Fact Table	Fact_Sales	Cosmetic_Sales
BQ4_data	HABA	Copy	Fact Table	Fact_Sales	HBA_Sales
ccount_clean	MONTH	Copy	Dimension Table	Date_Dim	MONTH
ccount_clean	STORE	Copy	Dimension Table	Store_Dim	Store_ID

ccount_clean	DATE	Copy	Dimension Table	Date_Dim	DATE
ccount_clean	DAY	Copy	Dimension Table	Date_Dim	DAY
ccount_clean	WEEK	Copy	Dimension Table	Date_Dim	WEEK
ccount_clean	QUARTER	Copy	Dimension Table	Date_Dim	QUARTER
ccount_clean	YEAR	Copy	Dimension Table	Date_Dim	YEAR
		Surrogate Key of Date Dimension Table.	Dimension Table	Date_Dim	Date_Key
		Surrogate Key of Store Dimension Table.	Dimension Table	Store_Dim	Store_Key
		Foreign Key of Fact table referencing Store_Key of Store_Dim dimension table	Fact Table	Fact_Sales	Store_Key
		Foreign Key of Fact table referencing Date_Key of Date_Dim dimension table	Fact Table	Fact_Sales	Date_Key
Data Mart 5					
Source data in staging	Staging table data field	Mapping	Data Mart Table type	Table name	Attribute

ccount_clean	WEEK	Copy	Dimension Table	Date_Dim	WEEK
ccount_clean	STORE	Copy	Dimension Table	Demography_ Dim_BQ5	STORE
BQ5_Demo_Cleaned	Single_perct	Copy	Dimension Table	Demography_ Dim_BQ5	Single
BQ5_Demo_Cleaned	Retired_perct	Copy	Dimension Table	Demography_ Dim_BQ5	Retired
BQ5_Cleaned	UPC	Copy	Dimension Table	Product_Dim _BQ5	UPC
BQ5_Cleaned	DESCRIP	Copy	Dimension Table	Product_Dim _BQ5	Product_Name
BQ5_Cleaned	Sales	Copy	Fact Table	Customer_Demographics_Fact	Sales
ccount_clean	YEAR	Copy	Dimension Table	Date_Dim	YEAR
ccount_clean	DATE	Copy	Dimension Table	Date_Dim	DATE
ccount_clean	MONTH	Copy	Dimension Table	Date_Dim	MONTH
ccount_clean	DAY	Copy	Dimension Table	Date_Dim	DAY
ccount_clean	QUARTER	Copy	Dimension Table	Date_Dim	QUARTER
		Surrogate key of the Date Dimension Table	Dimension Table	Date_Dim	Date_Key
		Surrogate key of the Demographic Dimension	Dimension Table	Demography_ Dim_BQ5	Demographic_Key

		Table			
		Surrogate key of the Product Dimension Table	Dimension Table	Product_Dim	Product_Key
		Surrogate key of the Fact table	Fact Table	Customer_Demographics_Fact	Fact_Key
		Foreign Key referencing Product_Key of the Product Dimension Table	Fact Table	Customer_Demographics_Fact	Product_Key
		Foreign Key referencing Demographic_Key of the Demography Dimension Table	Fact Table	Customer_Demographics_Fact	Demographic_Key
		Foreign Key referencing Date_Key of the Date Dimension Table	Fact Table	Customer_Demographics_Fact	Date_Key

4. Data Extraction Rules

Below are the steps for the data extraction that we executed for data extraction.

4.1 Extraction steps for CCOUNT

1. Created a new package in SSIS.

2. Using the SSIS Wizard, we have loaded into the cccount_stg.
3. Added components of the OLE DB Source (cccount_stg) and OLE DB Destination (ccount_clean).
4. Mapped columns and loaded data into the ccount_clean.
5. Performed necessary cleaning in ccount_clean.

4.2 Extraction steps for WBER

1. Created a new package in SSIS.
2. Using the SSIS Wizard, we have loaded into the Done-WBER.
3. Added components of the OLE DB Source (Done-WBER_csv) and OLE DB Destination (Done-WBER).
4. Mapped columns and loaded data into the Done-WBER.
5. Performed necessary cleaning in Done-WBER.

4.3 Extraction steps for UPCBER

1. Created a new package in SSIS.
2. Using the SSIS Wizard, we have loaded into the UPCBER.
3. Added components of the OLE DB Source (UPCBER.csv) and OLE DB Destination (UPCBER).
4. Mapped columns and loaded data into the UPCBER.
5. Performed necessary cleaning in UPCBER

4.4 Extraction steps for DEMO

1. Created a new package in SSIS.
2. Using the SSIS Wizard, we have loaded into the DEMO
3. Added a data flow task. For that we have added OLE DB Source (DEMO.csv) and OLE DB Destination (DEMO)
4. Mapped columns and loaded data into the DEMO table in staging
5. Performed necessary cleaning in DEMO

4.5 Extraction steps for UPCTPA

1. Created a new package in SSIS.
2. Using the SSIS Wizard, we have loaded data into the UPCTPA

3. Added a data flow task. For that we have added OLE DB Source (UPCTPA.csv) and OLE DB Destination (UPCTPA)
4. Mapped columns and loaded data into the DEMO table in staging
5. Performed necessary cleaning in DEMO

4.6 Extraction steps for WTPA

1. Created a new package in SSIS.
2. Using the SSIS Wizard, we have loaded data into the WTPA
3. Added a data flow task. For that we have added OLE DB Source (WTPA.csv) and OLE DB Destination (WTPA_done)
4. Mapped columns and loaded data into the WTPA_done table in staging
5. Performed necessary cleaning in WTPA_done

5. Data Transformation and Cleansing Rules

Transforming source data and loaded into a table that is usable is the next step after data extraction. This is important for the data quality and to maintain consistency across all the datasets before we load them into our destination database. Here we altered the columns to appropriate data types, extracted columns that are needed, generating new columns. For example, the DATE field in the CCOUNT was first converted to DATE format and then columns for day, month, year, quarter were added and populated. Also surrogate keys were added that can be used as primary keys and can be auto incremented. In conclusion, this serves as a foundation for efficiency and consistency and improves the quality of data and level of insights we can obtain from this data.

For all files, we have followed a standard format:

1. We first checked column names and formatted them correctly. Some files - CCOUNT, DEMO, contain “” for columns. We removed those “ and formatted them correctly.
2. We deleted the unnecessary columns in the staging table after the extraction of files. For instance in CCOUNT file - we removed all coupon related columns and other unnecessary columns based on our requirements. Similarly in DEMO File, we removed all columns except store, single, retired columns.

3. For each column we performed necessary cleaning such as - Removing records related to columns which contain value as - NULL, 0, '-0', '-', Blank (''), and numbers - [1-9].
4. After removing nulls and unnecessary data, We converted the column data type of each table as required. Initially DATE column of CCOUNT is in VARCHAR, but we converted it to DATE format and formatted as required 'YYYY-MM-DD'. Similarly we converted sales of departments (BAKERY, PHARMACY etc) into Decimal data type.
5. After conversion of data types, we created columns which are useful for all data marts - such as MONT, YEAR, QUARTER and derived data from DATE Column and loaded into these columns.
6. We have a main cleaned table for each file mentioned in the extraction step. We have used these main tables and created sub tables (Temp tables) for each question separately. For instance we created a BQ2_data temp table and loaded columns and data related to requirements.
7. After loading data into temp tables, we renamed a few column names as required.
8. We performed necessary transformations in temp tables by creating new columns and populating necessary data. For Instance, we created a column called WEEK_NAME which is of VARCHAR data type and updated it based on conditions. This column indicates whether the respective week of record is part of festival week or regular week.

6. Plan for Aggregating Tables

The approach to store data at the finest level of granularity gives the flexibility to users to manipulate and transform data as required to get specific insights on the data for reporting. One disadvantage of the approach is that this can impact the processing time and performance. Therefore it is important to analyze the data and define the required data granularity.Developing a data aggregation plan based on the business needs is essential this can impact the efficiency and performance of the dta warehouse.

For our project and business questions we require data based on week level, product level and data level. To do these we have executed lookup operations between our fact and dimension tables.

For Instance, We performed aggregation for BQ_2_data using YEAR and WEEK_Name columns and loaded aggregated data into BQ_2_Agg.

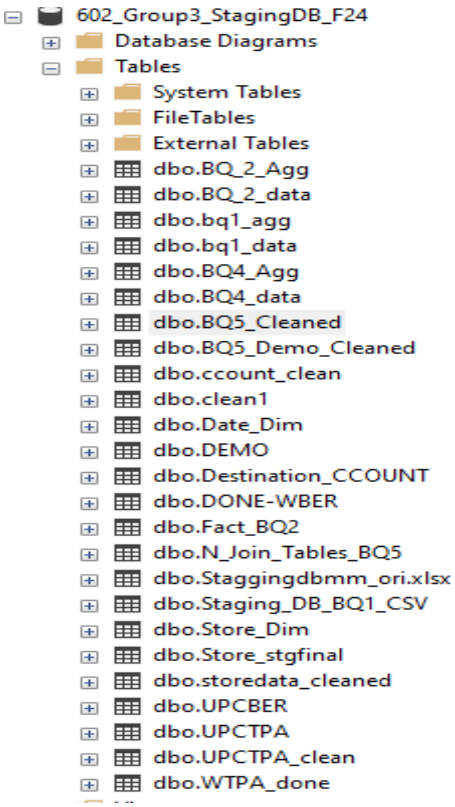
7. Organization of Data Staging Area

Organizing the Data staging area is very important as this serves as the connecting step between the source and the data warehouse. Proper organization of the staging area is pivotal for the management of the source data. In this data warehouse, we execute the data cleaning, transformation of the source data in the staging area before loading it in the destination of our data warehouse.

The staging area used was [602_Group3_StagingDB_F24] and below is the mapping of tables

Tables Created in Staging Database	**Source of the data**	**Operations performed**
[cccount_stg], [ccount_clean]	CCOUNT.csv	Cccount_stg is the copy of CCOUNT and cleaning was performed on this to then load the final cleaned data in ccount_clean
[DEMO], [BQ5_Demo_Cleaned]	DEMO.csv	DEMO is the copy of the DEMO.csv file. Performed cleaning on this and moved to BQ5_Demo_Cleaned.
[UPCTPA], [UPCTPA_clean]	UPCTPA.csv	The UPCTPA table contains raw data from the file. Cleaned data from the UPCTPA.csv

[WTPA_done]	WTPA.csv	Cleaned data from the WTPA.csv
[UPCBER]	UPCBER.csv	Cleaned data from the UPCBER.csv
[DONE-WBER]	WBER.csv	Cleaned data from the WBER.csv



8. Data Extraction and Loading Procedures

Here we focus on the extraction and loading of data from the source data into staging area and finally into the warehouse. SSIS is used for extraction and loading of the data. Special focus is given to correctly define the connection and mapping between the source and destination parameters. As instructed SSIS is used to load the flat file systems and move them to the SQL Server destination using the import export wizard. Another

important aspect to be considered is to make sure that data is not lost due to incorrect data type mapping.

In data loading, we execute the movement of data from the staging area to the data warehouse. For this we used data flows between OLE DB Source and OLE DB Destination using SSIS.

Steps:

1. We initially extracted files CCOUNT, DEMO etc into the Staging area. Performed necessary cleaning as mentioned in above steps.
2. After cleaning the data, we created temp tables in staging based on the requirements related to the business question.
3. We have used these temp tables to perform transformations needed and loaded data from temp tables to data marts in the presentation server.
4. We first loaded data into the dimension tables and we have used those dimensional tables to load data into the Fact table of the data marts.

9. ETL for Dimension Tables

Here we execute the Extract, Transform and Load processes for the dimension tables. Dimension table contain descriptive data and they provide context to the data in the fact table. Designing and executing ETL for dimension tables is crucial for the accuracy and the data analysis from the fact tables.

In this project we have dimension tables : date_dim, Store_Dim, Product_Dim, Brand, Demography_Dim. We have created separate SSIS packages for each dimension table and executed simple transformations for the data loading from cccount_stg table, demo.csv,UPCTPA_clean, WTPA_done, UPCBER and DONE-WBER.

10. ETL for Fact Tables

Here we execute the ETL processes for the Fact table which are used as the core for analytical queries and quantitative data. This data warehouse has five fact tables(Fact_BQ2, Fact_Sales, Store_Traffic_Fact, Customer_Demographic_Fact, Profit_Fact). The process used to load this fact table consists of using data from cccount_stg table, demo.csv,UPCTPA_clean, WTPA_done, UPCBER and DONE-WBER and extracting the attributes needed and executing lookups using the dimension tables.

Implementation of ETL Plan

Data Extraction

We have imported .csv files into a staging database that we created for ETL.

Database: [602_Group3_StagingDB_F24]

Tables:

Tables Created in Staging Database	Source of the data
[cccount_stg], [ccount_clean]	CCOUNT.csv
[BQ5_Demo_Cleaned]	DEMO.csv
[UPCTPA_clean]	UPCTPA.csv
[WTPA_done]	WTPA.csv
[UPCBER]	UPCBER.csv
[DONE-WBER]	WBER.csv

Data Extraction Steps:

Below are the steps taken to load the data into staging from source .csv files

CCOUNT: source to staging

- Created a new package in SSIS.
- Using the SSIS Wizard, we have loaded into the cccount_stg.
- Added components of the OLE DB Source (cccount_stg) and OLE DB Destination (ccount_clean).
- Mapped columns and loaded data into the ccount_clean.
- Performed necessary cleaning in ccount_clean.

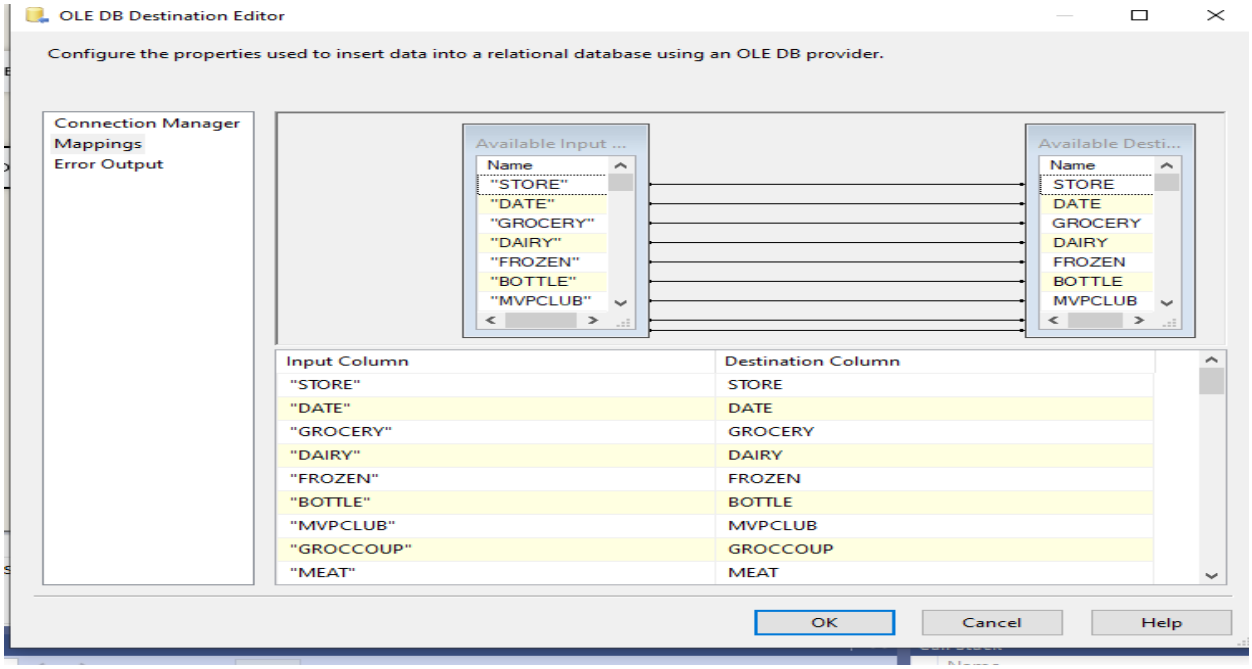


Fig: Mapping columns [cccount_stg to ccount_clean]

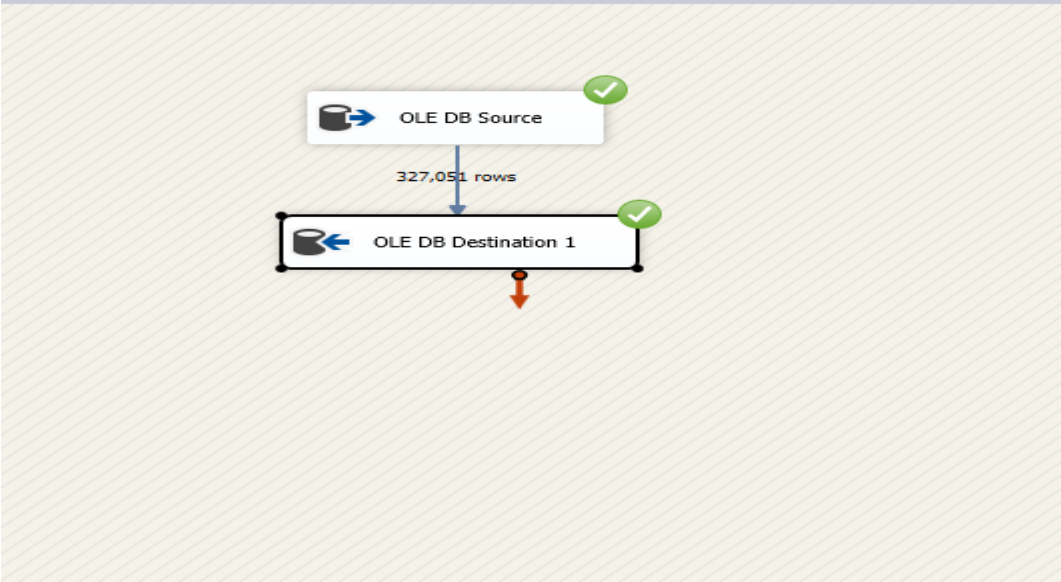


Fig: Extraction and loading into staging table

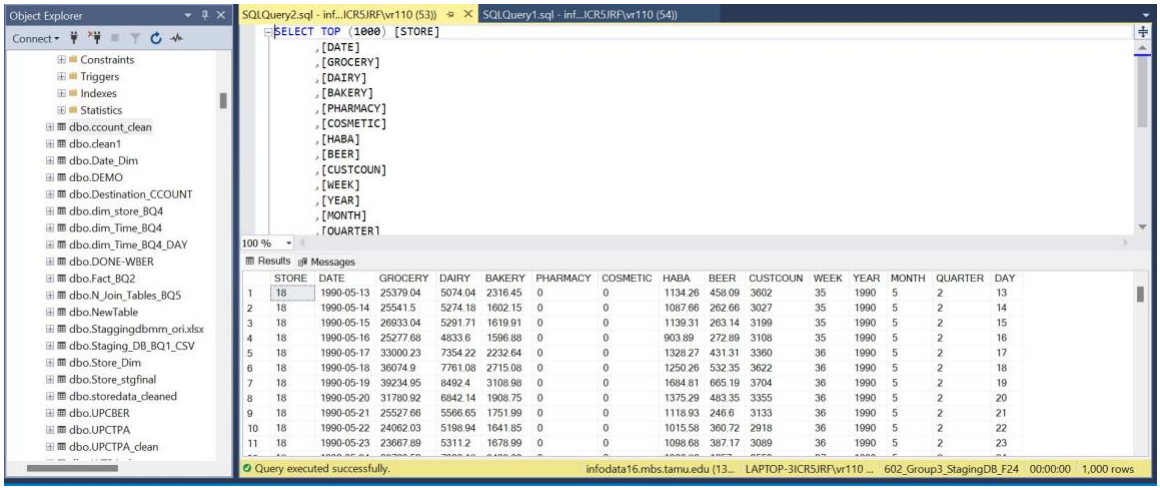


Fig: Validating the data in SSMS for ccount_clean

WBER: source to staging

- Created a new package in SSIS.
- Using the SSIS Wizard, we have loaded into the Done-WBER.
- Added components of the OLE DB Source (Done-WBER_csv) and OLE DB Destination (Done-WBER).
- Mapped columns and loaded data into the Done-WBER.
- Performed necessary cleaning in Done-WBER.

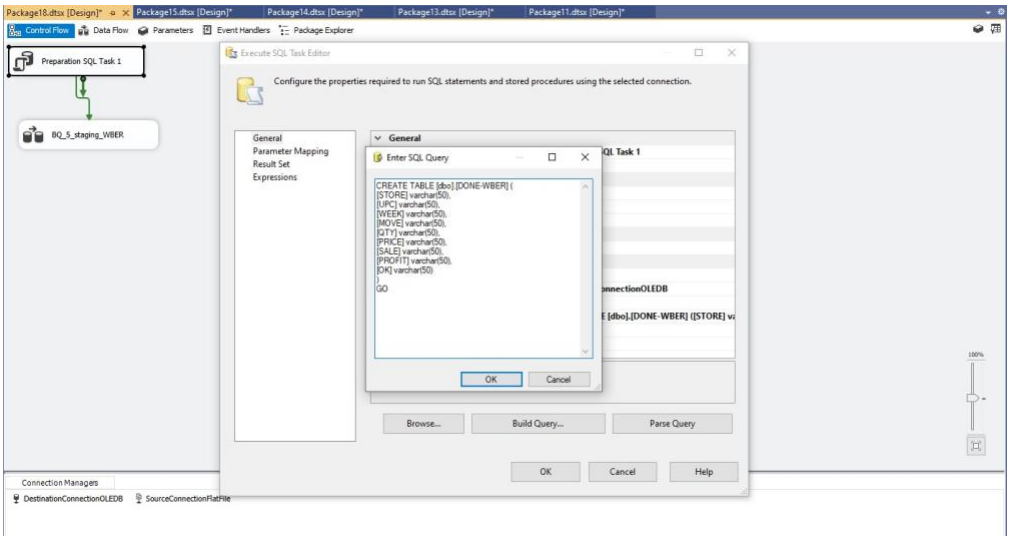


Fig Creating Table [DONE-WBER]

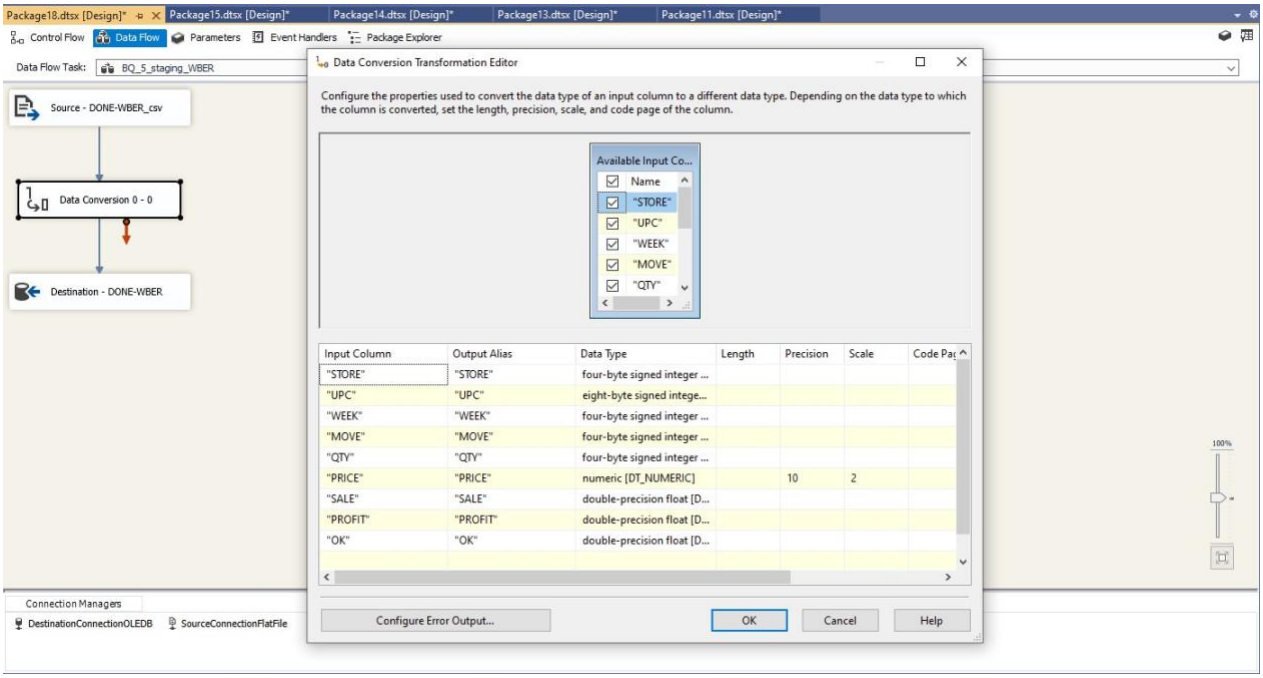


Fig Data Type Conversion

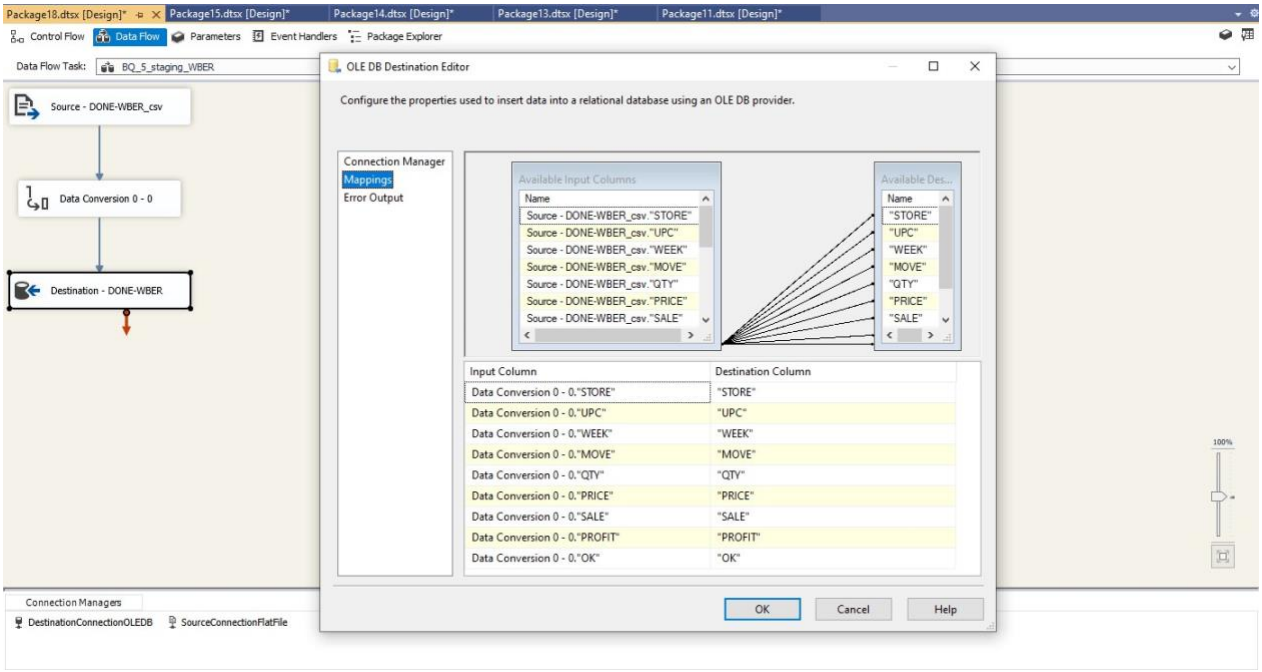


Fig Mapping Columns in Destination Table

UPCBER: source to staging

Steps for extracting only relevant data from source to staging:

- Created a new package in SSIS.
- Using the SSIS Wizard, we have loaded into the UPCBER.
- Added components of the OLE DB Source (UPCBER.csv) and OLE DB Destination (UPCBER).
- Mapped columns and loaded data into the UPCBER.
- Performed necessary cleaning in UPCBER

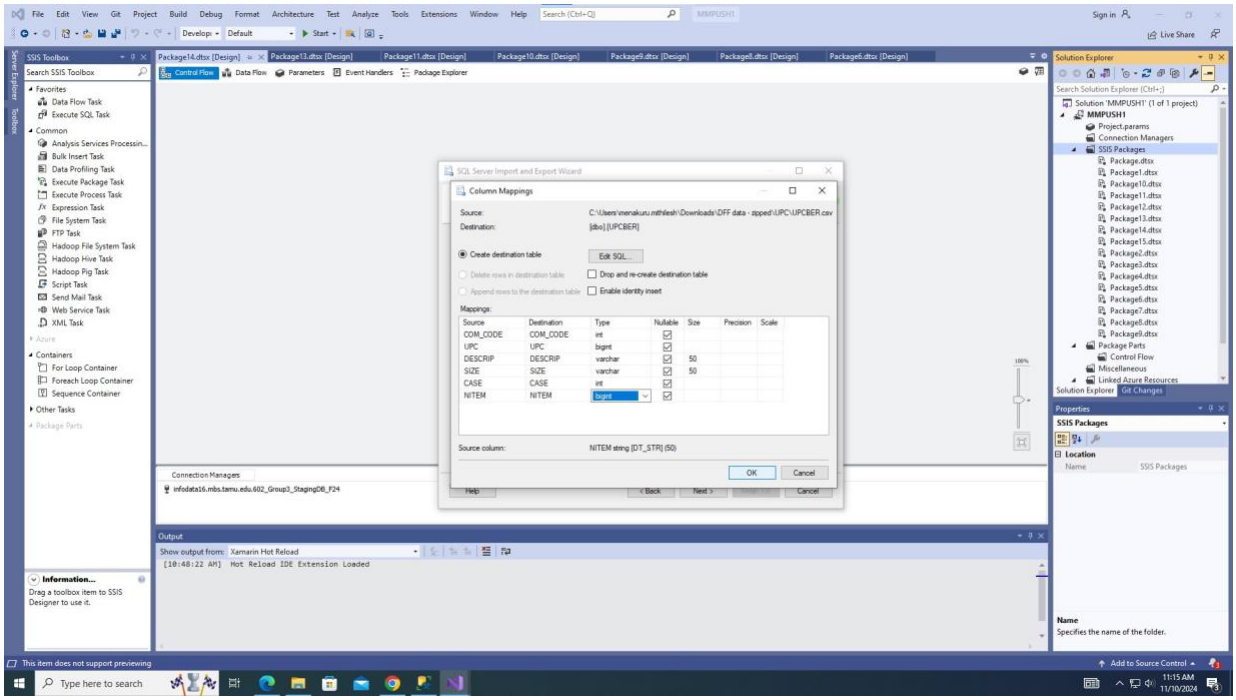


Fig: Loading csv file into to staging

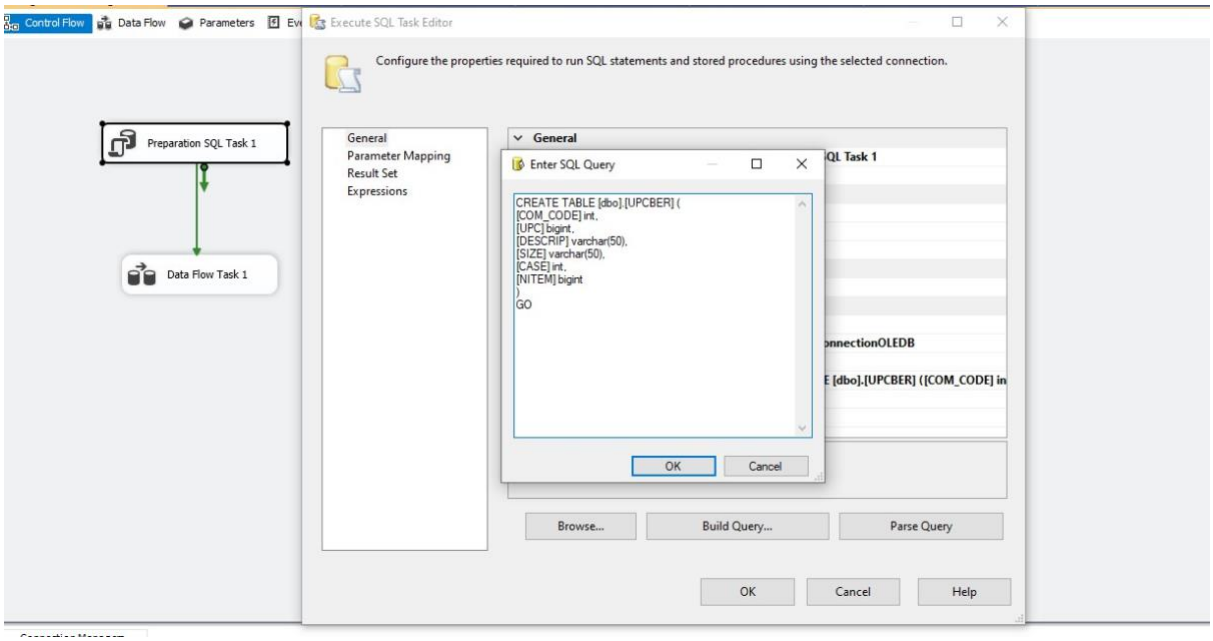


Fig: Creation of UPCBER

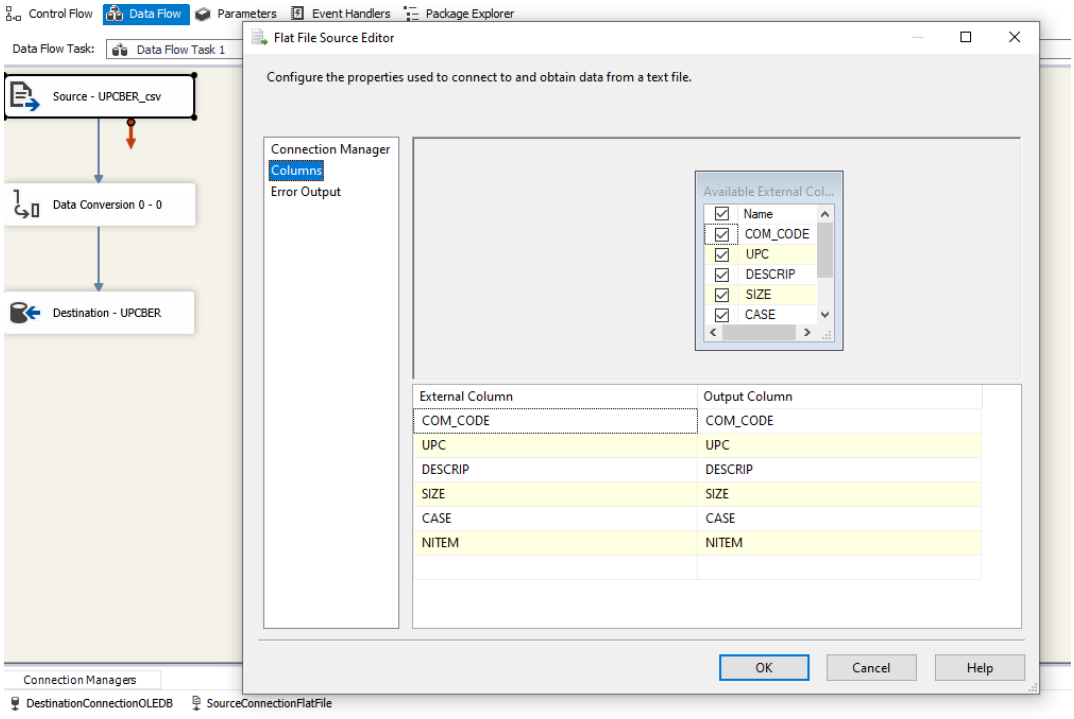


Fig: Columns of UPCBER from Source

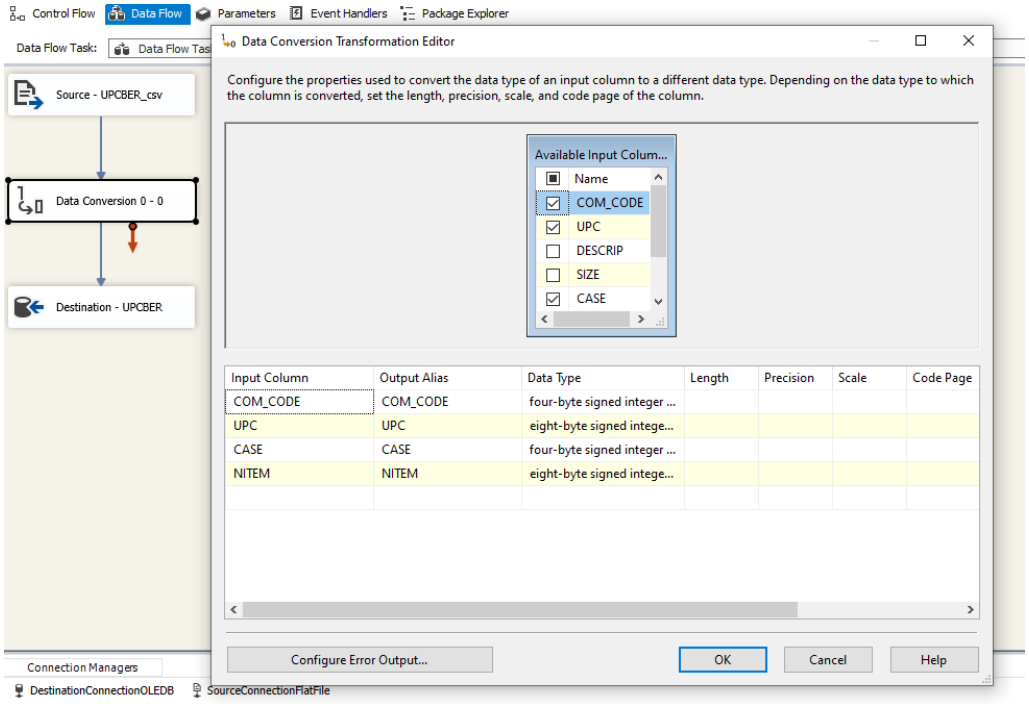


Fig: Data type Conversion

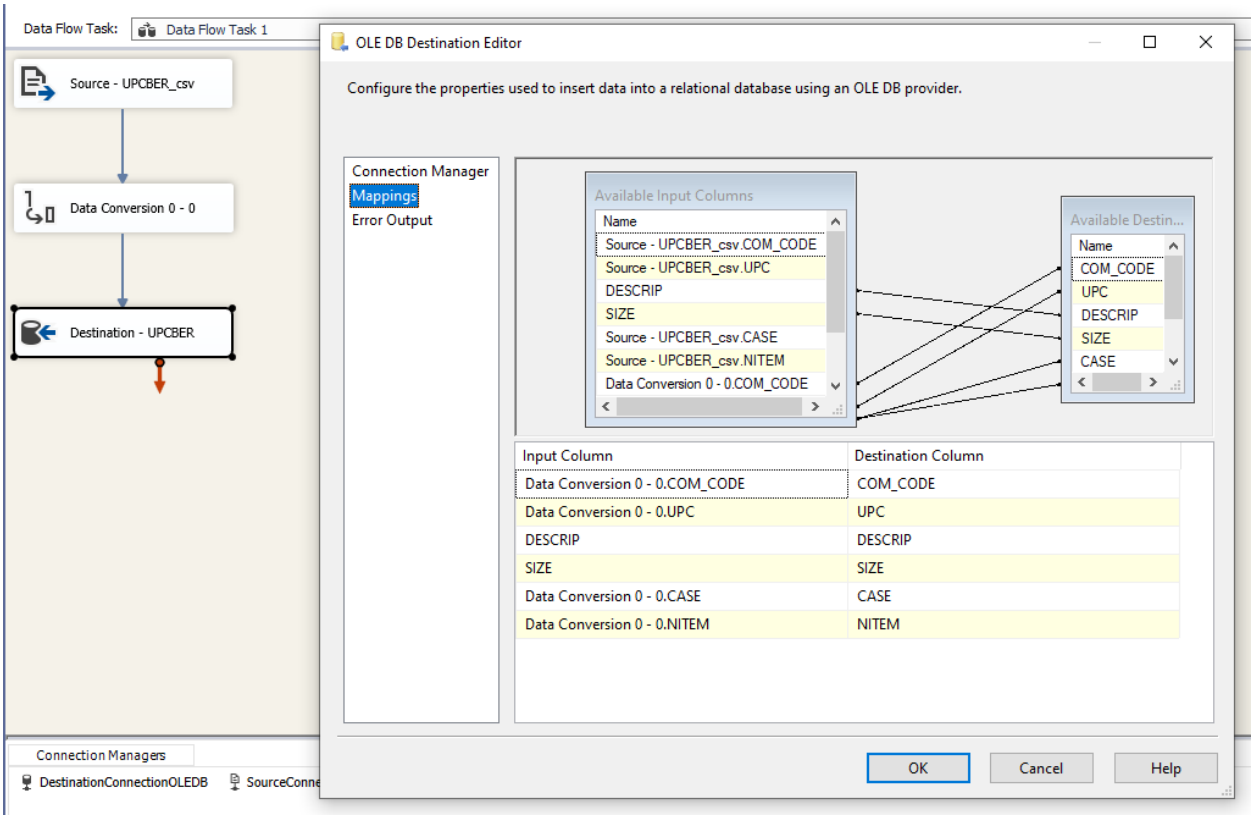
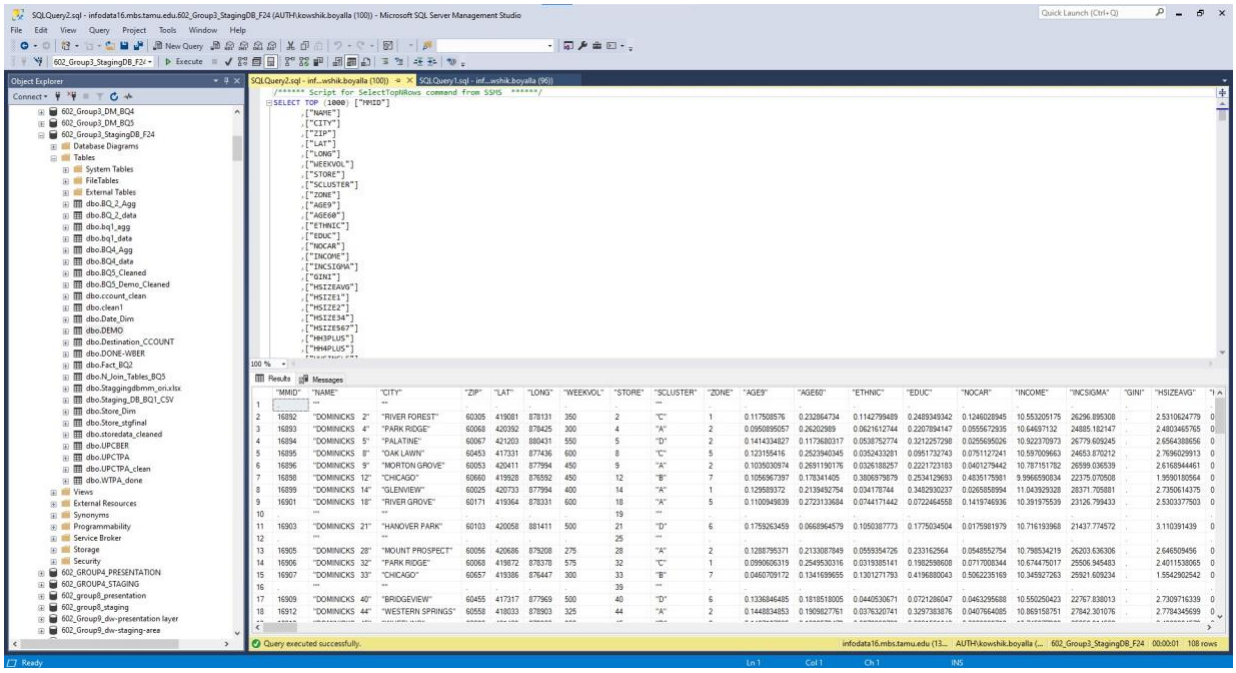


Fig: Destination Mapping

DEMO: source to staging

Steps for extracting only relevant data from source to staging:

- Created a new package in SSIS.
- Using the SSIS Wizard, we have loaded into the DEMO file.
- Added components of the OLE DB Source (DEMO.csv) and OLE DB Destination (DEMO).
- Mapped columns and loaded data into the DEMO.



- Performed necessary cleaning in DEMO.

WTPA.csv: source to staging

- Create a new project in SSMS
- Right click on **SSIS Packages**

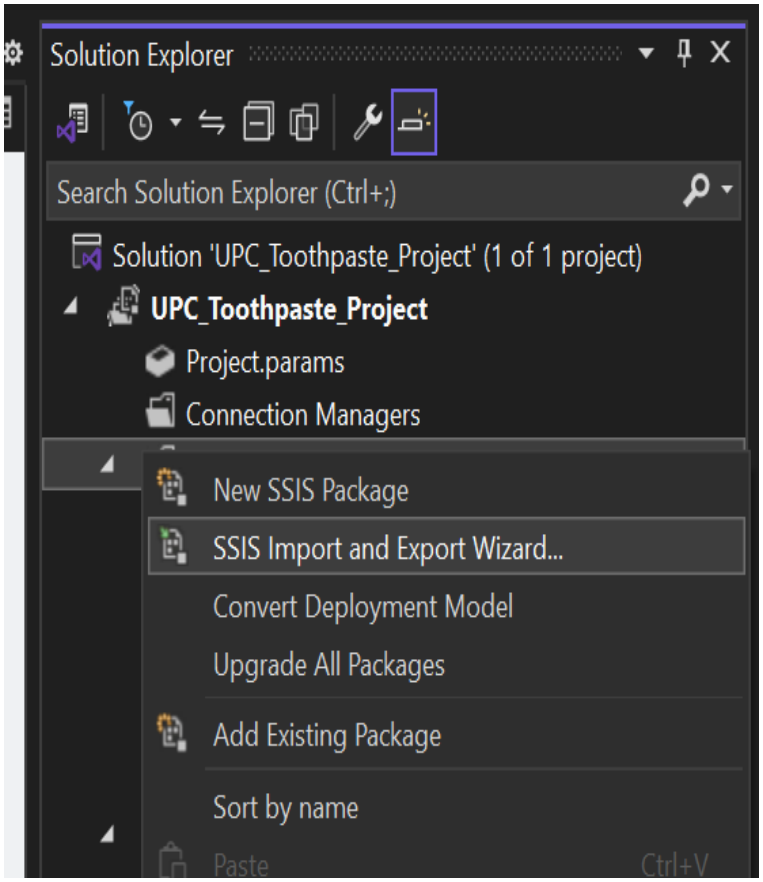


Fig: Select **SSIS Import and Export Wizard**

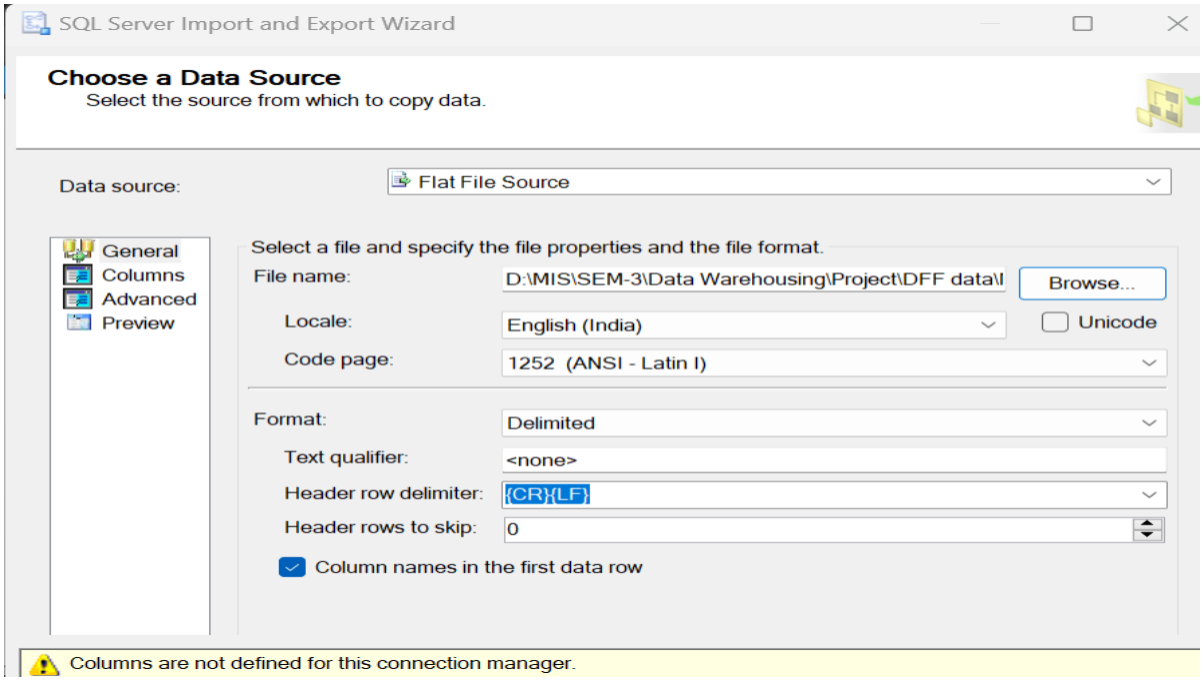


Fig: Select the data source and the WTPA.csv file

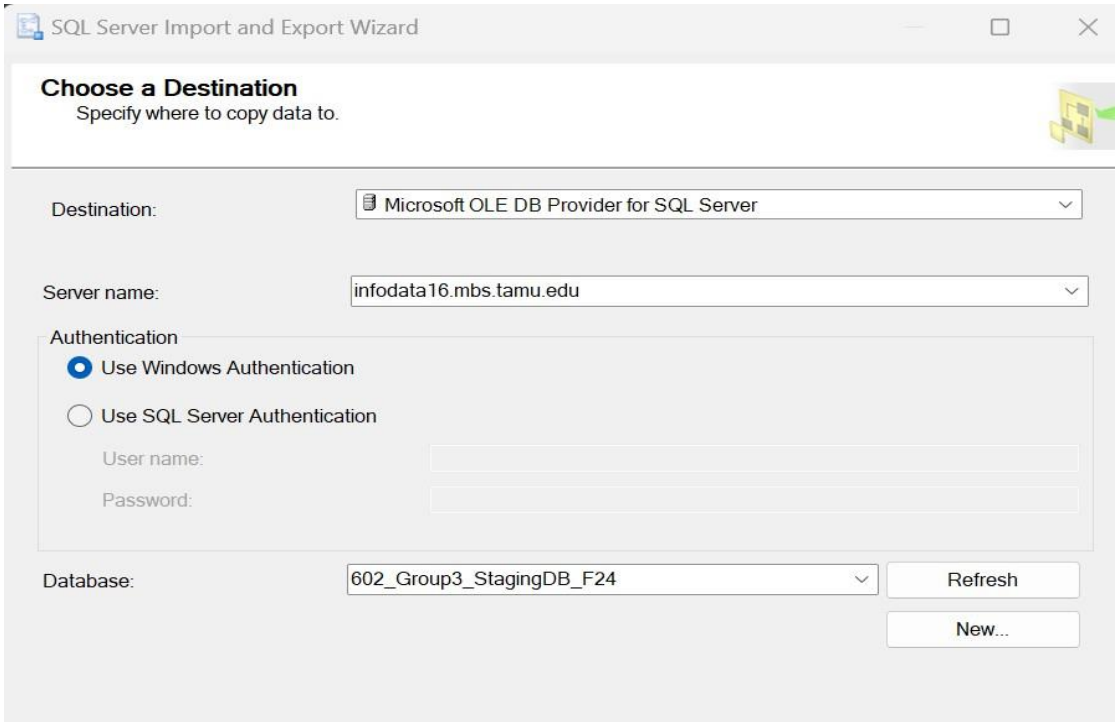
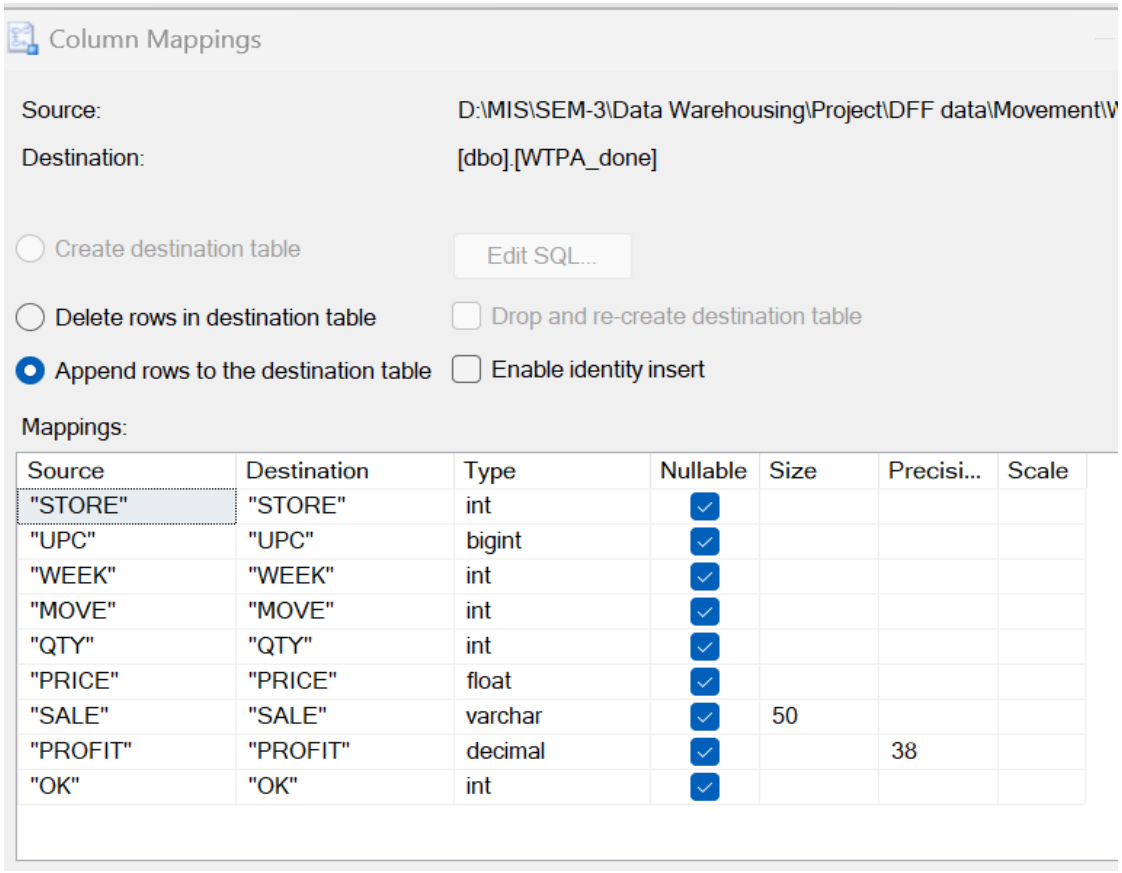


Fig: Select the destination



Source	Destination	Type	Nullable	Size	Precisi...	Scale
"STORE"	"STORE"	int	✓			
"UPC"	"UPC"	bigint	✓			
"WEEK"	"WEEK"	int	✓			
"MOVE"	"MOVE"	int	✓			
"QTY"	"QTY"	int	✓			
"PRICE"	"PRICE"	float	✓			
"SALE"	"SALE"	varchar	✓	50		
"PROFIT"	"PROFIT"	decimal	✓		38	
"OK"	"OK"	int	✓			

Fig: Edit the Mappings and make required changes to the data types

- Complete the setup
- Execute the package

```
SELECT TOP (1000) ["STORE"]
      ,["UPC"]
      ,["WEEK"]
      ,["MOVE"]
      ,["QTY"]
      ,["PRICE"]
      ,["SALE"]
      ,["PROFIT"]
      ,["OK"]
  FROM [602_Group3_StagingDB_F24].[dbo].[WTPA_done]
```

100 %

Results | Messages

	"STORE"	"UPC"	"WEEK"	"MOVE"	"QTY"	"PRICE"	"SALE"	"PROFIT"	"OK"
1	76	1111341101	208	6	1	1.34	""	40	1
2	76	1111341101	209	4	1	1.34	""	40	1
3	76	1111341101	210	16	1	0.99	"B"	19	1
4	76	1111341101	211	26	1	1.03	"B"	22	1
5	76	1111341101	212	0	1	0	""	0	1
6	76	1111341101	213	0	1	0	""	0	1
7	76	1111341101	214	1	1	1.34	""	40	1
8	76	1111341101	215	4	1	1.34	""	40	1
9	76	1111341101	216	4	1	1.34	""	40	1

Fig: Data is extracted and loaded into the staging table **dbo.WTPA_done**

UPCTPA: source to staging

- Create a new project in SSMS
- Right click on **SSIS Packages**
- Select **SSIS Import and Export Wizard**

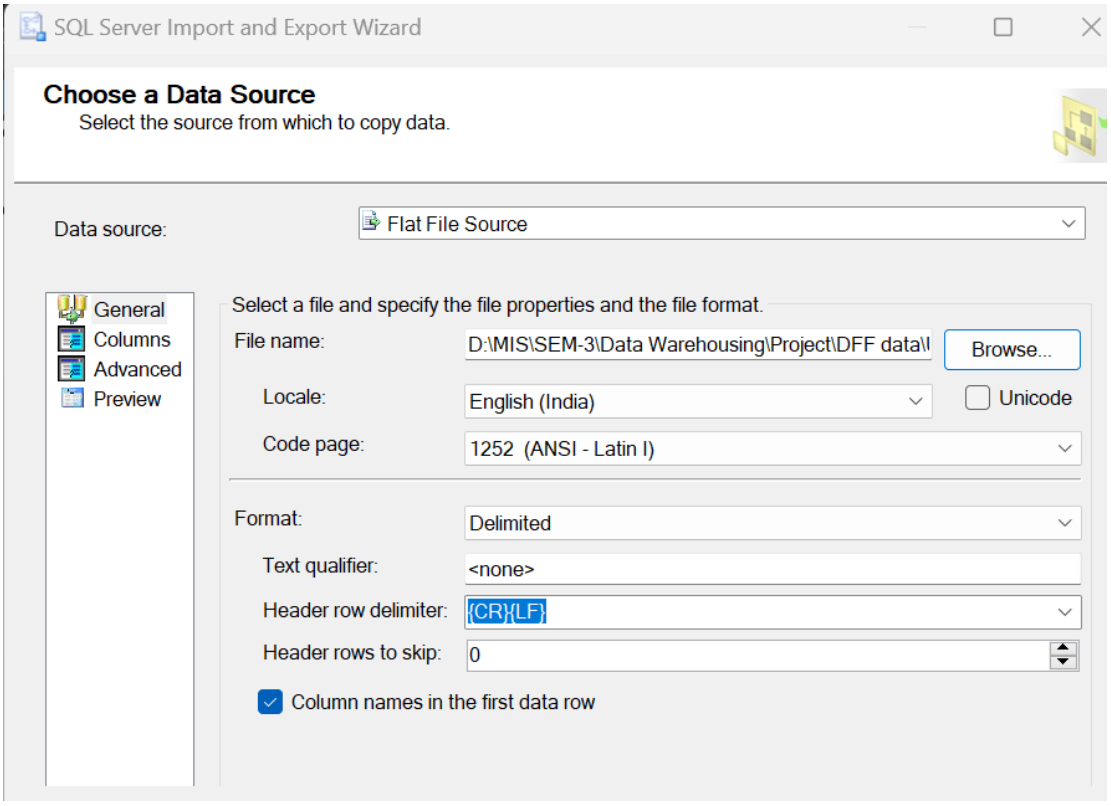


Fig: Select the data source and the UPCTPA.csv file

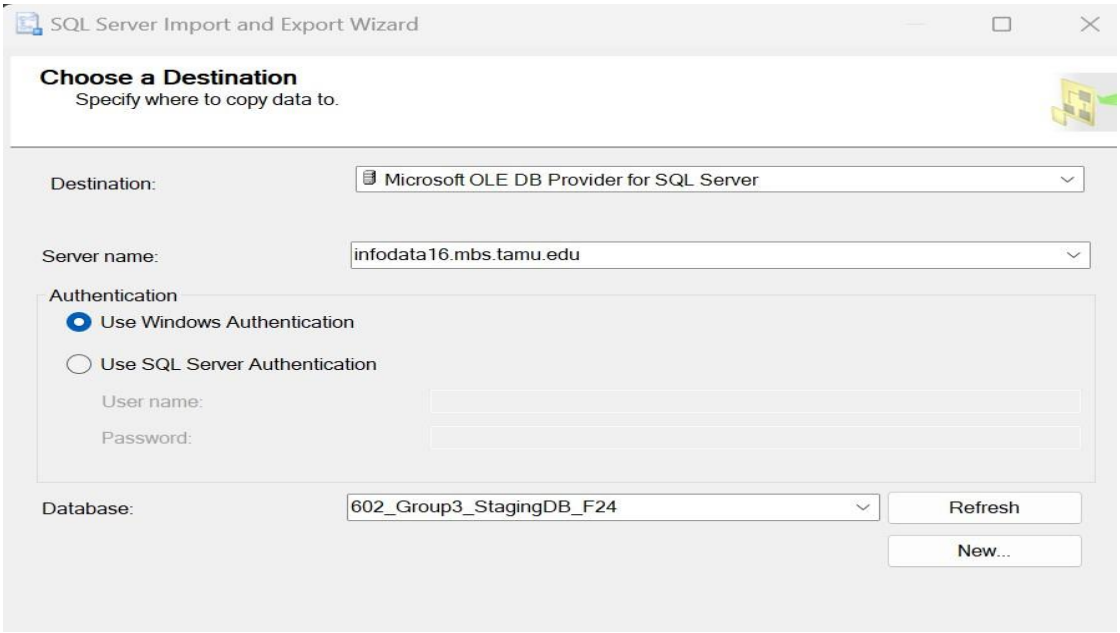


Fig : Select the destination

- Edit the mappings and make required changes to the data types
- Complete the setup

- Execute the package

```
SELECT TOP (1000) ["COM_CODE"]
      ,["UPC"]
      ,["DESCRIP"]
      ,["SIZE"]
      ,["CASE"]
      ,["NITEM"]
      ,[BRAND]
  FROM [602_Group3_StagingDB_F24].[dbo].[UPCTPA_clean]
```

0 %

Results | Messages

	"COM_CODE"	"UPC"	"DESCRIP"	"SIZE"	"CASE"	"NITEM"	BRAND
	956	1032718330	"ARM & HAMMER DENTAL"	"3 OZ"	36	7808111	ARM & HAMMER
	956	1032718350	"A/H DENTAL CARE TOOT"	"5 OZ"	12	6011271	ARM & HAMMER
	956	1032718370	"ARM & HAMMER DENTAL"	"7 OZ"	12	6011281	ARM & HAMMER
	956	1111304010	"CLOSE UP RED GEL W/4"	"8.2 OZ"	12	6014951	CLOSE UP
	956	1111307430	"MNTDNT TARTR TP W/FR"	"5.2 OZ"	12	6015871	MENTADENT
	956	1111307440	"MENTADENT TP W/FREE"	"3.5 OZ"	12	6015891	MENTADENT
	956	1111307450	"MNTDNT FRSH TP W/FRE"	"5.2 OZ"	12	6015811	MENTADENT
	956	1111307460	"MNTDNT COOL TP W/FRE"	"5.2 OZ"	12	6015831	MENTADENT
	956	1111310720	"MENTADENT REFILL TWN"	"10.4 O"	12	6015941	MENTADENT
0	956	1111311740	"PEPSODENT 28% FREE"	"8.2 OZ"	12	6014501	PEPSODENT

Fig: data is extracted and loaded into the staging table **dbo.UTPA_clean**

Data Cleaning and Data Transformation

2.1 Data cleaning of [cccount_stg]

Query 1: Renaming of columns

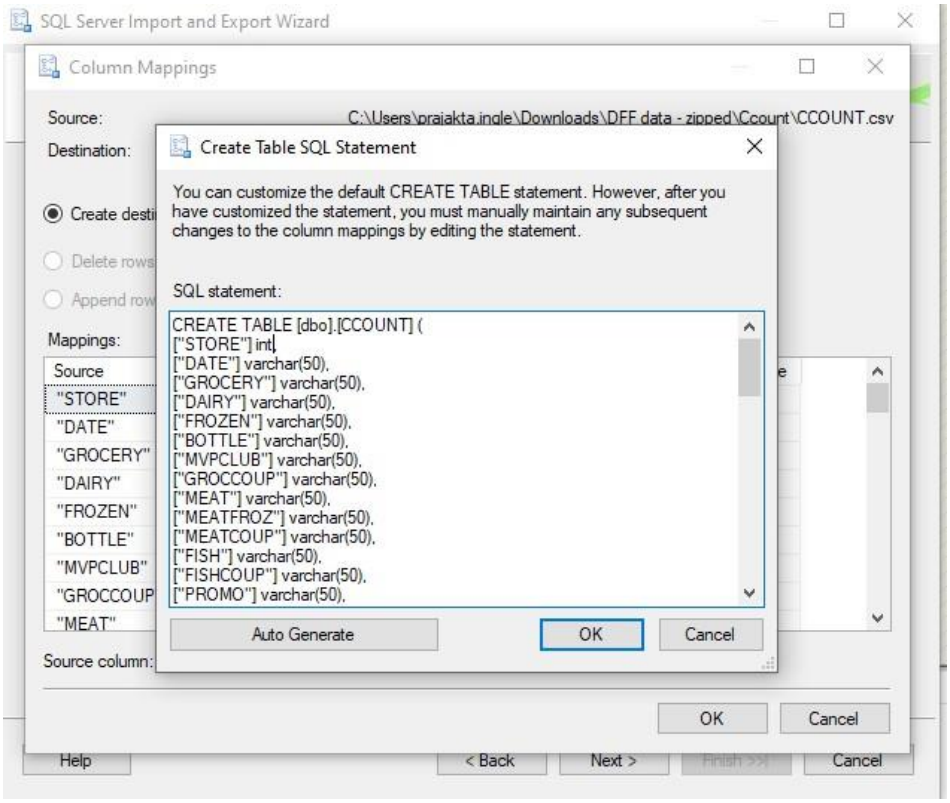


Figure: Column renaming using the SSIS.

Query 2: Clean STORE column data

Steps:

- Use “Execute SQL Task” in the Control Flow

```
DELETE FROM [dbo].[cccount_stg]
        WHERE STORE IS NULL
                OR STORE = ‘0’
                OR STORE = ‘-0’
                OR ( STORE LIKE ‘-%’ AND STORE != ‘-0’)
                OR STORE = ‘’
```

OR STORE NOT LIKE '%[1-9]%';

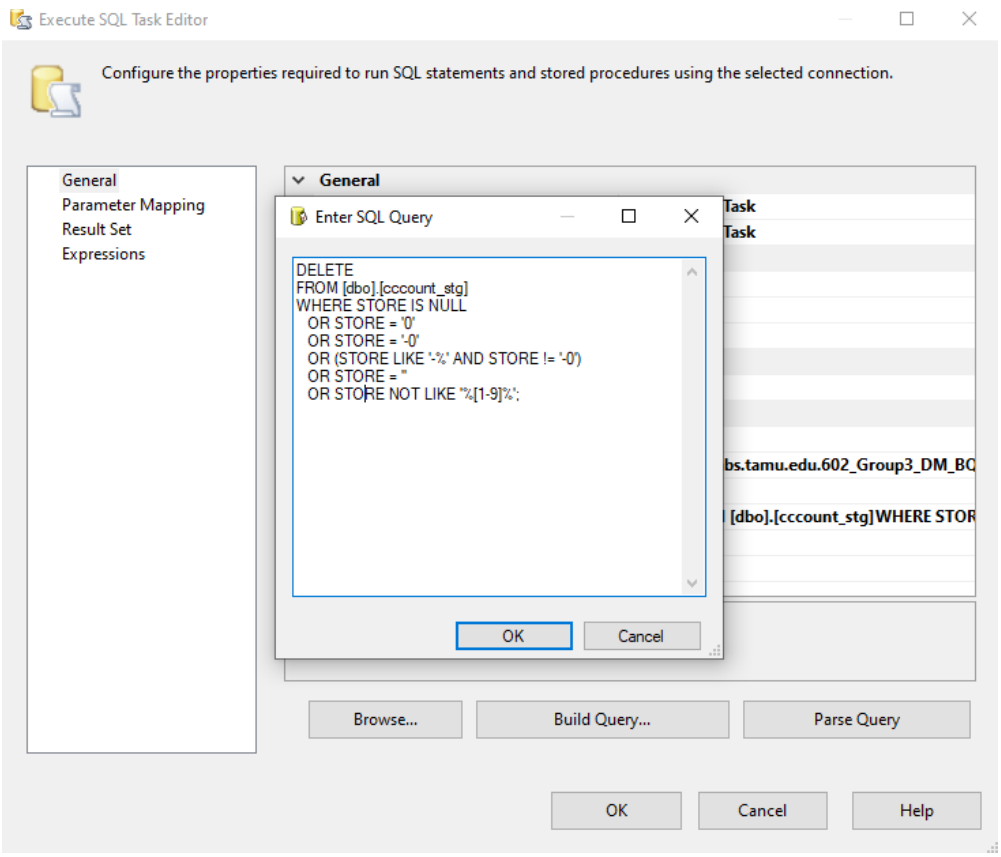


Figure: Cleaning STORE column using the SSIS

Query 3: Clean WEEK column data

Steps:

- Use "Execute SQL Task" in the Control Flow

DELETE FROM [dbo].[cccount_stg]

WHERE WEEK IS NULL

OR WEEK = '0'

OR WEEK = '-0'

OR (WEEK LIKE '-%' AND WEEK != '-0')

OR WEEK = ''

OR WEEK NOT LIKE '%[1-9]%';

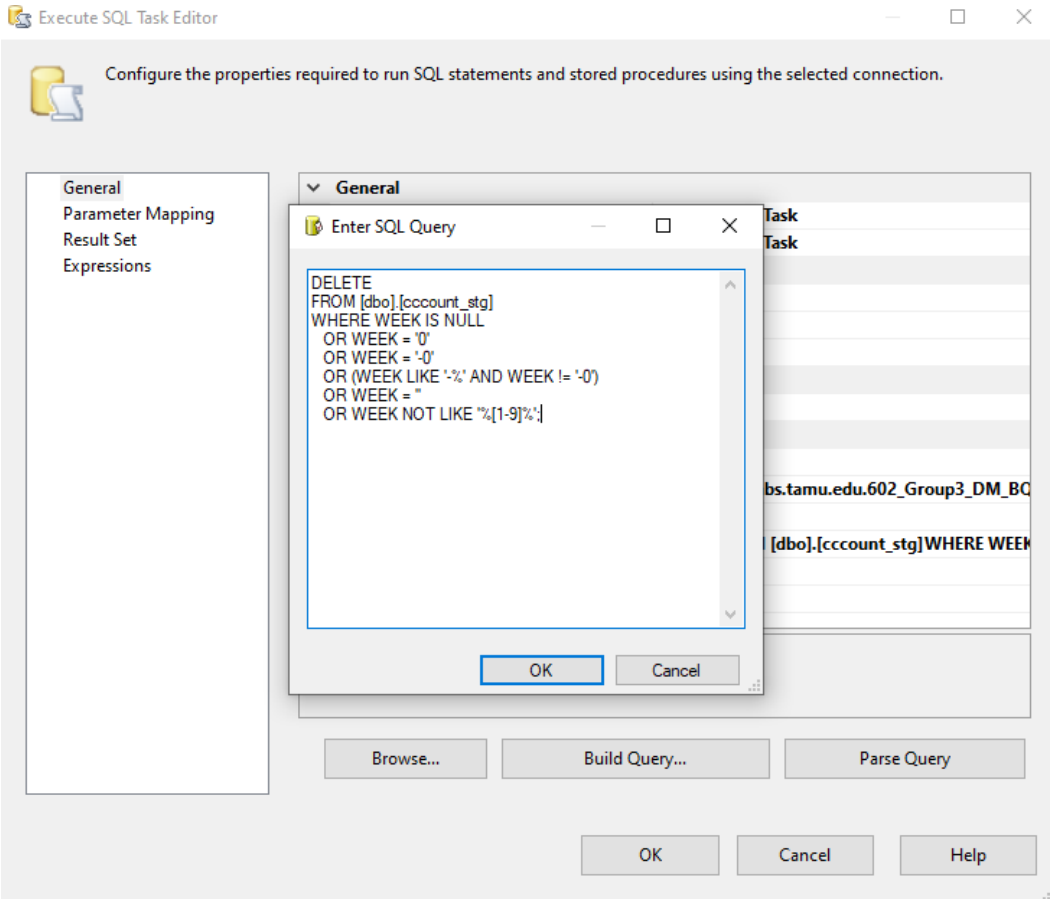


Figure: Cleaning WEEK column using the SSIS

Query 4: Changing the data type of the column

Steps:

- Use “Execute SQL Task” in the Control Flow

```
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [WEEK] INT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [STORE] INT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [DAIRY] FLOAT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [BAKERY] FLOAT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [GROCERY] FLOAT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [COSMETIC] FLOAT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [HABA] FLOAT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [BEER] FLOAT;
ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [DATE] DATE;
```

ALTER TABLE [dbo].[cccount_stg] ALTER COLUMN [CUSTCOUN] INT;

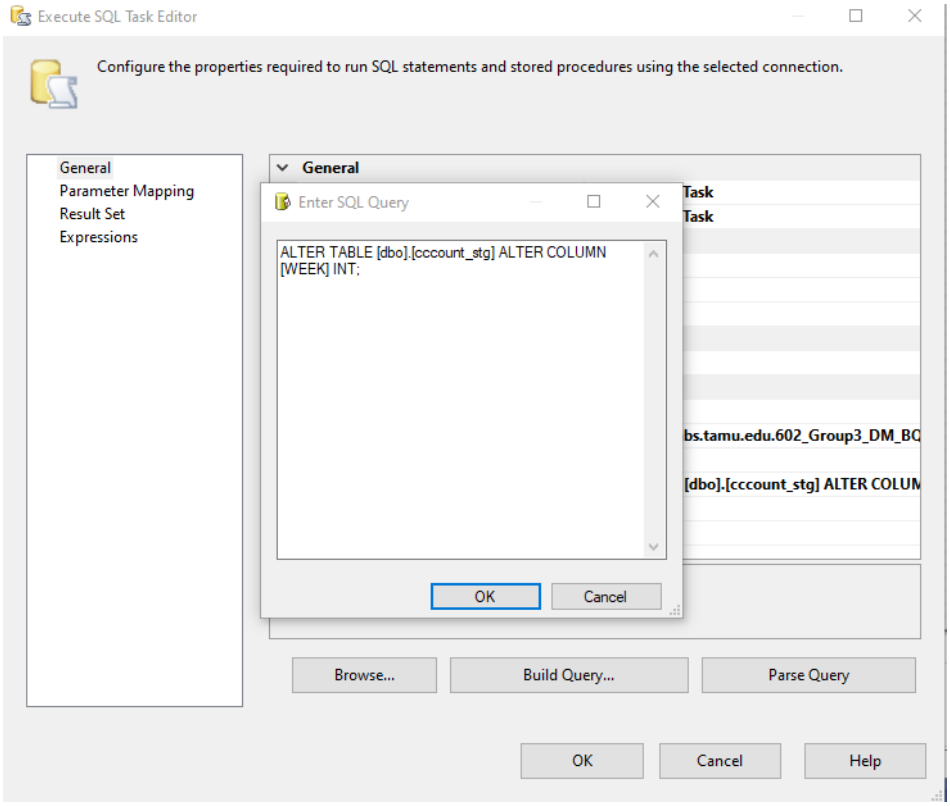


Figure: ALTER using the SSIS

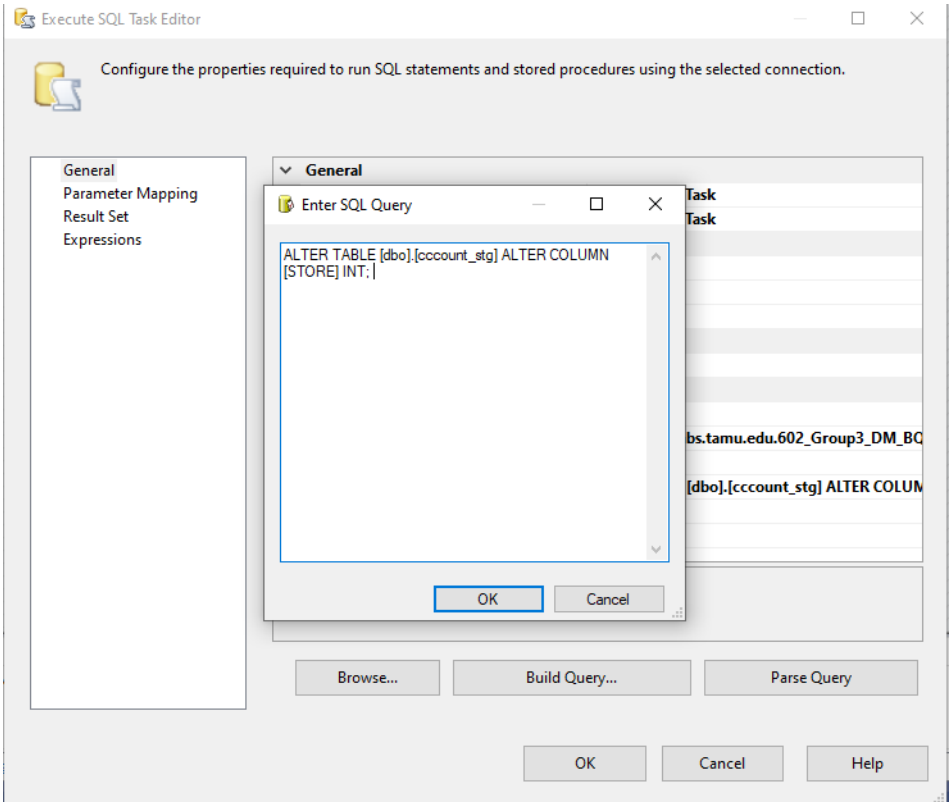


Figure: ALTER using the SSIS

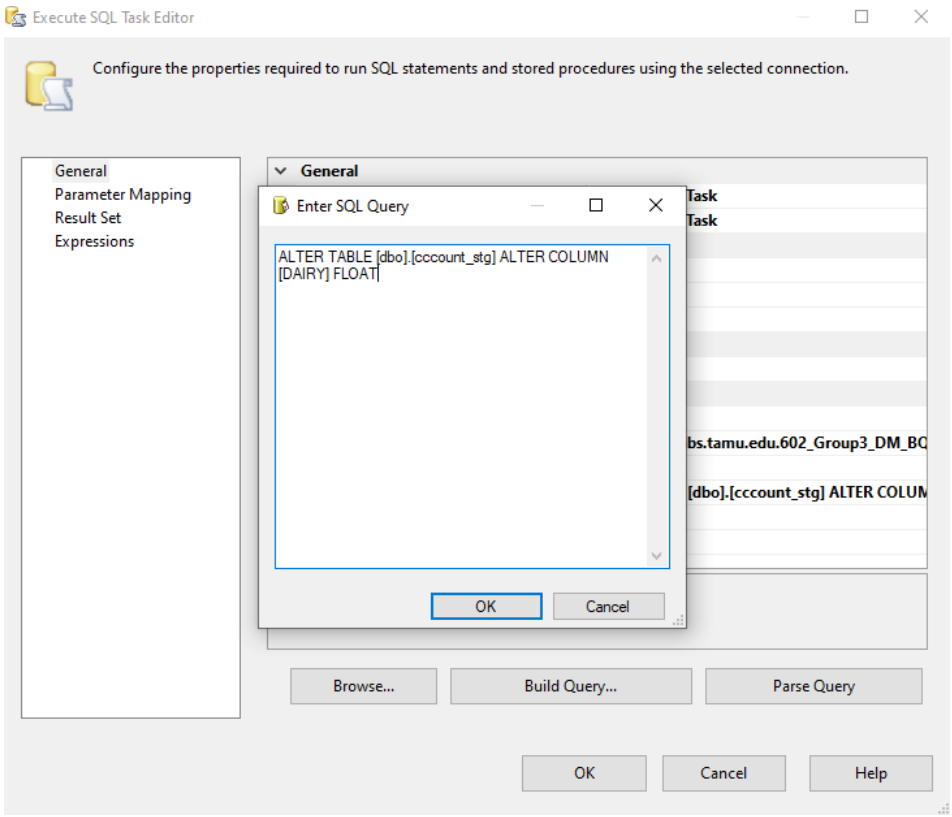


Figure: ALTER using the SSIS

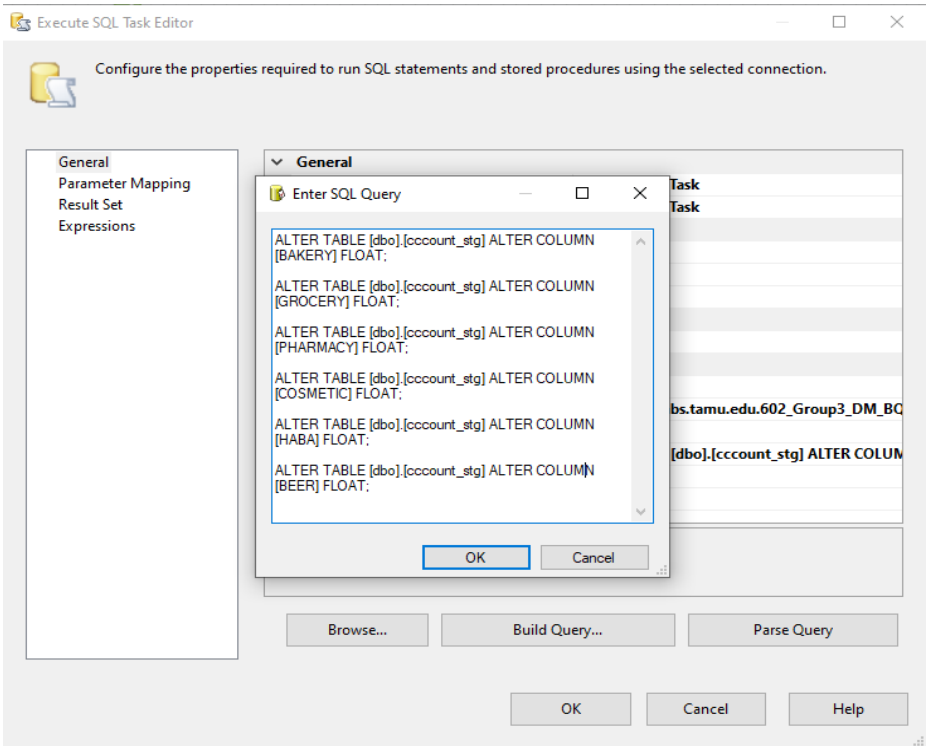


Figure: ALTER using the SSIS

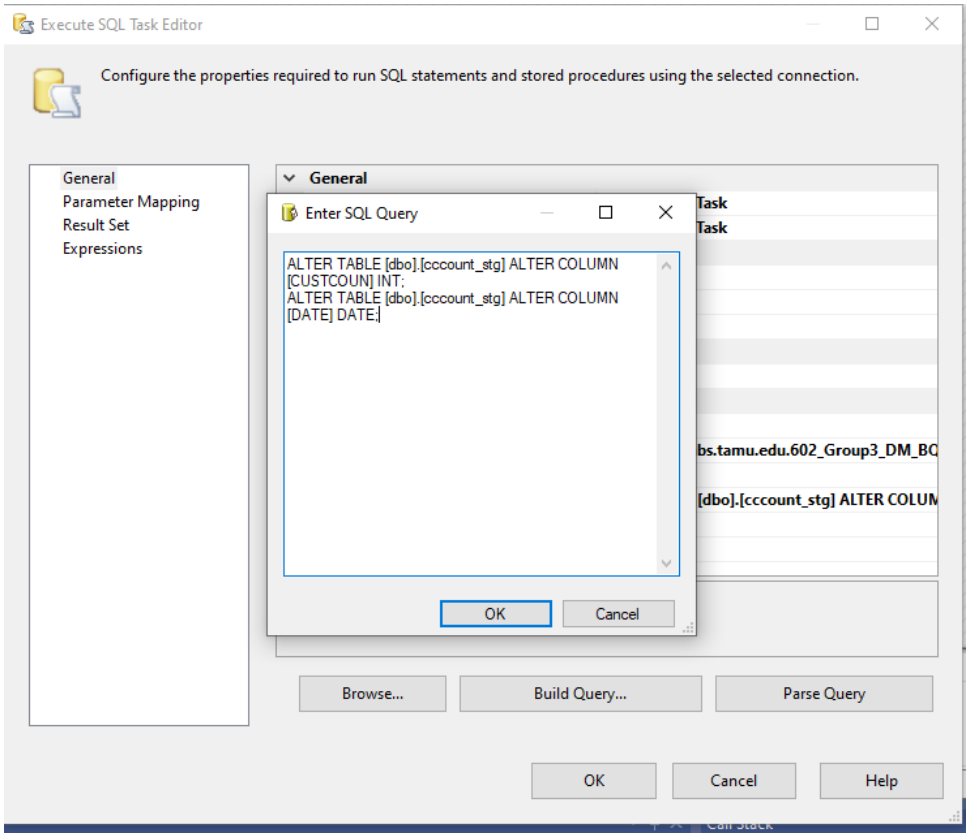


Figure: ALTER using the SSIS

Query 5: Clean CUSTCOUN column data

Steps:

- Use “Execute SQL Task” in the Control Flow

```
DELETE FROM [dbo].[cccount_stg]
        WHERE CUSTCOUN IS NULL
                OR CUSTCOUN = ‘0’
                OR CUSTCOUN = ‘-0’
                OR ( CUSTCOUN LIKE ‘-%’ AND CUSTCOUN != ‘-0’)
                OR CUSTCOUN = ‘’
                OR CUSTCOUN NOT LIKE ‘%[1-9]%’;
```

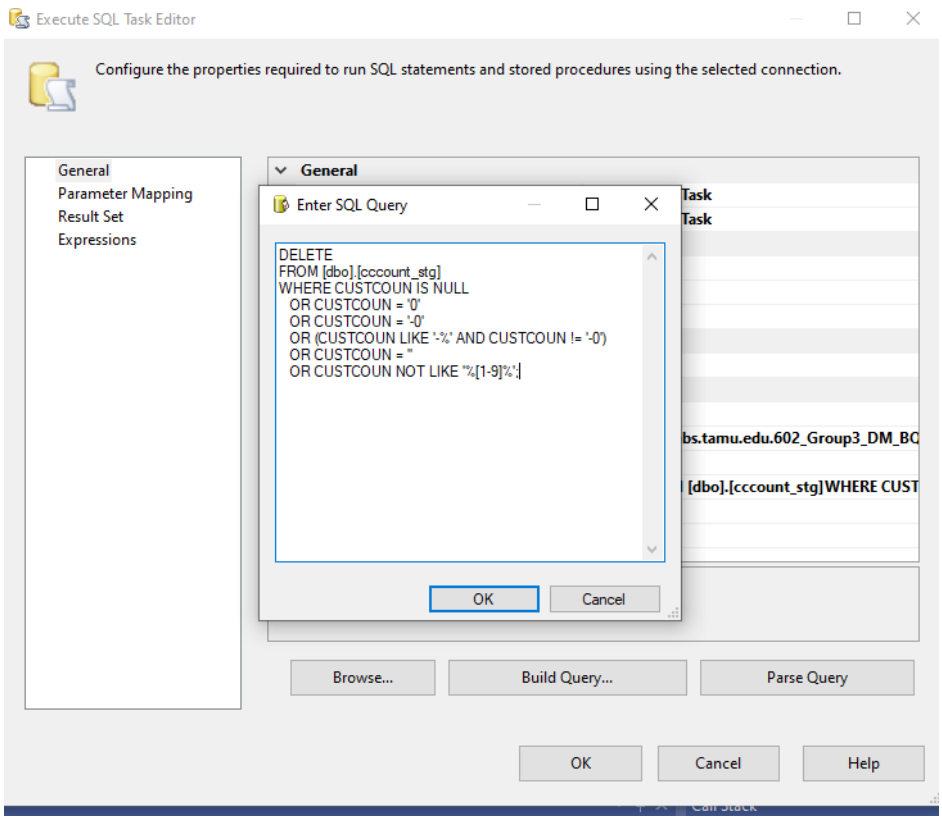


Figure: Cleaning CUSTCOUN column using the SSIS

Query 6: Update DATE column to convert into MM-DD-YYYY

Steps:

- Use “Execute SQL Task” in the Control Flow

```
UPDATE [dbo].[cccount_stg]
    SET DATE = SUBSTRING(DATE,1,2) +’-’  +
        SUBSTRING(DATE,3,2)+’-’ +SUBSTRING(DATE,5,2);
```

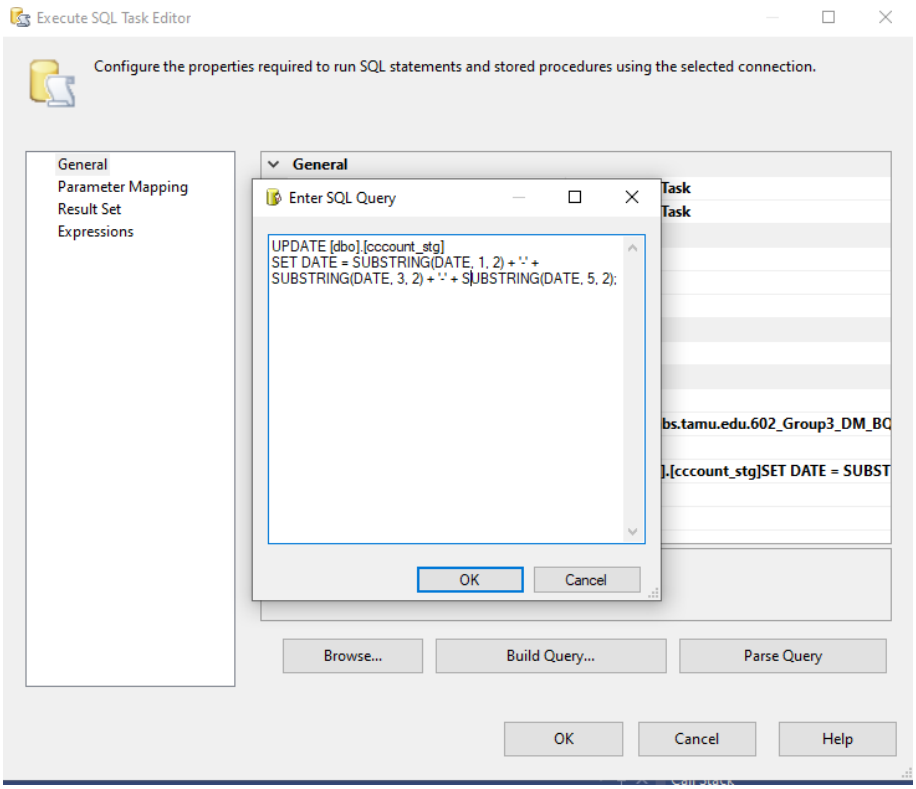


Figure: Updating DATE column using the SSIS

Converting DATE from String to DATE

```
UPDATE [dbo].[cccount_stg] SET DATE = CONVERT (DATE, DATE,2);
```

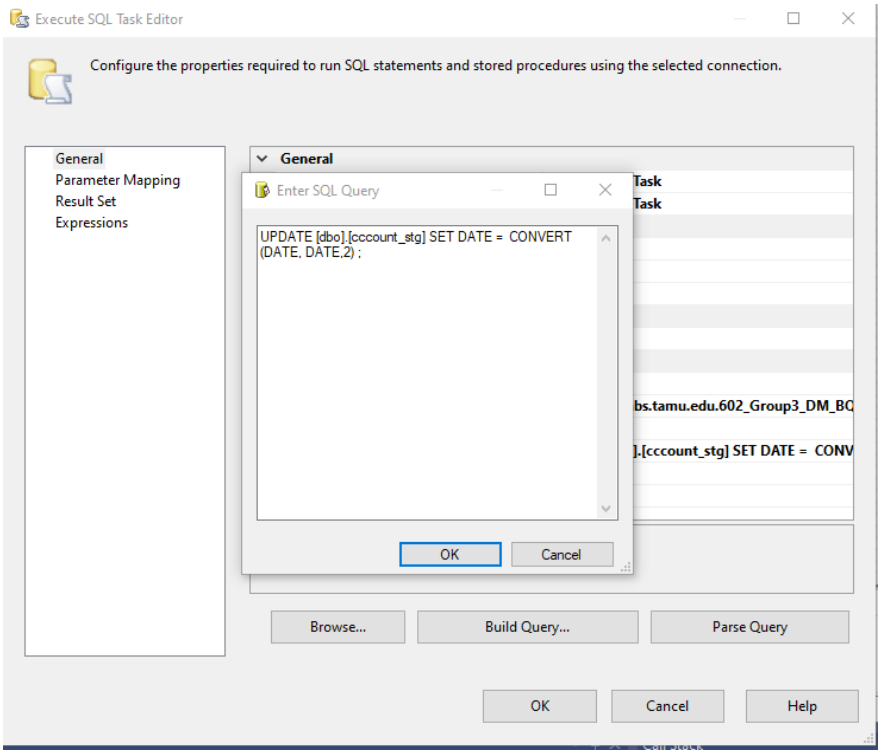


Figure: Converting the DATE column using the SSIS

```
ALTER TABLE [dbo].[cccount_stg] ADD COLUMN [YEAR] INT;
ALTER TABLE [dbo].[cccount_stg] ADD COLUMN [MONTH] INT;
ALTER TABLE [dbo].[cccount_stg] ADD COLUMN [QUARTER] INT;
ALTER TABLE [dbo].[cccount_stg] ADD COLUMN [DAY] INT;
```

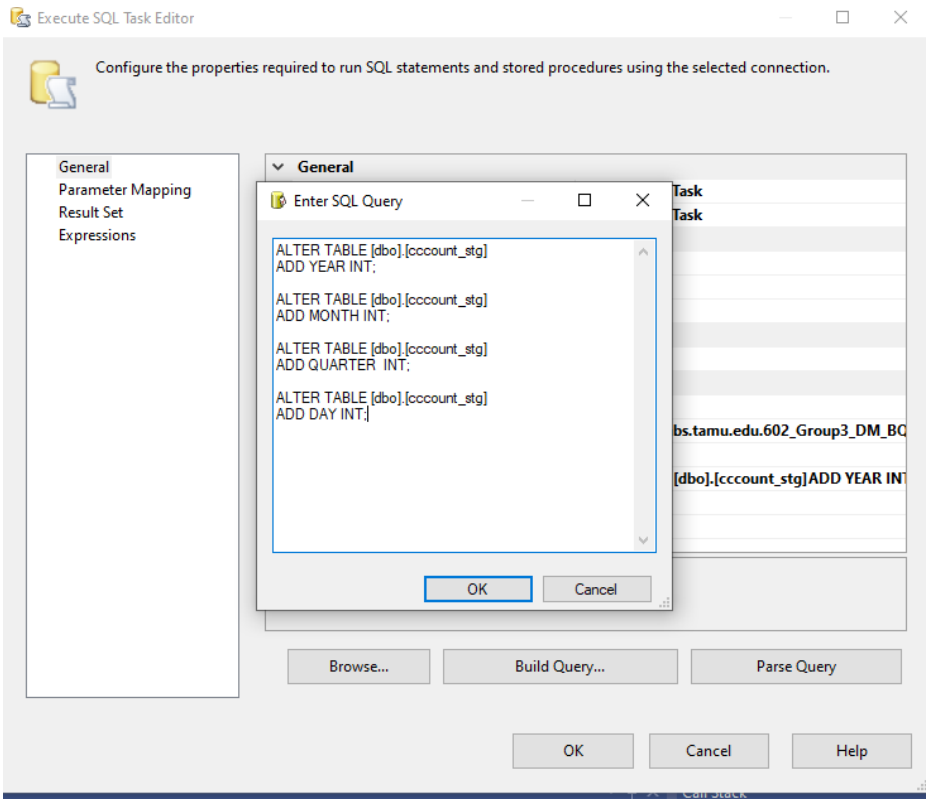


Figure: Adding attributes for YEAR, MONTH, QUARTER, DAY using the SSIS

Filling the values for DAY, MONTH, YEAR

```
UPDATE [dbo].[cccount_stg]
    SET YEAR = YEAR(DATE),
    MONTH = MONTH(DATE),
    DAY = DAY(DATE),
    QUARTER = DATEPART(QUARTER, DATE);
```

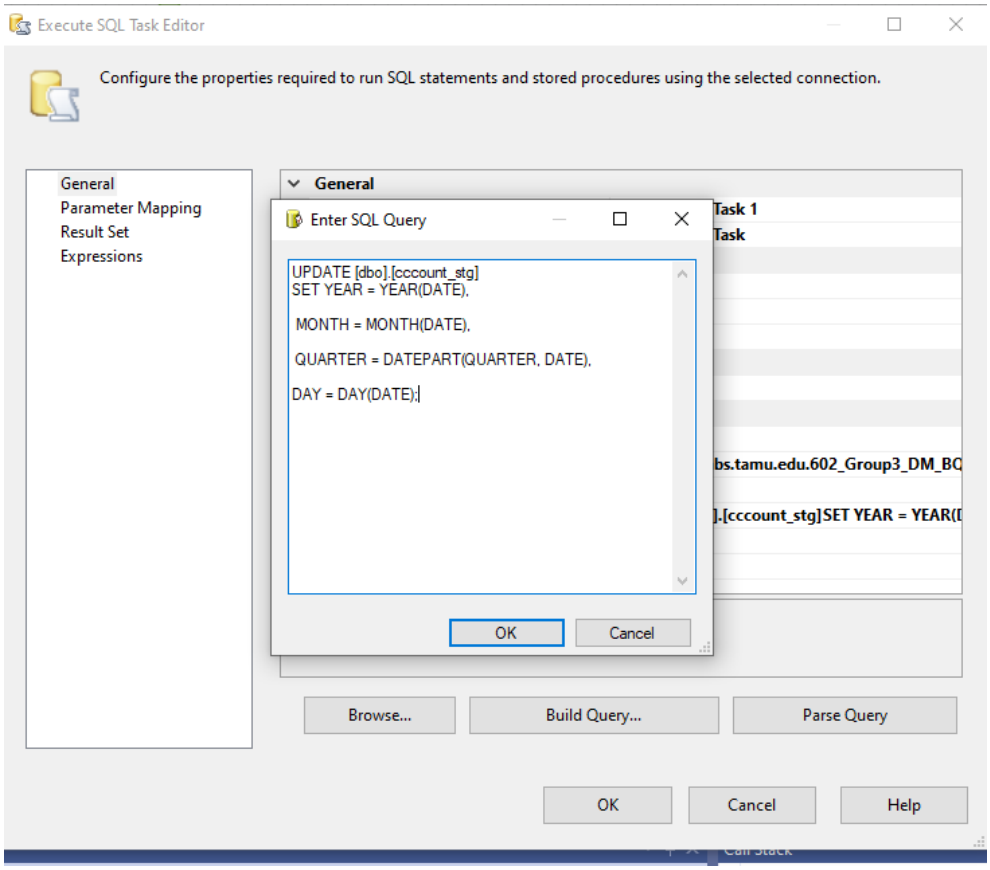


Figure: Filling for YEAR, MONTH, QUARTER, DAY using the SSIS

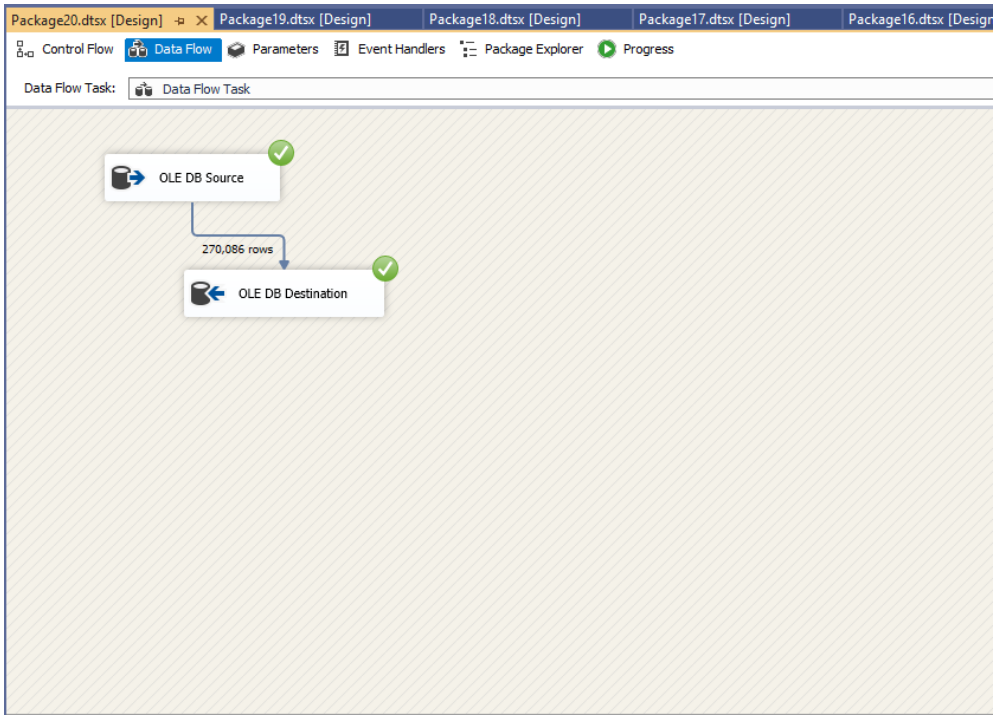


Figure: Execution for cleaning the cccount_stg

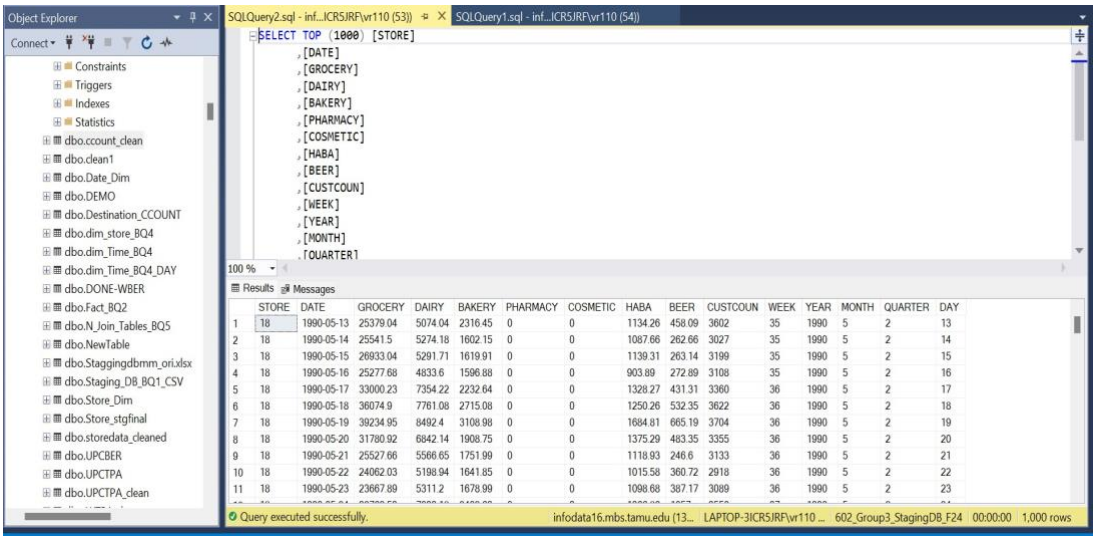


Figure: Validation for cleaning the cccount_stg

2.2 Cleaning For BQ_2_data:

Query 1: Updating the Table with New Column - WEEK_Name

Steps:

- Use “Execute SQL Task” in the Control Flow

ALTER TABLE [602_Group 3 _Staging DB_F24]. [dbo].[BQ_2_data]

ADD WEEK_ NAME VARCHAR(50):

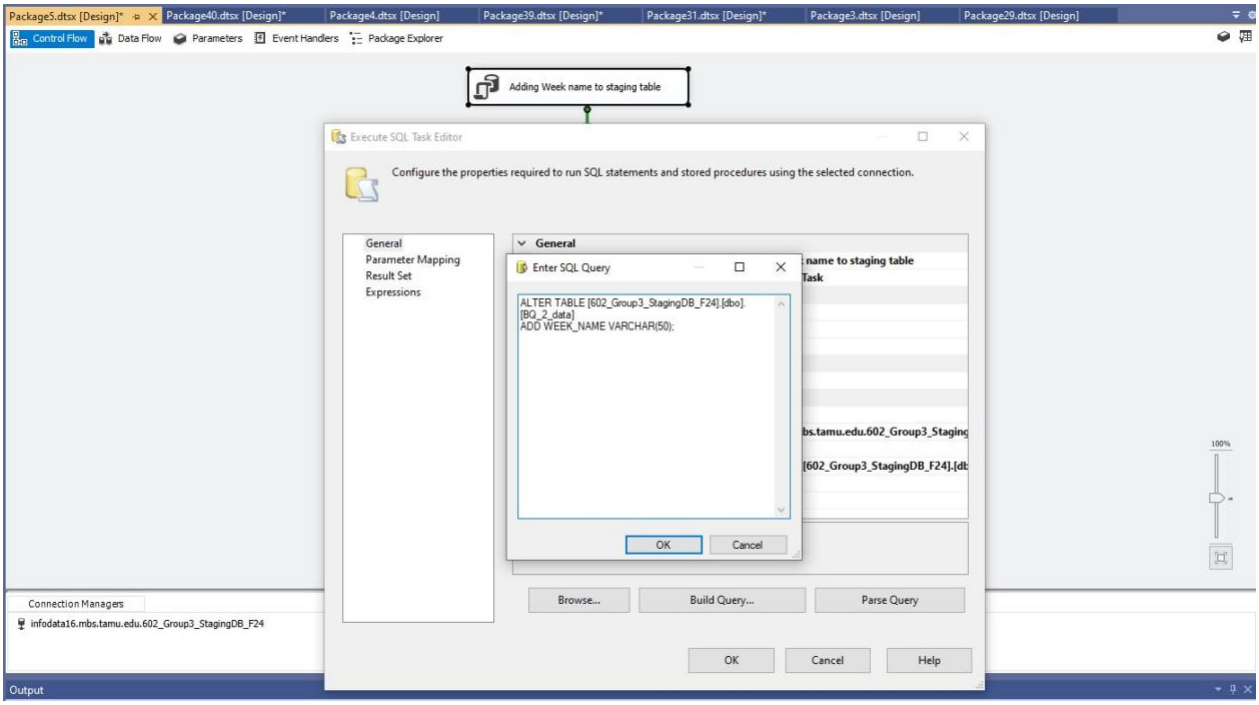


Fig Adding a New Column WEEk_NAME to BQ_2_data Table

Query 2: Updating the Table with New Column Data - WEEK_Name

Steps:

- Use “Execute SQL Task” in the Control Flow

```
UPDATE [602_Group 3_Staging DB_F24].[dbo].
[BQ_2 _data]
SET WEEK_NAME = |
CASE
WHEN WEEK = 68 THEN 'New-Year'
WHEN WEEK = 75 THEN 'Presidents Day'
WHEN WEEK = 81 THEN 'Easter'
WHEN WEEK = 89 THEN 'Memorial Day'
WHEN WEEK = 95 THEN '4th of July'
WHEN WEEK = 103 THEN Labor Day
WHEN WEEK = 112 THEN 'Halloween'
WHEN WEEK = 116 THEN Thanksgiving'
```

WHEN WEEK = 119 THEN Christmas*

WHEN WEEK = 120 THEN 'New-Year'

WHEN WEEK = 128 THEN 'Presidents Day'

WHEN WEEK = 133 THEN 'Easter'

WHEN WEEK = 141 THEN 'Memorial Day'

WHEN WEEK = 147 THEN '4th of July

WHEN WEEK = 156 THEN Labor Day'

WHEN WEEK = 164 THEN 'Halloween'

WHEN WEEK = 168 THEN Thanksgiving*

WHEN WEEK = 172 THEN Christmas*

WHEN WEEK = 173 THEN 'New-Year'

ELSE 'Regular Week'

END;

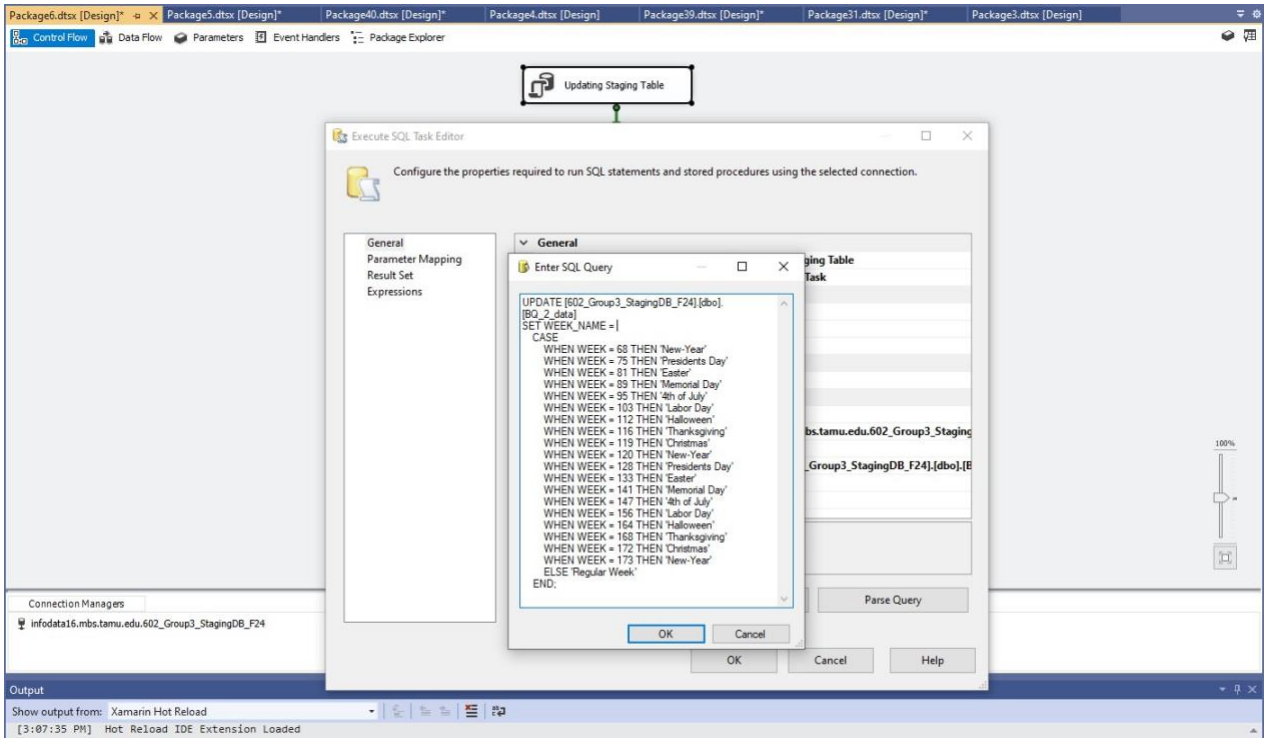


Fig Updating the table with required week names using SSIS

2.3 Cleaning For BQ5_data:

Query 1: Joining the Table UPCBER & DONE-WBER to get N_Join_Tables_BQ5

Steps:

- Use “Execute SQL Task” in the Control Flow

```
SELECT
UPCBER.DESCRIP,  UPCBER.UPC, UPCBER.[SIZE],
UPCBER.[CASE], UPCBER.NITEM, [DONE-WBER] ['STORE], [DONE-WBER].[
“UPC"],
[DONE-WBER].['WEEK"], [DONE-WBER]. ["MOVE"],
[DONE-WBER].[“QTY"],
[DONE-WBER].['PRICE"]. [DONE-WBER].['PROFIT"]
FROM
UPCBER INNER JOIN [DONE-WBER] ON UPCBER.UPC =[DONE-BER].[“UPC"]
```

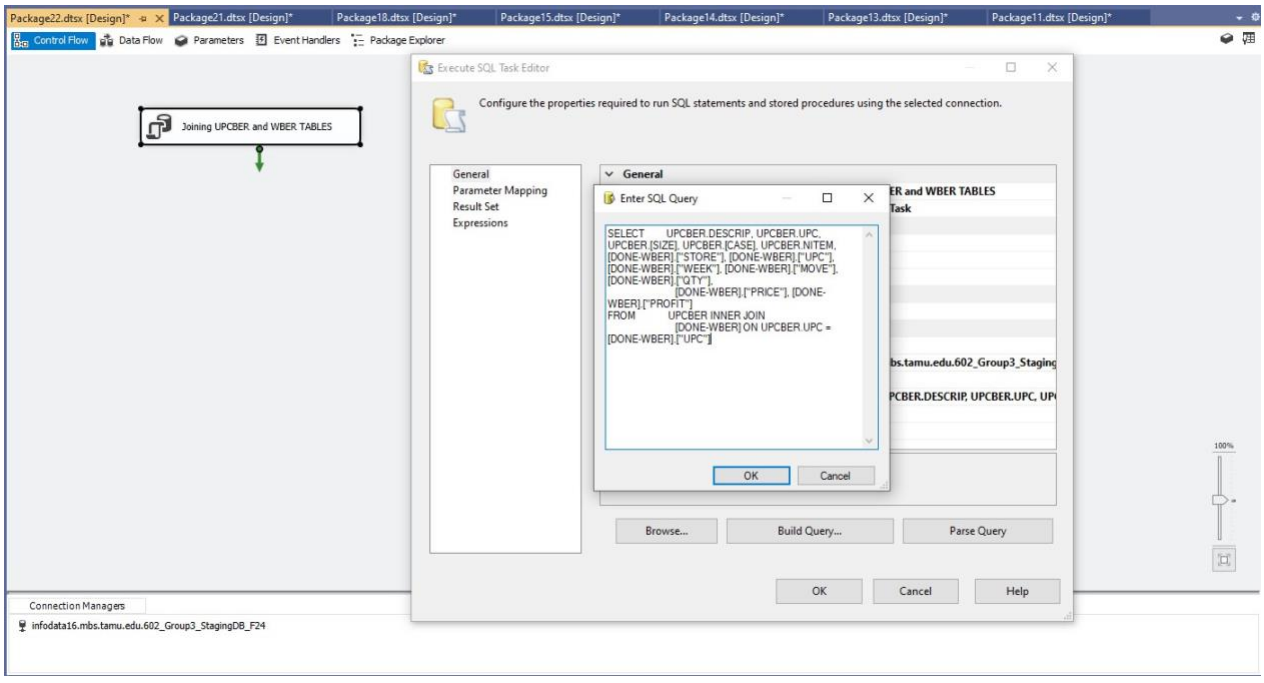


Fig Join UPCBER and DONE-WBER Tables

Query 2: Changing the data types for the columns in N_Join_Tables_BQ5

Steps:

- Use “Execute SQL Task” in the Control Flow

ALTER TABLE [dbo]. [N_ Join_Tables _BQ5]
ALTER COLUMN [STORE] INT;
ALTER TABLE [dbo]. [N_Join_Tables_BQ5]
ALTER COLUMN [WEEK] INT;
ALTER TABLE [dbo]. [N_Join_Tables _BQ5]
ALTER COLUMN [QTY] INT;
ALTER TABLE [dbo]. [N_Join_Tables_BQ5]
ALTER COLUMN [PRICE] FLOAT;
ALTER TABLE [dbo]. [N_Join_Tables _BQ5]
ALTER COLUMN [PROFIT] FLOAT;

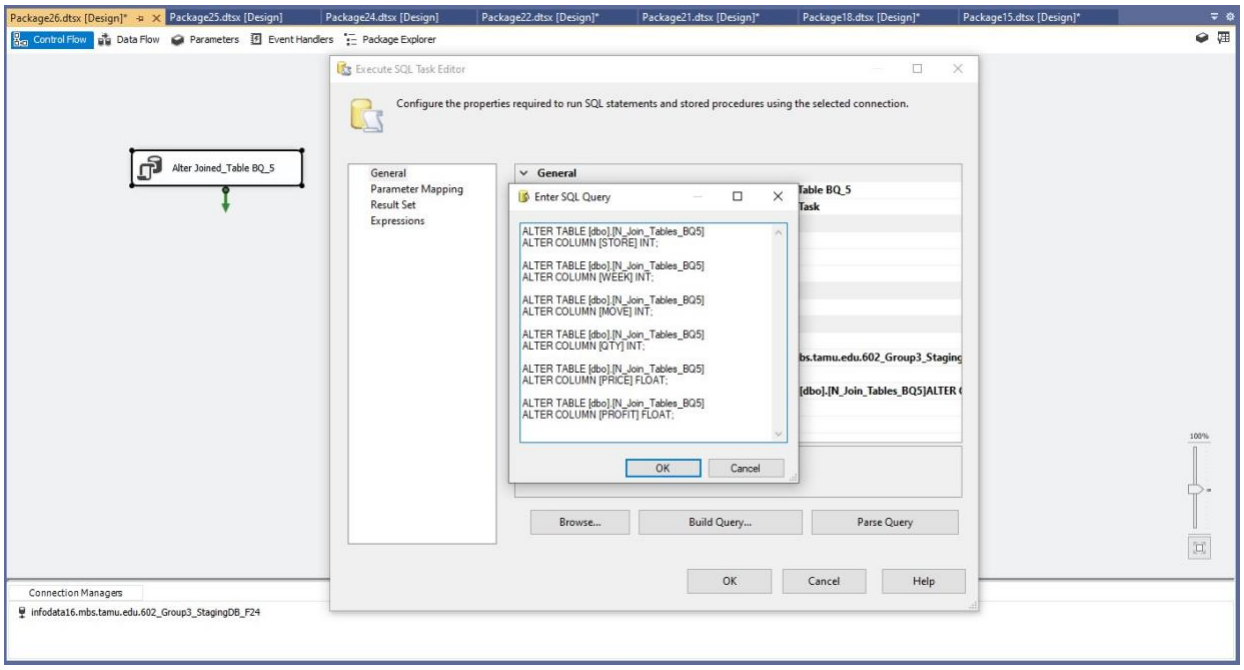


Fig Changing the Data types for the Joined table

Query 3: Adding a new Sales Column in BQ5_Cleaned

Steps:

- Use “Execute SQL Task” in the Control Flow

```
ALTER TABLE [dbo].[BQ5_Cleaned]
ADD Sales FLOAT;
```

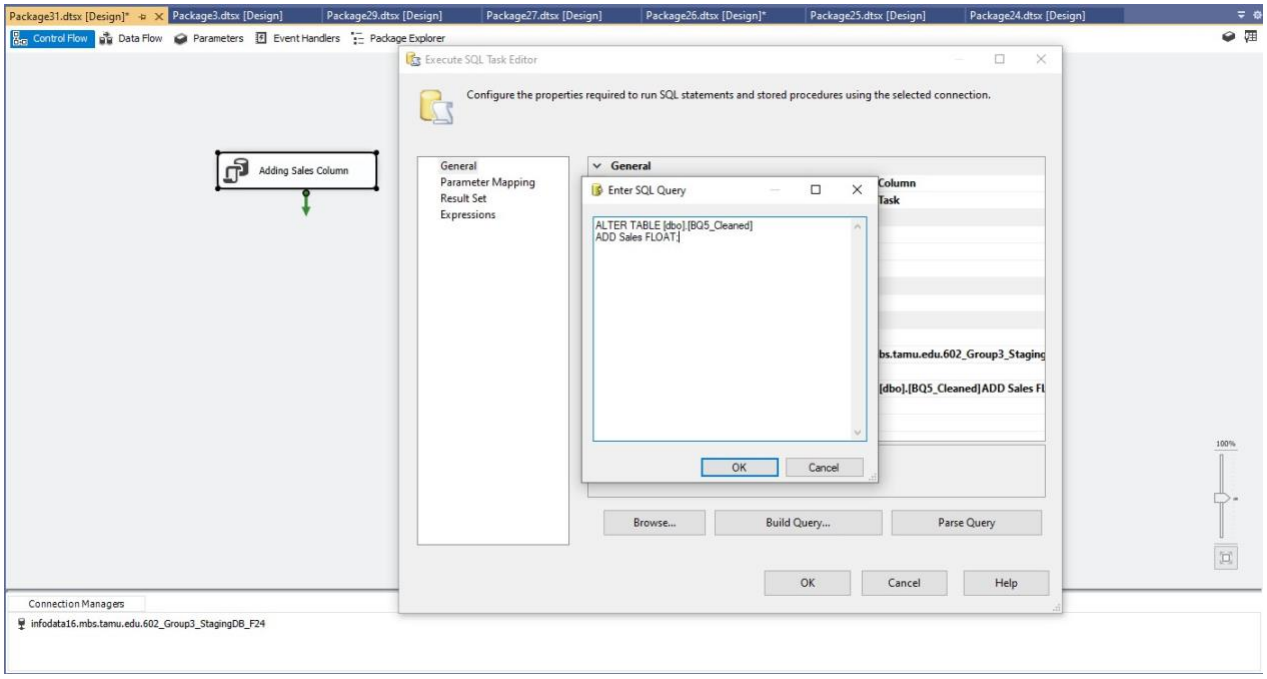


Fig Adding a New Sales Column

Query 4: updating the formula for new Sales Column in BQ5_Cleaned

Steps:

- Use “Execute SQL Task” in the Control Flow

```
UPDATE [dbo].[BQ5_Cleaned]
SET Sales = (Price * Move) / NULLIF(Qty, 0);
```

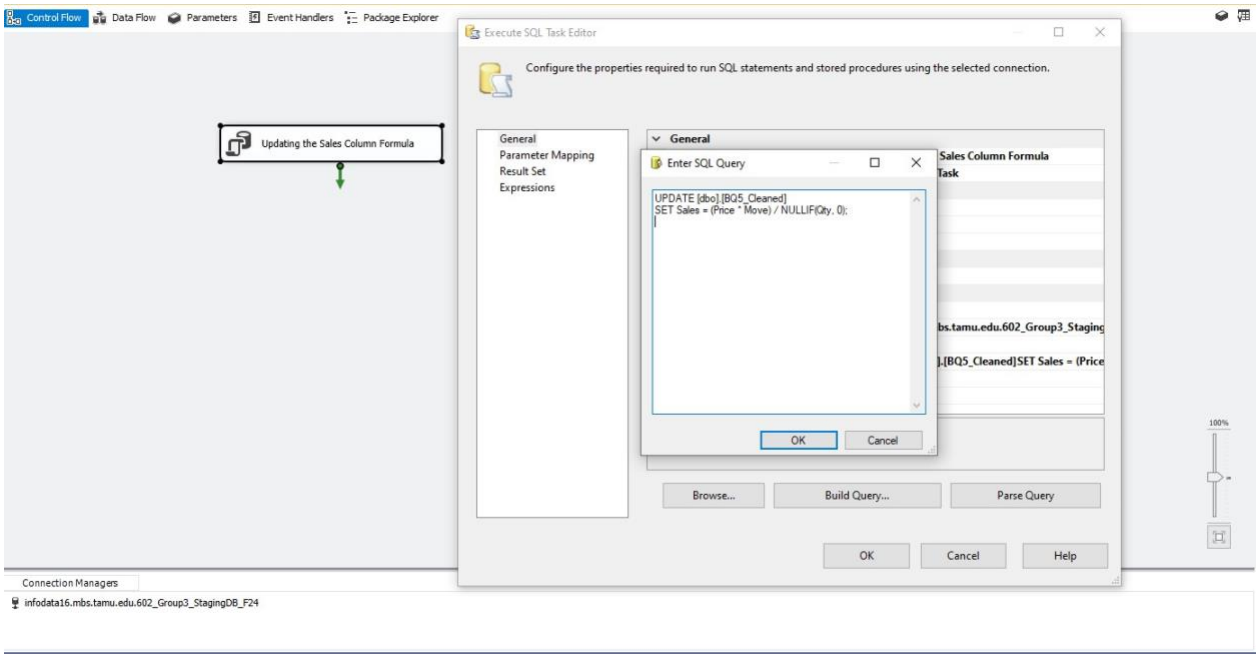


Fig Updating the formula for New Sales Column

Query 5: Adding a new DESCRIP Column in BQ5_Cleaned

Steps:

- Use “Execute SQL Task” in the Control Flow

```
ALTER TABLE [dbo].[BQ5_Cleaned]
ADD DESCRIP VARCHAR(250);
```

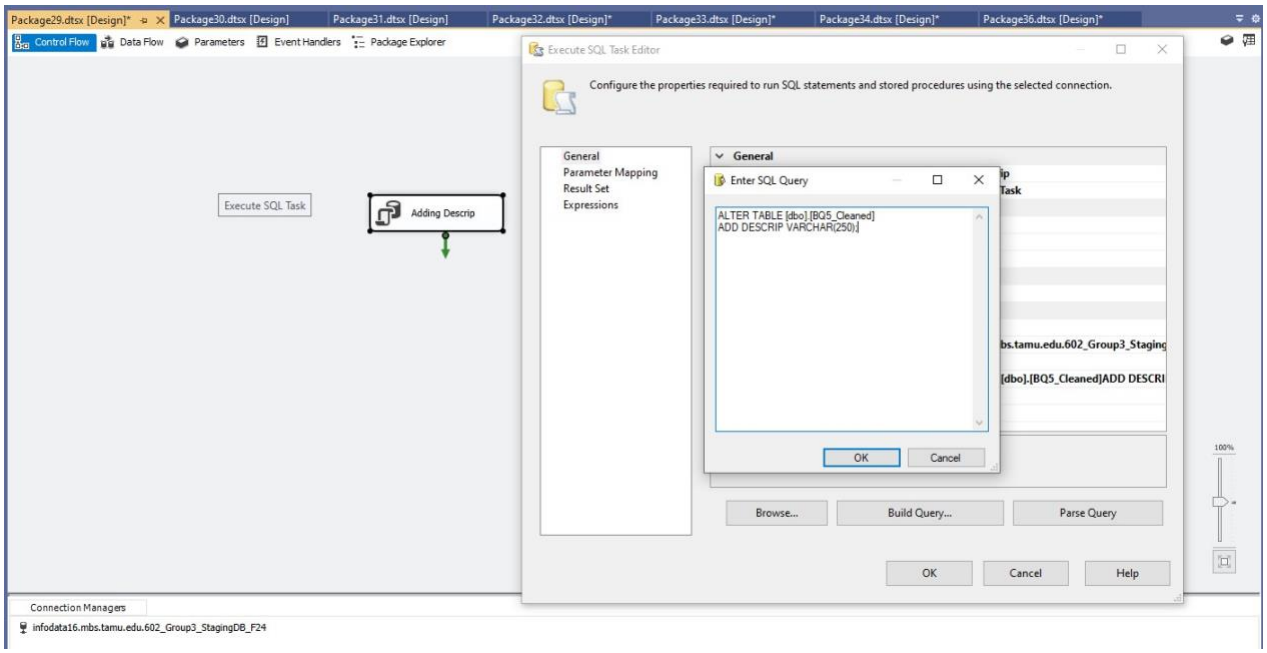


Fig Adding a new column DESCRIP

Query 6: Loading Data into **BQ5_Demo_Cleaned** from **DEMO** file

Steps:

- Use “Execute SQL Task” in the Control Flow

```
SELECT ["MMID"]
      ,["NAME"]
      ,["CITY"]
      ,["STORE"]
      ,["SINGLE"]
      ,["RETIRED"]
 INTO [dbo].[BQ5_Demo_Cleaned]
 FROM [602_Group3_StagingDB_F24].[dbo].[DEMO]
 WHERE ["STORE"]='112';
```

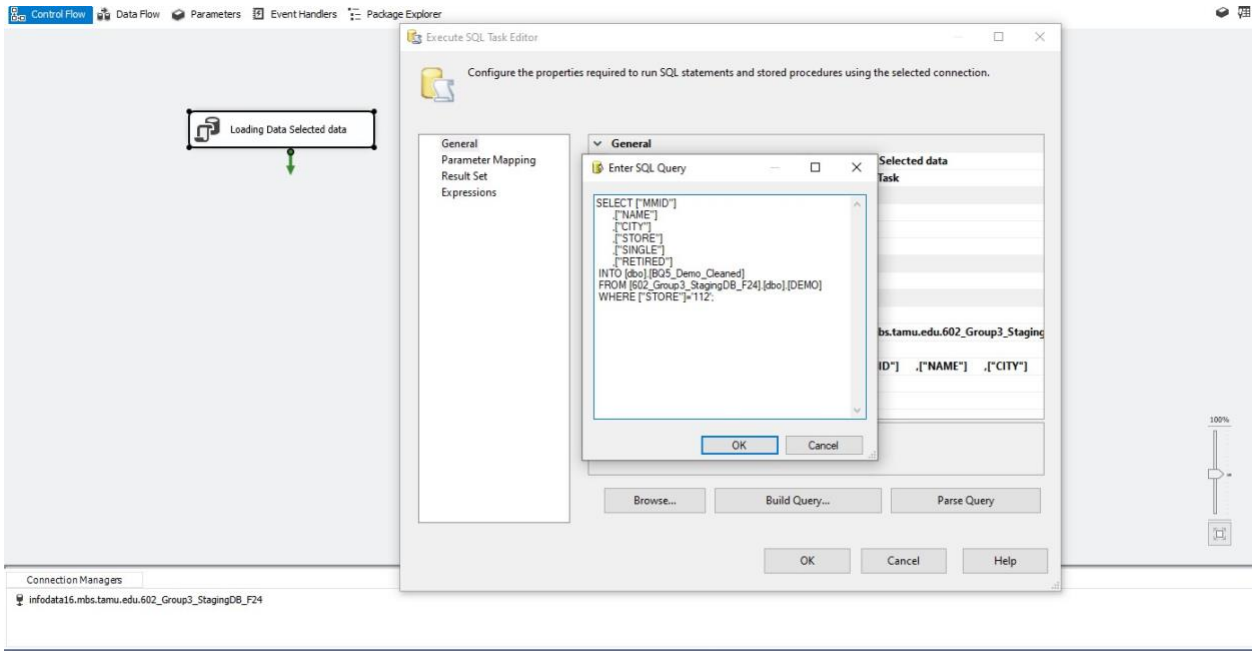


Fig Loading Selected data from Demo Table to BQ5_Demo_Cleaned Table

Query 7: Renaming the column names in BQ5_Demo_Cleaned

Steps:

- Use “Execute SQL Task” in the Control Flow

```
EXEC sp_rename 'dbo.BQ5_Demo_Cleaned.["MMID"]', 'MMID', 'COLUMN';
EXEC sp_rename 'dbo.BQ5_Demo_Cleaned.["NAME"]', 'NAME', 'COLUMN';
EXEC sp_rename 'dbo.BQ5_Demo_Cleaned.["CITY"]', 'CITY', 'COLUMN';
EXEC sp_rename 'dbo.BQ5_Demo_Cleaned.["STORE"]', 'STORE', 'COLUMN';
EXEC sp_rename 'dbo.BQ5_Demo_Cleaned.["SINGLE"]', 'SINGLE', 'COLUMN';
EXEC sp_rename 'dbo.BQ5_Demo_Cleaned.["RETIRED"]', 'RETIRED', 'COLUMN';
```

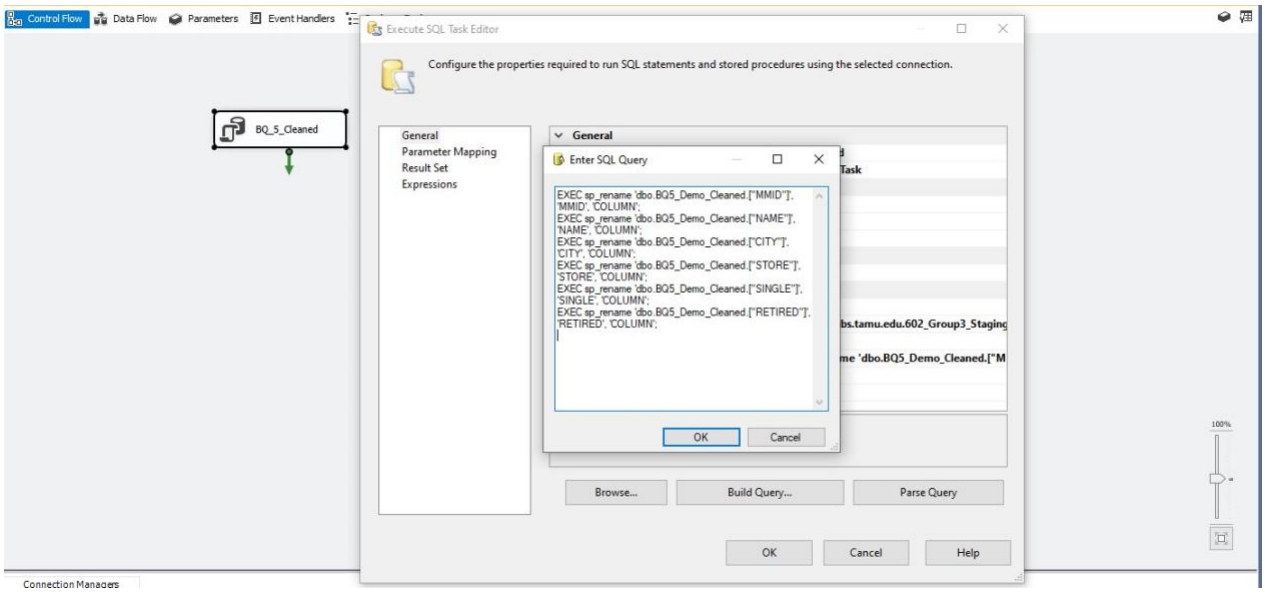


Fig Renaming the column names

2.4 dbo.WTPA_done

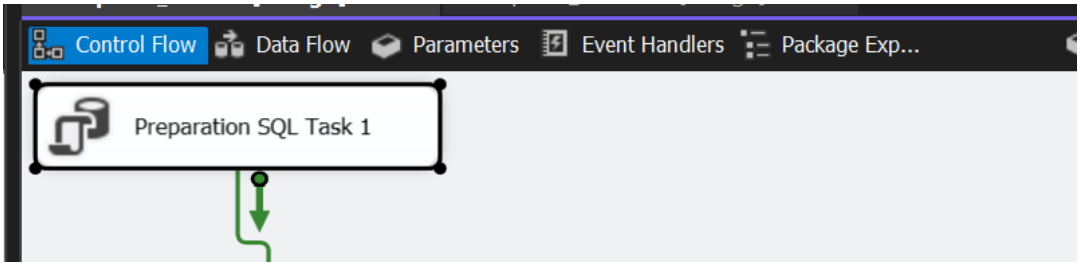


Fig: Drag *Execute SQL Task* to Control Flow

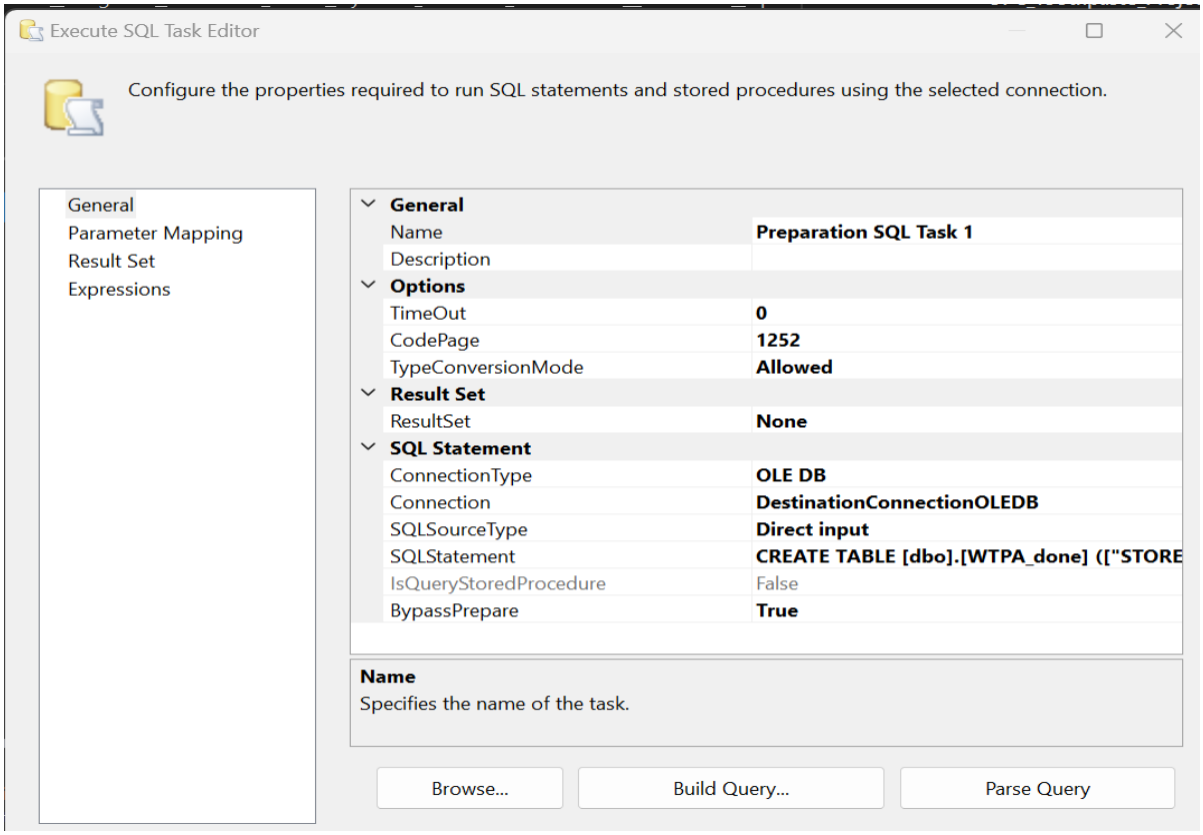


Fig: Update DB Connection details to staging database

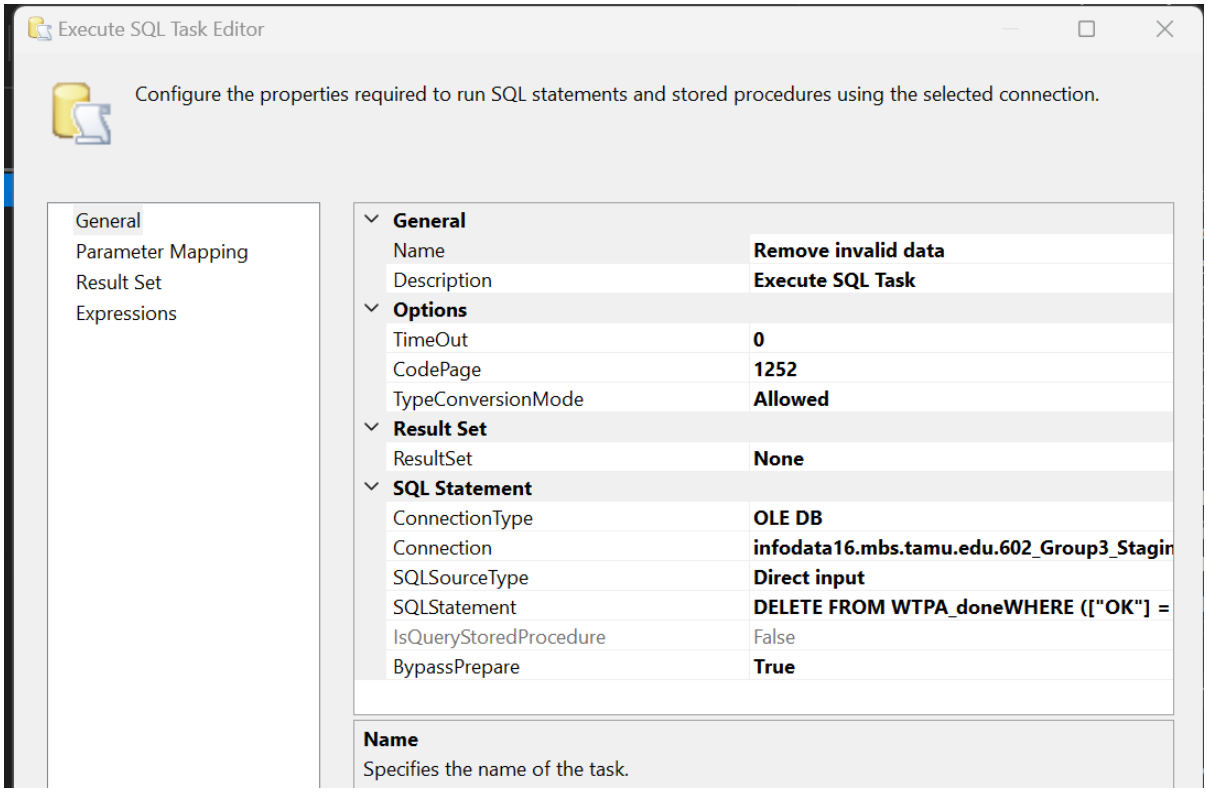


Fig: Remove invalid data [OK=0]

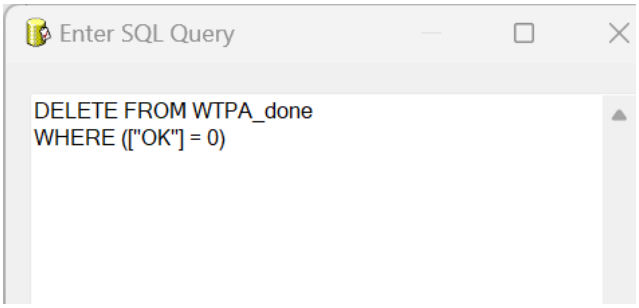


Fig: Query to delete invalid data

- Execute the package

2.5 Dbo.UPCTPA_clean

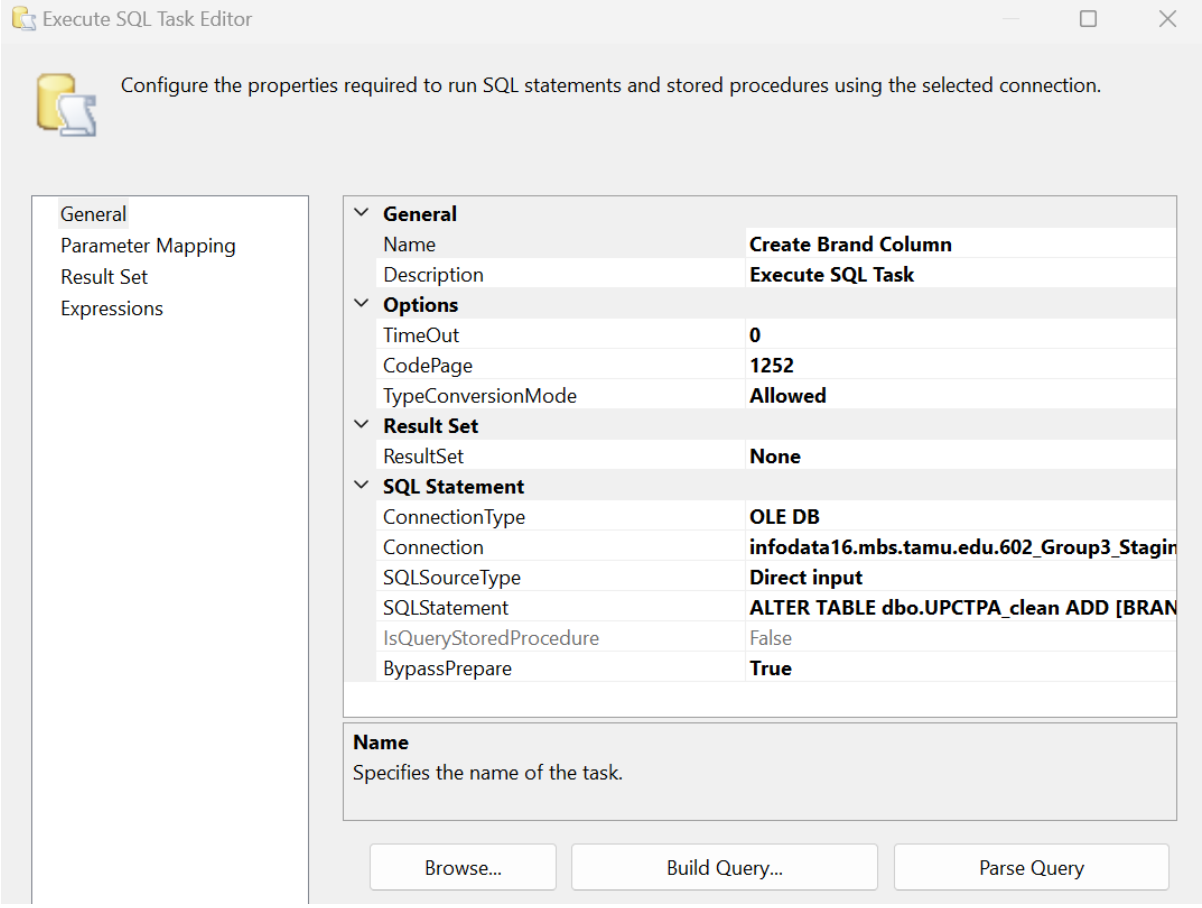


Fig: Update DB Connection details to staging database

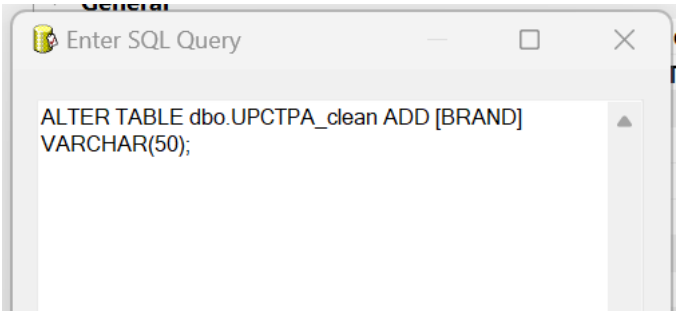


Fig: Build query to add a new column *BRAND*

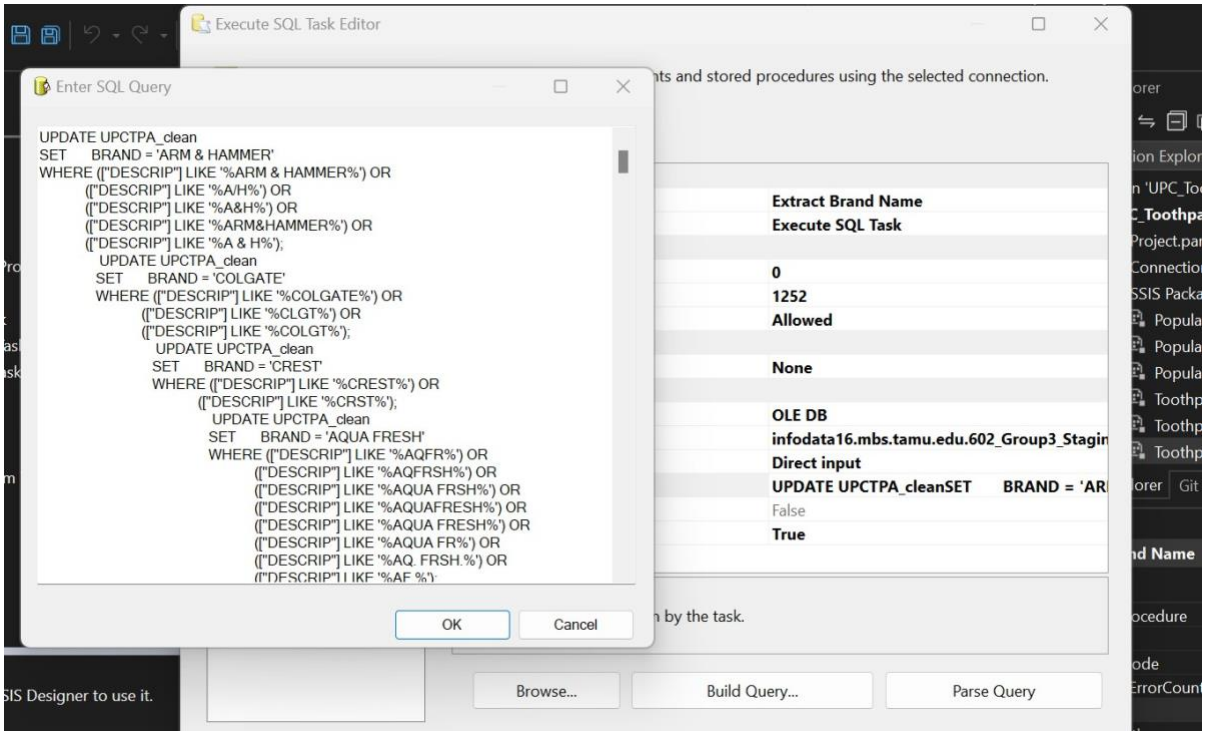


Fig: Extract the Brand name to the *BRAND* column

```
UPDATE UPCTPA_clean SET BRAND = 'ARM & HAMMER' WHERE (["DESCRIP"] LIKE '%ARM & HAMMER%') OR (["DESCRIP"] LIKE '%A/H%') OR (["DESCRIP"] LIKE
'%A&H%') OR (["DESCRIP"] LIKE '%ARM&HAMMER%') OR (["DESCRIP"] LIKE '%A & H%');
UPDATE UPCTPA_clean SET BRAND = 'COLGATE' WHERE (["DESCRIP"] LIKE '%COLGATE%') OR (["DESCRIP"] LIKE '%CLGT%');
UPDATE UPCTPA_clean SET BRAND = 'CREST' WHERE (["DESCRIP"] LIKE '%CREST%') OR (["DESCRIP"] LIKE '%CRST%');
UPDATE UPCTPA_clean SET BRAND = 'AQUA FRESH' WHERE (["DESCRIP"] LIKE '%AQFR%') OR (["DESCRIP"] LIKE '%AQFRSH%') OR (["DESCRIP"] LIKE '%AQUA FRSH%')
OR (["DESCRIP"] LIKE '%AQUAFRESH%') OR (["DESCRIP"] LIKE '%AQUA FRESH%') OR (["DESCRIP"] LIKE '%AQUA FR%') OR (["DESCRIP"] LIKE '%AQ. FRSH.%') OR
(["DESCRIP"] LIKE '%AF %');
UPDATE UPCTPA_clean SET BRAND = 'PLUS WHITE' WHERE (["DESCRIP"] LIKE '%PLUS+WHITE%') OR (["DESCRIP"] LIKE '%PLUS WHITE%') OR (["DESCRIP"] LIKE
'%PLUS-WHITE%');
UPDATE UPCTPA_clean SET BRAND = 'CLOSE UP' WHERE (["DESCRIP"] LIKE '%CLOSE UP%') OR (["DESCRIP"] LIKE '%CLOSE-UP%') OR (["DESCRIP"] LIKE '%CLS UP%')
OR (["DESCRIP"] LIKE '%CLSE-UP%') OR (["DESCRIP"] LIKE '%CLSUP%');
UPDATE UPCTPA_clean SET BRAND = 'AIM' WHERE (["DESCRIP"] LIKE '%AIM%');
UPDATE UPCTPA_clean SET BRAND = 'RAPID WHITE' WHERE (["DESCRIP"] LIKE '%RAPID WHITE%');
UPDATE UPCTPA_clean SET BRAND = 'DENTAGARD' WHERE (["DESCRIP"] LIKE '%DENTAGARD%');
UPDATE UPCTPA_clean SET BRAND = 'GLEEM' WHERE (["DESCRIP"] LIKE '%GLEEM%');
UPDATE UPCTPA_clean SET BRAND = 'MUPPETS' WHERE (["DESCRIP"] LIKE '%MUPPETS%');
```

- Execute the package

SQLQuery1.sql - inf...ICR5JRF\vr110 (54))

```
SELECT TOP (1000) ["COM_CODE"]
      ,["UPC"]
      ,["DESCRIP"]
      ,["SIZE"]
      ,["CASE"]
      ,["NITEM"]
      ,[BRAND]
  FROM [602_Group3_StagingDB_F24].[dbo].[UPCTPA_clean]
```

100 %

Results | Messages

	"COM_CODE"	"UPC"	"DESCRIP"	"SIZE"	"CASE"	"NITEM"	BRAND
1	956	1032718330	"ARM & HAMMER DENTAL"	"3 OZ"	36	7808111	ARM & HAMMER
2	956	1032718350	"A/H DENTAL CARE TOOT"	"5 OZ"	12	6011271	ARM & HAMMER
3	956	1032718370	"ARM & HAMMER DENTAL"	"7 OZ"	12	6011281	ARM & HAMMER
4	956	1111304010	"CLOSE UP RED GEL W/4"	"8.2 OZ"	12	6014951	CLOSE UP
5	956	1111307430	"MNTDNT TARTR TP W/FR"	"5.2 OZ"	12	6015871	MENTADENT
6	956	1111307440	"MENTADENT TP W/FREE"	"3.5 OZ"	12	6015891	MENTADENT
7	956	1111307450	"MNTDNT FRSH TP W/FRE"	"5.2 OZ"	12	6015811	MENTADENT
8	956	1111307460	"MNTDNT COOL TP W/FRE"	"5.2 OZ"	12	6015831	MENTADENT
9	956	1111310720	"MENTADENT REFILL TWN"	"10.4 O"	12	6015941	MENTADENT
10	956	1111311740	"PEPSODENT 28% FREE"	"8.2 OZ"	12	6014501	PEPSODENT
11	956	1111315120	"PEPSODENT ANTI TARTA"	"6 OZ"	12	6014521	PEPSODENT

Fig: New column *BRAND* is created and populated in the *dbo.UPCTPA_clean* table

Data Granularity

Week-Level Analysis:

Data at the week level is crucial for identifying store traffic in the last quarter of 1994. By looking at the weekly transaction counts for October, November, and December, we can find the store with the highest customer visits . Similarly examining weekly sales trends is used for finding the highest grocery sales during holiday weeks in 1991 and 1992 for low-tier Buffalo Grove stores.

Product-Level Analysis

To assess profit margins for toothpaste by brand we need to store data product-level granularity. By storing detailed product attributes like selling price, brand, and cost, we can calculate and analyze the profit margins across various toothpaste brands.

To find the sales contributions of CORONA EXTRA LIGHT during Thanksgiving week of 1993, product-level data is required.

Date-Level Analysis

Analyzing the monthly average sales for the Bakery, Dairy, Pharmacy, Cosmetic, and HABA departments in Naperville and Schaumburg during 1994 needs collecting daily sales data into monthly averages. This allows us to find sales trends.

Data Loading

Creating Dimension Tables:

1. Date_Dim

```
CREATE TABLE Date_Dim(
        DateKey INT IDENTITY(1,1) PRIMARY KEY,
        DATE DATE,
        YEAR INT,
        MONTH INT,
        DAY INT,
        WEEK INT,
        QUARTER INT
)
```

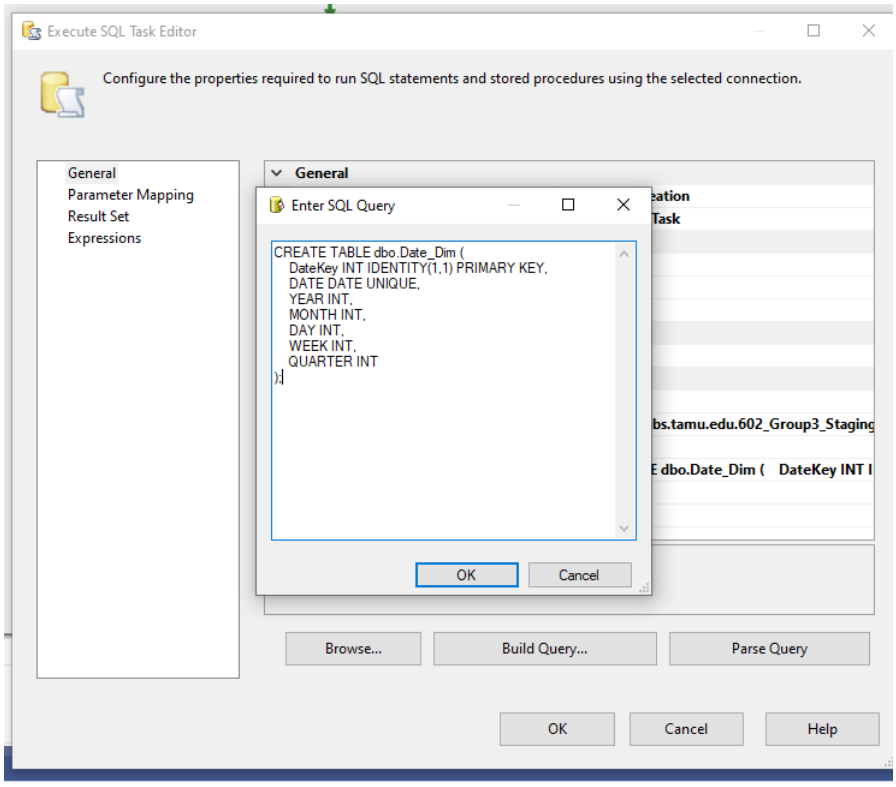


Figure: Date_Dim Creation

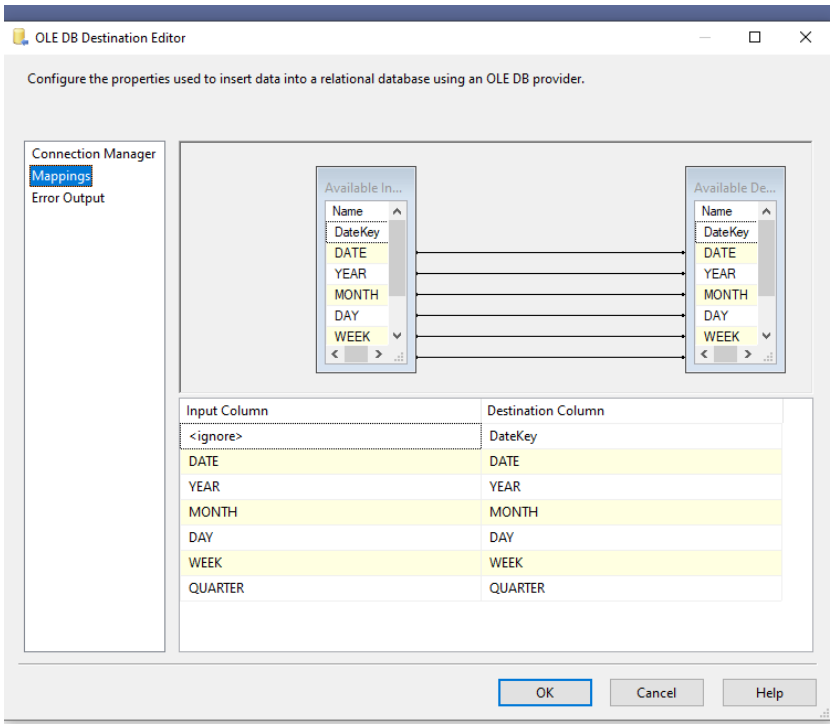


Figure: Mapping the columns for Date_Dim

2. Store_Dim

```
CREATE TABLE Store_Dim (
        Store_Key INT IDENTITY(1,1) PRIMARY KEY,
        Store_ID INT NOT NULL
);
```

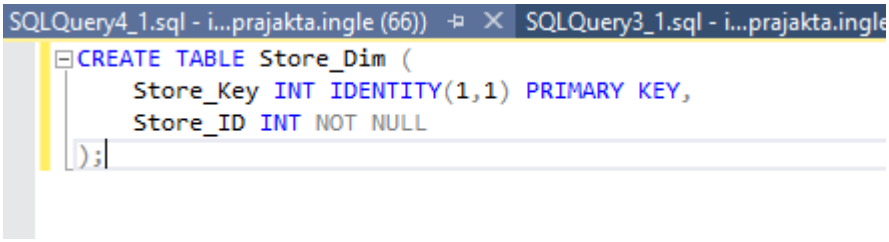


Figure: Store_Dim Creation

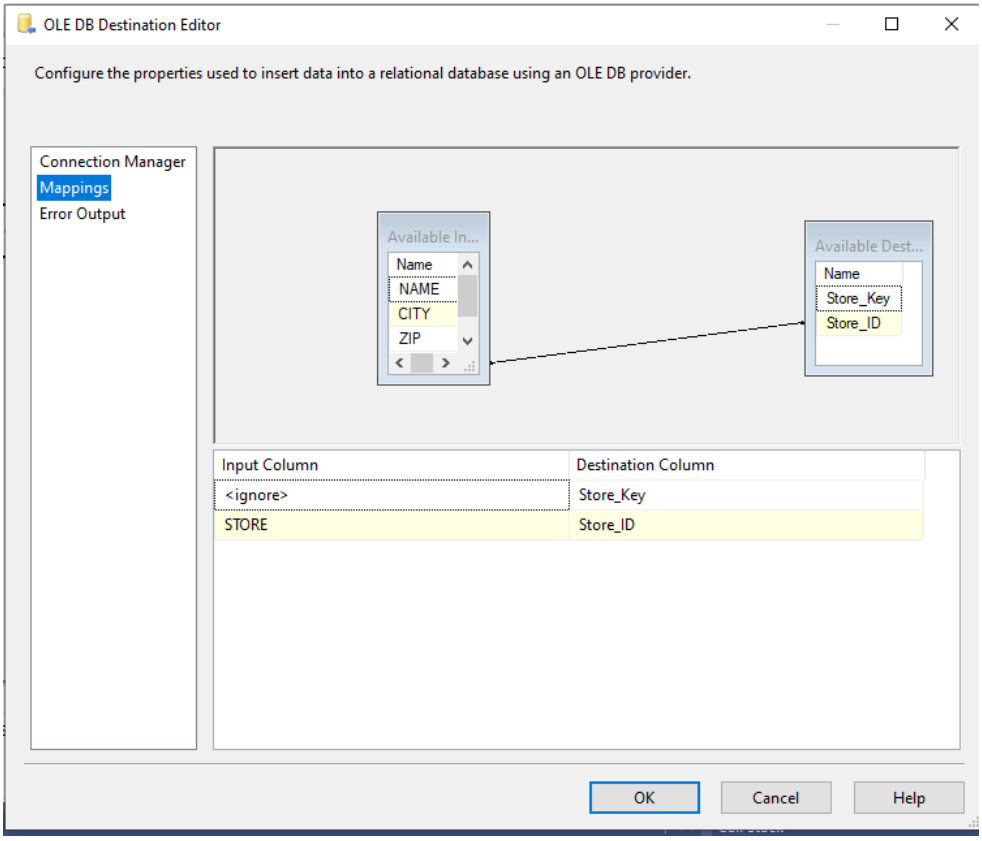


Figure: Mapping the columns for Store_Dim

3. Demography_Dimension:

```
CREATE TABLE [dbo].[Demography_Dim_BQ5] (
    Demography_Key INT IDENTITY(1,1) PRIMARY KEY,
    Store INT,
    Single FLOAT,
    Retired FLOAT,
    UNIQUE (Store)
);
```

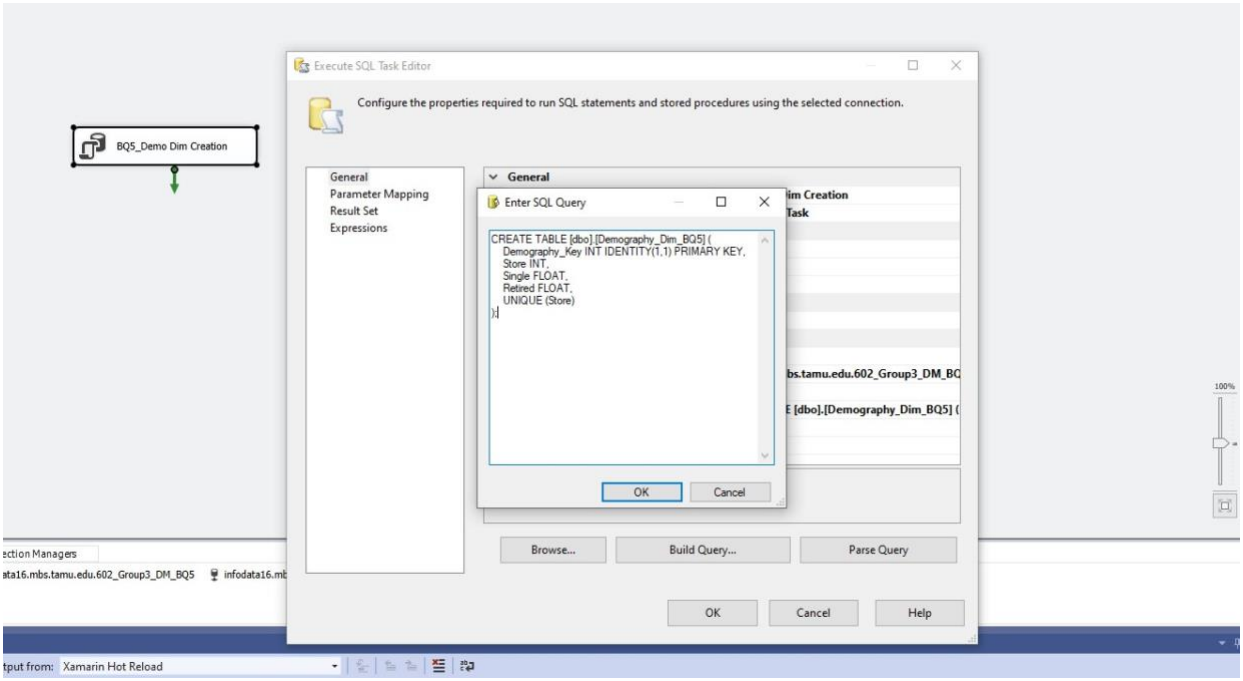


Fig Creation of Demographic Dimension Table

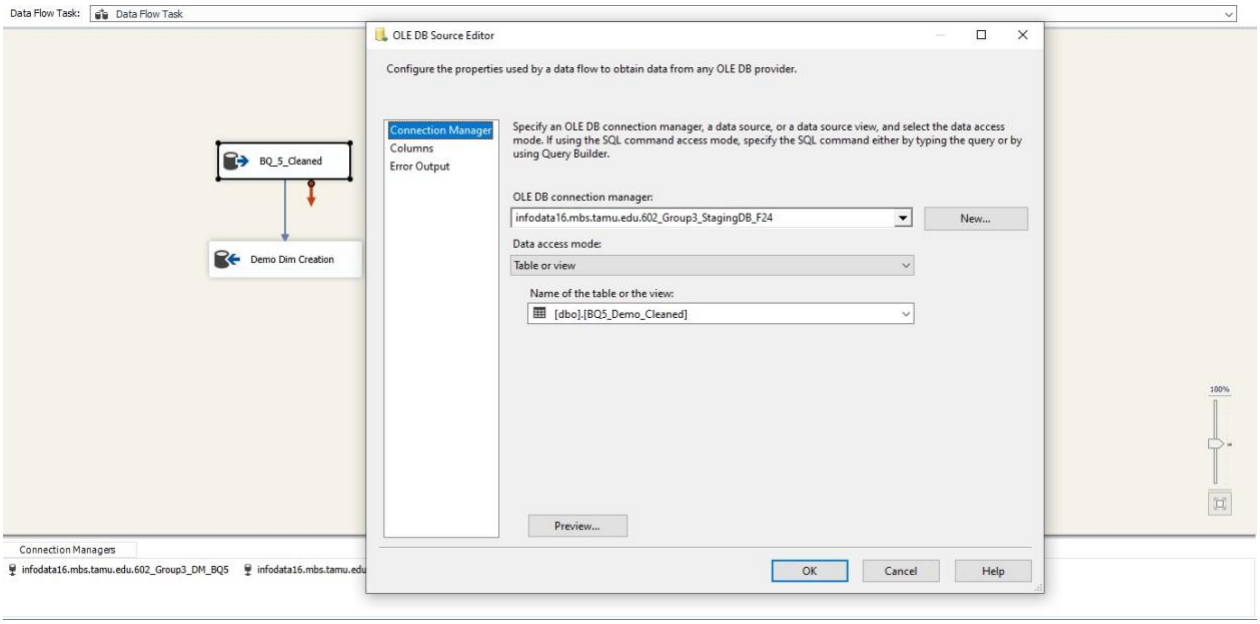


Fig Selecting Source table

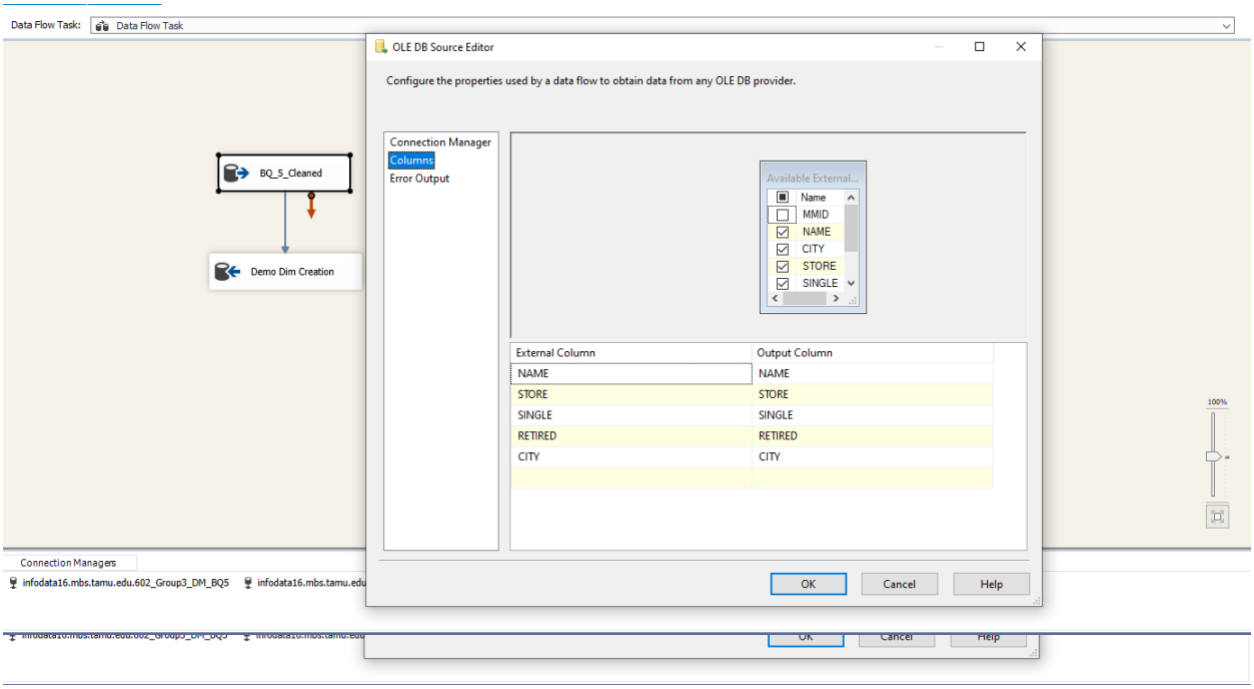


Fig Mapping the required Columns from Source table

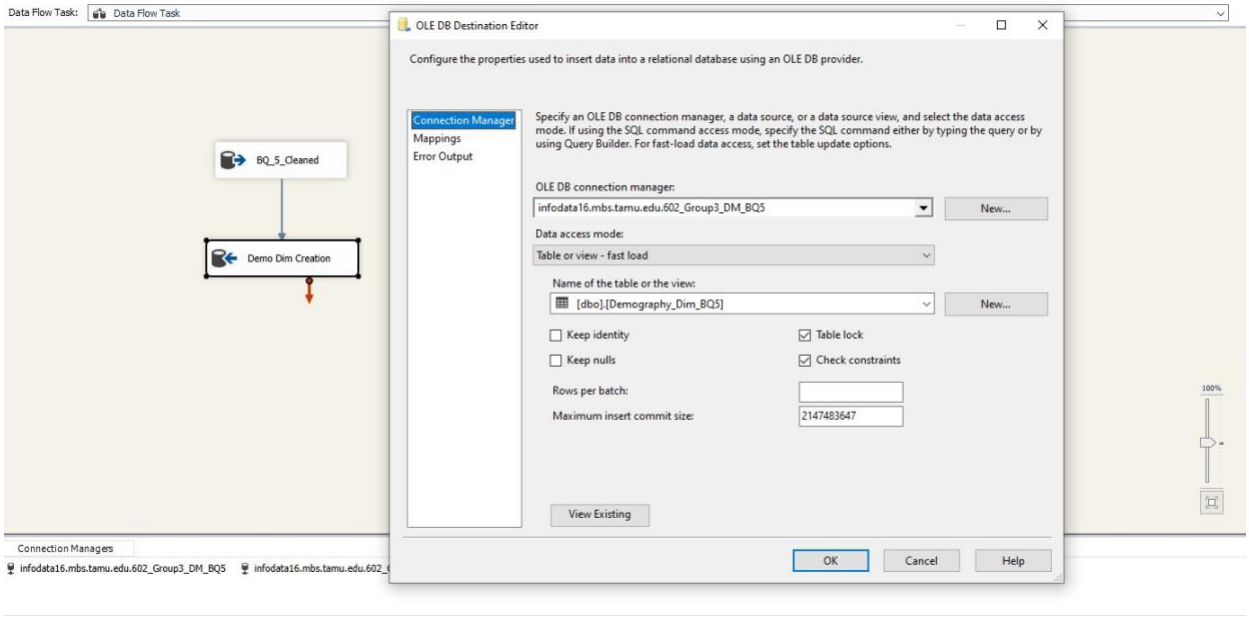


Fig Selecting the Destination dimension table

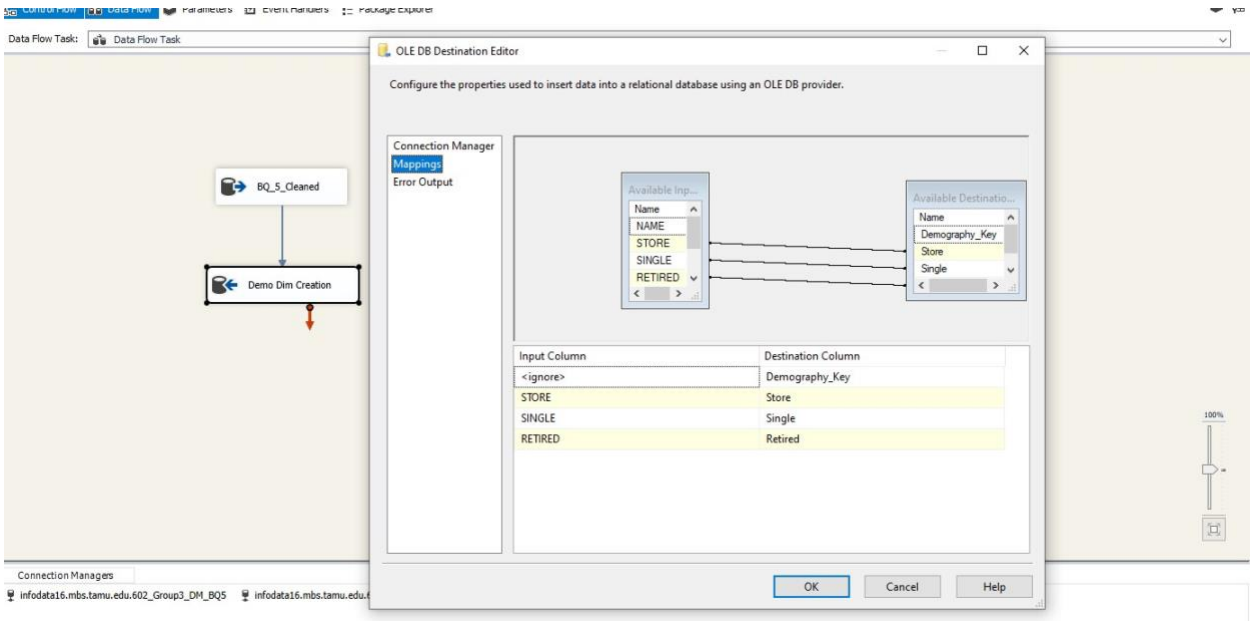


Fig Mapping the Columns in Destination Dimension table

4. Product_Dim_BQ5:

```
CREATE TABLE [dbo].[Product_Dim_BQ5] (
    Product_Key INT IDENTITY(1,1) PRIMARY KEY,
    UPC INT,
    Product_Name VARCHAR(64),
    UNIQUE (UPC)
);
```

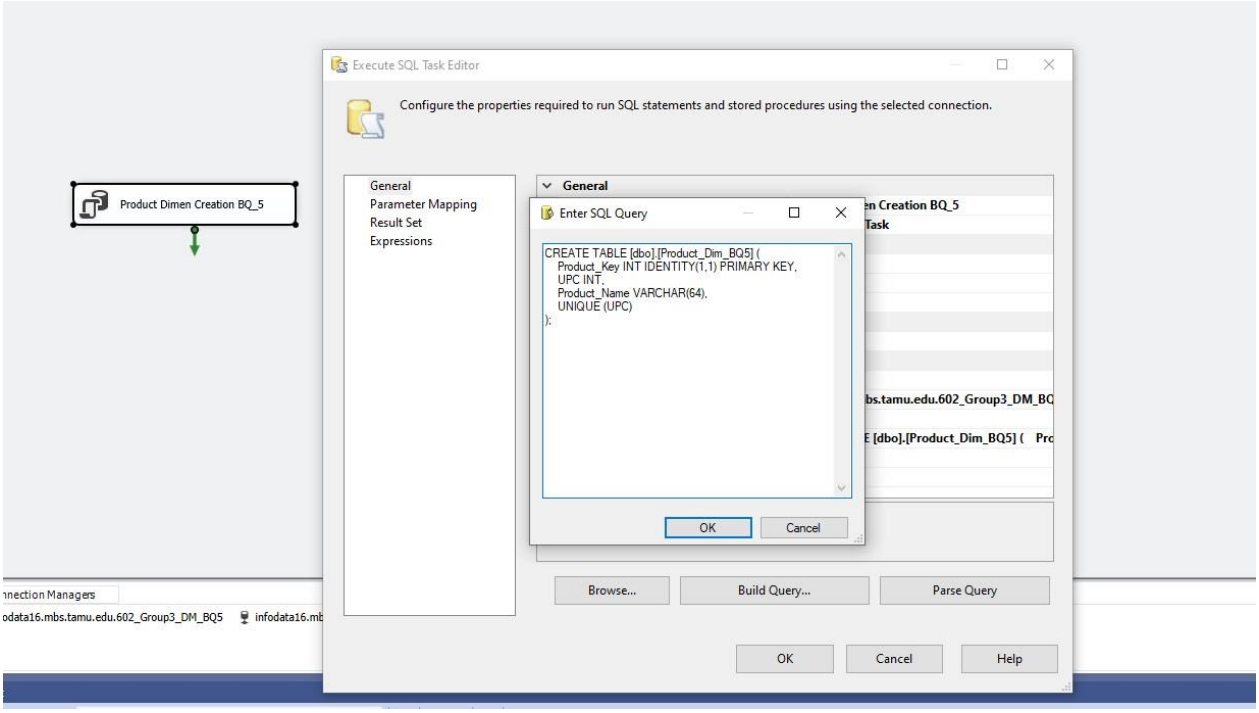


Fig Creating the Product Dimension table

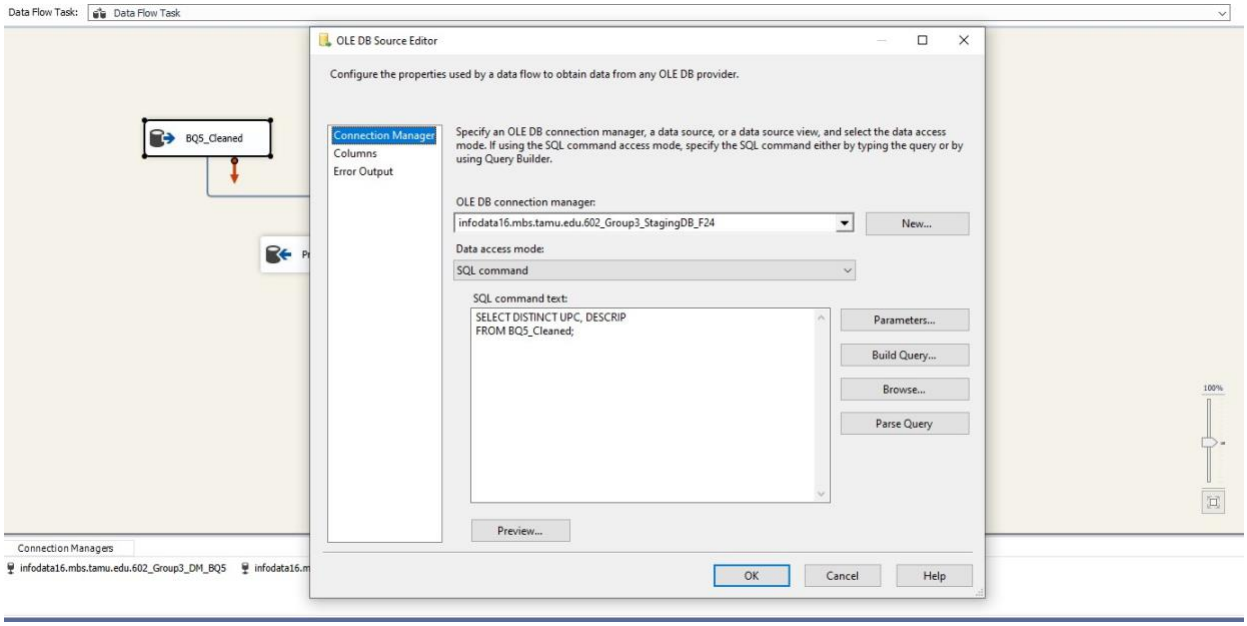


Fig Loading Data from BQ5_Cleaned as Source

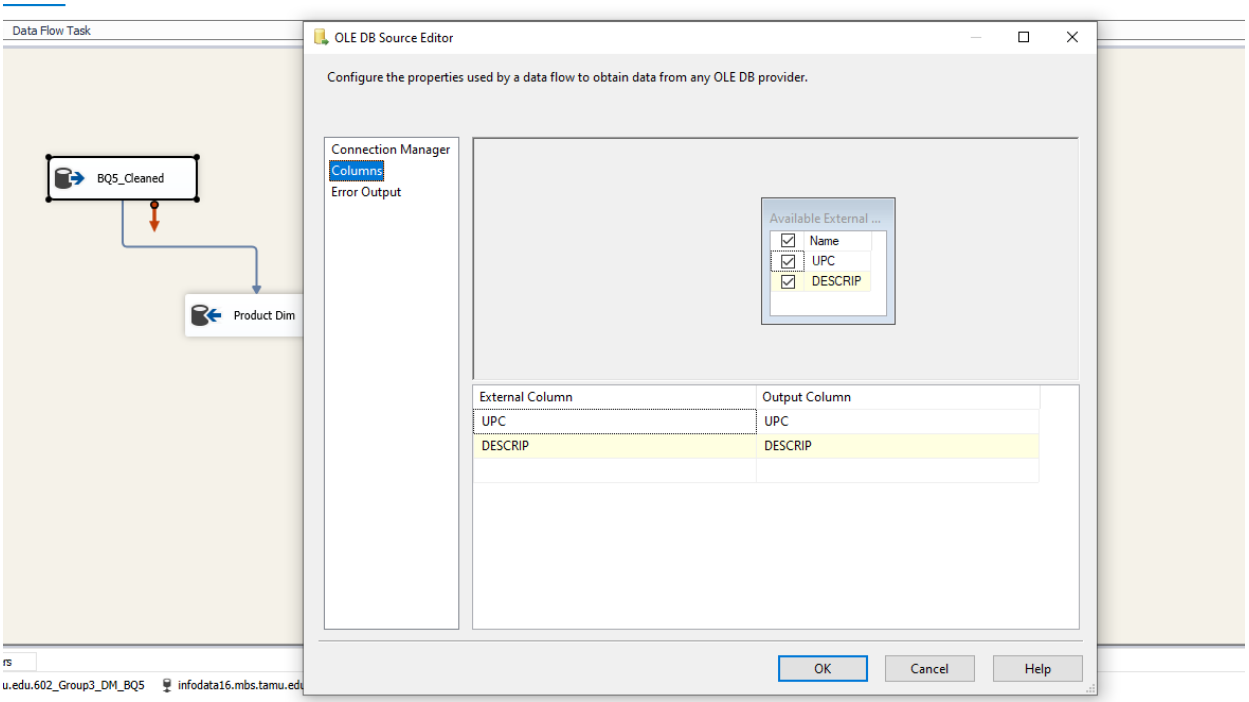


Fig Selecting the Columns from the Source table

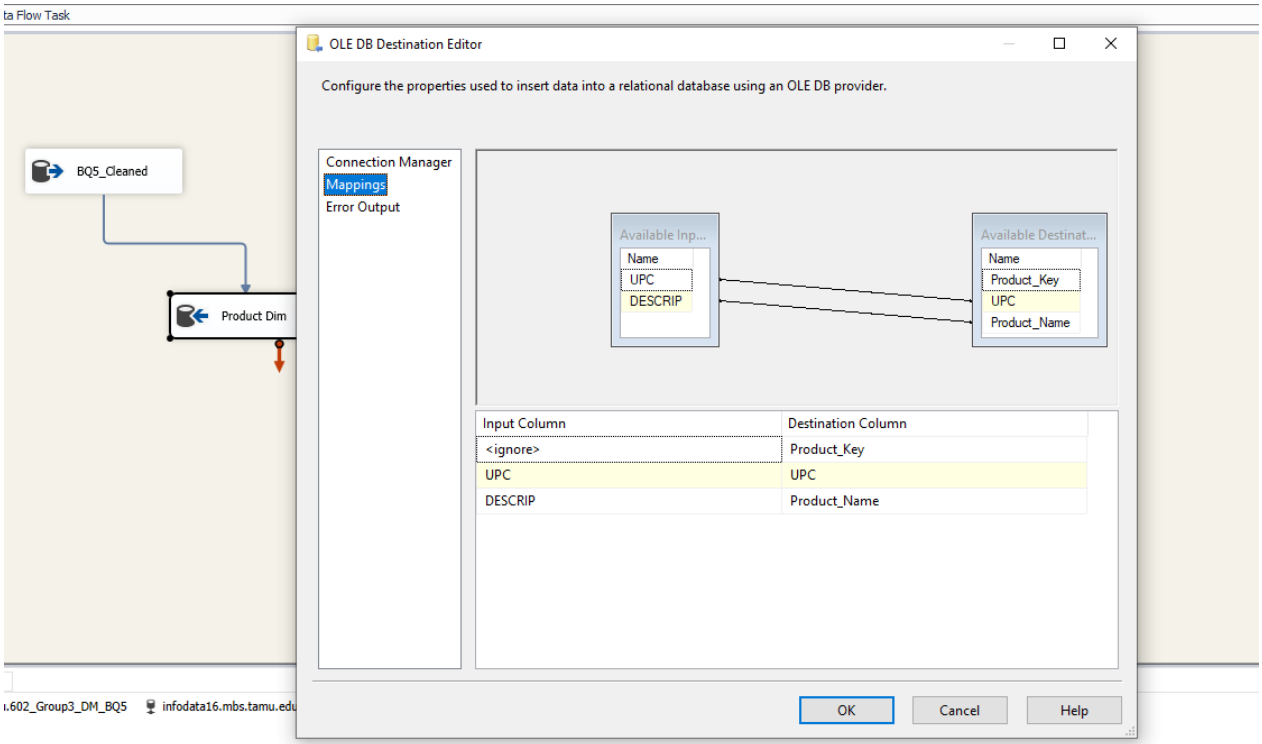


Fig Mapping the required Columns for Dimension table
Fig Loading Data into Product Dimension as Destination

5. Brand_Dimension

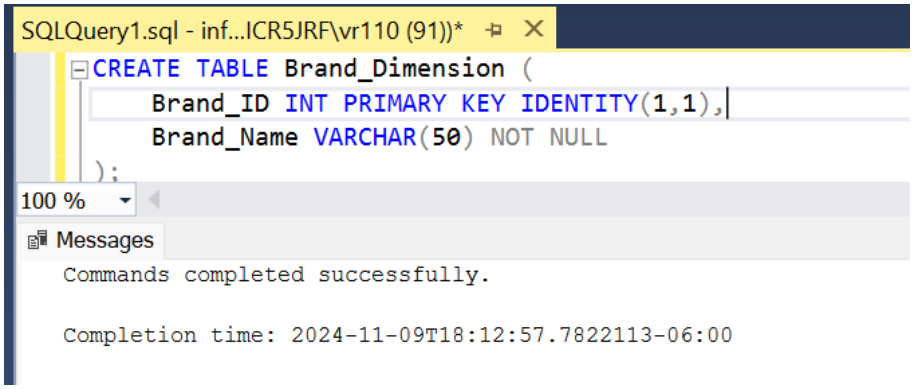


Fig: Create Brand_Dimension table

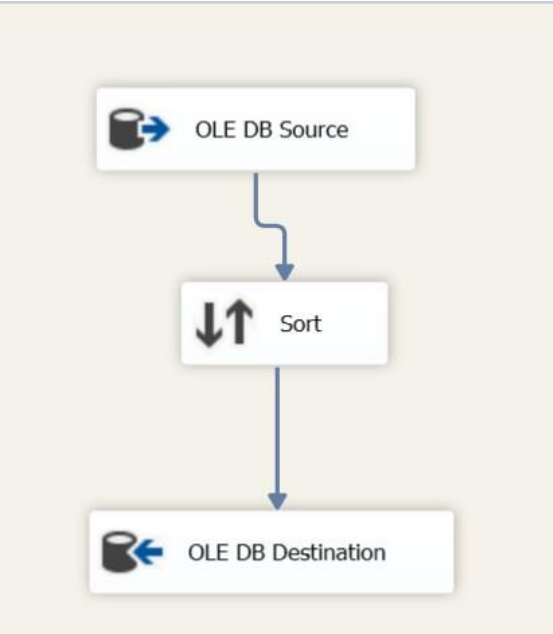


Fig: Data Flow for loading Brand_Dimesion Table

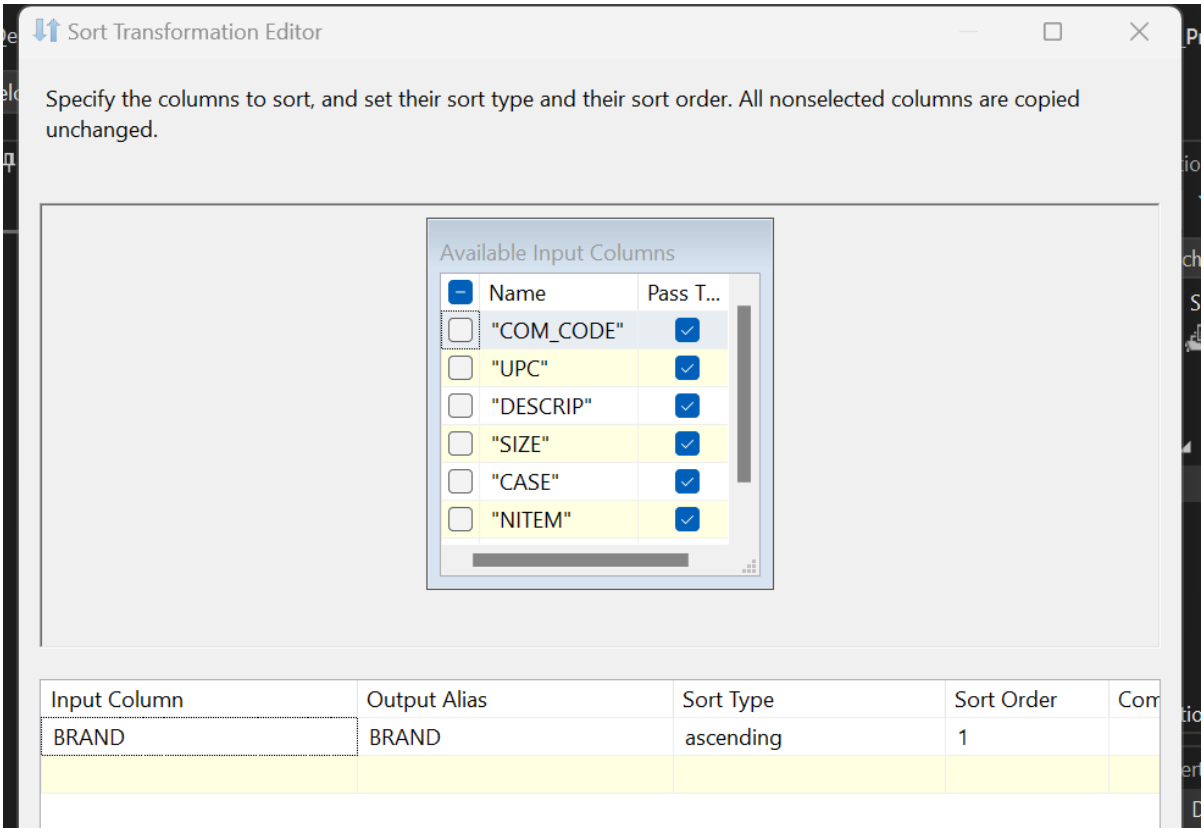


Fig: Remove duplicate brand names using sort

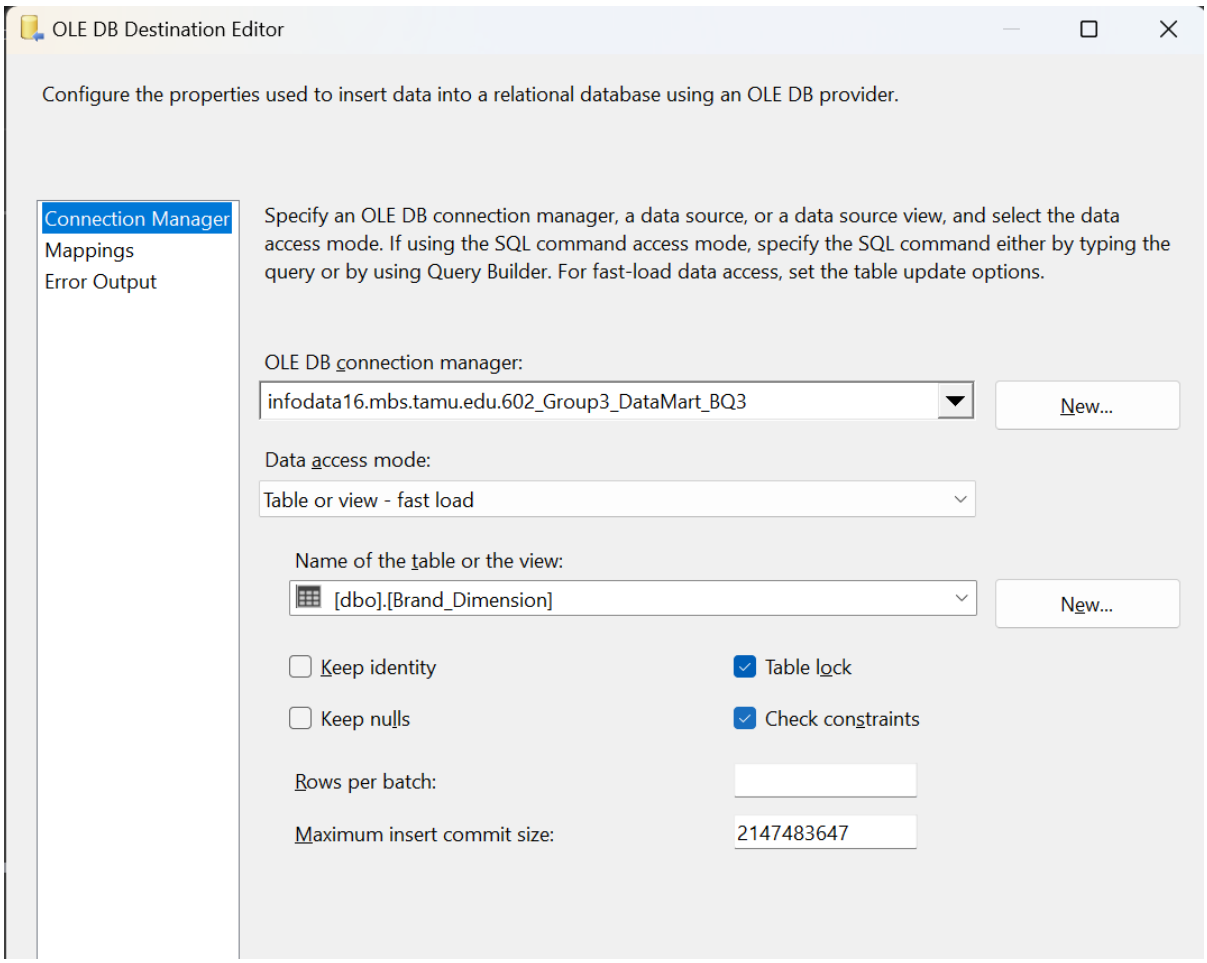


Fig: Load the cleaned data to Brand_Dimension Table

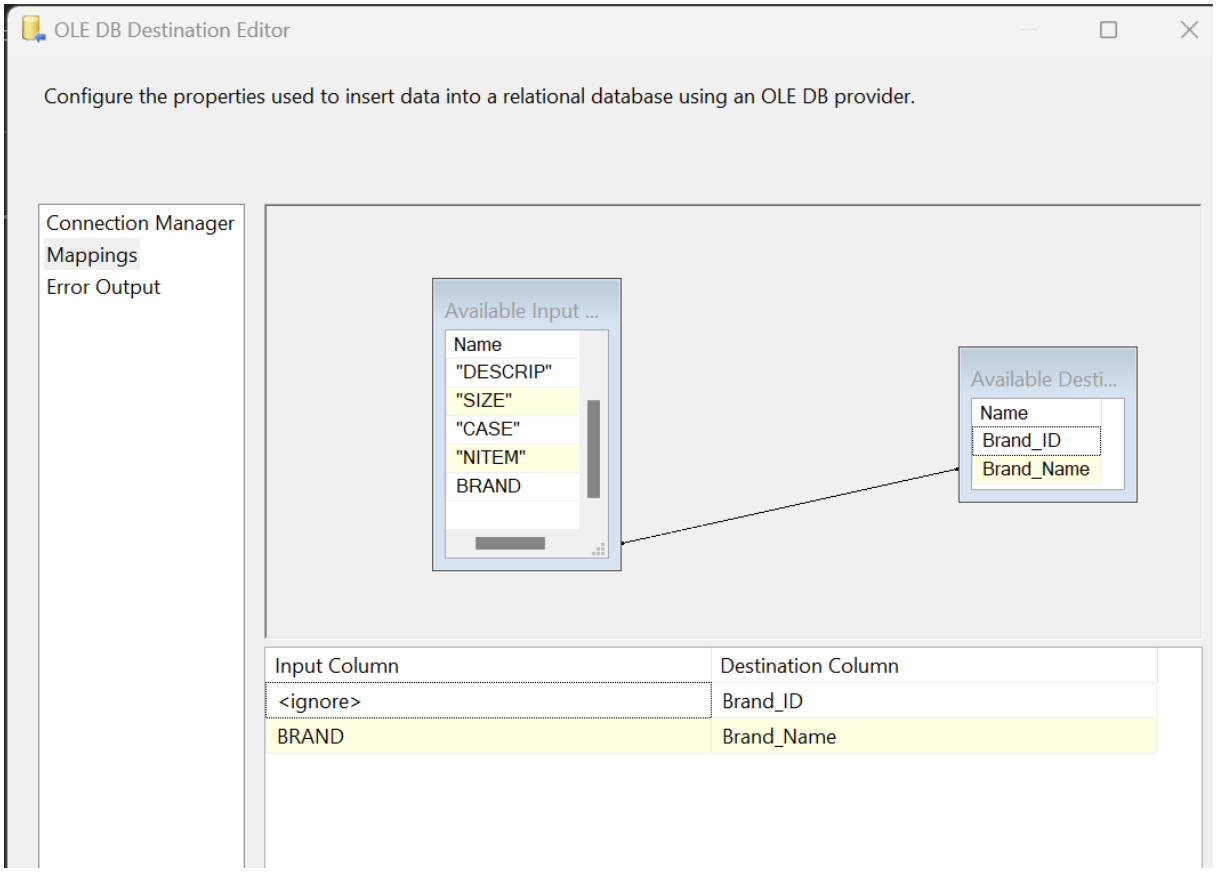
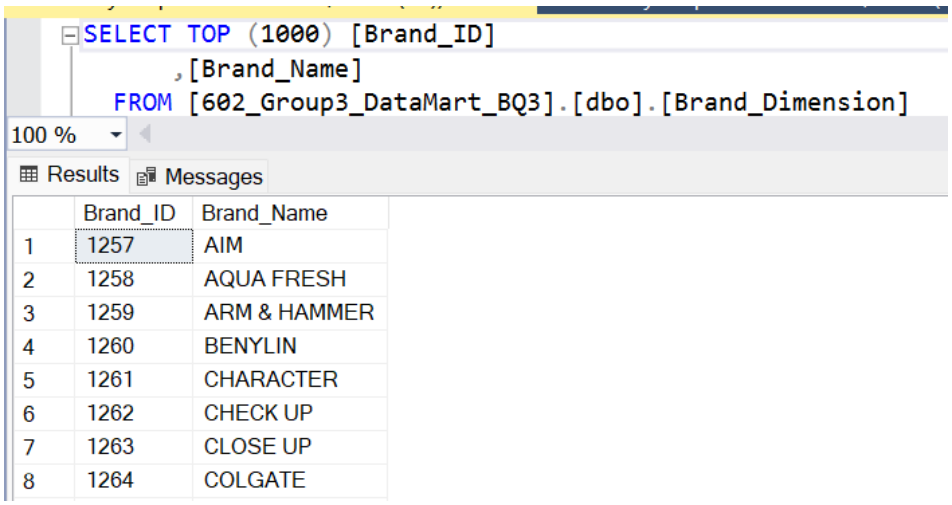


Fig: Brand_Dimension table mapping from dbo.UPCTPA_clean staging table



```
SELECT TOP (1000) [Brand_ID]
      ,[Brand_Name]
  FROM [602_Group3_DataMart_BQ3].[dbo].[Brand_Dimension]
```

100 %

Results | Messages

	Brand_ID	Brand_Name
1	1257	AIM
2	1258	AQUA FRESH
3	1259	ARM & HAMMER
4	1260	BENYLIN
5	1261	CHARACTER
6	1262	CHECK UP
7	1263	CLOSE UP
8	1264	COLGATE

Fig: Execute the package to load data to Brand_Dimension Table

6. Product_Dimension

```
CREATE TABLE Product_Dimension (
    UPC INT PRIMARY KEY,
    Brand_ID INT FOREIGN KEY REFERENCES Brand_Dimension(Brand_ID),
    Product_Name VARCHAR(100),
    Size VARCHAR(20)
);
```

Fig: Create Product_Dimension Table

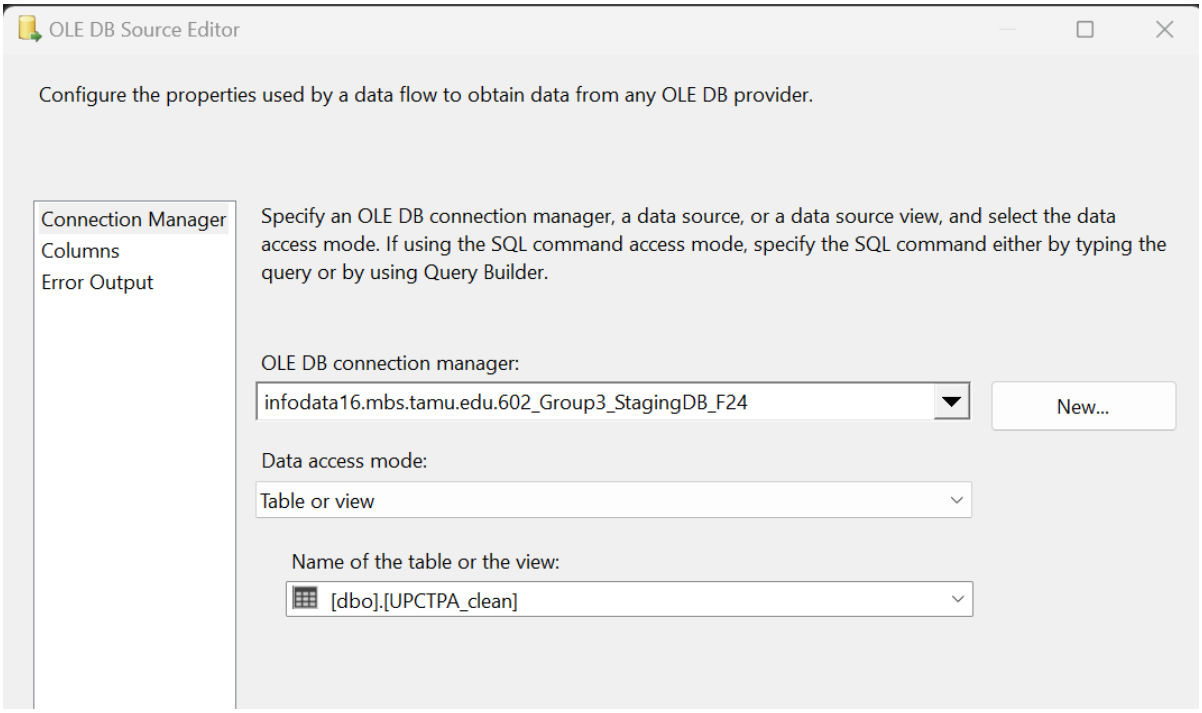


Fig: Configure dbo.UPCTPA_done as OLE DB source

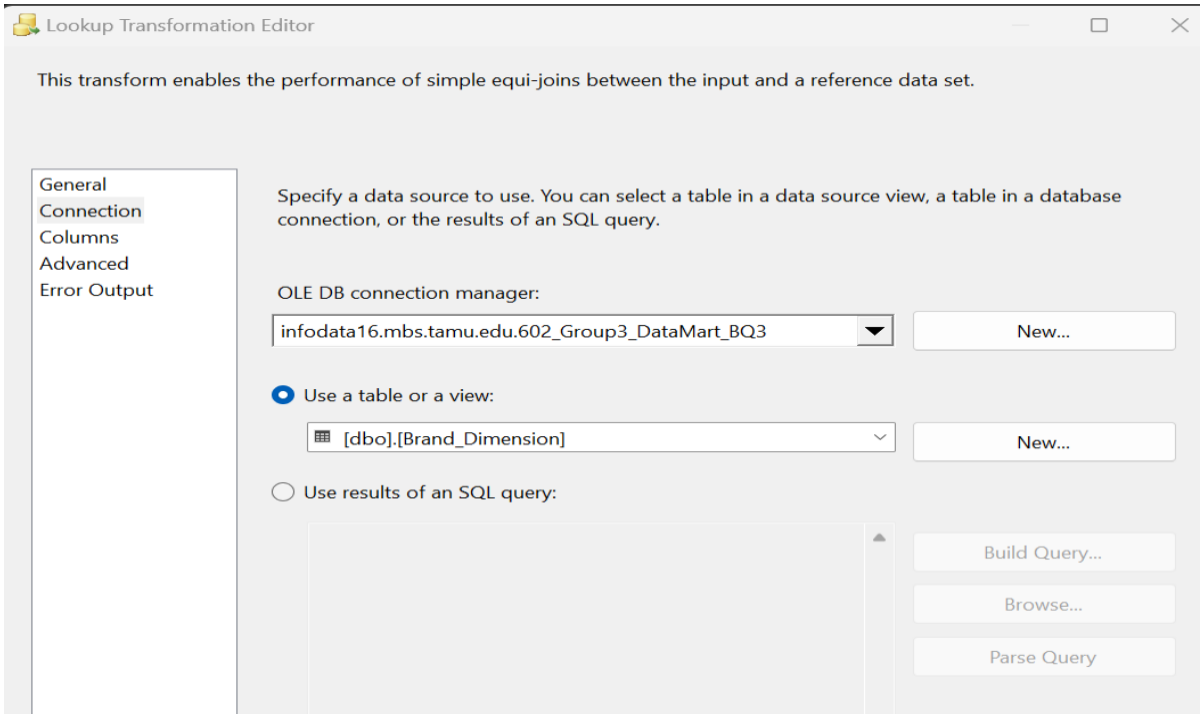


Fig: Use Lookup to map Brand_ID from Brand_Dimension to Product_Dimension

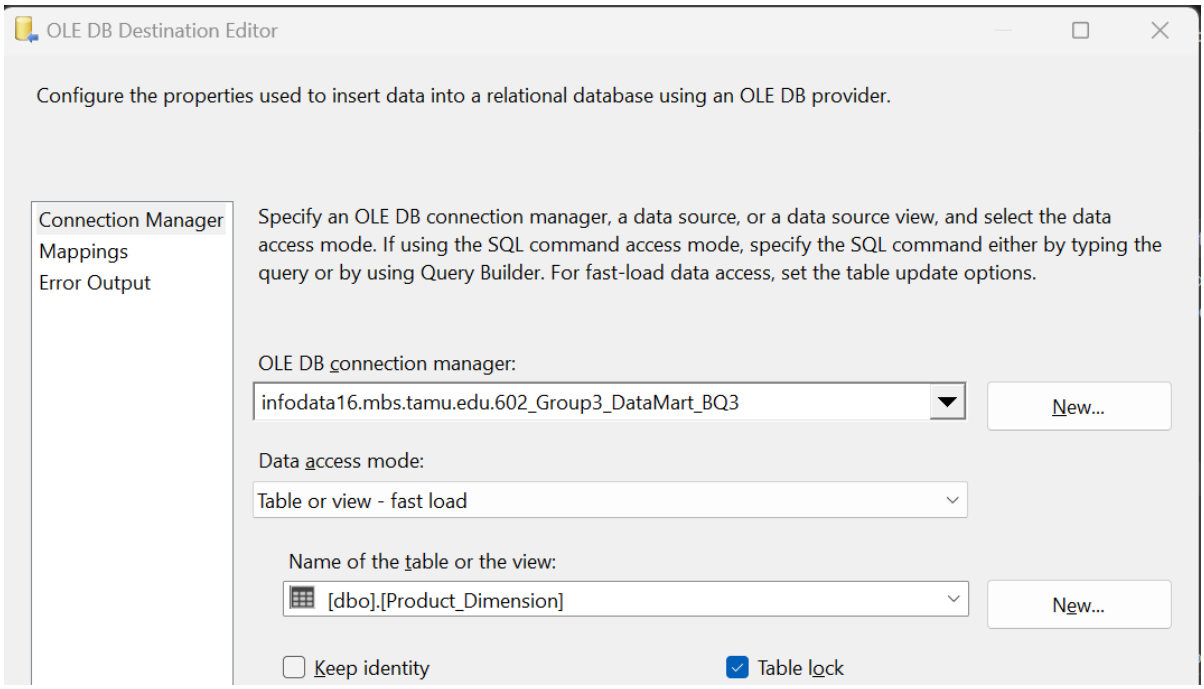


Fig: Configure Product_Dimension as destination in OLE DB Destination

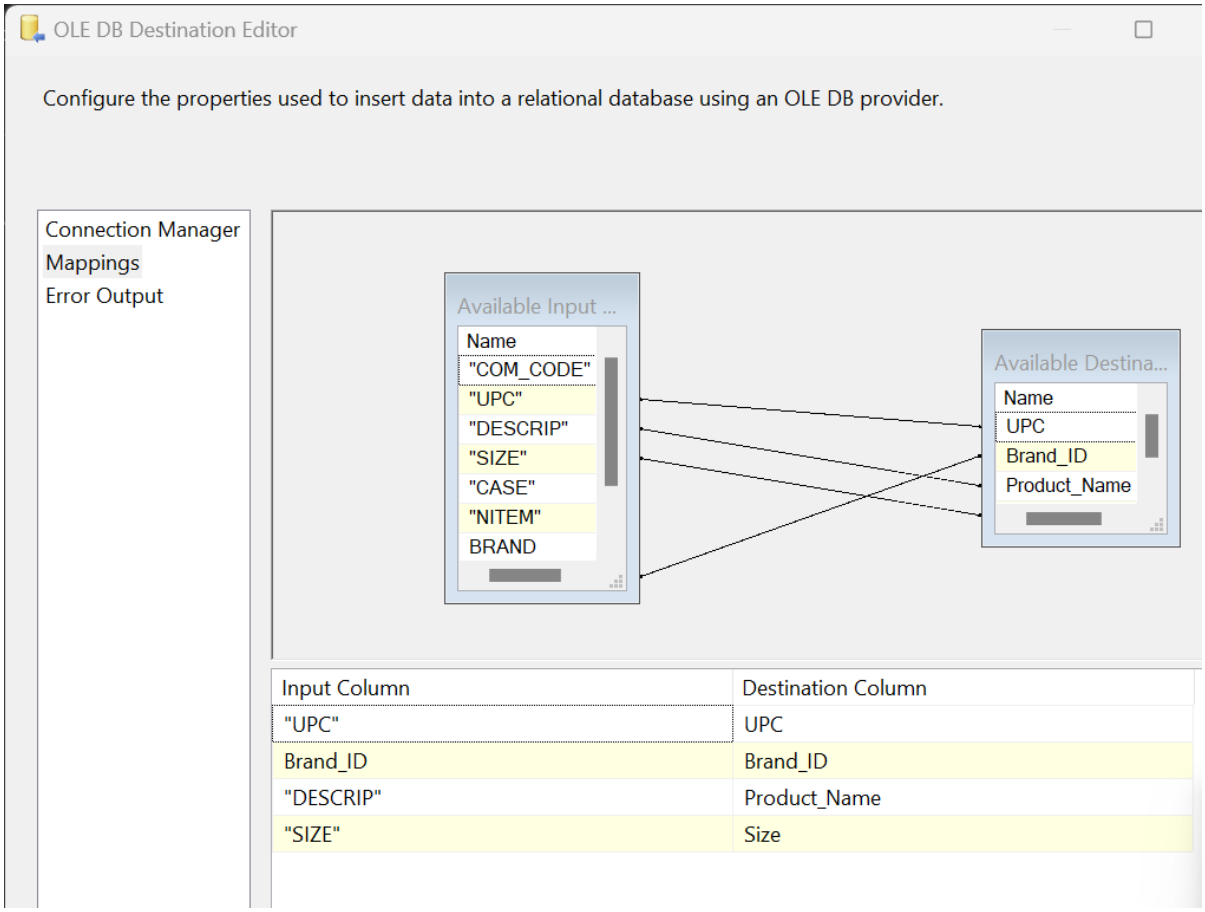


Fig: Mapping the data to Product_Dimension table

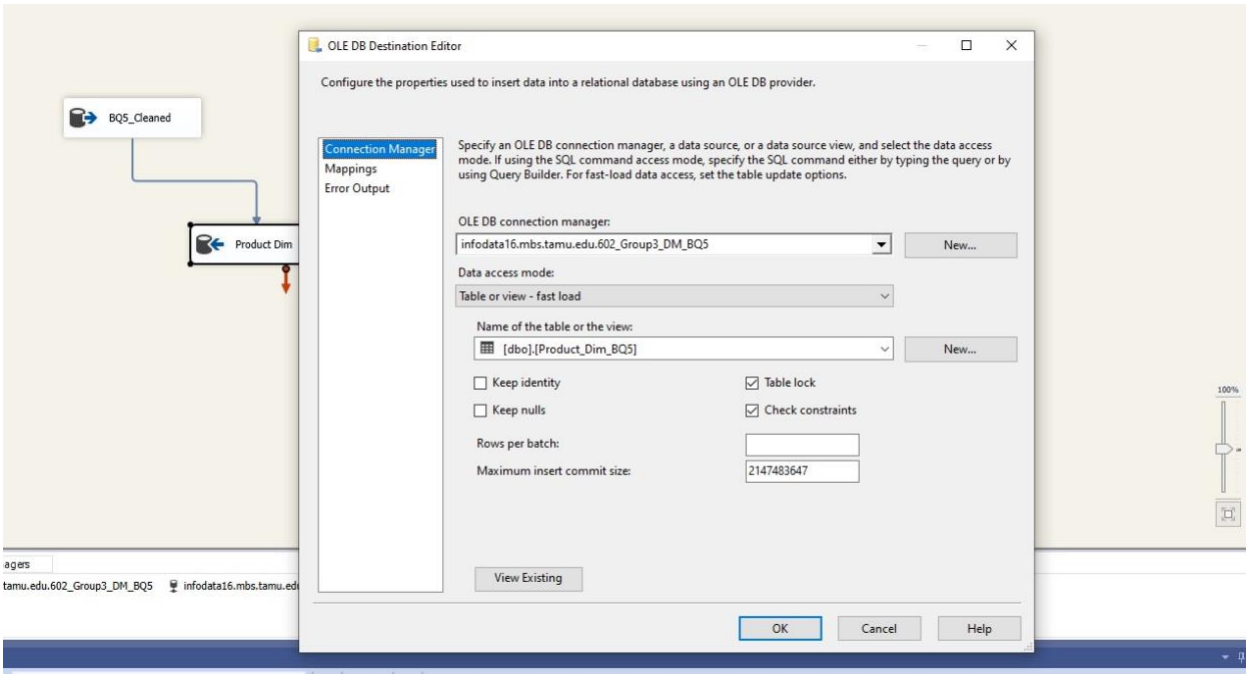
```
SELECT TOP (1000) [UPC]
      ,[Brand_ID]
      ,[Product_Name]
      ,[Size]
  FROM [602_Group3_DataMart_BQ3].[dbo].[Product_Dimension]
```

100 %

Results | Messages

	UPC	Brand_ID	Product_Name	Size
1	1032718330	1259	"ARM & HAMMER DENTAL"	"3 OZ"
2	1032718350	1259	"A/H DENTAL CARE TOOT"	"5 OZ"
3	1032718370	1259	"ARM & HAMMER DENTAL"	"7 OZ"
4	1111304010	1263	"CLOSE UP RED GEL W/4"	"8.2 OZ"
5	1111307430	1274	"MNTDNT TARTR TP W/FR"	"5.2 OZ"
6	1111307440	1274	"MENTADENT TP W/FREE"	"3.5 OZ"
7	1111307450	1274	"MNTDNT FRSH TP W/FRE"	"5.2 OZ"
8	1111307460	1274	"MNTDNT COOL TP W/FRE"	"5.2 OZ"
9	1111310720	1274	"MENTADENT REFILL TWN"	"10.4 O"

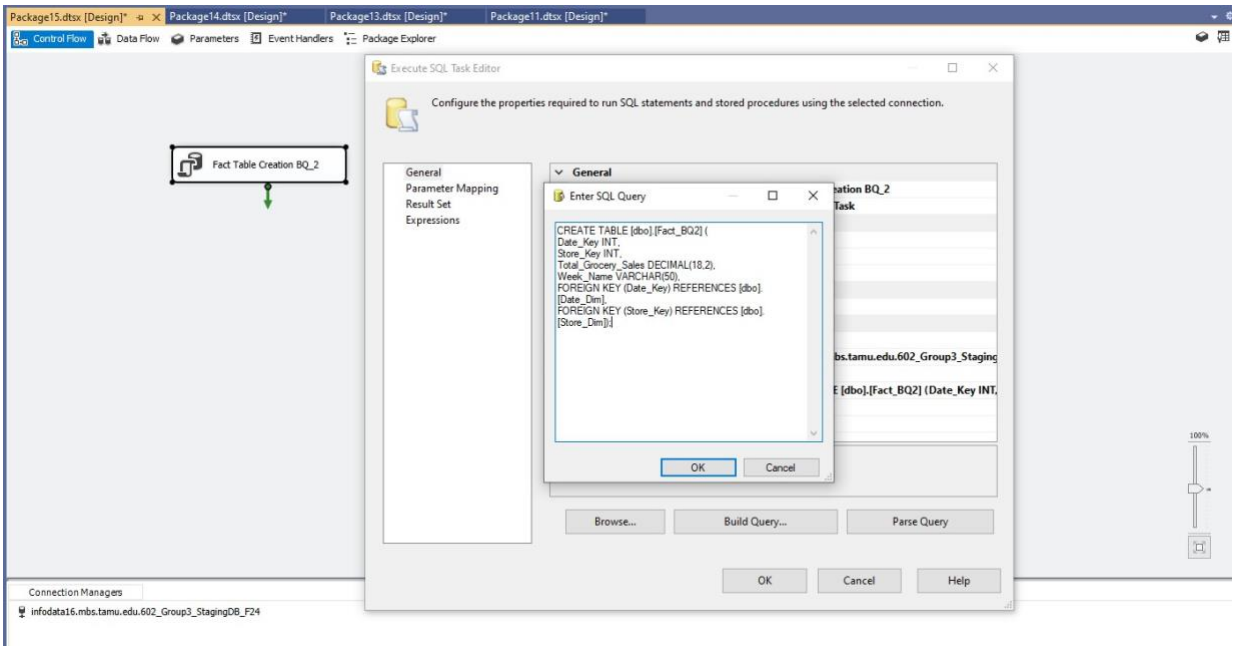
Fig: Execute the package to load data to Product_dimension table



Creating Fact Tables:

1. Fact_BQ2

We have created a new package in SSIS and used Execute SQL to create fact table called **"Fact_BQ2"**



Fact table loading:

- Created a new SSIS Package.
- Created OLEDB Source, 2 Lookups and OLEDB Destination and connected them.
- We have connected OLE DB Source to staging and fetching the Fact_BQ2 data table.
- In lookup 1, we connected to the time dimension of data mart and if the date value matches between staging table column DATE and Date column of Time dimension, then fetching Day_Key.
- In lookup 2, we connected to the store dimension of data mart and if store value matches between staging table column STORE and Store_Id column of store dimension , then fetching Store_Key.
- In the OLEDB destination, we made a connection to data mart and mapped respective columns from staging table, dimensions to fact table.

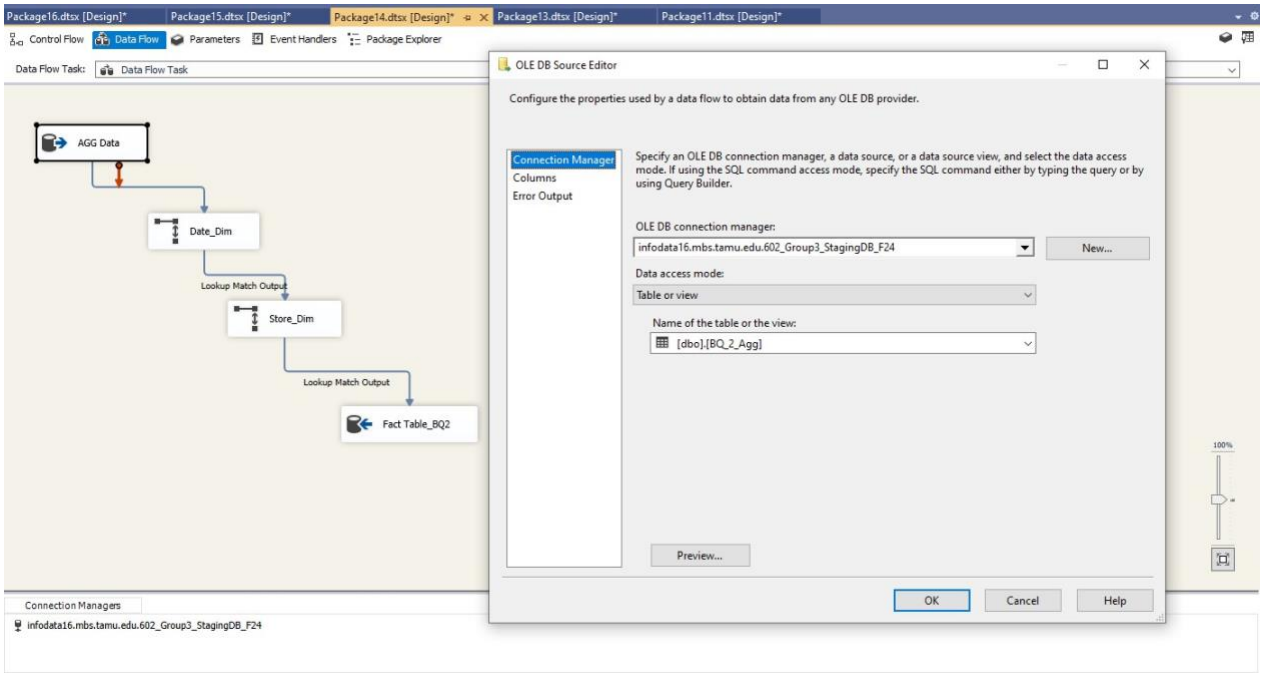


Fig Selecting Source data

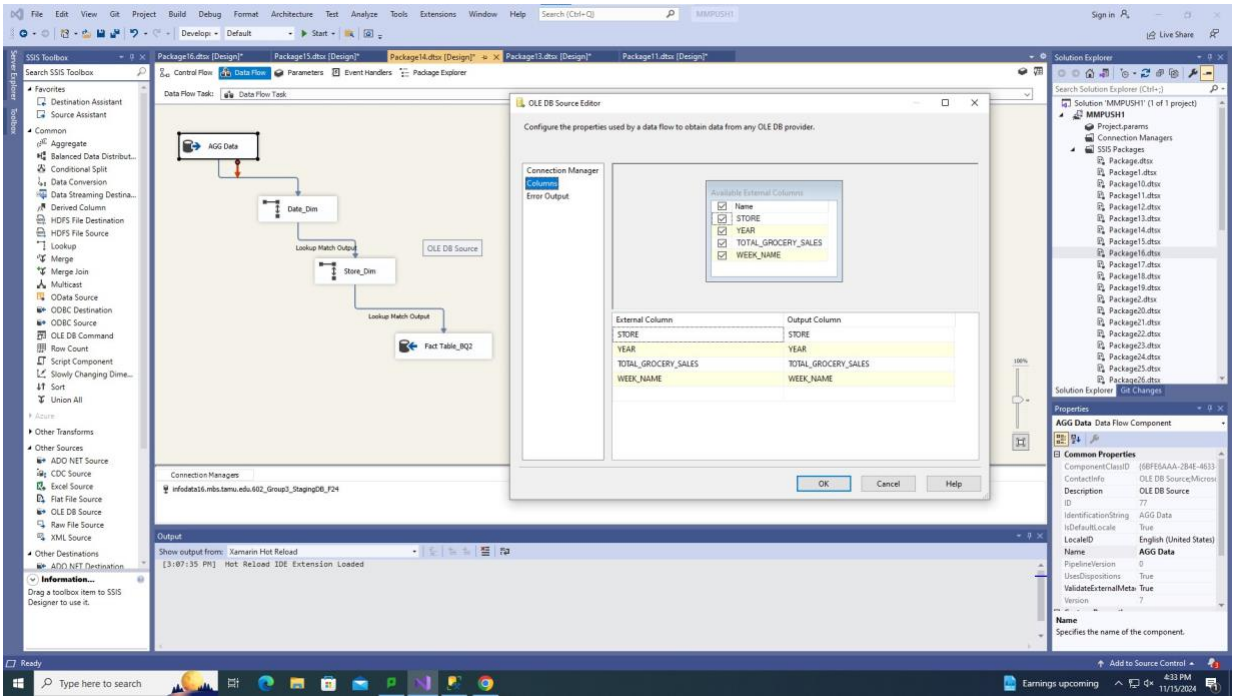


Fig Selecting required Columns from Source table

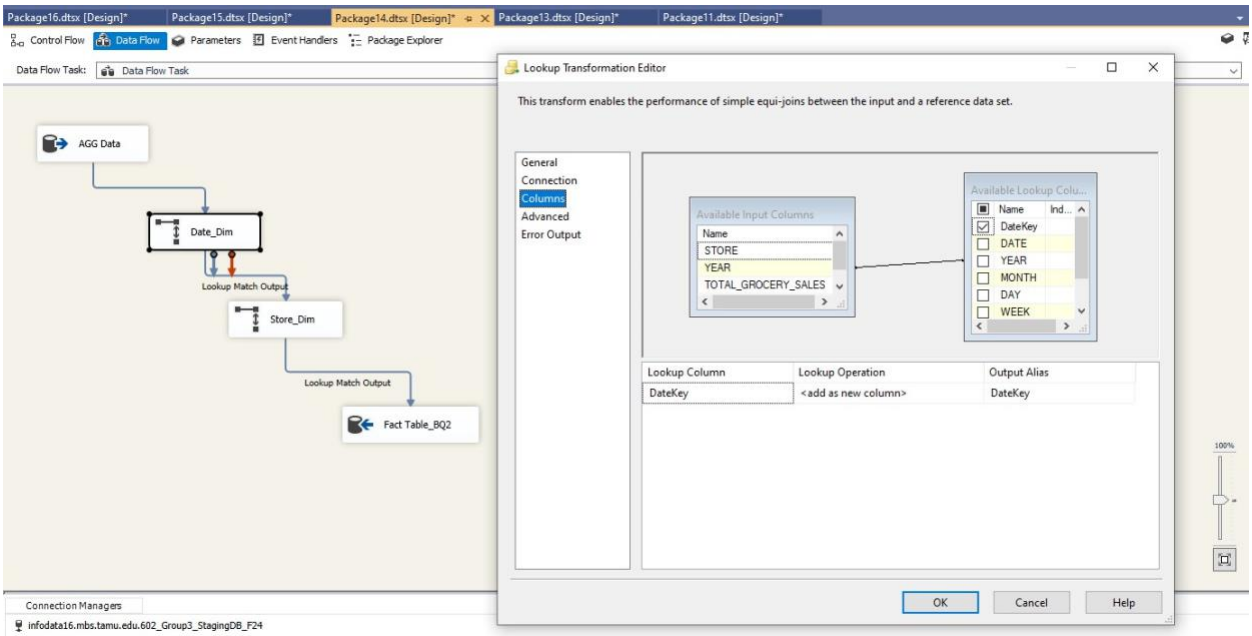


Fig Lookup 1 to - Mapping columns from Date Dimension Table

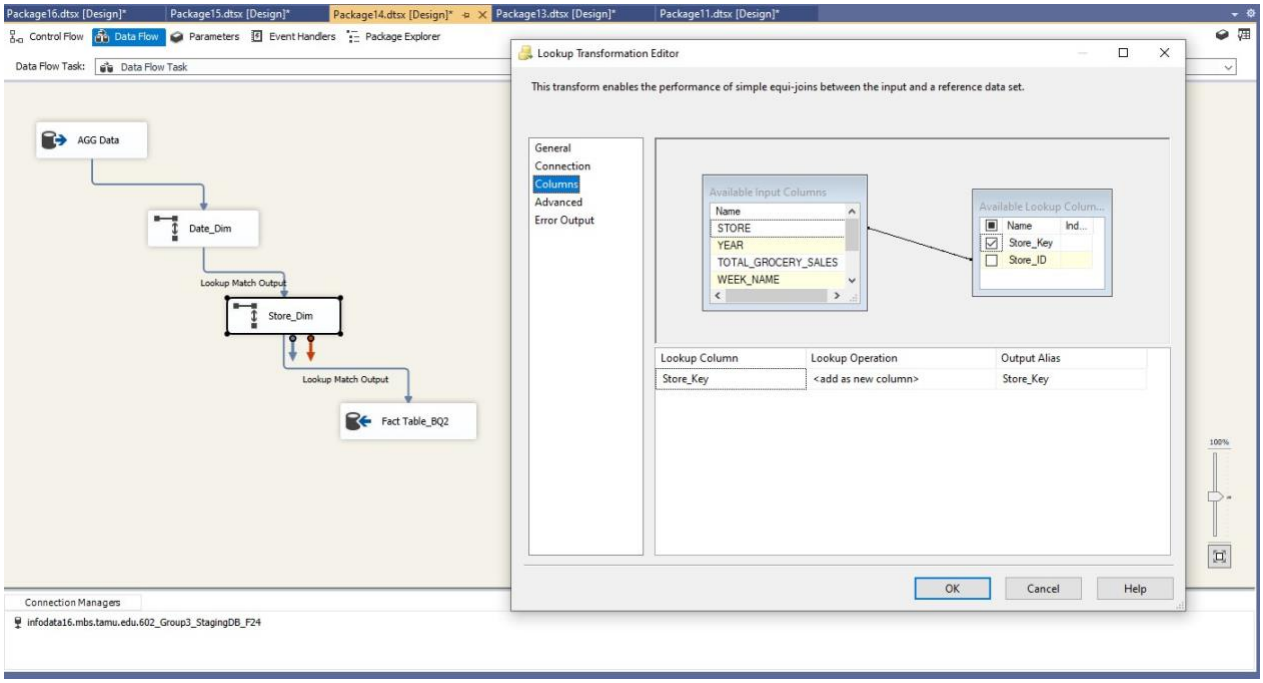


Fig Lookup 2 to - Mapping columns from Store Dimension Table

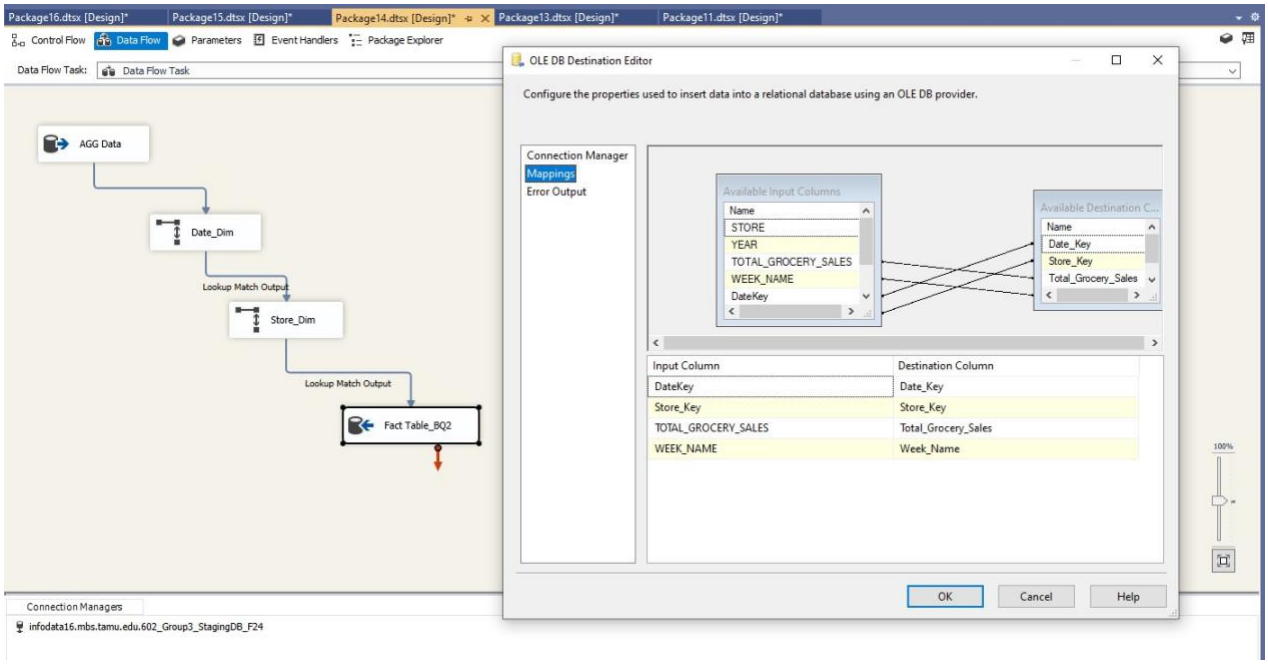


Fig Mapping columns from Lookups

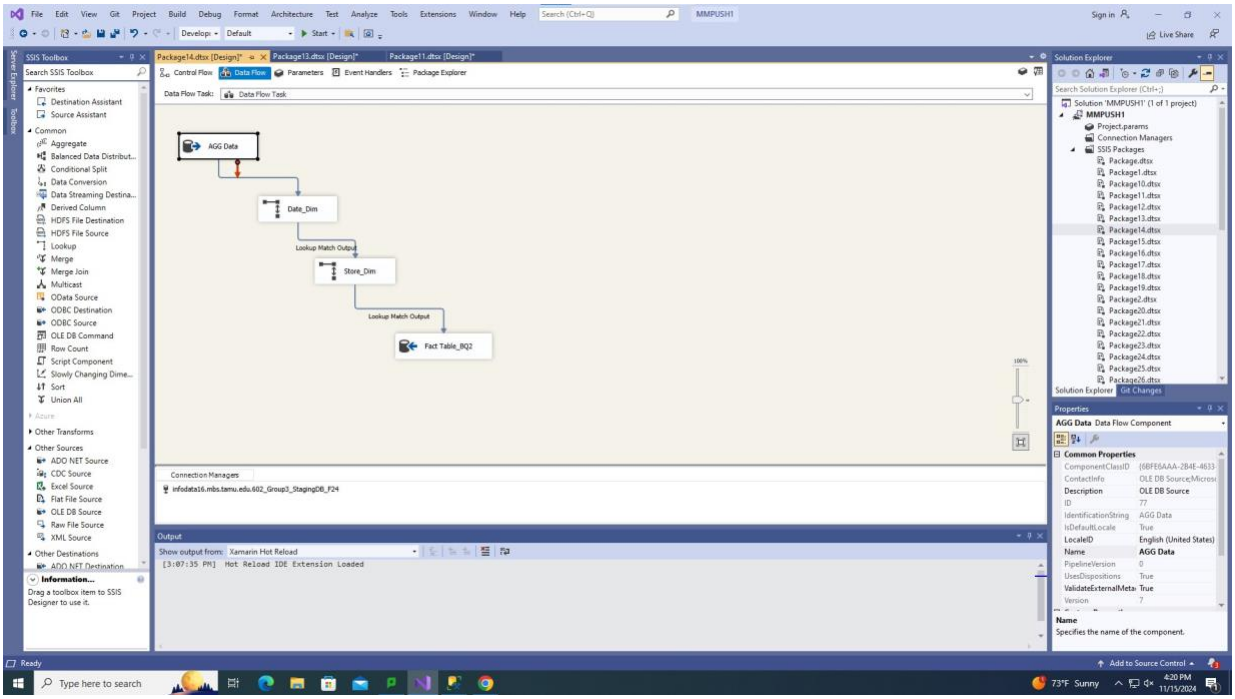


Fig Fact Table Loaded

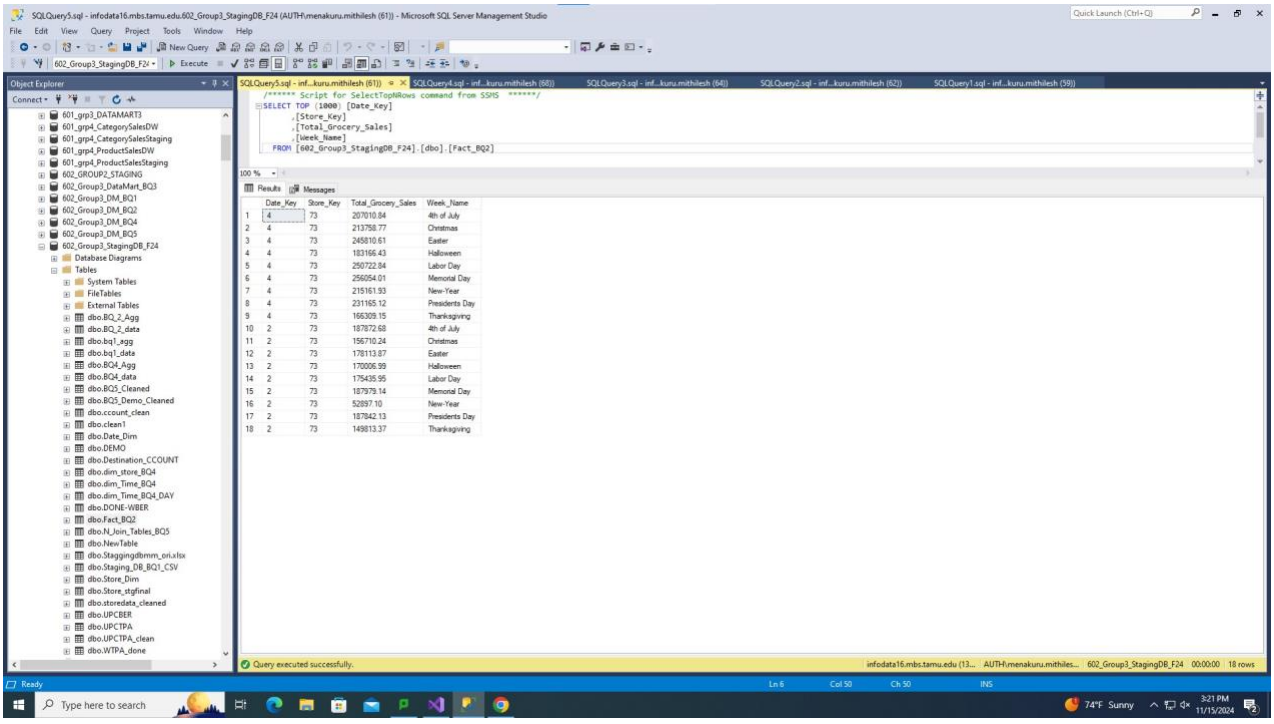


Fig Validating fact table records after loading in SSMS

2. Fact_Sales

We have created a new package in SSIS and used Execute SQL to create fact table called **“Fact_Sales”**

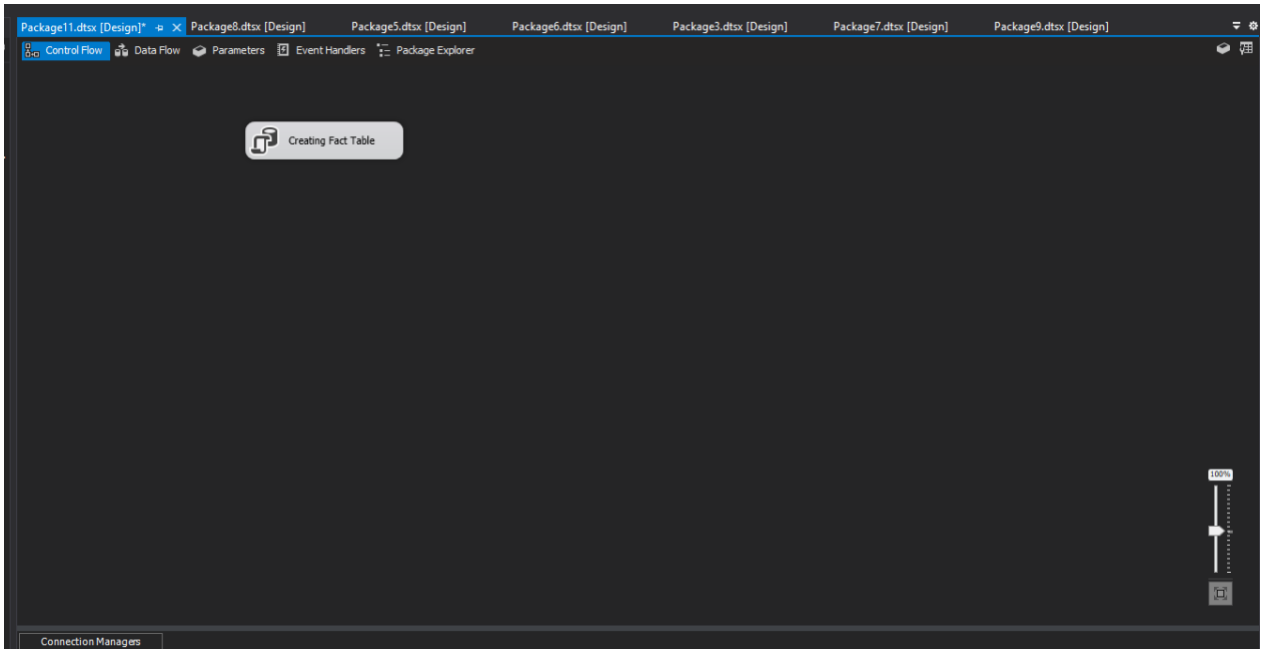


Fig: Creating Fact table

Create statement to create a fact table:

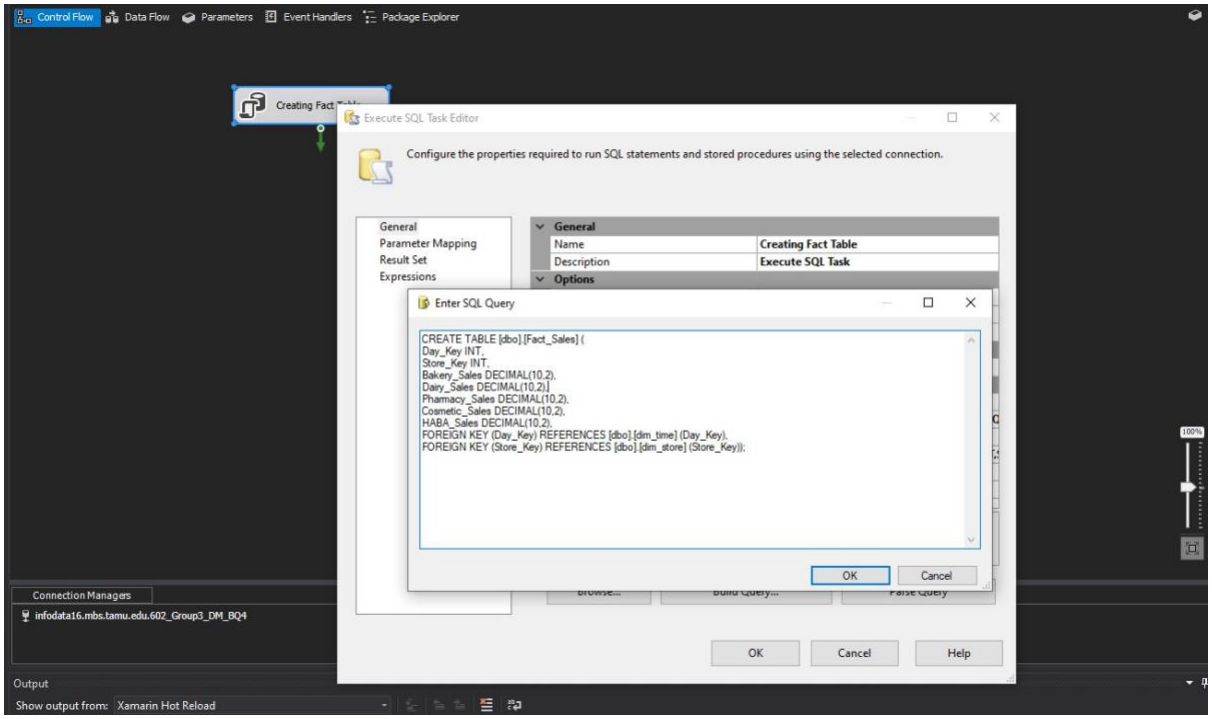


Fig: Create statement for fact table

Fact table loading:

- Created a new SSIS Package.
- Created OLEDB Source, 2 Lookups and OLEDB Destination and connected them.
- We have connected OLE DB Source to staging and fetching BQ4_data table.
- In lookup1, we connected to the store dimension of data mart and if store value matches between staging table column STORE and Store_Id column of store dimension , then fetching Store_Key.
- In lookup 2, we connected to the time dimension of data mart and if the date value matches between staging table column DATE and Date column of Time dimension, then fetching Day_Key.
- In the OLEDB destination, we made a connection to data mart and mapped respective columns from staging table, dimensions to fact table.

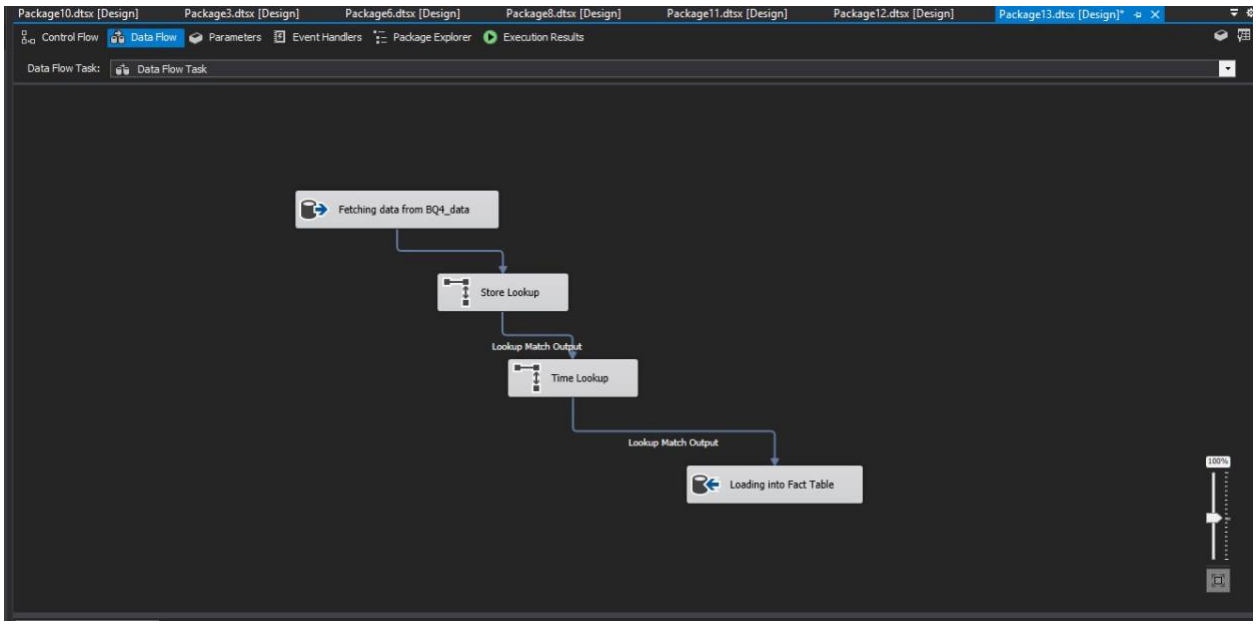


Fig: Fact table creation flow

Loading data from BQ4_data, a table in staging which contains data related to business question 4.

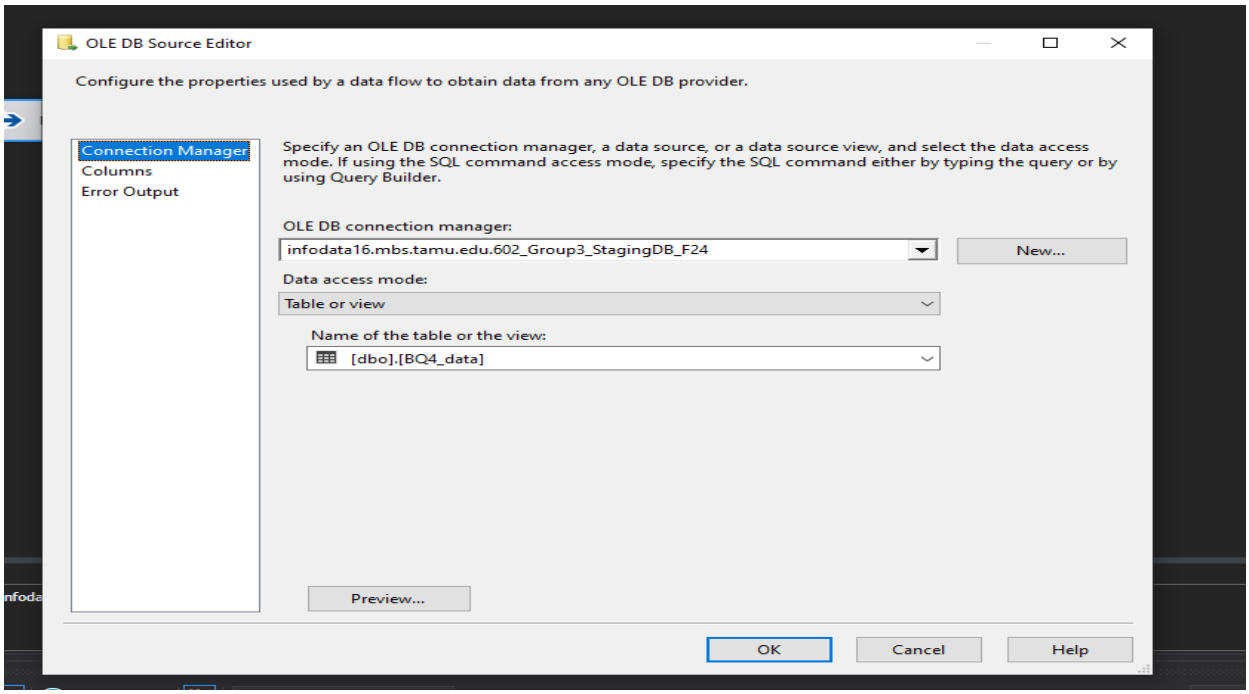


Fig: Selecting BQ4_data table from Staging database

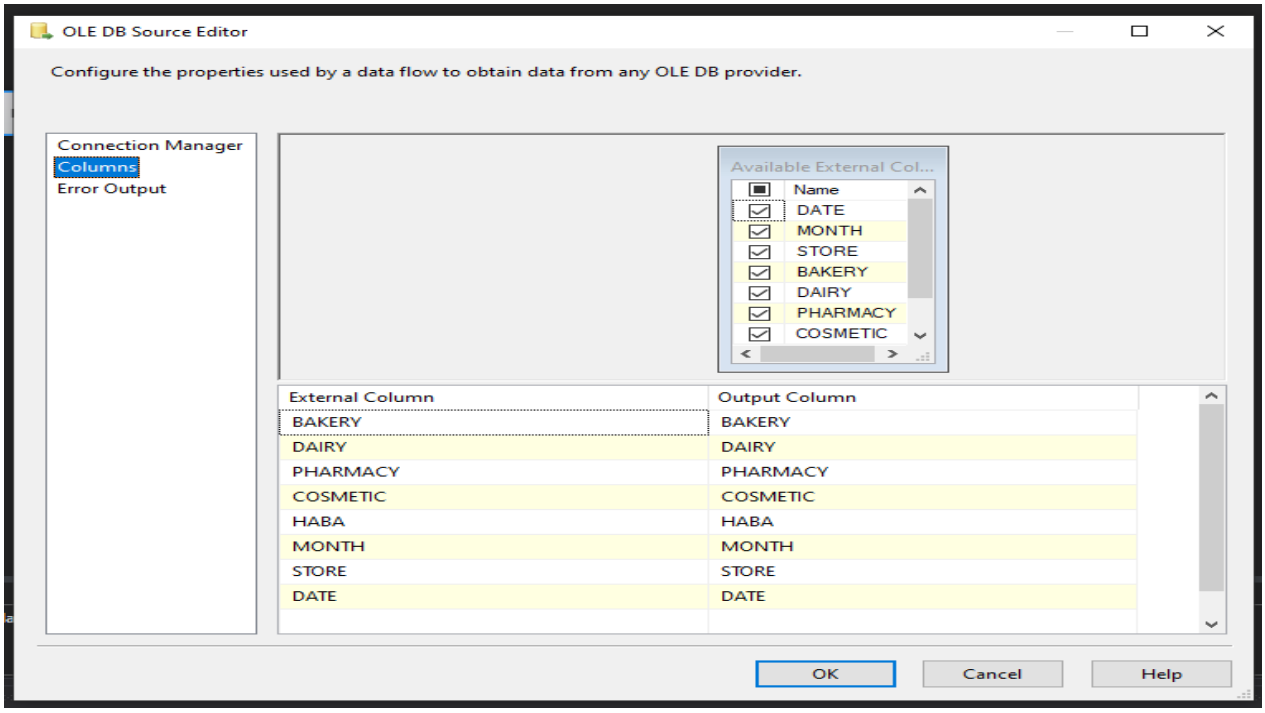


Fig: Columns selected from BQ4_data

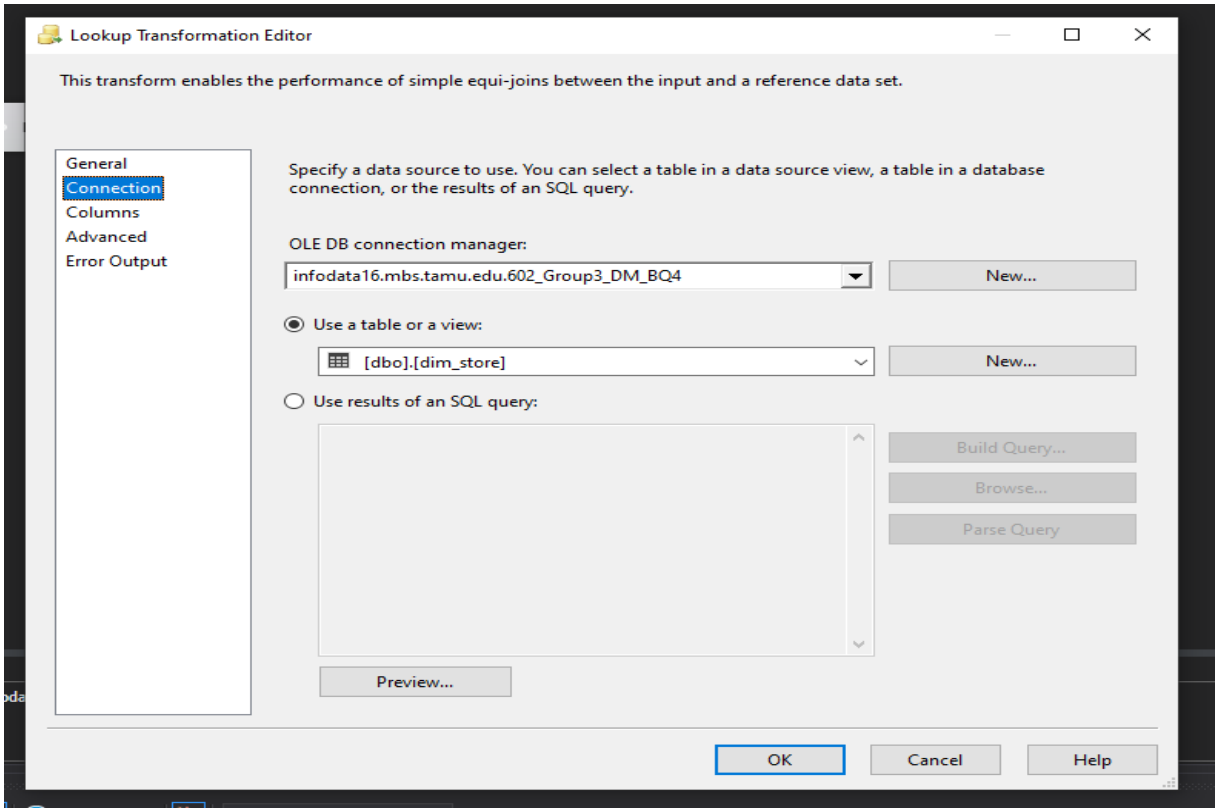


Fig: Selecting store dimension in Lookup 1

Mapping store column of BQ4_data from staging with Store_Id of dim_store dimension and selecting Store_Key as output.

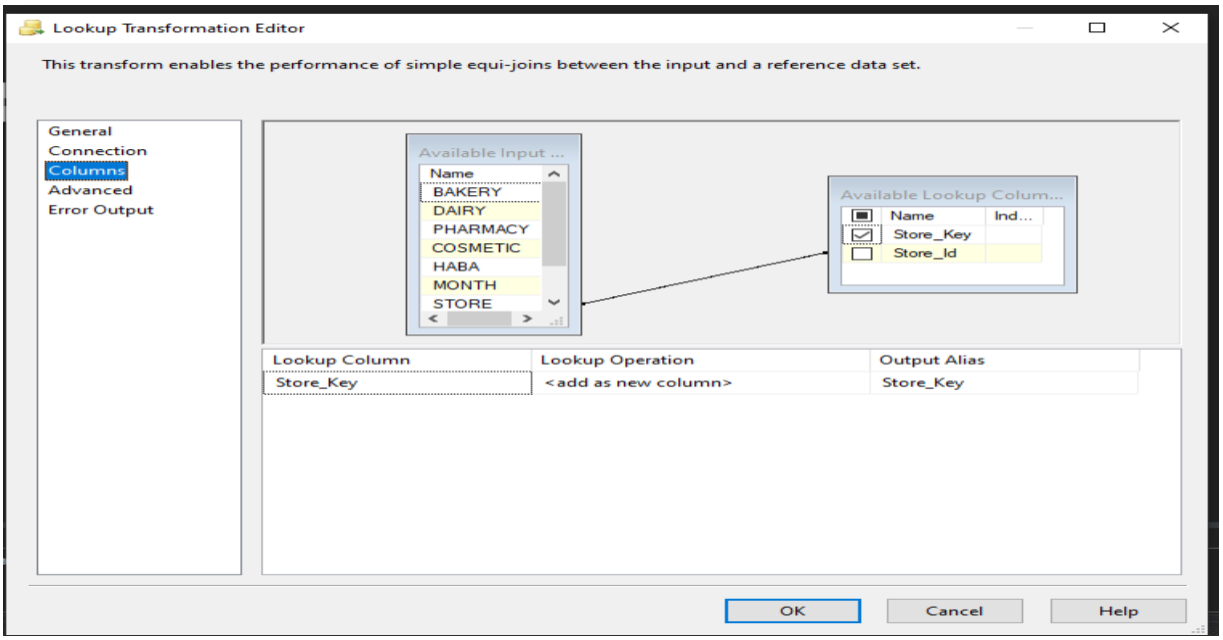


Fig: Store dimension mapping

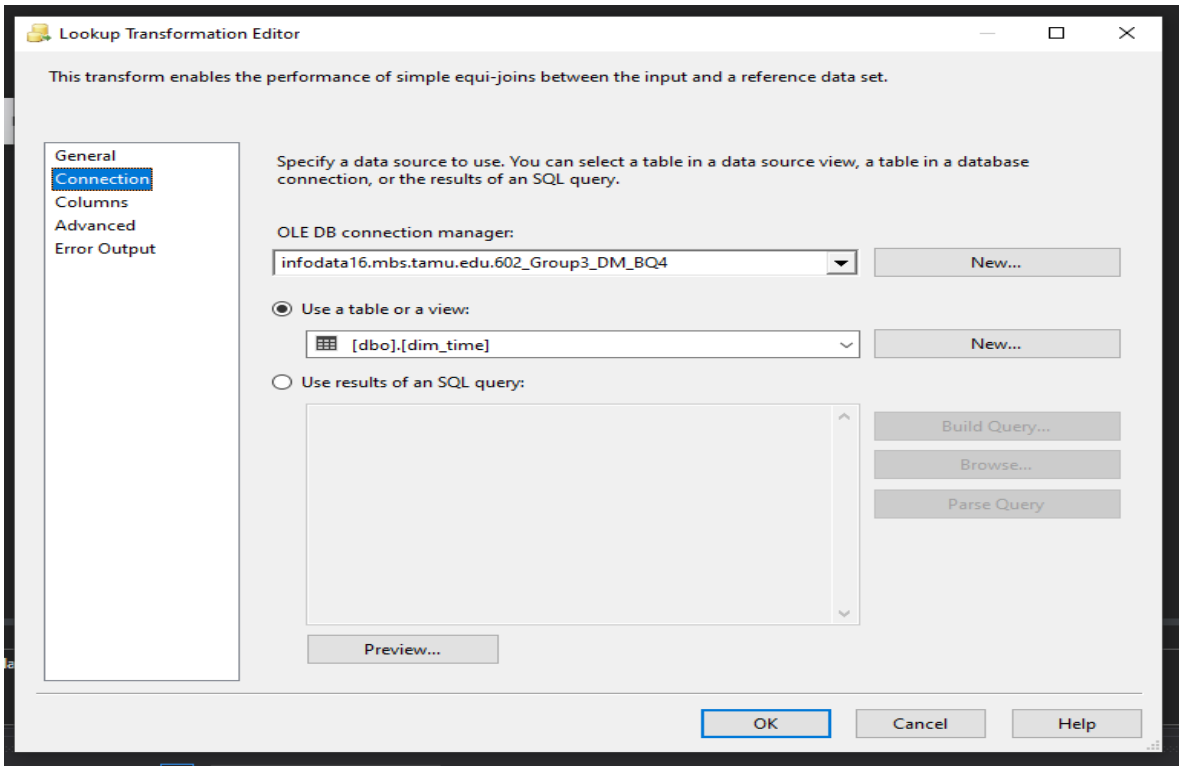


Fig: Selecting Time dimension for Look up 2

Mapping DATE column of BQ4_data from staging with DATE of dim_time dimension and selecting Day_Key as output.

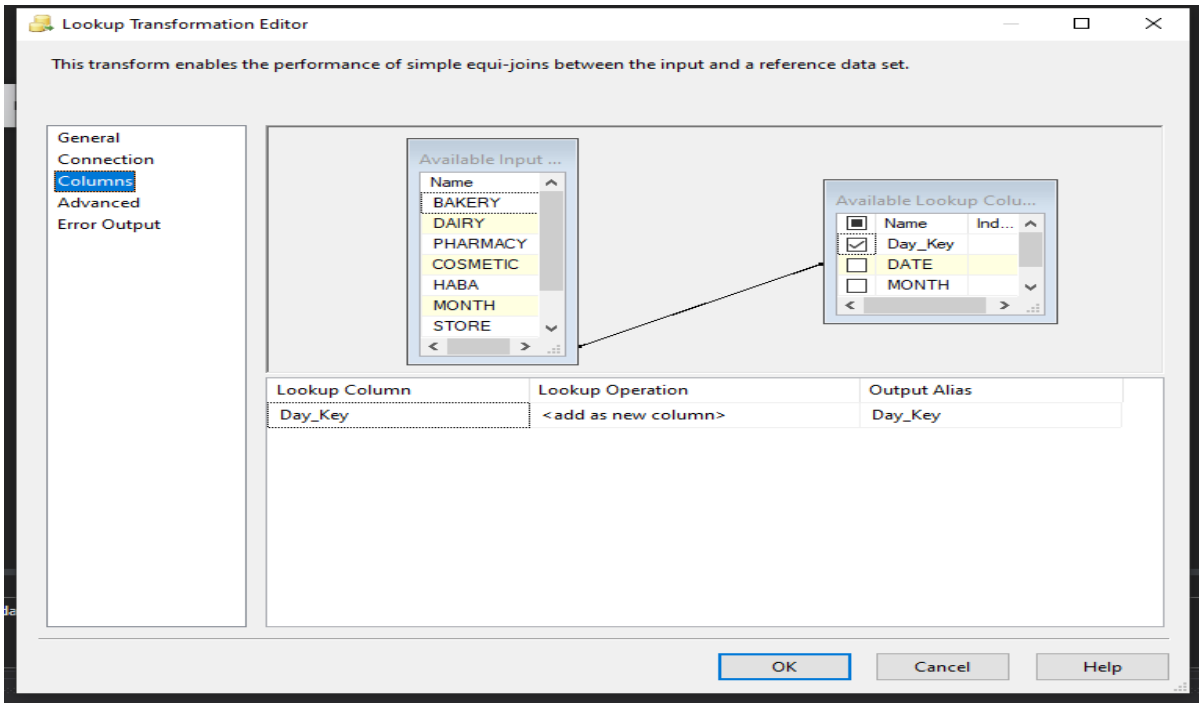


Fig: Time dimension mapping

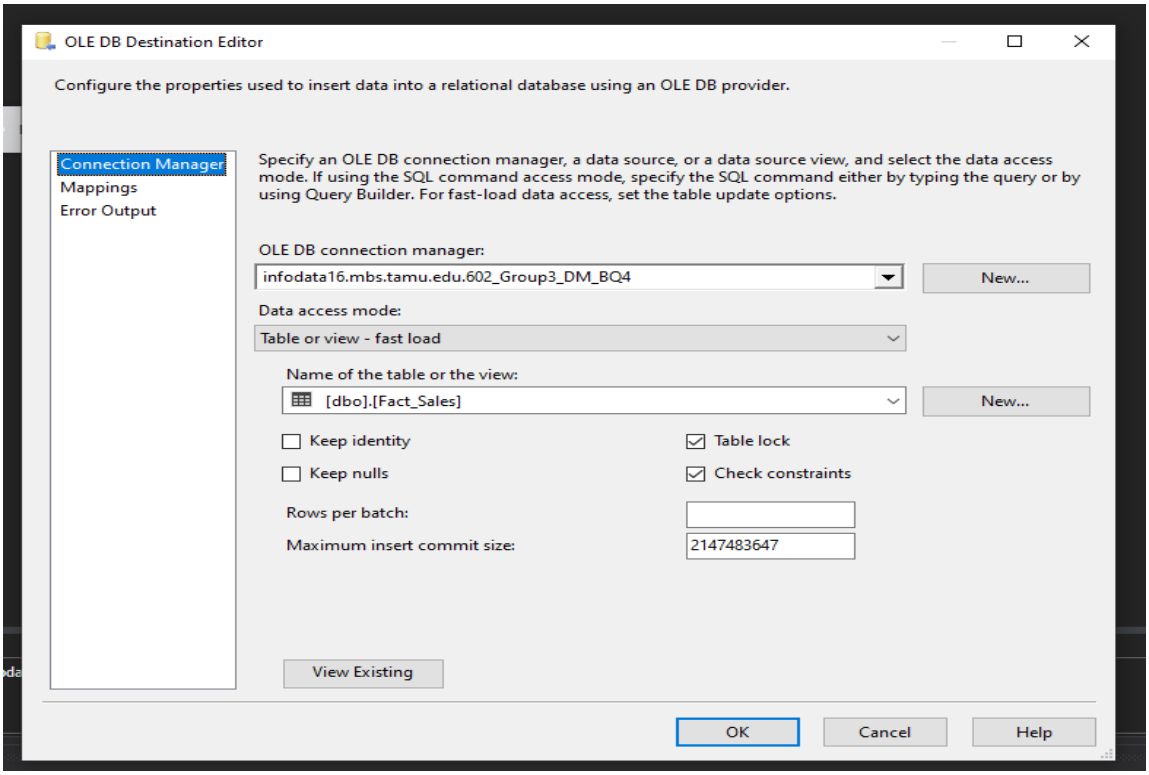


Fig: Connecting to Fact table of data mart

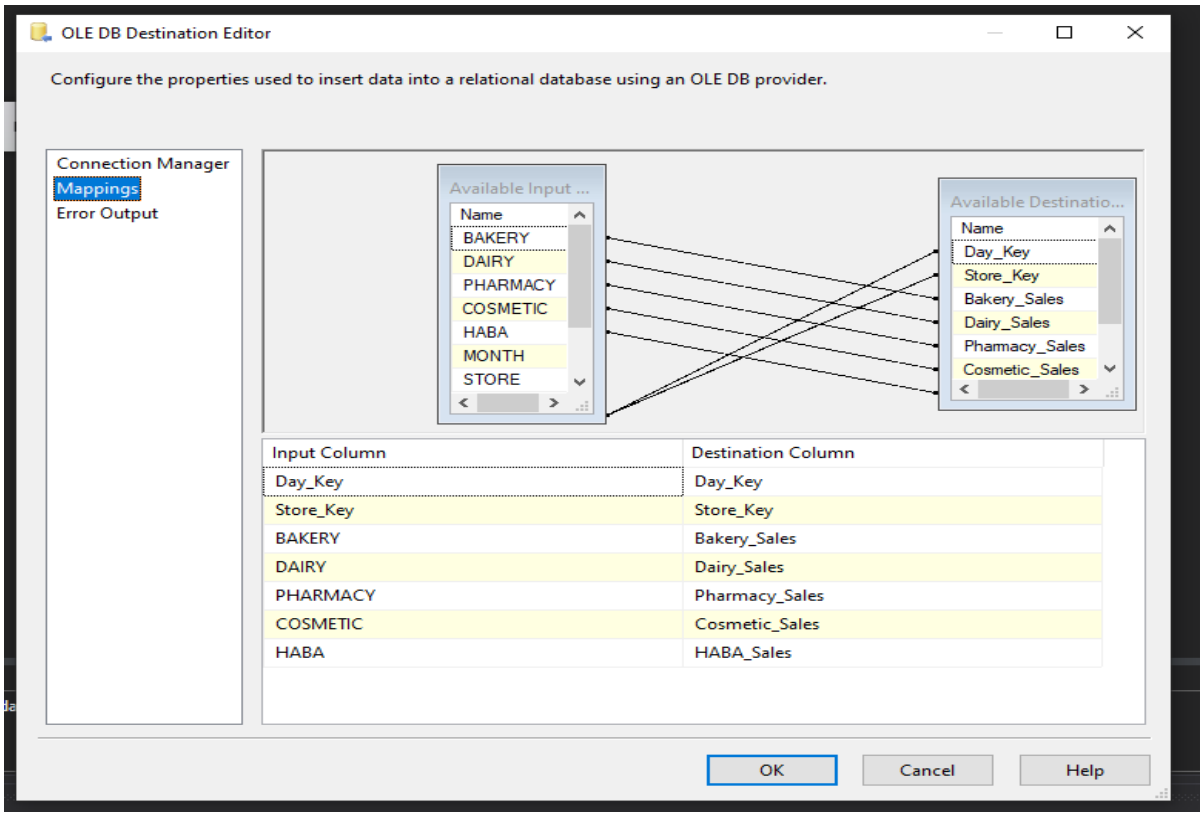


Fig: Mapping columns for fact table

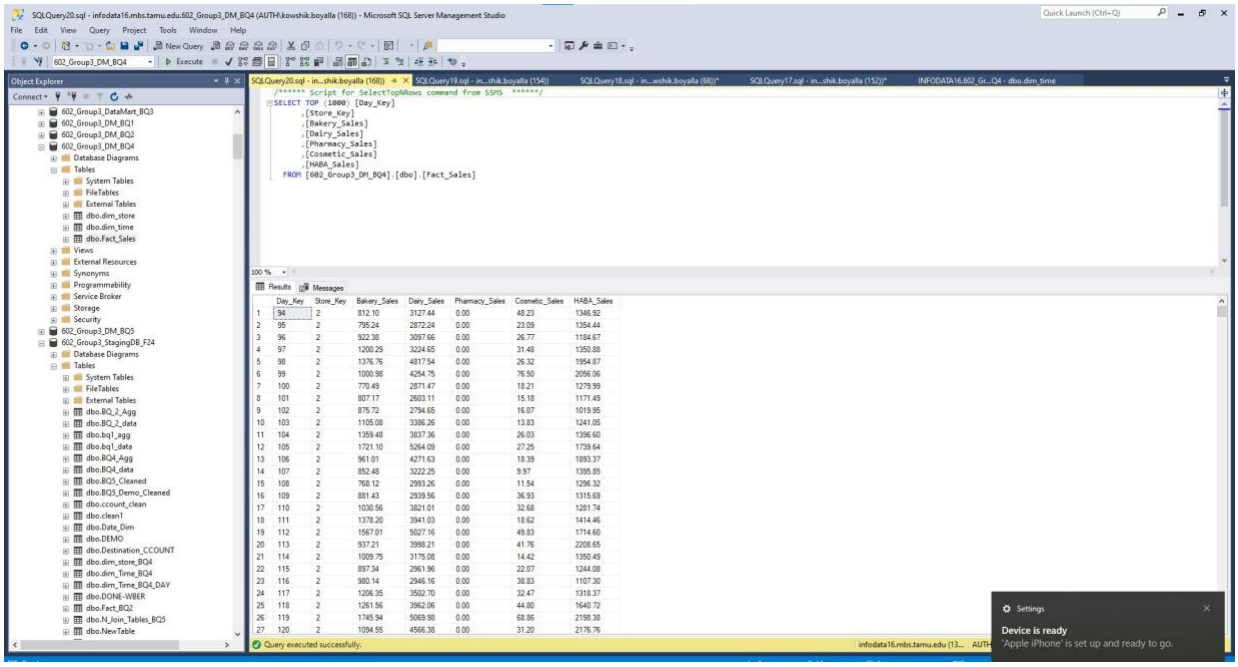


Fig: Validating fact table records after loading in SSMS

3. **Store_Traffic_Fact :**

We executed SQL to create fact table called **"Store_Traffic_Fact".**

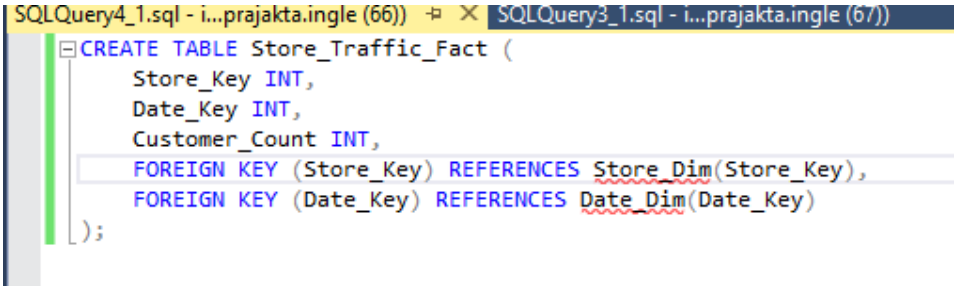


Fig: Created Store_Traffic_Fact

Fact table loading:

1. Created a new SSIS Package.
2. Created OLEDB Source, 2 Lookups and OLEDB Destination and connected them.
3. We have connected OLE DB Source to staging and fetching BQ1_data tables.
4. In lookup1, we connected to the store dimension of data mart and if the store value matches between the staging table column STORE and Store_Id column of store dimension , then fetching Store_Key.
5. In lookup 2, we connected to the week dimension of data mart and if the date value matches between staging table we store the week_ID
6. In the OLEDB destination, we made a connection to the data mart and mapped respective columns from the staging table, and dimensions to fact table.

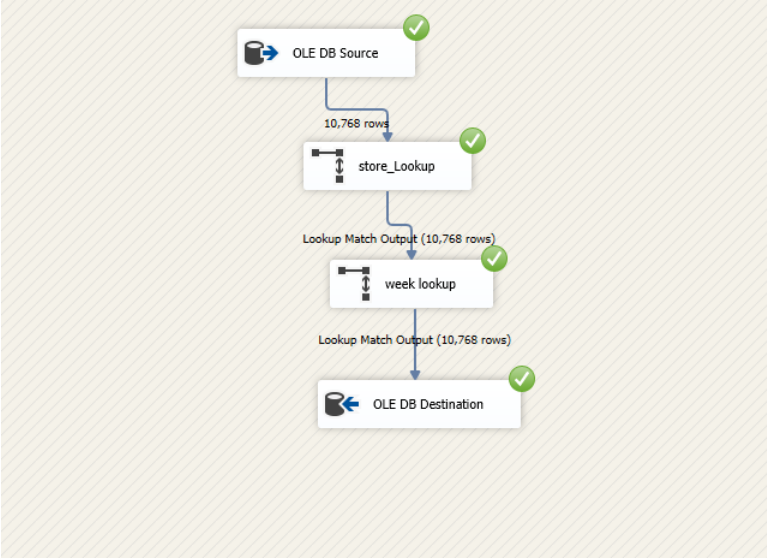


Fig: Fact table creation flow

Loading data from BQ1_data, a table in staging which contains data related to business question 1.

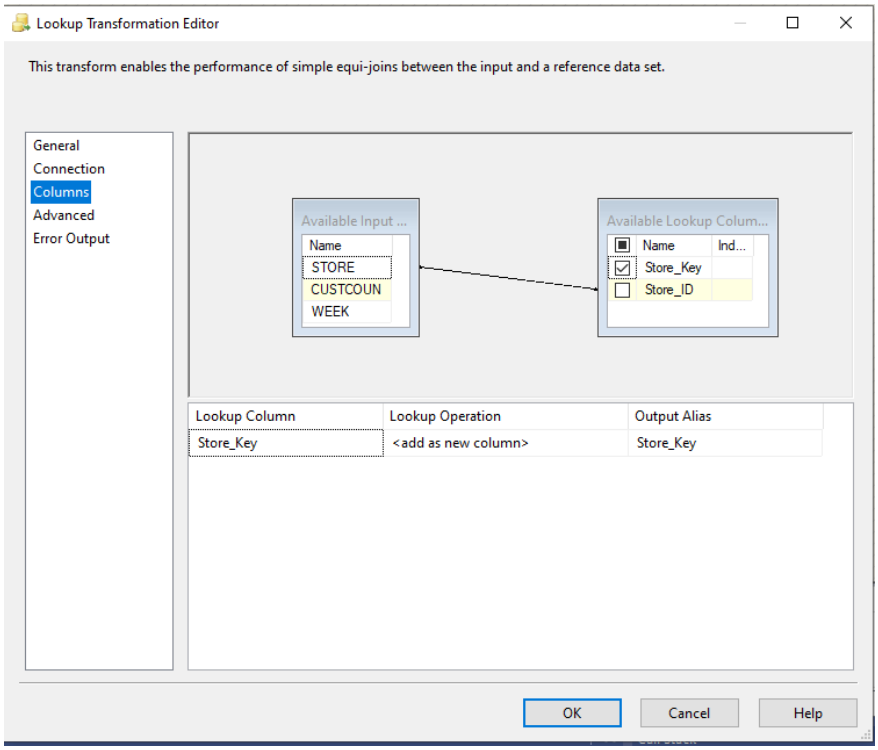


Fig: Store DImension table mapping

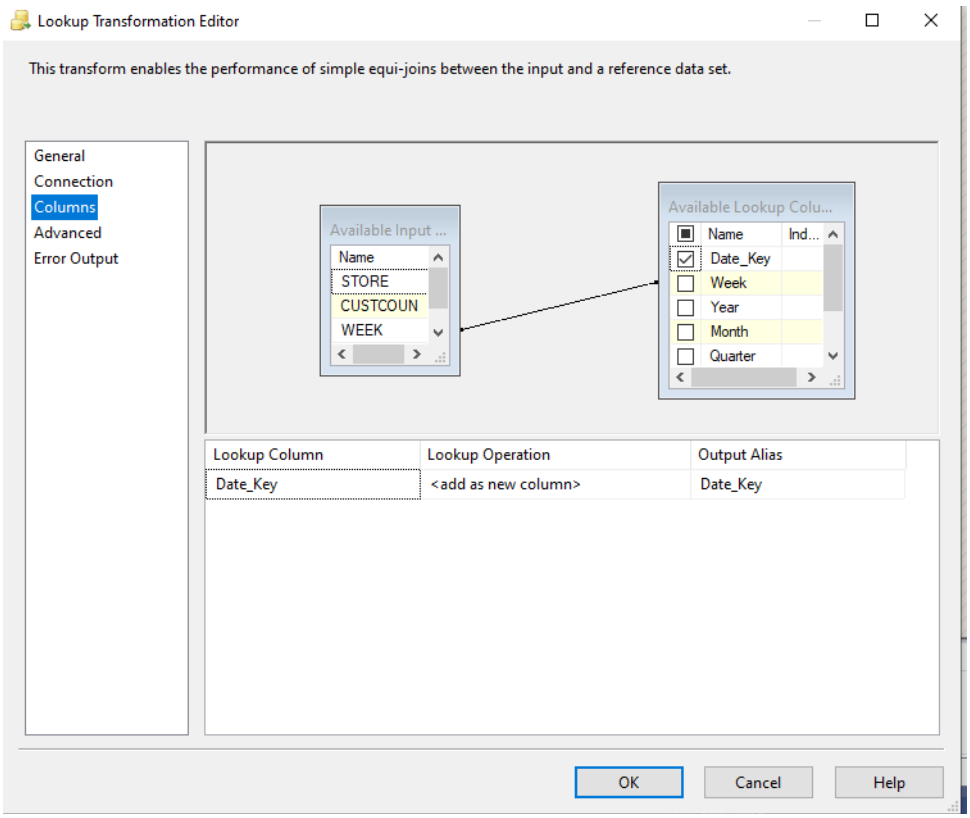


Fig: Date Dimension table mapping

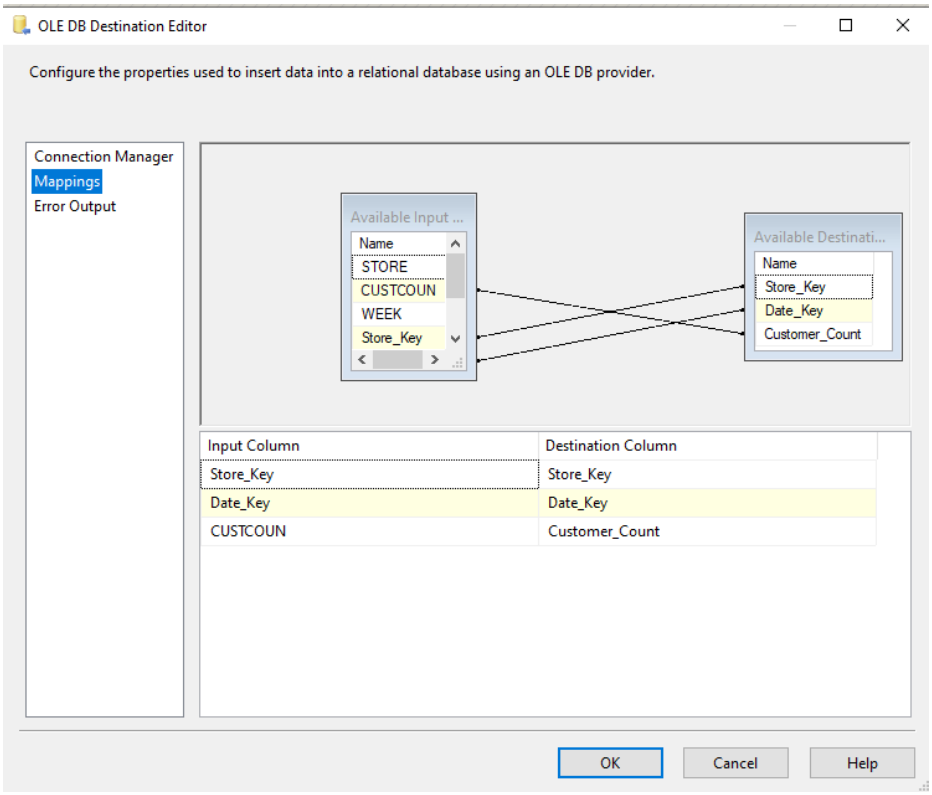


Fig: Fact table mapping

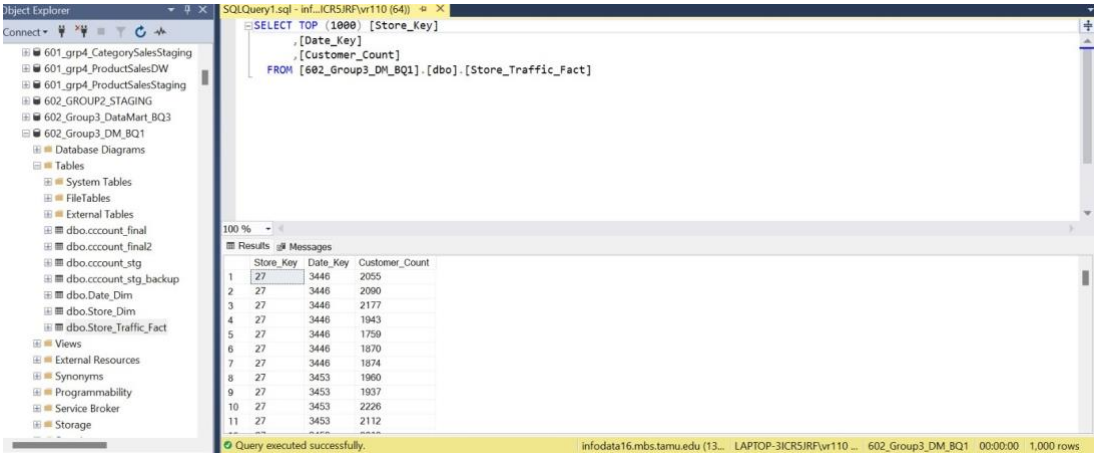


Fig: Successful validation of Fact table data

4. Customer_Demographics_Fact

We have created a new package in SSIS and used Execute SQL to create fact table called **"Customer_Demographics_Fact"**

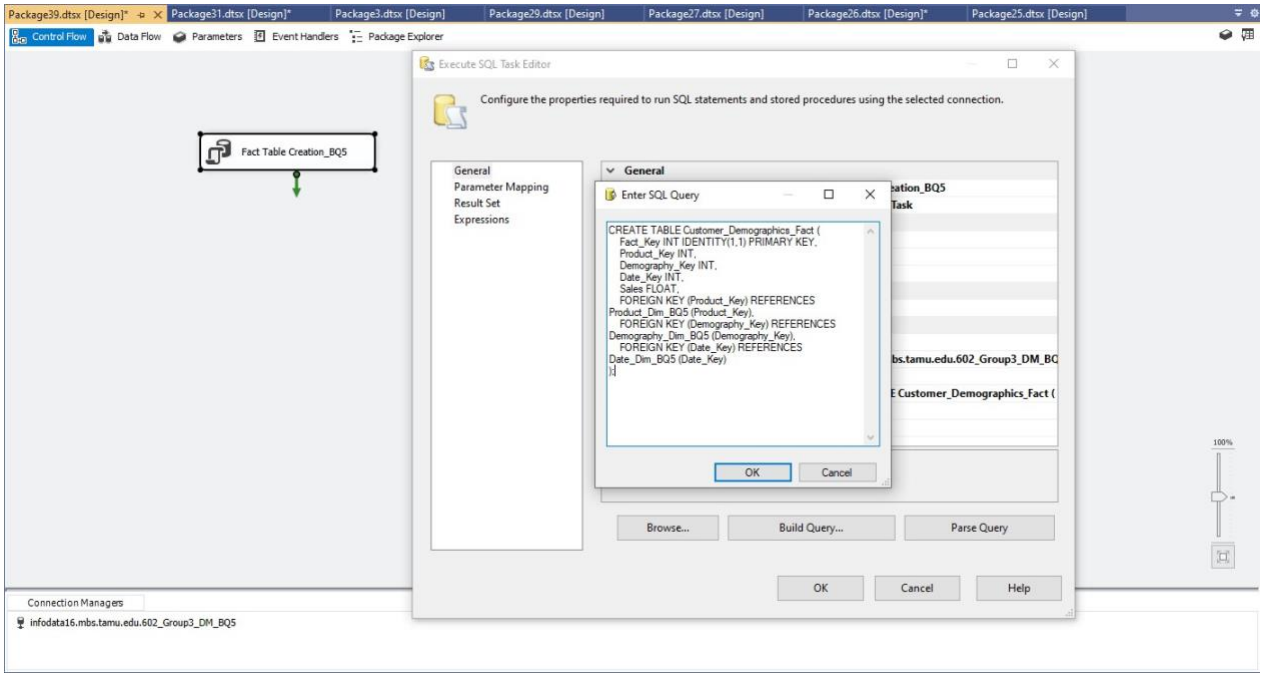


Fig Fact Table Creation

Fact table loading:

- Created a new SSIS Package.
- Created OLEDB Source, 2 Lookups and OLEDB Destination and connected them.

- We have connected OLE DB Source to staging and fetching the Customer_Demographics_Fact data table.
- In lookup 1, we connected to the date dimension of data mart and if the Week value matches between staging table column DATE and Date column of Date dimension, then fetching Day_Key.
- In lookup 2, we connected to the demographic dimension of data mart and if demo_key value matches between staging table column and Demo_ID column of demographic_dimension , then fetching Demographic_Key.
- In lookup 3, we connected to the Product dimension of data mart and if Product_key value matches between staging table Descrip column and Prdouct_ID column of Product_dimension , then fetching Product_Key.
- In the OLEDB destination, we made a connection to data mart and mapped respective columns from staging table, dimensions to fact table.

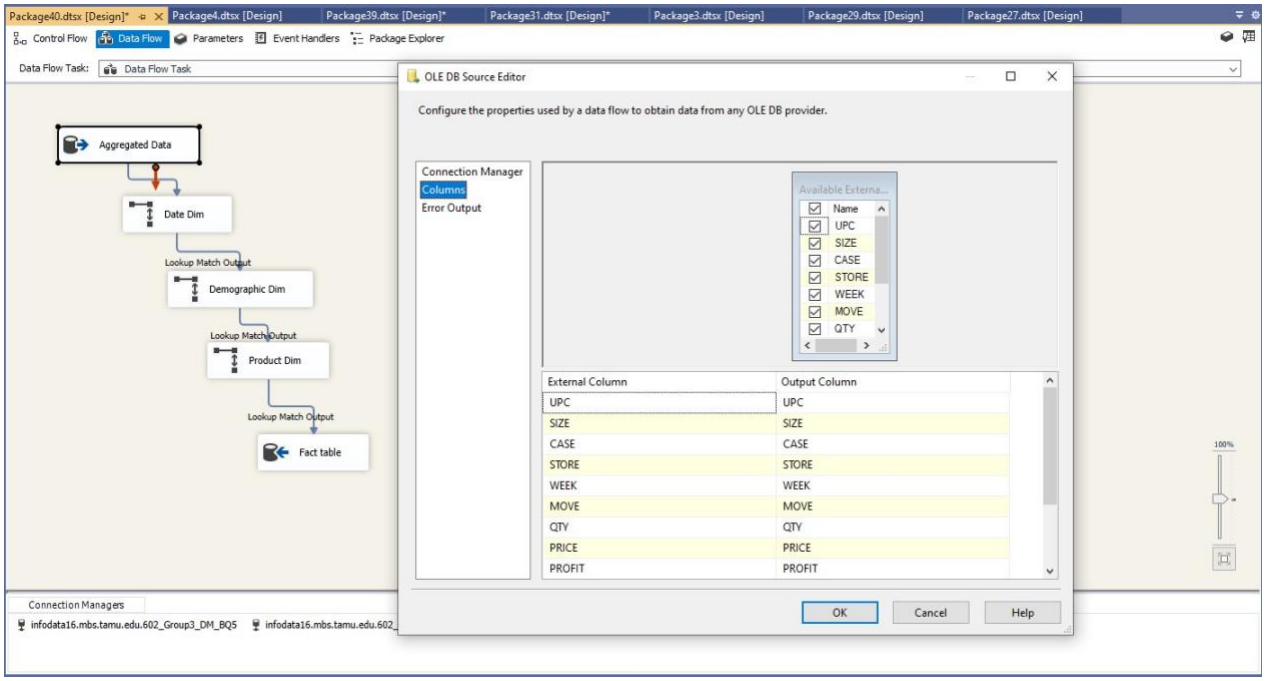


Fig Selecting the Columns from Source Table

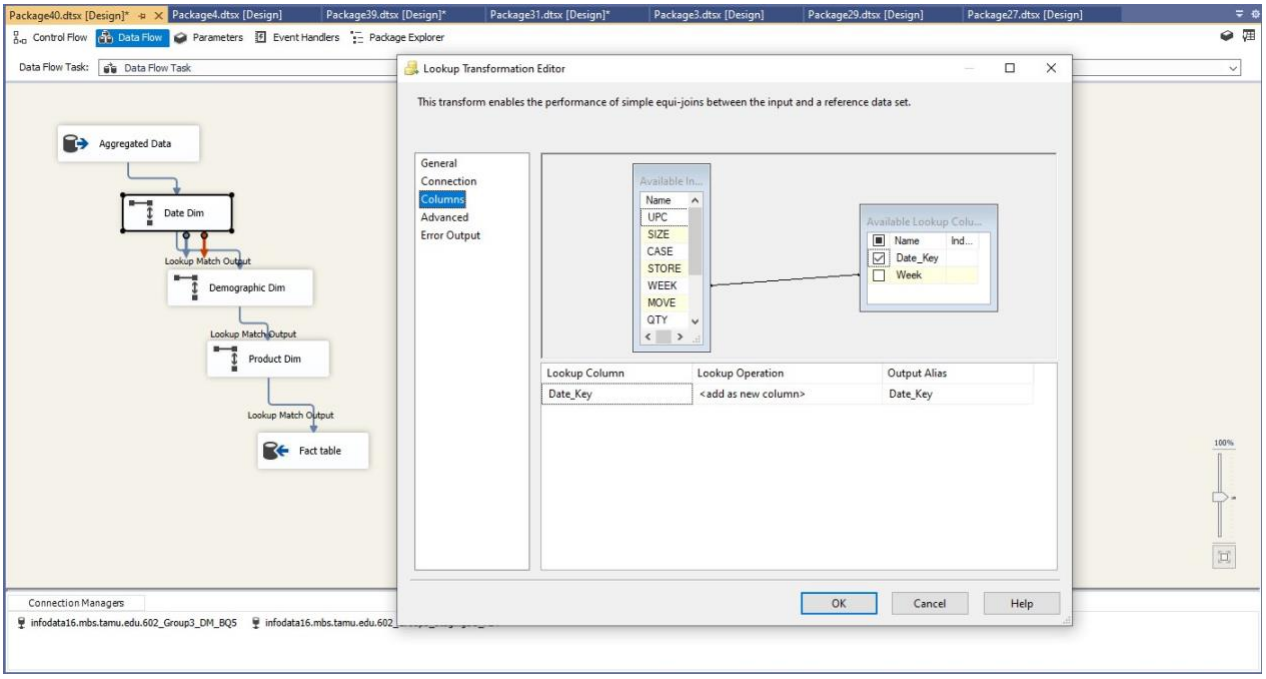


Fig Selecting the Columns from Date Table in Look up 1

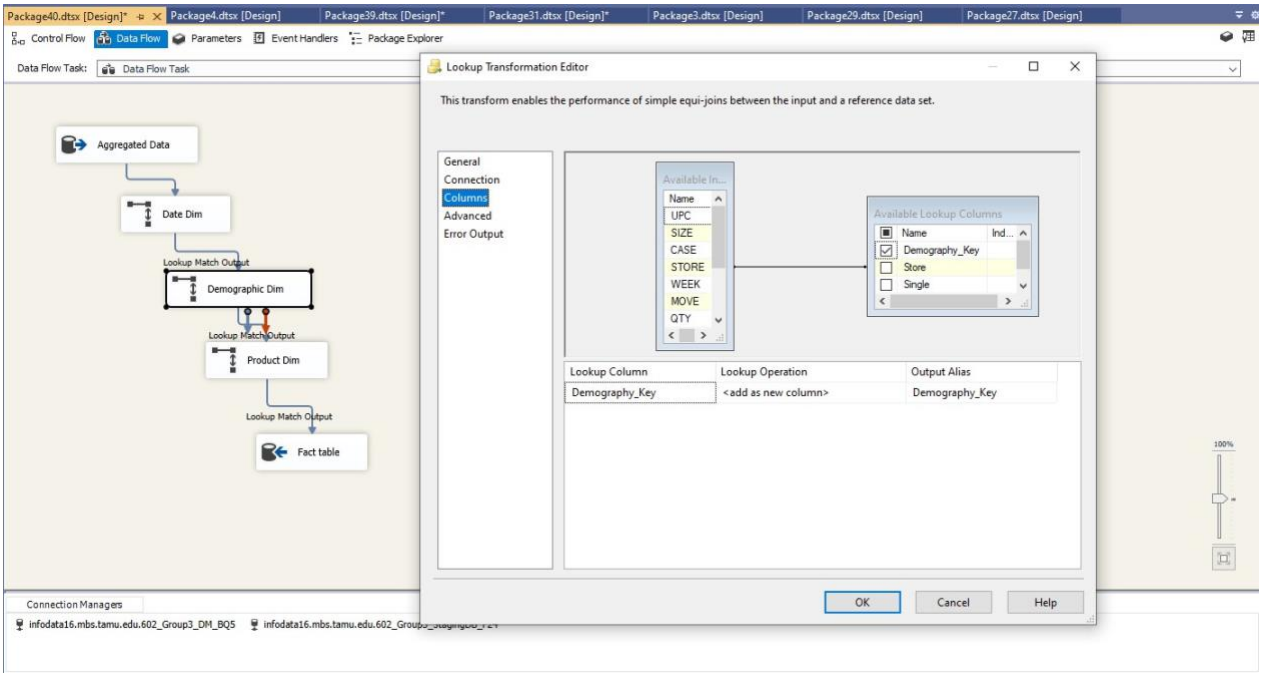


Fig Selecting the Columns from Demo Table in Look up 2

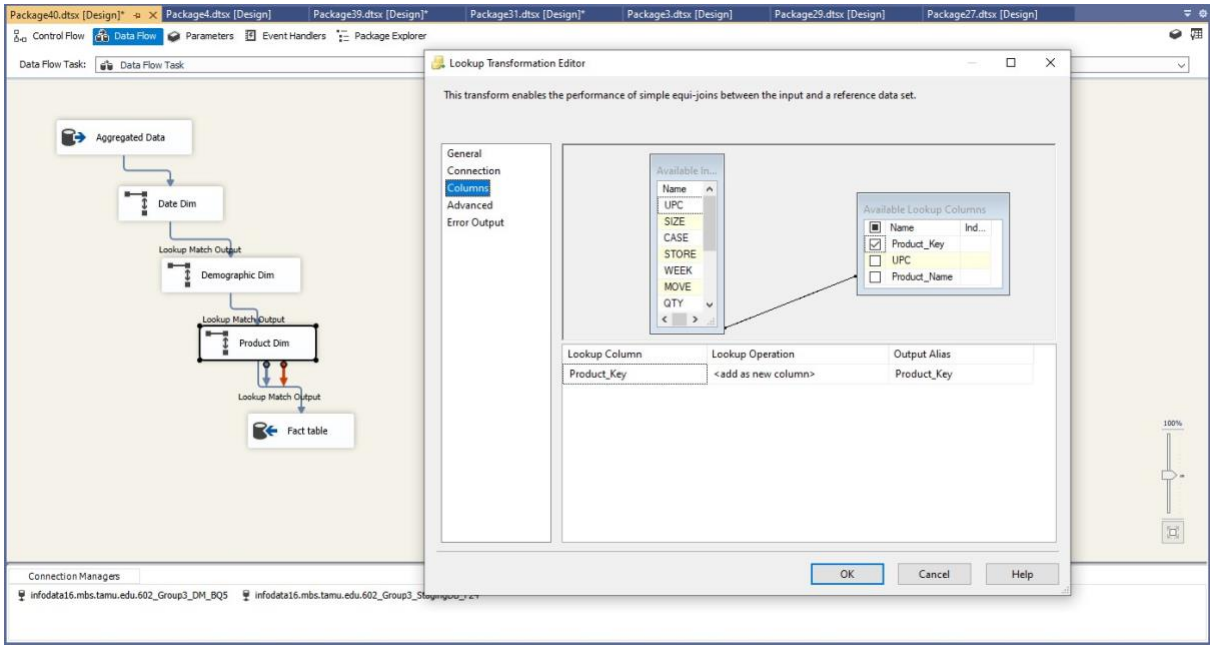


Fig selecting the Columns from ProductTable in Look up 3

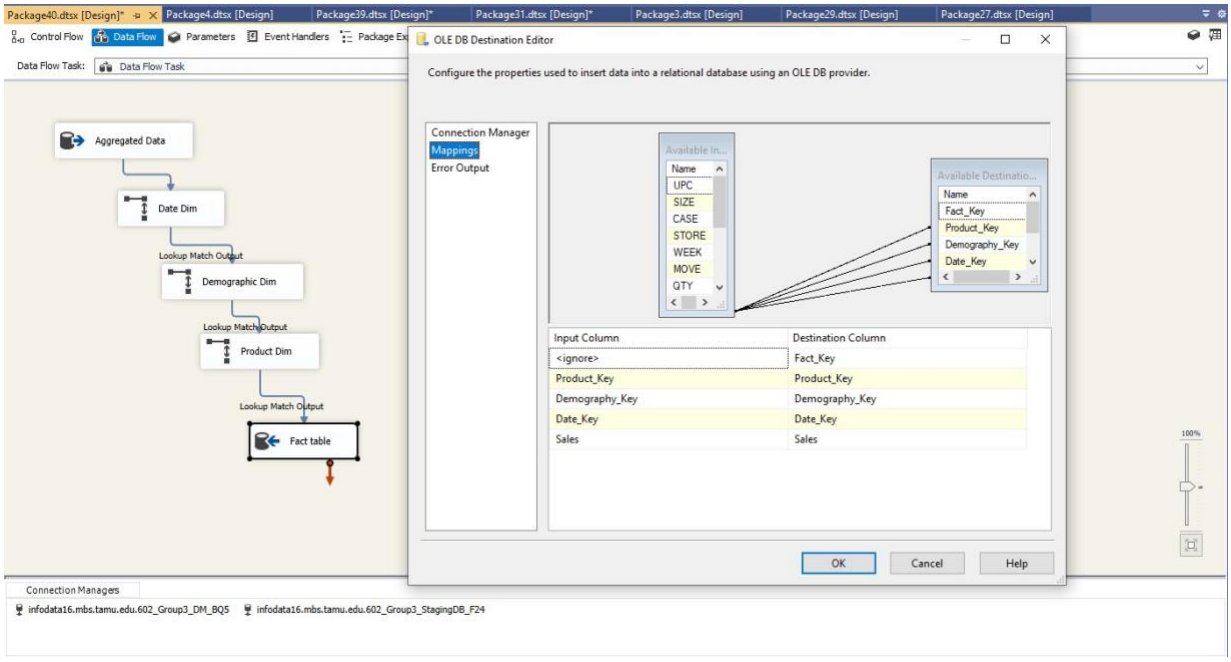


Fig Mapping of Fact Table

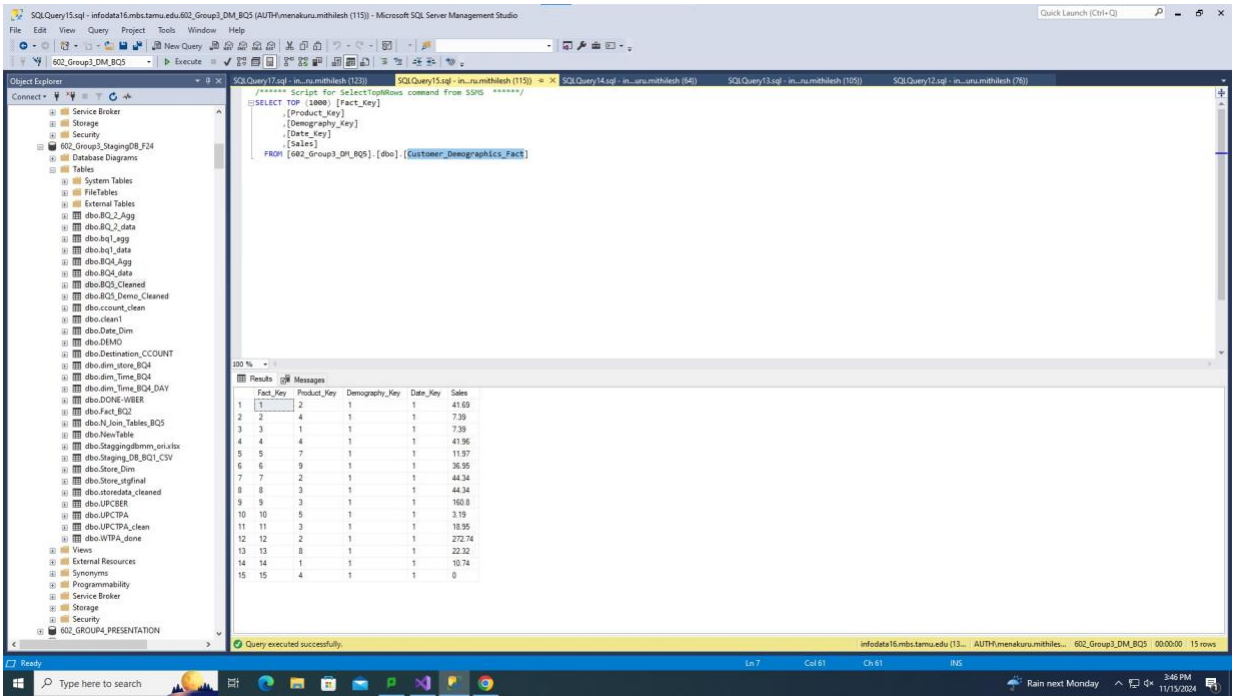


Fig Successful validation of Fact table data

5. Profit_Fact

```
CREATE TABLE Profit_Fact (
    Profit_ID INT PRIMARY KEY IDENTITY(1,1),
    Product_ID INT,
    Units_sold INT,
    PRICE DECIMAL(10, 2),
    QTY INT,
    PROFIT DECIMAL(5, 2),
    FOREIGN KEY (Product_ID) REFERENCES Product_Dimension(UPC)
);
```

Fig: Create Profit_Fact Table creation

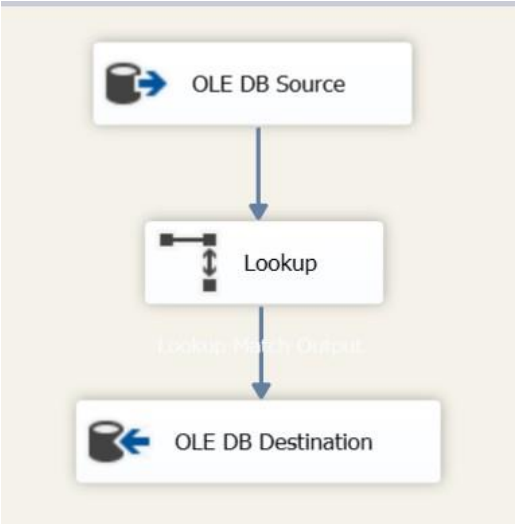


Fig: Profit_Fact Table Data Flow

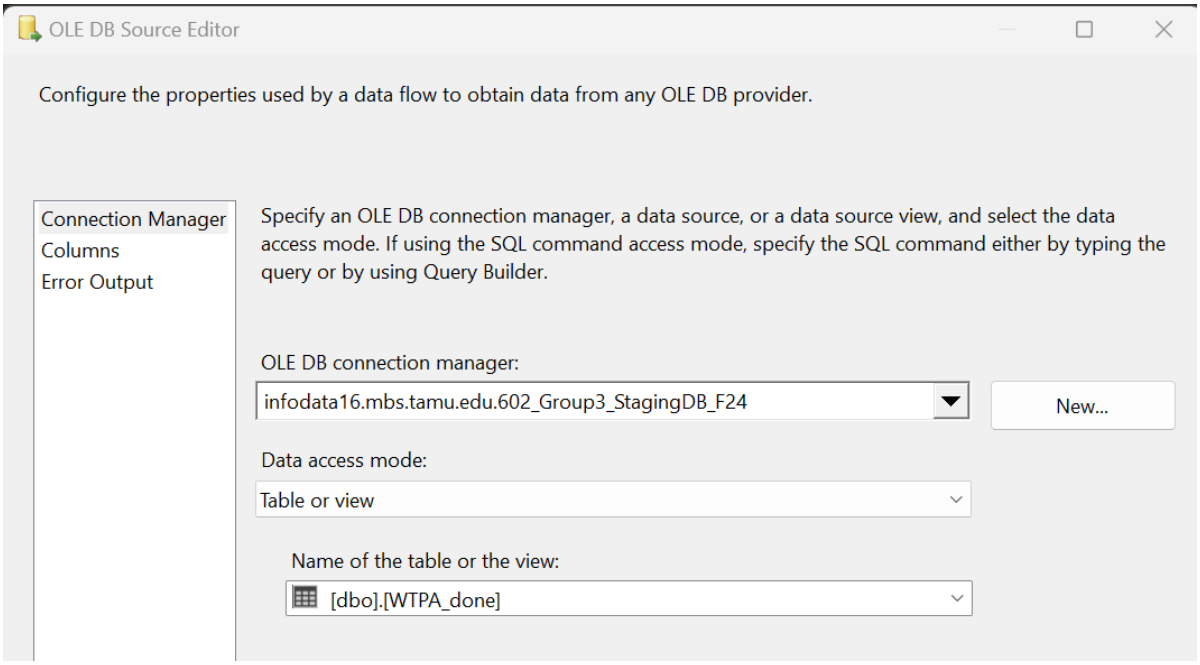


Fig: Configure OLE DB Source as dbo.WTPA_done

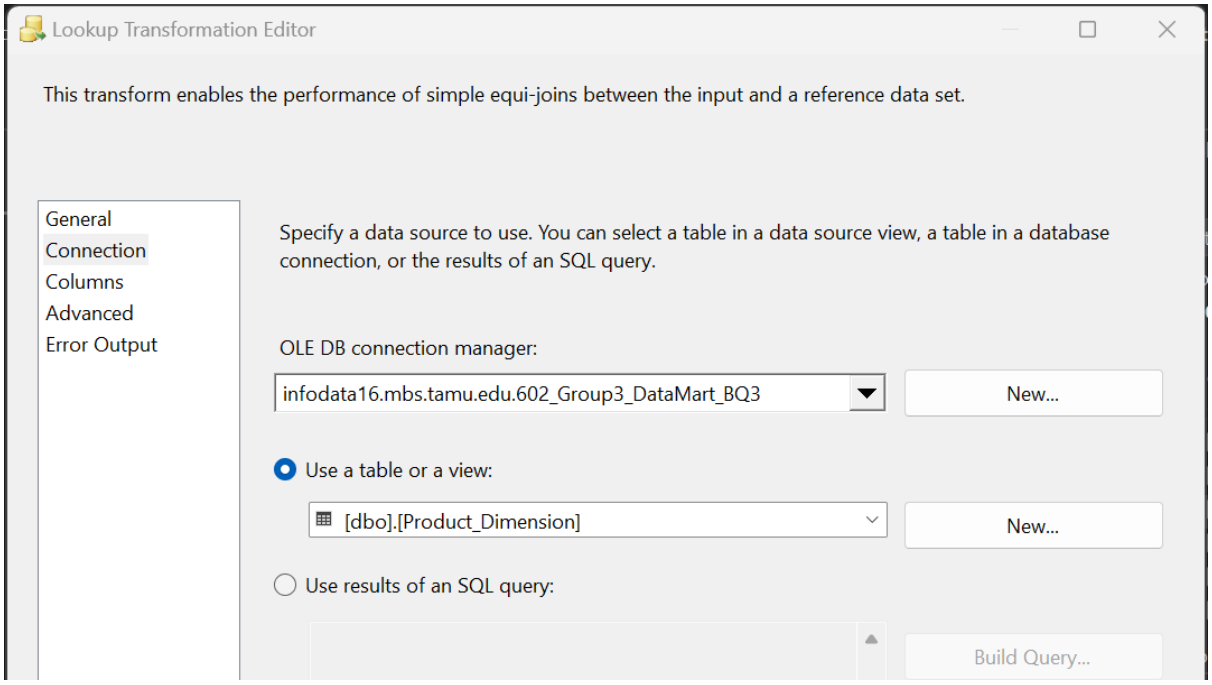


Fig: Use Lookup to get Brand_ID from Product_Dimension Table

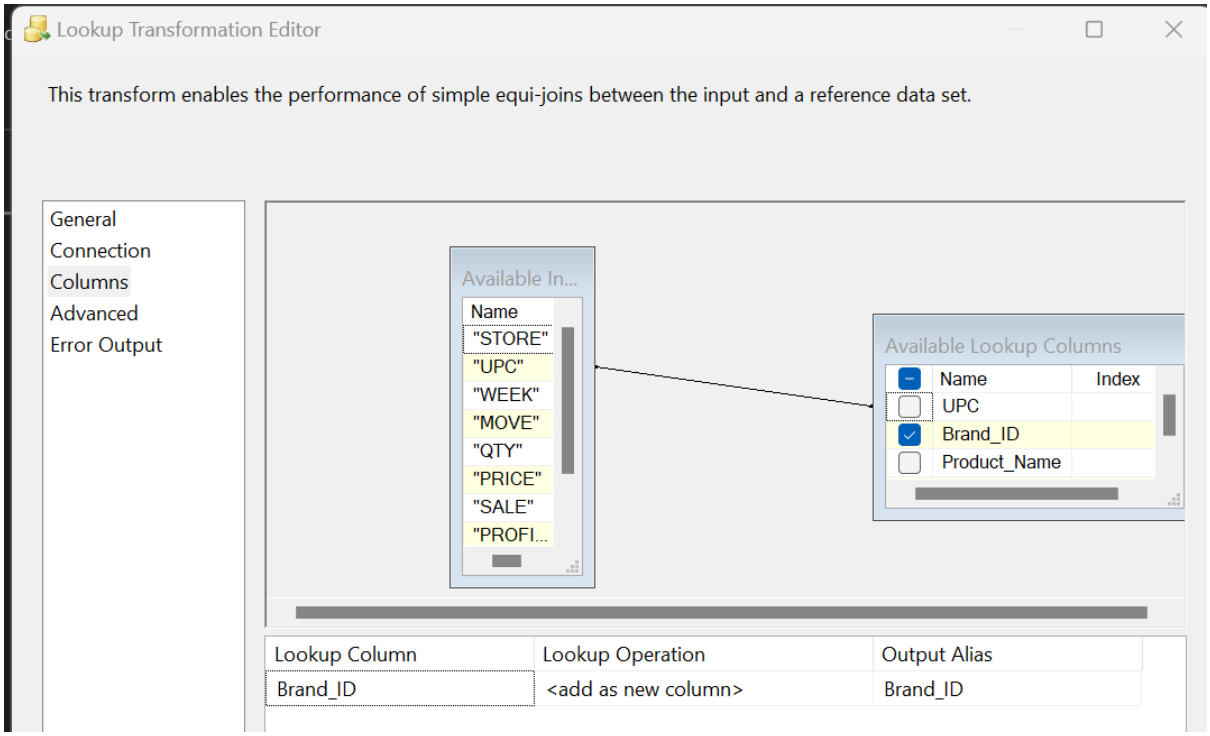


Fig: Lookup Transformation Mapping

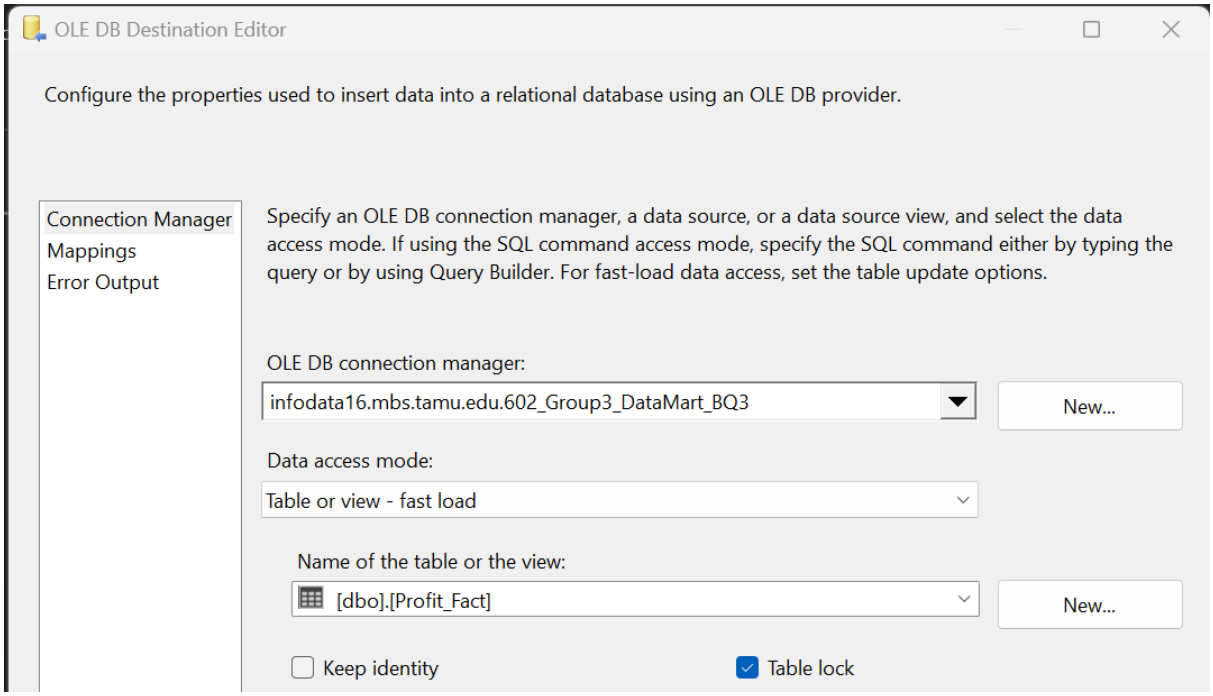


Fig: Configure Profit_Fact as the destination for OLE DB Destination

```
SELECT TOP (1000) [Profit_ID]
      ,[UPC]
      ,[Brand_ID]
      ,[Units_Sold]
      ,[PRICE]
      ,[PROFIT]
  FROM [602_Group3_DataMart_BQ3].[dbo].[Profit_Fact]
```

100 %

Results | Messages

	Profit_ID	UPC	Brand_ID	Units_Sold	PRICE	PROFIT
1	1	1111341101	1279	6	1.34	40.00
2	2	1111341101	1279	4	1.34	40.00
3	3	1111341101	1279	16	0.99	19.00
4	4	1111341101	1279	26	1.03	22.00
5	5	1111341101	1279	0	0	0.00
6	6	1111341101	1279	0	0	0.00
7	7	1111341101	1279	1	1.34	40.00
8	8	1111341101	1279	4	1.34	40.00

Fig: Execute the package to load the data to Profit_Fact

Temporary Tables

BQ4_data:

We have used this temporary table to load data from the main ccount cleaned table in staging named **“ccount_clean” .** We have fetched DATE, MONTH, STORE, BAKERY, DAIRY, PHARMACY, COSMETIC, HABA, YEAR columns data for stores located in Naperville and Schaumburg during the year 1994 and loaded into BQ4_data temporary table.

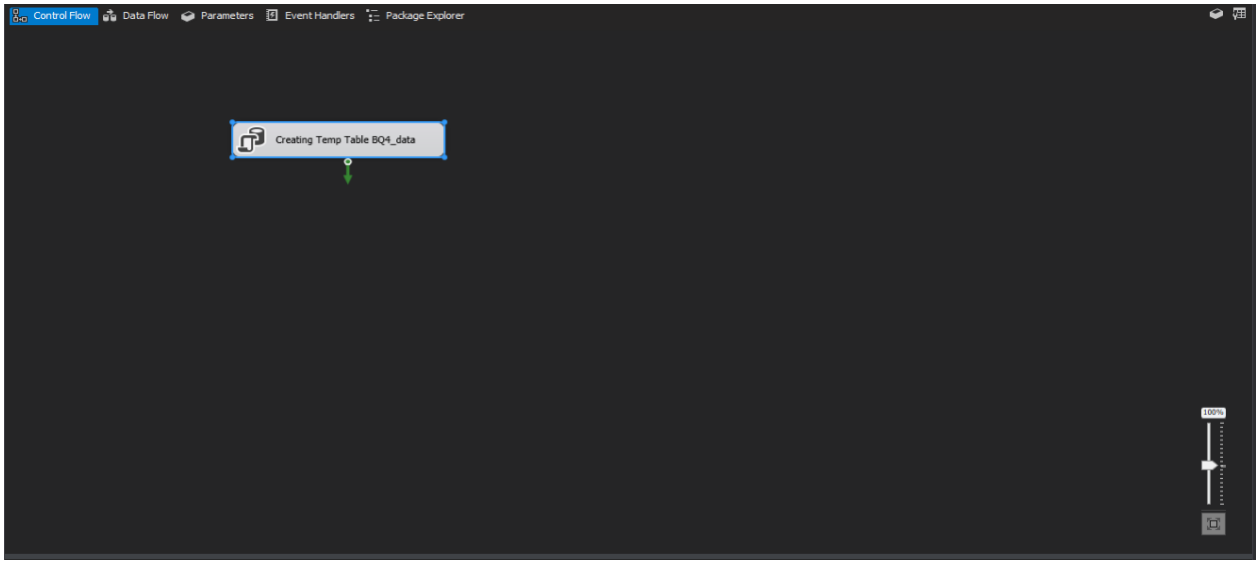


Fig: Execute SQL Task to create BQ4_data Temp table

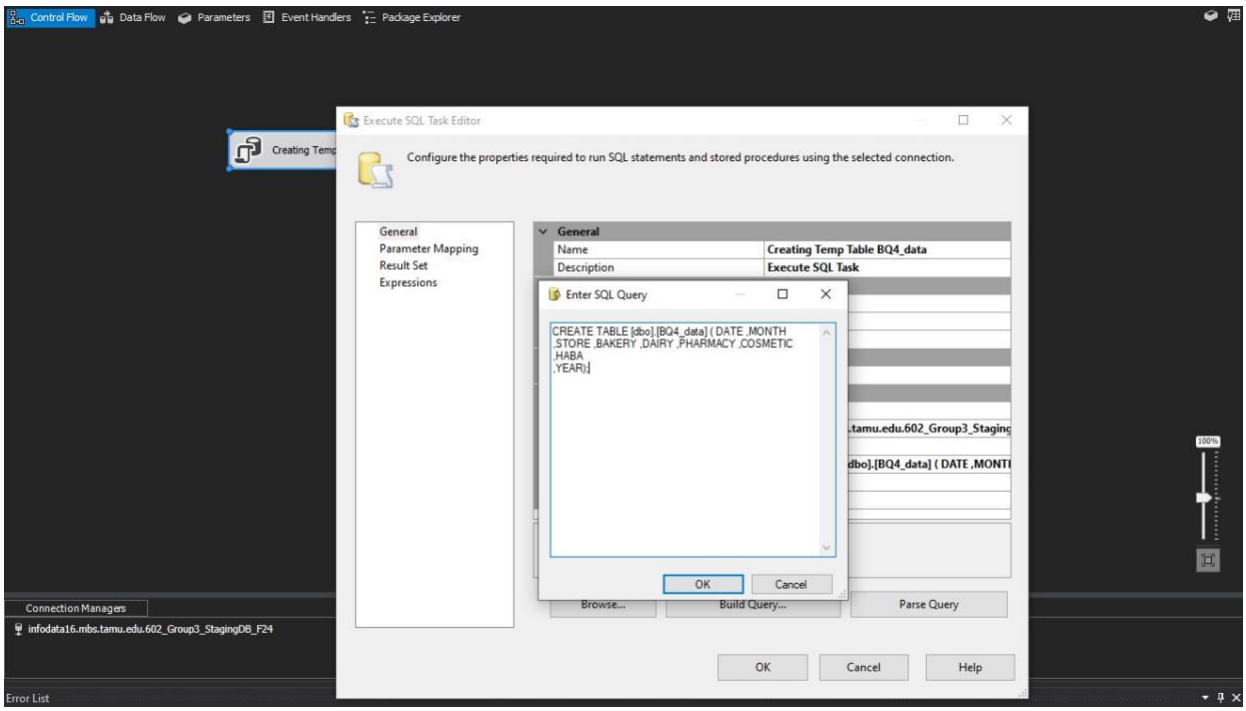


Fig: Create statement of BQ4_data temporary table

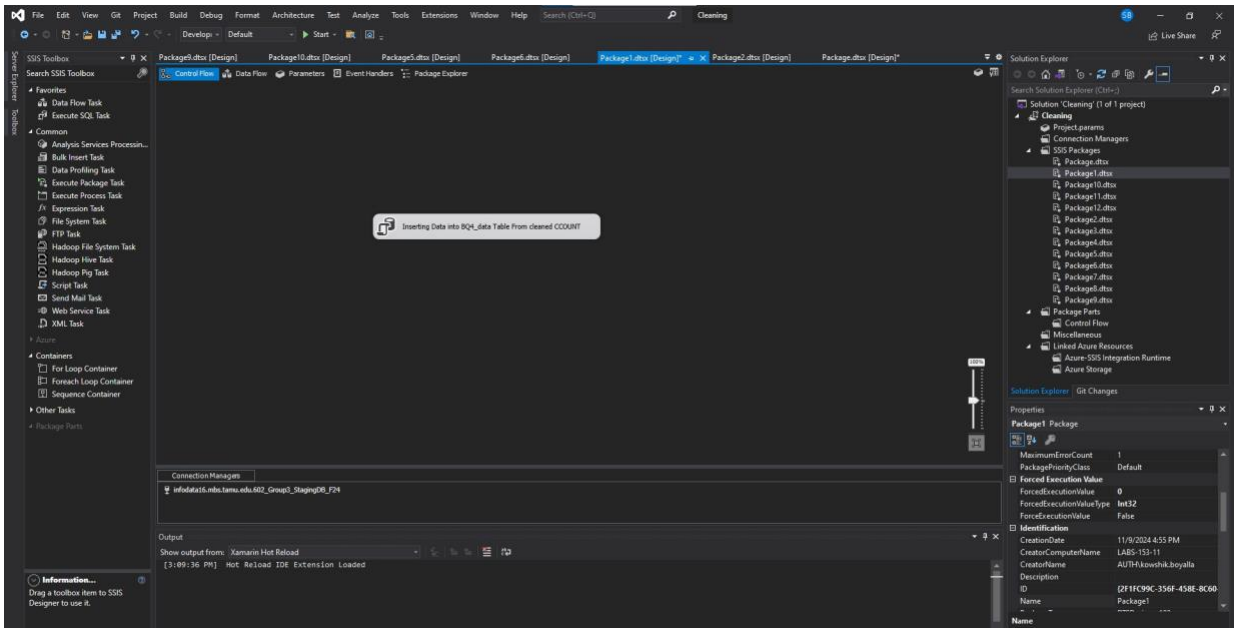


Fig: Execute SQL to insert data into BQ4_data

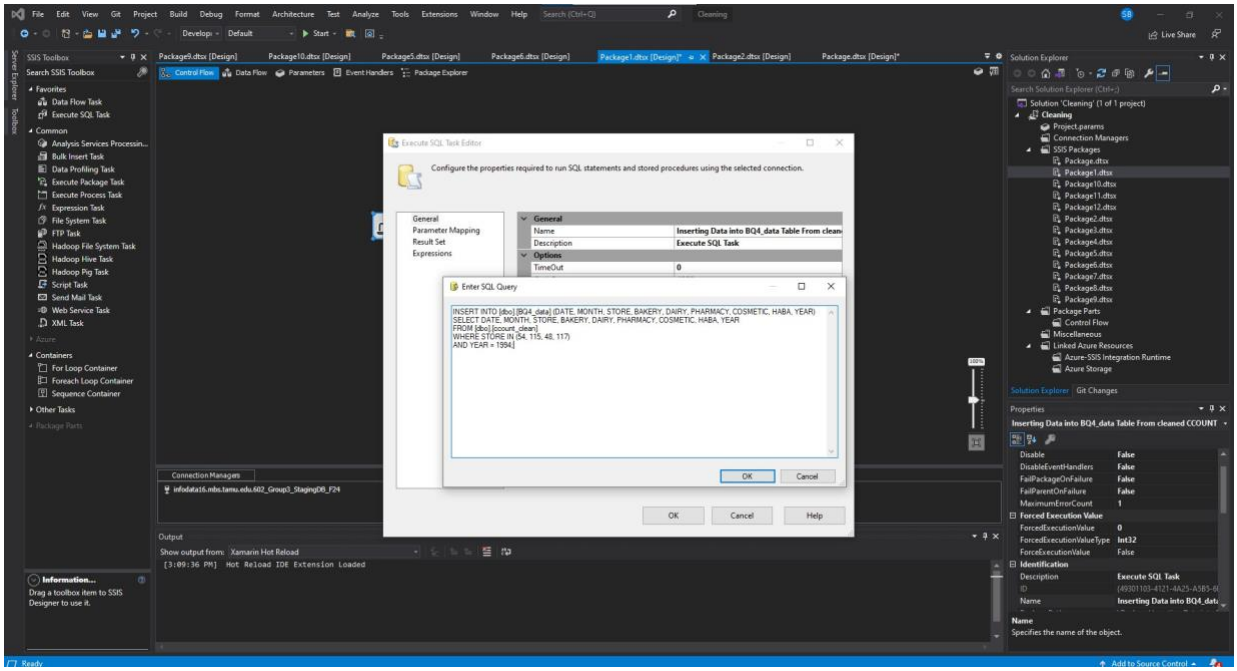
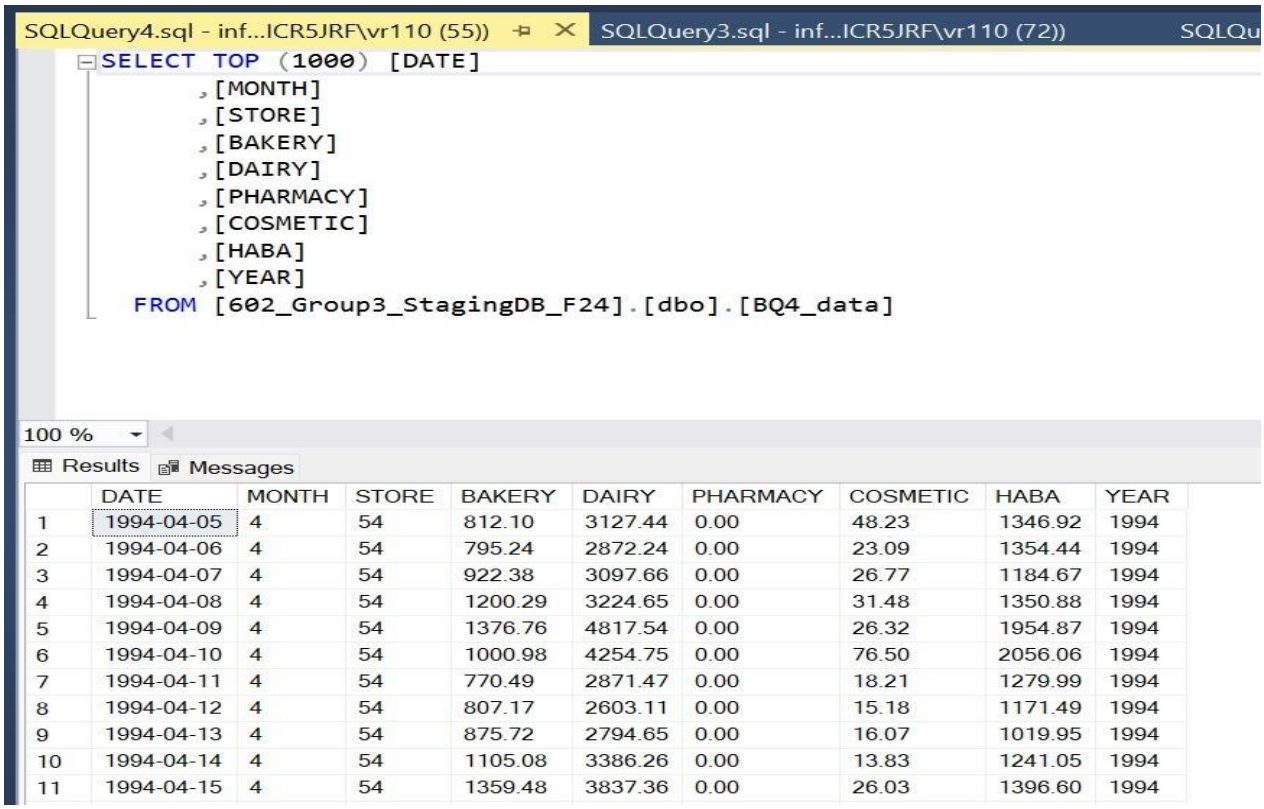


Fig: Inserting into BQ4_data Temporary table.



```
SELECT TOP (1000) [DATE]
      ,[MONTH]
      ,[STORE]
      ,[BAKERY]
      ,[DAIRY]
      ,[PHARMACY]
      ,[COSMETIC]
      ,[HABA]
      ,[YEAR]
  FROM [602_Group3_StagingDB_F24].[dbo].[BQ4_data]
```

100 %

Results | Messages

	DATE	MONTH	STORE	BAKERY	DAIRY	PHARMACY	COSMETIC	HABA	YEAR
1	1994-04-05	4	54	812.10	3127.44	0.00	48.23	1346.92	1994
2	1994-04-06	4	54	795.24	2872.24	0.00	23.09	1354.44	1994
3	1994-04-07	4	54	922.38	3097.66	0.00	26.77	1184.67	1994
4	1994-04-08	4	54	1200.29	3224.65	0.00	31.48	1350.88	1994
5	1994-04-09	4	54	1376.76	4817.54	0.00	26.32	1954.87	1994
6	1994-04-10	4	54	1000.98	4254.75	0.00	76.50	2056.06	1994
7	1994-04-11	4	54	770.49	2871.47	0.00	18.21	1279.99	1994
8	1994-04-12	4	54	807.17	2603.11	0.00	15.18	1171.49	1994
9	1994-04-13	4	54	875.72	2794.65	0.00	16.07	1019.95	1994
10	1994-04-14	4	54	1105.08	3386.26	0.00	13.83	1241.05	1994
11	1994-04-15	4	54	1359.48	3837.36	0.00	26.03	1396.60	1994

Fig: Validating Temporary Table in SSMS

BQ1_data:

We have used this temporary table to load data from the main ccount cleaned table in staging named **“ccount_clean”.** We have fetched the store_id and custCoun and Week which are needed for the business question

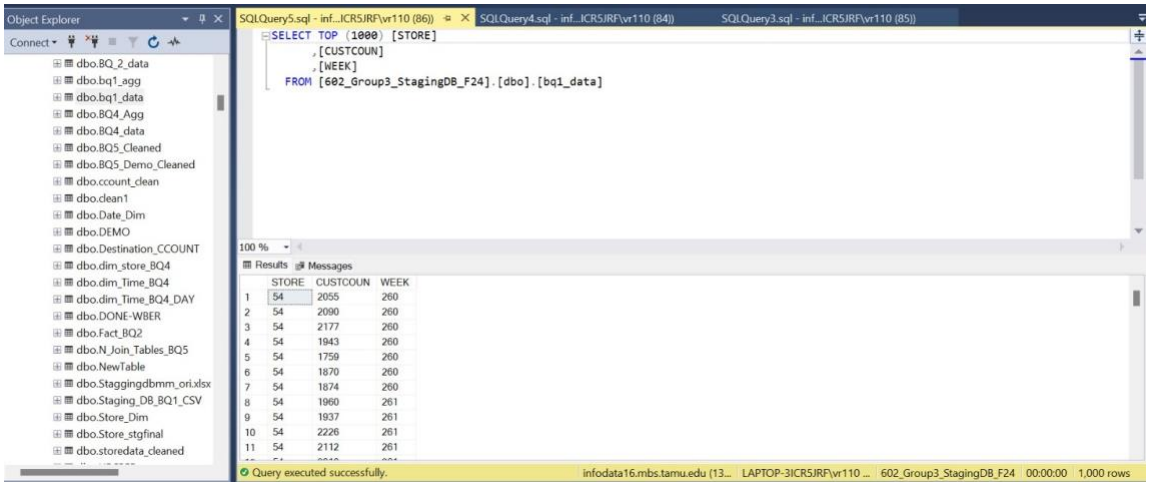


Fig: Validating Temporary Table in SSMS

BQ2_Data:

We have used this temporary table to load data from the main ccount cleaned table in staging named **“ccount_clean”.** We have fetched the STORE, DATE, GROCERY, WEEK, YEAR, MONTH, QUARTER, DAY,WEEK NAME which are needed for the business question into **BQ_2_data**

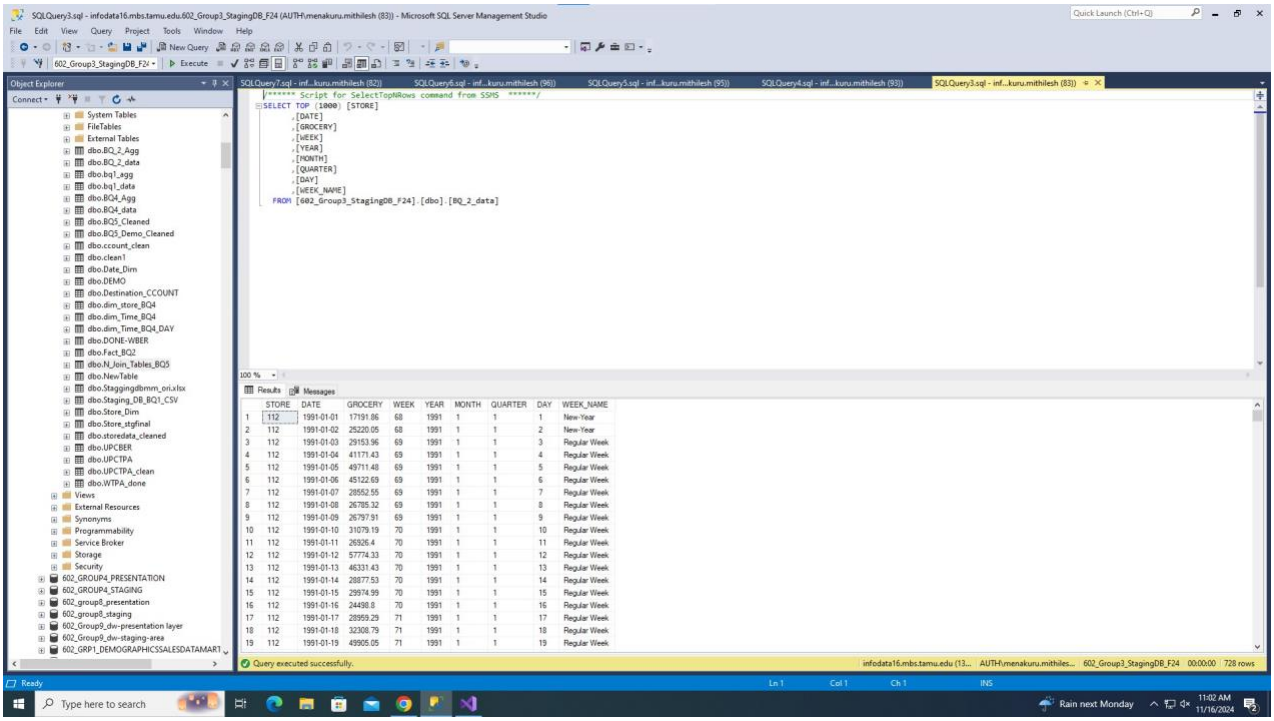
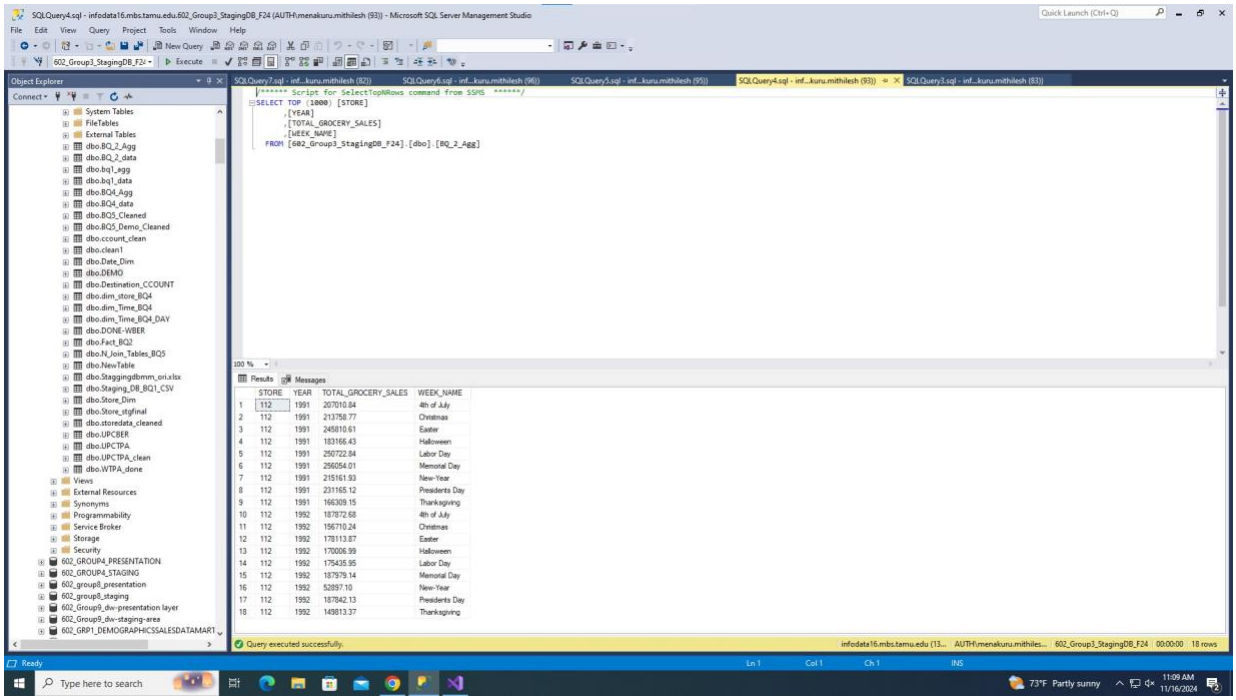


Fig Temp table for BQ_2_data

Then we used this table (**BQ_2_Agg**) as another TEMP table which is has Aggregated data init



BQ5_Data:

We have used this temporary table **BQ5_Demo_Cleaned** to load data from the main demo table in staging named **"DEMO".** We have fetched the MMID, NAME, CITY, STORE, SINGLES, RETIRED which are needed for the business question into **BQ_5_data**

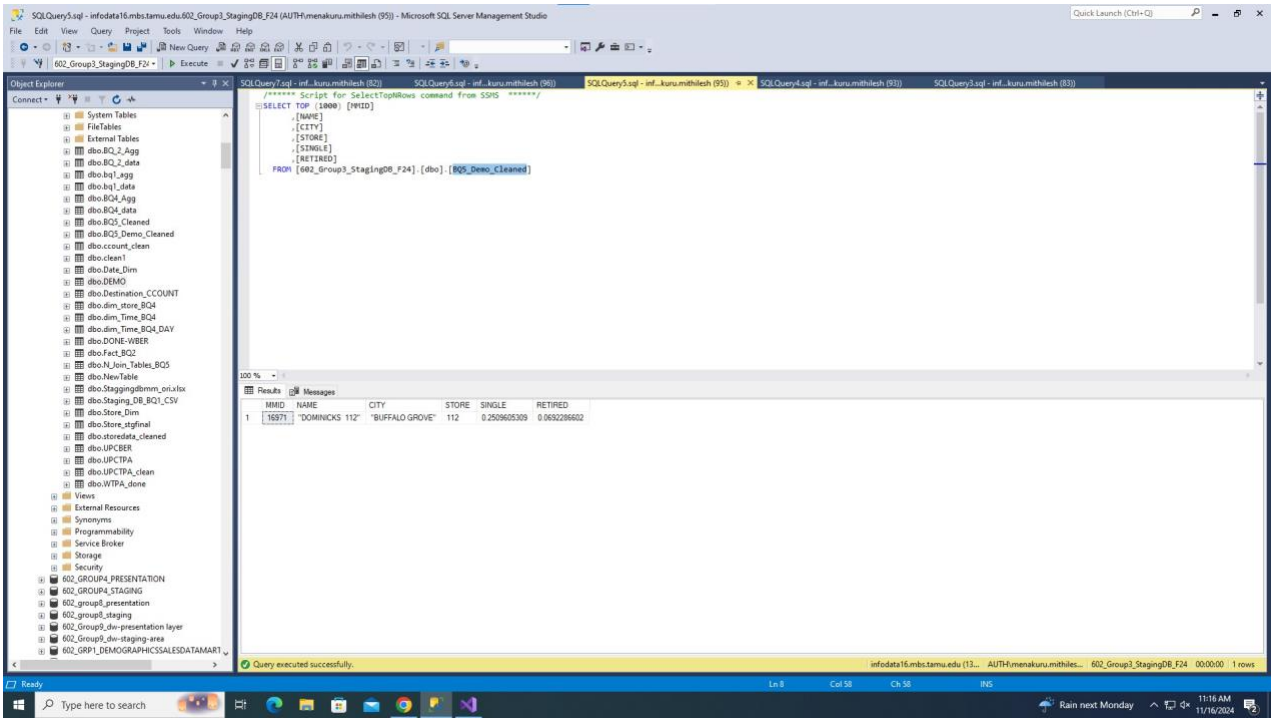
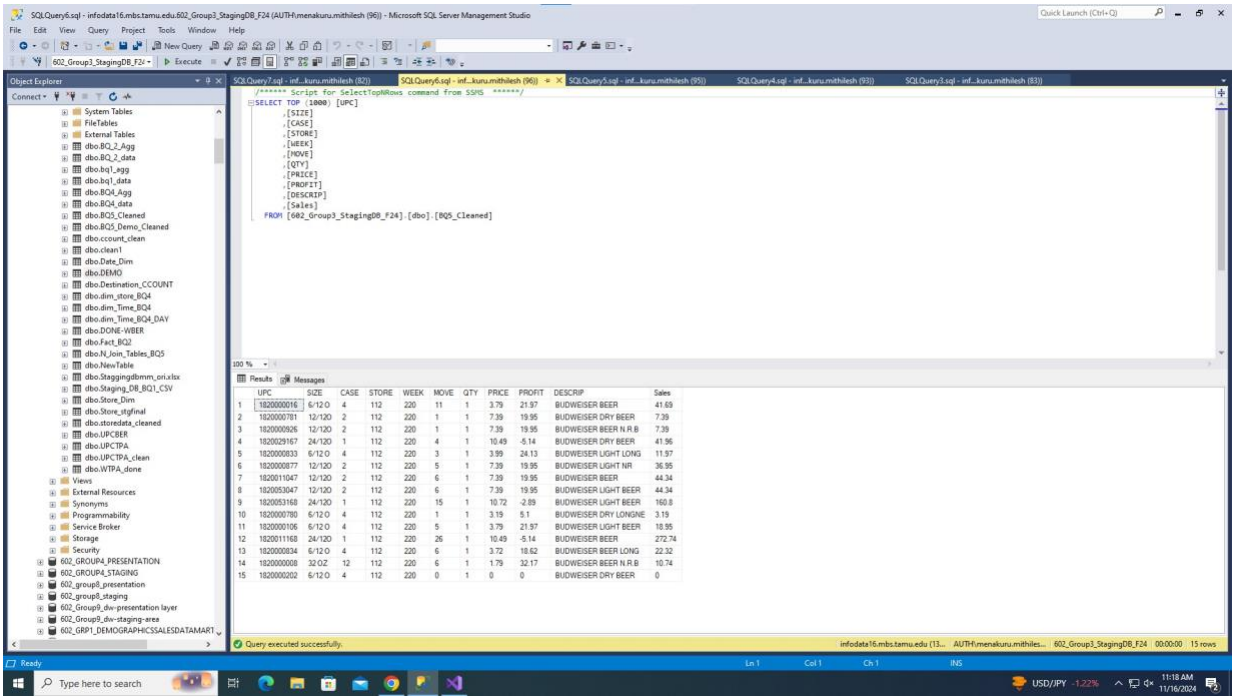


Fig Temp table BQ5_Demo_Cleaned

We have used another temp table **BQ5_Cleaned**, as this is an aggregated data table for the business question 5



Section 5: BI Reporting

Reporting Plan

a. Target Reports

The below table provides the details on the target reports satisfy the business questions

Target Reports	Business Questions	Justification
SSAS	Which store has the highest store traffic for the last quarter of 1994?	Cube using SSAS is used to answer the customer count related numbers across stores.
SSRS	Which holiday week in 1991 and 1992 saw the highest grocery sales in the low-tier Buffalo Grove stores?	To determine which holiday week in 1991 and 1992 saw the highest grocery sales in the low-tier Buffalo Grove stores using SQL Server Reporting Services (SSRS), you would create a report with specific filters and aggregations. Start by setting up parameters in SSRS to filter data by store tier (low-tier), location (Buffalo Grove), and the years 1991 and 1992. Then, group the filtered data by holiday weeks and aggregate the total grocery sales for each week. Finally, sort the results to identify the week with the highest sales. This approach leverages SSRS's dynamic filtering and grouping capabilities to deliver precise and actionable insights.
Microsoft PowerBI	How does the profit margin of toothpaste vary by brand?	Data Visualization capabilities of Power BI allows data integration from multiple sources and allows us to create DAX measures. The comparative charts allow us to see the Profit margin by specific brand and how they compare with each other.

Tableau with SSAS Cube	What are the monthly average sales amounts for the BAKERY, DAIRY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores located in Naperville and Schaumburg during the year 1994, and how do these averages compare among the different departments within these locations?	SSAS multidimensional cube is used to calculate the monthly average sales of different departments for each store and stored into a table with the help of a named query. Created relationships with store and time dimensions. Later connected Tableau with SSAS cube and created visualization such as Line Chart which indicates the monthly average sales of each department across all stores (a filter can be used to select a particular store out of 4). This will help stakeholders to recognize which department performed well in which month. Very useful to make comparisons between different departments' performance on a monthly basis.
SSRS	What were the highest sales contributions from single and retired individuals in the Buffalo Grove stores for **BUDWEISER BEER N.R.B** during the Thanksgiving week of 1993?	To analyze the highest sales contributions from single and retired individuals for products like **BUDWEISER BEER N.R.B** during Thanksgiving week of 1993 in Buffalo Grove stores, filter the data to the specific week, location, and product. Segment the sales by customer demographics to isolate data for single and retired individuals. Aggregate the sales for each group and identify their contributions to the total. This approach provides clear insights into demographic-specific purchasing behavior for the given product.

b. Mappings from the independent data marts to the report attributes

Business Question 1:

Datamart Attribute	Table	Filters	Report Attribute

Store ID	Store_Dim	None	Store ID
Customer Count	Store_Traffic_Fact	Date_Key referencing Date_Dim	Customer Count

Business Question 2:

Datamart Attribute	Table	Filters	Report Attribute
Date	Date_Dim	None	Date
Year	Date_Dim	None	Year
Week	Date_Dim	None	Week
Total_Grocery_Sales	Fact_T_BQ2	None	Total_Grocery_Sales
Week_Name	Fact_T_BQ2	None	Week_Name
Store_Id	Store_Dim	None	Store_Id

Business Question 3:

Datamart Attribute	Table	Filters	Report Attribute
BRAND_NAME	Brand_Dimension	N/A	Brand_Name
Profit	Profit_Fact	BRAND_ID	Avg_Profit_Margin
UPC	Product_Dimension	N/A	Product Code

Fig: Mapping Business Question 3

Business Question 4:

Datamart Attribute	Table	Filters	Report Attribute
Store_Id	dim_store	Store_Id	Store_Id
MONTH	dim_time	N/A	Month
Bakery_Sales	Fact_Sales	N/A	Avg_Bakery_Sales
Diary_Sales	Fact_Sales	N/A	Avg_Diary_Sales

Pharmacy_Sales	Fact_Sales	N/A	Avg_Pharmacy_Sales
Cosmetic_Sales	Fact_Sales	N/A	Avg_Cosmetic_Sales
HABA_Sales	Fact_Sales	N/A	Avg_HABA_Sales

Business Question 5:

Datamart Attribute	Table	Filters	Report Attribute
Sales	Customer_Demographics_Fact	N/A	Sales
Week	Date_Dim	N/A	Week
Store	Demography_Dim_BQ5	N/A	Store
Single	Demography_Dim_BQ5	N/A	Single
Retired	Demography_Dim_BQ5	N/A	Retired
Product_Name	Product_Dim_BQ5	N/A	Product_Name

c. Report Templates

In the reporting phase, we utilized diverse business intelligence tools to create interactive and insightful reports, addressing key business questions for DFF. The primary goal is to enable DFF to easily understand business trends and performance metrics by leveraging the information stored in fact and dimension tables. Through this, we has gained the proficiency in using different tools to visualize data:

SSRS - Microsoft SQL Server Reporting Services :

This is a server based platform that is used for creating reports and visualizations. It has a closer association with MS Visual Studio and Microsoft SQL formatting tools because of which it uses Ad-hoc and drill down reports seamlessly for the SQL Server Data warehouse.

SSAS - Microsoft SQL Server Analysis Services:
Microsoft SQL Server Analysis Services uses a analytical data engine to help in business analytics and decision support. It is implemented by deploying a multidimensional model as a new database to the server instance . The multidimensional model consists of cubes having measures for data aggregation and can be represented as a report that can be deployed in Analytics Services Project.

Microsoft Power BI
Microsoft Power BI is a powerful business intelligence tool that enables users to transform, and visualize data from multiple sources. It provides insights through interactive reports and dashboards. Power BI's visualization options, such as charts, graphs, and maps, allow users to present data in a visually engaging manner, enabling better decision-making.

Tableau
Tableau is a strong business intelligence and data visualization tool that helps users to connect to the data, analyze data and create interactive reports and visualizations. It has a easy drag and drop that makes it easy to analyze complex data and identify insights and analysis.

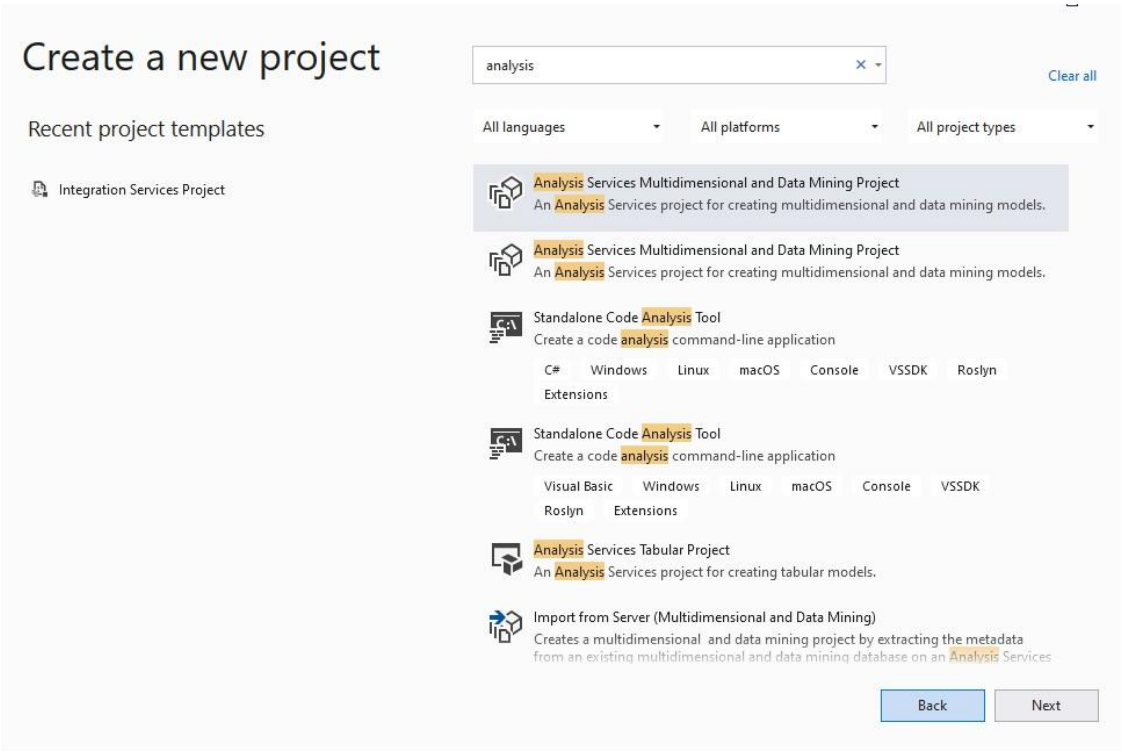
Report Implementation

1. Which store has the highest store traffic for the last quarter of 1994?

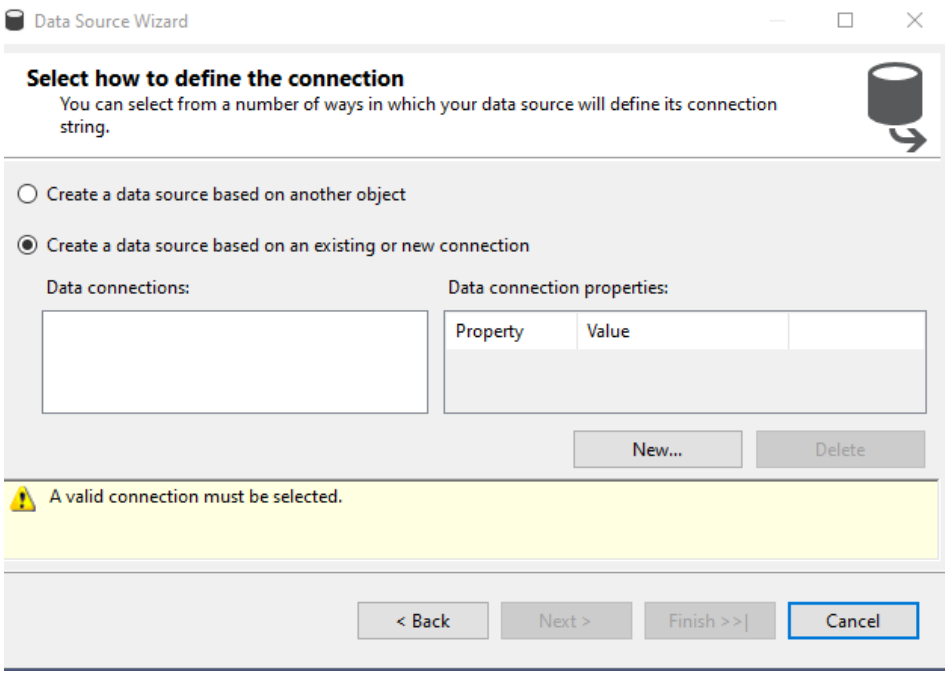
Support : Store traffic is a key indicator of the overall business performance and sales. By identifying the highest-traffic store, DFF can analyze what contributed to its success and replicate the same across other locations. The focus on the last quarter of the year is important as it's the holiday season, and the insights from this can be used to improve the performance across other stores as well.

BI Visualization tool used - SSAS

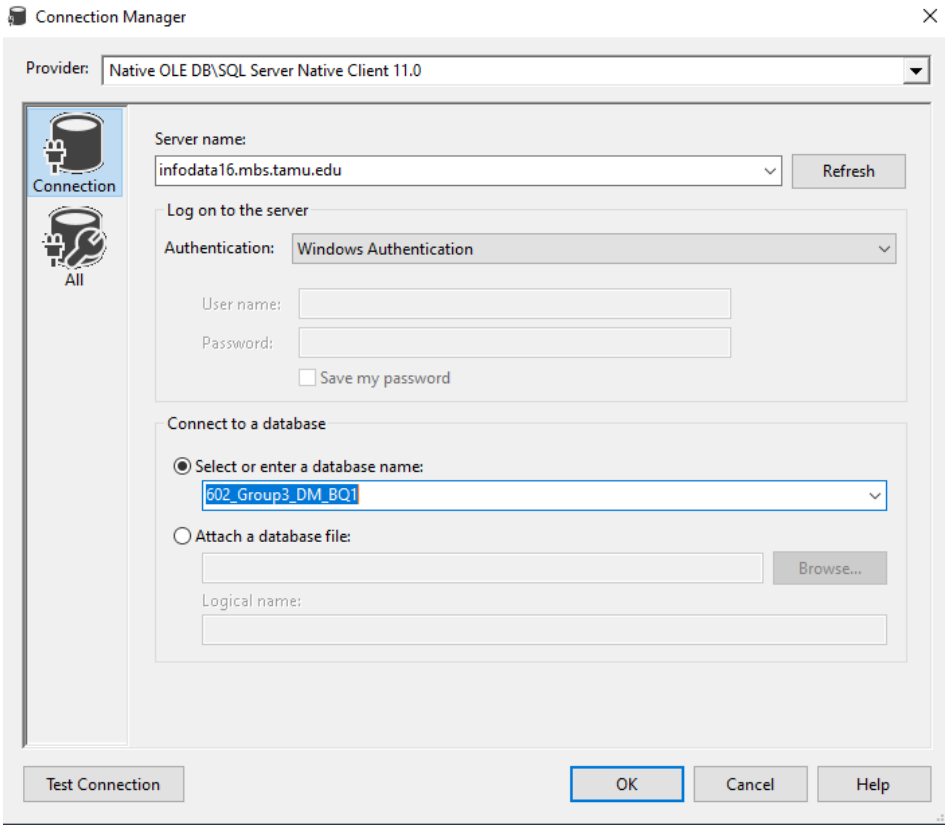
Step 1 : Create a new Analysis Service Multidimensional and data mining project



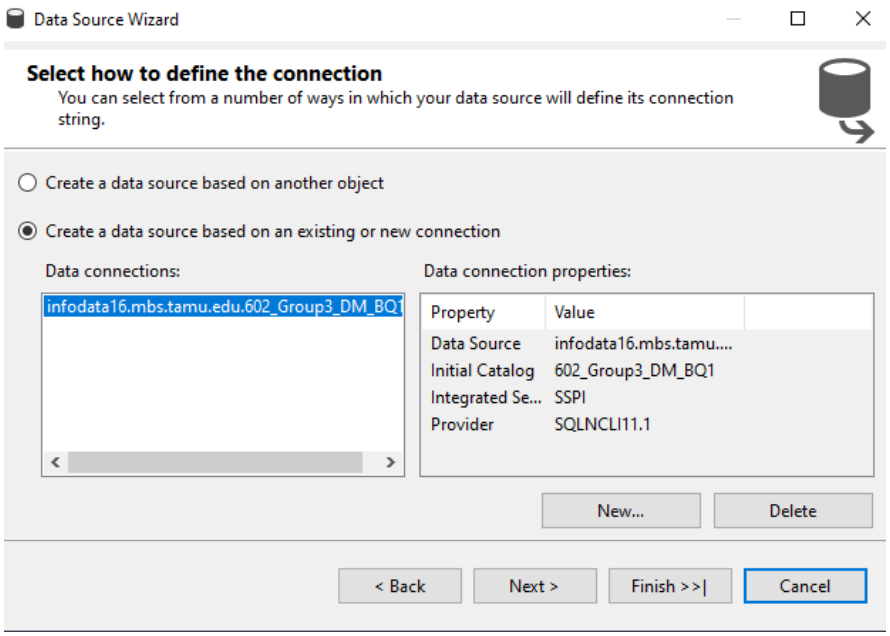
Step 2 : Create a new data source and select the option to create a database based on existing connection



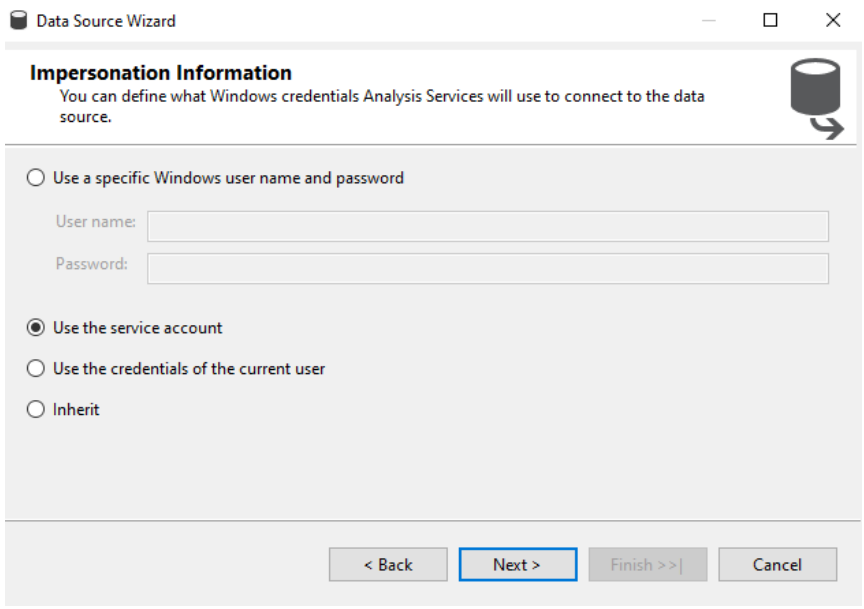
Step 3 : Enter the servername and select the data warehouse



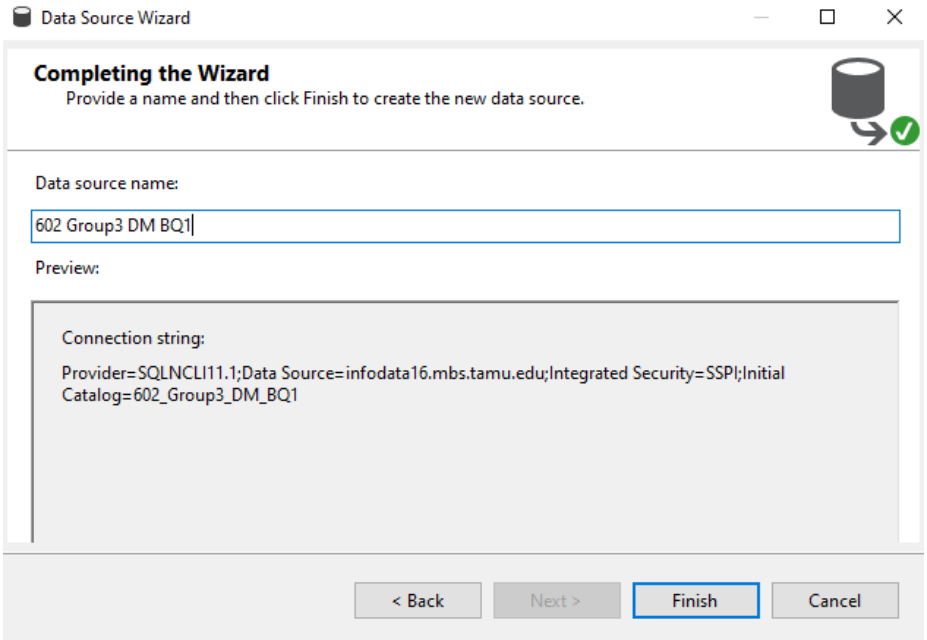
Step 4 : Select the newly added connection



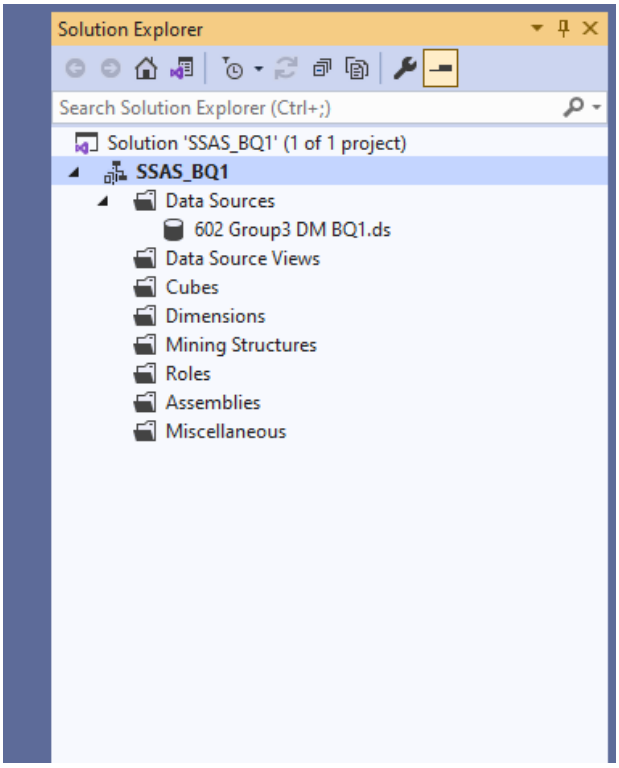
Click on “Use the service account” for impersonation information



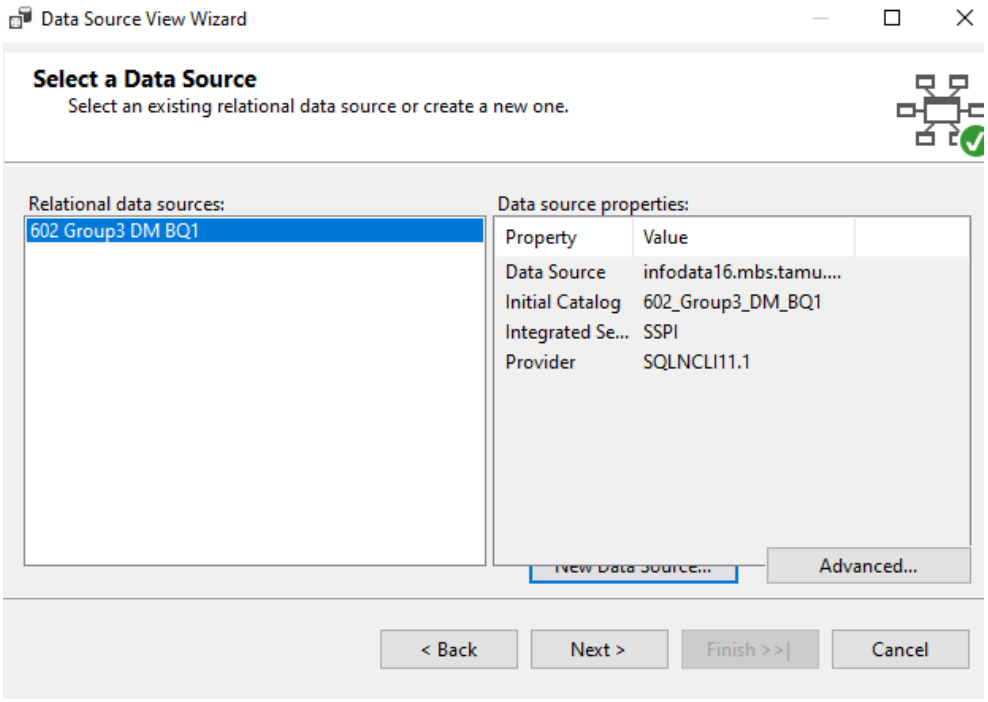
Step 5 : Add the name for data source and enter finish



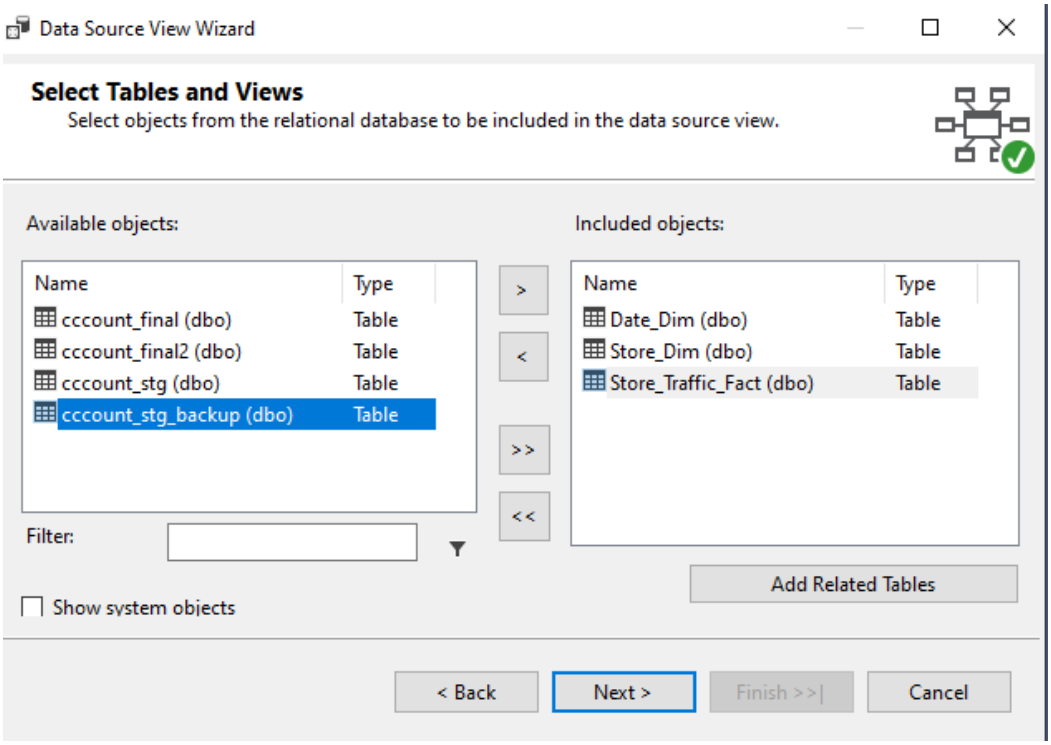
The project is visible on the Solution Explorer



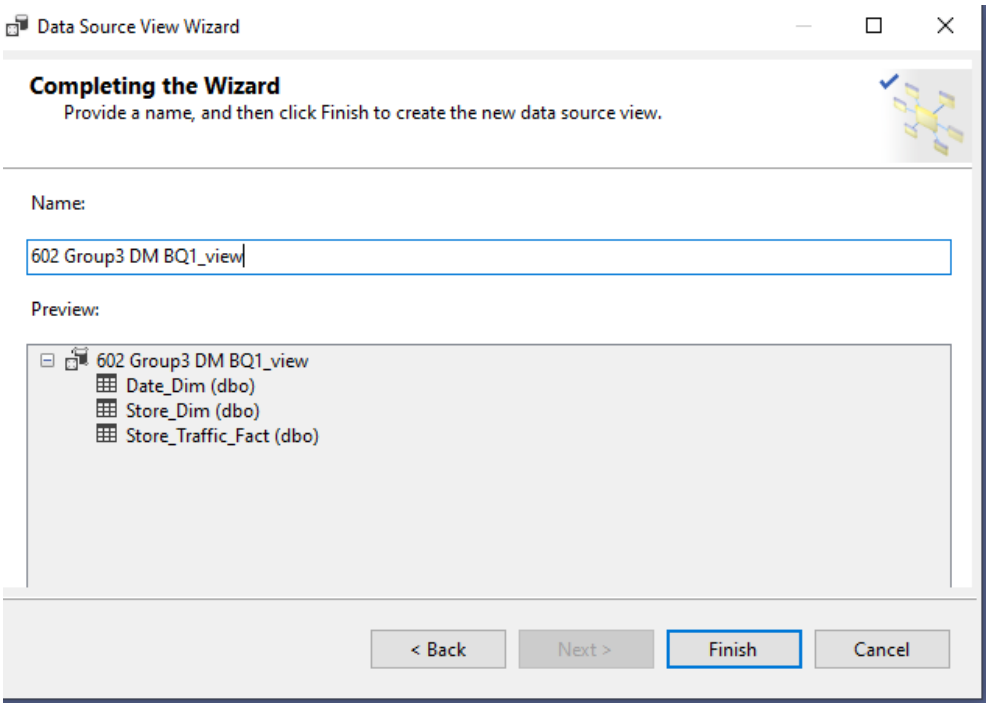
Step 6 : Create a Data Source View



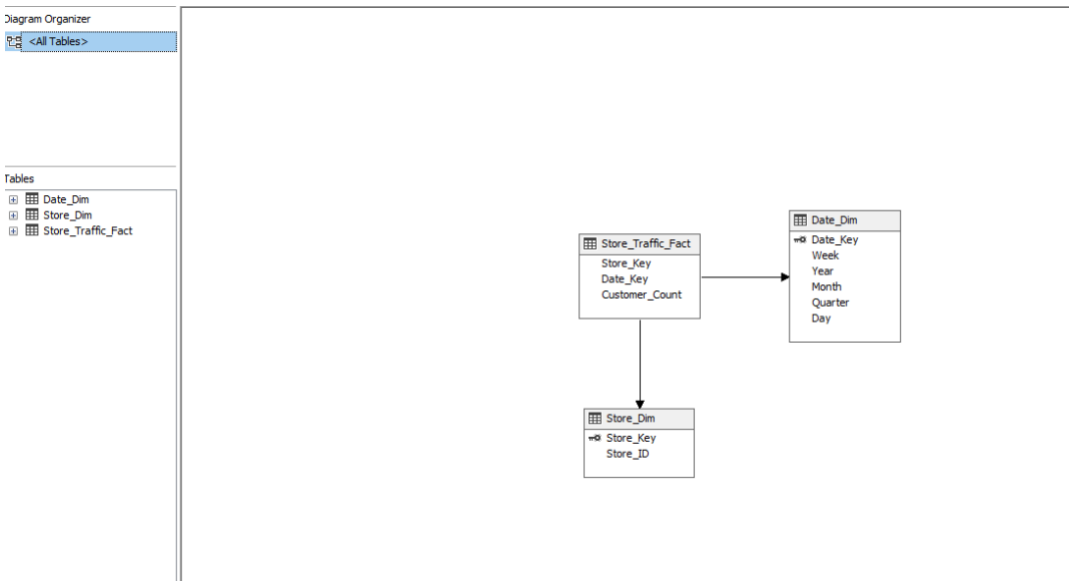
Step 7 : Select all the tables needed



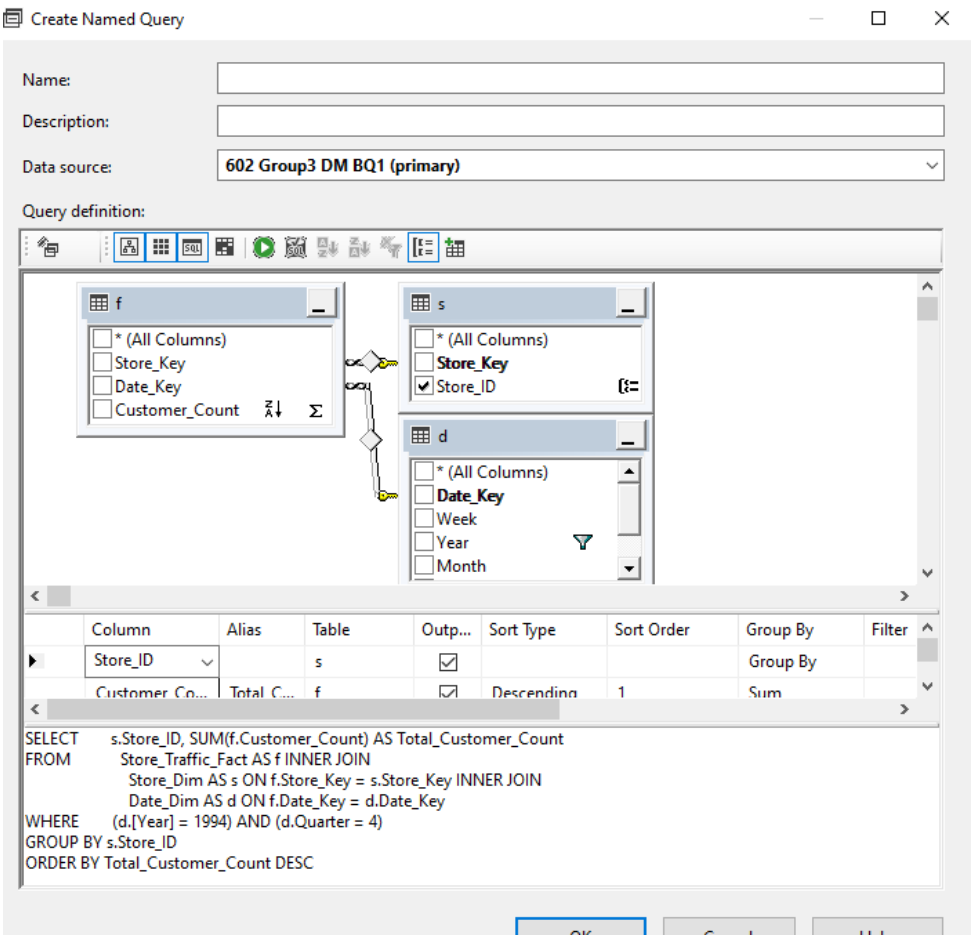
Step 8 : Name the view and clik Finish



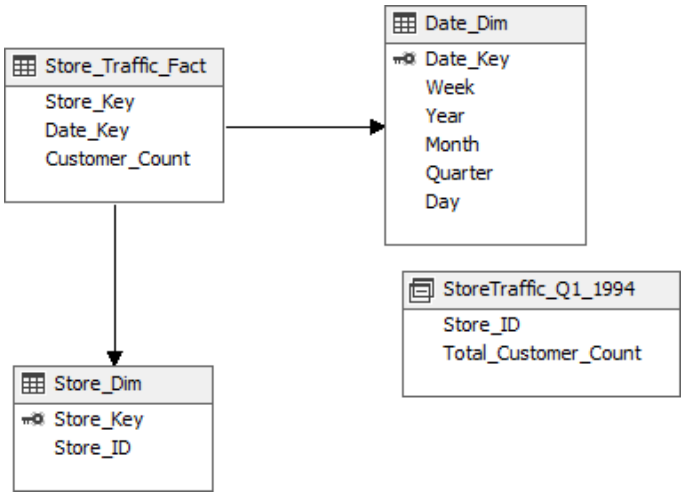
The relationship are as below

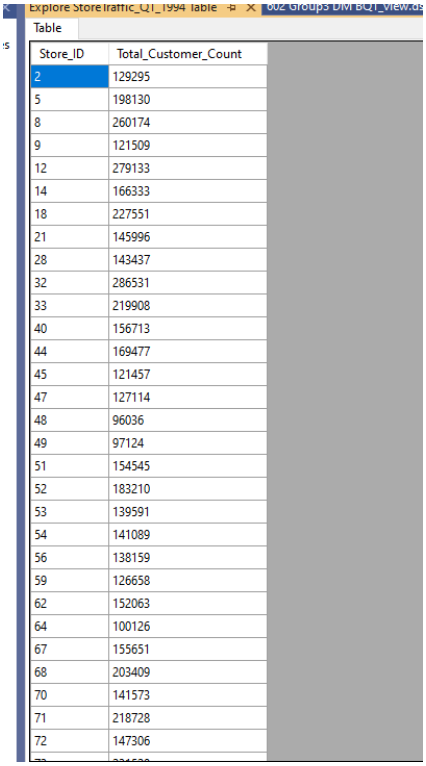


Step 9 : Select Create a new Named Query



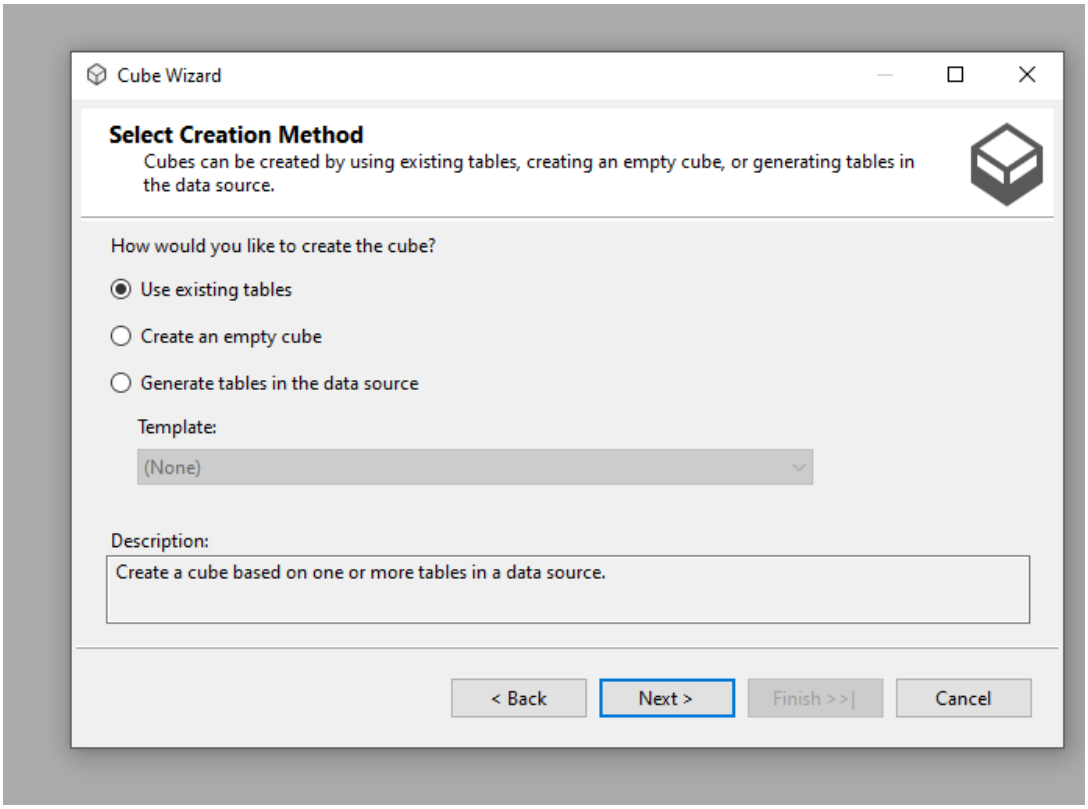
Step 10 : Explore the data



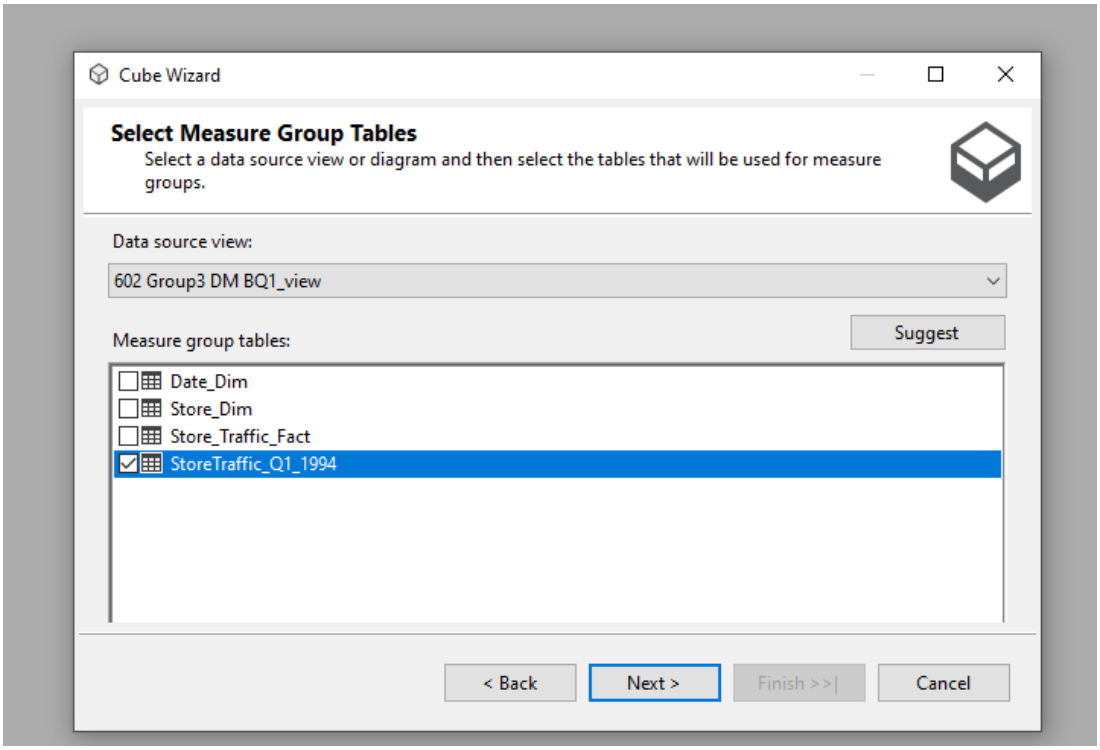


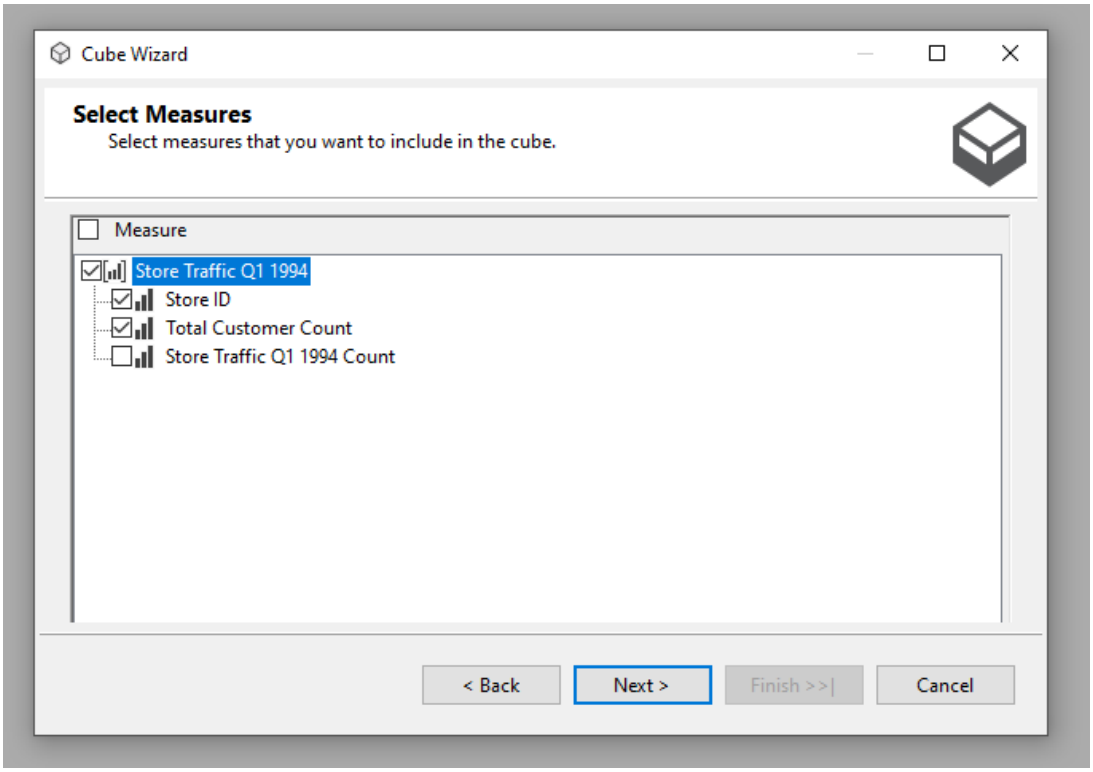
Creation of cube

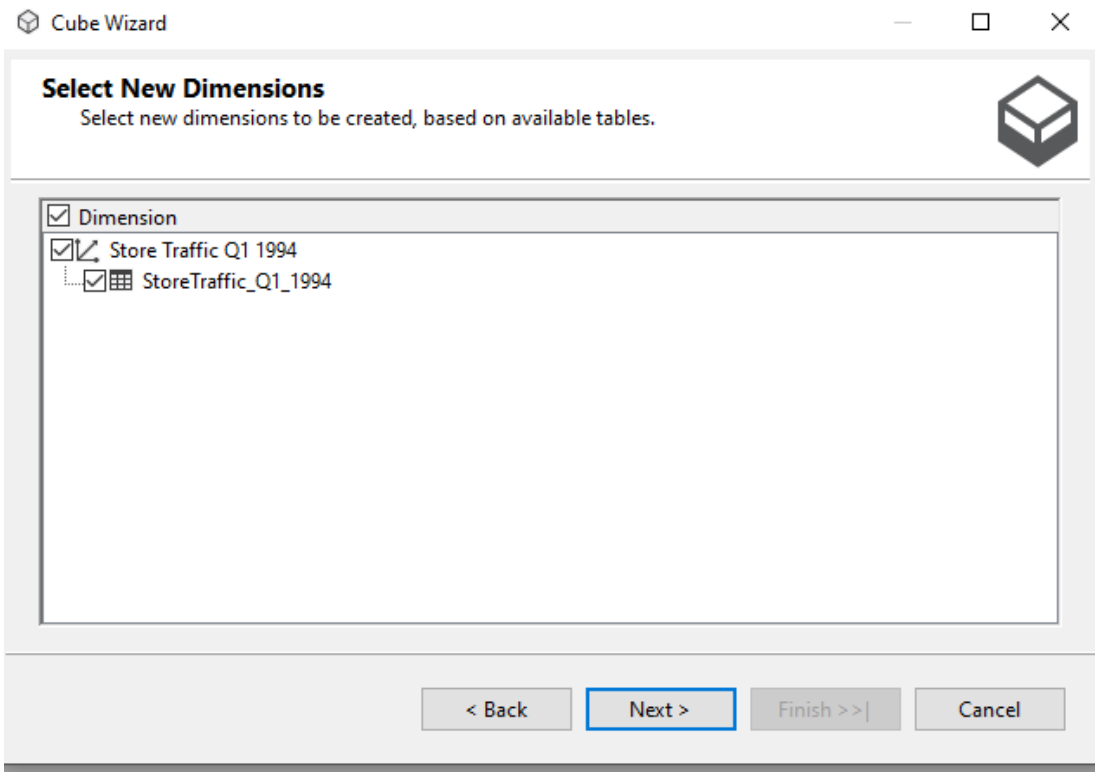
Step 1 : When creating a cube select “Use Existing Table”

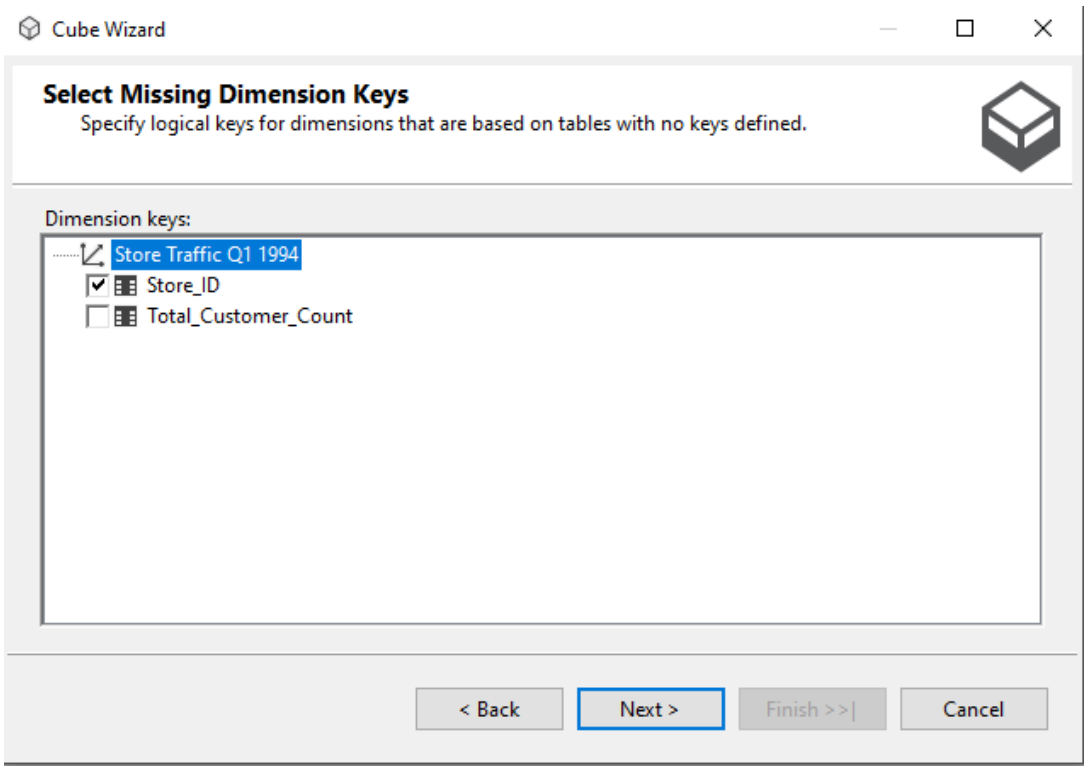


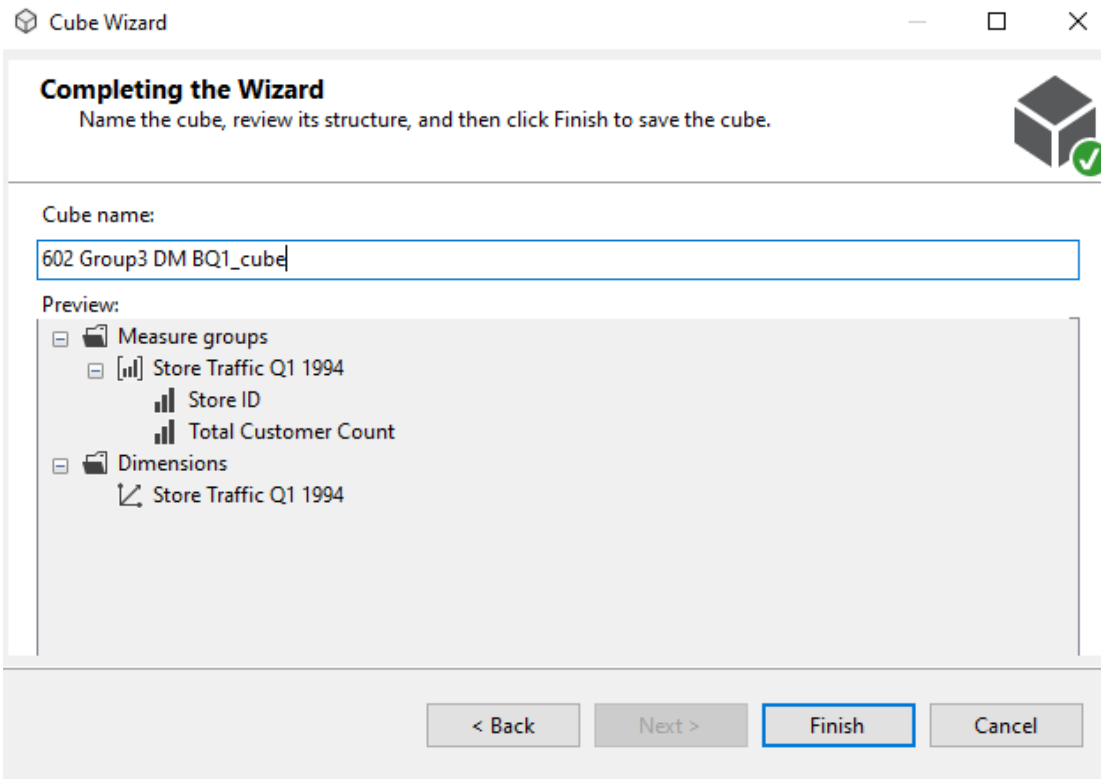
Step 2 : Select the measure and dimension key





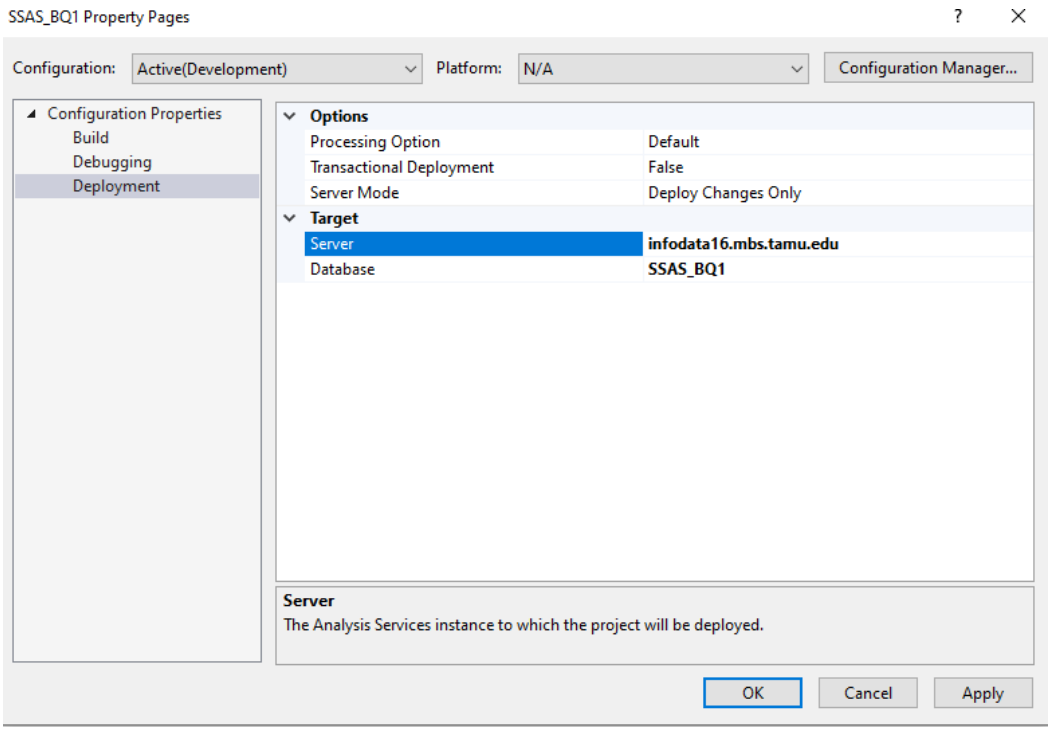

Cube Wizard
Select New Dimensions
Select new dimensions to be created, based on available tables.
Dimension
Store Traffic Q1 1994
StoreTraffic_Q1_1994
< Back
Next >
Finish >>|
Cancel


Cube Wizard
Select Missing Dimension Keys
Specify logical keys for dimensions that are based on tables with no keys defined.
Dimension keys:
Store Traffic Q1 1994
Store_ID
Total_Customer_Count
< Back
Next >
Finish >>|
Cancel

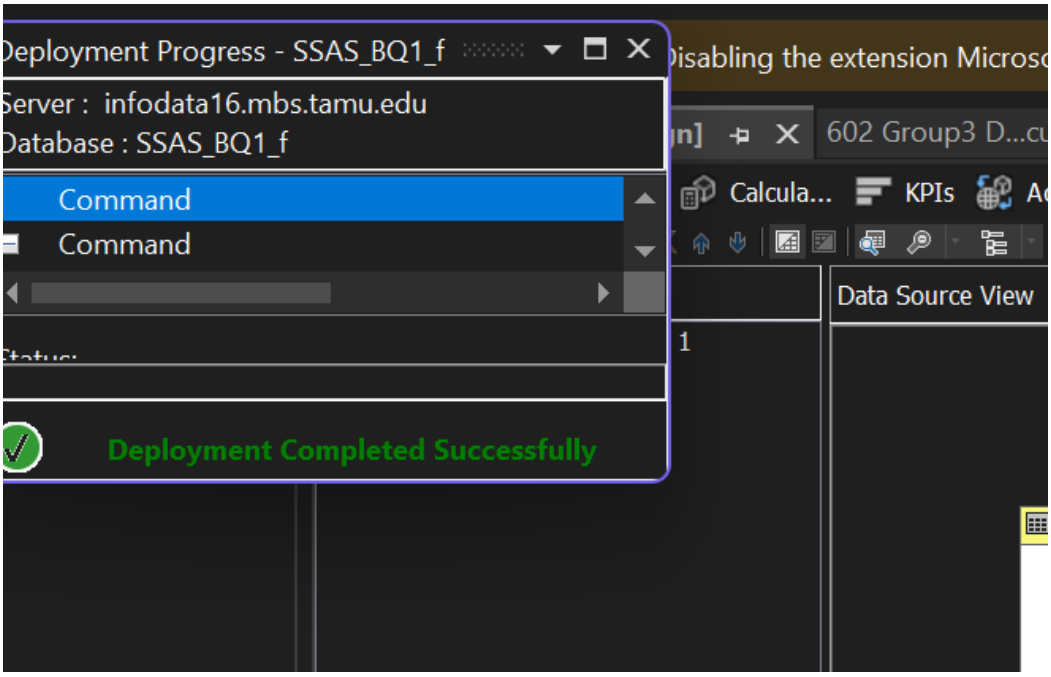


Deployment of Project to the server

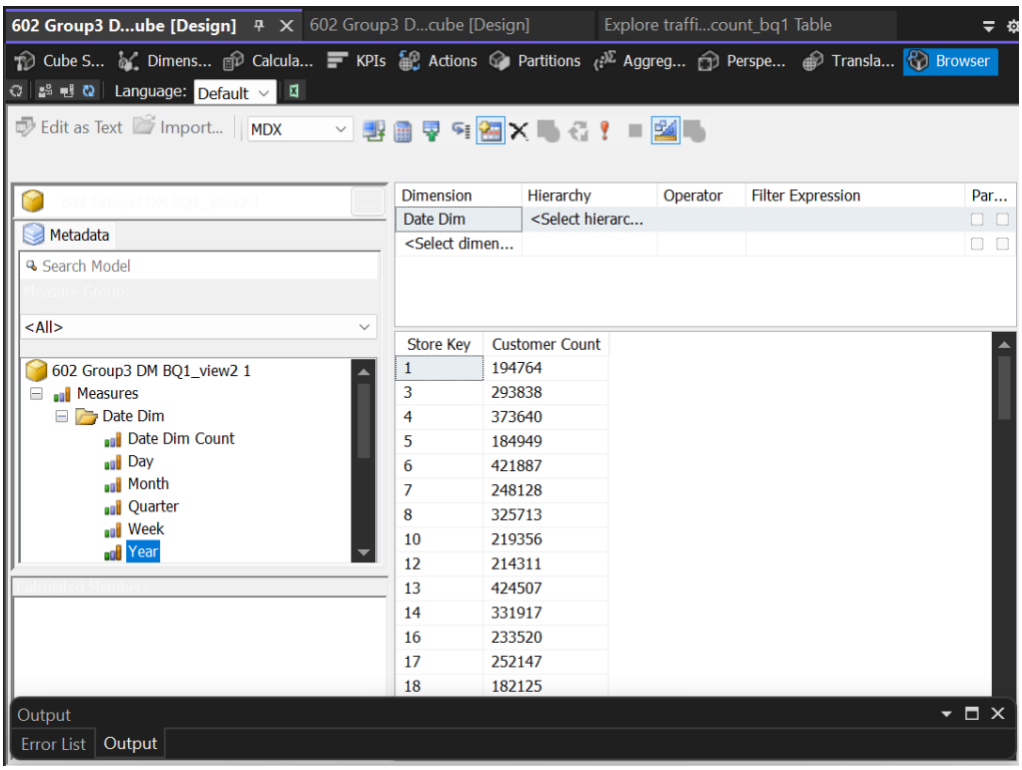
Step 1 : Entering the infodata16.mbs.tamu.edu and the database name



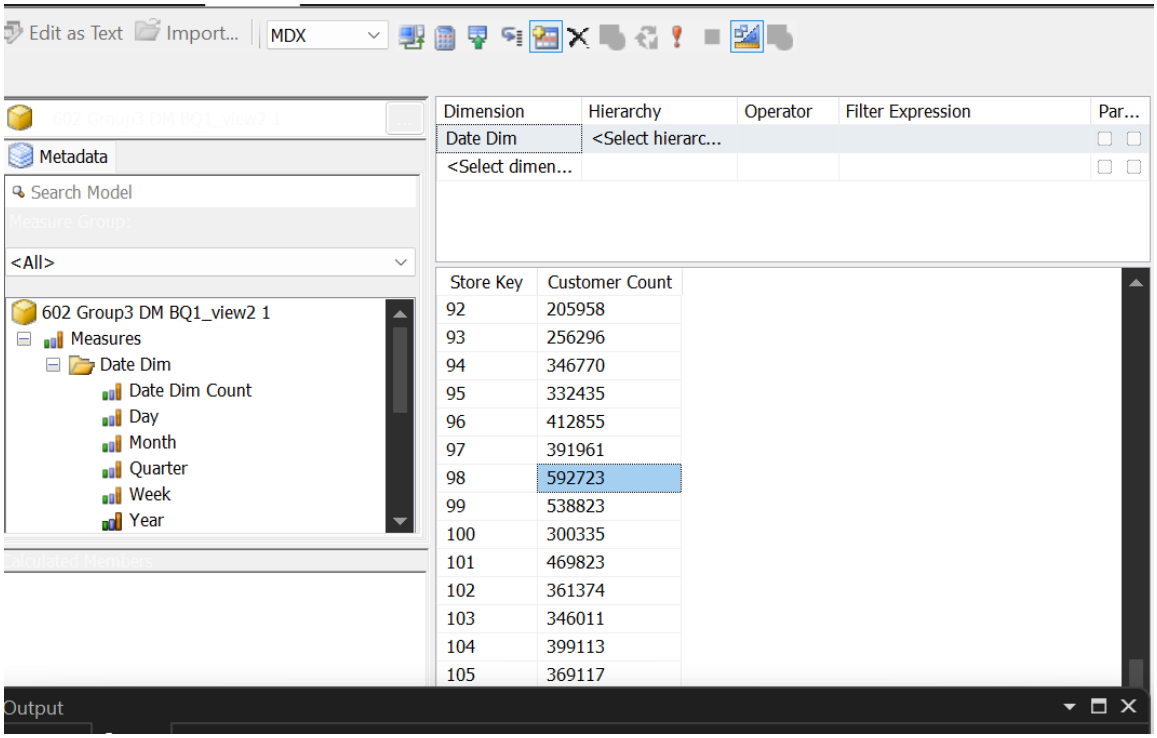
Step 2 : Deploying the project



Step 3 : Browsing the cube to create the report



Step 4: We can see store 98 ie Store 303 having the highest sale



2. Which holiday week in 1991 and 1992 saw the highest grocery sales in the low-tier Buffalo Grove stores?

Support: This analysis is valuable for **Dominick's Fine Foods (DFF)** as it helps identify peak grocery sales periods in low-tier stores during key holidays. By pinpointing which holiday weeks in 1991 and 1992 had the highest sales, DFF can optimize future inventory management, staffing, and promotions for low-tier locations. This allows DFF to better align supply with demand, minimizing stockouts while maximizing revenue during high-traffic holiday periods

Method: The report created to answer this business question from independent Data Marts using SSRS is employed here. This report is published to infodata16.mbs.tamu.edu Report Server

SSRS :

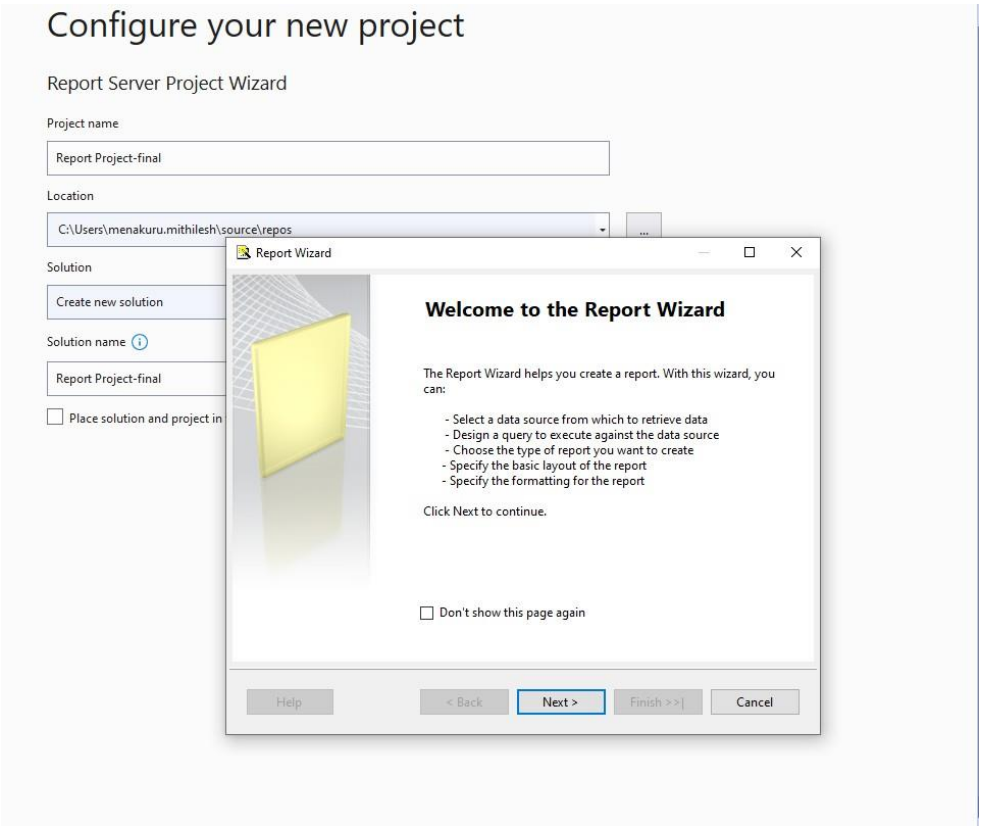


Fig Selecting Report Wizard through SSIS

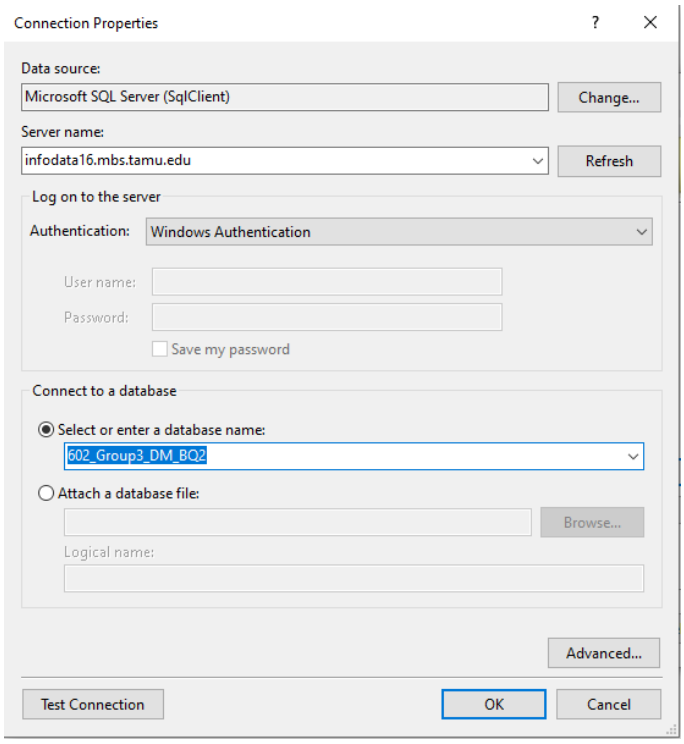


Fig Selecting Selectinfodata16.mbs.tamu.edu server

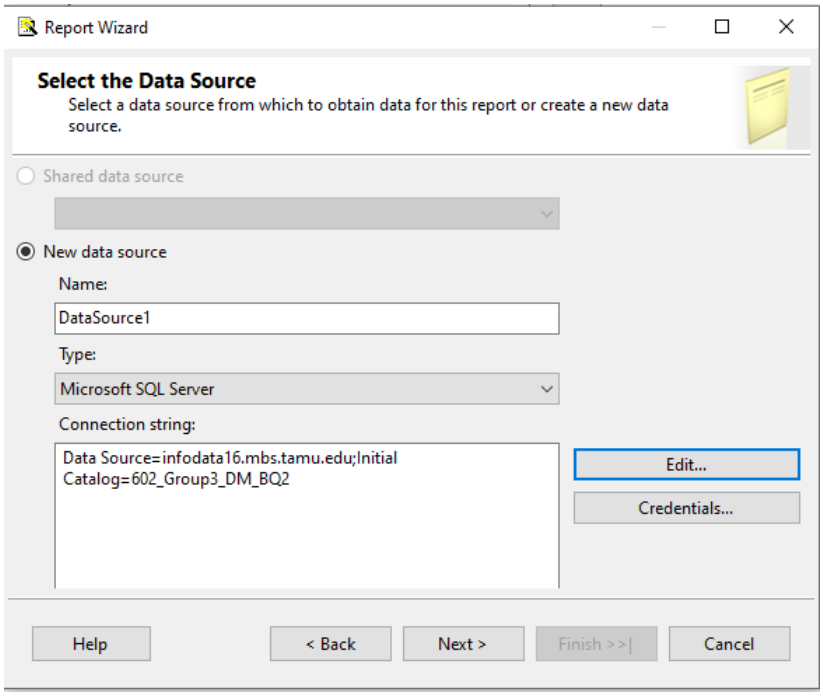


Fig Data Source and Catalog String

QUERY :

```
SELECT
        Fact_T_BQ2. Total_Grocery_Sales,
        Fact_ T_BQ2.Week_ Name,
        Date_Dim.YEAR,
        Date_Dim.DATE,
        Store_Dimen.Store_ID,
        Date_Dim.DAY,
        Date_Dim.WEEK
FROM
        Date_Dim
INNER JOIN
        Fact_ T_BQ2
ON
        Date_Dim.DateKey = Fact_T_BQ2.Date_Key
INNER JOIN
        Store_Dimen
ON
        Fact_T_BQ2.Store_Key = Store_Dimen. StoreKey
```

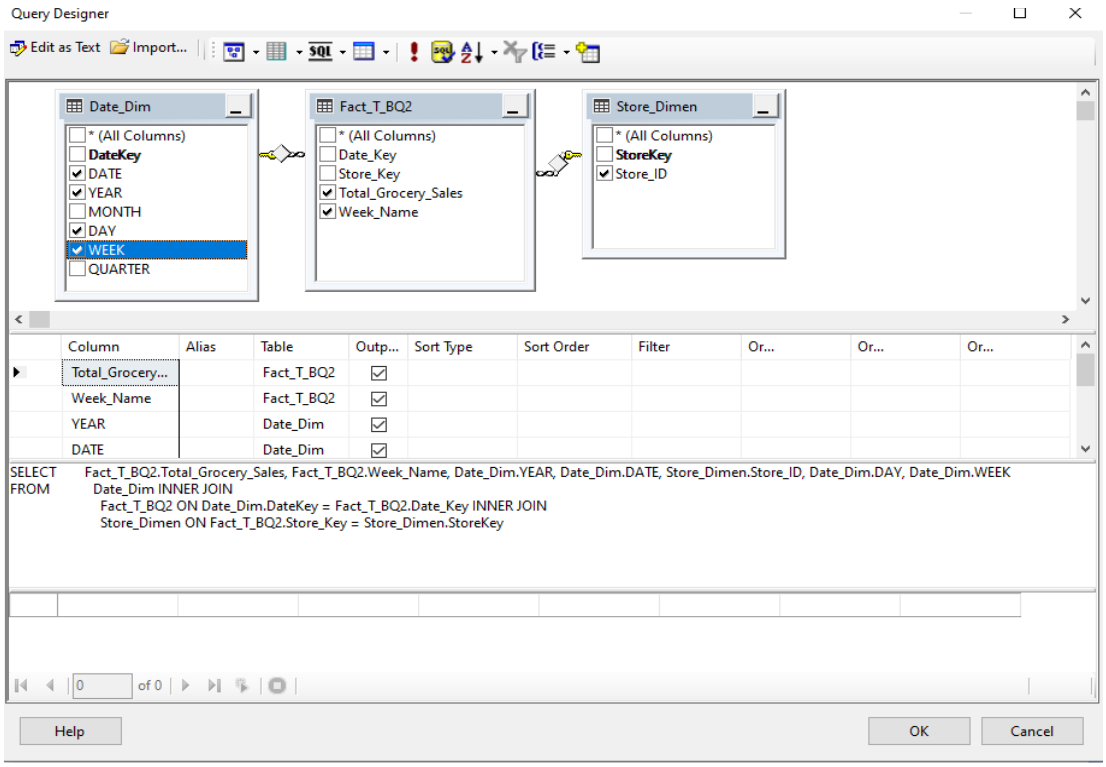


Fig Query Builder

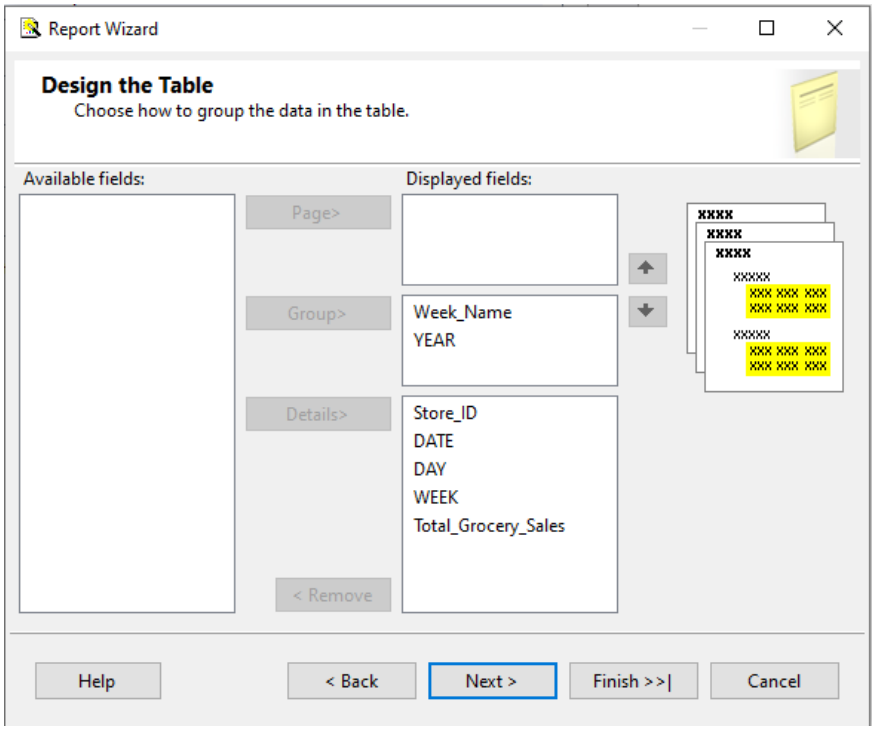


Fig Designing the table

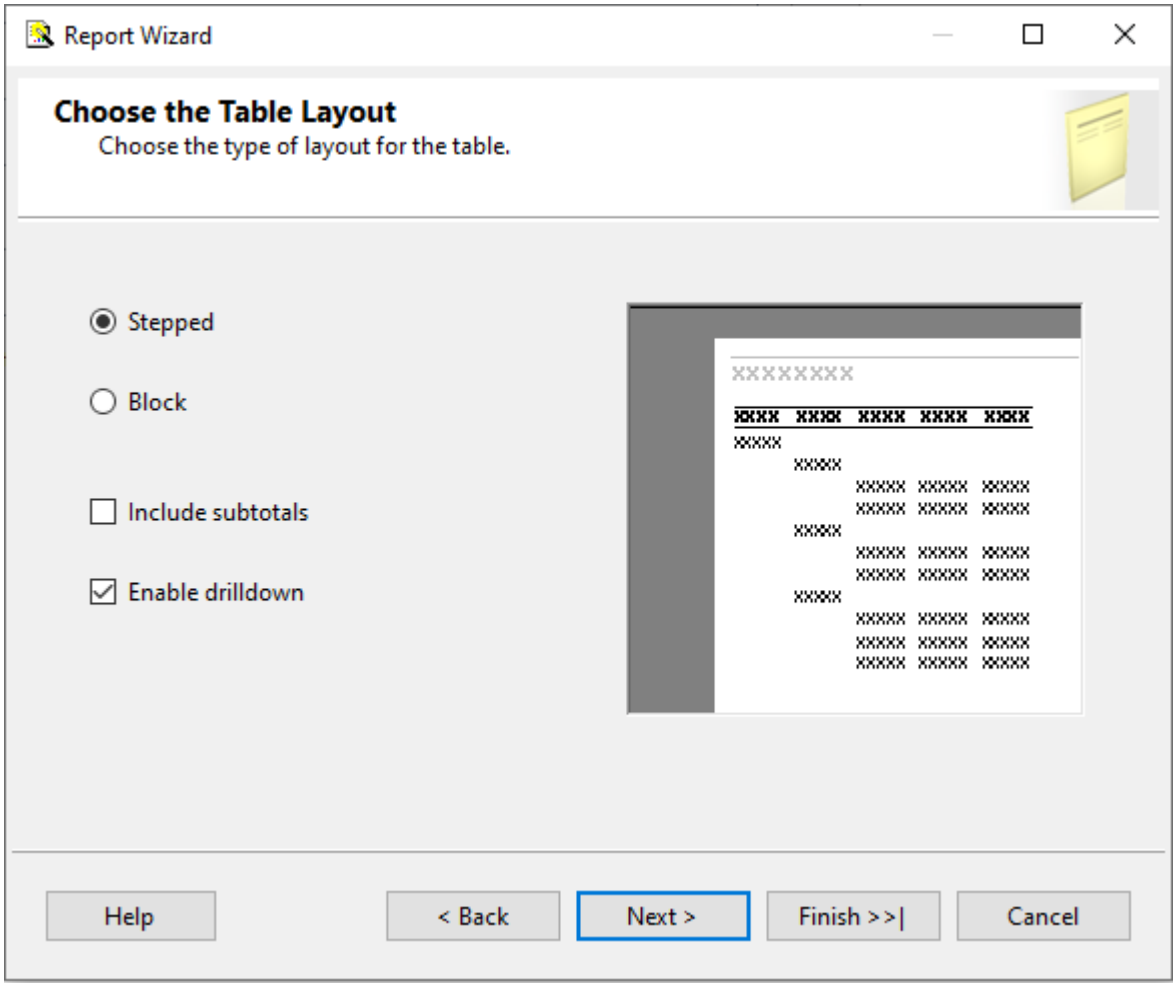


Fig Choosing the table output

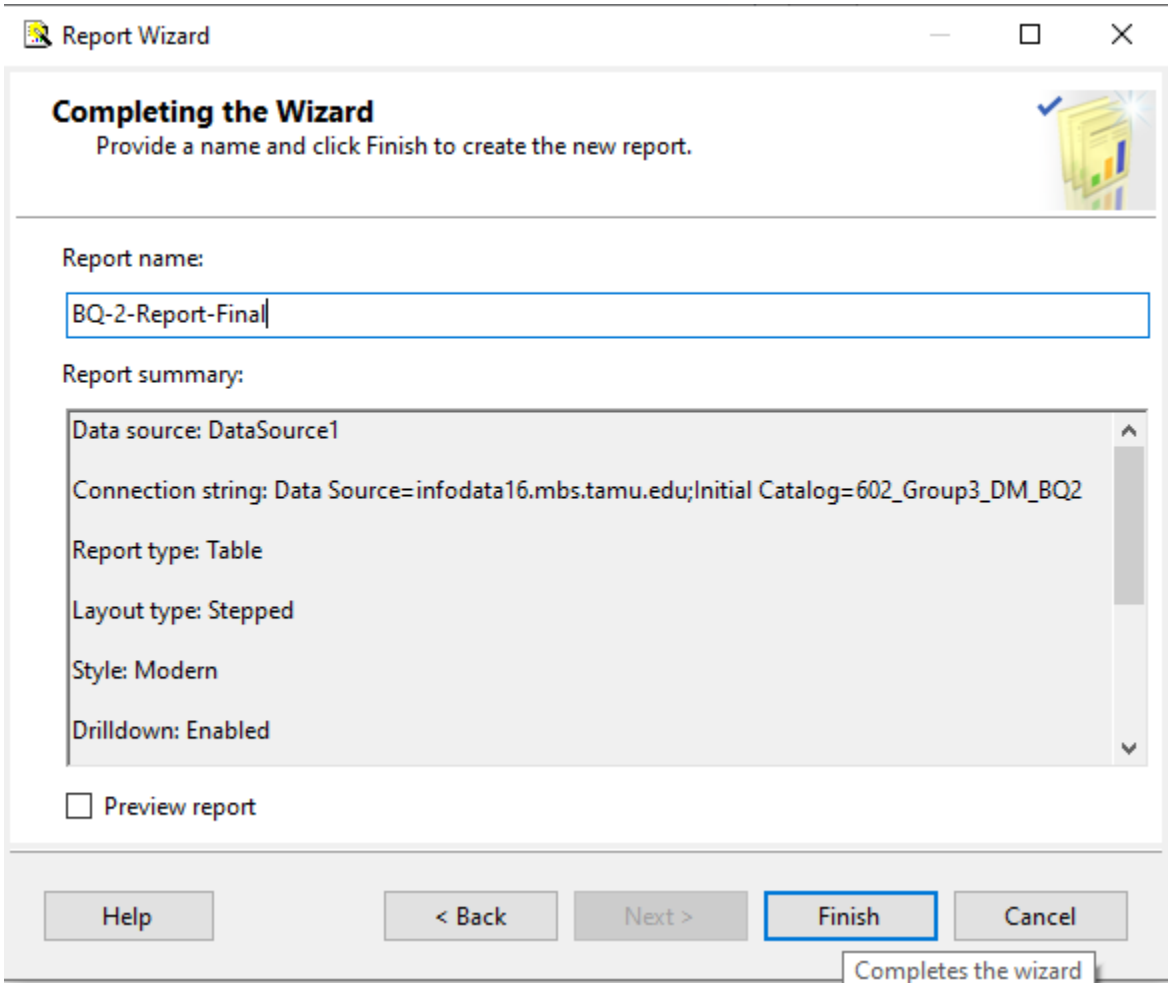


Fig Completing the setup in the wizard

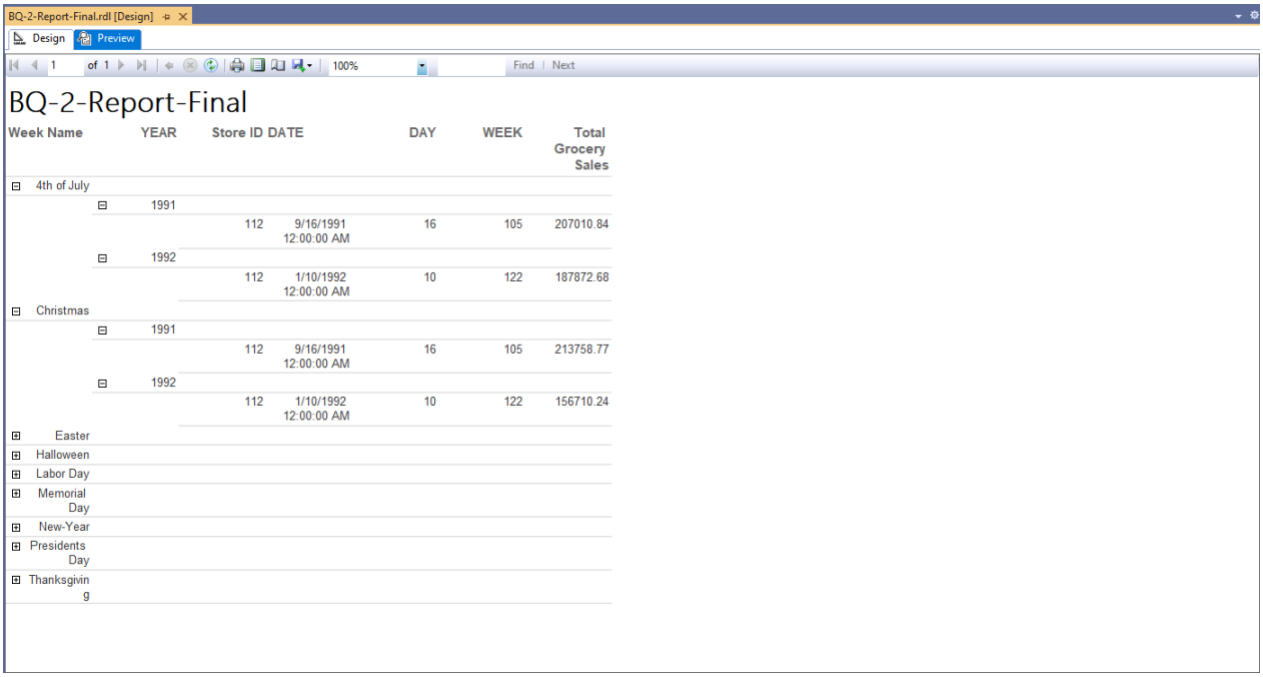


Fig Report preview successful

Deploying the report to server, “Properties” has target server

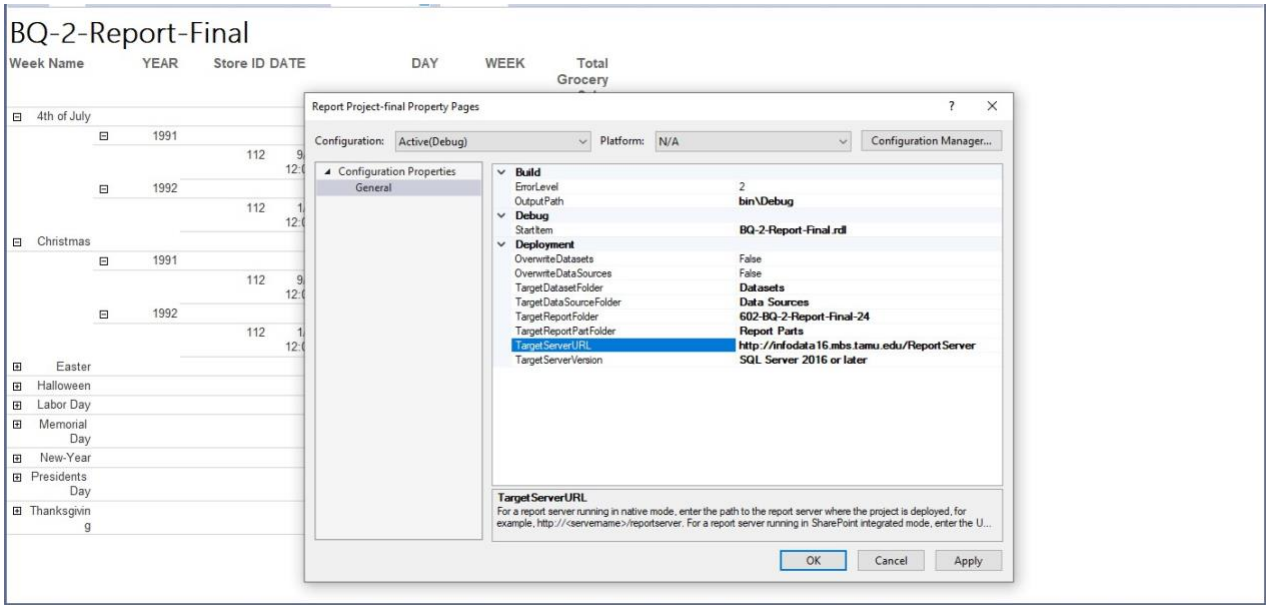


Fig Deploying the report to server

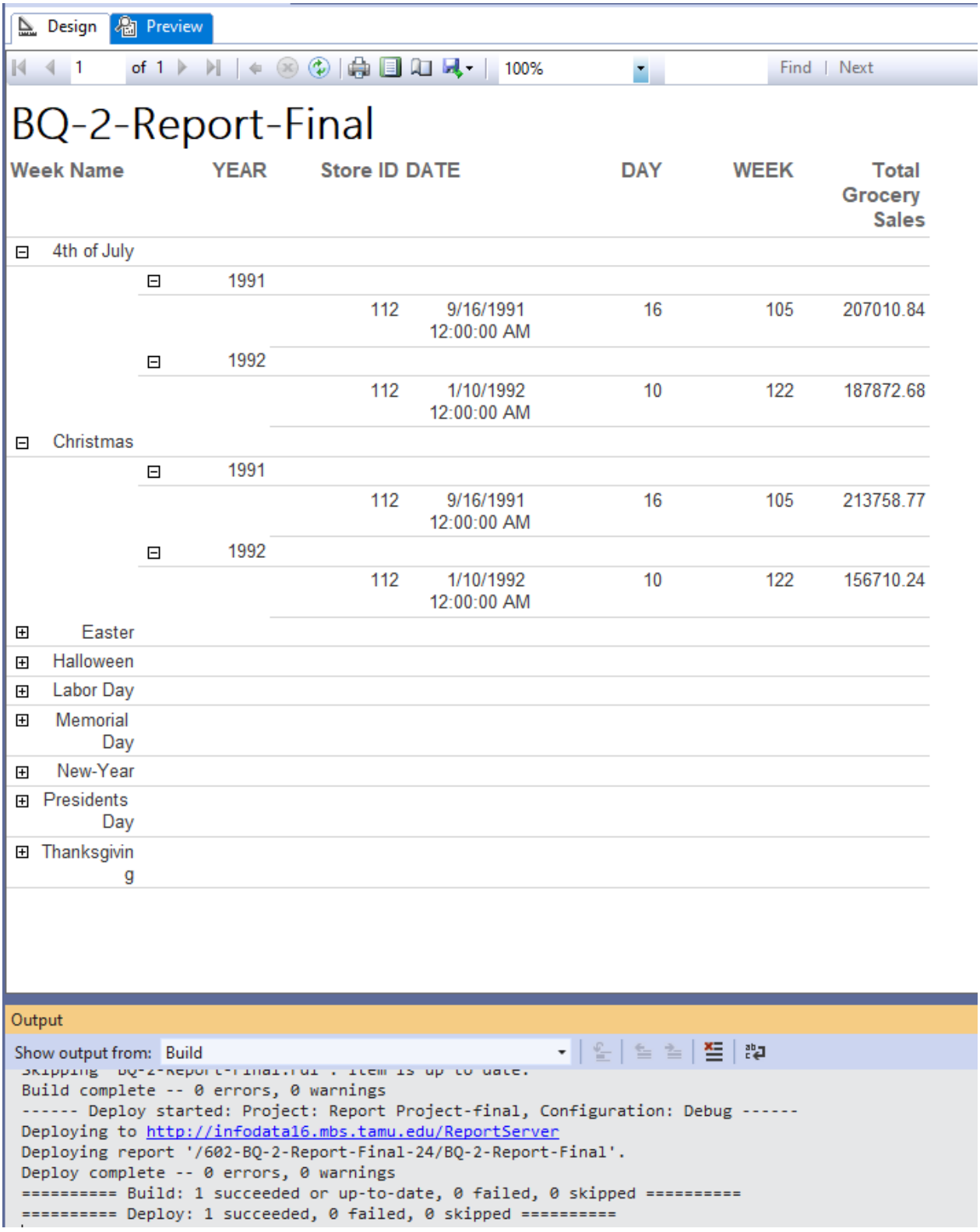


Fig Successfully deployed to the server

infodata16.mbs.tamu.edu/ReportServer - /

Tuesday, November 28, 2023 11:59 PM	<dir>	601_Group8_BQ1
Monday, November 27, 2023 12:08 AM	<dir>	601_Group8_BQ4 Bundle Sales
Tuesday, November 28, 2023 6:09 PM	<dir>	601_group8_Report1
Friday, November 22, 2024 4:39 PM	<dir>	601_grp2_q4.8_draft2
Friday, November 22, 2024 8:08 PM	<dir>	601_grp2_question9
Thursday, November 30, 2023 1:18 AM	<dir>	601-Group1ReportProjectAnalysisBQ3
Thursday, November 30, 2023 12:40 AM	<dir>	602_GR5_BQ_Meat
Tuesday, November 28, 2023 11:10 AM	<dir>	602_Group4_Project_Report-Q1
Thursday, November 30, 2023 2:26 AM	<dir>	602_Group4_Report-Q1-SSRS
Wednesday, November 29, 2023 11:46 PM	<dir>	602_Group4_Report-Q3
Thursday, November 30, 2023 1:03 AM	<dir>	602_Group4_Report-Q4
Monday, November 27, 2023 4:47 PM	<dir>	602_Group8_Reports
Friday, November 29, 2024 6:02 PM	<dir>	602-BQ-2-Report-Final-24
Tuesday, November 28, 2023 2:55 PM	<dir>	602-GR5-report

Fig Report deployed

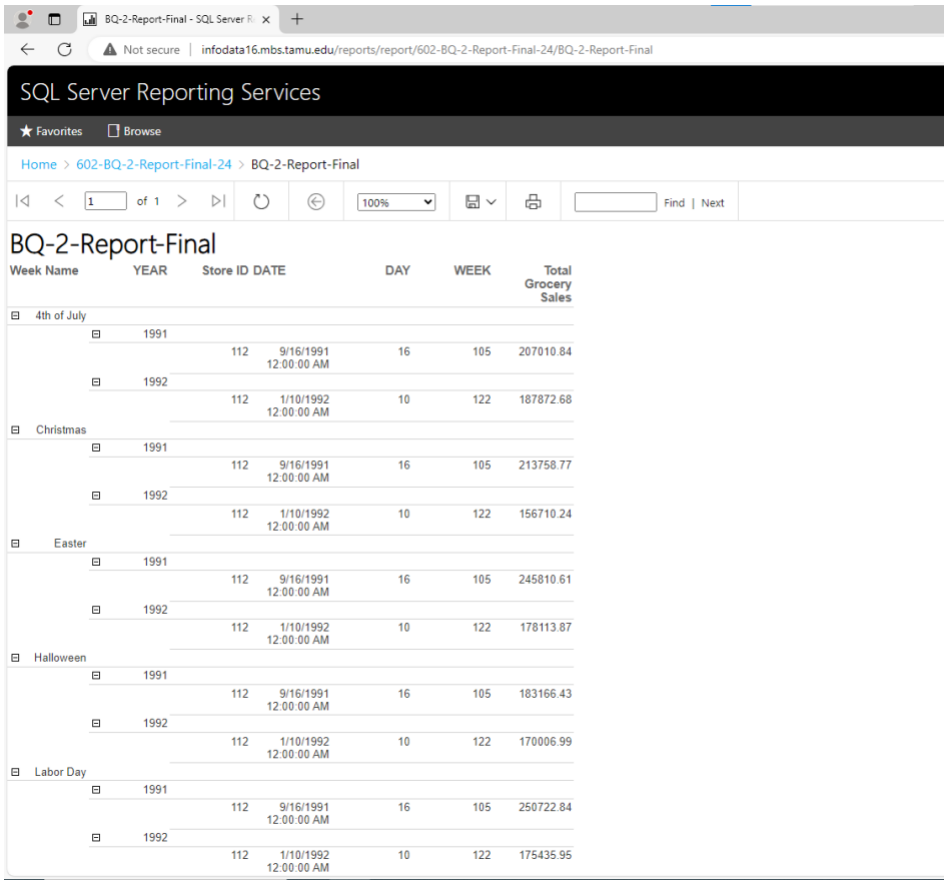


Fig Published report validated

3. How does the profit margin of toothpaste vary by brand?

Support: Profit margins are a crucial indicator of profitability and financial health in any organization. By understanding which brands offer the higher profit margin then it helps DFF make informed decisions about pricing strategy, promotions, and product optimization. It will provide insights into potential cost-saving opportunities, and where to focus marketing efforts. Additionally, it provides crucial insights for financial planning, ensuring resources are allocated efficiently and effectively to maximize overall profitability.

BI Visualization tool used - Microsoft Power BI

Import the data

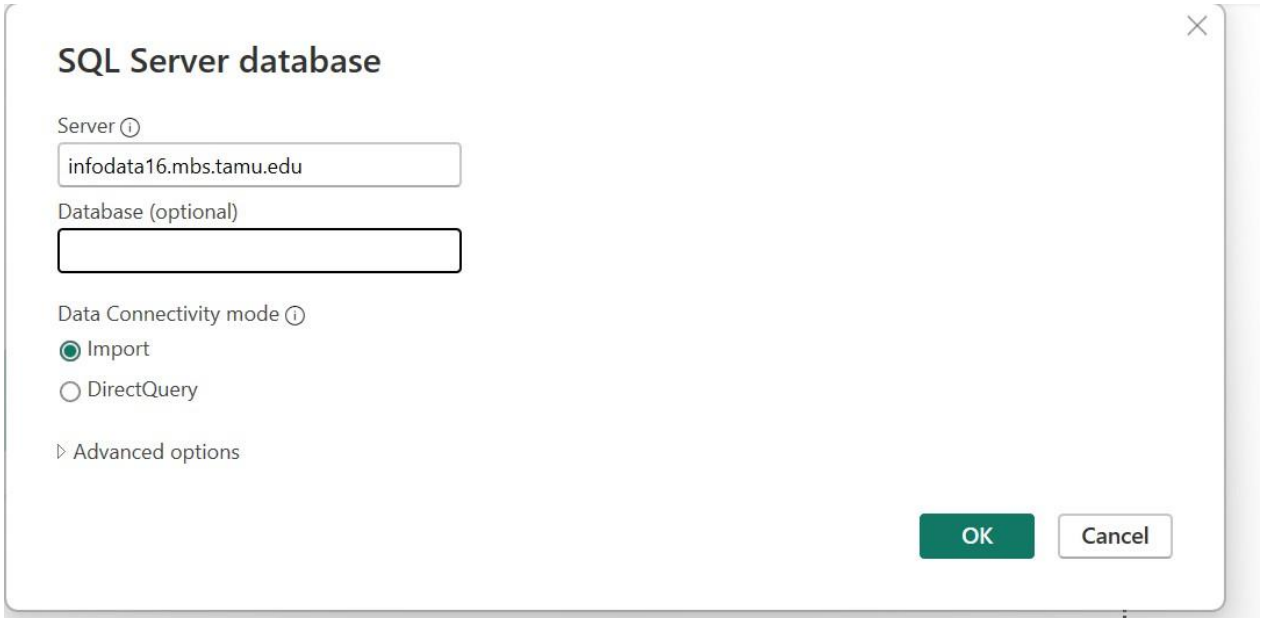


Fig: Connect to the server

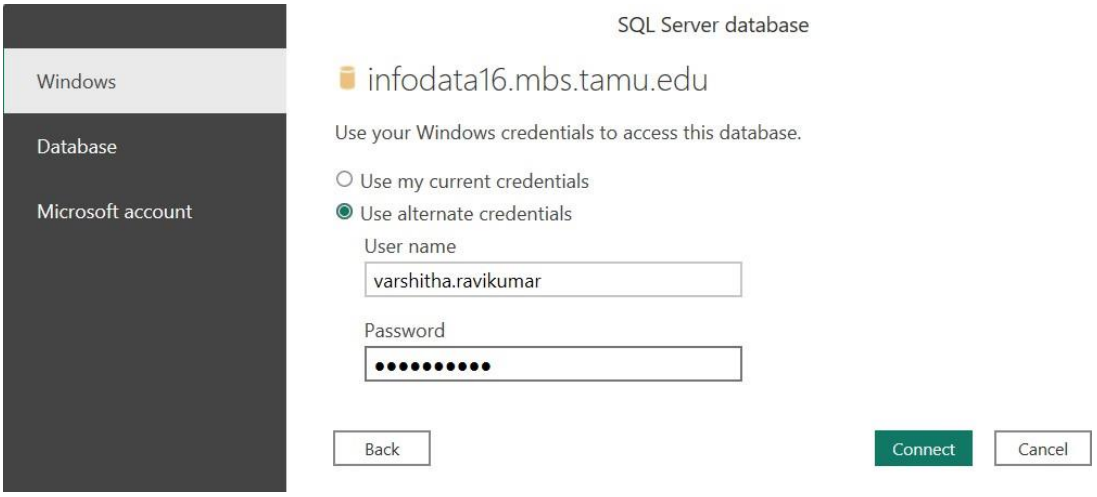


Fig: Authenticate

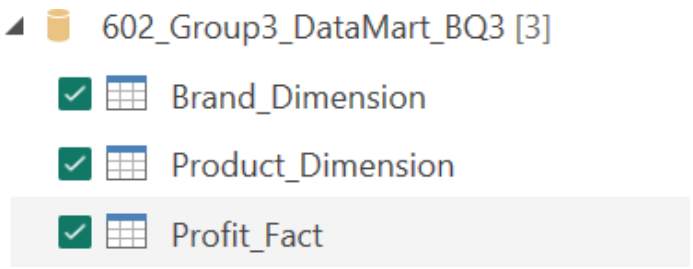


Fig: Select the required tables

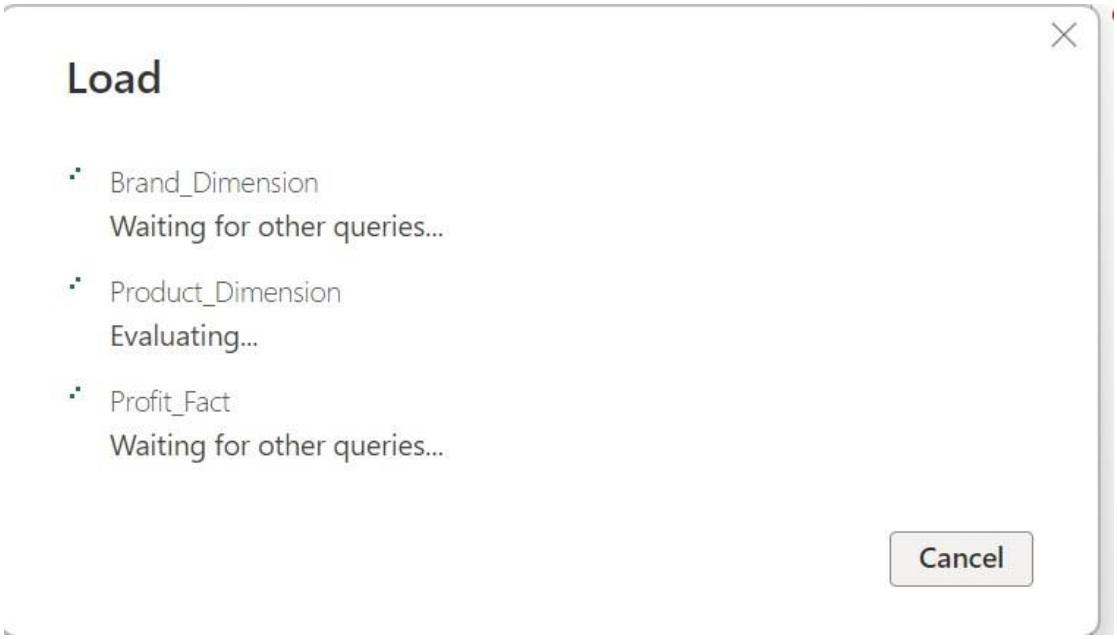


Fig: Data loading from the database to Power BI

Create new measure

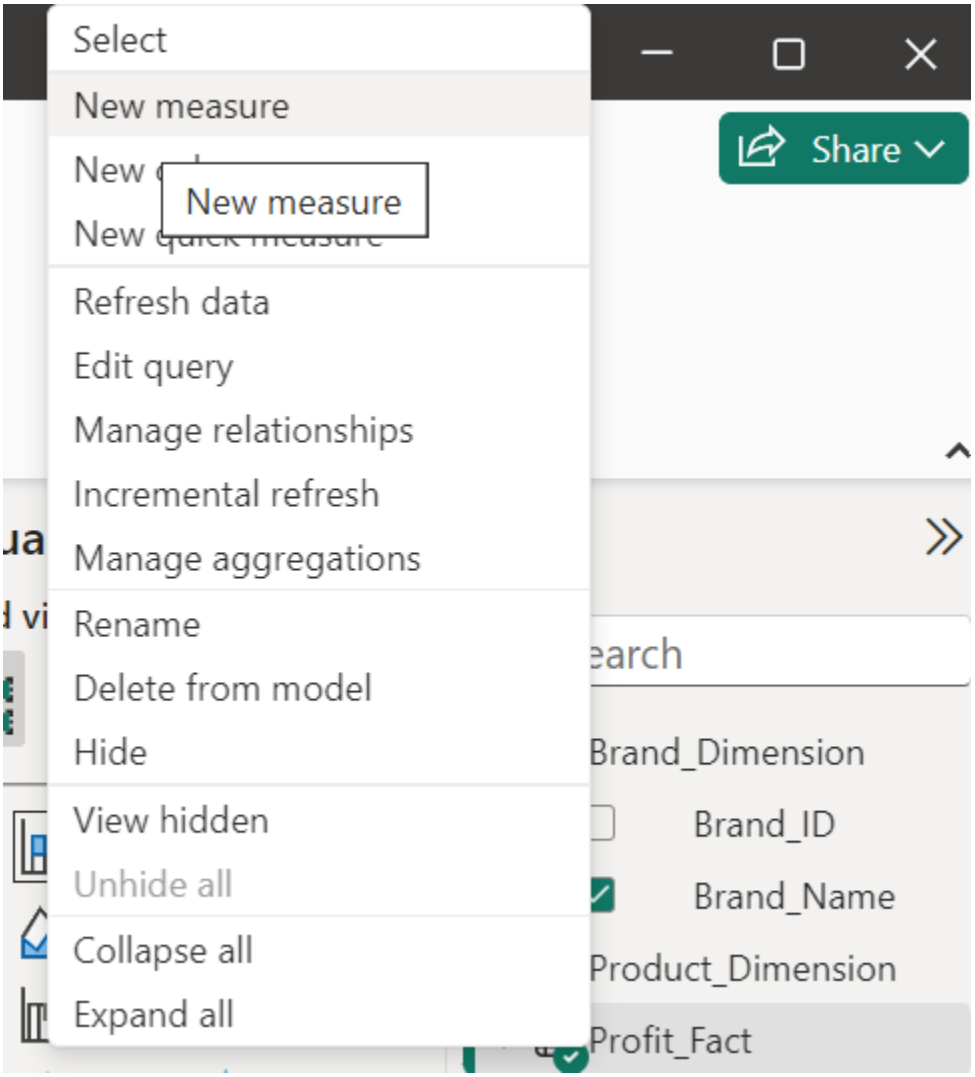


Fig: Create a new measure in the Profit_Fact data model

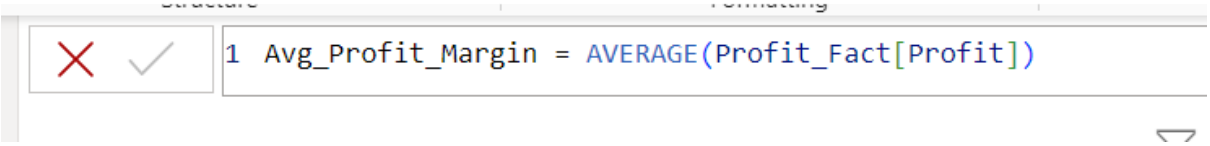


Fig: Assign the DAX formula to calculate the Average of the Profit for each brand

Visualize

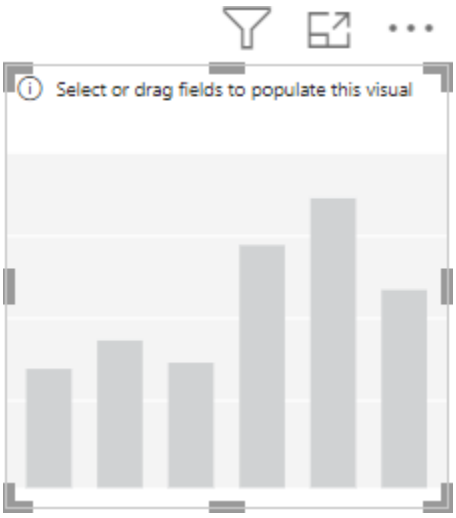


Fig: Select Bar graph for visualization

Select the Fields and Filters to Visualize the data:
Filters: select Profit_Fact data model
X-axis = Brand_Name
Y-axis = Avg_Profit_Margin

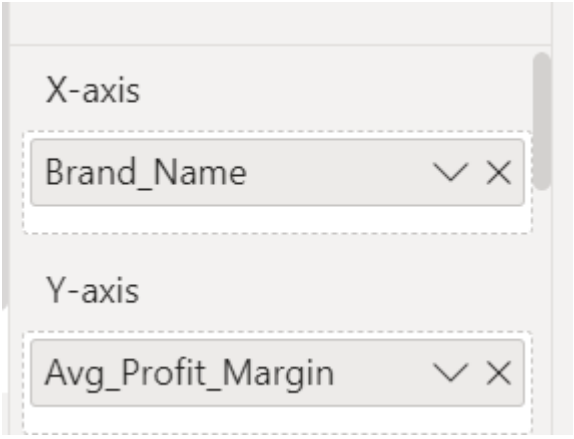


Fig: Select the filter to apply

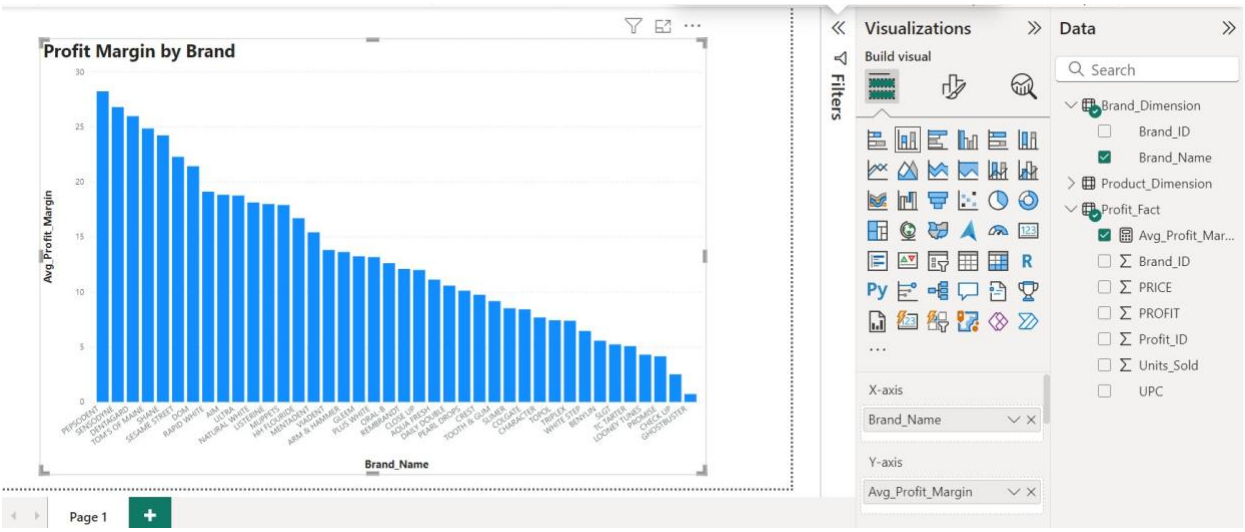


Fig: Filters and fields selected for the visualization

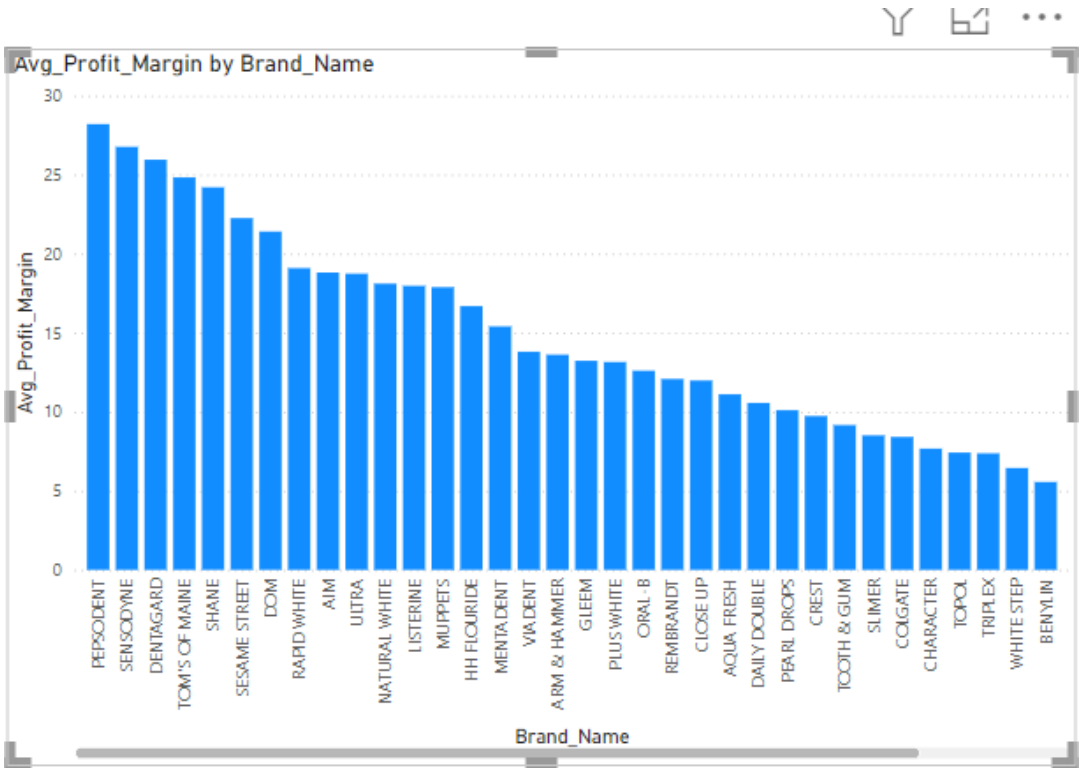


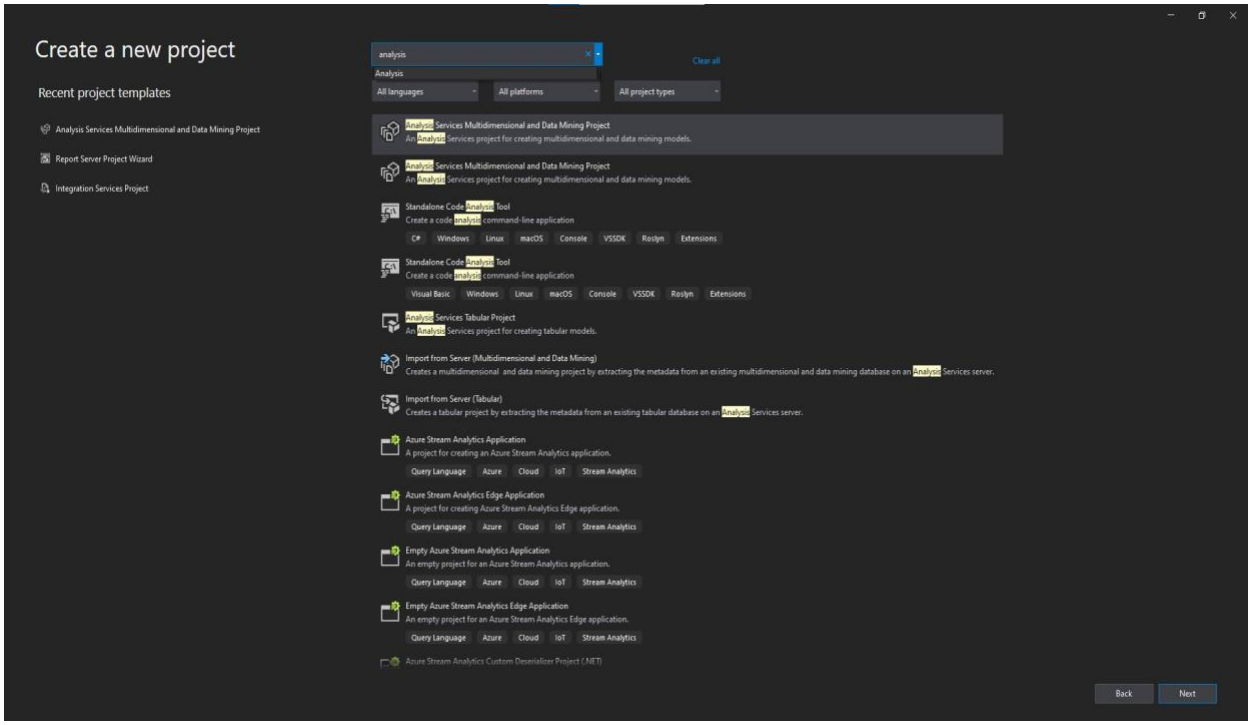
Fig: Profit Margin trend across different toothpaste brands

4. **What are the monthly average sales amounts for the BAKERY, DAIRY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores located in Naperville and Schaumburg during the year 1994, and how do these averages compare among the different departments within these locations?**

Support: Analyzing the monthly average sales amounts for the BAKERY, DIARY, PHARMACY, COSMETIC, and HABA departments across Dominick's Fine Food stores in Naperville and Schaumburg during 1994 is beneficial for DFF as it provides insights into consumer preferences and sales trends. Understanding which departments perform better can guide inventory management, marketing strategies, and resource allocation. This data-driven approach enables DFF to optimize promotions and improve customer satisfaction, ultimately enhancing profitability.

BI Visualization Tools Used: SSAS and Tableau

Step 1: Create a new Analysis Service Multidimensional and data mining project



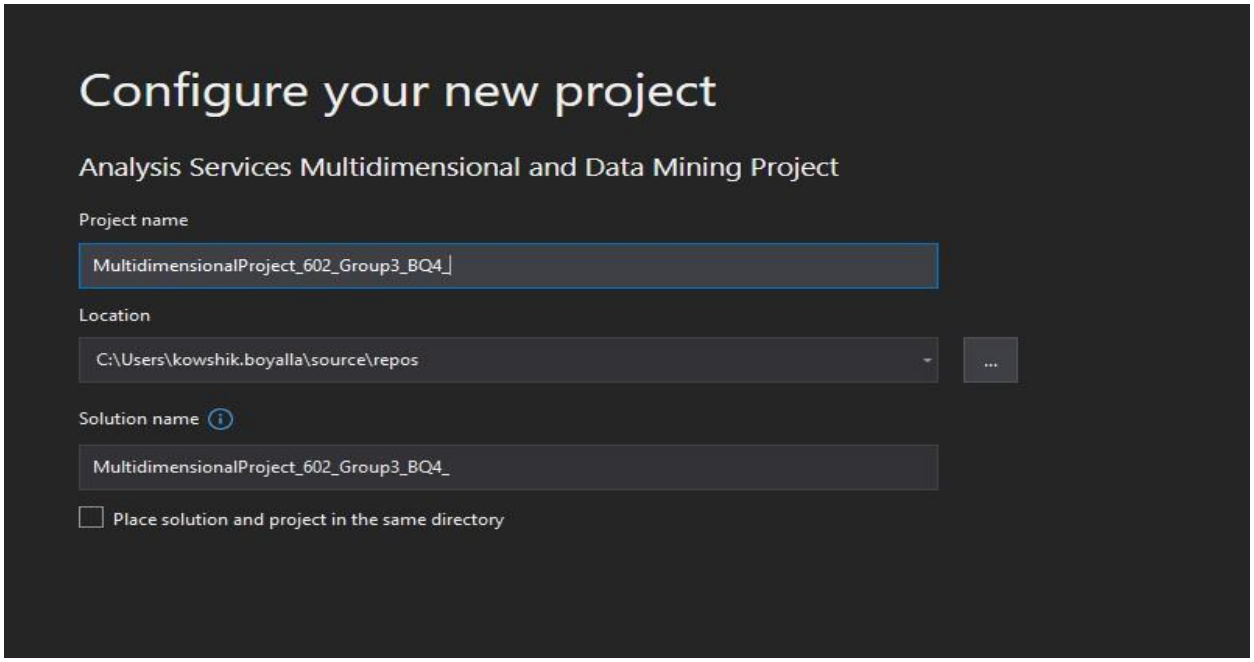


Fig: Creating a multidimensional Analysis project

Step 2: Create a new data source and select option to create a database based on existing connection

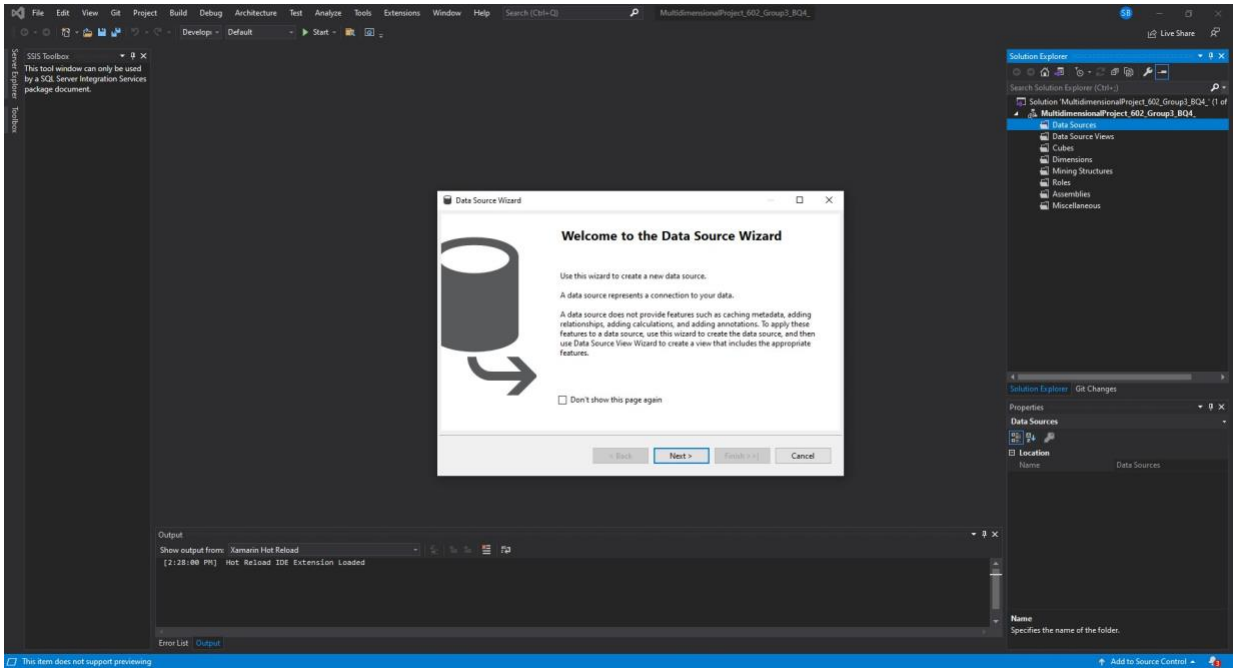


Fig: Creating a new data source

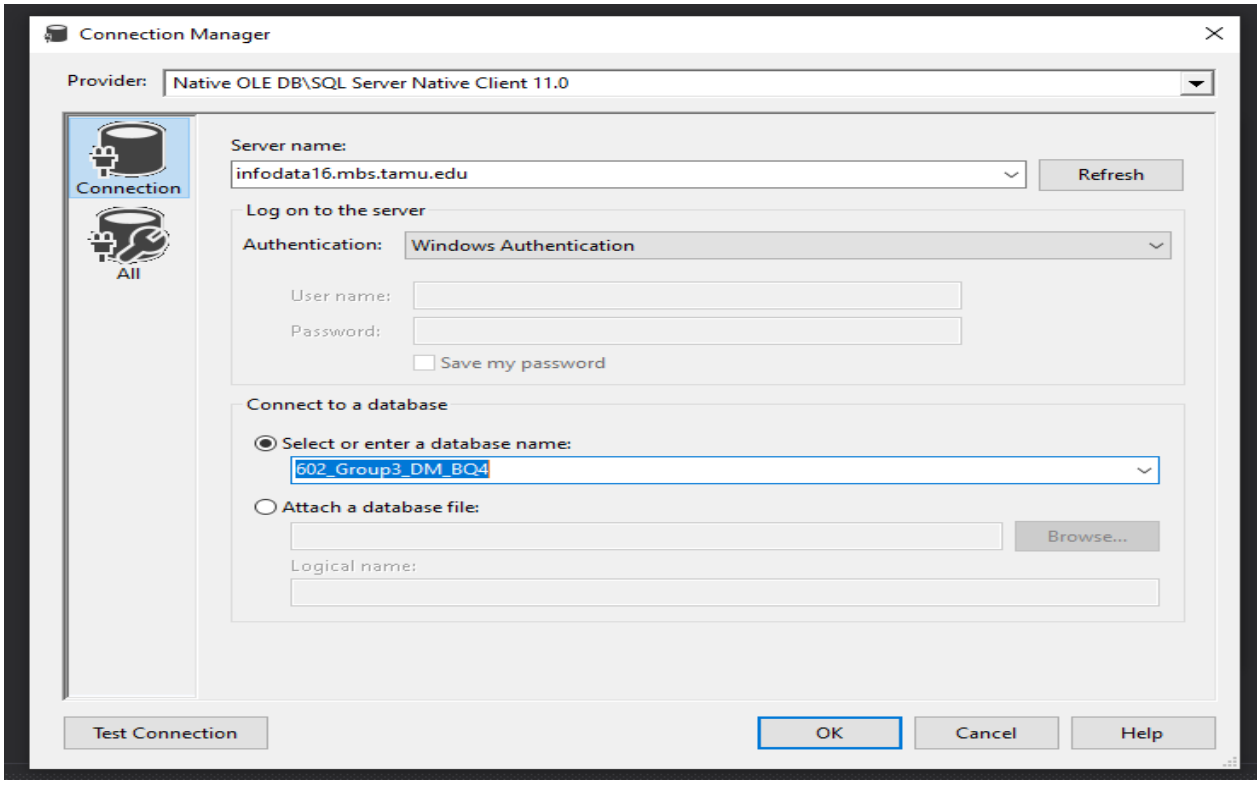


Fig: Making a connection

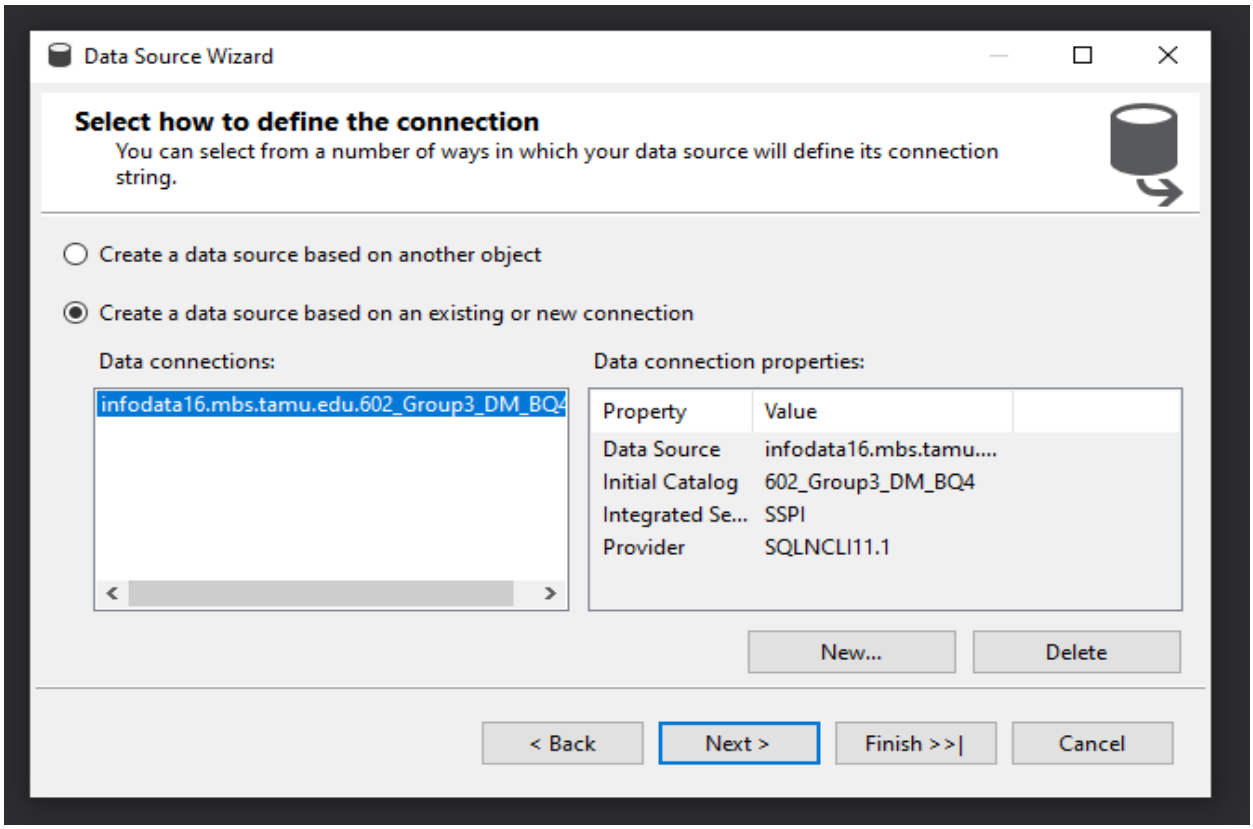


Fig: Connecting to data mart in SQL Server

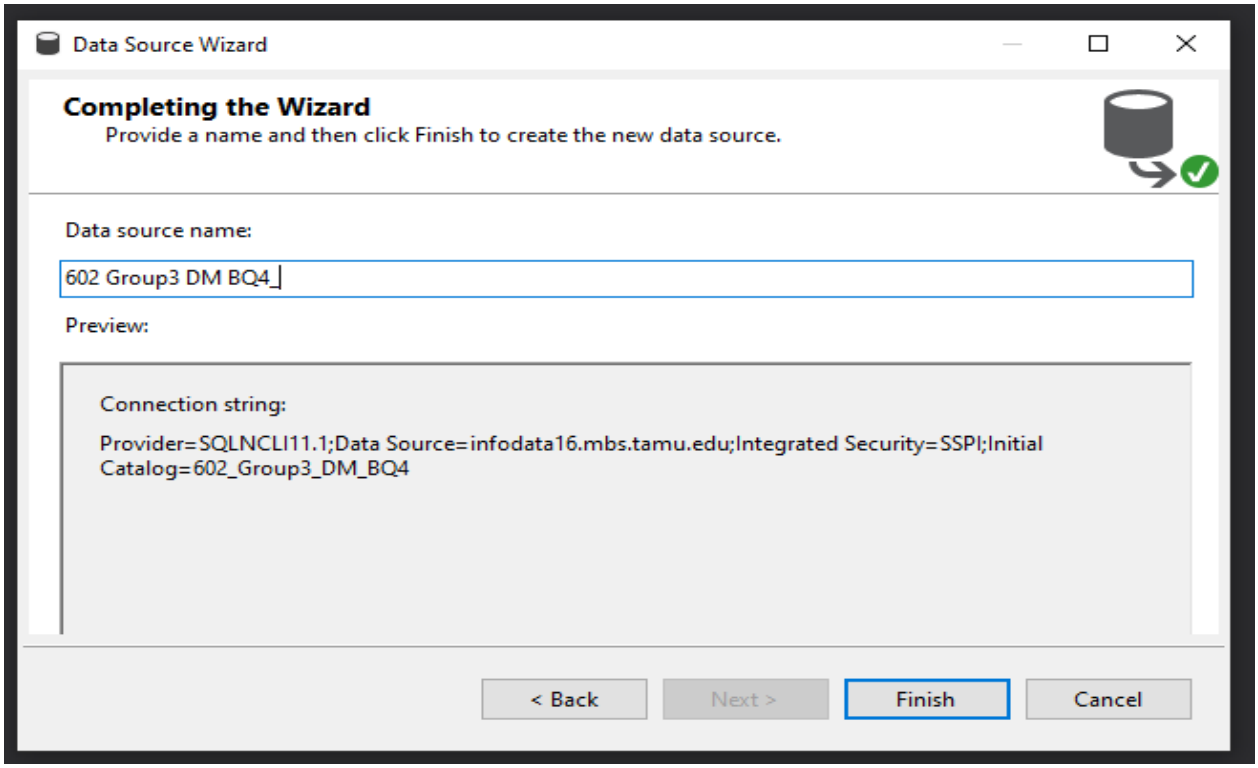


Fig: Setting up the Data Source name

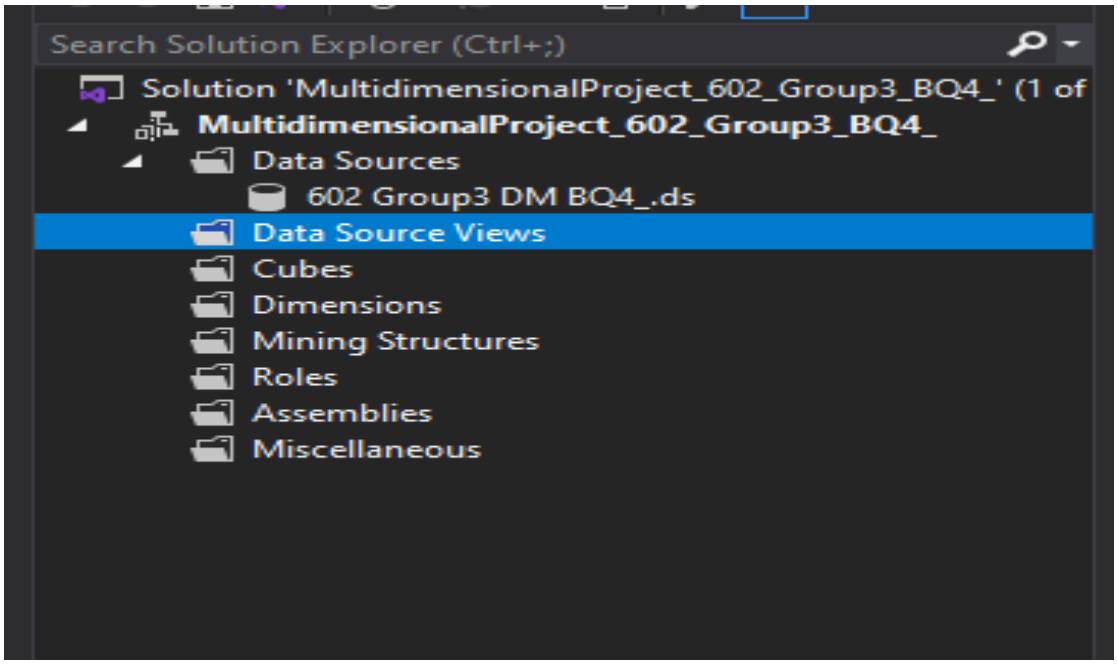


Fig: Successful creation of data source

Step 3: Creating a new data source view

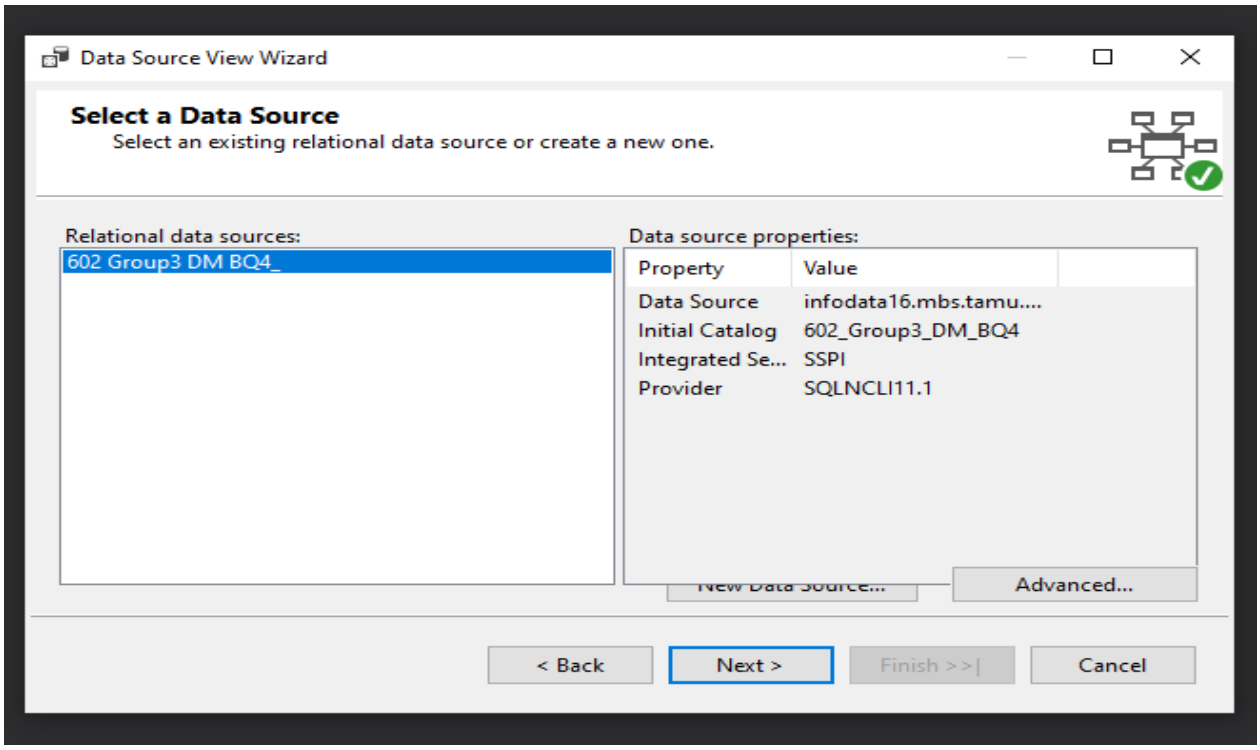


Fig: Selecting a data source

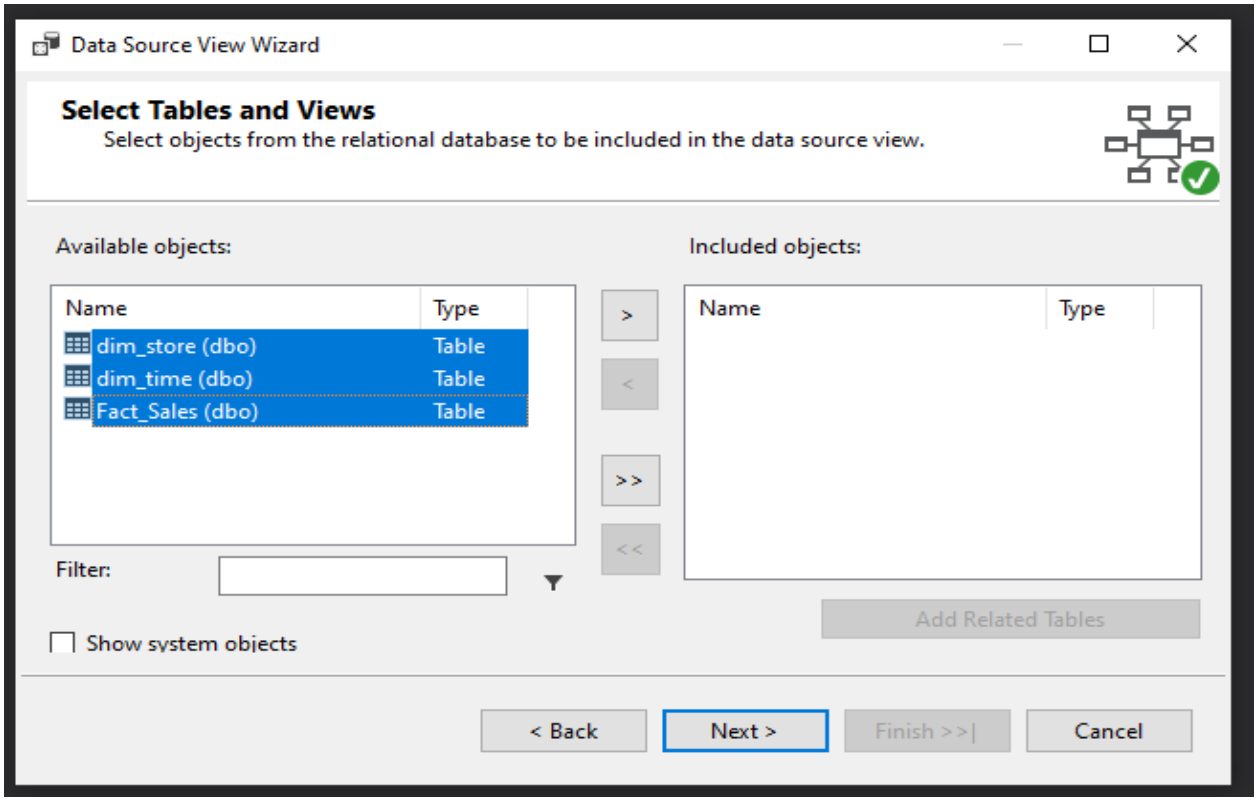


Fig: Selecting Fact and dimension tables

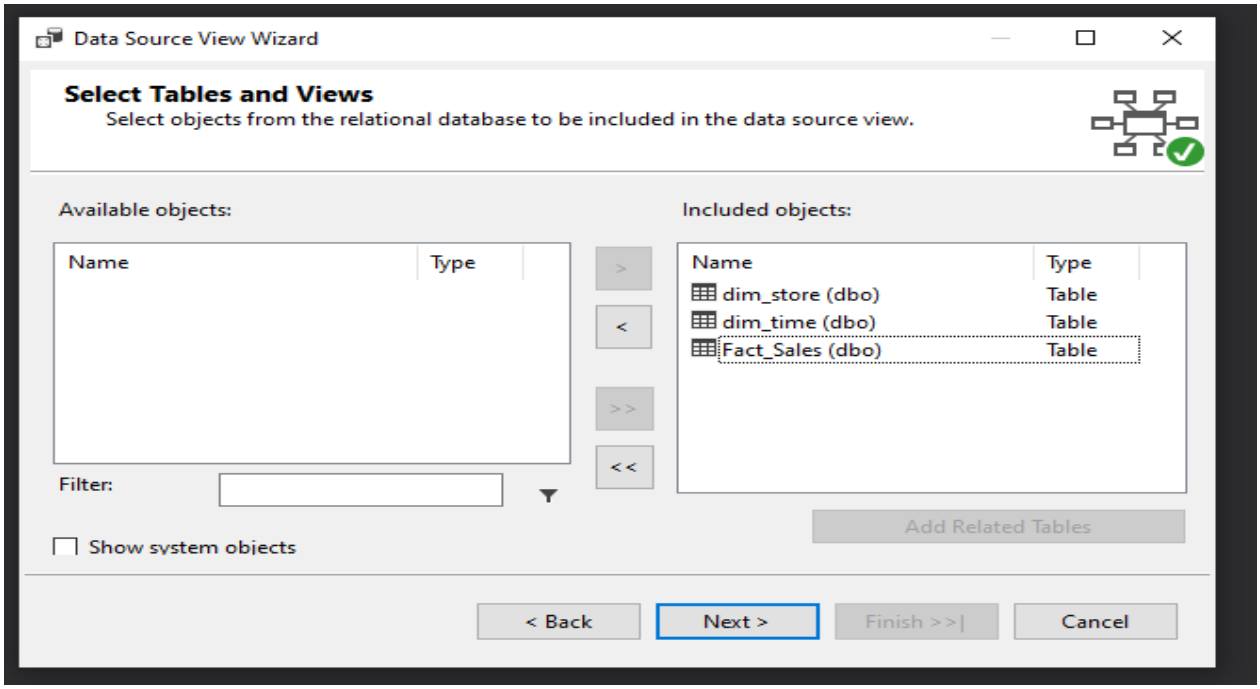


Fig: Selected Fact and dimension tables

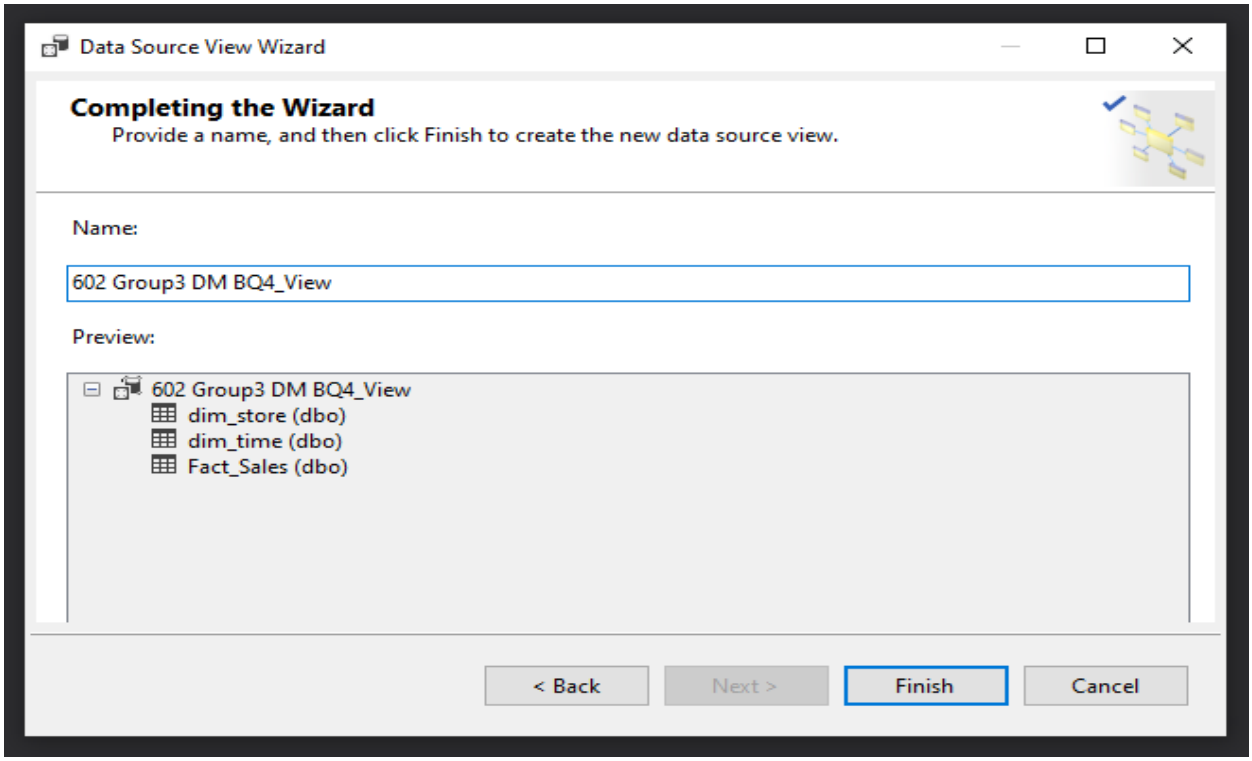


Fig: creating data source view

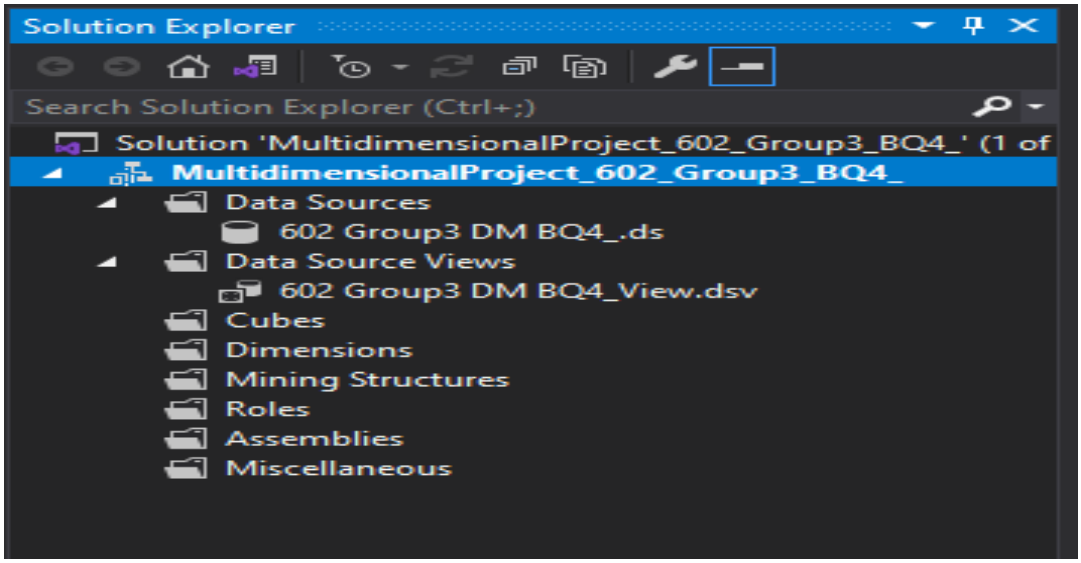


Fig: Successful creation of data source view

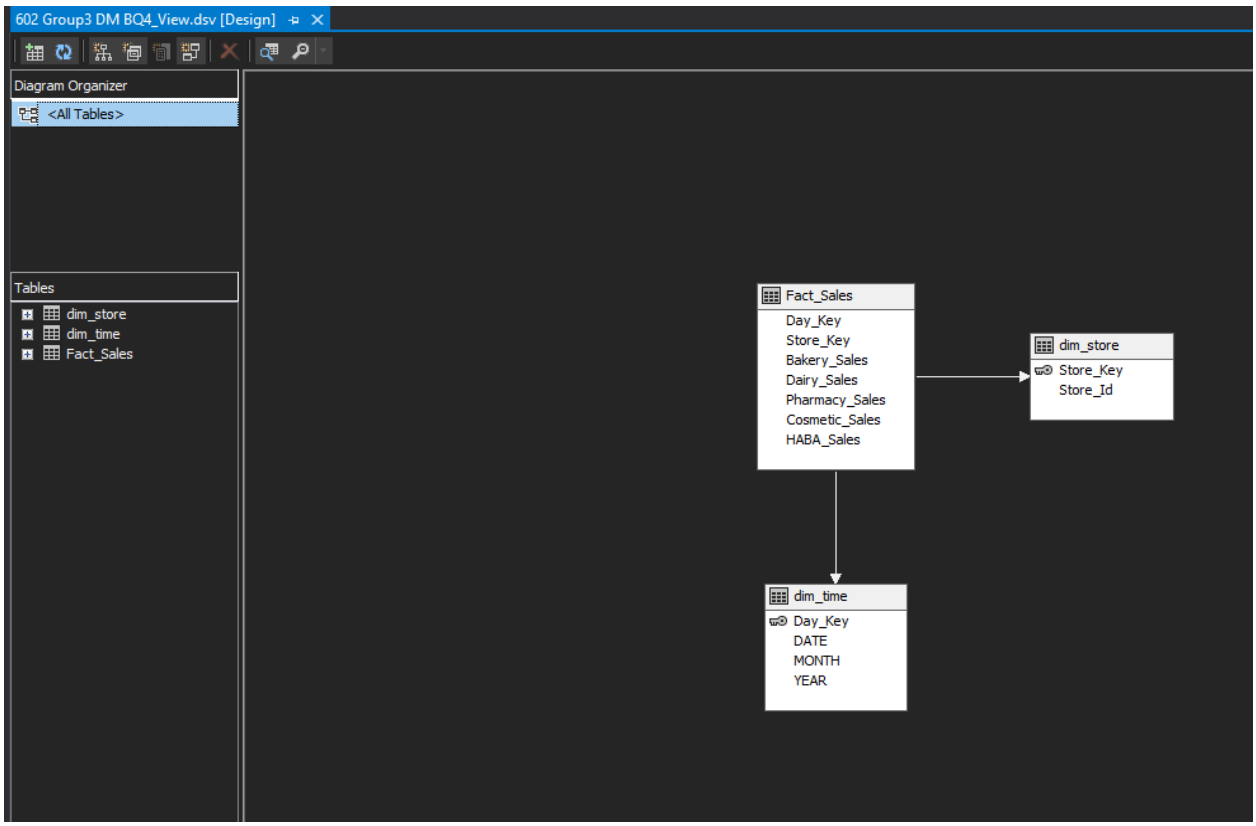


Fig: Data Source View

Step 4: Creating a named query to create a table which contains monthly average sales of departments.

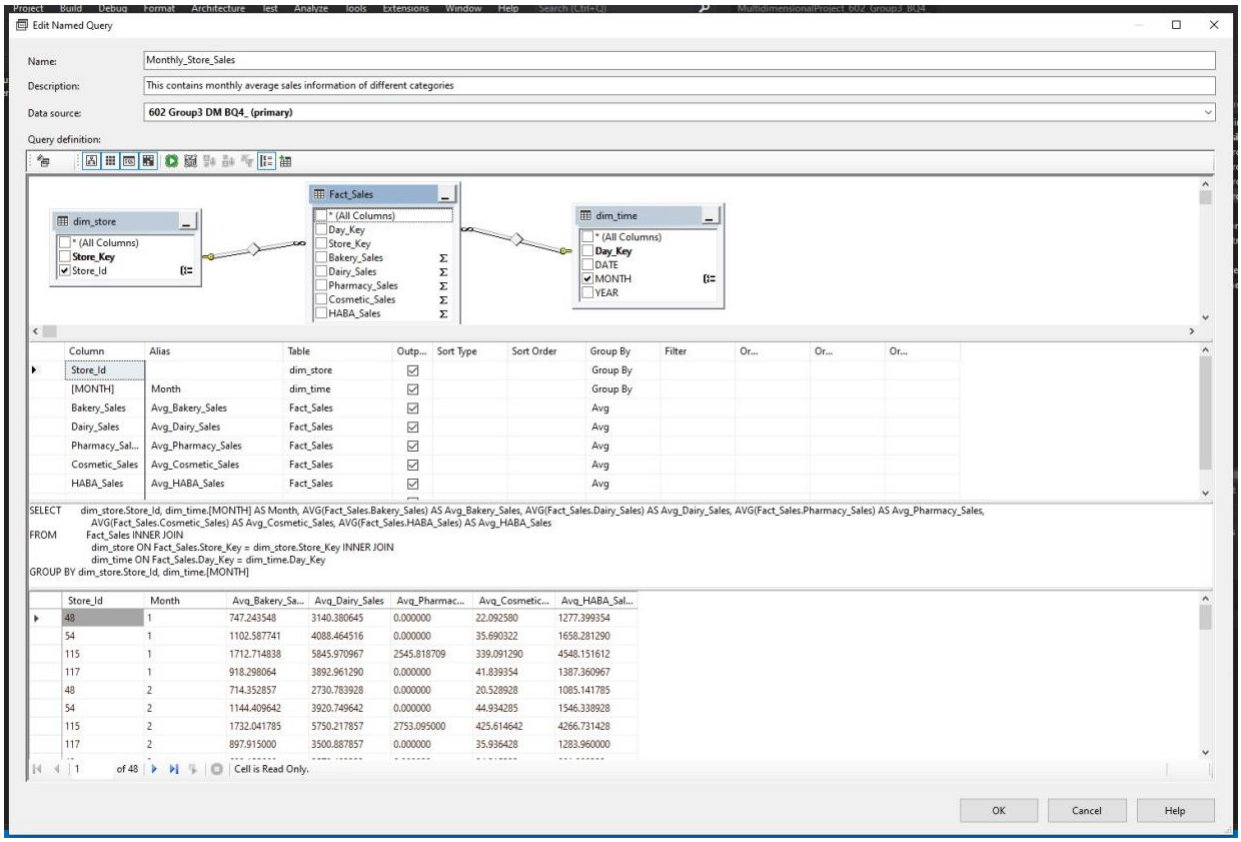


Fig: Named Query to create a table with monthly average sales amounts

Query:

```
SELECT        dim_store.Store_Id, dim_time.[MONTH] AS Month,
AVG(Fact_Sales.Bakery_Sales) AS Avg_Bakery_Sales, AVG(Fact_Sales.Dairy_Sales) AS
Avg_Dairy_Sales, AVG(Fact_Sales.Pharmacy_Sales) AS Avg_Pharmacy_Sales,
                    AVG(Fact_Sales.Cosmetic_Sales) AS Avg_Cosmetic_Sales,
AVG(Fact_Sales.HABA_Sales) AS Avg_HABA_Sales
FROM            Fact_Sales INNER JOIN
                    dim_store ON Fact_Sales.Store_Key = dim_store.Store_Key INNER JOIN
                    dim_time ON Fact_Sales.Day_Key = dim_time.Day_Key
GROUP BY dim_store.Store_Id, dim_time.[MONTH]
```

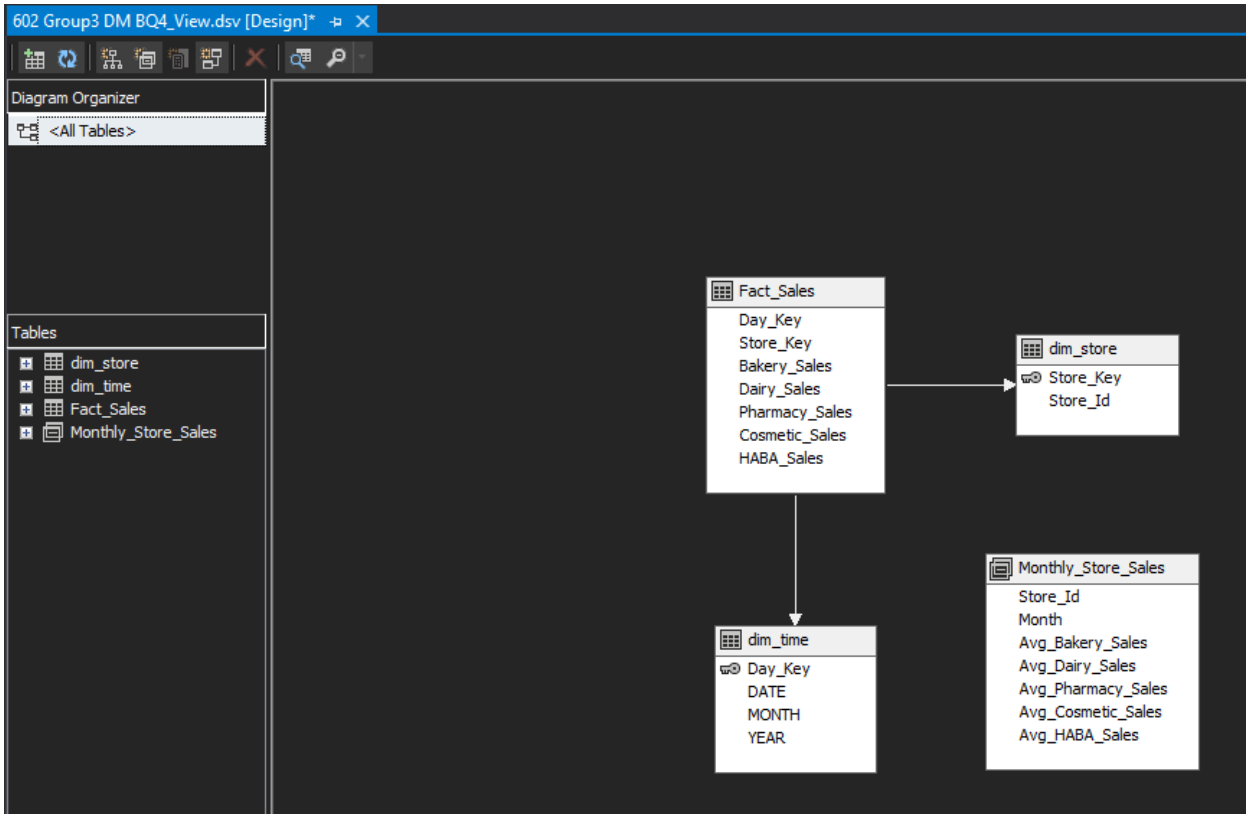


Fig: Successful creation of Monthly Store Sales table

Step 5: Setting up the relationships for the Monthly_Store_Sales table

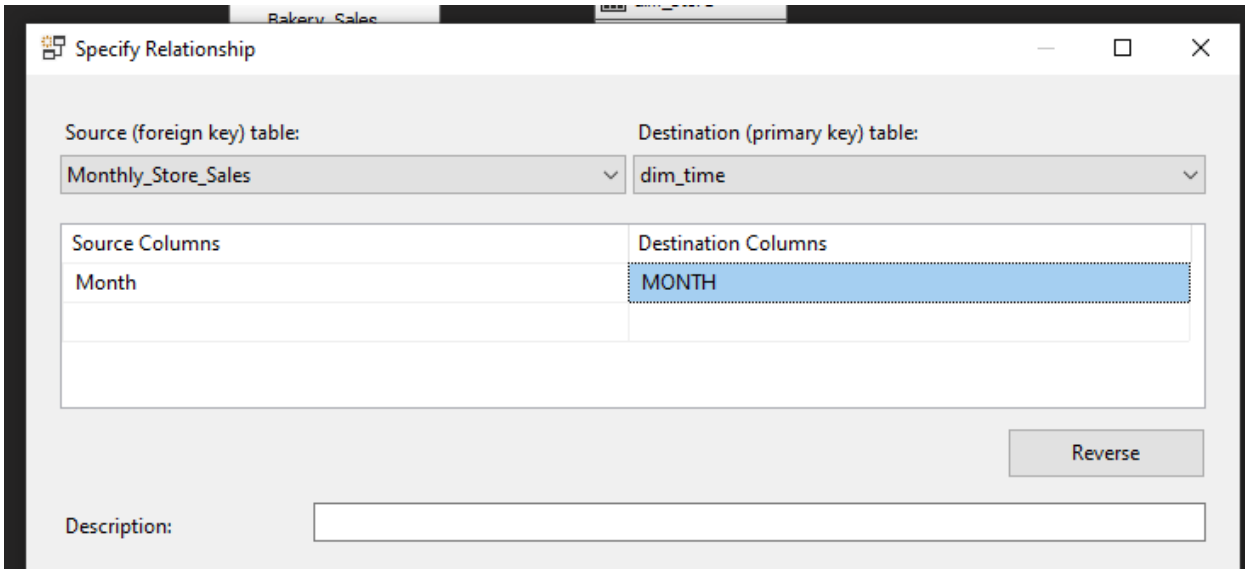


Fig: Creating a relationship with time dimension

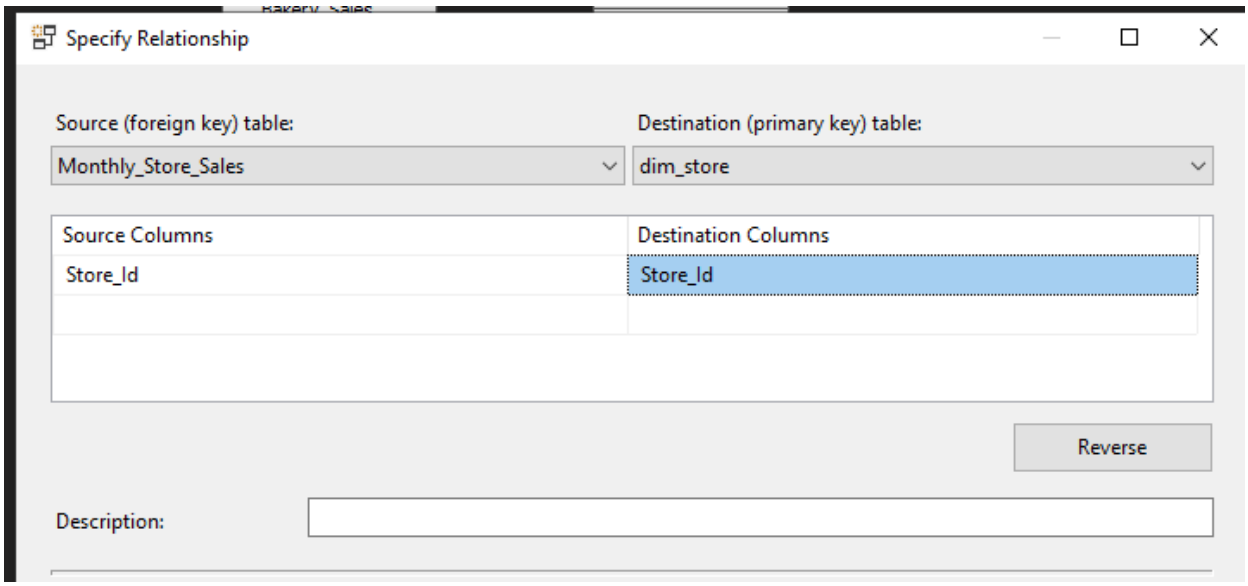


Fig: Creating a relationship with store dimension

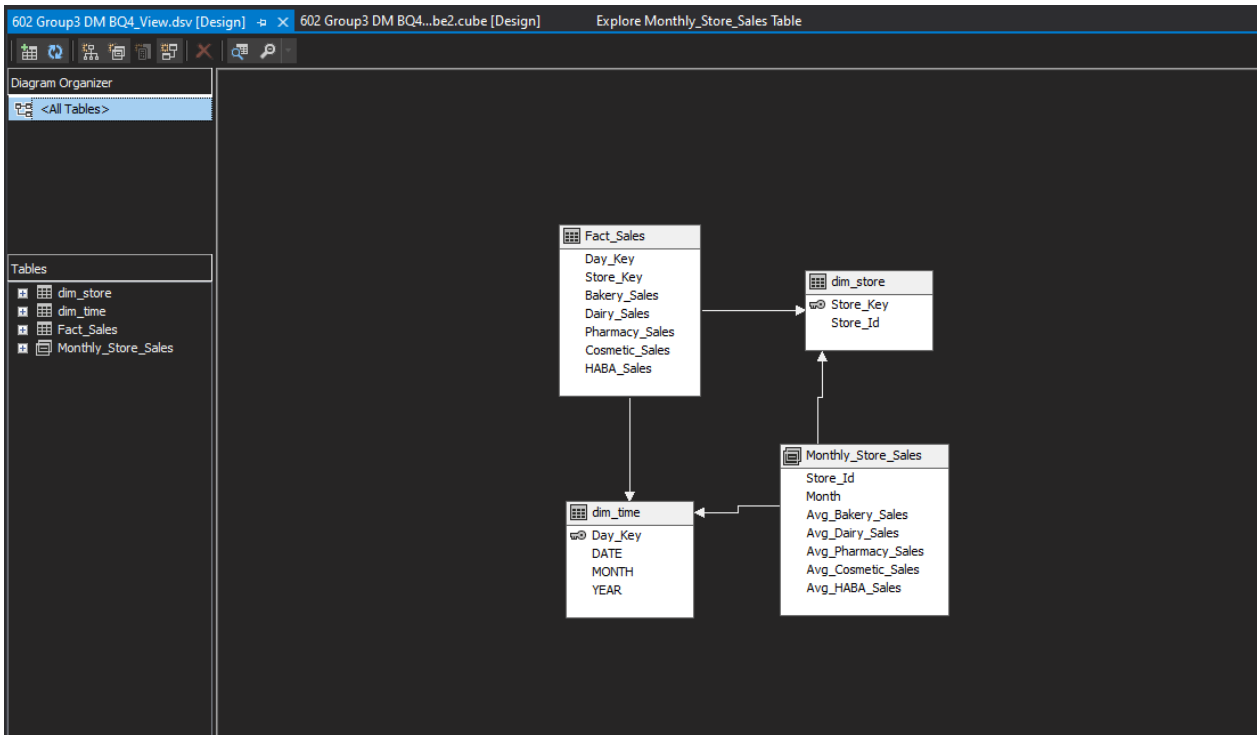
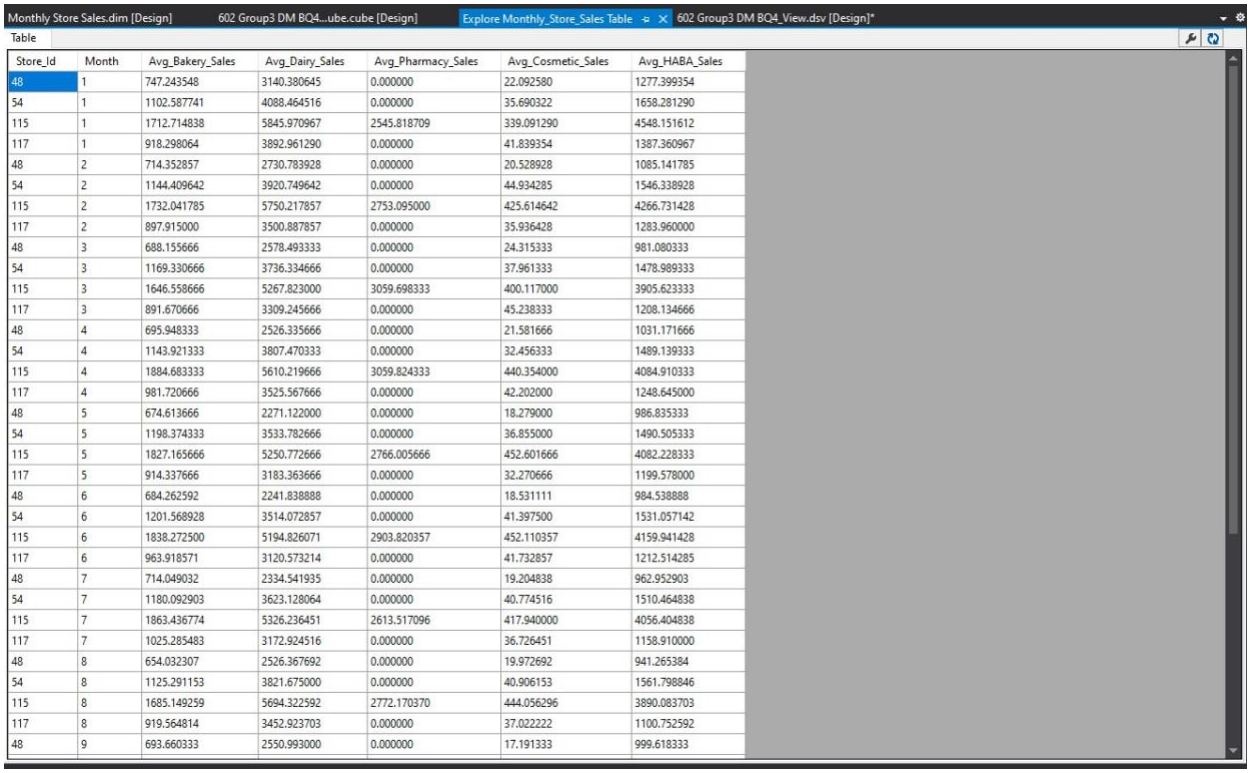


Fig: Data Source View after adding relationships



Monthly Store Sales.dim [Design] | 602 Group3 DM BQ4...ube.cube [Design] | Explore Monthly_Store_Sales Table | 602 Group3 DM BQ4_View.dsv [Design]*

Table

Store_Id	Month	Avg_Bakery_Sales	Avg_Dairy_Sales	Avg_Pharmacy_Sales	Avg_Cosmetic_Sales	Avg_HABA_Sales
48	1	747.243548	3140.380645	0.000000	22.092580	1277.399354
54	1	1102.587741	4088.464516	0.000000	35.690322	1658.281290
115	1	1712.714838	5845.970967	2545.818709	339.091290	4548.151612
117	1	918.298064	3892.961290	0.000000	41.839354	1387.360967
48	2	714.352857	2730.783928	0.000000	20.528928	1085.141785
54	2	1144.409642	3920.749642	0.000000	44.934285	1546.338928
115	2	1732.041785	5750.217857	2753.095000	425.614642	4266.731428
117	2	897.915000	3500.887857	0.000000	35.936428	1283.960000
48	3	688.155666	2578.493333	0.000000	24.315333	981.080333
54	3	1169.330666	3736.334666	0.000000	37.961333	1478.989333
115	3	1646.558666	5267.823000	3059.698333	400.117000	3905.623333
117	3	891.670666	3309.245666	0.000000	45.238333	1208.134666
48	4	695.948333	2526.335666	0.000000	21.581666	1031.171666
54	4	1143.921333	3807.470333	0.000000	32.456333	1489.139333
115	4	1884.683333	5610.219666	3059.824333	440.354000	4084.910333
117	4	981.720666	3525.567666	0.000000	42.202000	1248.645000
48	5	674.613666	2271.122000	0.000000	18.279000	986.835333
54	5	1198.374333	3533.782666	0.000000	36.855000	1490.505333
115	5	1827.165666	5250.772666	2766.005666	452.601666	4082.228333
117	5	914.337666	3183.363666	0.000000	32.270666	1199.578000
48	6	684.262592	2241.838888	0.000000	18.531111	984.538888
54	6	1201.568928	3514.072857	0.000000	41.397500	1531.057142
115	6	1838.272500	5194.826071	2903.820357	452.110357	4159.941428
117	6	963.918571	3120.573214	0.000000	41.732857	1212.514285
48	7	714.049032	2334.541935	0.000000	19.204838	962.952903
54	7	1180.092903	3623.128064	0.000000	40.774516	1510.464838
115	7	1863.436774	5326.236451	2613.517096	417.940000	4056.404838
117	7	1025.285483	3172.924516	0.000000	36.726451	1158.910000
48	8	654.032307	2526.367692	0.000000	19.972692	941.265384
54	8	1125.291153	3821.675000	0.000000	40.906153	1561.798846
115	8	1685.149259	5694.322592	2772.170370	444.056296	3890.083703
117	8	919.564814	3452.923703	0.000000	37.022222	1100.752592
48	9	693.660333	2550.993000	0.000000	17.191333	999.618333

Fig: Exploring the Monthly Store Sales table data

Step 6: Creating a cube

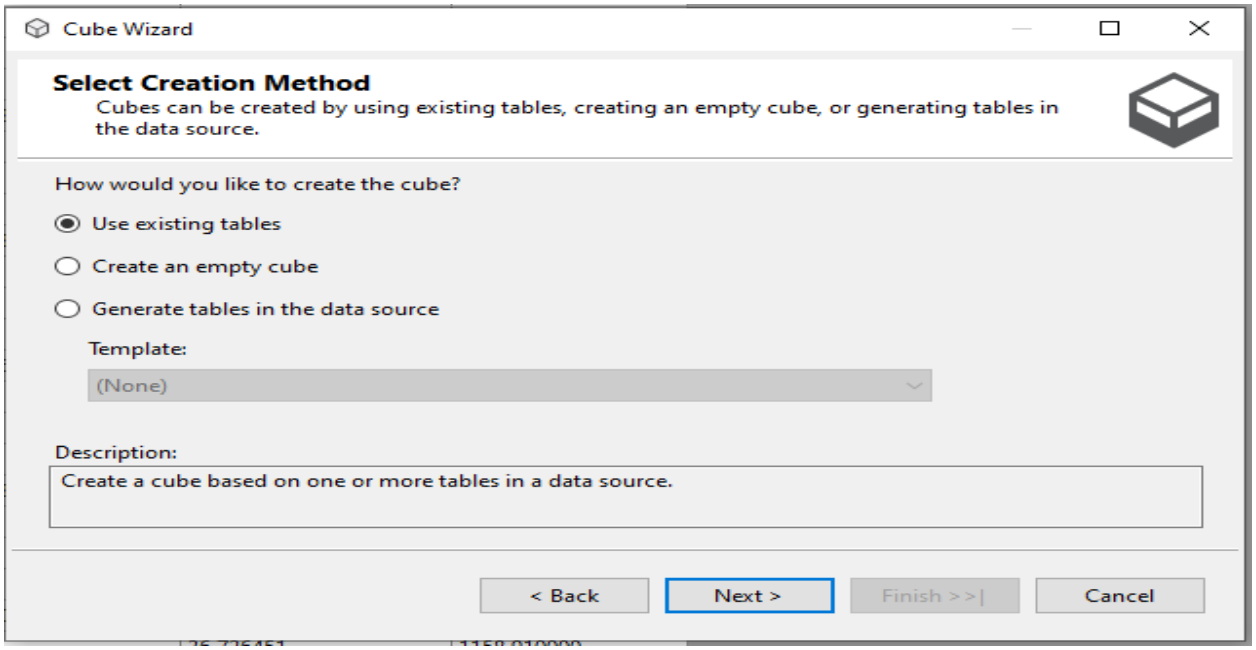


Fig: Creating a multidimensional hyper cube

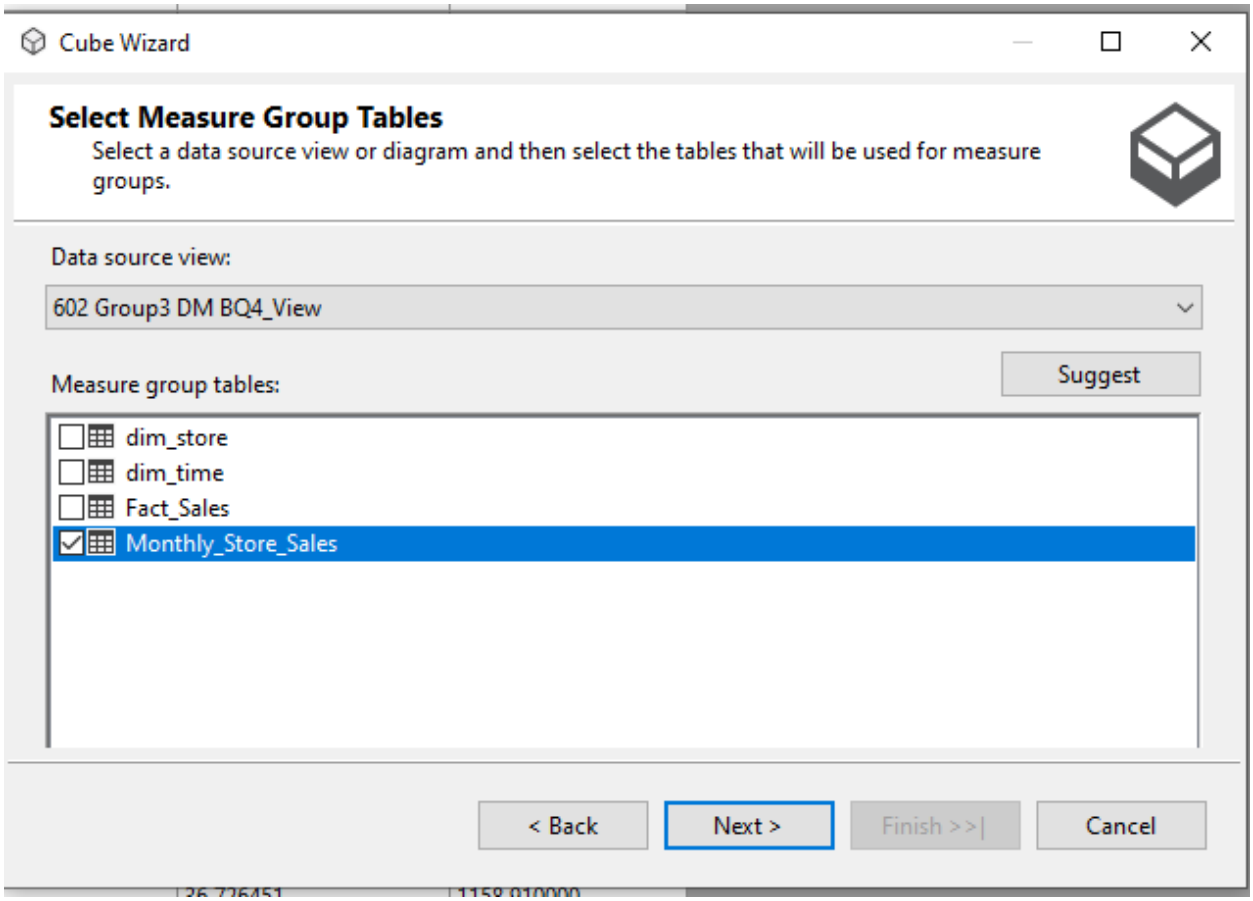


Fig: Selecting Monthly Store Sales as a Measure Group

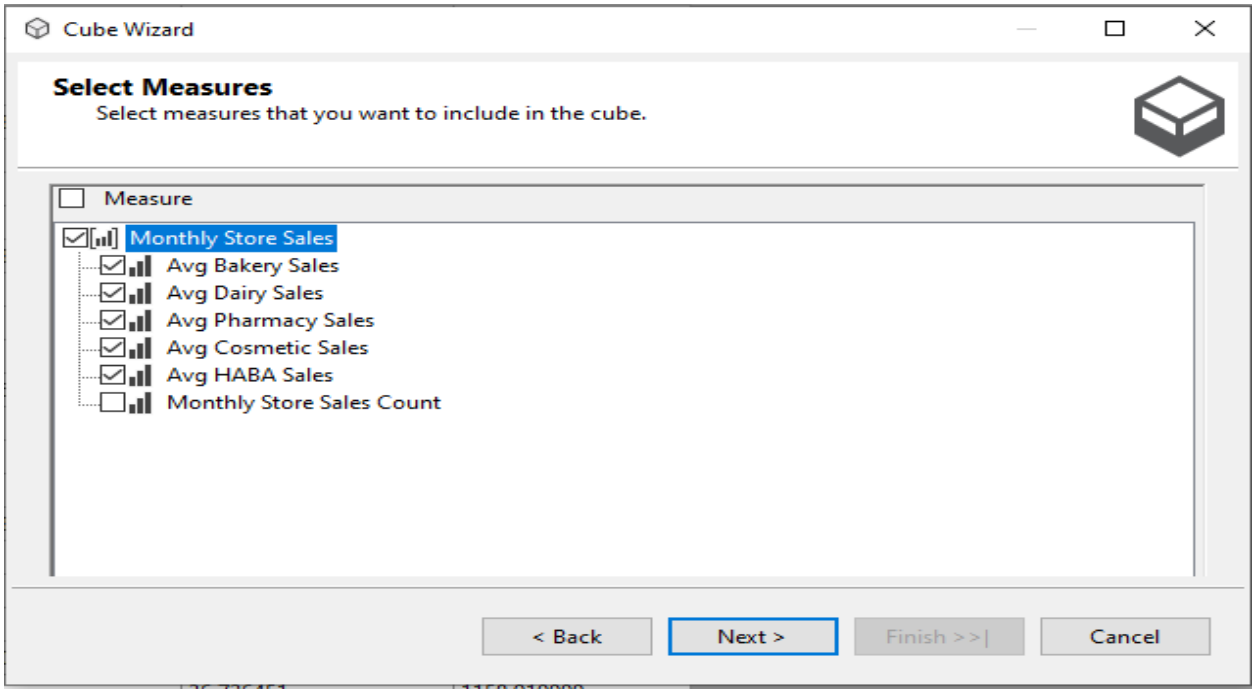


Fig: Selecting measures in the group

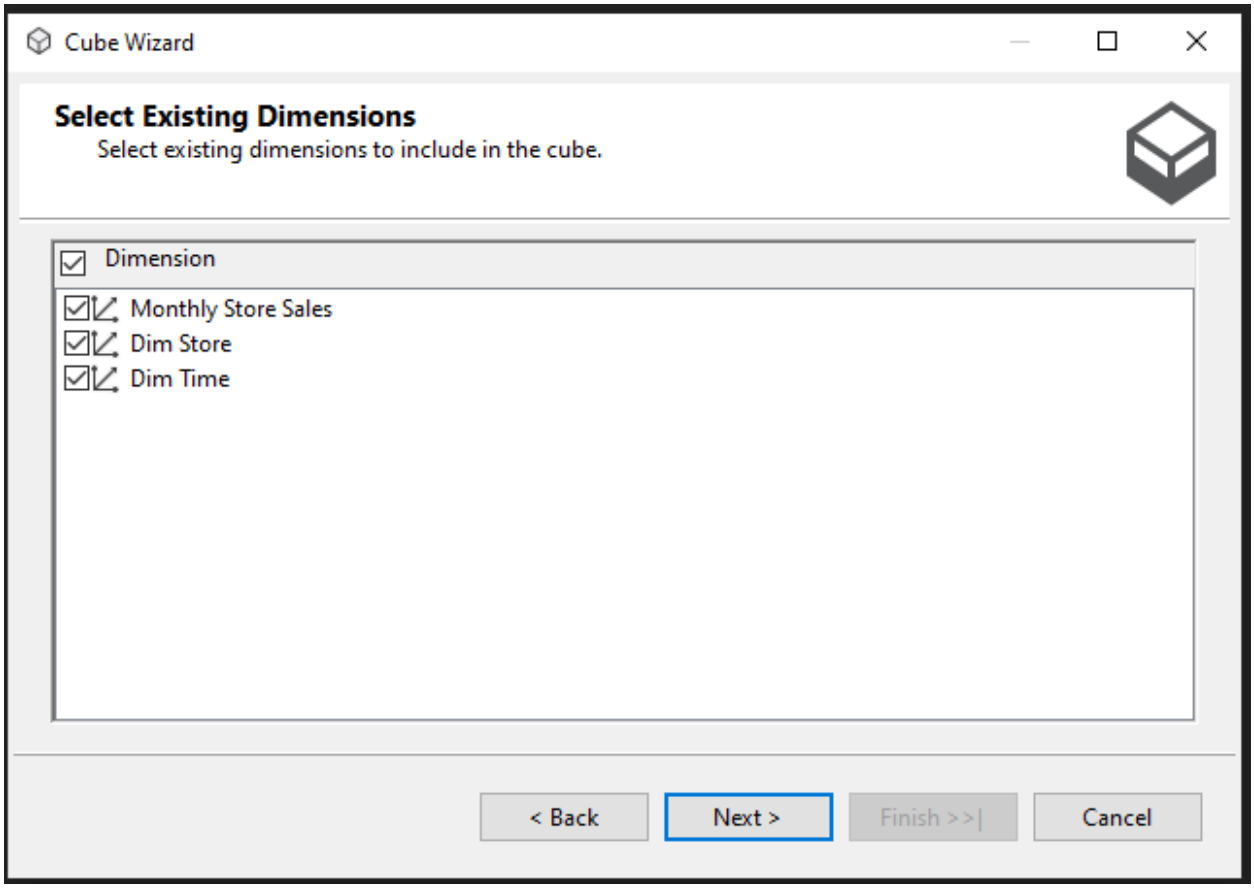


Fig: Selecting Dimensions

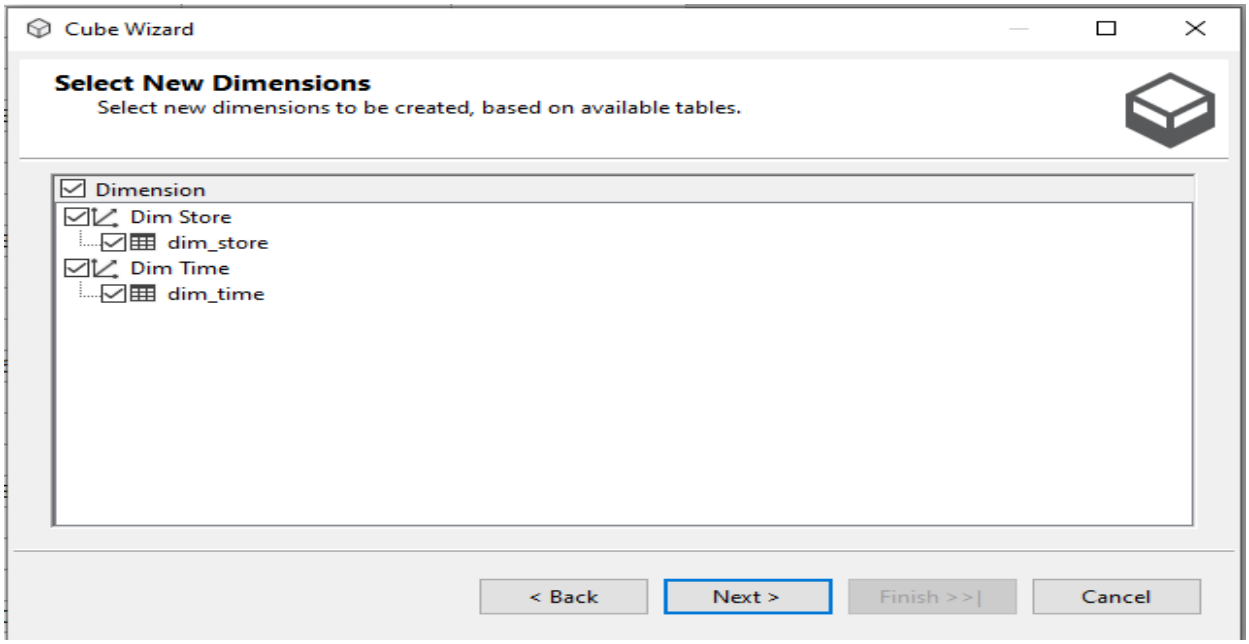


Fig: Finalizing the new dimensions

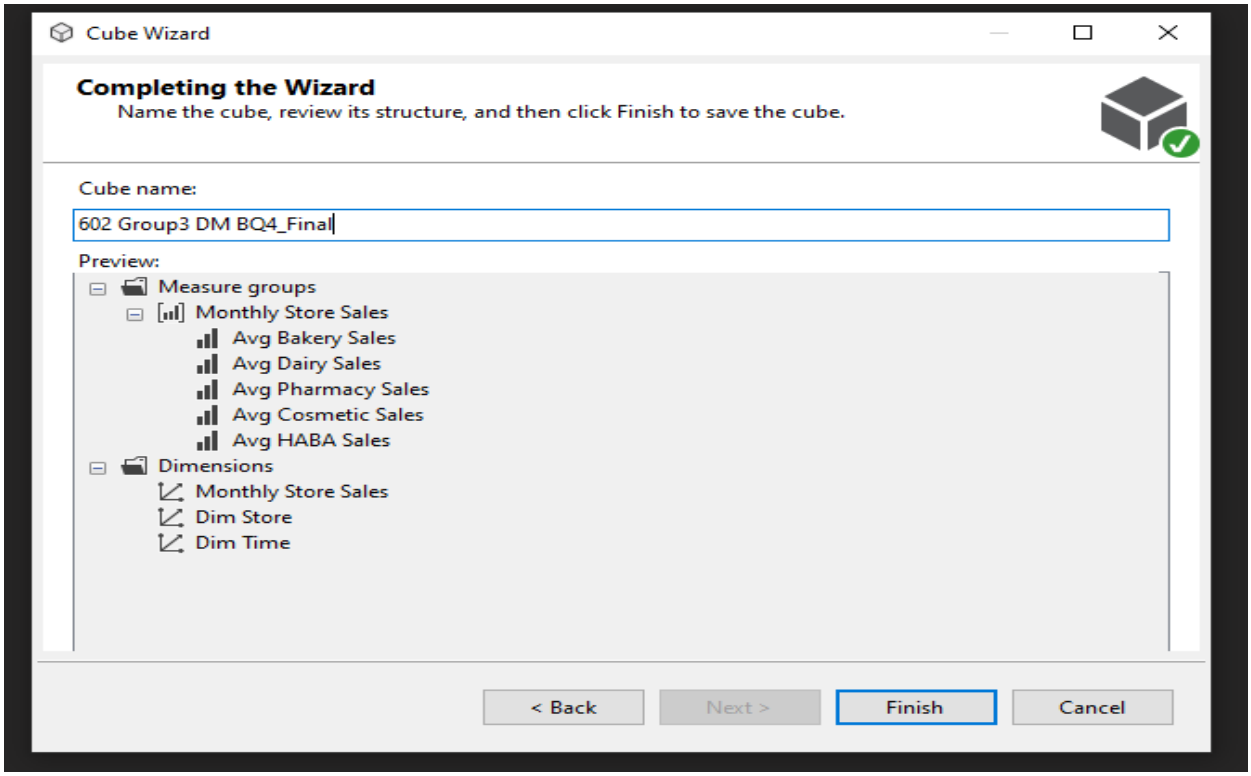


Fig: Cube set up and adding a name

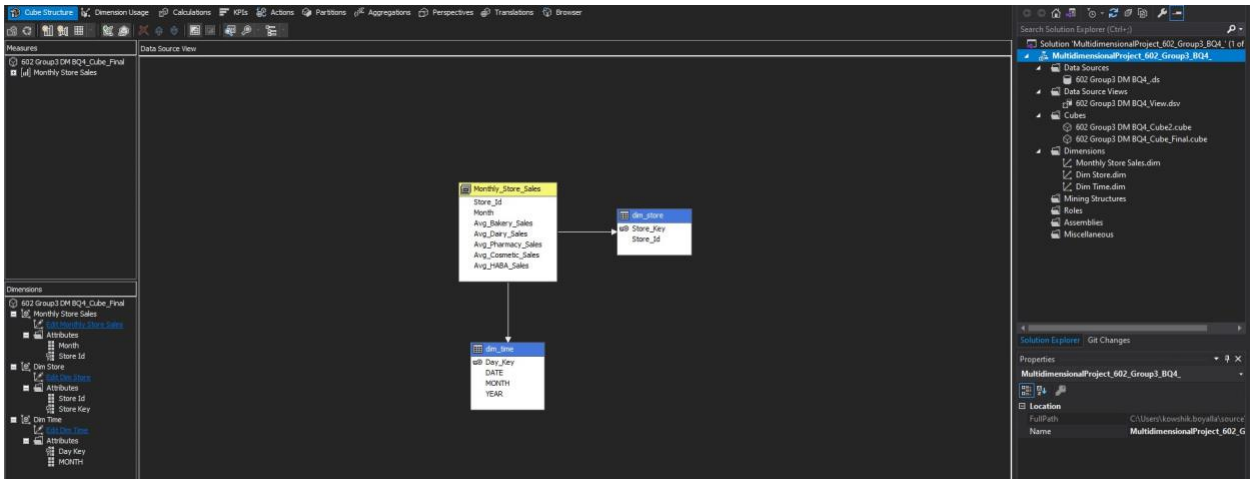


Fig: Successful completion of cube

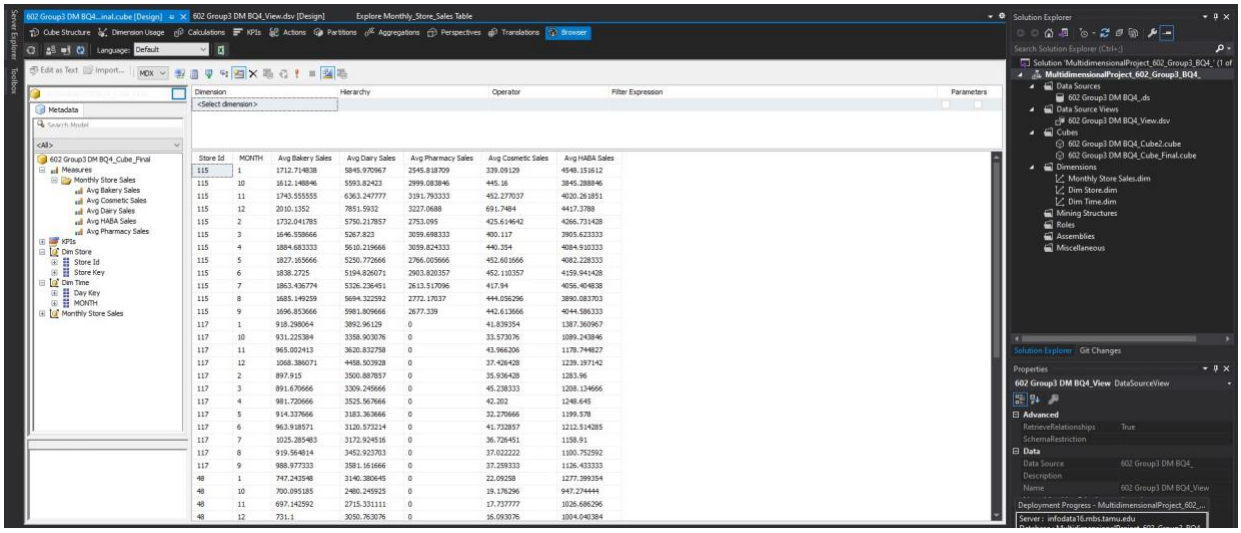


Fig: Browsing the cube data

Step 7: Deploying the multidimensional project to SSAS. First, going to the project properties and modifying the server name. Adding infodata16.mbs.tamu.edu as server.

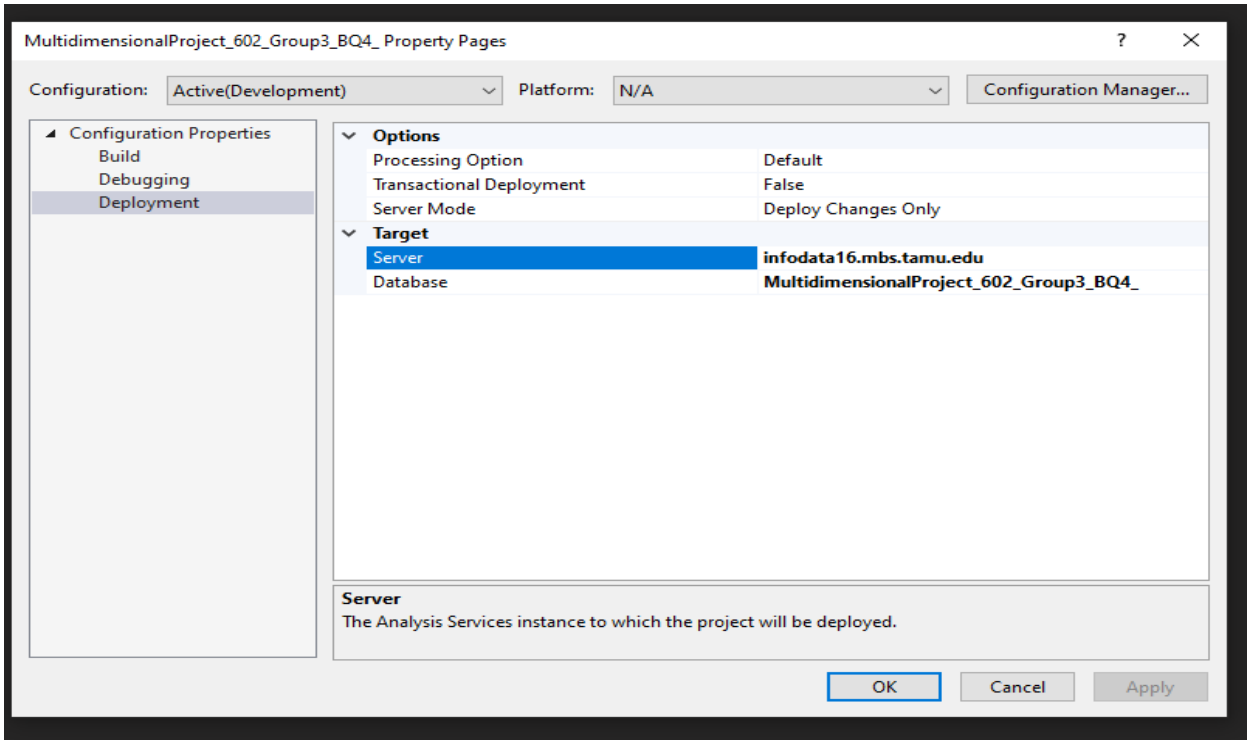


Fig: Making a connection to SSAS (SQL Server)

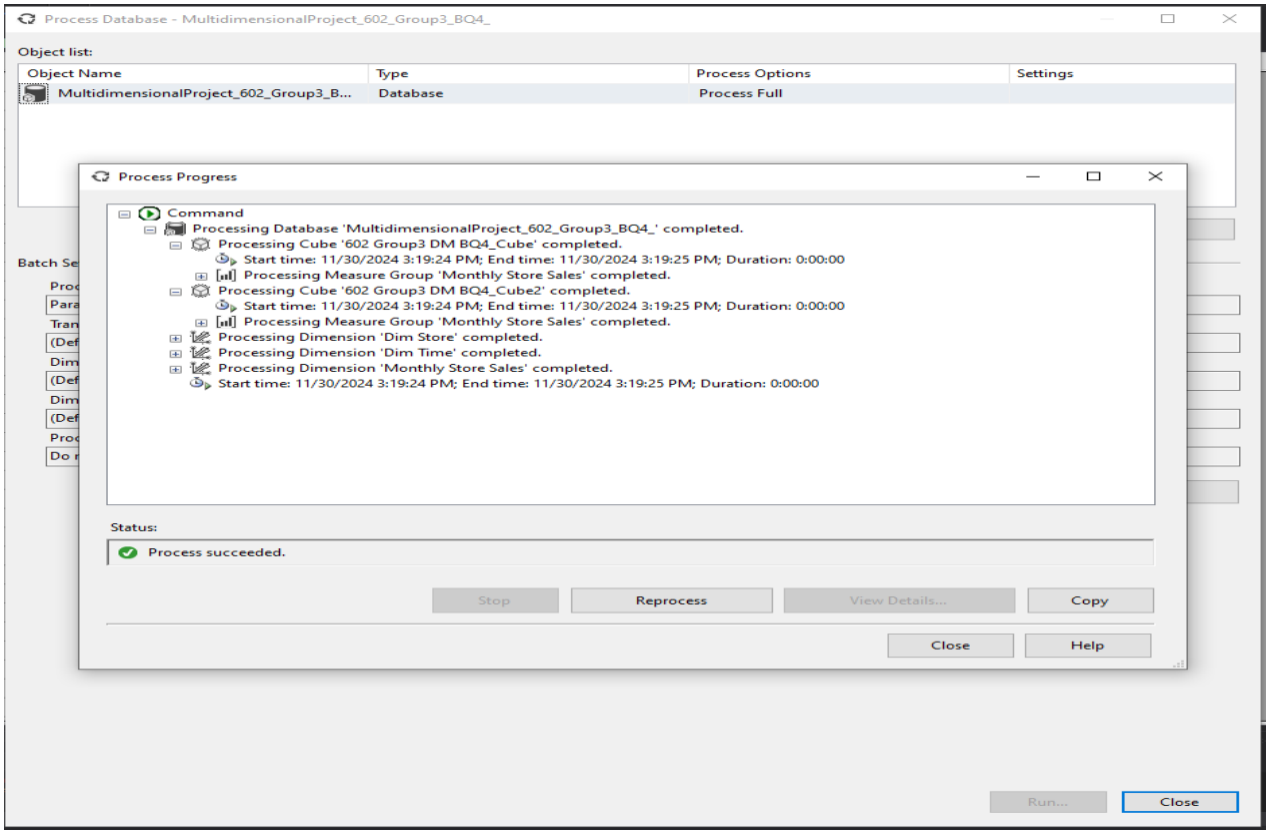


Fig: Deploying the project

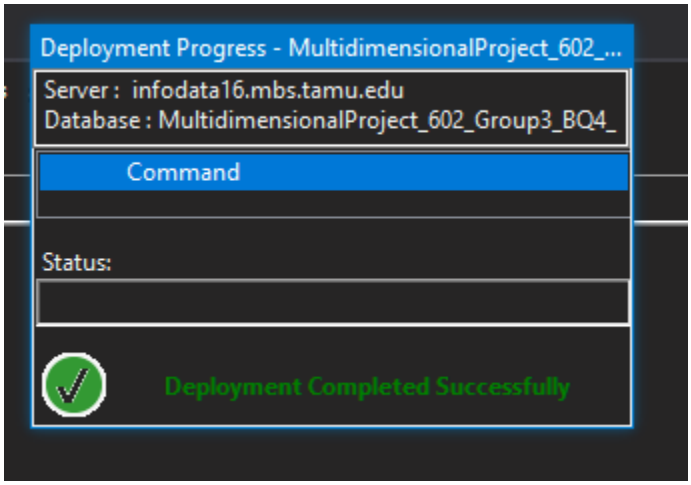


Fig: Successful completion of deployment

Step 8: Validating deployment in SSAS

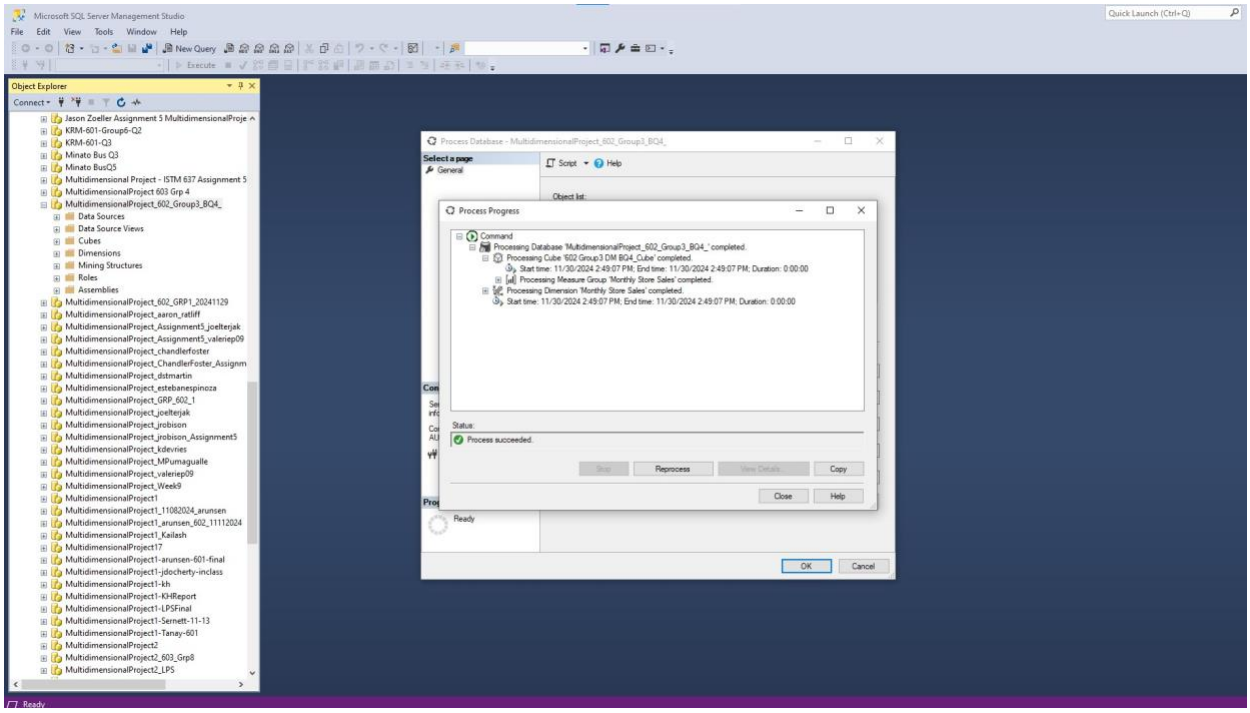


Fig: Processing deployed cube in SSAS

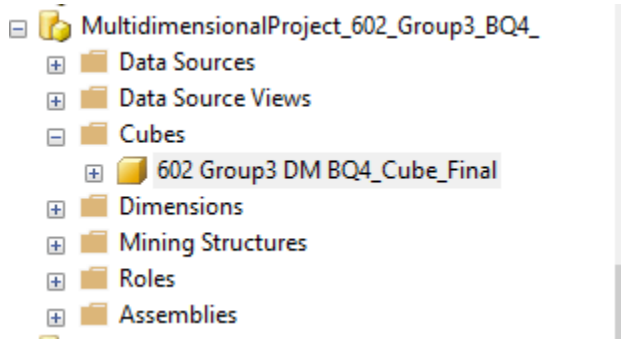


Fig: Multidimensional project in SSAS

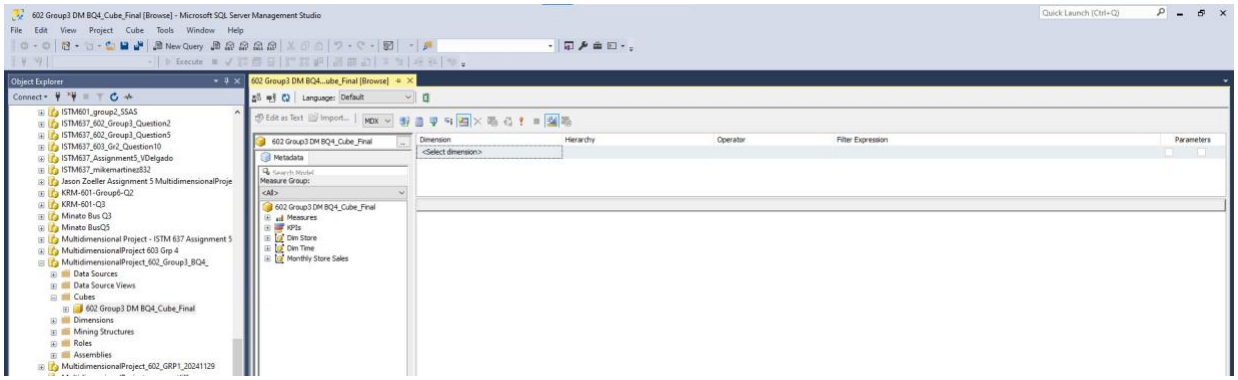


Fig: Creating query to browse cube information

To create a query, we just have to drag and drop the measures and dimensions. After adding them to the panel, click on the execute query.

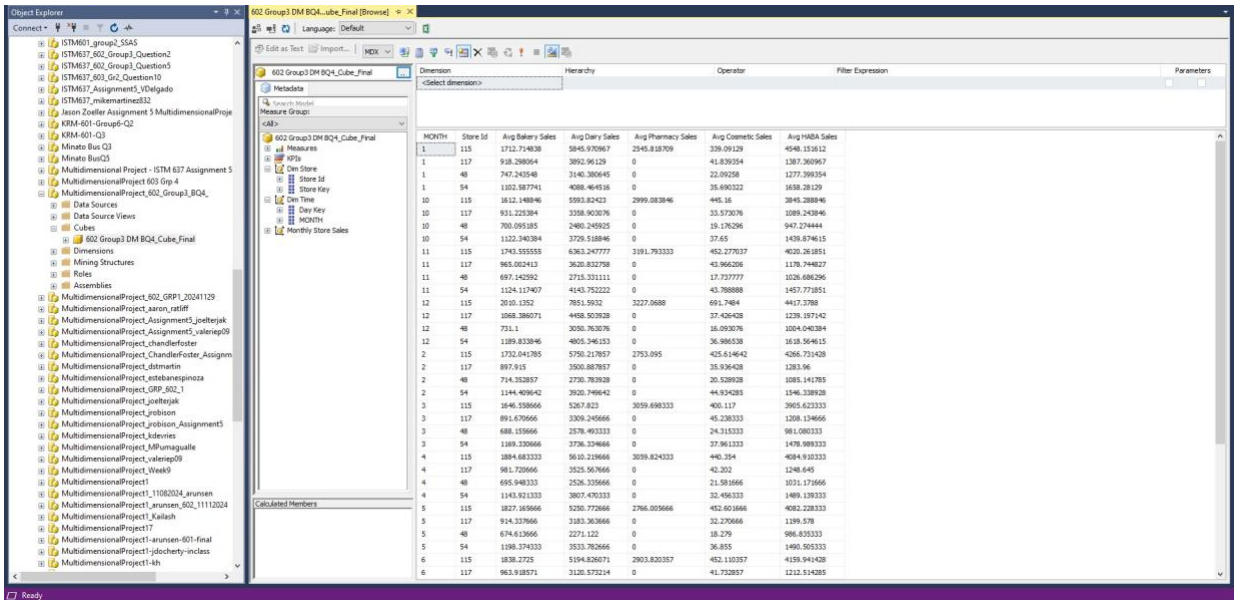


Fig: Query result - Average monthly sales data across departments for all stores

Step 9: Connecting SSAS with Tableau

To connect SSAS with Tableau - Open Tableau and select Microsoft SQL Server Analysis Services from “To A Server” section and enter server name (infodata16.mbs.tamu.edu) and sign in.

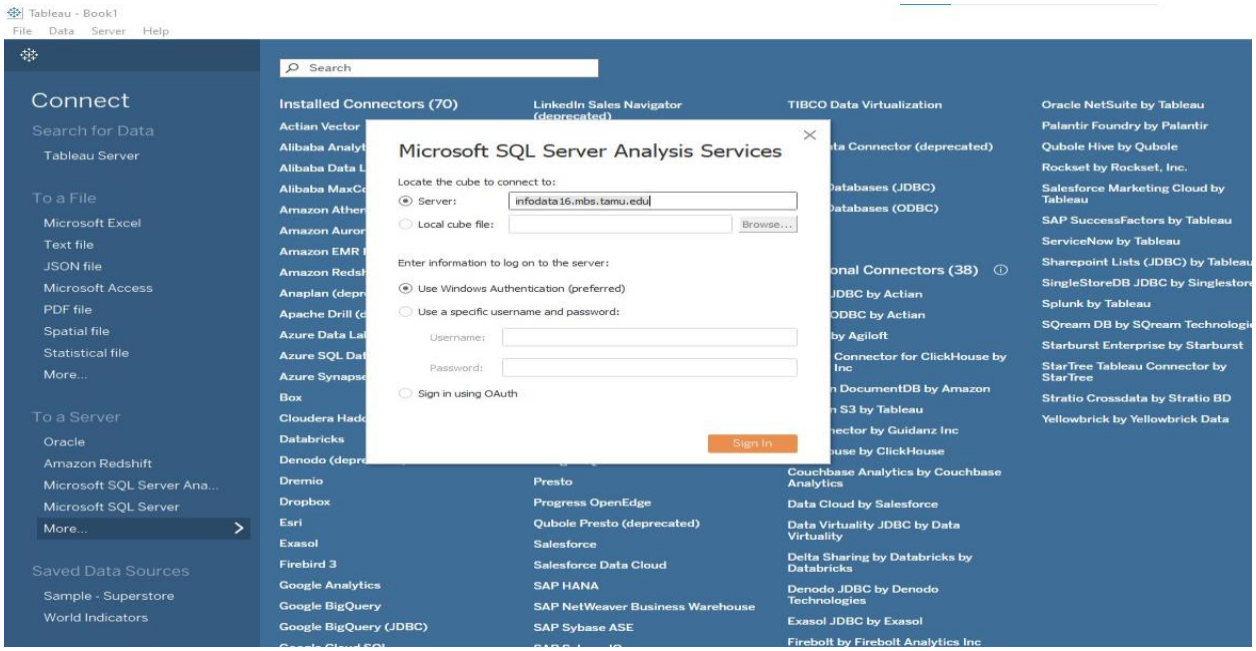


Fig: Connecting SSAS with Tableau

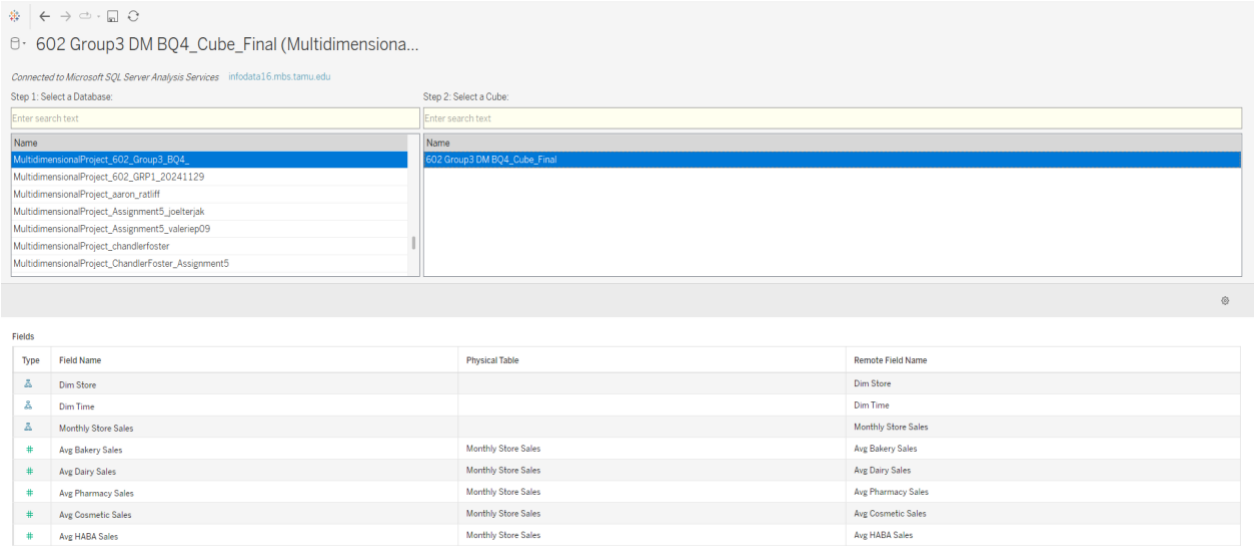


Fig: Selecting the project and the cube

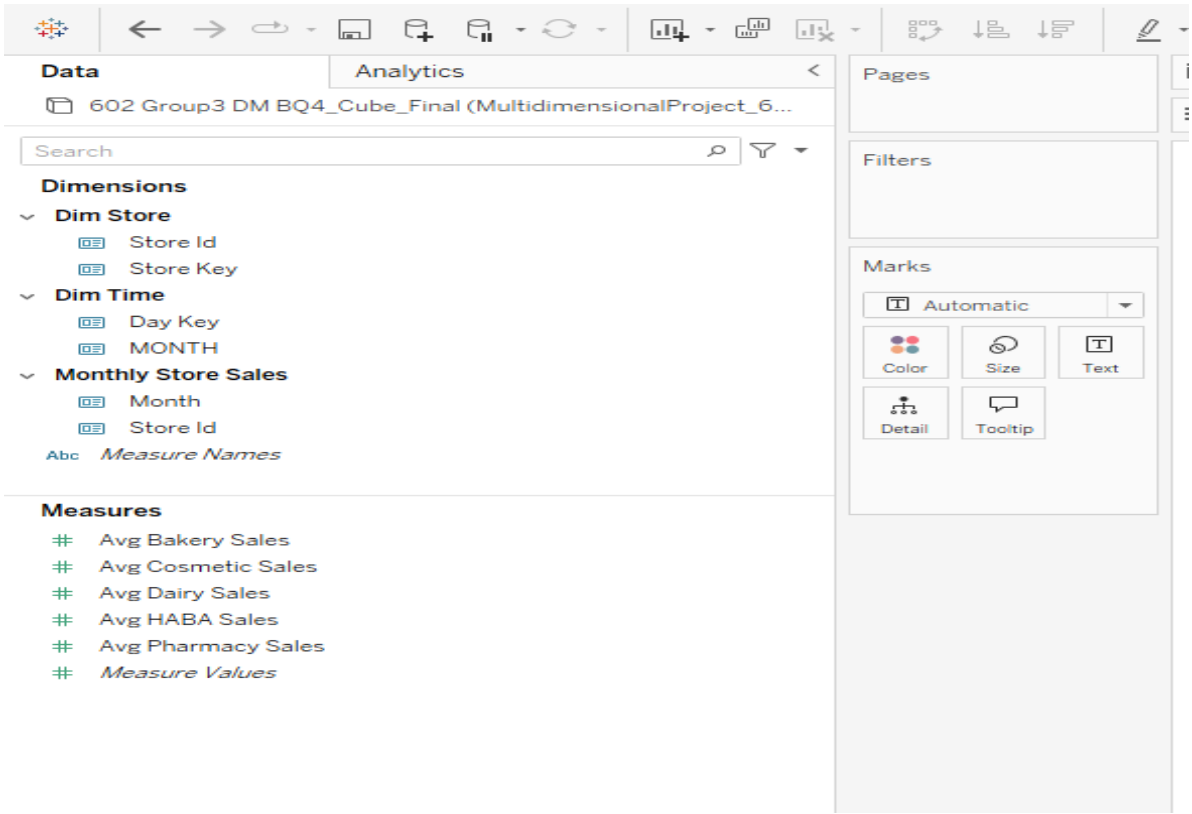


Fig: Cube Dimensions, Measures in Tableau sheet

Step 10: Creating a visualization in Tableau

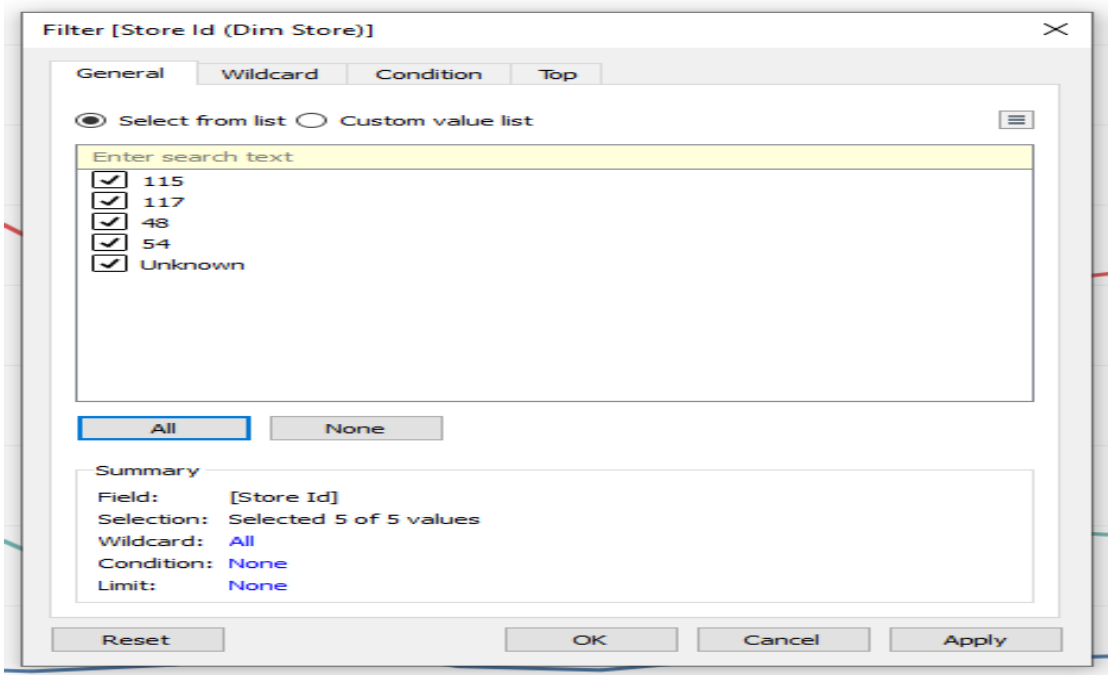


Fig: Creating a filter with the help of Store_Id

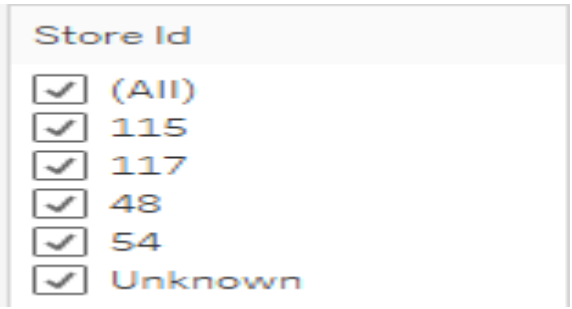


Fig: Successful creation of Store_Id filter

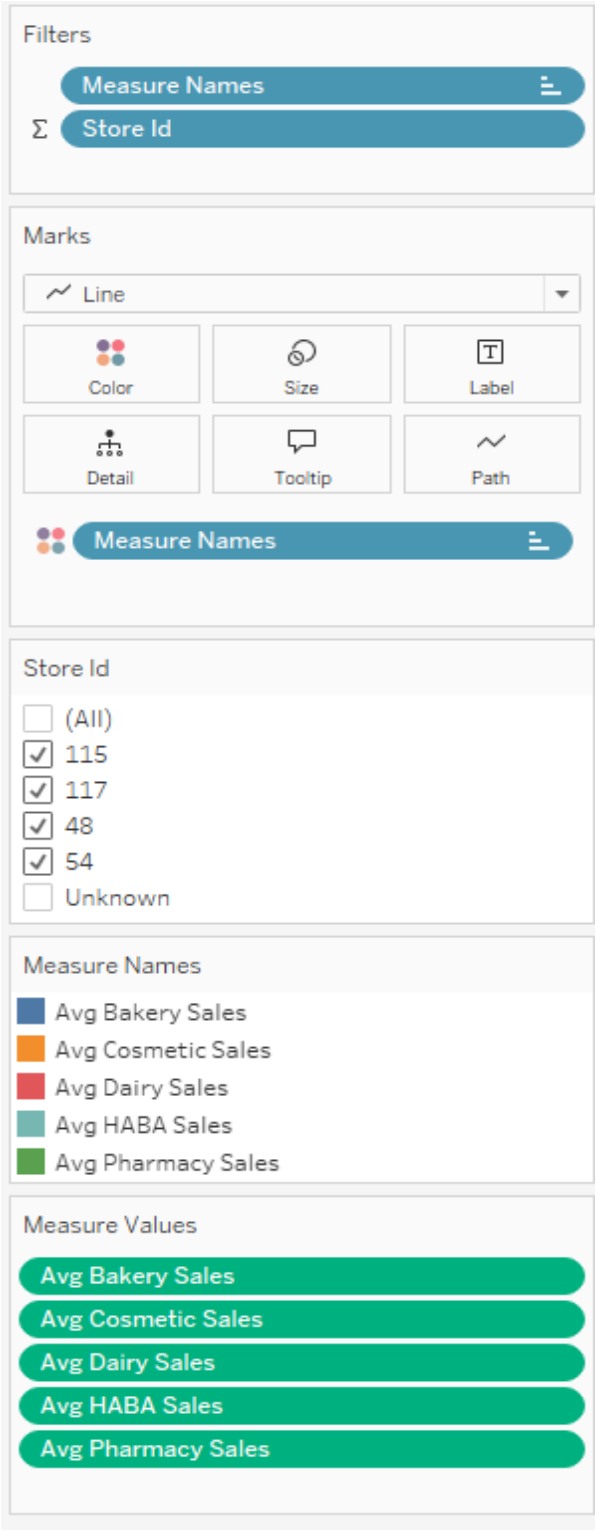


Fig: Tableau panel with Measure names, values and filter

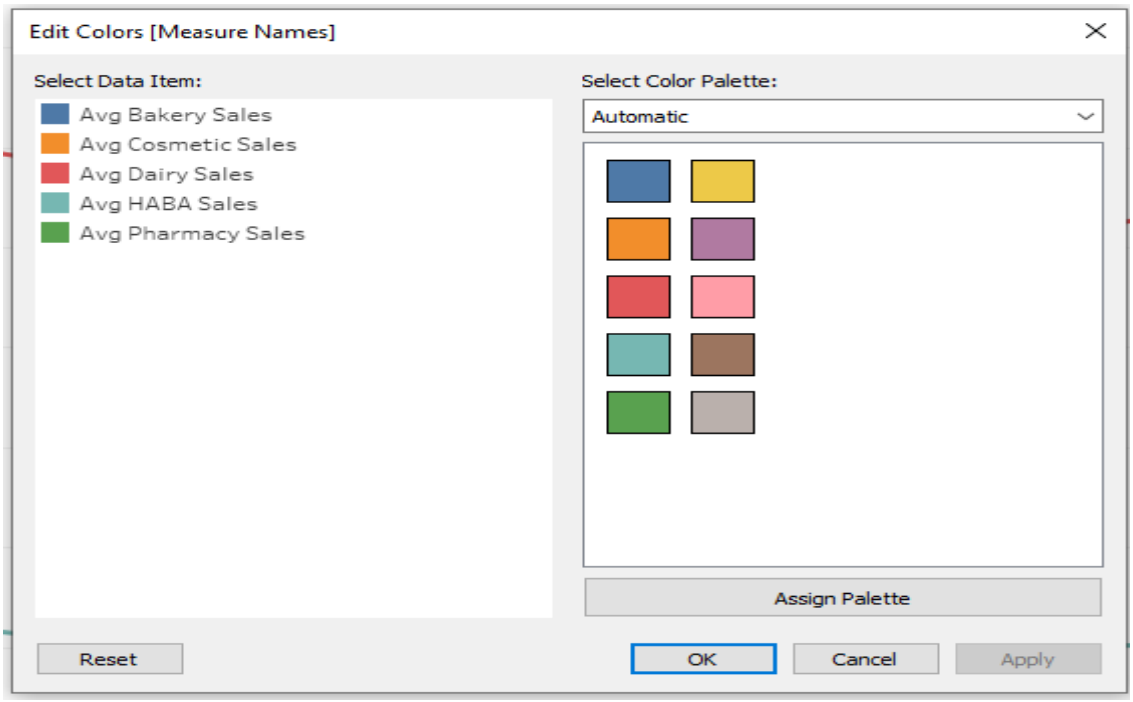


Fig: Representation of department monthly averages with colors

1. Created a Line chart by mentioning Month in Columns (X-axis) and Measure Values (Avg_Bakery_Sales, Avg_Dairy_Sales, Avg_Cosmetic_Sales, Avg_HABA_Sales, Avg_Pharmacy_Sales) as rows (Y-axis).
2. Used different colors to represent different departments.

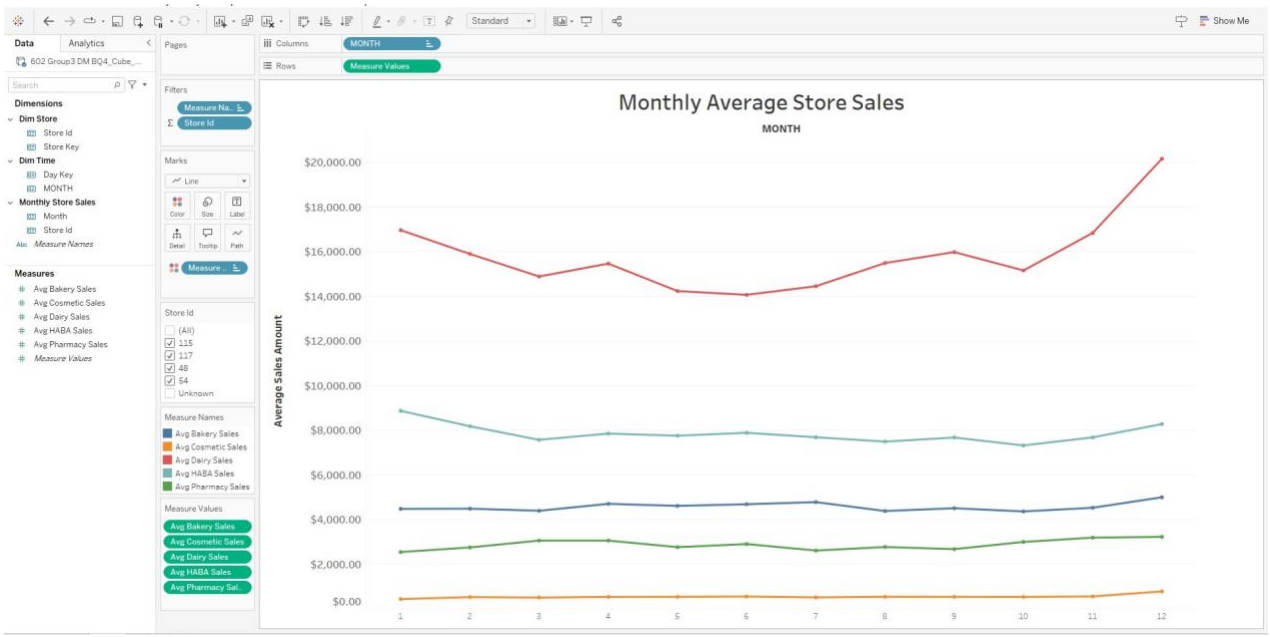


Fig: Average sales across all stores for different departments

3. As shown in the figure, different color lines indicate average monthly sales across selected stores. According to the chart, the monthly average sales of the Diary Department is higher compared to all other departments when we select all stores. It especially peaked in Month 12 (December) with $20,166.21. The monthly average sales for cosmetics department is very low compared to all other departments.

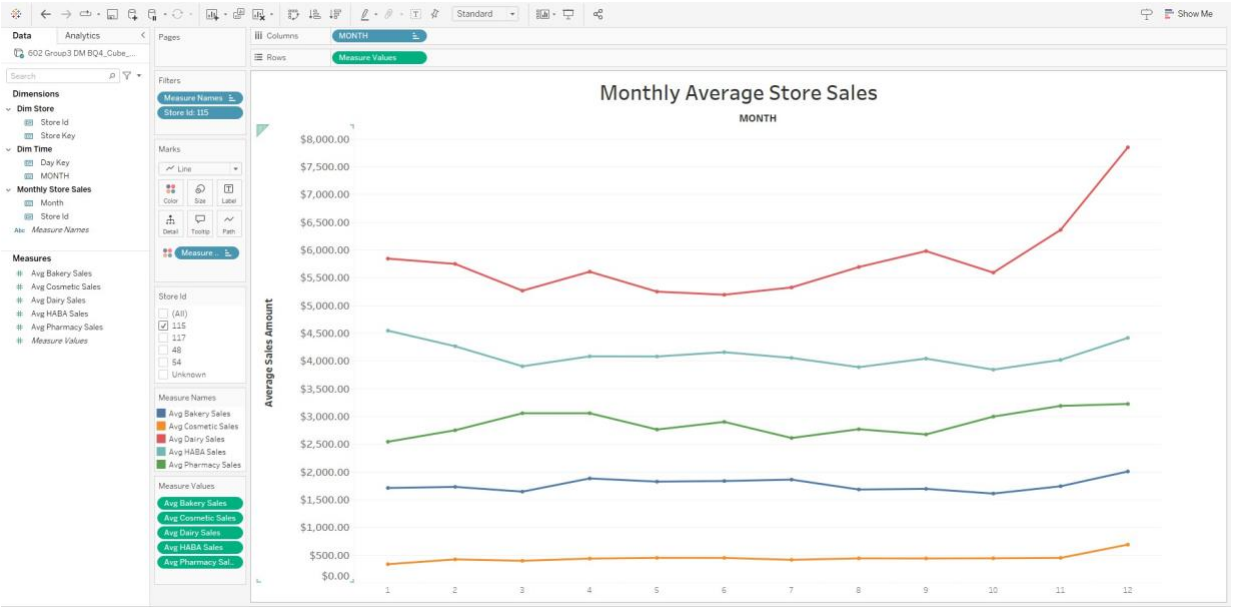


Fig: Average sales of departments for store - 115

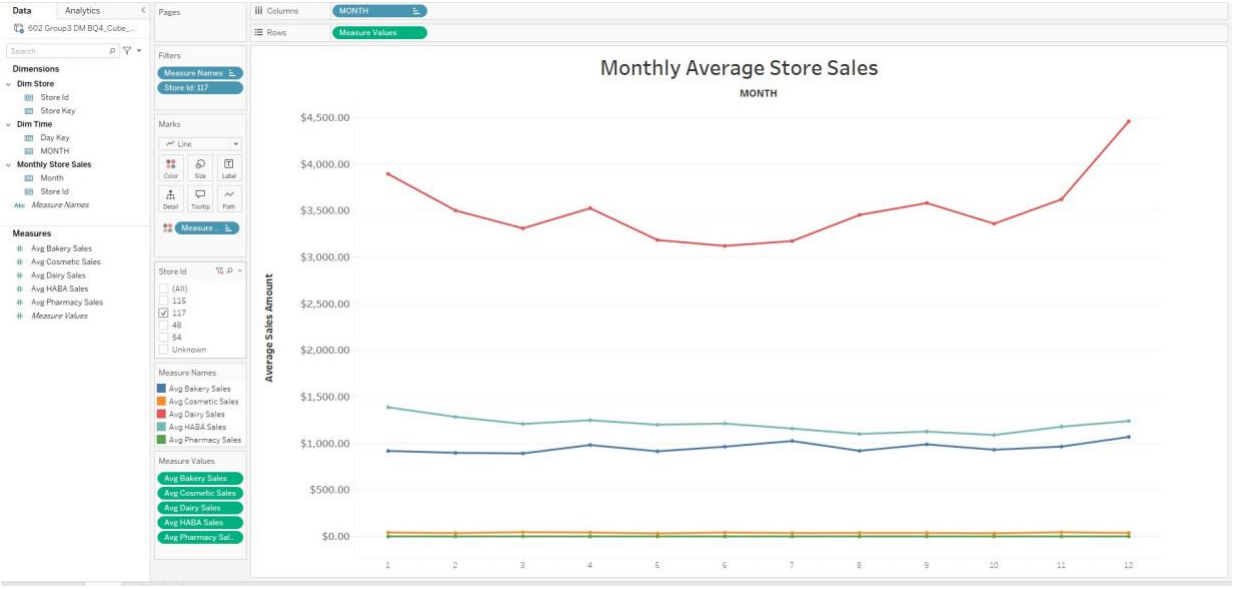


Fig: Average sales of departments for store - 117

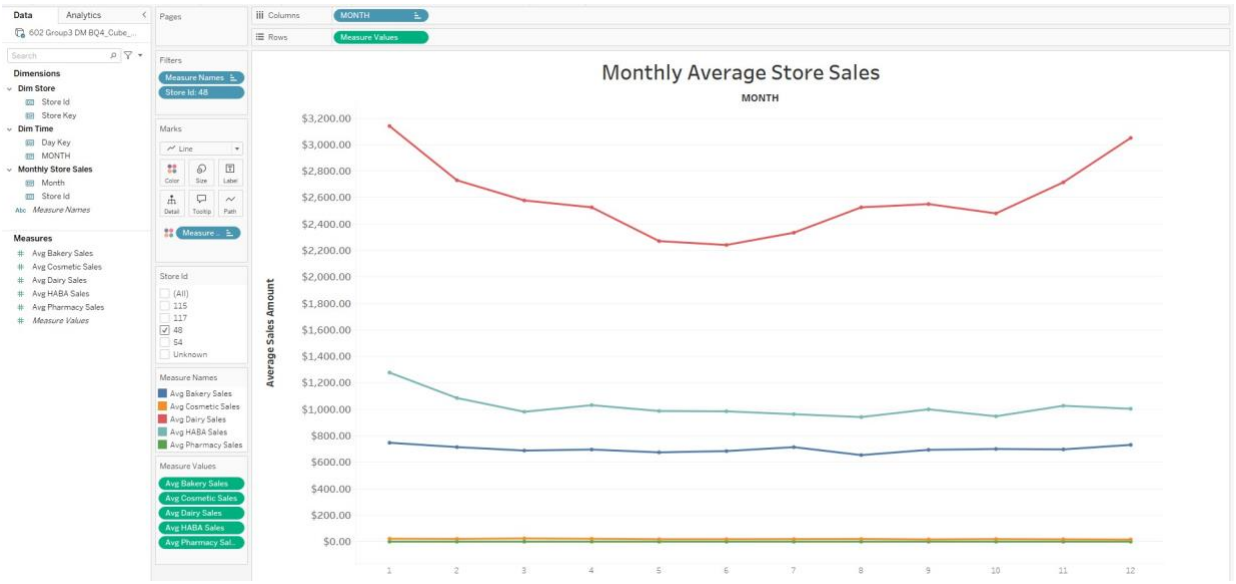


Fig: Average sales of departments for store - 48

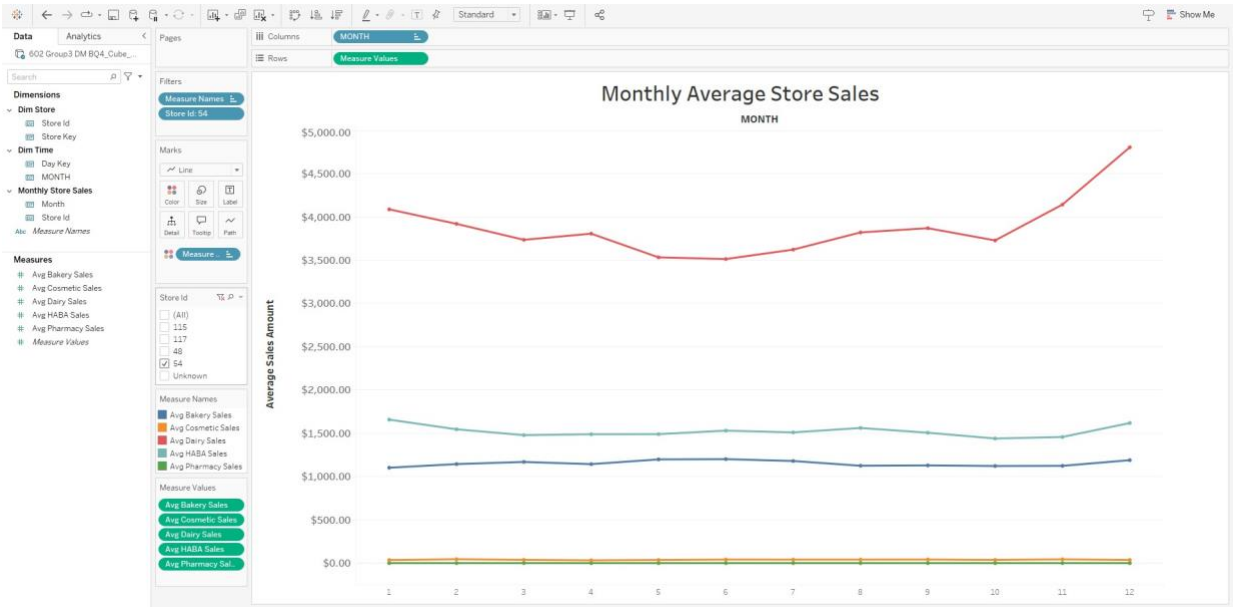


Fig: Average sales of departments for store - 54

The Store_Id filter and the different lines of departments are so much helpful to analyze the performance of different departments across all stores or for a selected store.

5. What were the highest sales contributions from single and retired individuals in the Buffalo Grove stores for BUDWEISER BEER N.R.B during the Thanksgiving week of 1993?

Support: This analysis helps **Dominick's Fine Foods** by revealing how **single** and **retired** shoppers contribute to sales during **Thanksgiving week** (220) in the year 1993 alone at **Buffalo Grove stores** (112). Understanding demographic-driven sales for high-demand products like **BUDWEISER BEER N.R.B** enables DFF to optimize promotions and inventory for these customer segments. It allows for more targeted marketing and better stocking strategies, helping the retailer maximize sales and meet customer demand during key holiday periods.

Method: The report created to answer this business question from independent Data Marts using SSRS is employed here. This report is published to infodata16.mbs.tamu.edu Report Server

SSRS:

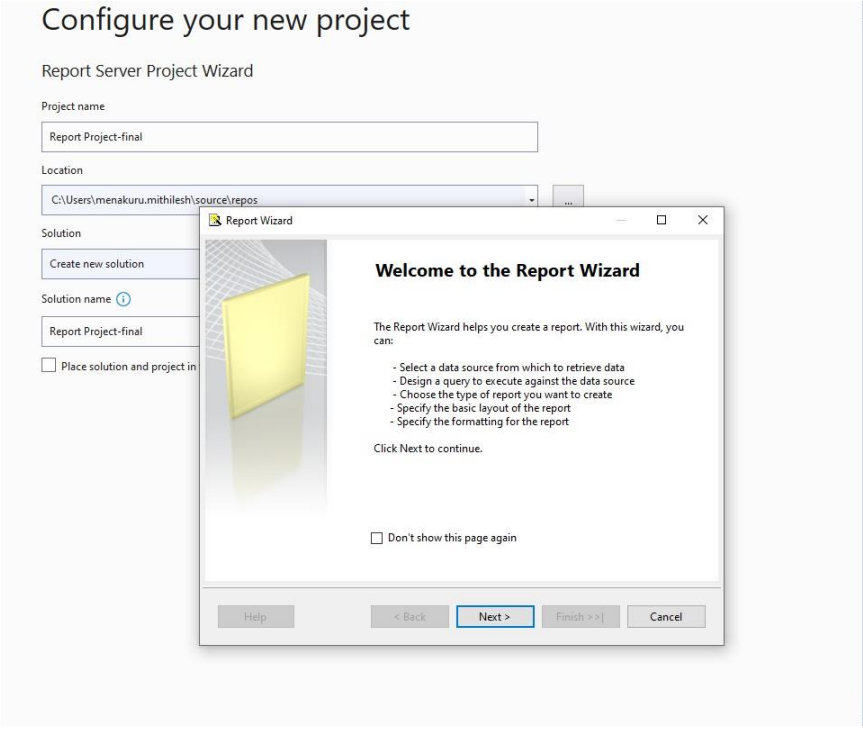


Fig Selecting the report wizard through SSIS

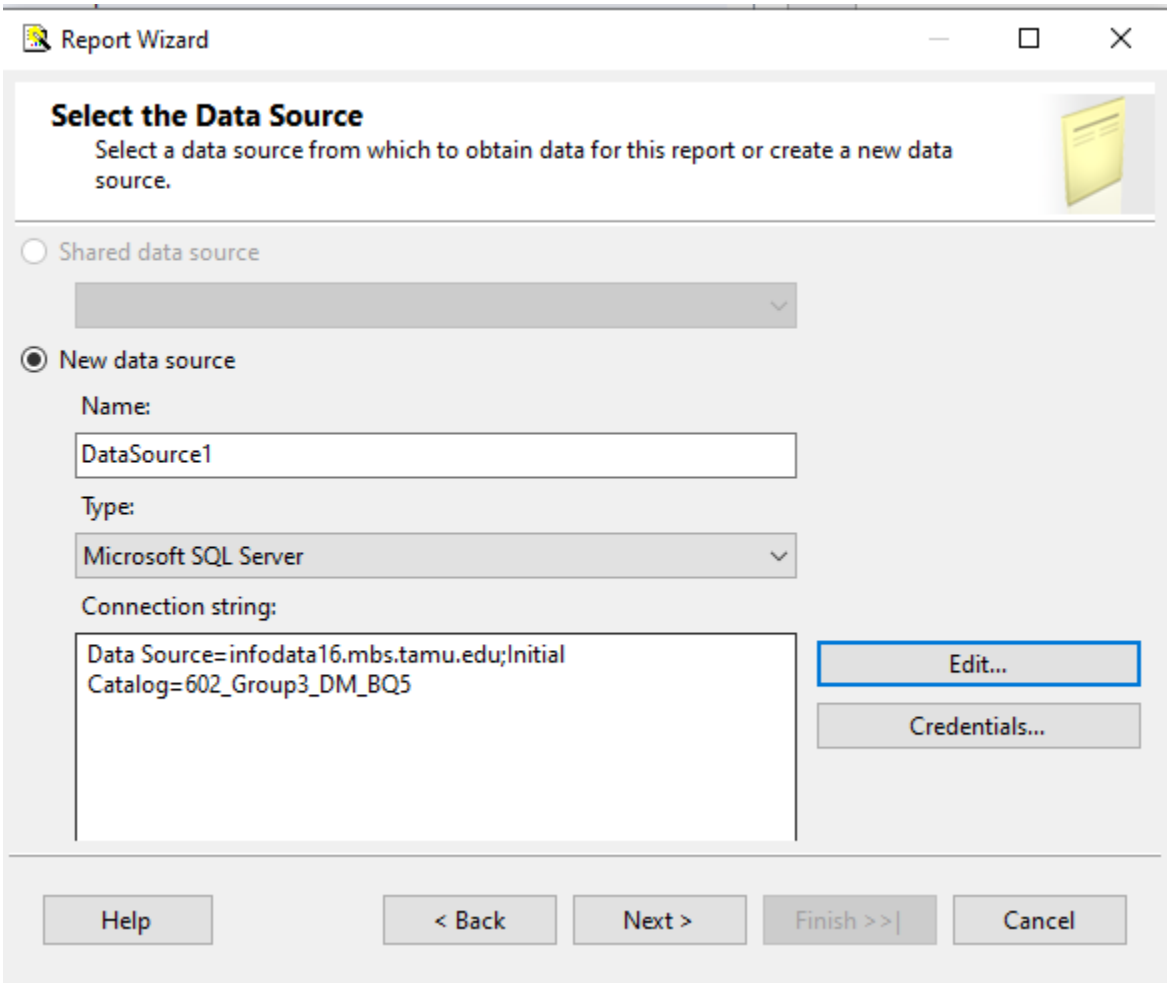


Fig Data Source and catalog String

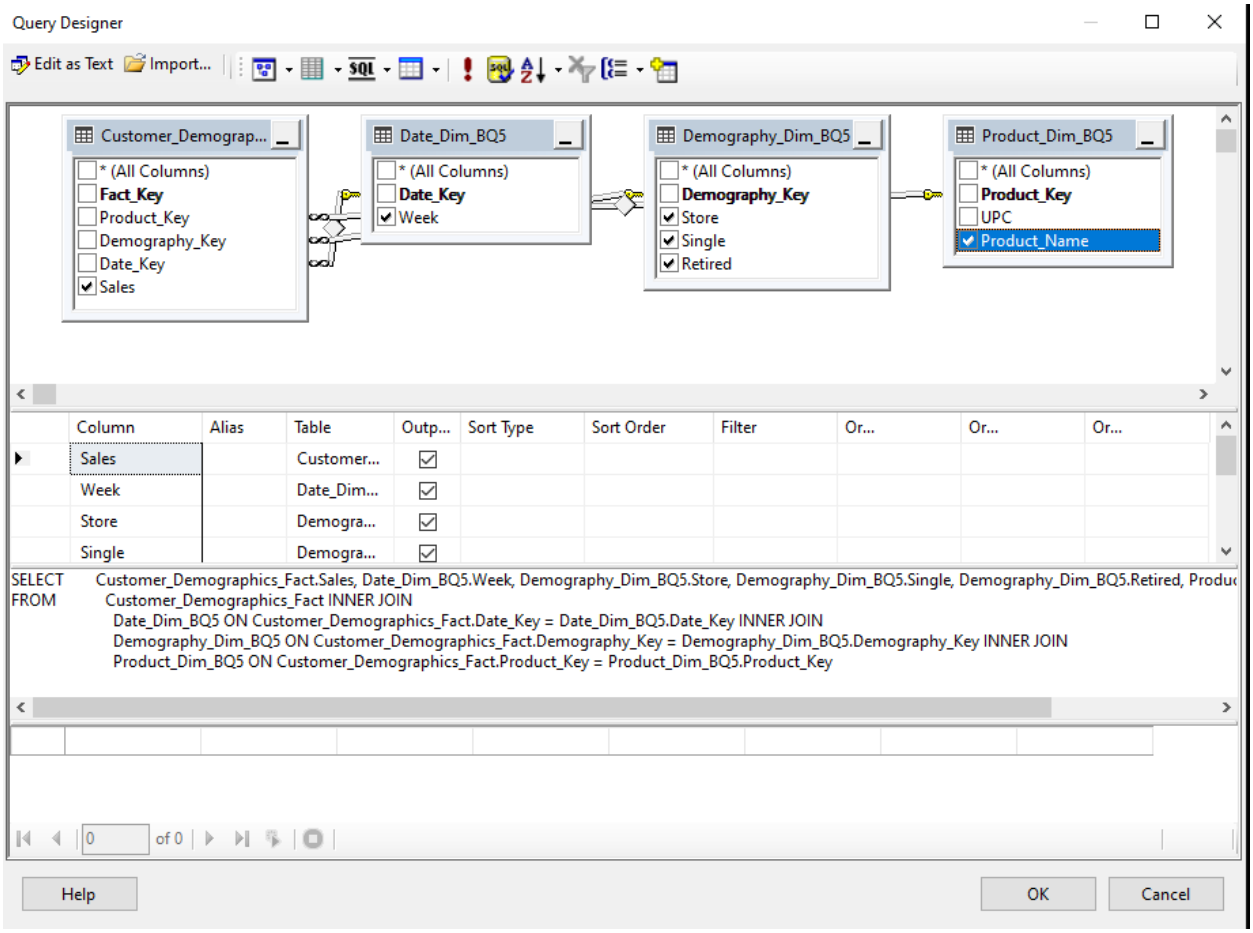


Fig Query Builder

QUERY:

```
SELECT
        Customer_Demographics_Fact.Sales,
        Date_Dim_BQ5.Week,
        Demography_Dim_BQ5.Store,
        Demography_Dim_BQ5.Single,
        Demography_Dim_BQ5.Retired,
        Product_Dim_BQ5.Product_Name
FROM
        Customer_Demographics_Fact
INNER JOIN
        Date_Dim_BQ5
ON
        Customer_Demographics_Fact.Date_Key = Date_Dim_BQ5.Date_Key
INNER JOIN
```

```
        Demography_Dim_BQ5
ON
        Customer_Demographics_Fact.Demography_Key =
Demography_Dim_BQ5.Demography_Key
INNER JOIN
        Product_Dim_BQ5
ON
        Customer_Demographics_Fact.Product_Key = Product_Dim_BQ5.Product_Key
```

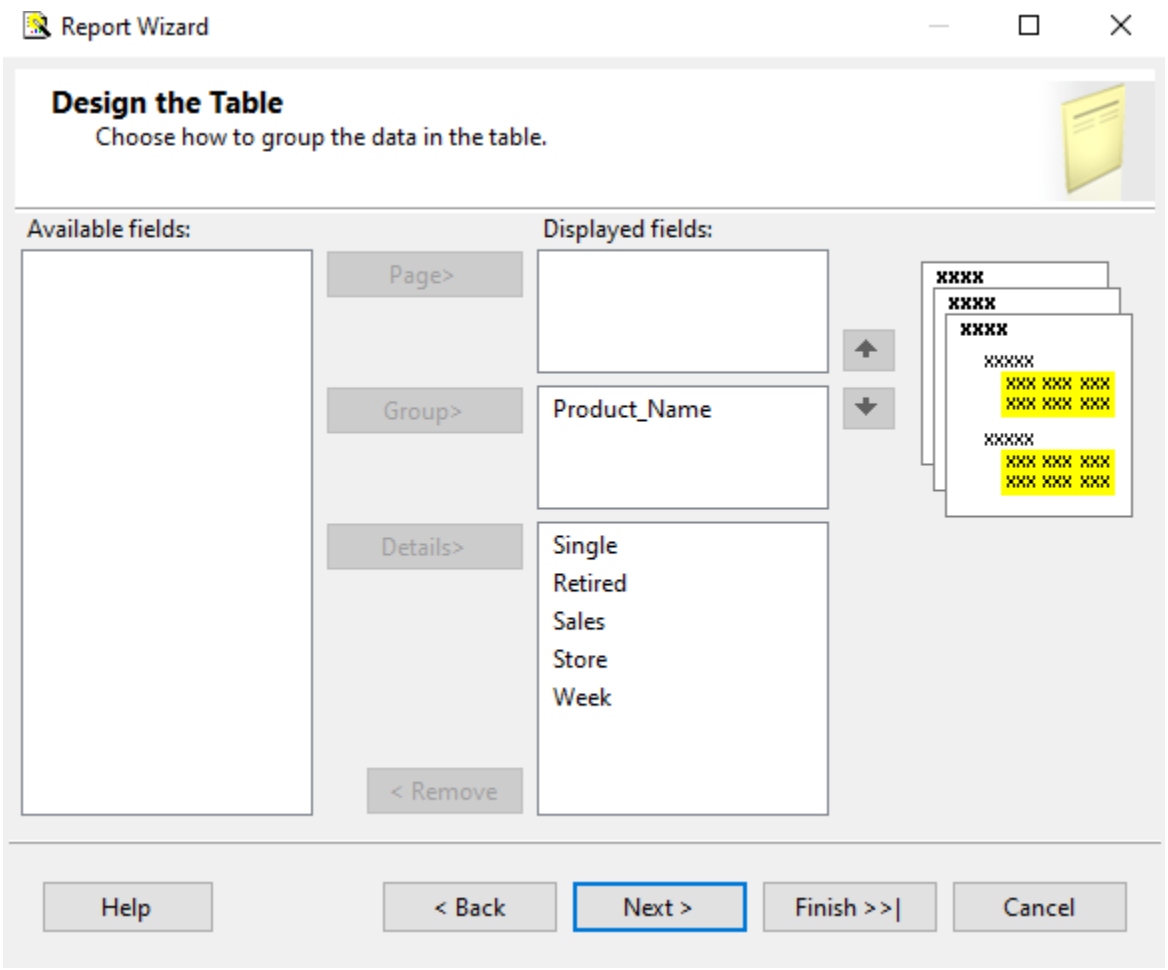


Fig Designing the table

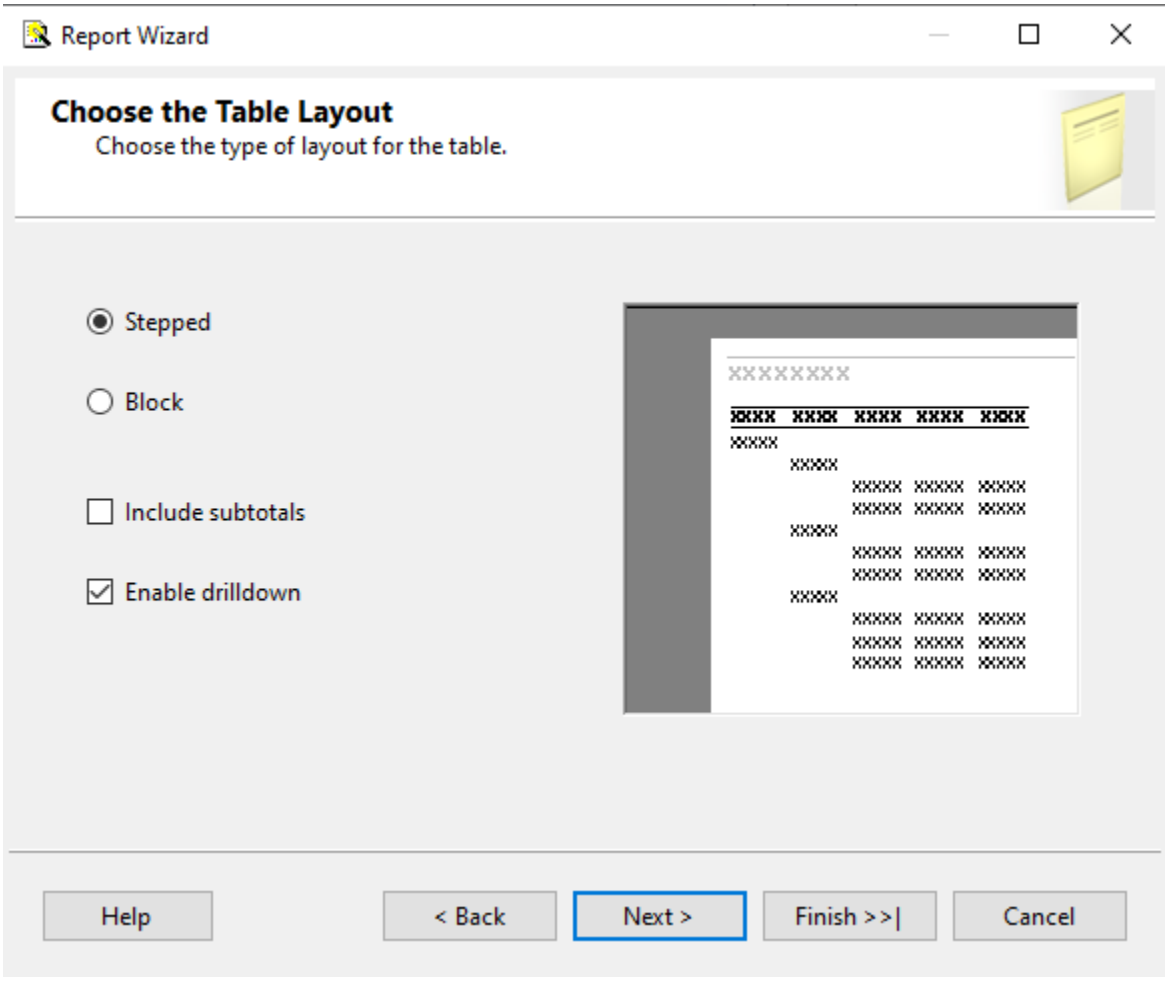


Fig Choosing the table layout

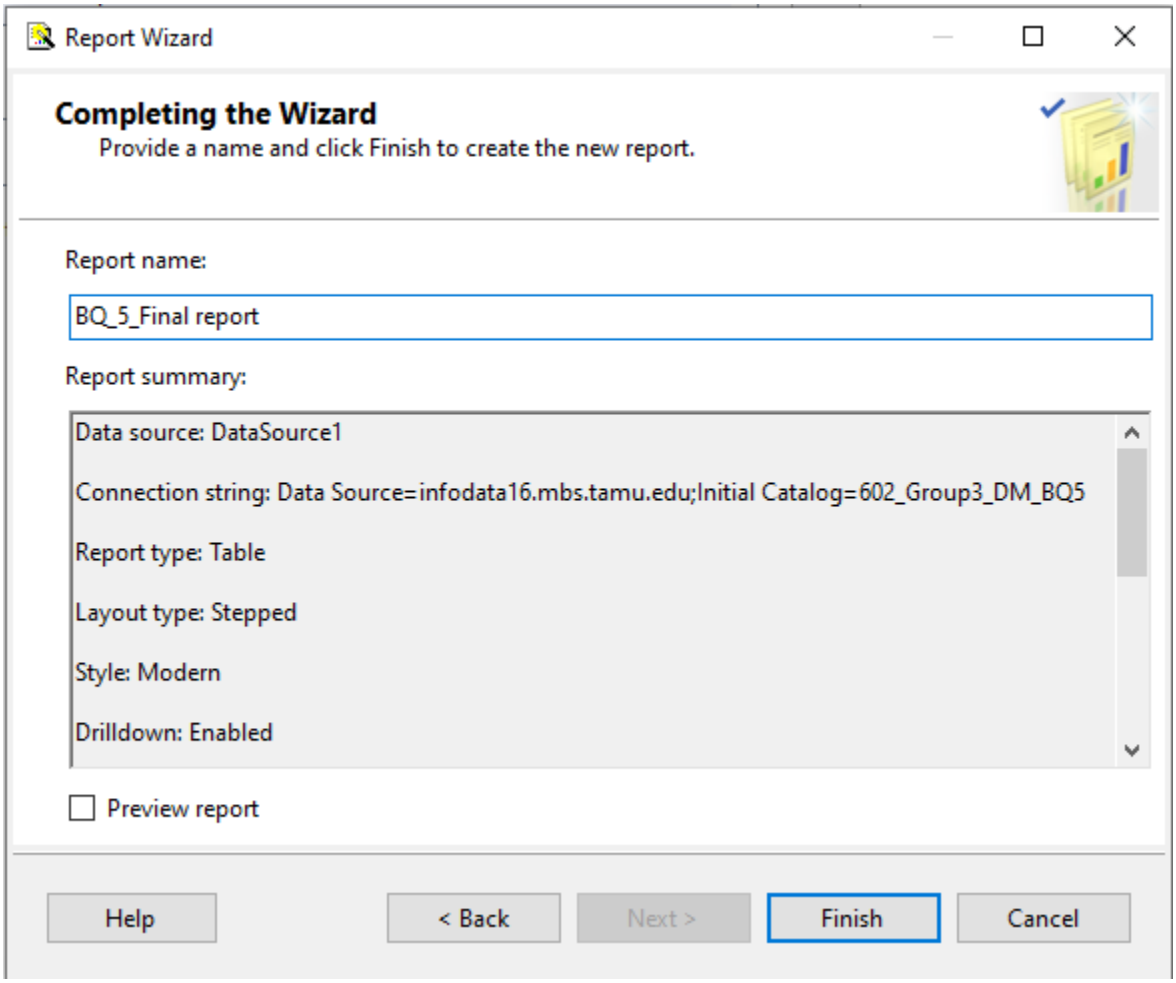


Fig Completing the wizard

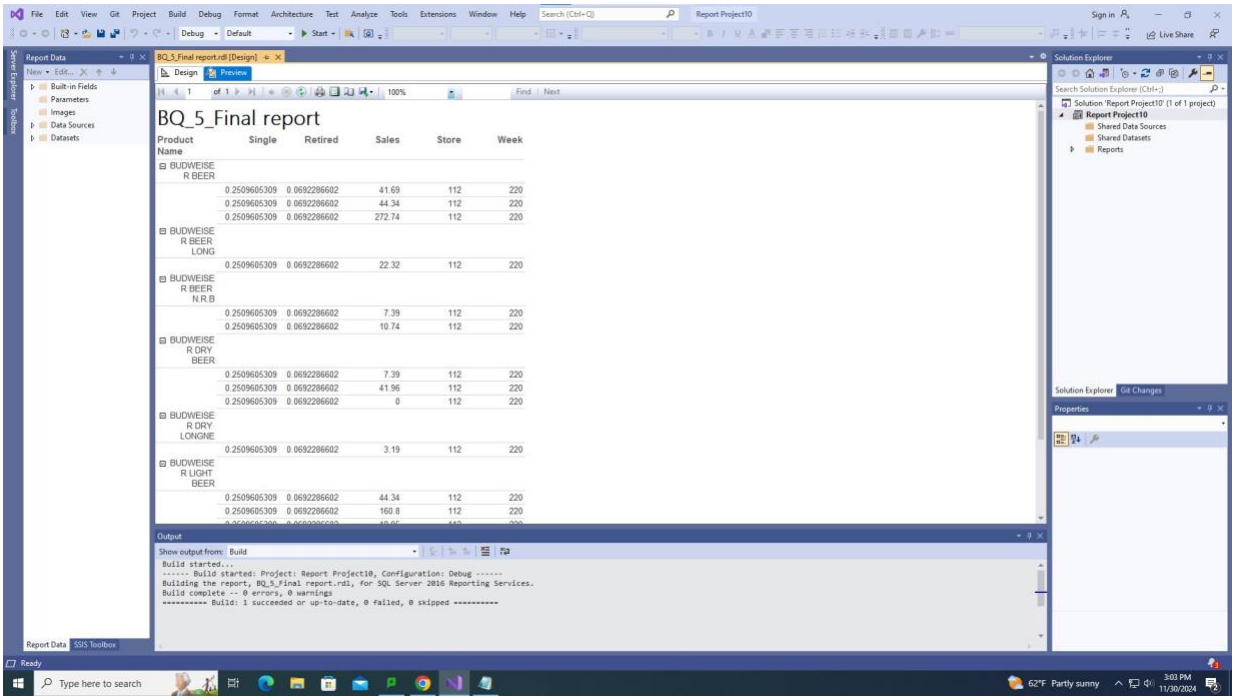


Fig Report Preview successful

Deploying the report to server, "Properties" has target server
infodata16.mbs.tamu.edu - Server

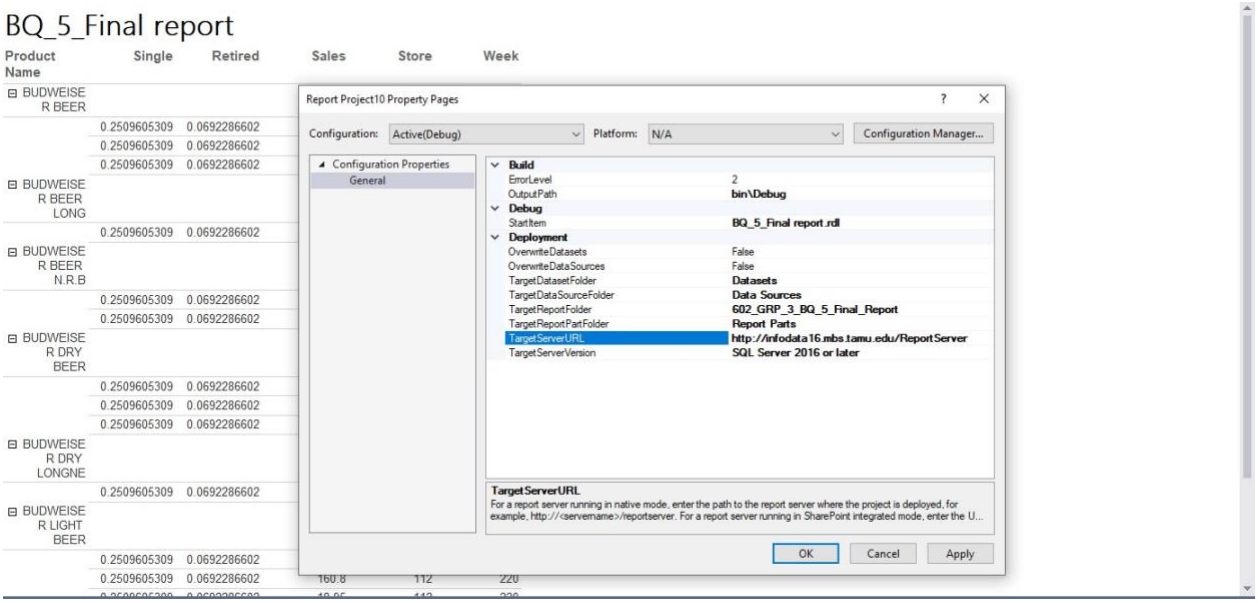


Fig Deploying the report to server

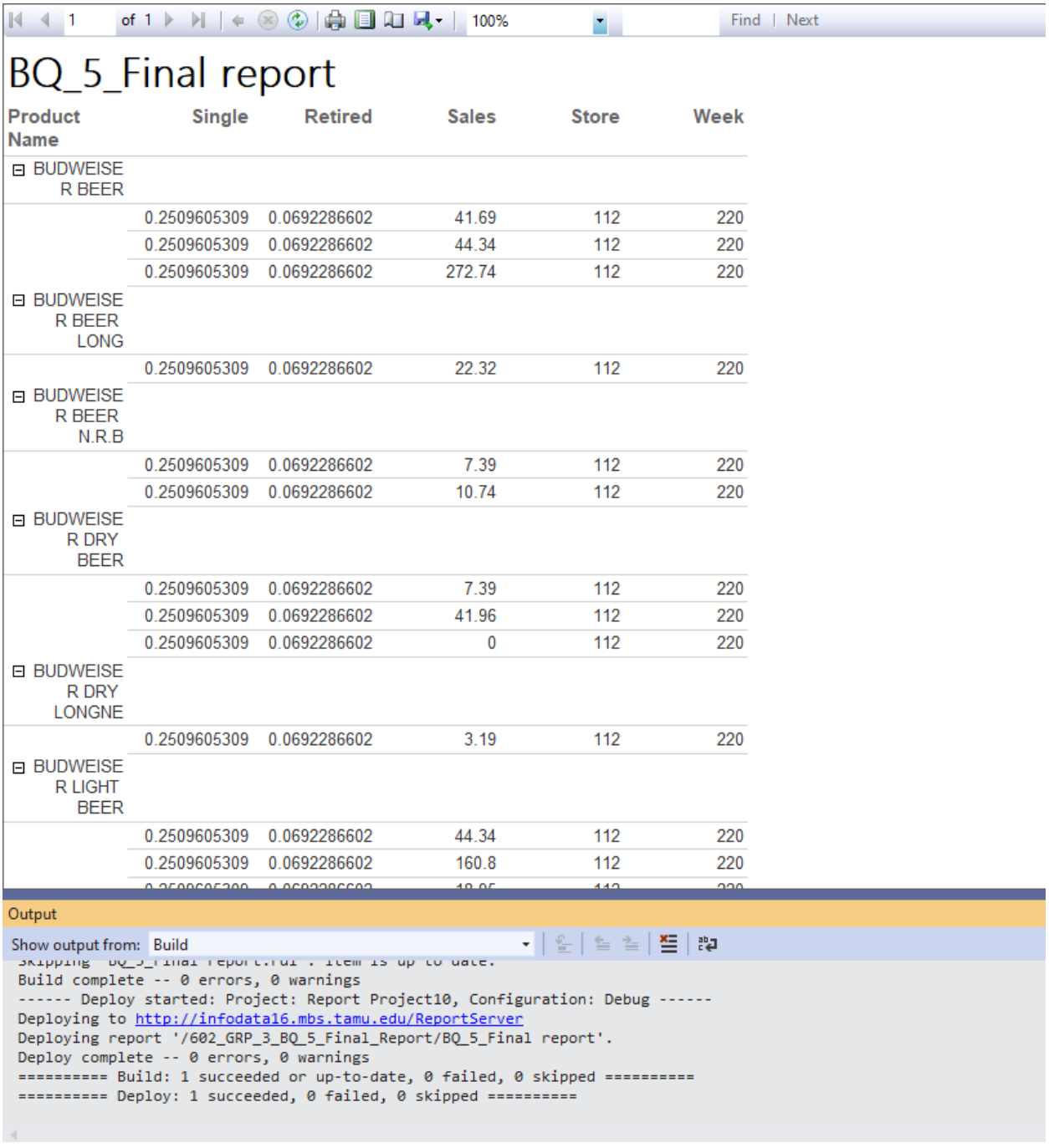


Fig Deployment Successful

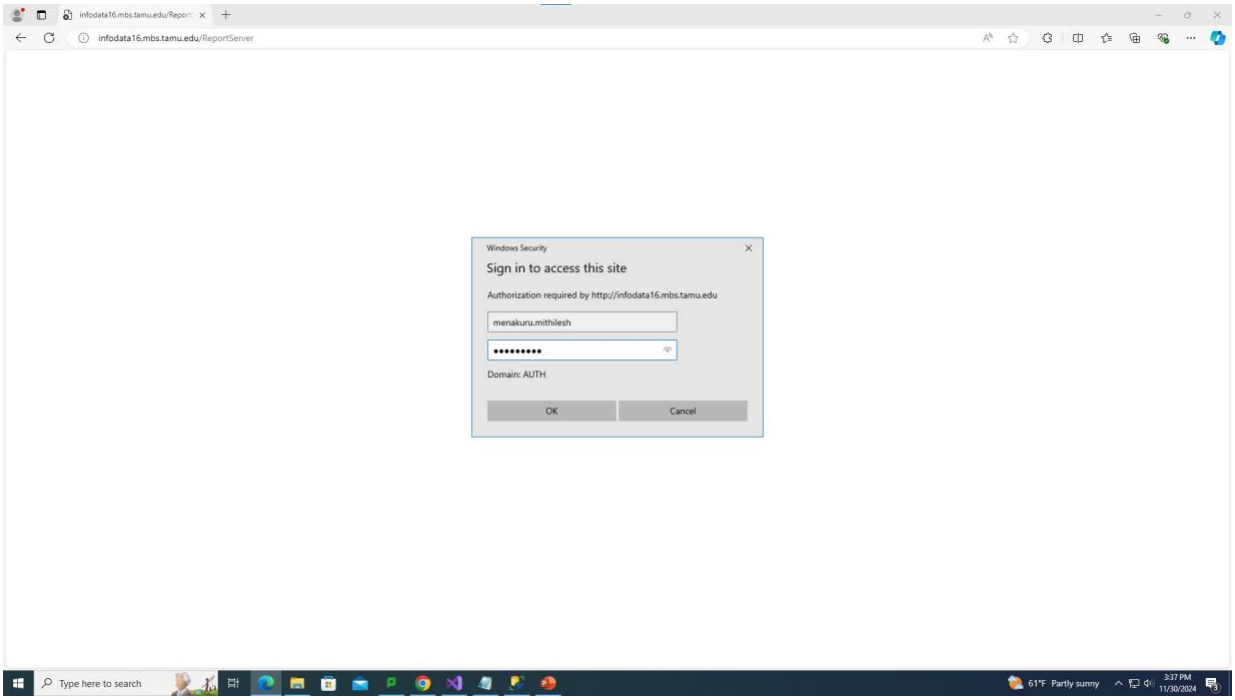


Fig Sign in with NetID

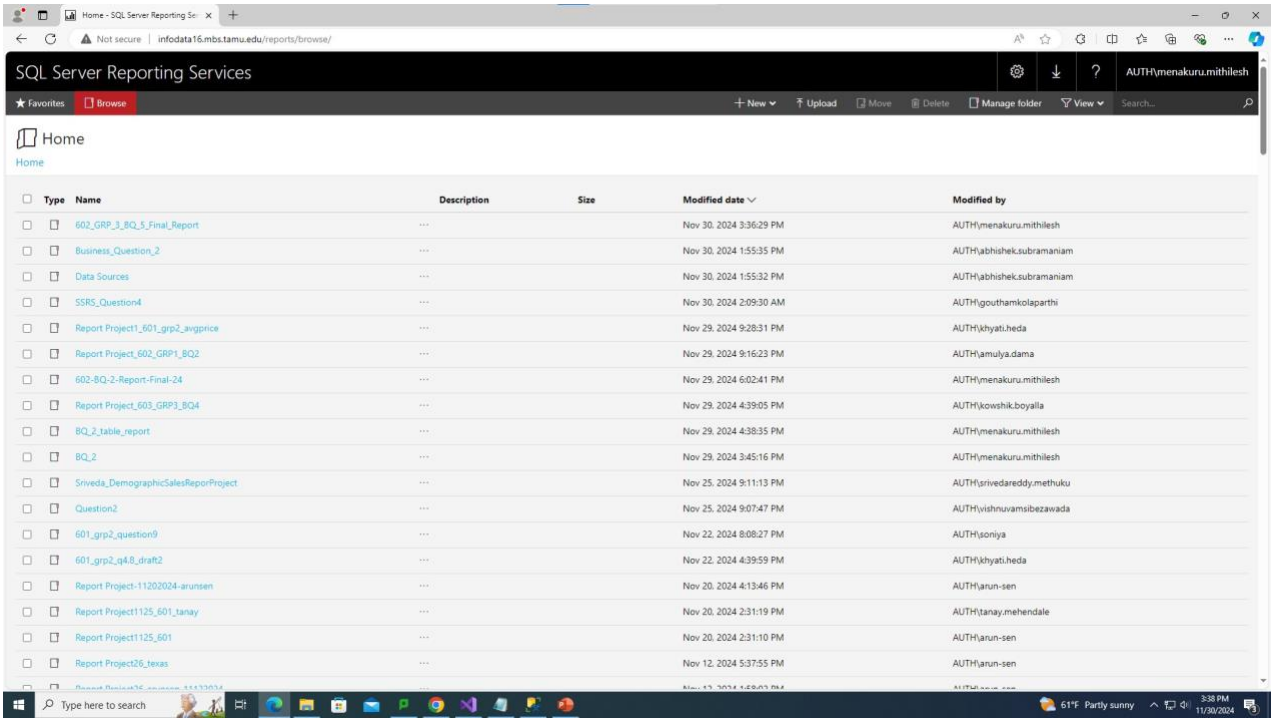
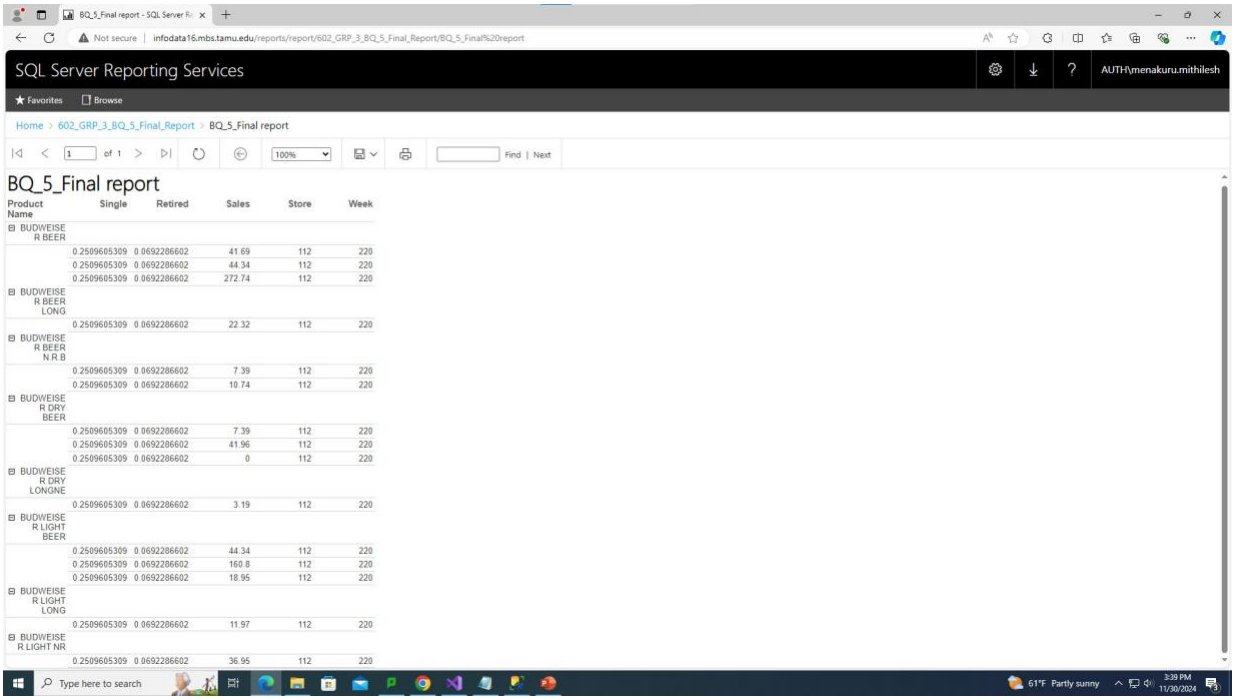


Fig Deployed to mbs server



Product Name	Single	Retired	Sales	Store	Week
⊟ BUDWEISER BEER					
	0.2509605309	0.0692286602	41.69	112	220
	0.2509605309	0.0692286602	44.34	112	220
	0.2509605309	0.0692286602	272.74	112	220
⊟ BUDWEISER BEER LONG					
	0.2509605309	0.0692286602	22.32	112	220
⊟ BUDWEISER BEER N.R.B					
	0.2509605309	0.0692286602	7.39	112	220
	0.2509605309	0.0692286602	10.74	112	220
⊟ BUDWEISER DRY BEER					
	0.2509605309	0.0692286602	7.39	112	220
	0.2509605309	0.0692286602	41.96	112	220
	0.2509605309	0.0692286602	0	112	220
⊟ BUDWEISER DRY LONGNE					
	0.2509605309	0.0692286602	3.19	112	220
⊟ BUDWEISER LIGHT BEER					
	0.2509605309	0.0692286602	44.34	112	220
	0.2509605309	0.0692286602	160.8	112	220
	0.2509605309	0.0692286602	18.95	112	220
⊟ BUDWEISER LIGHT LONG					
	0.2509605309	0.0692286602	11.97	112	220
⊟ BUDWEISER LIGHT NR					
	0.2509605309	0.0692286602	36.95	112	220

Fig Published report validated