5/1/2018

Consulting Report

ISTM 637 603, Group -12

Aditya Wagholikar, Kailun Xu, Prashant Shisodia

DOMINIC'S FINER FOOD

Credentials for accessing the data warehouse, reporting and analysis services
SQL Server Authentication
Server Name: infodata.tamu.edu

CONFIDENTIAL

The above credentials can be used to access:

1) Database Engine in the Microsoft SQL Server Management Studio

Staging Area: grp0004-602-ISTM-637-602-staging-area
Data Warehouse Area: grp0004-602-ISTM-637-602-dw-area

2) Analysis Service of the SQL Server

Cube deployed for SSAS

ISTM637_SoftDrinkAnalysis
ISTM637_602_Grp4_BQ5-Cigarette Analysis

3) http://infodata16.tamu.edu/ReportServer

Project Folder: 604_Group4_ProjectGroup

Table of Contents

1. Introduction 67

1.1 Project Background 4

1.2 Domain Knowledge 4

1.3 Data Details 4

1.4 Entity-Relationship Diagram 5

2. Business Questions and Justification 6

2.1 BQ1 and Justification 6

2.2 BQ2 and Justification 7

2.3 BQ3 and Justification 9

2.4 BQ4 and Justification 10

2.5 BQ5 and Justification 12

3. Data Warehouse Design 13

3.1 Dimension Matrix and Information Package 13

3.2 Dimension Tables 14

3.3 Fact Tables 15

3.4 Star Schema 16

4. Data Marts 18

4.1 Dimension Mapping Table 18

4.2 Category Sales Data Mart 20

4.2.1 Business Question Justification 20

4.2.2 Category Sales Mapping Table 20

4.3 Product Sales Data Mart 21

4.3.1 Business Question Justification 21

4.3.2 Product Sales Mapping Table 22

5. Data Integration 22

5.1 Data Quality Issues 22

5.2 ETL Plan 23

5.2.1 Target Data 23

5.2.2 Data Sources 24

5.2.3 Data Mappings from Sources to Staging to Data Warehouse 24

5.2.4 Data Extraction Rules 28

5.2.5 Data Transformation and Cleansing Rules 28

5.3 ETL Implementation 30

5.3.1 Time Dimension Table Creation 30

5.3.2 Store Dimension Table Creation 33
5.3.3 Product Dimension Table Creation 36
5.3.4 Category Dimension Table Creation 38
5.3.5 CategorySales Fact Table Creation 42
5.3.6 ProductSales Fact Table Creation 44
5.3.7 Category Fact Table Creation 48
5.3.7 Table Structure for Data Marts 51
5.2.3 Table Structure for Category Sales Data Mart 51
5.3 Snapshot of Data Warehouse Area 52
5.4 SQL Queries for Staging and Data Warehouse area 53
5.4.1 Staging 53
6. Business Intelligence (BI) Reports 61
6.1 Reporting Plan 61
6.1.1 Target Reports for Business Questions 61
6.1.2 Mapping from Tables in Data Marts to Report Attributes **Error! Bookmark not defined.**
6.2 SSRS for BQ-1 67
6.3 SSRS for BQ-2 76
6.4 SSAS+SSRS for BQ-3 83
6.4.1 SSAS Part 83
6.4.2 SSRS on top of SSAS 98
6.5 ReportBuilder3.0 for BQ-4 102
6.6 SSAS for BQ-5 109
7. Group Special Work 119
Appendix A 119

1. Introduction

1.1 Project Background

As a core component of business intelligence, a data warehouse functions as a system to achieve data analysis and reporting. It is a central repository that stores current and historical data from one or more disparate sources in a single place [1]. The system is used to solve business problems and support business decisions. Thus, to build the system in a better way, the first step is to understand the business domain and gather requirements to create business problems.

In this project, a data warehouse will be built for a retail store chain called Dominick's Fine Food (DFF). DFF is a grocery store chain and subsidiary of Safeway Inc. with locations mainly in the Chicago area. The data for this project includes store-level data covering shelf management and pricing from all the branches of DFF. For academic purpose, we will build a data warehouse to study the internal business model reflected by the data. And this consulting report is dedicated to finish the first and primary step of business domain understanding and business problem recognition.

1.2 Domain Knowledge

Four papers are synthesized to help us build our understanding of this domain. Paper 1 is "Holiday Price Rigidity and Cost of Price Adjustment" which grasps Thanksgiving-Christmas holiday period as a major sales period for US retailers. Paper 2 is "Cross-Brand Pass-Through in Supermarket Pricing" which investigates the sensitivity of cross brand pass-through estimate. Paper 3 is "Chain-wide and store-level analysis for cross-category management" which forward two important aspects of chain-wide and store-level. And paper 4 is "Why Do Some Prices In The Retail Sector Drop When Demand Rises" which reviews literature on price rigidity, its implications, the role of psychological pricing.

From the papers, we have learned:

- In sales domain, there exist differences between non-holiday and holiday sales. And sales pattern happened in holiday is even more worthwhile to investigate because of the high profit the vendors can receive.
- Price prediction of cross-brand category is worth to investigate and sale differences among different stores can be another study point.
- Variations in price are considered crucial for understanding price adjustments. How to depict the price tendency over time is of great value.

1.3 Data Details

The Dominick's database covers store-level scanner data collected at Dominick's Finer Foods over a period of more than seven years. Basically, the database contains two types of files: category-specific files and general files. The general files contain information pertaining to all the categories in the project.

Table 1 shows the detailed information about the files and metadata.

Table 1. File Details

File	Metadata	File Description
Customer Count	Store traffic and coupon usage by store	● In-store daily traffic, including number of customers visiting the store and purchasing ● Variables: *Date, Week, Store*
Store-specific Demographics	store-specific demographic data information	● Records in-store targeted promotion to customer groups ● Variables: *age, ethnicity, education, income, etc*
UPC	Record for each UPC in a category	● Records same-brand and cross-brand sale data ● Variables: *upc, com_code, nitem, etc*
Movement	Weekly sales for each UPC by store	● Records the margin profit for each upc in a category following a weekly basis ● Variables: *upc, store, week, move, price, qty, profit, sale* and *ok*

1.4 Entity-Relationship Diagram

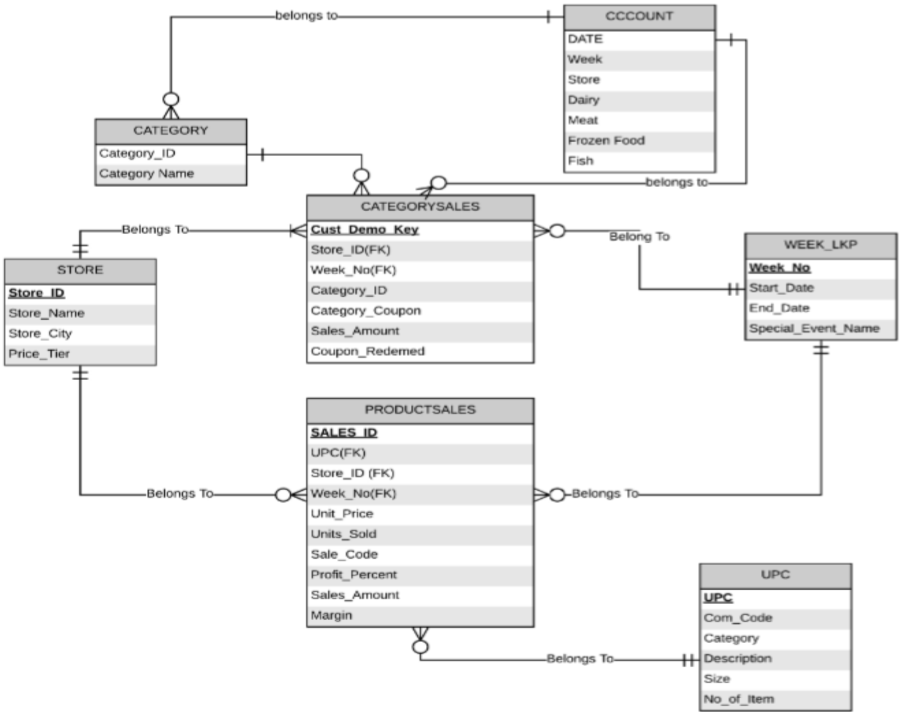


Figure 1 ERD for Dominick's Data Warehouse

2. Business Questions and Justification

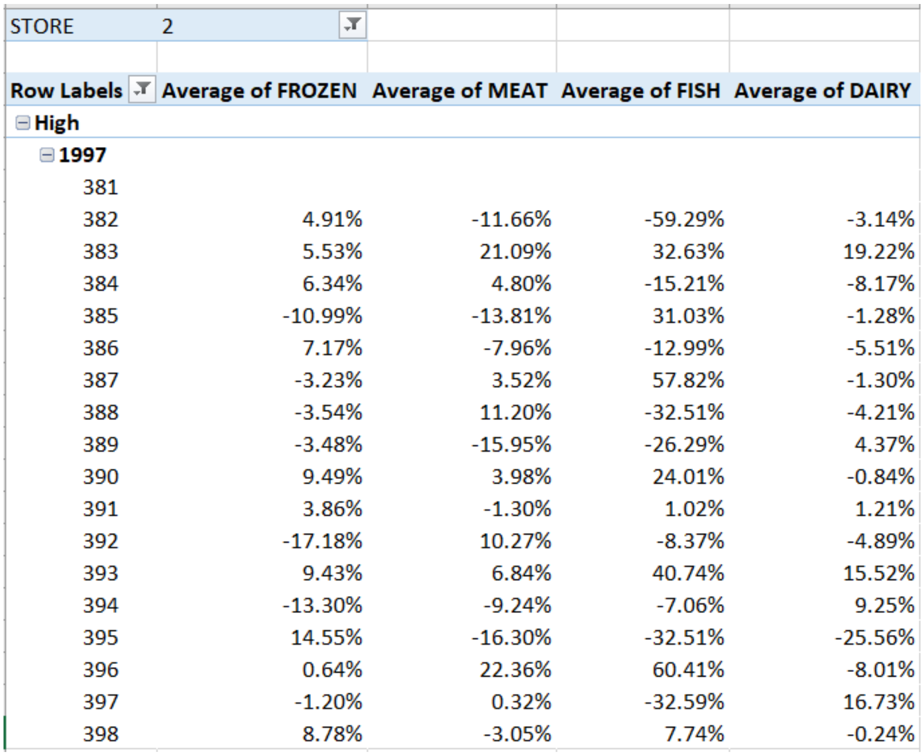
2.1 BQ1 and Justification

BQ 1 - What is the trend of weekly change in sales of Frozen Food, Meat, Fish and Dairy till current week in 1997 for store 2?

Justification

Frozen foods, meat, fish and dairy are edible food items. They are key items in customer grocery list. By understanding the sales trend of these items, the store manager can plan the inventory to ramp up or down their quantity. Additionally, these graphs below show the micro and macro seasonality of various categories of food products which requires refrigeration. It displays the % change in average price of basket of products under a category. Knowledge of this change can help store managers determine price reduction/change strategy at macro level on a weekly basis and insulate the store from erratic costs of price reduction. Our observation through the subset of data leads us to a preliminary conclusion fish sale are quite bipolar on weekly basis. There is a tremendous amount of fluctuation points to the nature of fish consumption. People buy fish for consumption on fortnightly basis. Extra procurement of any anticipated spike in consumption is not the best idea. The similar pattern of crests and dips in other frozen items point towards an integrated strategy of cross selling various meat and dairy products. Their peaks and dips are in lockstep. A combo package-based discounting offer on combination of above products could lead to higher and predictable volumes. Consequently, the customer side push in volumes could be leveraged to get supplier side rebates and discounts to improve margins.

Data Snapshot



STORE	2			
Row Labels	Average of FROZEN	Average of MEAT	Average of FISH	Average of DAIRY
⊟ High				
⊟ 1997				
381				
382	4.91%	-11.66%	-59.29%	-3.14%
383	5.53%	21.09%	32.63%	19.22%
384	6.34%	4.80%	-15.21%	-8.17%
385	-10.99%	-13.81%	31.03%	-1.28%
386	7.17%	-7.96%	-12.99%	-5.51%
387	-3.23%	3.52%	57.82%	-1.30%
388	-3.54%	11.20%	-32.51%	-4.21%
389	-3.48%	-15.95%	-26.29%	4.37%
390	9.49%	3.98%	24.01%	-0.84%
391	3.86%	-1.30%	1.02%	1.21%
392	-17.18%	10.27%	-8.37%	-4.89%
393	9.43%	6.84%	40.74%	15.52%
394	-13.30%	-9.24%	-7.06%	9.25%
395	14.55%	-16.30%	-32.51%	-25.56%
396	0.64%	22.36%	60.41%	-8.01%
397	-1.20%	0.32%	-32.59%	16.73%
398	8.78%	-3.05%	7.74%	-0.24%

Figure 2 Data for BQ-1

Pivot Table

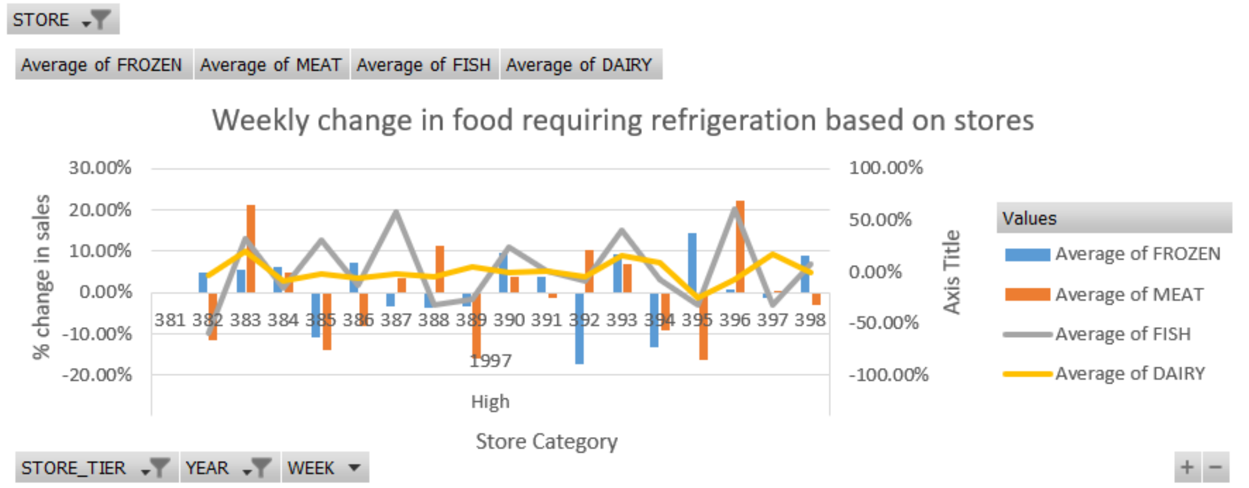


Figure 3 Pivot Table for BQ-1

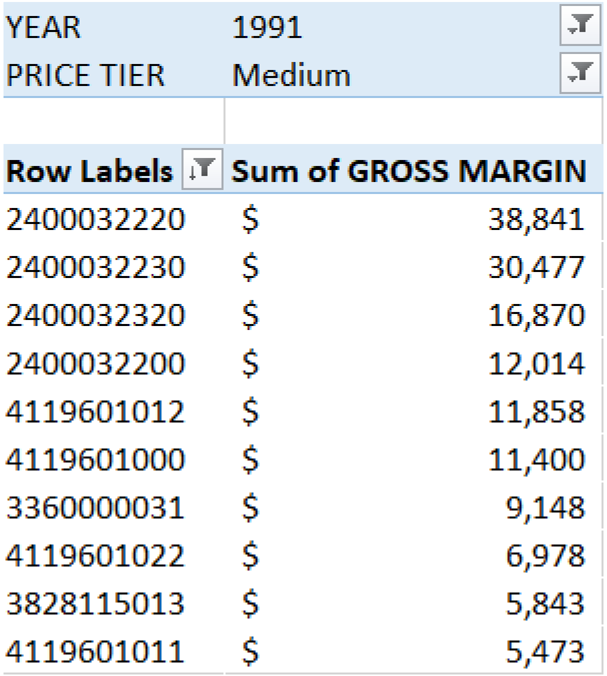
2.2 BQ2 and Justification

BQ 2 - Which are the top 10 products with respect to profitability in Medium Tier stores w.r.t "Canned Soup" category of in year 1991?

Justification

Medium Tier stores have a demographic consisting of high number of college graduates and working population with medium to disposable income. Canned soup is one of the popular products in this demographic. By focusing on popular products, sales of this category can be augmented. Ready to eat canned soups can be bundled in Buy One Get One offers or Combo offers of various varieties to drive sales in this category. If this category is consistently on a growth curve, other products in ready to eat category could be bundled and to increase their volumes. Certain new products in non-competing category could be introduced on a pilot basis by bundling with these popular (top 10) products. Dominick's could also sell certain related in-house products with these products on introductory basis. These products should be placed in shelves which are easily visible to customer to further increase the sales.

Data Snapshot



YEAR	1991
PRICE TIER	Medium

Row Labels	Sum of GROSS MARGIN
2400032220	$ 38,841
2400032230	$ 30,477
2400032320	$ 16,870
2400032200	$ 12,014
4119601012	$ 11,858
4119601000	$ 11,400
3360000031	$ 9,148
4119601022	$ 6,978
3828115013	$ 5,843
4119601011	$ 5,473

Figure 4 Data for BQ-2

Pivot Table

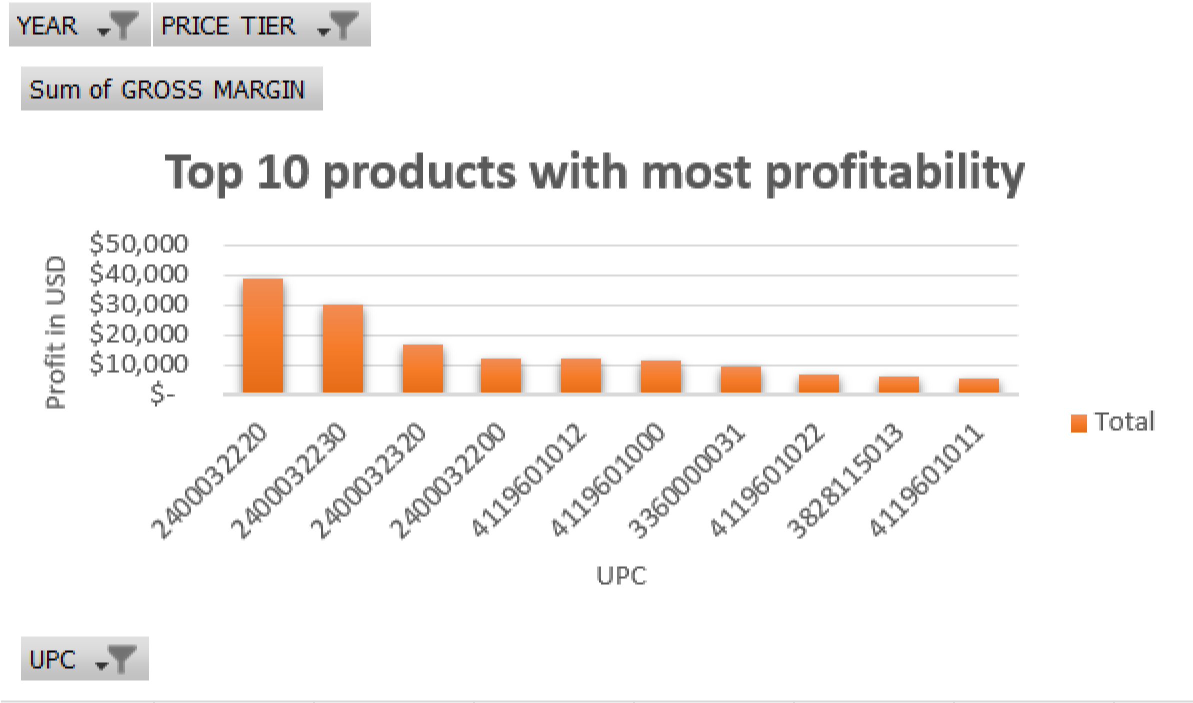


Figure 5 Pivot Table for BQ-2

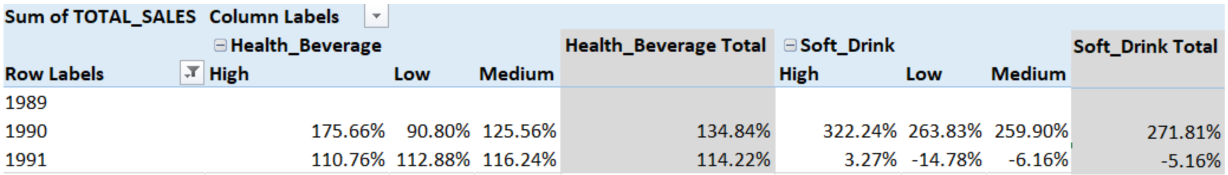
2.3 BQ3 and Justification

BQ 3 - What is the growth trend of Soft Drinks versus Bottled Juice for year 1990 and 1991? Which of these has negative growth rate? What is the trend for bottled juice especially in Low Tier?

Justification

Globally, the soda based soft drink consumption is on decline in absolute terms due to healthier alternatives. As a retailer, Dominick's has some key challenges to manage the dip in soda sales. As the data shows, bottled juice is picking up sales especially in stores in low tier. By giving better visibility to bottled juices along with attractive pricing, bottle juice consumption could be driven up. On the other hand, to arrest decrease in soft drink sales, focused efforts to promote their sales by using various discounting techniques are suggested by our team. Cross selling of soft drinks with other frozen foods like frozen pizza, snack crackers and ready to eat food can help to improve sales.

Data Snapshot



Sum of TOTAL_SALES	Column Labels							
	⊟Health_Beverage			Health_Beverage Total	⊟Soft_Drink			Soft_Drink Total
Row Labels	High	Low	Medium		High	Low	Medium	
1989								
1990	175.66%	90.80%	125.56%	134.84%	322.24%	263.83%	259.90%	271.81%
1991	110.76%	112.88%	116.24%	114.22%	3.27%	-14.78%	-6.16%	-5.16%

Figure 6 Data for BQ-3

Pivot Table

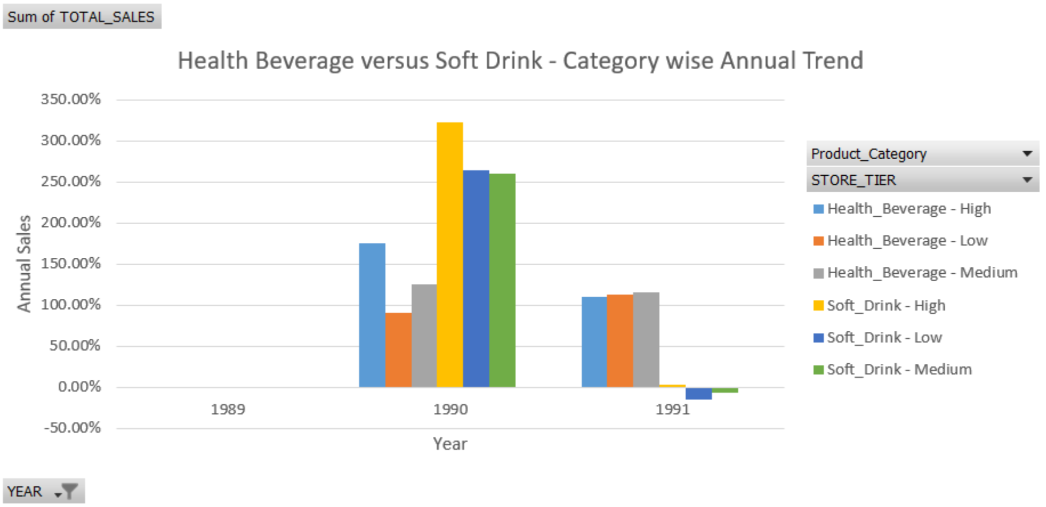


Figure 7 Pivot Table for BQ-3

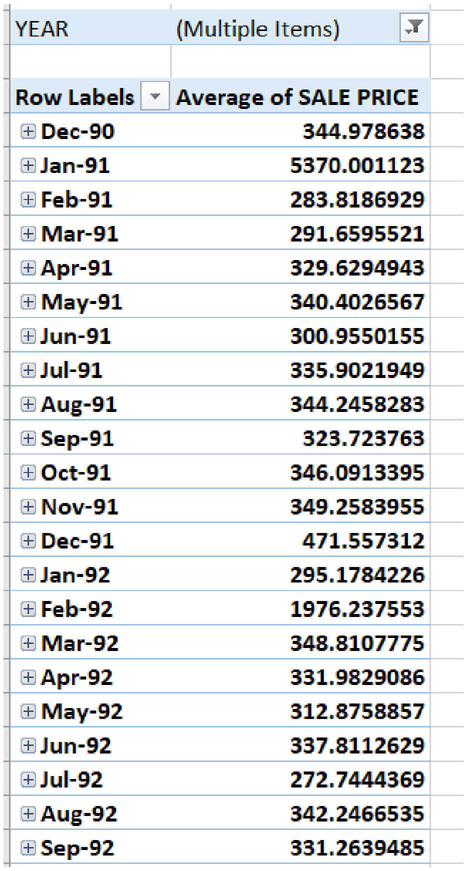
2.4 BQ4 and Justification

BQ 4 - What effect does the time of the year have on sales of each Tuna? Determine the impact of seasonality on Average Sales of Tuna from December-1990 to September 1992?

Justification

Tuna is a popular fish consumed in North America. Although some of it is native is grown in America, most of it is imported from outside USA. Tuna is perishable commodity with less shelf life. Tuna is a costly fish in general. Being perishable not in frozen foods category, it is important for Dominick's that they are able to sell maximum Tuna fish from their existing stock and replenish it with newer one. After analyzing data, it is evident that Tuna sales peaks up during January or February and remains stable for rest of the year. There are number of external factors contributing to this consumer behavior - festive season, high cost due to less supply, etc. By using below trend, proper planning for Tuna procurement and sales could be done. It could be also assisted by proper discounting strategy.

Data Snapshot



YEAR	(Multiple Items)
Row Labels	**Average of SALE PRICE**
⊞ Dec-90	344.978638
⊞ Jan-91	5370.001123
⊞ Feb-91	283.8186929
⊞ Mar-91	291.6595521
⊞ Apr-91	329.6294943
⊞ May-91	340.4026567
⊞ Jun-91	300.9550155
⊞ Jul-91	335.9021949
⊞ Aug-91	344.2458283
⊞ Sep-91	323.723763
⊞ Oct-91	346.0913395
⊞ Nov-91	349.2583955
⊞ Dec-91	471.557312
⊞ Jan-92	295.1784226
⊞ Feb-92	1976.237553
⊞ Mar-92	348.8107775
⊞ Apr-92	331.9829086
⊞ May-92	312.8758857
⊞ Jun-92	337.8112629
⊞ Jul-92	272.7444369
⊞ Aug-92	342.2466535
⊞ Sep-92	331.2639485

Figure 8 Data for BQ-4

Pivot Table

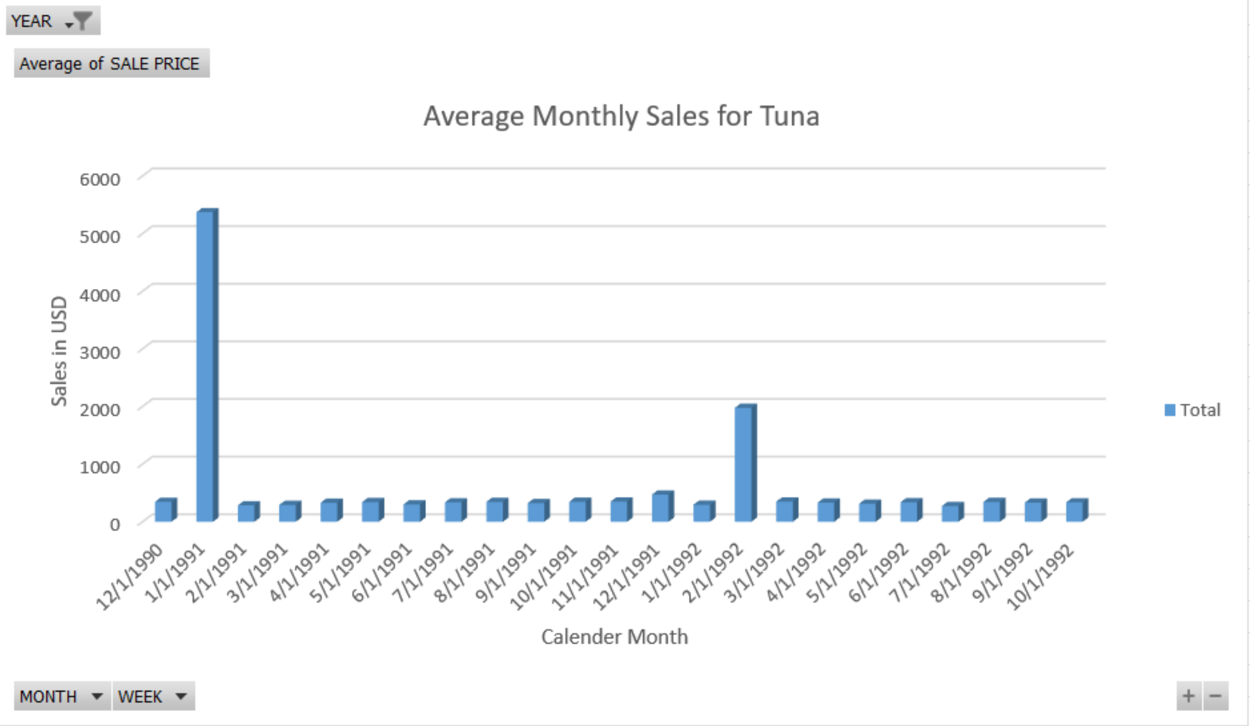


Figure 9 Pivot Table for BQ-4

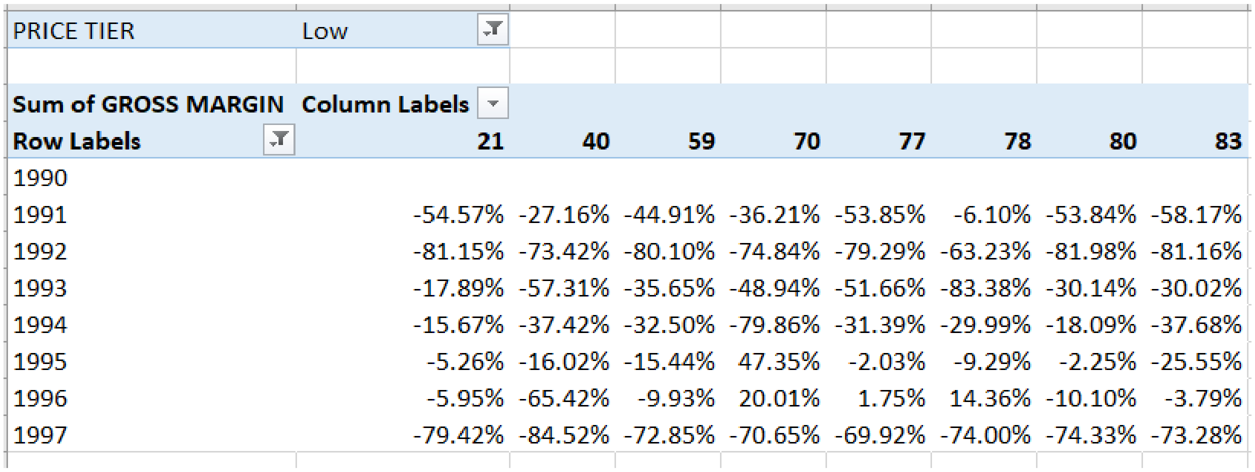
2.5 BQ5 and Justification

BQ 5 - What is annual growth rate of cigarette margins in low tier stores? What is the decreasing trend?

Justification

Globally, there is reduction in Tobacco consumption. Be it cigarettes or any other medium. However, there is still some traction in areas with higher proportion of college graduates. By knowing the trend in stores having demographics in favor of college students, cigarette sales strategy can be formulated. As per the data analyzed by our team, it is observed that cigarette sales are decreasing even in these localities. They are exponentially decreasing in all the stores in given locality. Dominick's needs to use discounting or price reduction strategy to arrest this decline and change its directions. Generally, college graduates are cost sensitive. So simple price reduction is the strategy to go forward. Secondly, if there is concrete evidence of dip is sales, Dominick's could reassess its space utilization to see if some less selling products could be replaced by popular products in the category.

Data Snapshot



PRICE TIER	Low								
Sum of GROSS MARGIN	**Column Labels**								
Row Labels		**21**	**40**	**59**	**70**	**77**	**78**	**80**	**83**
1990									
1991		-54.57%	-27.16%	-44.91%	-36.21%	-53.85%	-6.10%	-53.84%	-58.17%
1992		-81.15%	-73.42%	-80.10%	-74.84%	-79.29%	-63.23%	-81.98%	-81.16%
1993		-17.89%	-57.31%	-35.65%	-48.94%	-51.66%	-83.38%	-30.14%	-30.02%
1994		-15.67%	-37.42%	-32.50%	-79.86%	-31.39%	-29.99%	-18.09%	-37.68%
1995		-5.26%	-16.02%	-15.44%	47.35%	-2.03%	-9.29%	-2.25%	-25.55%
1996		-5.95%	-65.42%	-9.93%	20.01%	1.75%	14.36%	-10.10%	-3.79%
1997		-79.42%	-84.52%	-72.85%	-70.65%	-69.92%	-74.00%	-74.33%	-73.28%

Figure 10 Data for BQ-5

Pivot Table

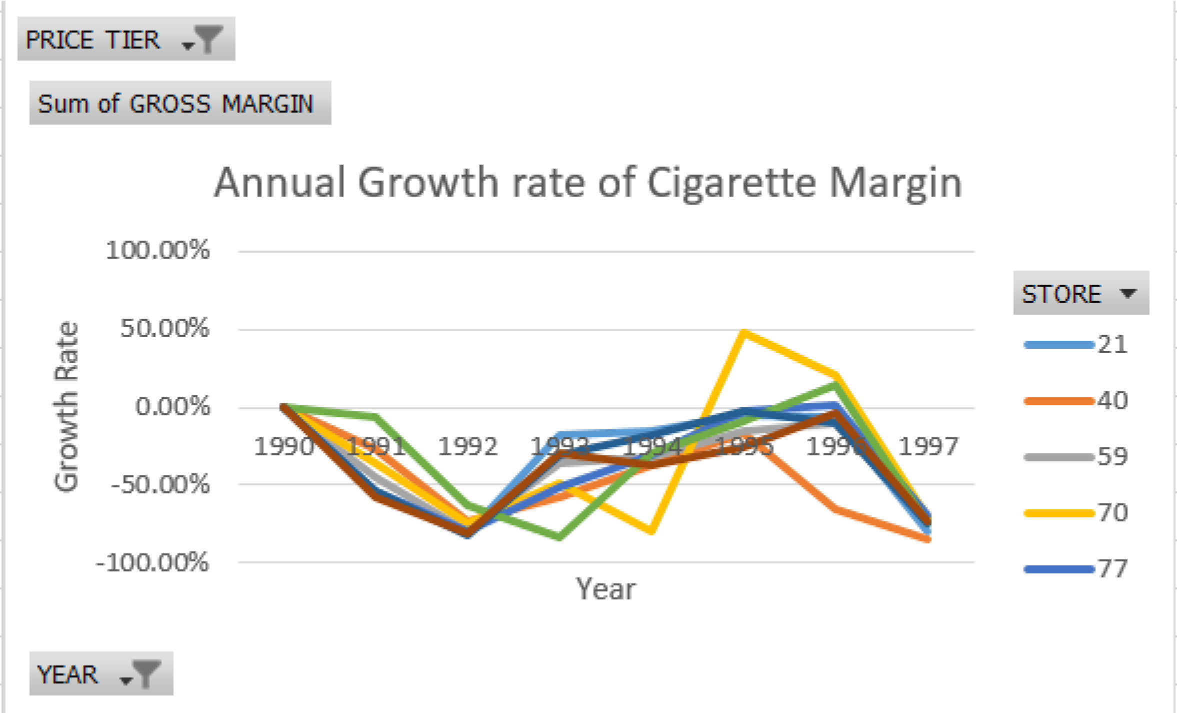


Figure 11 Pivot Table for BQ-5

3. Data Warehouse Design

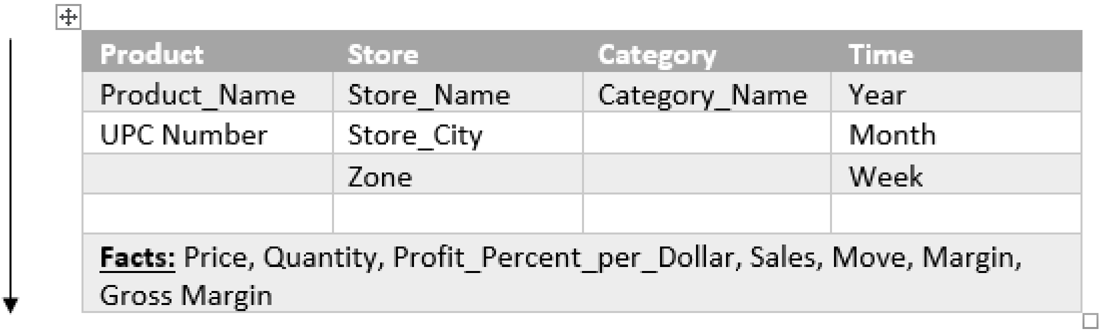
3.1 Dimension Matrix and Information Package

Five business questions can be summarized as weekly food sales, top 10 canned soup, soft drinks verses bottle juices, seasonally tuna sale and annual cigarette growth in low tire store. Two kinds of data marts are designed: Product Sales and Category Sales.

Table 2. Dimension Matrix

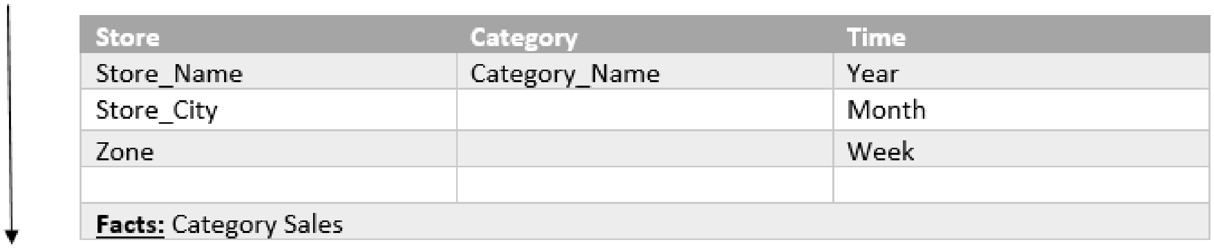
				Dimensions	
BQs	**Data Marts**	Product	Store	Category	Time
	Product Sales	X	X	X	X
	Category Sales		X	X	X

Based on the Dimension Matrix of ***Kimball Approach*** of data warehouse design, below are the ***information package*** diagrams which specify the fact tables design by giving details of level of hierarchy, attributes, and metrics of measurement. We must admit that while making data mart, the information package technique helped us to create cohesive data marts – Product Sales and Category Sales.



Product	Store	Category	Time
Product_Name	Store_Name	Category_Name	Year
UPC Number	Store_City		Month
	Zone		Week
Facts: Price, Quantity, Profit_Percent_per_Dollar, Sales, Move, Margin, Gross Margin			

Figure 12 Product Sales Information Package



Store	Category	Time
Store_Name	Category_Name	Year
Store_City		Month
Zone		Week
Facts: Category Sales		

Figure 13 Category Sales Information Package

3.2 Dimension Tables

- Product Dimension

Product dimension table contains information about the product.

Table 3. Product Dimension

Attribute Name	**Description**

Product_ID	Surrogate key (unique identify) of Product Dimension
Product_Name	Product name
UPC_Number	Upc number
Source_System_Code	Indicates the source system
Active	Active flag

- Store Dimension

Store Dimension Table contains information of the concrete store in the DFF retail store chain

Table 4. Store Dimension

Attribute Name	**Description**
Store_ID	Surrogate key (unique identify) of Store Dimension
Store_Number	Store number
Store_Name	Name of the store
Store_City	City where the store is located
Price_Tier	Includes high and low tier
Zone	Area of the store
Source_System_Code	Indicates the source system
Active	Active flag

- Category Dimension

Category dimension table contains information about the category classification in the DFF store chain.

Table 5. Category Dimension

Attribute Name	**Description**
Category_ID	Surrogate key (unique identify) of Category Dimension
Category_Name	Category number
Source_System_Code	Indicates the source system
Active	Active flag

- Time Dimension

Time dimension table contains information about time attribute.

Table 6. Time Dimension

Attribute Name	**Description**
Time_ID	Surrogate key (unique identify) of Time Dimension
Year	Year of date
Month	Month of date
Week_Number	Number of week
Special_Event	Holiday or other special promotion date
Source_System_Code	Indicates the source system

3.3 Fact Tables

- Product Sales Fact Table

Table 7. Product Sales Fact Attribute

Attribute Type	Attribute Name	Description
Keys	Product_Sales_ID	Unique identifier for each product sale. Primary Key
	Product_ID	Unique identifier which help divide data from product dimension. Foreign key
	Store_ID	Unique identifier for store which help divide data from store dimension. Foreign key
	Category_ID	Unique identifier for category which help divide data by category dimension. Foreign key
	Time_ID	Unique identifier for time which helps divide data by time dimension. Foreign key
Measures	Price	Unit price of the product sales
	Quantity	Quantity of each product sold
	Profit_Percent_per_Dollar	Profit made on each dollar sold. It is percentage
	Sale_Promotion_Code	Promotion information
	Sales	Total sale value
	Move	The number of actual items sold in a bundle
	Margin	The difference between cost price and sell price
	Gross-Margin	The absolute margins

- Category Sales Fact Table

Table 8. Category Sales Fact Attribute

Attribute Type	Attribute Name	Description
Keys	Category_Sales_ID	Unique identifier for each product sale. Primary Key
	Store_ID	Unique identifier for store which help divide data from store dimension. Foreign key
	Category_ID	Unique identifier for category which help divide data by category dimension. Foreign key
	Time_ID	Unique identifier for time which helps divide data by time dimension. Foreign key
Measures	Category Sales	the sale value of the category sales across multiple dimensions

3.4 Star Schema

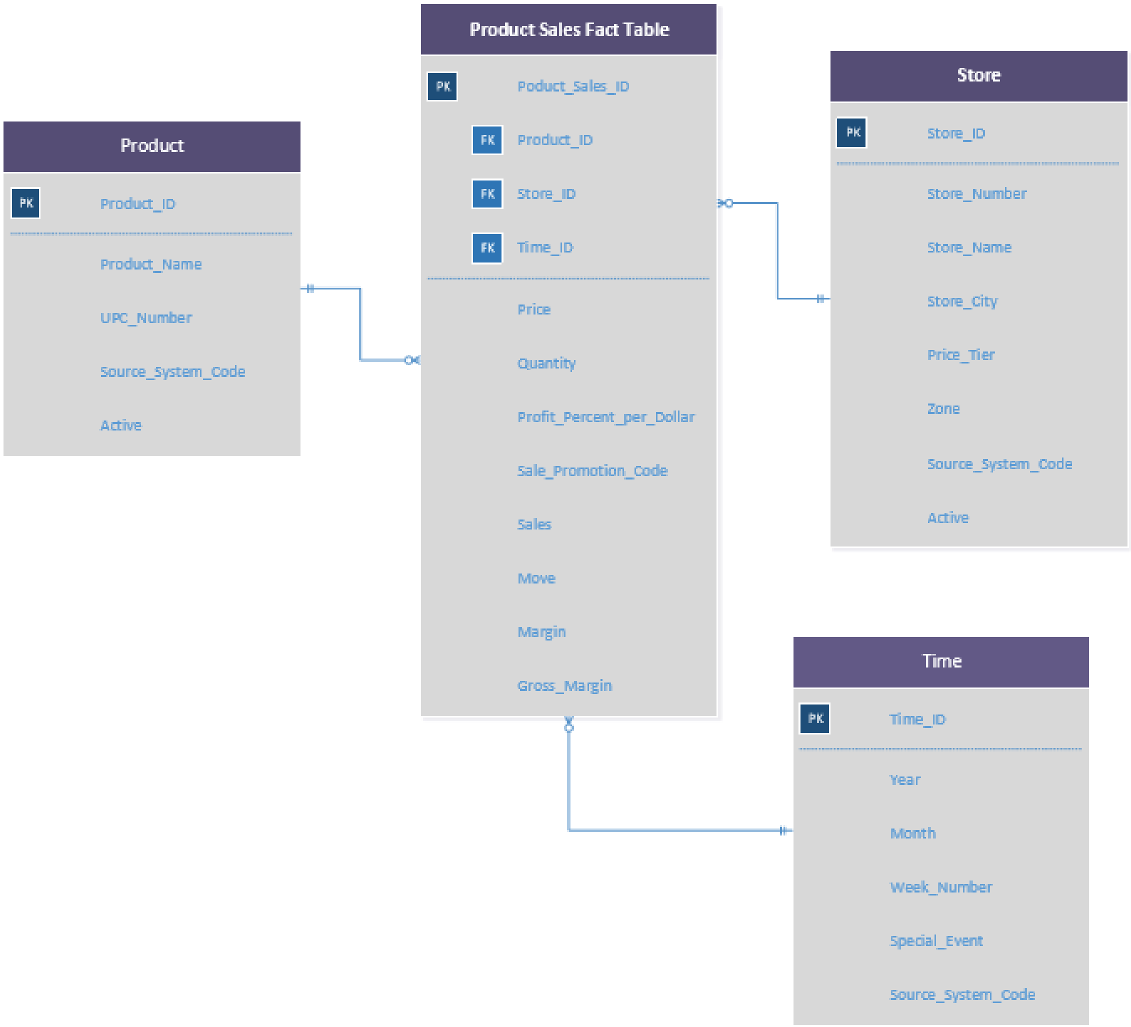


Figure 14 Star Schema for Product Sales data mart

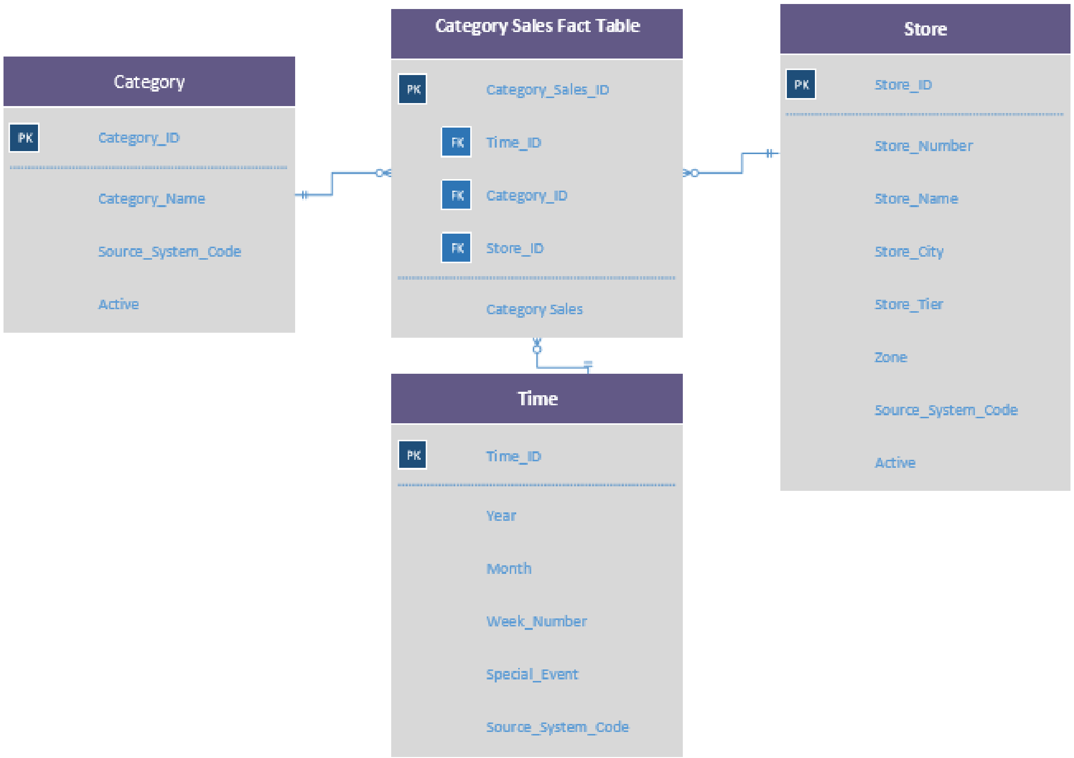


Figure 15 - Star Schema for Category Sales data mart

4. Data Marts

4.1 Dimension Mapping Table

Table 9. Mapping Table for Store Dimension

DW Dimension Table	**DW Dimension Attribute**	**Source Table**	**Source Table Attribute**	**Mapping Functions**
		Dominick's Store and Store Specific Demographics		
Store	Store_ID (Row Number)			Auto-generated
	Store_Number		Store	
	Store_Name		Name	
	Store_City		City	
	Store_Tier		Price_Tier	
	Zone		Zone	

	Source_System_Code			Taken from System code generator function. Value is “DEMO”
	Active			Flag for slowly changing dimensions. Open record has value = 1 Closed record has value = 0

Table 10. Mapping Table for Category Dimension

DW Dimension Table	DW Dimension Table Attribute	Source Table	Source Table Attribute	Mapping Functions
Category		CCount		
	Category_ID (Row number)			Auto-generated
	Category_Name			Various category attributes like Fish, grocery, beer, etc from CCount.
	Source_System_Code			Taken from System code generator function during ETL specifying the details of source file.
	Active			Flag for slowly changing dimensions. Open record has value = 1 Closed record has value = 0

Table 11. Mapping Table for Product Dimension

DW Dimension Table	DW Dimension Attribute	Source Table	Source Table Attribute	Mapping Functions
Product		UPC Table		
	Product_ID (Row Number)			Auto-generated
	Product_Name		Description	
	UPC_Number		UPC	
	System_Source_Code			Specifies specific UPC source file.
	Active			Flag for slowly changing dimensions. Open record has value = 1 Closed record has value = 0

Table 12. Mapping Table for Time Dimension

DW Dimension Table	DW Dimension Attribute	Source Table	Source Table Attribute	Mapping Functions
Time		Week Decode Table		
	Time_ID (Row Number)			Auto-generated
	Year		Start End	To obtain Year, we can use start (date) and end (date) from Week Decode Table.
	Month		Start End	To obtain Year, we can use start (date) and end (date) from Week Decode Table.
	Week_Number		Week #	
	Special_Event		Special Events	
	Source_System_Code			Taken from System code generator function

4.2 Category Sales Data Mart

4.2.1 Business Question Justification

Category Sale Data Mart helps to resolve BQ-1.

BQ-1: What is the trend of weekly change in sales of Frozen Food, Meat, Fish and Dairy till current week in 1997 for store 2?

Justification – This business question is answered by *Category Sales* data mart. The Store Number attribute is used from Store dimension table to divide the data by Store. The week number and year attribute are used from Time dimension table to further divide data. The Category name attribute is used from Category dimension table to divide data in frozen meat, fish and dairy food category. The measure *Category_sales* from the *Category Sales* fact table is used to find the trend of weekly change in sales of Frozen Food, Meat, Fish and Dairy till current week in 1997 for store 2.

4.2.2 Category Sales Mapping Table

Table 13. Mapping Table for Category Sales

DW Fact Table	DW Fact Table Attribute	Source Table	Source Table Attribute	Mapping Functions
		CCount		
	CategorySales_ID			ETL-generated surrogate key
	Category_ID			Primary key of the Category dimension

Category Sales	Time_ID	Primary key of the Time dimension
	Store_ID	Primary key of the Store dimension
	Category Sales	Sales of categories like Fish, grocery, beer, etc from ccount.

4.3 Product Sales Data Mart

4.3.1 Business Question Justification

Product Sale Data Mart helps to resolve BQ-2 – BQ-5.

BQ-2: Which are the top 10 products with respect to profitability in Medium Tier stores w.r.t "Canned Soup" category of in year 1991?
Justification - This business question is answered by *ProductSales* data mart. We have used store_tier attribute from Store dimension table and year from Time dimension table as filters. Product details like UPC and Name is taken from Product dimension table. The measure Gross_Total is taken from Product Sales fact table and summed for given products and ranked. We display the top 10 products by applying rank function.

BQ-3: What is the growth trend of Soft Drinks versus Bottled Juice for year 1990 and 1991? Which of these has negative growth rate? What is the trend for bottled juice especially in Low Tier?
Justification - This business question is answered by *Product Sales* data mart. We have used store_tier attribute from Store dimension table, year from Time dimension table. Year filter acts as a filter criterion. The measure Gross_Total is taken from Product Sales fact table and summed for given years. We display the total sales growth with respect to previous year for product type combination.

BQ-4: What effect does the time of the year have on sales of each Tuna? Determine the impact of seasonality on Average Sales of Tuna from December-1990 to September 1992?
Justification - This business question is answered by *ProductSales* data mart. We have used year attribute from *Time* dimension table to filter on the basis of year. Month and Week attributes from Time dimension table are used to measure time dimension against which we plot average of Sales. Sales attribute comes from Product Sales Fact table.

BQ-5: What is annual growth rate of cigarette margins in low tier stores? What is the decreasing trend?
Justification - This business question is answered by *Product Sales* data mart. We have used *store_tier* attribute from *Store* dimension table to filter. *Year* attributes from Time dimension table are used to measure time dimension against which we plot sum of *Gross_Margin*.

Gross_Margin comes from the data mart - *Product_Sales.* Store_Num attribute comes from Store dimension table. We show a store wise trend plotted against year.

4.3.2 Product Sales Mapping Table

Table 14. Mapping Table for Product Sales

DW Fact Table	DW Fact Table Attribute	Source Table	Source Table Attribute	Mapping Functions
Product Sales		Movement Table		
	Product_Sales_ID			ETL-generated surrogate key
	Product_ID			Primary key of the Product dimension
	Store_ID			Primary key of the Store dimension
	Time_ID			Primary key of the Time dimension
	Price		Price	
	Quantity		qty	
	Profit_Percent_per_Dollar		profit	
	Sale_Promotion_Code		sale	
	Sales			Derived from the formula Sales = (price*move)/qty
	Move		move	
	Margin			(Sales*Profit_Percent_Per_Dollar)/100
	Gross_Margin			Abs(Margin)

5. Data Integration

5.1 Data Quality Issues

To solve the business problem, the quality of input data matters. However, the provided source data is not perfect. There exist dummy values, absence values, inconsistent values, and incorrect values. And these will cause problem in subsequent analysis. Hence, data quality issues in DFF data sets are summarized first before moving to ETL plan.

Table 15 - Data Quality Issues

Issue Type	Issue Name	Issue Description
Data Relationship	Referential Integrity	A lot of null values in all the excel sheet, even in keys which could be assumed to be primary key. E.g. Store number was repeated and had inconsistent values
	Cardinality	Structural inconsistency between entities and attributes
Fields Structure	Format	Values did not follow consistent formatting standards
	Standard	Confusion about what data represents. E.g. Coupons had decimal values, so shall a user assume it is coupon's dollar value or it is dirty data
	Consistent	Amongst files there was an overlap with the data values
Data Value	Completenes s	Attributes had no value. E.g. "Sale" field was blank for most records
	Accuracy	There exist null values where value was expected. E.g. negative values for Sales data and decimal value for coupon
	Validity	Data values did not fall within acceptable ranges defined by the business. Store number have out of range values

The above table discusses the Data Quality of Dominic Finer Foods. For working on the ETL implementation, we need to have a good understanding of data. Dummy values, Absence of data values, unofficial use of fields, Cryptic values, Violation of business rules, Non-unique identifiers, Inconsistent values, and incorrect values are the problems faced by users during data analysis. Hence, problems are required to be classified and then appropriate steps in ETL will take care that data generated for business intelligence purpose is accurate.

5.2 ETL Plan

ETL plan refers to detailed plan for better implementing extract, transform and load. Extract is the process of reading data from a database; transform is the process of converting the extracted data from its previous form into required form in appropriate storage system; load is the process of writing data into target database.

Generally, ETL tools are used to pull data from one or more source systems, transform and load in another database. In ETL plan, we will determine the whole processes using tools.

5.2.1 Target Data

Table 16. Target Data

Data	Data Extraction	Data Transformation	Data Destination
UPC	UPCCSO, UPCSDR, UPCBJC, UPCTNA, UPCCIG	UPCCSO, UPCSDR, UPCBJC, UPCTNA, UPCCIG	PRODUCT_DIM
MOVEMENT	Done-WCSO, Done-SDR, Done-WBJC, Done-WTNA, Done-WCIG		PRODUCT_SALES_FACT
STORE	Store	Store	STORE_DIM
WEEK	Week_Decode	FinalWeekDecode	TIME_DIM
CCOUNT	CCOUNT	cleanCOUNT	CATEGORY_DIM, CATEGORYSALES_FACT

5.2.2 Data Sources

The data source for the data warehouse is the Dominick's database provided by the University of Chicago Booth School of Business.

Table 17. Data Sources

Data	Souce Files
UPC	UPC<product_acronym>.csv
MOVEMENT	W<product_acronym>.csv
STORE	STORE(created)
WEEK	WEEK_DECODE(created)
CUSTOMER COUNT	CCOUNT.csv

5.2.3 Data Mappings from Sources to Staging to Data Warehouse

Table 18. Data Mapping from Sources to Staging

SOURCE FILE NAME	SOURCE FILE ATTRIBUTES	STAGING AREA NAME	STAGING AREA TABLE ATTRIBUTES
Week_Decode.csv	week number start date end date special events	Week_decode	[Week] [Start] [End] [Special Events]
Store.csv	store number city price tier zone zip code address	Store	[Store] [City] [Price Tier] [Zone] [Zip Code] [Address]
CCout.csv	store number Frozen Food Date Meat Fish Dairy	CCOUNT_Clean_STG	[STORE] [FROZEN] [Date] [MEAT] [FISH] [DAIRY]

	Week		[Week]
upc<*product_acronym*>.csv		UPC<*product_acronym*> 5 files – UPCCSO UPCBJC UPCSDR UPCTNA UPCCIG	
	com_code		com_code
	upc		upc
	descrip		descrip
	nitem		nitem
	size		size
	case		case
w<product_acronym>.csv		We load 5 files in staging Movement<product_acronym> Movement_WCSO Movement_WSDR Movement_WBJC Movement_WTNA Movement_WCIG	
	STORE NUMBER		STORE
	UPC		UPC
	WEEK		WEEK
	MOVE		MOVE
	QTY		QTY
	PRICE		PRICE
	SALE		SALE
	PROFIT		PROFIT_PER_DOLLAR
	OK		OK

Table 19. Data Mapping from Staging to Data Warehouse

STAGING AREA TABLE NAME	**STAGING AREA ATTRIBUTE**	**DATA WAREHOUSE TABLE**	**DATA WAREHOUSE TABLE ATTRIBUTES**	**MAPPING FUNCTION**
Week_decode		TIME_DIM	TIME_ID	Auto-generated sequence at ETL level
	[Week]		Week_Number	
	[Start]		Start_Date	
			End_Date	
	[End]		Year	Year([End])
			Month	Month([End])

	[Special Events]		Special Events	IF([Special Events]==”” , “Normal Day”, [Special Events])
			Source_System_Code	
Store, Demo		STORE_DIM	Store_ID	Auto-generated sequence at ETL level
	[Store]		Store_Number	
	[Demo].[“Name”]		Store_Name	SELECT D.["NAME"] FROM STORE S INNER JOIN DEMO D ON S.Store = D.["STORE"] WHERE (S.Zone IS NOT NULL) AND (S.Address NOT LIKE 'Closed') AND (D.["NAME"] NOT LIKE '""');
	[City]		Store_City	
	[Price Tier]		Price_Tier	
	[Zone]		Zone	
			Source_System_Code	“Store”
			Active	1 for OPEN record , 0 for CLOSED record to maintain SCD
Product(All 5 source tables for products)		PRODUCT_DIM	Product_ID	Auto-generated sequence at ETL level
	UPC		UPC_Number	
	DESCRIP		Product_Name	
	SIZE		Product_Type	Based on the name of UPC file this value is mapped in

				domain value (“Canned Soup”, “Soft Drinks”, “Cigarette”, “Tuna”, “Bottled Juice”)
			Source_System_Code	Based on the type of UPC File this value is mapped in domain value (“Canned Soup”, “Soft Drinks”, “Cigarette”, “Tuna”, “Bottled Juice”)
CATEGORY		CATEGORY_DIM	Category_ID	Auto-generated sequence at ETL level
	category		Category_Name	
			Source_System_Code	“Category”
			Active	1 for OPEN record , 0 for CLOSED record to maintain SCD
	Product_Sales_ID	PRODUCT_SALES_FACT	Product_Sales_ID	surrogate key
ProductSales_STG				
	Product_ID		Product_ID	
	Store_ID		Store_ID	
	Category_ID		Category_ID	
	Time_ID		Time_ID	
	Price		Price	
	Quantity		Quantity	
	Profit_Percent_per_Dollar		Profit_Percent_per_Dollar	
	Sale_Promotion_Code		Sale_Promotion_Code	

Sales
Move
Margin
Gross-Margin

Sales
Move
Margin
Gross-Margin
Category_Sales_ID

surrogate key

CATEGORYSALES_FACT

Categorydim_Staging

Store_ID
Category_ID
Time_ID

Sales

Category Sales

Weekly Sales for a particular category

Week
Store
Category

5.2.4 Data Extraction Rules

Data extraction is a process of extracting data from the source system and the extracted data is loaded into staging area (a relational database usually different from the data warehouse database) which is transformed and loaded into the data warehouse in remaining ETL process. Data extraction has been applied with proper discretion to build an effective data warehouse. For the purpose of extraction of data, all the data sources have been combined to a single format i.e. tables in Microsoft SQL Server Studio to be later used in data transformation and loading. Source data is in Comma Separated Value (.csv) formats. The data for Week and Store details was extracted from the data manual for Dominick Finer Foods. Comma Separated Value (.csv) files were created for such tabular data. Data from source files is loaded into the staging area.

5.2.5 Data Transformation and Cleansing Rules

Data extracted from source files present in the staging area is the input for data transformation and cleansing process. Data is transformed and cleansing rules are applied on it resulting in cleaned and consistent data throughout. Moreover, dead and useless data is removed to shed light on business people information requirements. Hence, data transformation activities has been properly implemented in this project to produce clean, condensed, complete and standardized data.

General transformation and cleansing rules that are applied to the data are as follows:

1) **Data cleansing**

Data that violates business rules is conformed to these rules. It is done through ETL programs that determine correct data values and the write them into BI target databases. Other steps taken in data cleansing are:

- **Removal of NULL values:** Existing null values in the data will be removed.

- **Removal of dirty data:_** In this part, we will remove blank records and rows containing just a '.' (Dot) in any of their column values. Attributes which are not part of the business questions will also be removed such as attributes apart from STORE, DATE, FROZEN, MEAT, FISH, DAIRY, WEEK have been removed from Customer Count file. Moreover, any other dirt data such as redundant negative values will be removed. Records having sales value of various departments as '.' Will be removed.

2) **Data conversion**

Data extracted from source files are stored as attributes of type varchar (i.e., as string) in the staging area. These have been converted into their respective data types such as int, float and date. Date type field was stored as varchar. Using functionalities in SSIS, date has been split and stored as Year and Month attributes.

3) **Creation of surrogate keys**

Surrogate keys have been created for all the dimension tables and fact tables before loading the data in the data warehouse.

4) **Derived Attributes**

Derived attributes in dimension and fact tables are defined as follows:

- **Product Dimension:** Source_System_Code is set using derived column component feature specifying details of source file. Active flag is also used as derived attribute to show a particular product is presently sold.
- **Store Dimension:** Source_System_Code is set using system derived column component feature specifying details of source file. Active flag is also used as derived attribute to show a particular store is currently in function.
- **Category Dimension:** Source_System_Code is set using system derived column component feature specifying details of source file. Active flag is also used as derived attribute to show that products of a particular category are presently being sold.
- **Time Dimension:** Year and month are derived using start date and end date from functions YEAR(date) and MONTH (date) respectively. Source_System_Code is set using system code generator function feature specifying details of source file.

- **Product Sales fact:** Sales attribute is derived from formula Sales = (price*move)/qty. Margin attribute is derived from formula Margin= (Sales*Profit_Percent_Per_Dollar)/100. Gross_Margin is derived from formula Abs(Margin).

5) SSIS Functions

The SSIS functions that aided during the transformation and cleaning process include:

- **LOOKUP:** It joins additional columns to the data flow by looking up values in a table.
- **Data Conversion:** It converts data from one data type to another.
- **Derived Columns:** It creates new column values by applying expressions to input columns.
- **AGGREGATE:** It aggregate data with functions such as count and sum.
- **UNPIVOT:** It makes an un-normalized dataset into a more normalized version by expanding values from multiple columns in a single record into multiple records with the same values in a single column.

6) Procedure for data loading:

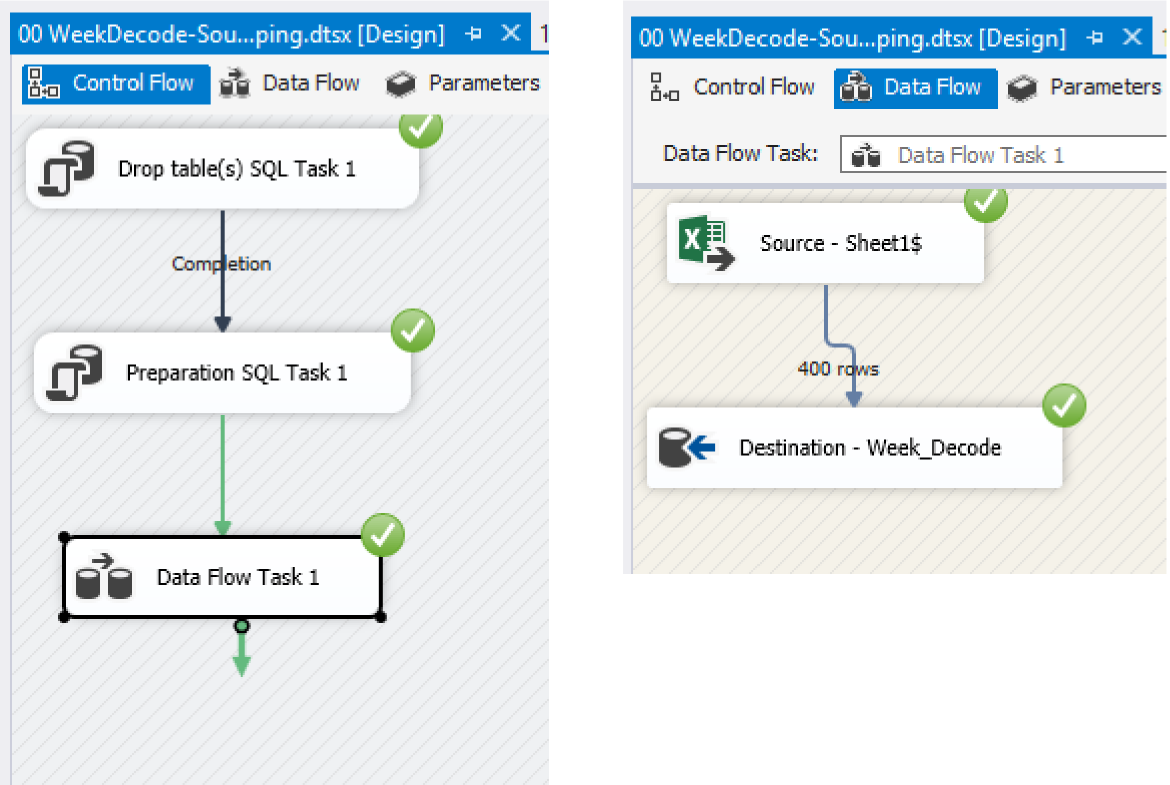
Once the data is cleansed and transformed into a structure according to data warehouse requirements, data is ready to be loaded into the data warehouse. Initially, the tables are populated in the data warehouse and then checking that the data is ready for use. Then the referential integrity between dimension and fact tables is verified to ensure that all records relate to appropriate records in other tables, example - every record in a fact table relates to a record in each dimension table that will be used with that fact table.

5.3 ETL Implementation

The ETL process for the data warehouse was implemented using the SQL Server Management Studio and SSIS (Microsoft Visual studio – SQL Server Data Tools). The steps involved in this process are discussed as below, along with the SSIS tool screen shots explaining the entire process step-by-step.

5.3.1 Time Dimension Table Creation

ETL of Week decode source file



Data Flow Task: Data Flow Task 1

Figure 16 Week Decode Staging

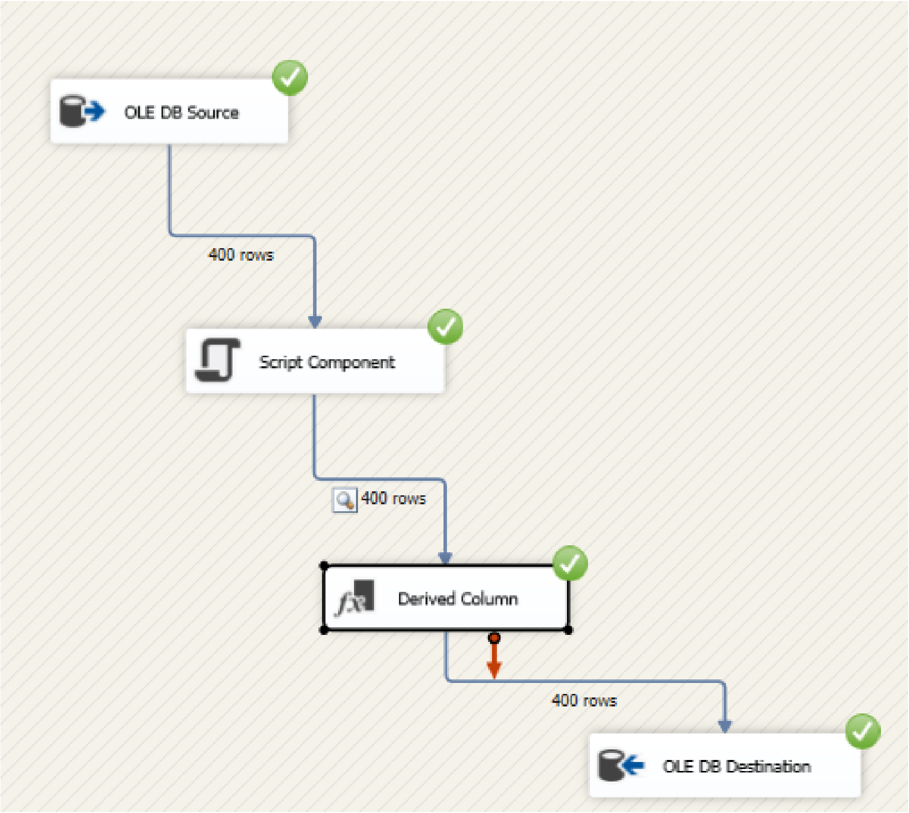
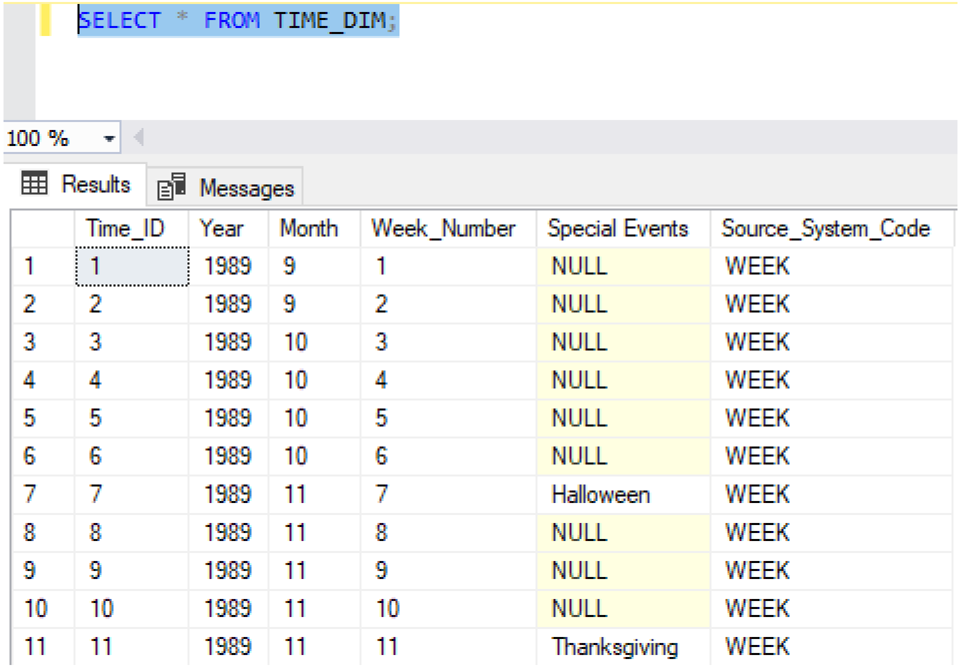


Figure 17 Time Dimension Transformation

Snapshot of Data in Product Dimension



```
SELECT * FROM TIME_DIM;
```

	Time_ID	Year	Month	Week_Number	Special Events	Source_System_Code
1	1	1989	9	1	NULL	WEEK
2	2	1989	9	2	NULL	WEEK
3	3	1989	10	3	NULL	WEEK
4	4	1989	10	4	NULL	WEEK
5	5	1989	10	5	NULL	WEEK
6	6	1989	10	6	NULL	WEEK
7	7	1989	11	7	Halloween	WEEK
8	8	1989	11	8	NULL	WEEK
9	9	1989	11	9	NULL	WEEK
10	10	1989	11	10	NULL	WEEK
11	11	1989	11	11	Thanksgiving	WEEK

Figure 18 Time Dimension Table

5.3.2 Store Dimension Table Creation

ETL of Store and Demo source file

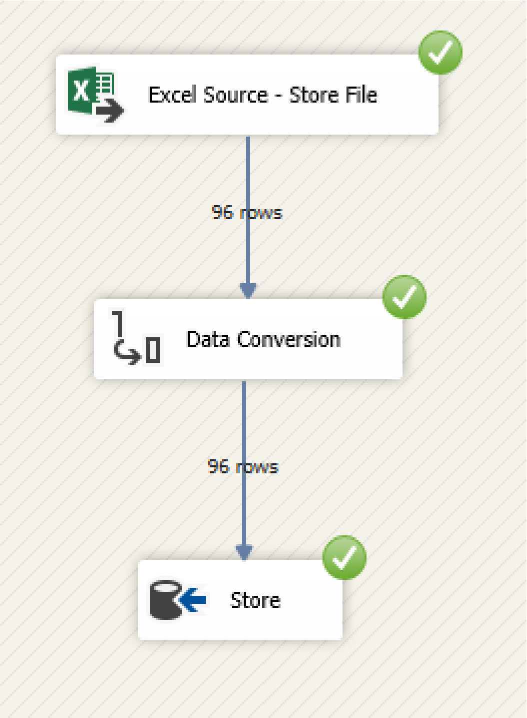


Figure 18- Data Flow for Demo

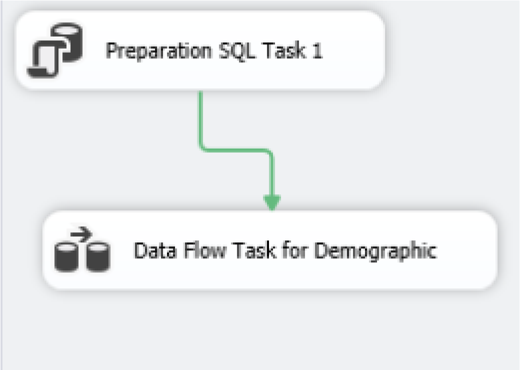
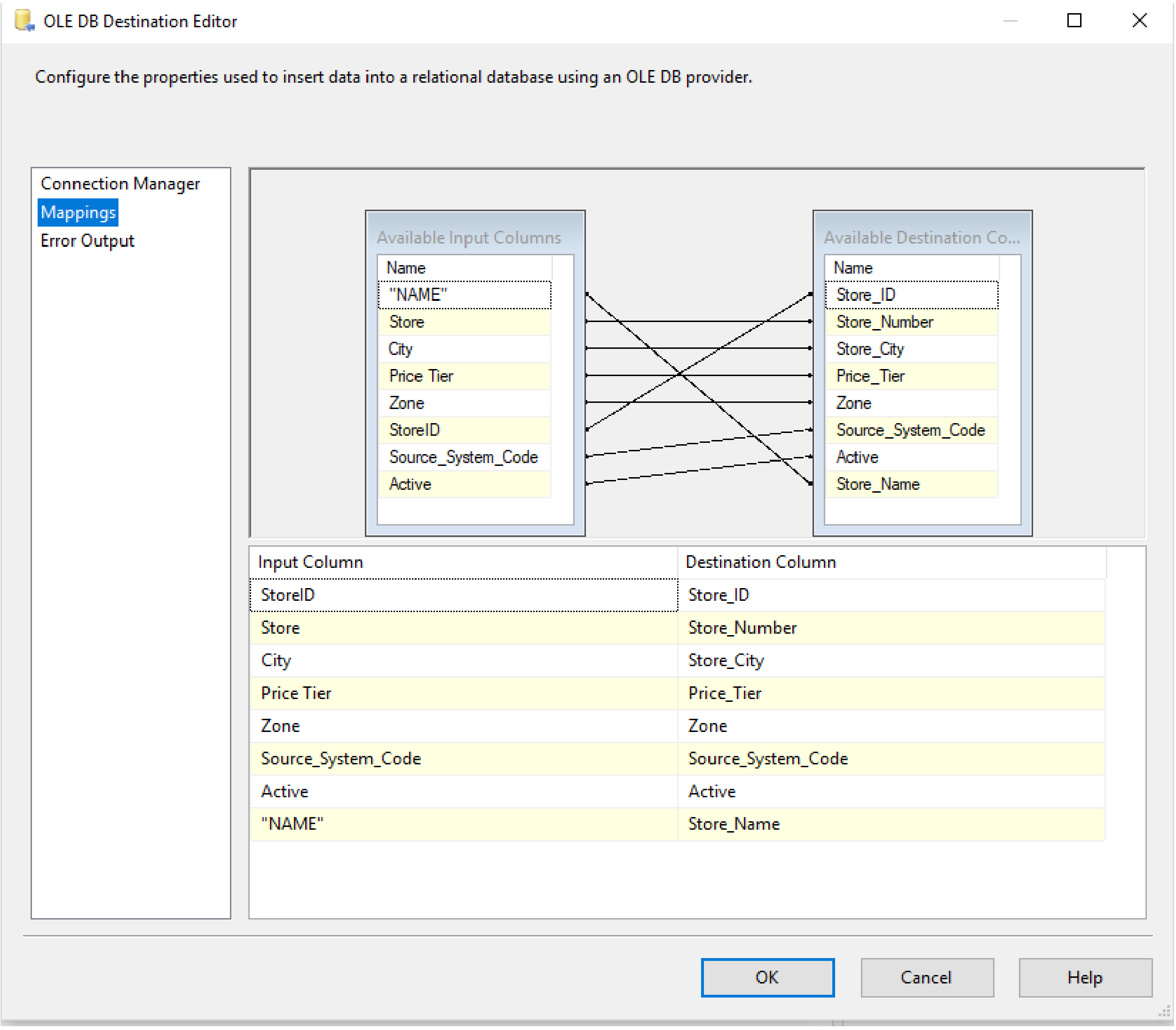


Figure 19- Control Flow for Demo

Figure 20- Store Dimension Staging



Configure the properties used to insert data into a relational database using an OLE DB provider.

Input Column	Destination Column
StoreID	Store_ID
Store	Store_Number
City	Store_City
Price Tier	Price_Tier
Zone	Zone
Source_System_Code	Source_System_Code
Active	Active
"NAME"	Store_Name

Figure 20 Store Dimension Mapping

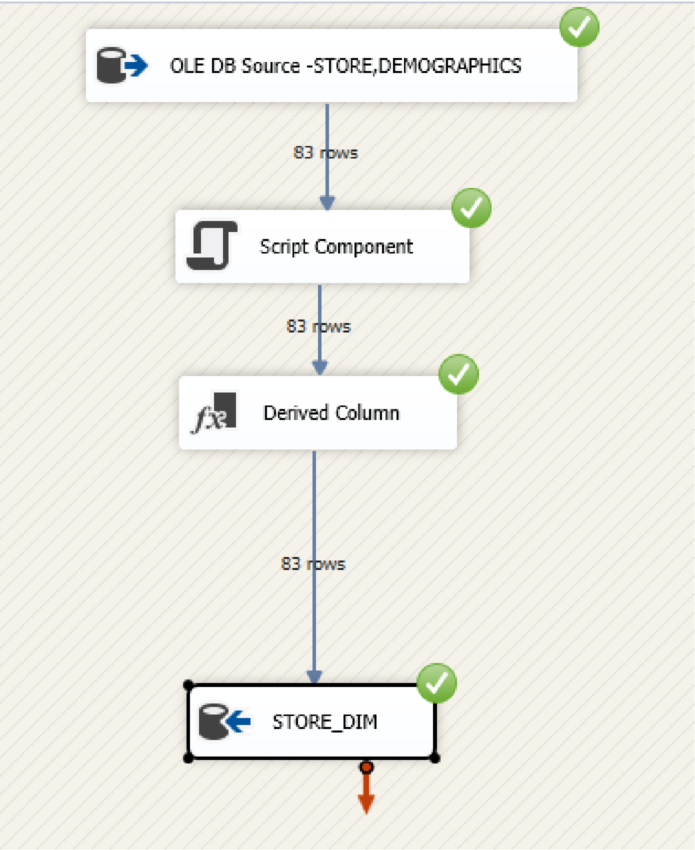
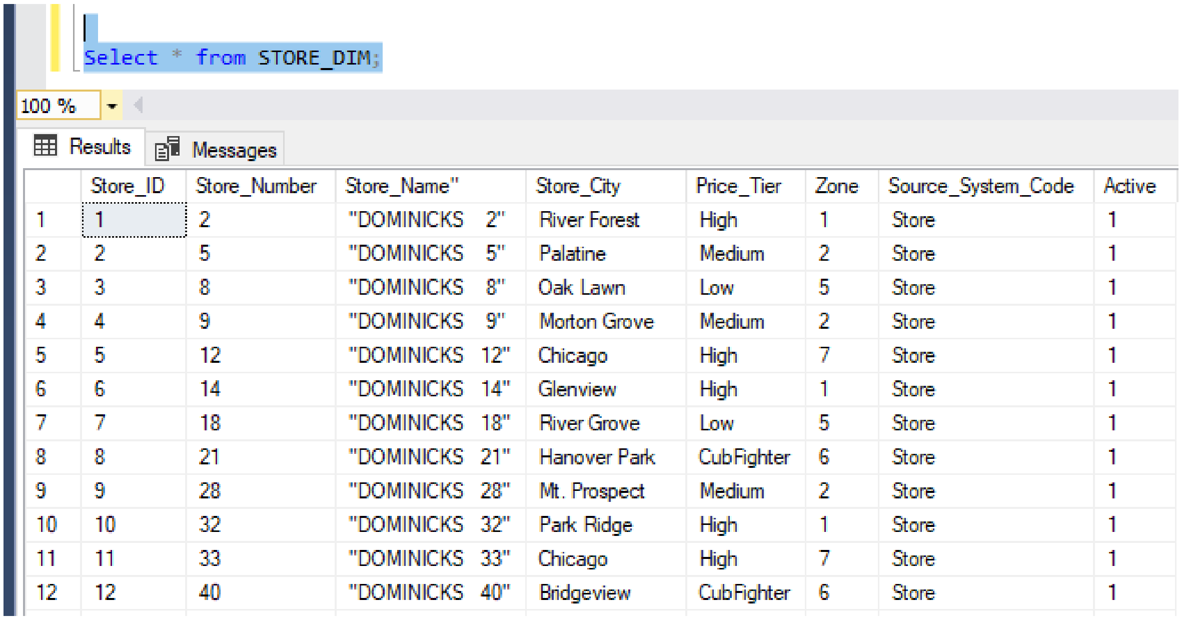


Figure 21 Store Dimension Transformation

Snapshot of Data in Product Dimension



```
Select * from STORE_DIM;
```

	Store_ID	Store_Number	Store_Name"	Store_City	Price_Tier	Zone	Source_System_Code	Active
1	1	2	"DOMINICKS 2"	River Forest	High	1	Store	1
2	2	5	"DOMINICKS 5"	Palatine	Medium	2	Store	1
3	3	8	"DOMINICKS 8"	Oak Lawn	Low	5	Store	1
4	4	9	"DOMINICKS 9"	Morton Grove	Medium	2	Store	1
5	5	12	"DOMINICKS 12"	Chicago	High	7	Store	1
6	6	14	"DOMINICKS 14"	Glenview	High	1	Store	1
7	7	18	"DOMINICKS 18"	River Grove	Low	5	Store	1
8	8	21	"DOMINICKS 21"	Hanover Park	CubFighter	6	Store	1
9	9	28	"DOMINICKS 28"	Mt. Prospect	Medium	2	Store	1
10	10	32	"DOMINICKS 32"	Park Ridge	High	1	Store	1
11	11	33	"DOMINICKS 33"	Chicago	High	7	Store	1
12	12	40	"DOMINICKS 40"	Bridgeview	CubFighter	6	Store	1

Figure 22 Store Dimension Table

5.3.3 Product Dimension Table Creation

<u>ETL of UPC Source Files</u>

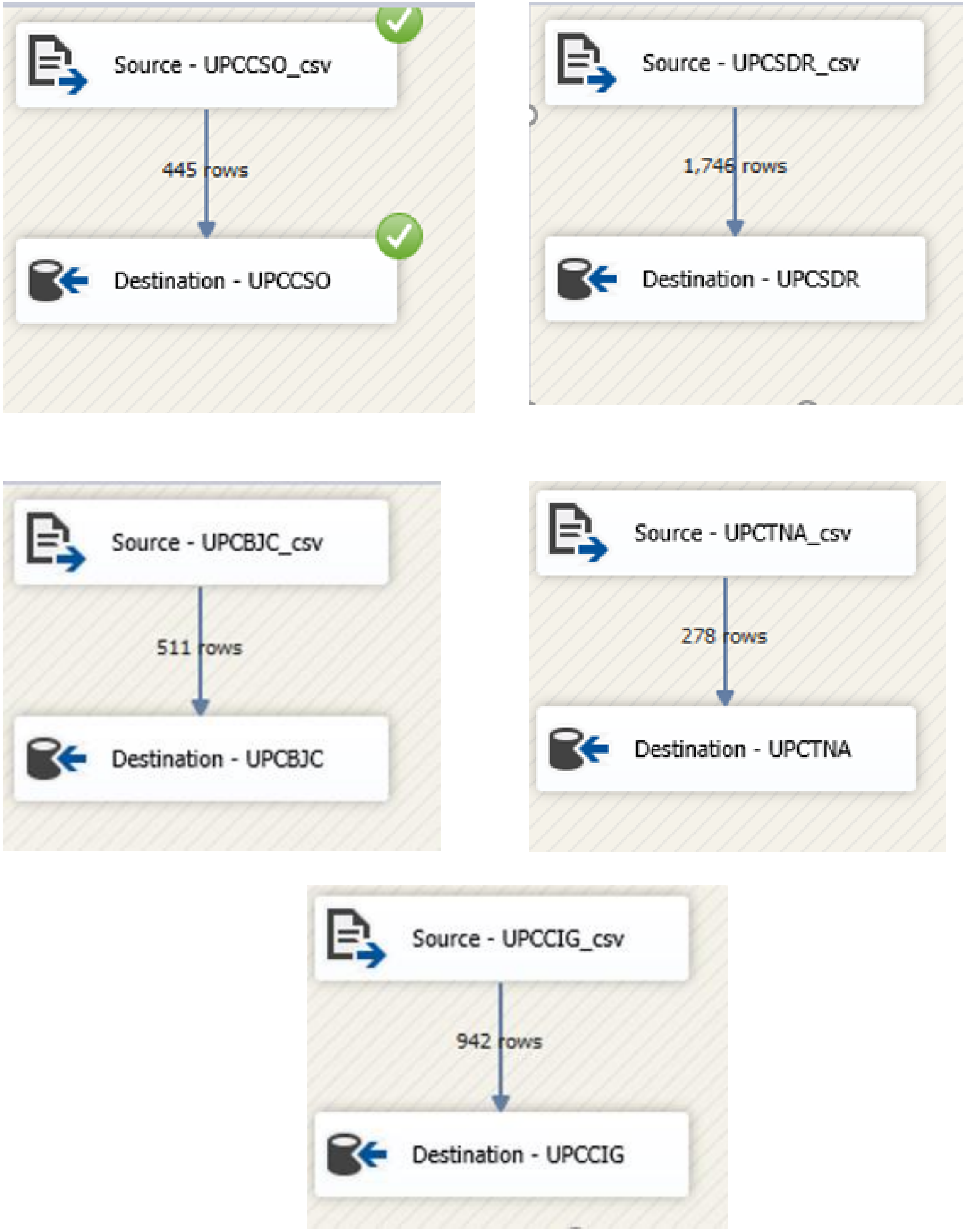
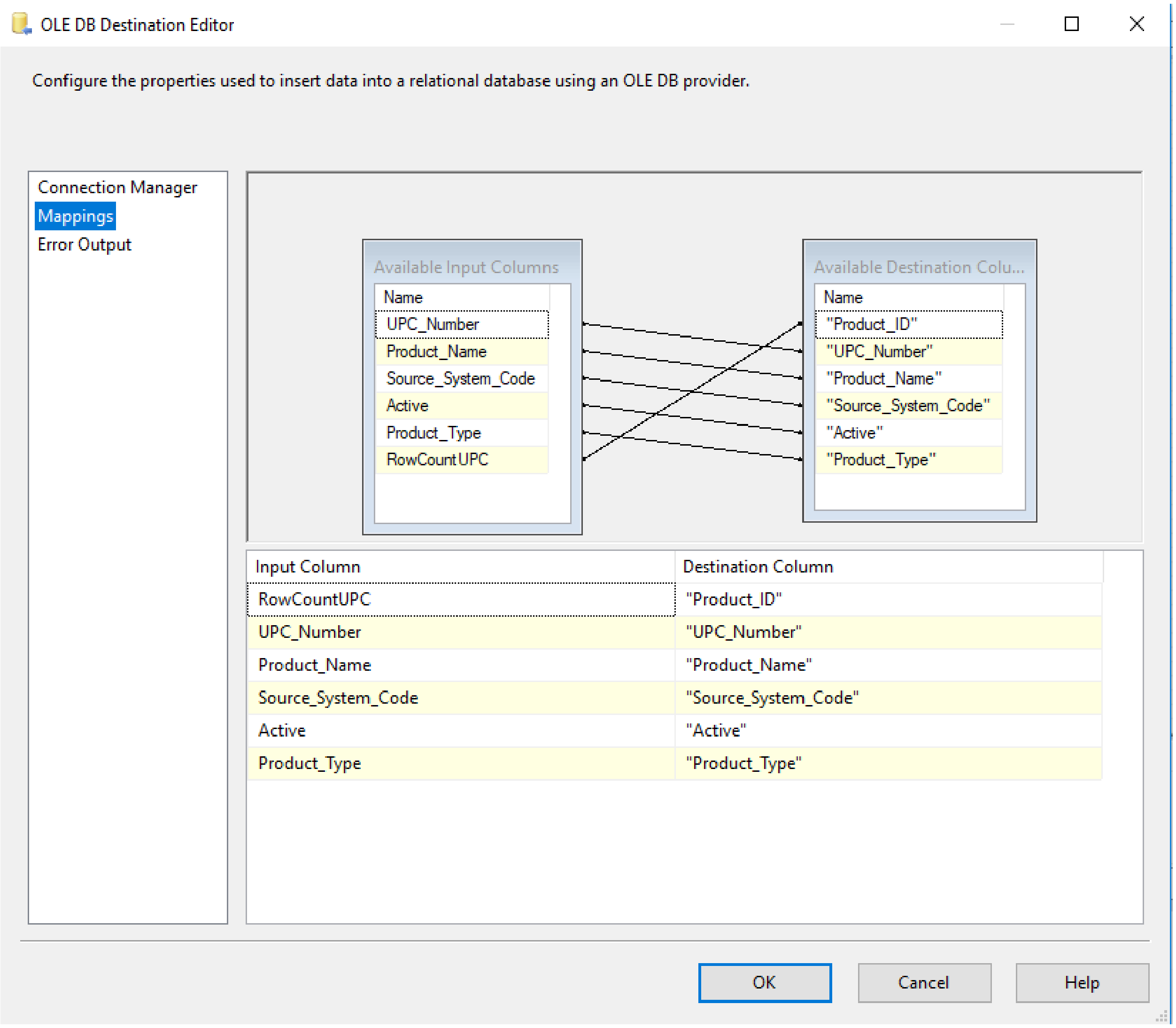


Figure 23 UPC Source File Staging



OLE DB Destination Editor

Configure the properties used to insert data into a relational database using an OLE DB provider.

Input Column	Destination Column
RowCountUPC	"Product_ID"
UPC_Number	"UPC_Number"
Product_Name	"Product_Name"
Source_System_Code	"Source_System_Code"
Active	"Active"
Product_Type	"Product_Type"

Figure 24 Product Dimension Mapping

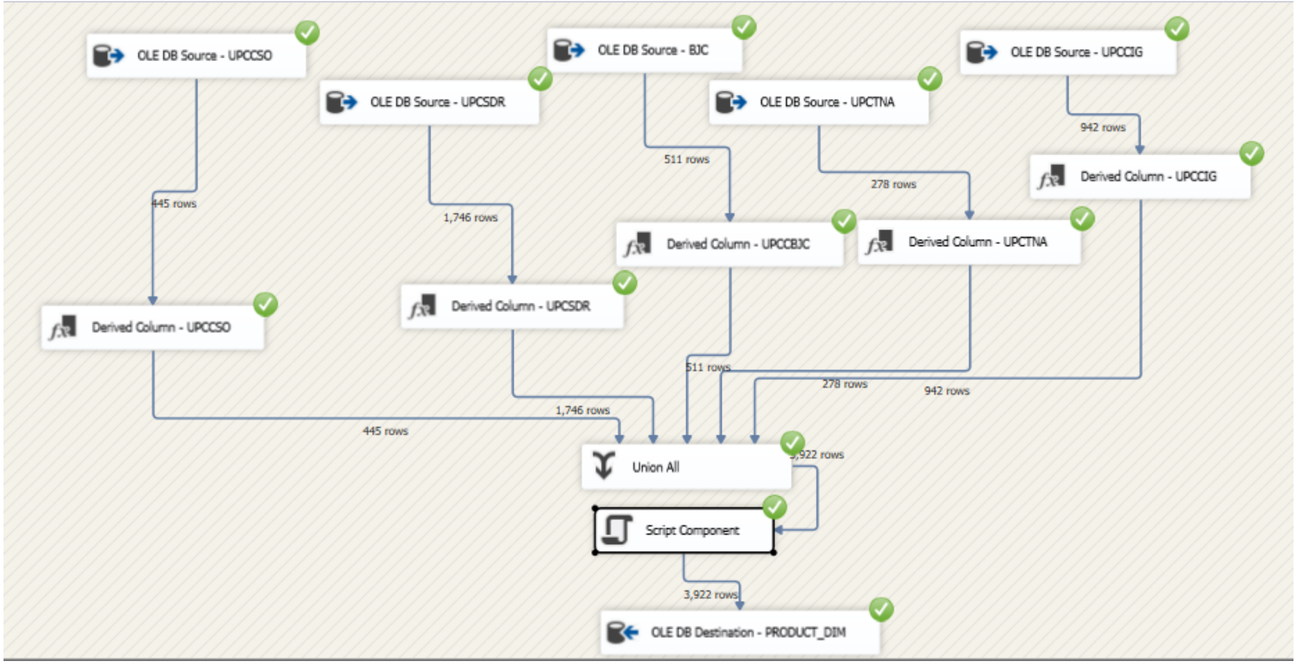
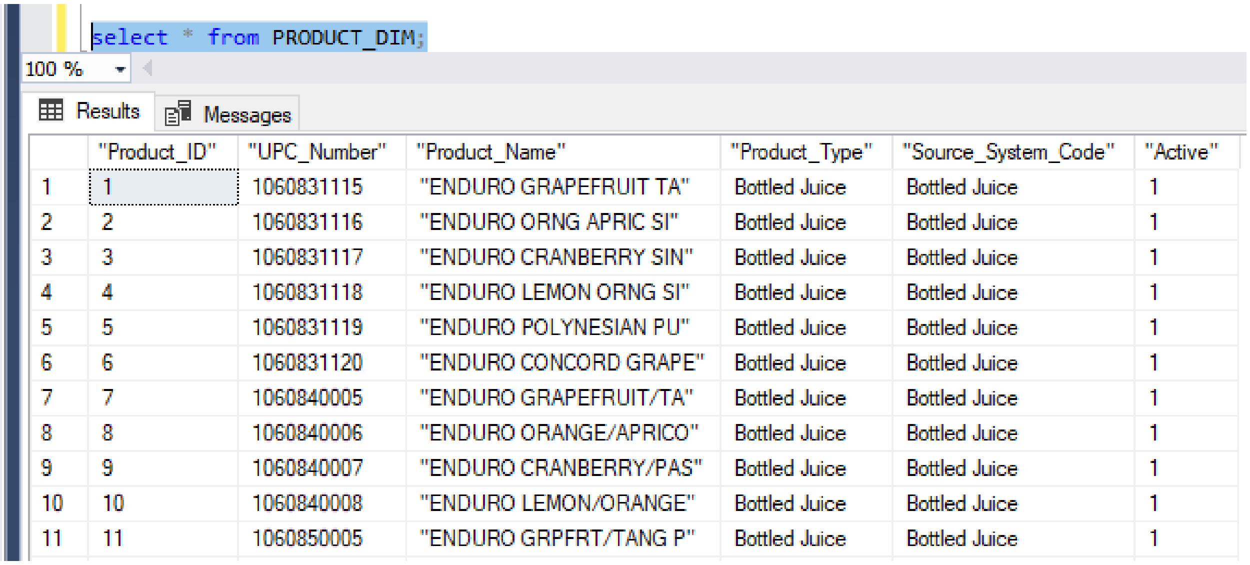


Figure 25 Product Dimension Transformation

Snapshot of Data in Product Dimension



```
select * from PRODUCT_DIM;
```

	"Product_ID"	"UPC_Number"	"Product_Name"	"Product_Type"	"Source_System_Code"	"Active"
1	1	1060831115	"ENDURO GRAPEFRUIT TA"	Bottled Juice	Bottled Juice	1
2	2	1060831116	"ENDURO ORNG APRIC SI"	Bottled Juice	Bottled Juice	1
3	3	1060831117	"ENDURO CRANBERRY SIN"	Bottled Juice	Bottled Juice	1
4	4	1060831118	"ENDURO LEMON ORNG SI"	Bottled Juice	Bottled Juice	1
5	5	1060831119	"ENDURO POLYNESIAN PU"	Bottled Juice	Bottled Juice	1
6	6	1060831120	"ENDURO CONCORD GRAPE"	Bottled Juice	Bottled Juice	1
7	7	1060840005	"ENDURO GRAPEFRUIT/TA"	Bottled Juice	Bottled Juice	1
8	8	1060840006	"ENDURO ORANGE/APRICO"	Bottled Juice	Bottled Juice	1
9	9	1060840007	"ENDURO CRANBERRY/PAS"	Bottled Juice	Bottled Juice	1
10	10	1060840008	"ENDURO LEMON/ORANGE"	Bottled Juice	Bottled Juice	1
11	11	1060850005	"ENDURO GRPFRT/TANG P"	Bottled Juice	Bottled Juice	1

Figure 26 Product Dimension Table

5.3.4 Category Dimension Table Creation

ETL of CCOUNT files

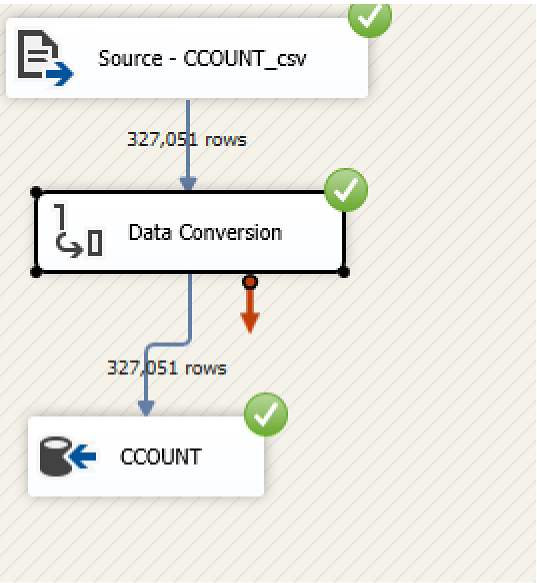
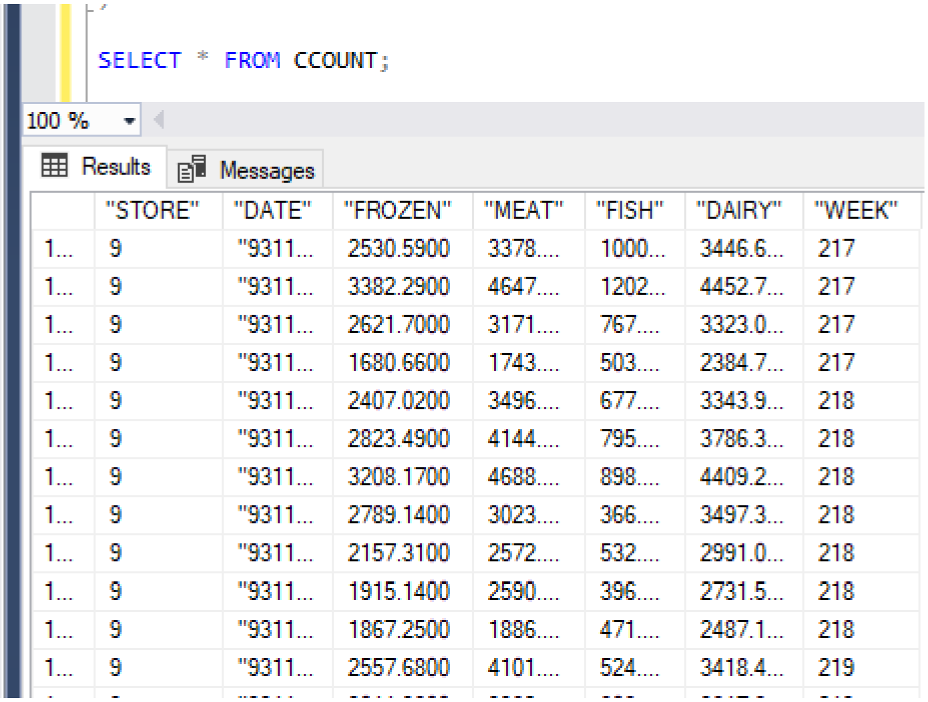


Figure 27 CCount File Staging

Snapshot of CCOUNT

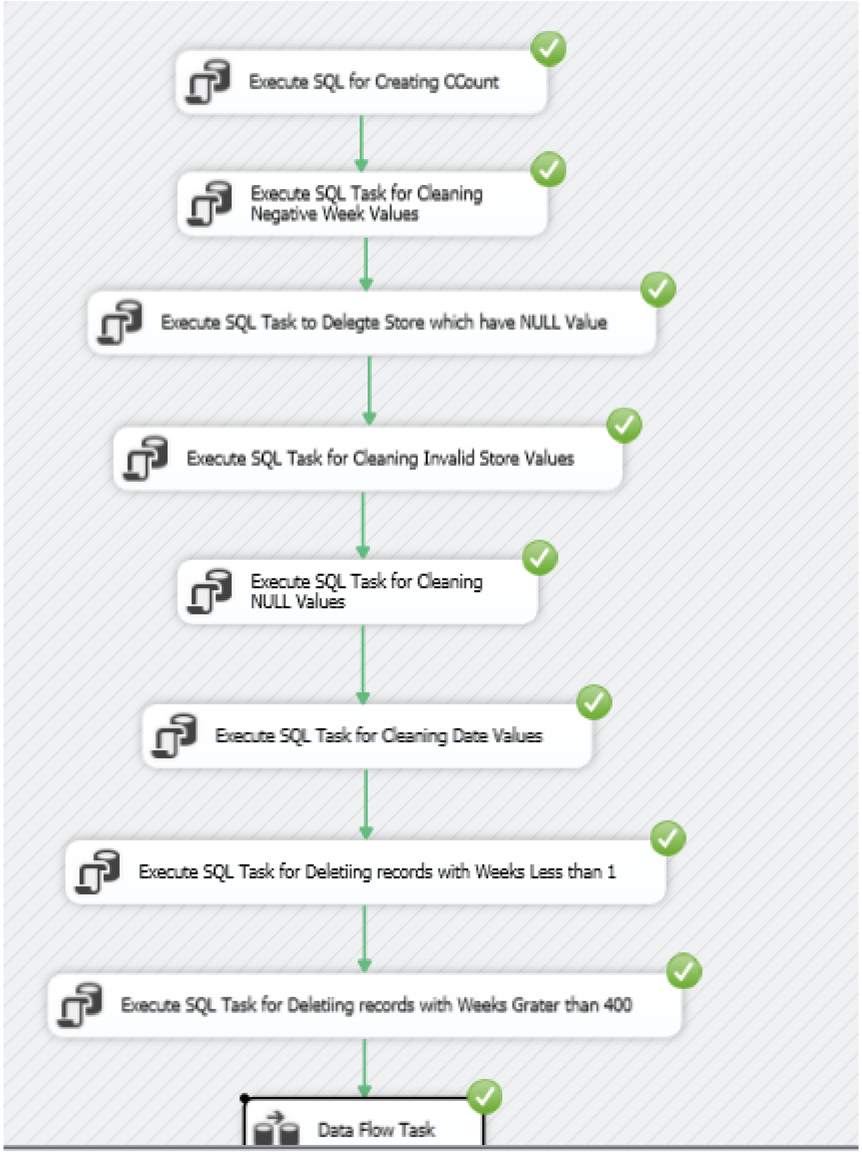


```
SELECT * FROM CCOUNT;
```

	"STORE"	"DATE"	"FROZEN"	"MEAT"	"FISH"	"DAIRY"	"WEEK"
1...	9	"9311...	2530.5900	3378....	1000...	3446.6...	217
1...	9	"9311...	3382.2900	4647....	1202...	4452.7...	217
1...	9	"9311...	2621.7000	3171....	767....	3323.0...	217
1...	9	"9311...	1680.6600	1743....	503....	2384.7...	217
1...	9	"9311...	2407.0200	3496....	677....	3343.9...	218
1...	9	"9311...	2823.4900	4144....	795....	3786.3...	218
1...	9	"9311...	3208.1700	4688....	898....	4409.2...	218
1...	9	"9311...	2789.1400	3023....	366....	3497.3...	218
1...	9	"9311...	2157.3100	2572....	532....	2991.0...	218
1...	9	"9311...	1915.1400	2590....	396....	2731.5...	218
1...	9	"9311...	1867.2500	1886....	471....	2487.1...	218
1...	9	"9311...	2557.6800	4101....	524....	3418.4...	219

Figure 28 CCount File Snapshot

Cleaning CCOUNT files using SQL Queries



Execute SQL for Creating CCount
Execute SQL Task for Cleaning Negative Week Values
Execute SQL Task to Delegte Store which have NULL Value
Execute SQL Task for Cleaning Invalid Store Values
Execute SQL Task for Cleaning NULL Values
Execute SQL Task for Cleaning Date Values
Execute SQL Task for Deletiing records with Weeks Less than 1
Execute SQL Task for Deletiing records with Weeks Grater than 400
Data Flow Task

Figure 29 CCount File Cleaning

<u>Transforming CCount to Category Dimension</u>

<u>SQL Queries used</u>

<u>SQL for creating CCOUNT</u>

SELECT
["STORE"], ["DATE"], ["DAIRY"], ["FROZEN"], ["MEAT"],["FISH"],["WEEK"]
INTO CCOUNT_Clean_STG
FROM CCOUNT

<u>SQL for cleaning negative week Values</u>

```
DELETE FROM CCOUNT_Clean_STG WHERE (["WEEK"] < 0)
```

Delete stores which have NULL Value

```
DELETE FROM CCOUNT_Clean_STG WHERE ["STORE"] is NULL
```

SQL Task for cleaning Invalid Store Values

```
DELETE FROM CCOUNT_Clean_STG WHERE ["STORE"] NOT IN (SELECT [Store_Number] FROM
Store_DIM_TEMP)
```

SQL Task for Cleaning NULL Values

```
DELETE FROM CCOUNT_Clean_STG WHERE (["STORE"] IS NULL) OR (["DATE"] IS NULL) OR
(["DAIRY"] IS NULL) OR (["FROZEN"] IS NULL) OR (["MEAT"] IS NULL) OR (["FISH"] IS NULL) OR
(["WEEK"] IS NULL)
```

SQL Task for Cleaning Date Values

```
UPDATE CCOUNT_Clean_STG SET ["DATE"] = REPLACE(["DATE"], '"', '')
```

SQL Task for Deleting records with Weeks Less than 1

```
DELETE FROM CCOUNT_Clean_STG WHERE ["WEEK"] <1
```

SQL Task for Deleting records with Weeks Grater than 400

```
DELETE FROM CCOUNT_Clean_STG
WHERE ["WEEK"] > 400
```

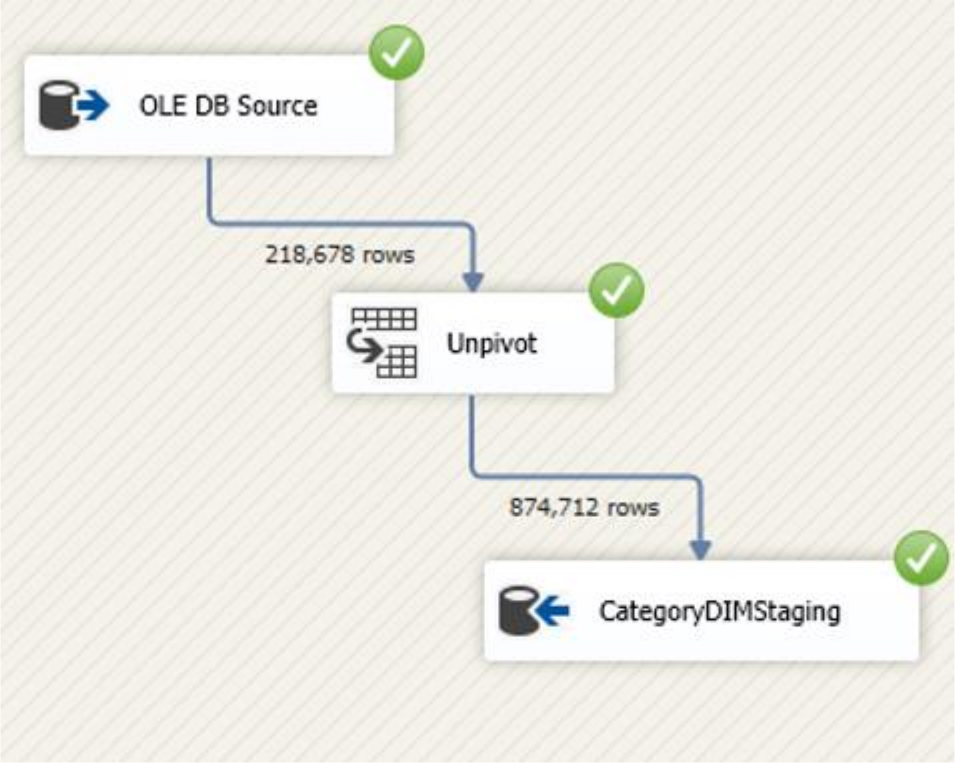


Figure 30 Category Dimension Transformation

5.3.5 CategorySales Fact Table Creation

Store Lookup, Time Lookup, Category Lookup are used in creating CategorySales fact table.

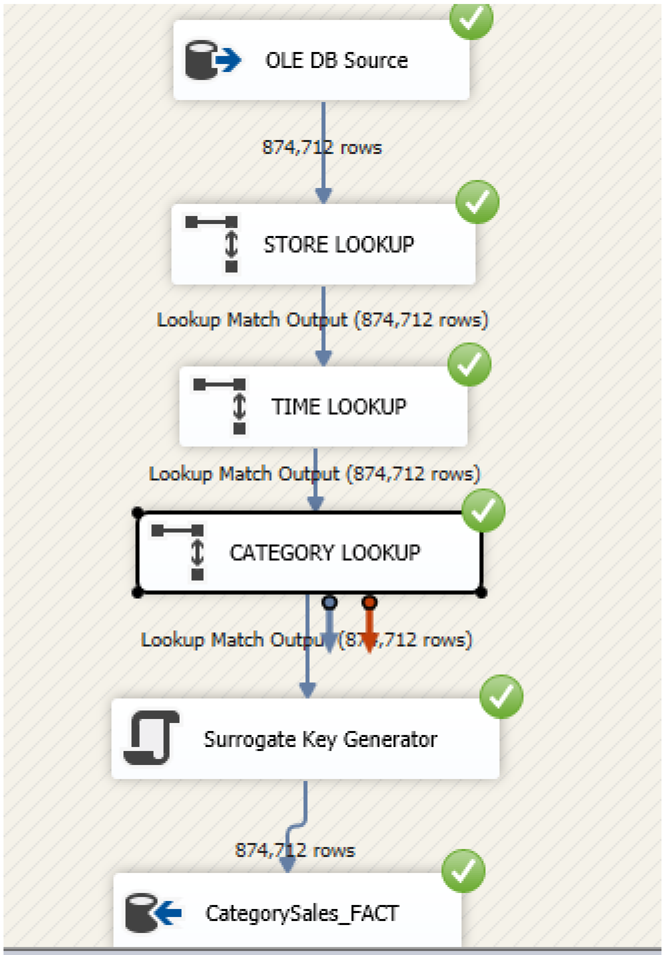
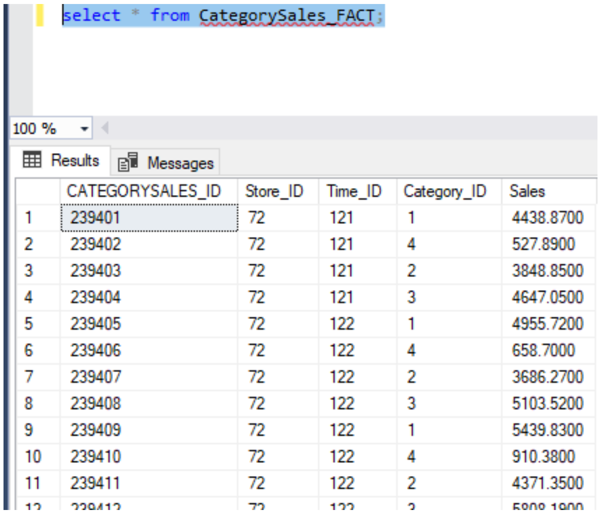


Figure 31 ETL of CategorySales Fact Table

Snapshot of data in CategorySales_FACT



```
select * from CategorySales_FACT;
```

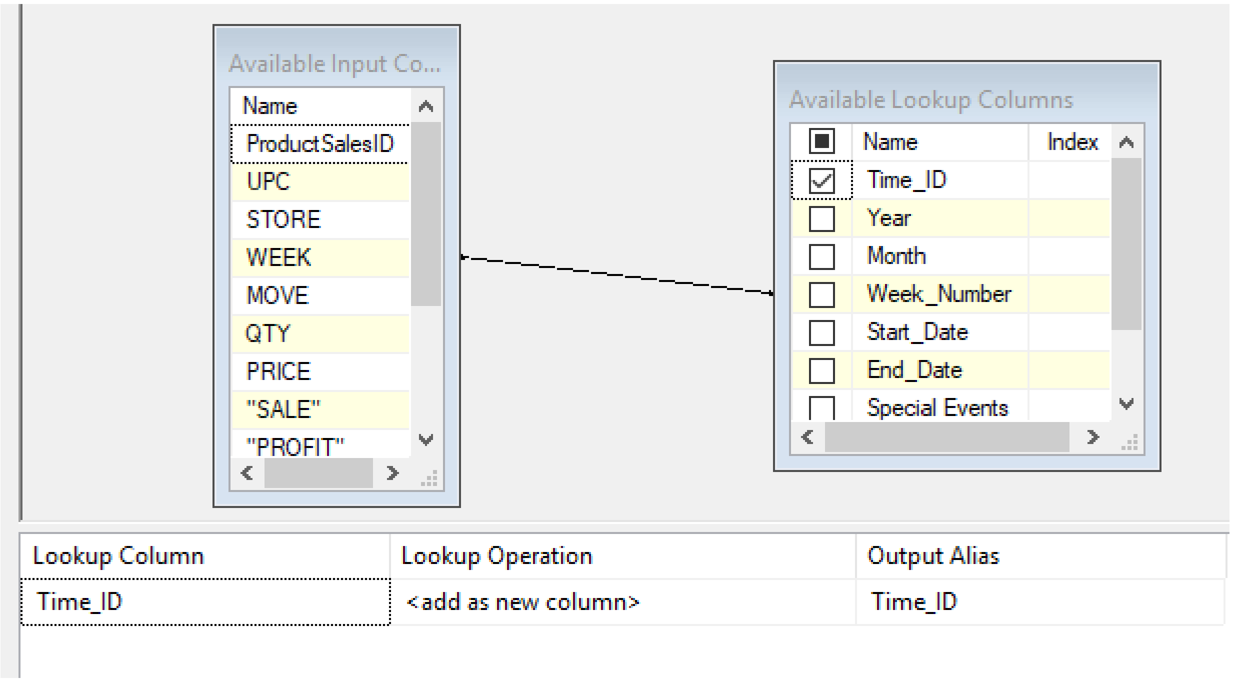
	CATEGORYSALES_ID	Store_ID	Time_ID	Category_ID	Sales
1	239401	72	121	1	4438.8700
2	239402	72	121	4	527.8900
3	239403	72	121	2	3848.8500
4	239404	72	121	3	4647.0500
5	239405	72	122	1	4955.7200
6	239406	72	122	4	658.7000
7	239407	72	122	2	3686.2700
8	239408	72	122	3	5103.5200
9	239409	72	122	1	5439.8300
10	239410	72	122	4	910.3800
11	239411	72	122	2	4371.3500
12	239412	72	122	3	5808.1900

Figure 32 CategorySales Fact Table Snapshot

5.3.6 ProductSales Fact Table Creation

Store Lookup, Time Lookup, Category Lookup and Product Lookup are used in creating ProductSales fact table.

Time Lookup



Available Input Co...

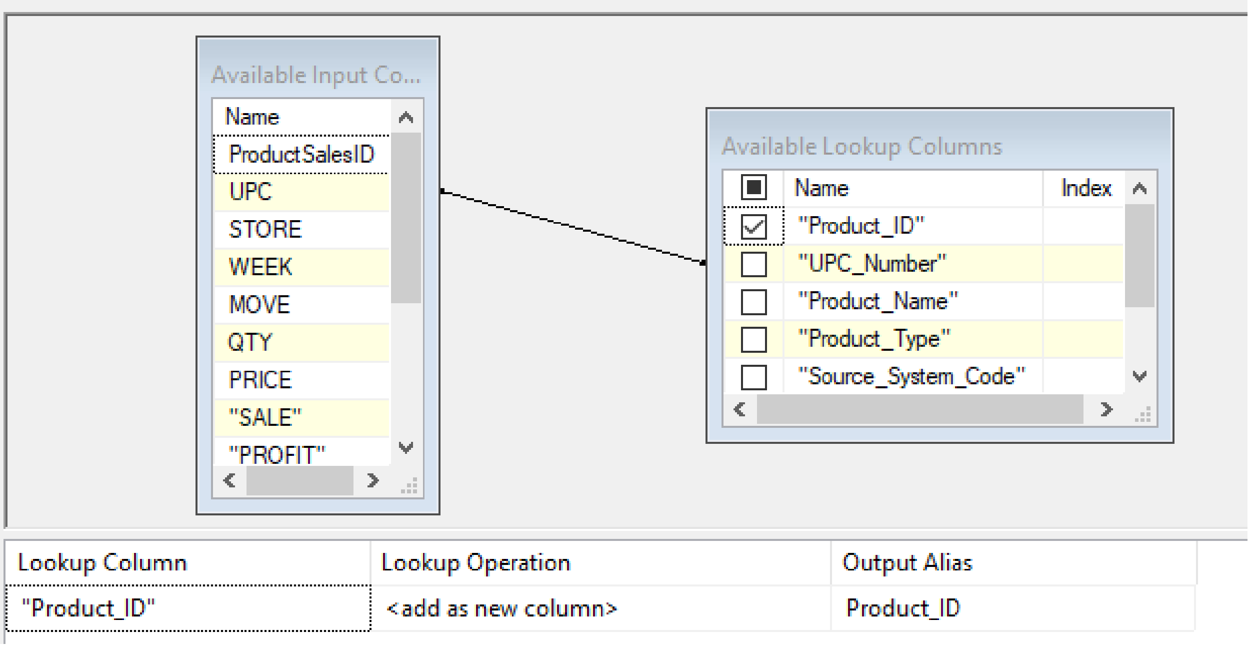
Name
ProductSalesID
UPC
STORE
WEEK
MOVE
QTY
PRICE
"SALE"
"PROFIT"

Available Lookup Columns

■	Name	Index
☑	Time_ID	
□	Year	
□	Month	
□	Week_Number	
□	Start_Date	
□	End_Date	
□	Special Events	

Lookup Column	Lookup Operation	Output Alias
Time_ID	<add as new column>	Time_ID

Product Lookup



Available Input Co...

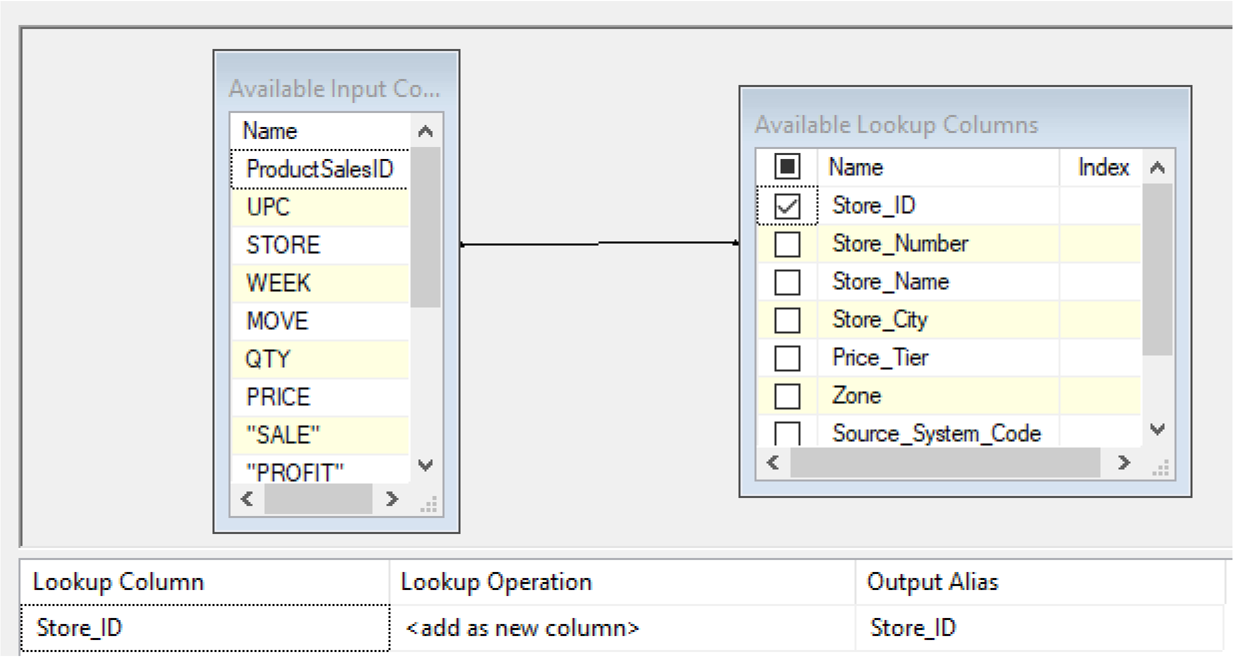
Name
ProductSalesID
UPC
STORE
WEEK
MOVE
QTY
PRICE
"SALE"
"PROFIT"

Available Lookup Columns

■	Name	Index
☑	"Product_ID"	
□	"UPC_Number"	
□	"Product_Name"	
□	"Product_Type"	
□	"Source_System_Code"	

Lookup Column	Lookup Operation	Output Alias
"Product_ID"	<add as new column>	Product_ID

Store Lookup



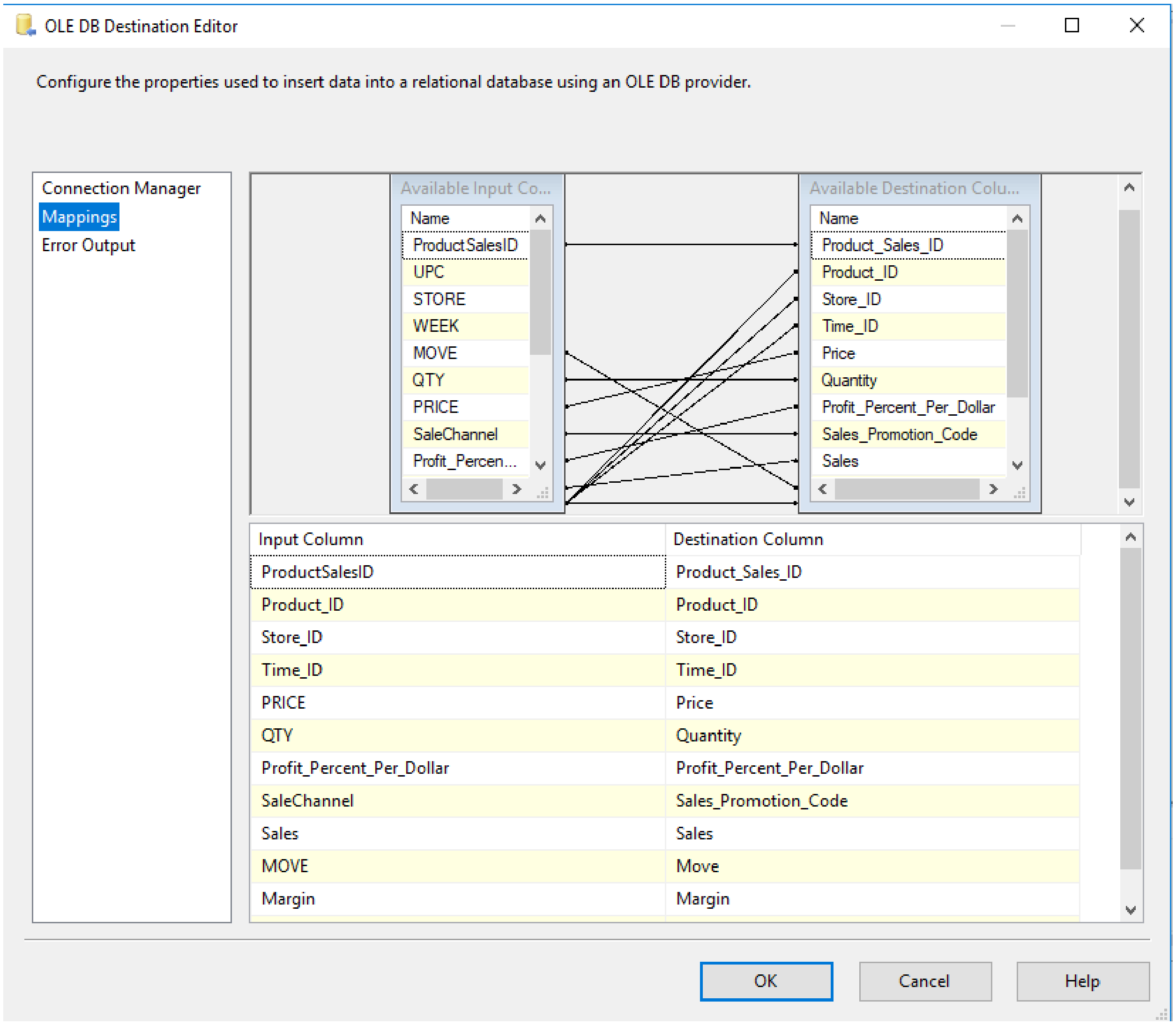
Available Input Co...

Name
ProductSalesID
UPC
STORE
WEEK
MOVE
QTY
PRICE
"SALE"
"PROFIT"

Available Lookup Columns

▣	Name	Index
☑	Store_ID	
□	Store_Number	
□	Store_Name	
□	Store_City	
□	Price_Tier	
□	Zone	
□	Source_System_Code	

Lookup Column	Lookup Operation	Output Alias
Store_ID	<add as new column>	Store_ID



Configure the properties used to insert data into a relational database using an OLE DB provider.

Input Column	Destination Column
ProductSalesID	Product_Sales_ID
Product_ID	Product_ID
Store_ID	Store_ID
Time_ID	Time_ID
PRICE	Price
QTY	Quantity
Profit_Percent_Per_Dollar	Profit_Percent_Per_Dollar
SaleChannel	Sales_Promotion_Code
Sales	Sales
MOVE	Move
Margin	Margin

Figure 33 ProductSales Fact Table Mapping

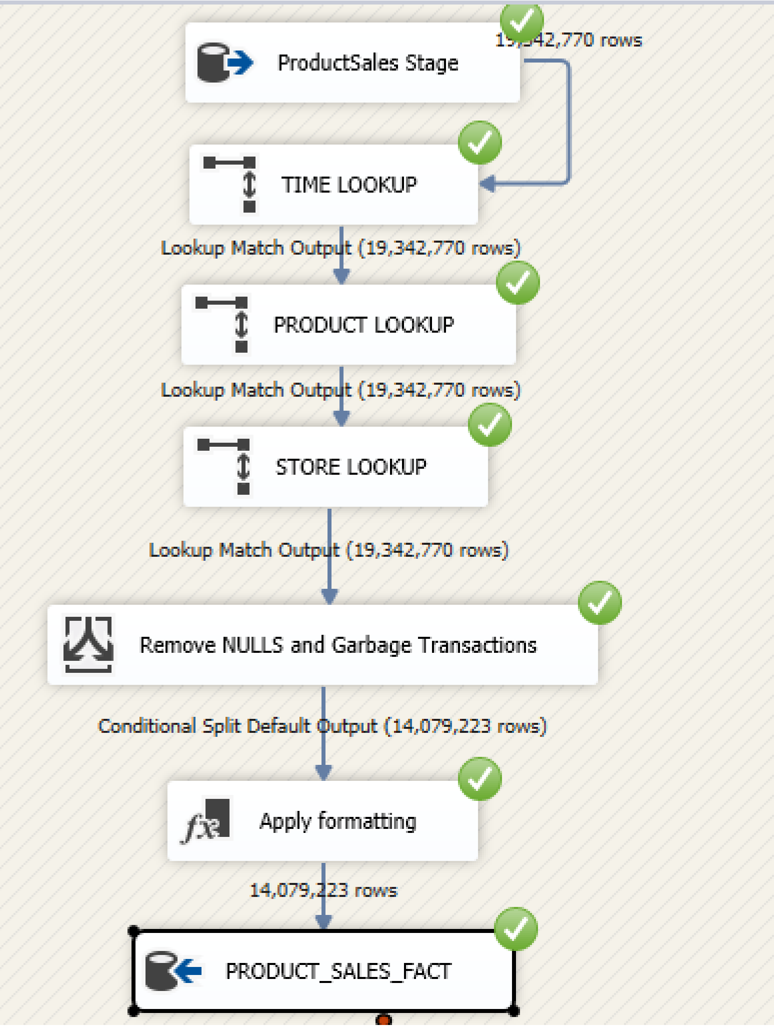
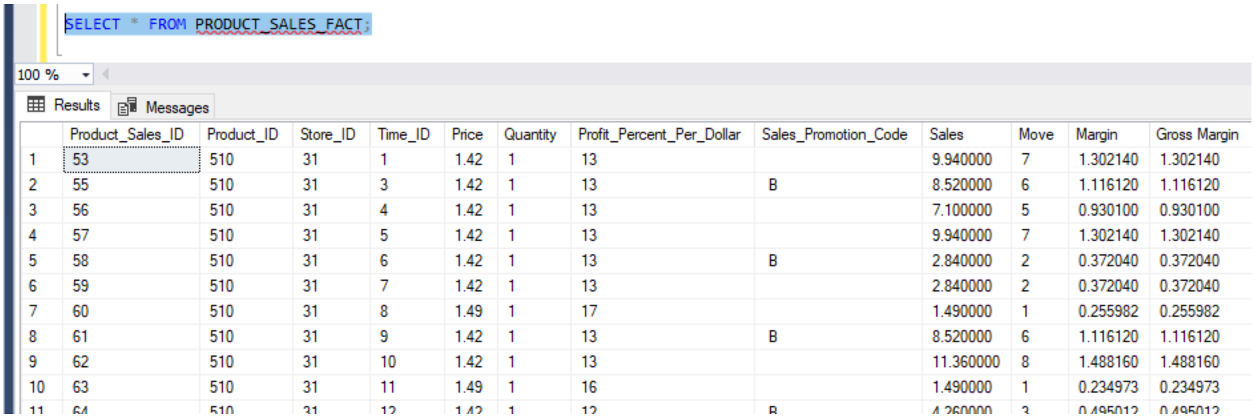

ProductSales Stage
19,342,770 rows
TIME LOOKUP
Lookup Match Output (19,342,770 rows)
PRODUCT LOOKUP
Lookup Match Output (19,342,770 rows)
STORE LOOKUP
Lookup Match Output (19,342,770 rows)
Remove NULLS and Garbage Transactions
Conditional Split Default Output (14,079,223 rows)
Apply formatting
14,079,223 rows
PRODUCT_SALES_FACT

Figure 34 ETL of ProductSales Fact Table Mapping

<u>Snapshot of data in ProductSales_FACT</u>

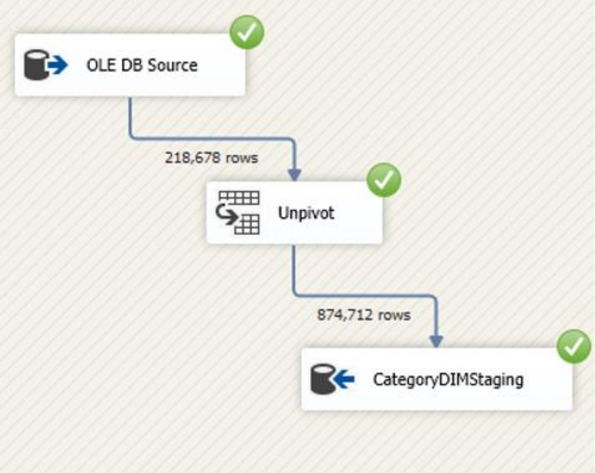


```
SELECT * FROM PRODUCT_SALES_FACT;
```

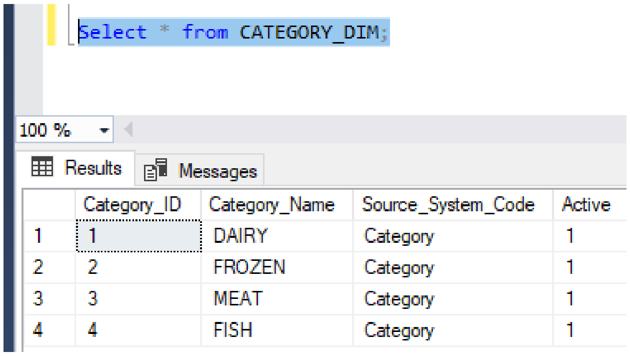
	Product_Sales_ID	Product_ID	Store_ID	Time_ID	Price	Quantity	Profit_Percent_Per_Dollar	Sales_Promotion_Code	Sales	Move	Margin	Gross Margin
1	53	510	31	1	1.42	1	13		9.940000	7	1.302140	1.302140
2	55	510	31	3	1.42	1	13	B	8.520000	6	1.116120	1.116120
3	56	510	31	4	1.42	1	13		7.100000	5	0.930100	0.930100
4	57	510	31	5	1.42	1	13		9.940000	7	1.302140	1.302140
5	58	510	31	6	1.42	1	13	B	2.840000	2	0.372040	0.372040
6	59	510	31	7	1.42	1	13		2.840000	2	0.372040	0.372040
7	60	510	31	8	1.49	1	17		1.490000	1	0.255982	0.255982
8	61	510	31	9	1.42	1	13	B	8.520000	6	1.116120	1.116120
9	62	510	31	10	1.42	1	13		11.360000	8	1.488160	1.488160
10	63	510	31	11	1.49	1	16		1.490000	1	0.234973	0.234973
11	64	510	31	12	1.42	1	13	B	4.260000	3	0.495012	0.495012

Figure 35 ProductSales Fact Table Snapshot

Transforming CCount to Category Dimension

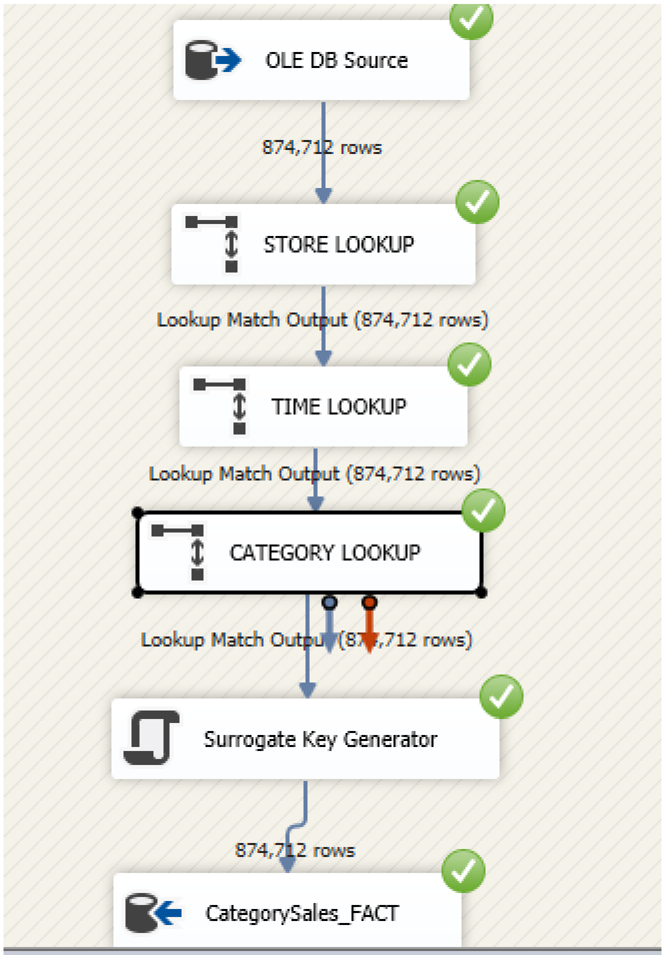


Snapshot of Category Dimension

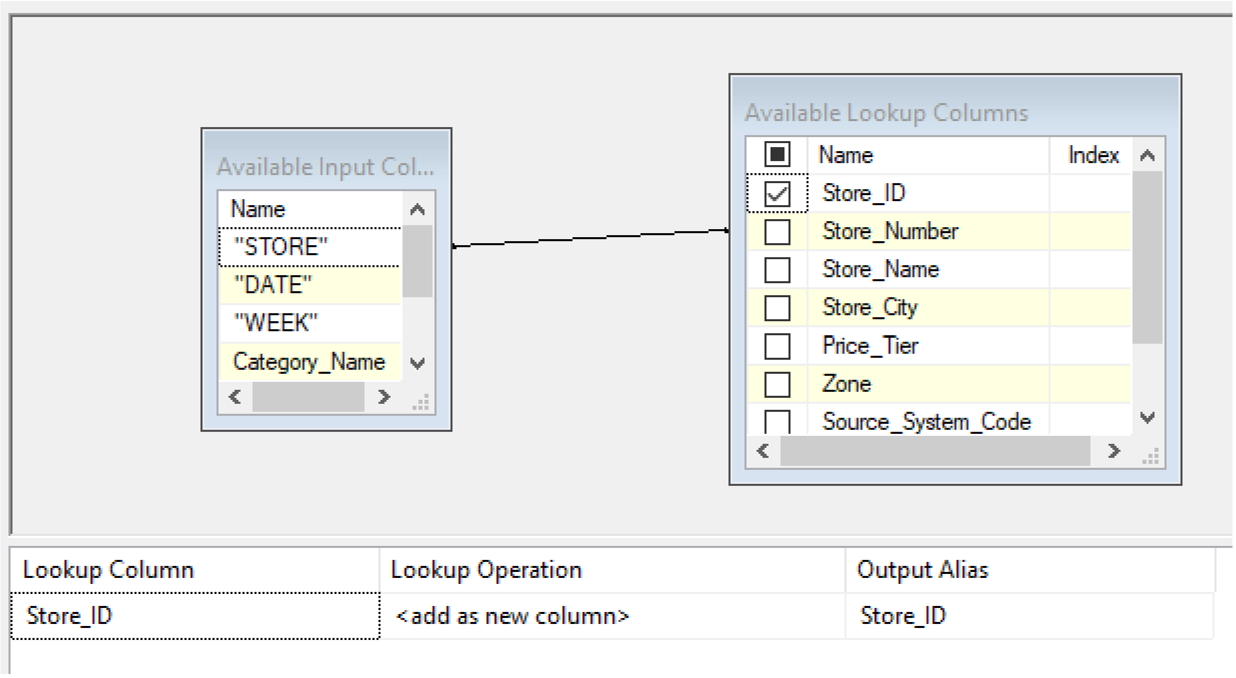

```
Select * from CATEGORY_DIM;
```

	Category_ID	Category_Name	Source_System_Code	Active
1	1	DAIRY	Category	1
2	2	FROZEN	Category	1
3	3	MEAT	Category	1
4	4	FISH	Category	1

5.3.7 Category Fact Table Creation

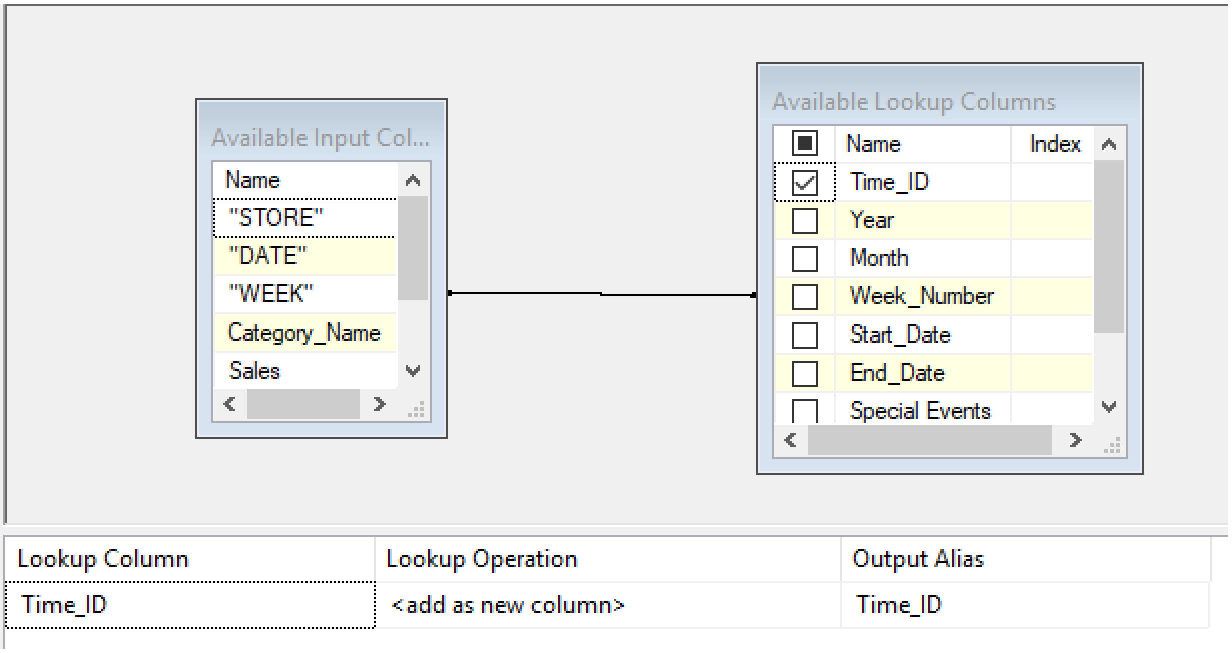


Store Lookup



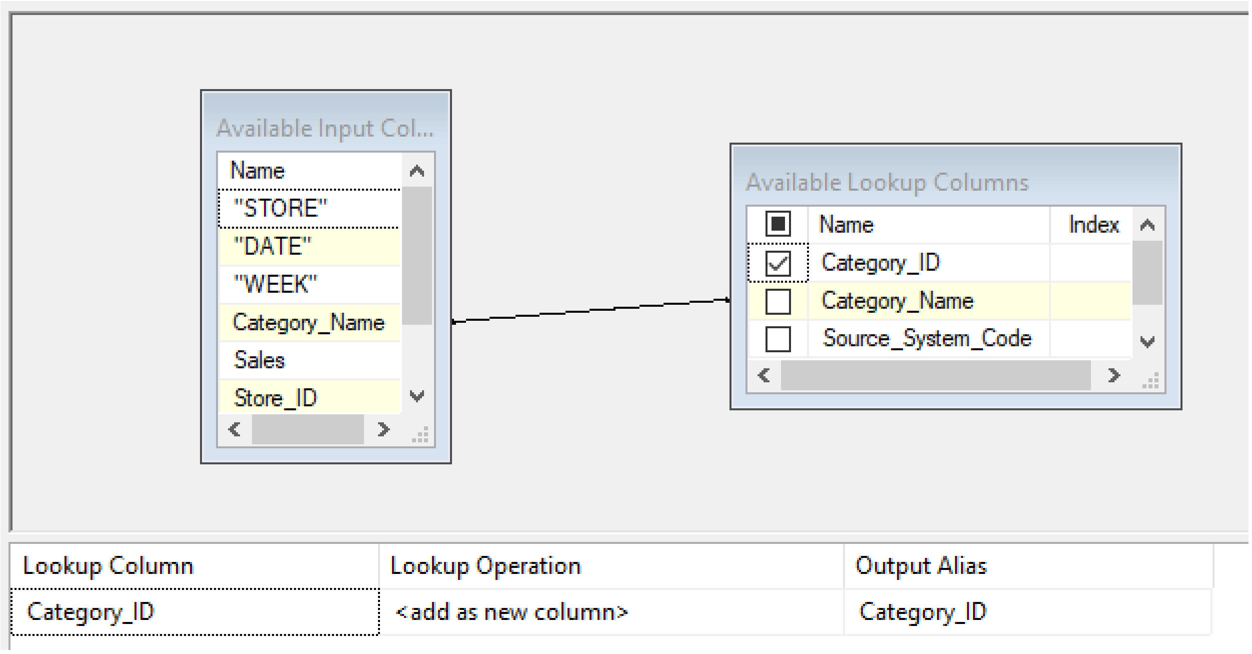
Lookup Column	Lookup Operation	Output Alias
Store_ID	<add as new column>	Store_ID

Time Lookup



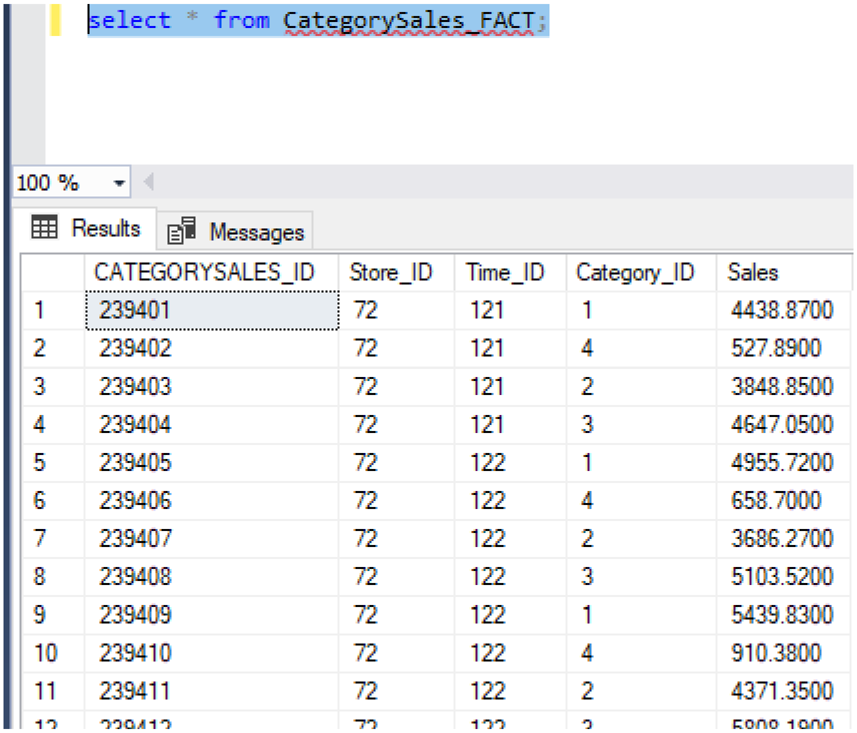
Lookup Column	Lookup Operation	Output Alias
Time_ID	<add as new column>	Time_ID

Category Lookup



Lookup Column	Lookup Operation	Output Alias
Category_ID	<add as new column>	Category_ID

Snapshot of data in CategorySales_FACT



```sql
select * from CategorySales_FACT;
```

	CATEGORYSALES_ID	Store_ID	Time_ID	Category_ID	Sales
1	239401	72	121	1	4438.8700
2	239402	72	121	4	527.8900
3	239403	72	121	2	3848.8500
4	239404	72	121	3	4647.0500
5	239405	72	122	1	4955.7200
6	239406	72	122	4	658.7000
7	239407	72	122	2	3686.2700
8	239408	72	122	3	5103.5200
9	239409	72	122	1	5439.8300
10	239410	72	122	4	910.3800
11	239411	72	122	2	4371.3500

5.3.7 Table Structure for Data Marts

Product Sales Data Mart Structure

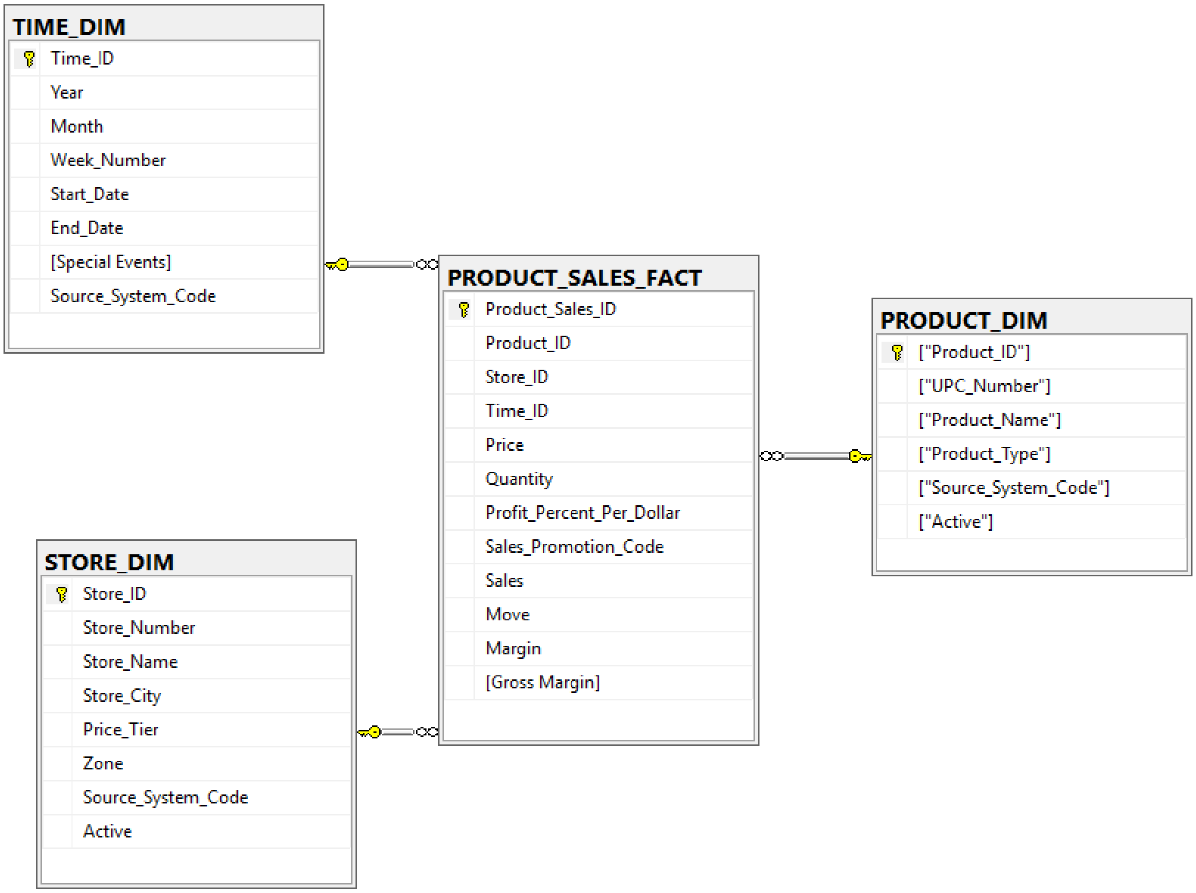


Figure 14 - Table Structure for Product Sales Data Mart

5.2.3 Table Structure for Category Sales Data Mart

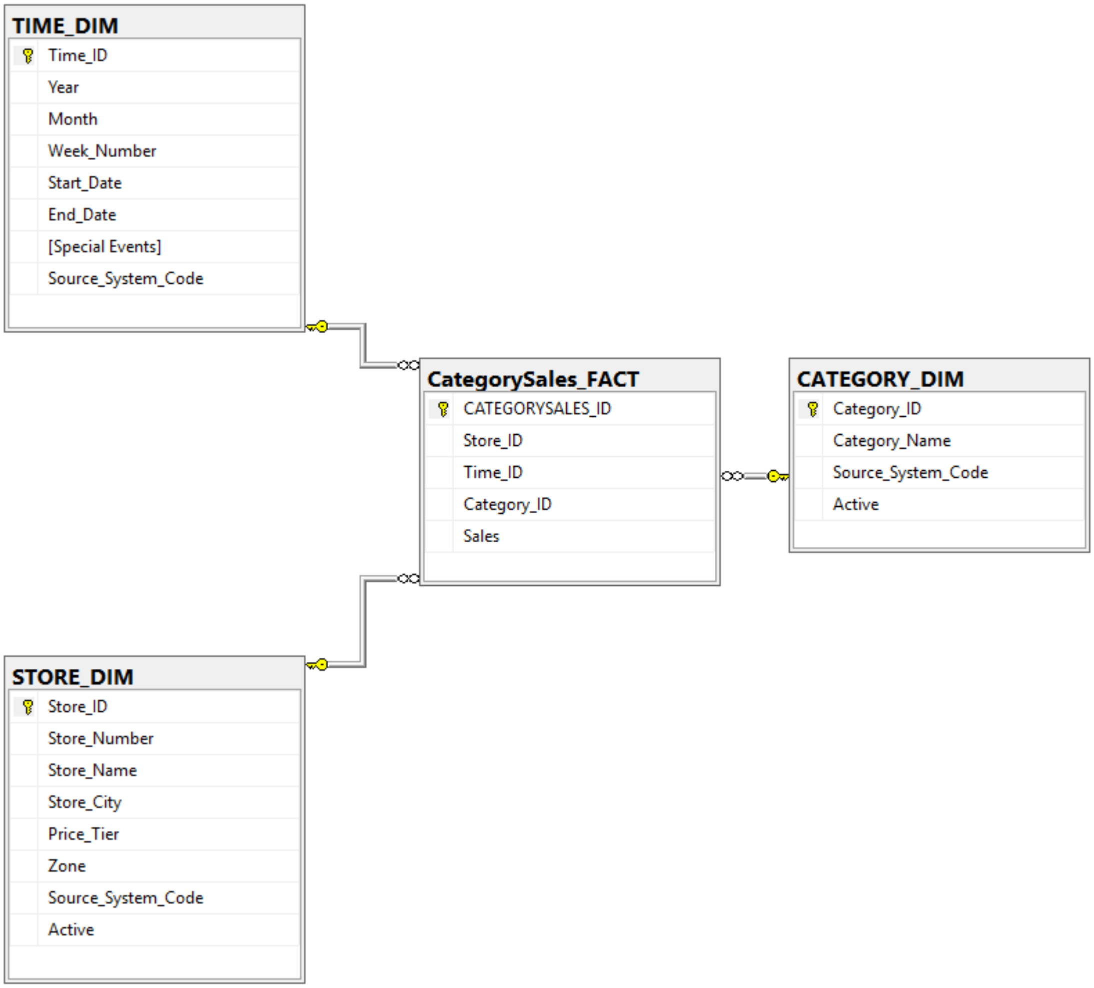
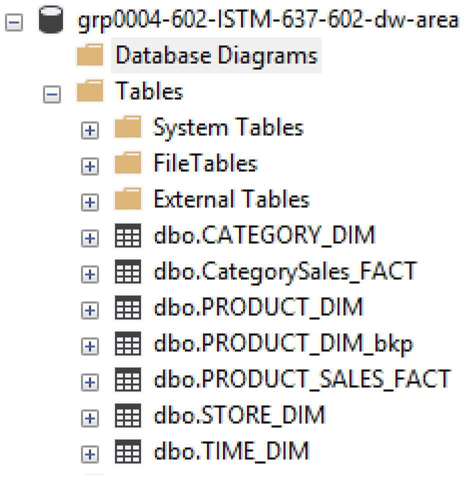


Figure 15 - Table Structure for CategorySales Data Mart

5.3 Snapshot of Data Warehouse Area



- grp0004-602-ISTM-637-602-dw-area
  - Database Diagrams
  - Tables
    - System Tables
    - FileTables
    - External Tables
    - dbo.CATEGORY_DIM
    - dbo.CategorySales_FACT
    - dbo.PRODUCT_DIM
    - dbo.PRODUCT_DIM_bkp
    - dbo.PRODUCT_SALES_FACT
    - dbo.STORE_DIM
    - dbo.TIME_DIM

Figure 16 - Snapshot for Data Warehouse Area

5.4 SQL Queries for Staging and Data Warehouse area

5.4.1 Staging

WEEK_DECODE

```
CREATE TABLE [dbo].[WEEK_DECODE](
        [Week] [float] NULL,
        [Start] [datetime] NULL,
        [End] [datetime] NULL,
        [Special Events] [nvarchar](255) NULL
)
```

Store

```
CREATE TABLE [dbo].[Store](
        [Store] [numeric](18, 0) NULL,
        [City] [nvarchar](255) NULL,
        [Price Tier] [nvarchar](255) NULL,
        [Zone] [numeric](18, 0) NULL,
        [Zip Code] [numeric](5, 0) NULL,
        [Address] [nvarchar](255) NULL
)
```

CategoryDIMStaging

```
CREATE TABLE [dbo].[CategoryDIMStaging](
        [Category_Name] [nvarchar](255) NULL,
        [Sales] [numeric](18, 4) NULL,
	["STORE"] [numeric](18, 0) NULL,
	["DATE"] [varchar](50) NULL,
	["WEEK"] [numeric](18, 0) NULL
)
```

CCOUNT

```
CREATE TABLE [dbo].[CCOUNT](
	["STORE"] [numeric](18, 0) NOT NULL,
	["DATE"] [varchar](50) NULL,
	["FROZEN"] [numeric](18, 4) NULL,
	["MEAT"] [numeric](18, 4) NULL,
	["FISH"] [numeric](18, 4) NULL,
	["DAIRY"] [numeric](18, 4) NULL,
	["WEEK"] [numeric](18, 0) NULL
)
```

CCOUNT_Clean_STG

```
CREATE TABLE [dbo].[CCOUNT_Clean_STG](
	["STORE"] [numeric](18, 0) NOT NULL,
	["DATE"] [varchar](50) NULL,
	["DAIRY"] [numeric](18, 4) NULL,
	["FROZEN"] [numeric](18, 4) NULL,
	["MEAT"] [numeric](18, 4) NULL,
	["FISH"] [numeric](18, 4) NULL,
	["WEEK"] [numeric](18, 0) NULL
)
```

Movement_WBJC

```
CREATE TABLE [dbo].[Movement_WBJC](
	["STORE"] [varchar](50) NULL,
	["UPC"] [varchar](50) NULL,
	["WEEK"] [varchar](50) NULL,
	["MOVE"] [varchar](50) NULL,
	["QTY"] [varchar](50) NULL,
	["PRICE"] [varchar](50) NULL,
	["SALE"] [varchar](50) NULL,
	["PROFIT"] [varchar](50) NULL,
	["OK"] [varchar](50) NULL
)
```

Movement_WCIG

```
CREATE TABLE [dbo].[Movement_WCIG](
	["STORE"] [varchar](50) NULL,
	["UPC"] [varchar](50) NULL,
	["WEEK"] [varchar](50) NULL,
	["MOVE"] [varchar](50) NULL,
	["QTY"] [varchar](50) NULL,
	["PRICE"] [varchar](50) NULL,
	["SALE"] [varchar](50) NULL,
	["PROFIT"] [varchar](50) NULL,
	["OK"] [varchar](50) NULL
)
```

Movement_WCSO

```
CREATE TABLE [dbo].[Movement_WCSO](
	["STORE"] [varchar](50) NULL,
	["UPC"] [varchar](50) NULL,
	["WEEK"] [varchar](50) NULL,
	["MOVE"] [varchar](50) NULL,
	["QTY"] [varchar](50) NULL,
	["PRICE"] [varchar](50) NULL,
	["SALE"] [varchar](50) NULL,
	["PROFIT"] [varchar](50) NULL,
	["OK"] [varchar](50) NULL
)
```

Movement_WSDR

```
CREATE TABLE [dbo].[Movement_WSDR](
	[STORE] [varchar](50) NULL,
	[UPC] [varchar](50) NULL,
	[WEEK] [varchar](50) NULL,
	[MOVE] [varchar](50) NULL,
	[QTY] [varchar](50) NULL,
	[PRICE] [varchar](50) NULL,
	[SALE] [varchar](50) NULL,
	[PROFIT] [varchar](50) NULL,
	[OK] [varchar](50) NULL
)
```

Movement_WTNA

```
CREATE TABLE [dbo].[Movement_WTNA](
	["STORE"] [varchar](50) NULL,
	["UPC"] [varchar](50) NULL,
	["WEEK"] [varchar](50) NULL,
	["MOVE"] [varchar](50) NULL,
	["QTY"] [varchar](50) NULL,
	["PRICE"] [varchar](50) NULL,
	["SALE"] [varchar](50) NULL,
	["PROFIT"] [varchar](50) NULL,
	["OK"] [varchar](50) NULL
)
```

ProductSales_STG

```
CREATE TABLE [dbo].[ProductSales_STG](
	[ProductSalesID] [int] NULL,
	[UPC] [numeric](18, 0) NULL,
	[STORE] [numeric](18, 0) NULL,
	[WEEK] [numeric](18, 0) NULL,
	[MOVE] [numeric](18, 0) NULL,
	[QTY] [numeric](18, 0) NULL,
	[PRICE] [float] NULL,
	["SALE"] [varchar](50) NULL,
	["PROFIT"] [varchar](50) NULL,
	[Sales] [numeric](38, 6) NULL,
	[Margin] [numeric](38, 6) NULL,
	[Gross Margin] [numeric](38, 6) NULL,
	[Product_Type] [nvarchar](20) NULL
)
```

Store

```
CREATE TABLE [dbo].[Store](
	[Store] [numeric](18, 0) NULL,
	[City] [nvarchar](255) NULL,
	[Price Tier] [nvarchar](255) NULL,
	[Zone] [numeric](18, 0) NULL,
	[Zip Code] [numeric](5, 0) NULL,
	[Address] [nvarchar](255) NULL
)
```

Store_DIM_TEMP

```
CREATE TABLE [dbo].[Store_DIM_TEMP](
        [Store_ID] [int] NULL,
        [Store_Number] [numeric](18, 0) NULL,
        [Store_Name] [varchar](50) NULL,
        [Store_City] [nvarchar](255) NULL,
        [Price_Tier] [nvarchar](255) NULL,
        [Zone] [numeric](18, 0) NULL,
        [Source_System_Code] [nvarchar](40) NULL,
        [Active] [int] NULL
)
```

UPCBJC

```
CREATE TABLE [dbo].[UPCBJC](
        ["COM_CODE"] [varchar](50) NULL,
        ["UPC"] [varchar](50) NULL,
        ["DESCRIP"] [varchar](50) NULL,
        ["SIZE"] [varchar](50) NULL,
        ["CASE"] [varchar](50) NULL,
        ["NITEM"] [varchar](50) NULL
)
```

UPCSDR

```
CREATE TABLE [dbo].[UPCSDR](
        ["COM_CODE"] [varchar](50) NULL,
        ["UPC"] [varchar](50) NULL,
        ["DESCRIP"] [varchar](50) NULL,
        ["SIZE"] [varchar](50) NULL,
        ["CASE"] [varchar](50) NULL,
        ["NITEM"] [varchar](50) NULL
)
```

UPCCIG

```
CREATE TABLE [dbo].[UPCCIG](
        ["COM_CODE"] [varchar](50) NULL,
        ["UPC"] [varchar](50) NULL,
        ["DESCRIP"] [varchar](50) NULL,
        ["SIZE"] [varchar](50) NULL,
        ["CASE"] [varchar](50) NULL,
        ["NITEM"] [varchar](50) NULL
)
```

UPCCSO

```
CREATE TABLE [dbo].[UPCCSO](
        ["COM_CODE"] [varchar](50) NULL,
        ["UPC"] [varchar](50) NULL,
        ["DESCRIP"] [varchar](50) NULL,
        ["SIZE"] [varchar](50) NULL,
        ["CASE"] [varchar](50) NULL,
        ["NITEM"] [varchar](50) NULL
)
```

UPCTNA

```
CREATE TABLE [dbo].[UPCTNA](
        ["COM_CODE"] [varchar](50) NULL,
        ["UPC"] [varchar](50) NULL,
        ["DESCRIP"] [varchar](50) NULL,
        ["SIZE"] [varchar](50) NULL,
        ["CASE"] [varchar](50) NULL,
        ["NITEM"] [varchar](50) NULL
)
```

DATAWAREHOUSE TABLES

TIME_DIM

```
CREATE TABLE [dbo].[TIME_DIM](
        [Time_ID] [int] NOT NULL,
        [Year] [int] NULL,
        [Month] [int] NULL,
        [Week_Number] [numeric](18, 0) NOT NULL,
        [Start_Date] [date] NULL,
        [End_Date] [date] NULL,
        [Special Events] [nvarchar](255) NULL,
        [Source_System_Code] [nvarchar](20) NULL,
PRIMARY KEY CLUSTERED
(
        [Time_ID] ASC
)
)
```

STORE_DIM

```
CREATE TABLE [dbo].[STORE_DIM](
        [Store_ID] [int] NOT NULL,
	[Store_Number] [numeric](18, 0) NOT NULL,
	[Store_Name] [varchar](50) NULL,
	[Store_City] [nvarchar](255) NULL,
	[Price_Tier] [nvarchar](255) NULL,
	[Zone] [numeric](18, 0) NULL,
	[Source_System_Code] [nvarchar](40) NULL,
	[Active] [int] NOT NULL,
PRIMARY KEY CLUSTERED
(
	[Store_ID] ASC
))
```

PRODUCT_DIM

```
CREATE TABLE [dbo].[PRODUCT_DIM](
	["Product_ID"] [int] NOT NULL,
	["UPC_Number"] [numeric](18, 0) NULL,
	["Product_Name"] [varchar](200) NULL,
	["Product_Type"] [nvarchar](50) NULL,
	["Source_System_Code"] [nvarchar](40) NULL,
	["Active"] [int] NULL,
PRIMARY KEY CLUSTERED
(
	["Product_ID"] ASC
)
)
```

CATEGORY_DIM

```
CREATE TABLE [dbo].[PRODUCT_DIM](
	["Product_ID"] [int] NOT NULL,
	["UPC_Number"] [numeric](18, 0) NULL,
	["Product_Name"] [varchar](200) NULL,
	["Product_Type"] [nvarchar](50) NULL,
	["Source_System_Code"] [nvarchar](40) NULL,
	["Active"] [int] NULL,
PRIMARY KEY CLUSTERED
(
	["Product_ID"] ASC
)
)
```

PRODUCT_SALES_FACT

```
CREATE TABLE [dbo].[PRODUCT_SALES_FACT](
        [Product_Sales_ID] [int] NOT NULL,
        [Product_ID] [int] NOT NULL,
        [Store_ID] [int] NOT NULL,
        [Time_ID] [int] NOT NULL,
        [Price] [float] NULL,
        [Quantity] [numeric](18, 0) NULL,
        [Profit_Percent_Per_Dollar] [int] NULL,
        [Sales_Promotion_Code] [varchar](50) NULL,
        [Sales] [numeric](38, 6) NULL,
        [Move] [numeric](18, 0) NULL,
        [Margin] [numeric](38, 6) NULL,
        [Gross Margin] [numeric](38, 6) NULL,
 CONSTRAINT [PK_PRODUCT_SALES_FACT] PRIMARY KEY CLUSTERED
(
        [Product_Sales_ID] ASC
),
CONSTRAINT [FK_PRODUCT_SALES_FACT_1] FOREIGN KEY([Product_ID])
REFERENCES [dbo].[PRODUCT_DIM] (["Product_ID"])
ON UPDATE CASCADE
ON DELETE CASCADE,
CONSTRAINT [FK_PRODUCT_SALES_FACT_2] FOREIGN KEY([Store_ID])
REFERENCES [dbo].[STORE_DIM] ([Store_ID])
ON UPDATE CASCADE
ON DELETE CASCADE,
ALTER TABLE [dbo].[PRODUCT_SALES_FACT]  WITH CHECK ADD  CONSTRAINT
[FK_PRODUCT_SALES_FACT_3] FOREIGN KEY([Time_ID])
REFERENCES [dbo].[TIME_DIM] ([Time_ID])
ON UPDATE CASCADE
ON DELETE CASCADE
)
```

CATEGORYSALES_FACT

```
CREATE TABLE [dbo].[CategorySales_FACT](
        [CATEGORYSALES_ID] [int] NOT NULL PRIMARY KEY,
        [Store_ID] [int] NOT NULL FOREIGN KEY REFERENCES [STORE_DIM]([Store_ID]),
        [Time_ID] [int]NOT NULL FOREIGN KEY REFERENCES [TIME_DIM]([Time_ID]),
        [Category_ID] [numeric](18, 0) NOT NULL FOREIGN KEY REFERENCES
[CATEGORY_DIM]([Category_ID]),
        [Sales] [numeric](18, 4) NULL
);
```

6. Business Intelligence (BI) Reports

6.1 Reporting Plan

Business intelligence enables user to store data that represents some business processes, to organize the data in a useful manner, and to present the data as meaningful information to solve business questions. To deliver satisfactory BI reports, good planning is a key part.

In our report plan, three parts will be addressed. According to each specific business question, the target reports are specified, and the reasons have been given. Then, based on the report type, all the mappings from the independent data marts to the report attributes have been determined. Besides, all report templates are well described.

SSRS, SSAS, SSAS + SSRS and ReportBuilder3.0 are used in the report building as described below:

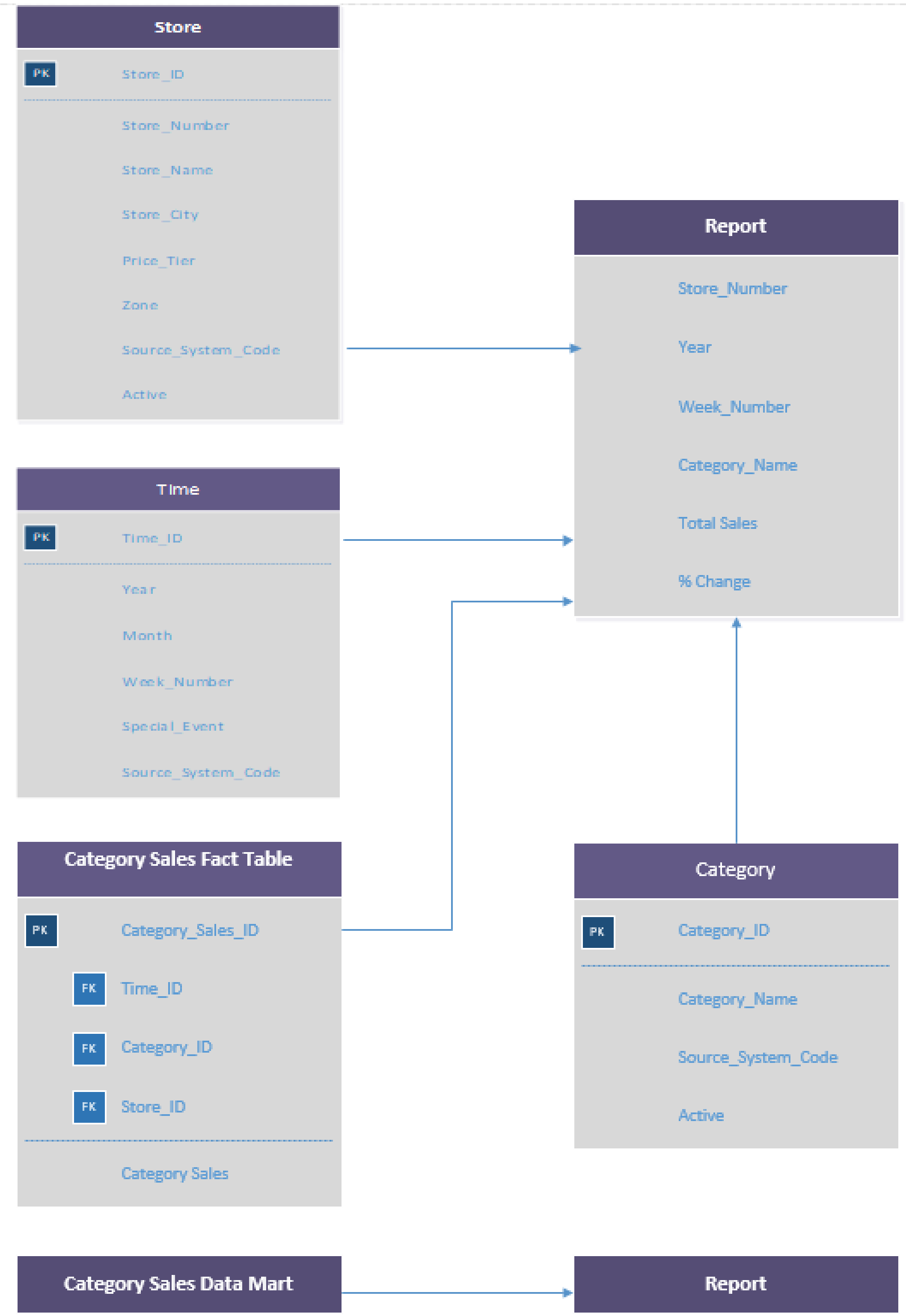
Table 20. Report Tool for Question Number

Reporting Tool	Question Number
SSRS	BQ -1, BQ - 2
SSAS	BQ -5
SSAS+SSRS	BQ -3
Report Builder 3.0	BQ-4

The details about mapping from data marts to designed table of each business question are listed below.

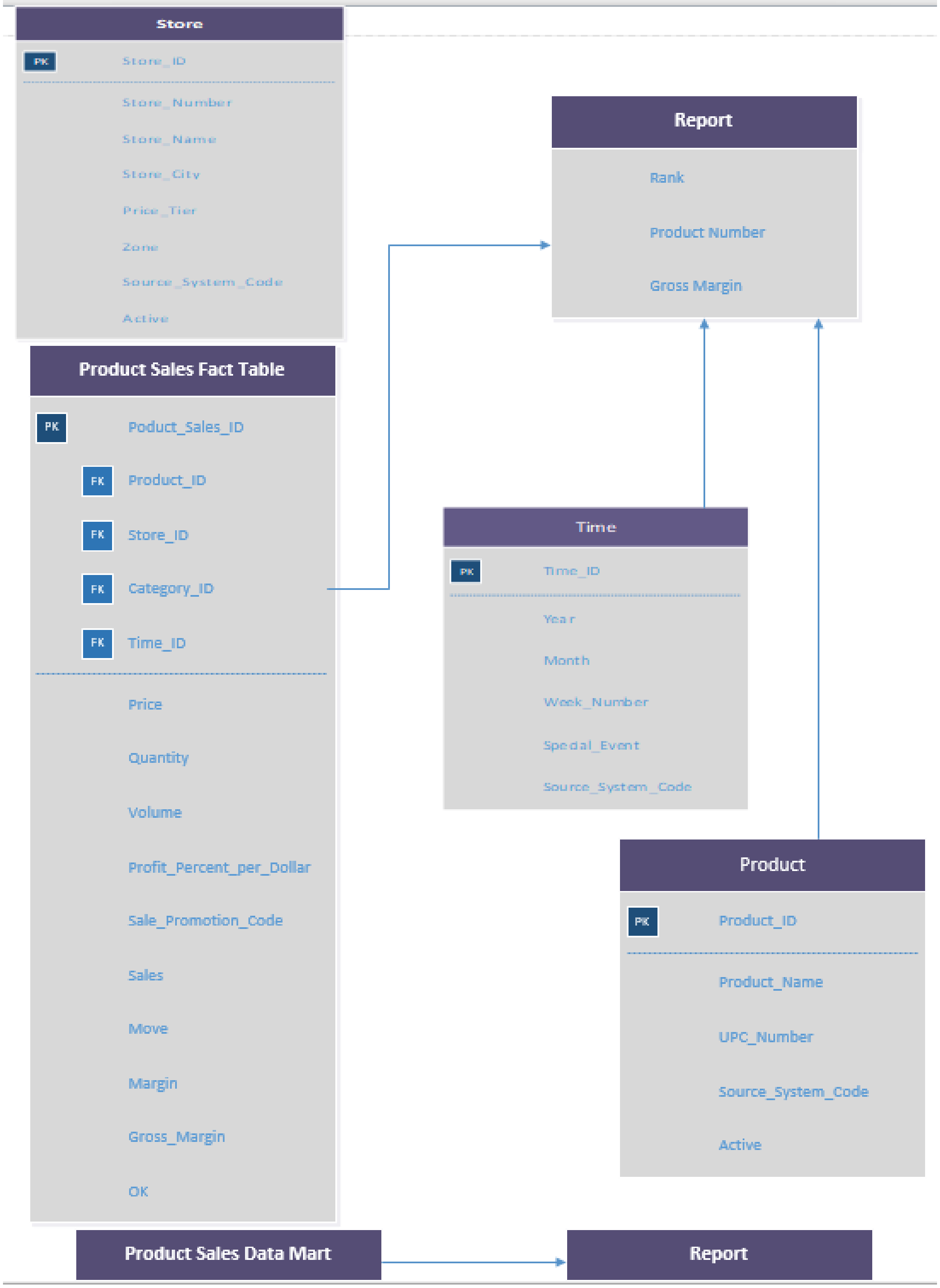
BQ-1: What is the trend of weekly change in sales of Frozen Food, Meat, Fish and Dairy till current week in 1997 for store 2?

Report generate from SSRS


Store
PK Store_ID
Store_Number
Store_Name
Store_City
Price_Tier
Zone
Source_System_Code
Active
Report
Store_Number
Year
Week_Number
Category_Name
Total Sales
% Change
Time
PK Time_ID
Year
Month
Week_Number
Special_Event
Source_System_Code
Category Sales Fact Table
PK Category_Sales_ID
FK Time_ID
FK Category_ID
FK Store_ID
Category Sales
Category
PK Category_ID
Category_Name
Source_System_Code
Active
Category Sales Data Mart
Report

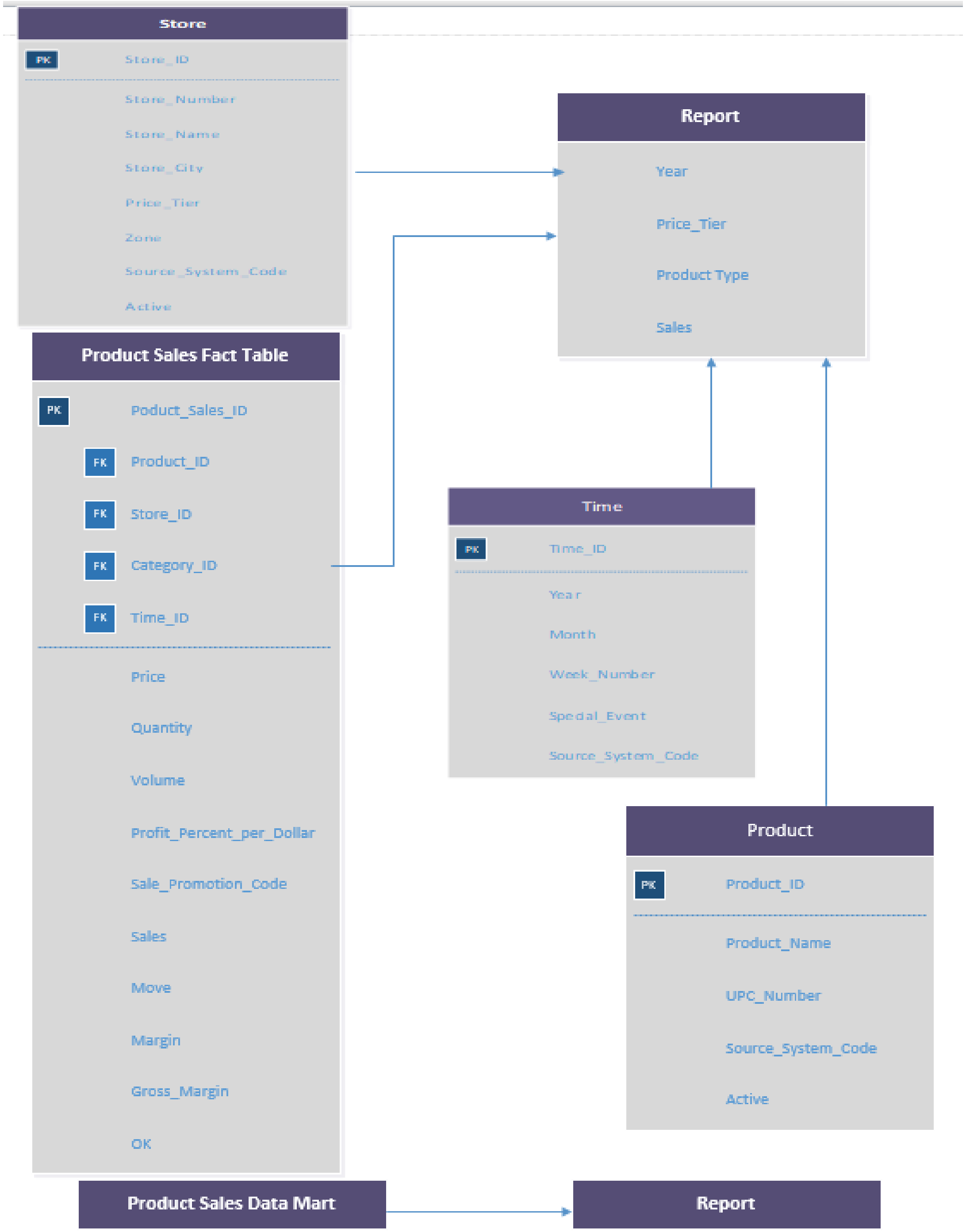
BQ-2: Which are the top 10 products with respect to profitability in Medium Tier stores w.r.t "Canned Soup" category of in year 1991?

Report generate from SSRS



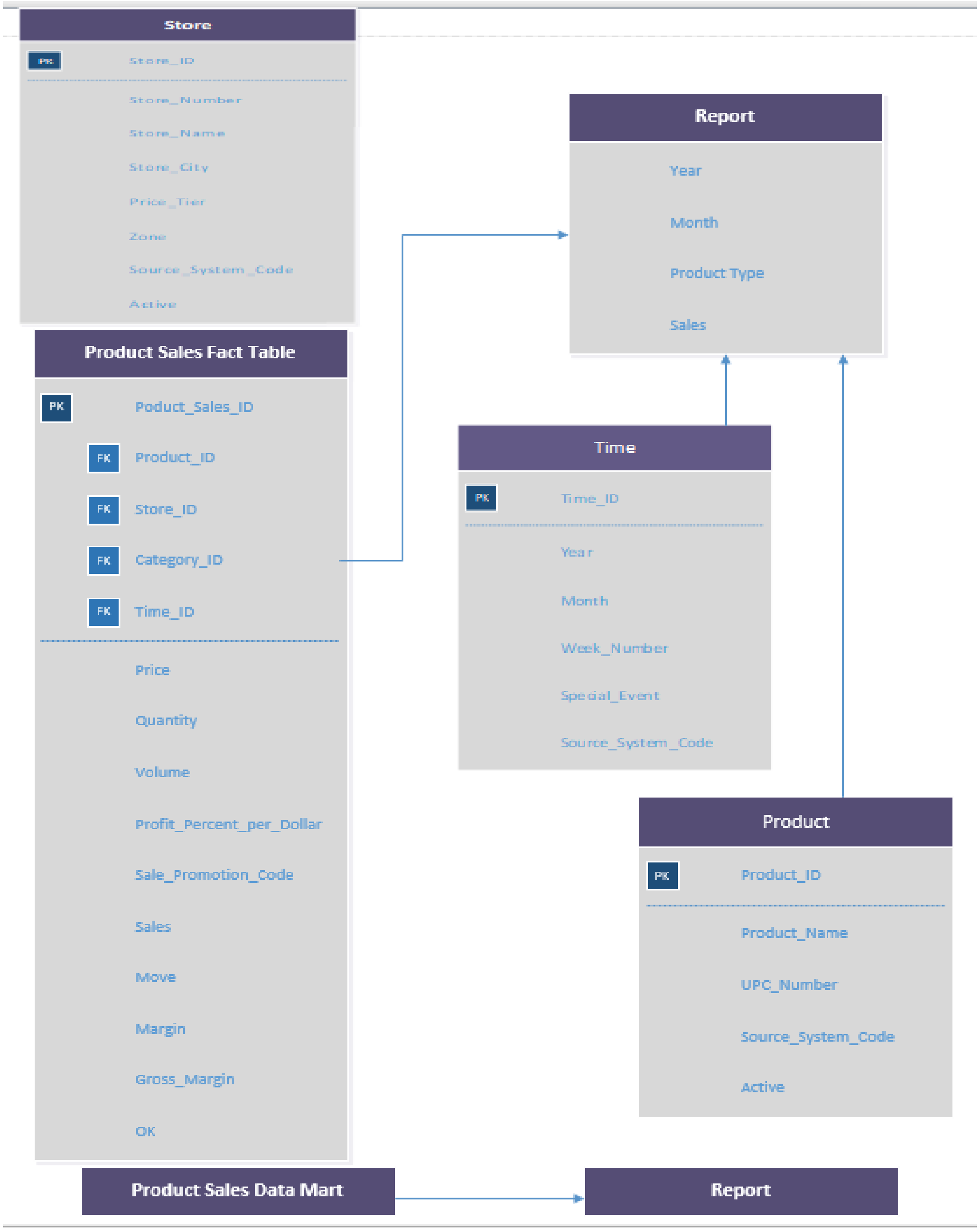
BQ-3: What is the growth trend of Soft Drinks versus Bottled Juice for year 1990 and 1991? Which of these has negative growth rate? What is the trend for bottled juice especially in Low Tier?

Report generate from SSAS+SSRS



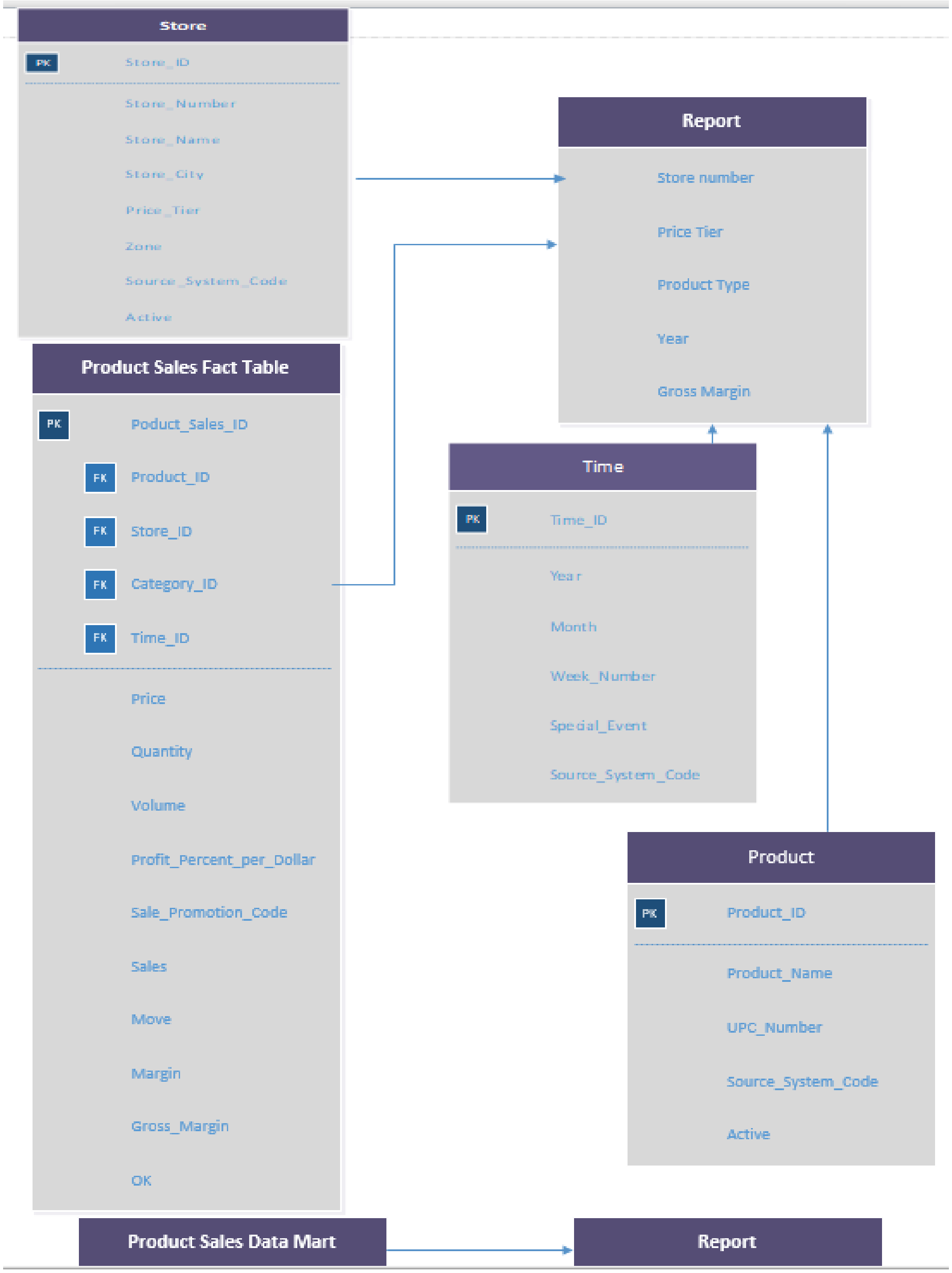
BQ-4: What effect does the time of the year have on sales of each Tuna? Determine the impact of seasonality on Average Sales of Tuna from December-1990 to September 1992?

Report generate from Report Builder 3.0



BQ-5: What is annual growth rate of cigarette margins in low tier stores? What is the decreasing trend?

Report generate from SSAS

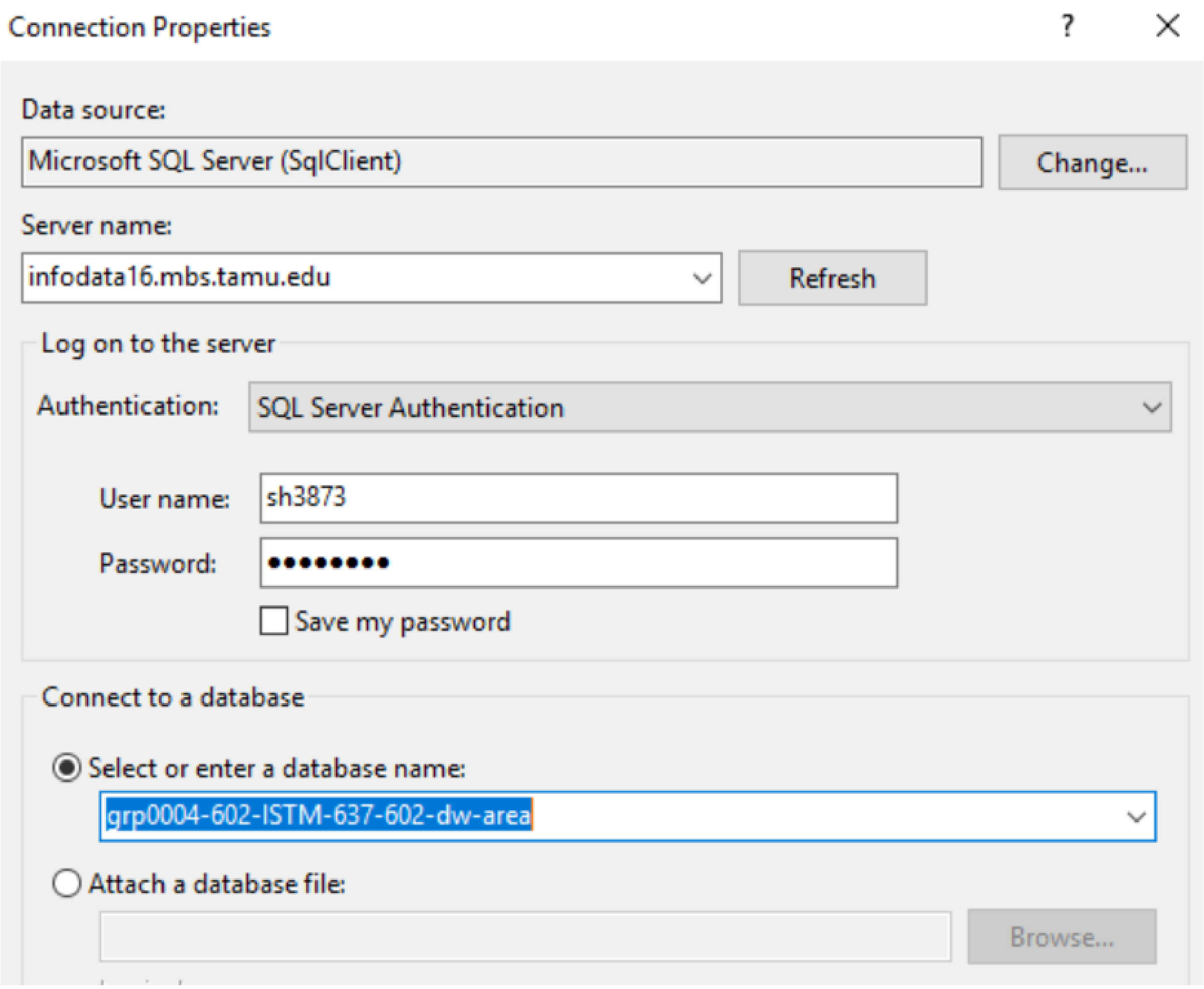


6.2 SSRS for BQ-1

Based on the report plan, SSRS is adopted to create visualization for business question 1.

BQ-1: What is the trend of weekly change in sales of Frozen Food, Meat, Fish and Dairy till current week in 1997 for store 2?

Connection to Data Source



Connection Properties

Data source:
Microsoft SQL Server (SqlClient)

Server name:
infodata16.mbs.tamu.edu

Log on to the server

Authentication: SQL Server Authentication

User name: sh3873

Password: ●●●●●●●●●

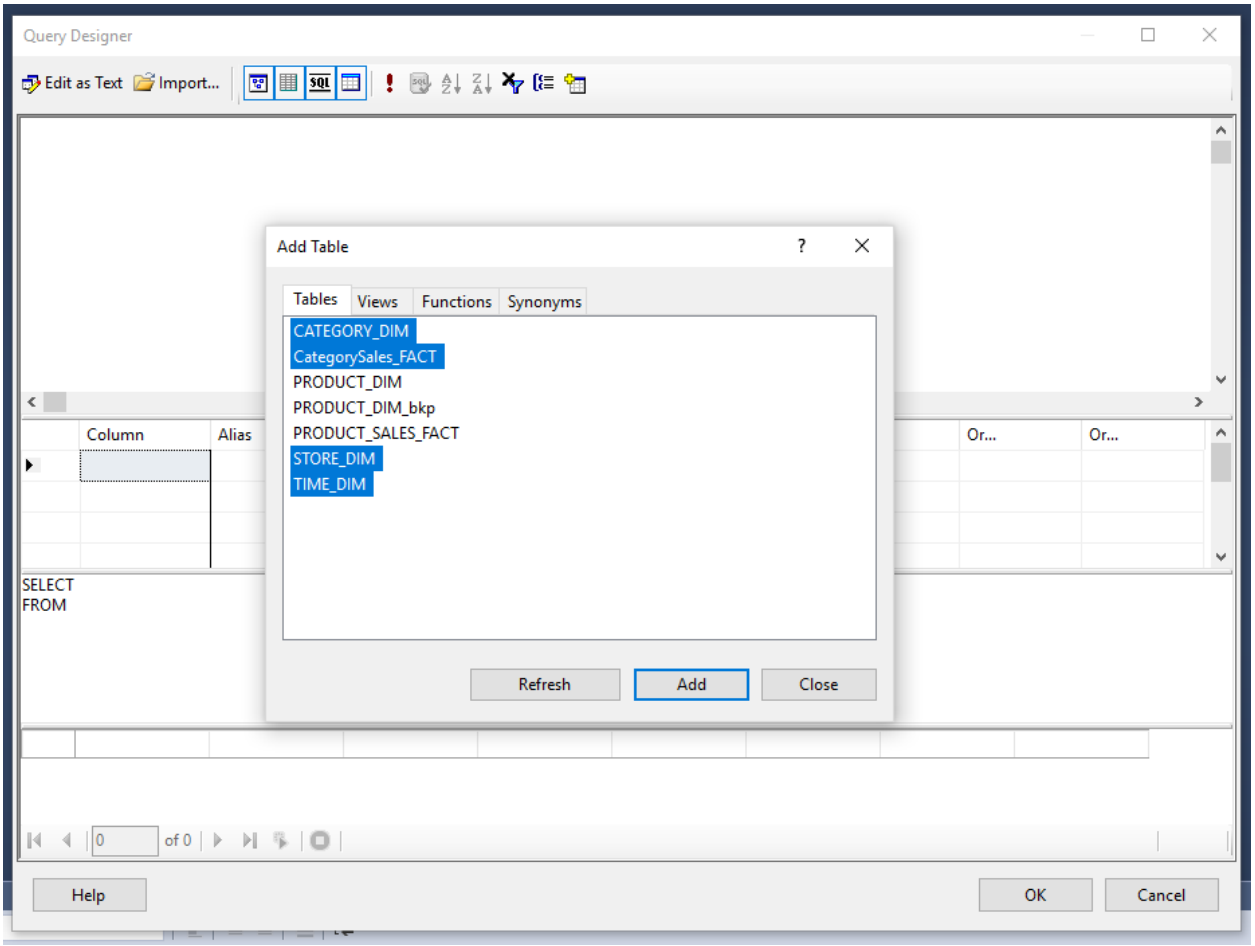
☐ Save my password

Connect to a database

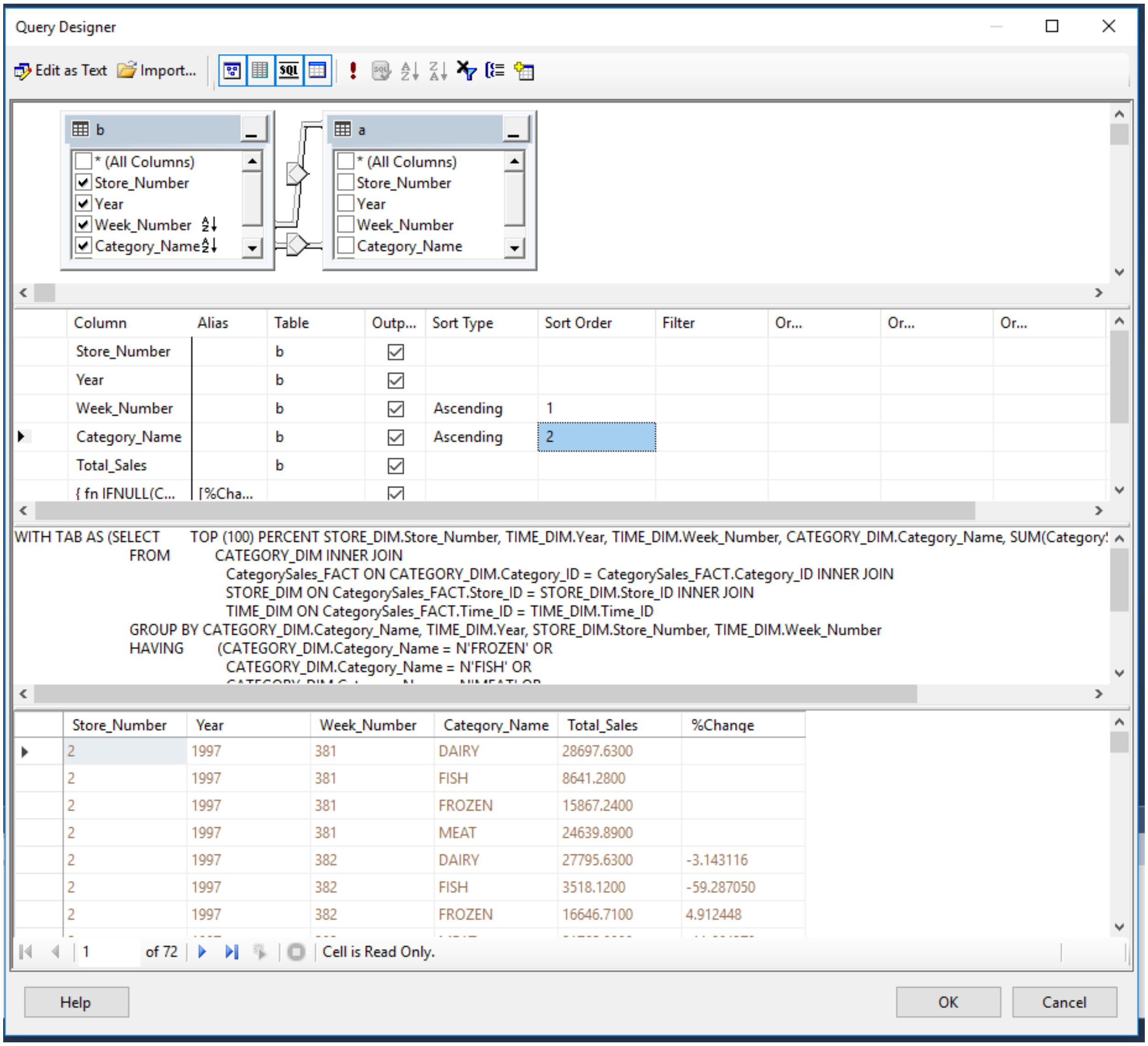
● Select or enter a database name:
grp0004-602-ISTM-637-602-dw-area

○ Attach a database file:

Query Design


Query Designer
Edit as Text
Import...
Add Table
?
Tables
Views
Functions
Synonyms
CATEGORY_DIM
CategorySales_FACT
PRODUCT_DIM
PRODUCT_DIM_bkp
PRODUCT_SALES_FACT
STORE_DIM
TIME_DIM
Refresh
Add
Close
Column
Alias
Or...
Or...
0
of 0
Help
OK
Cancel

```
SELECT
FROM
```

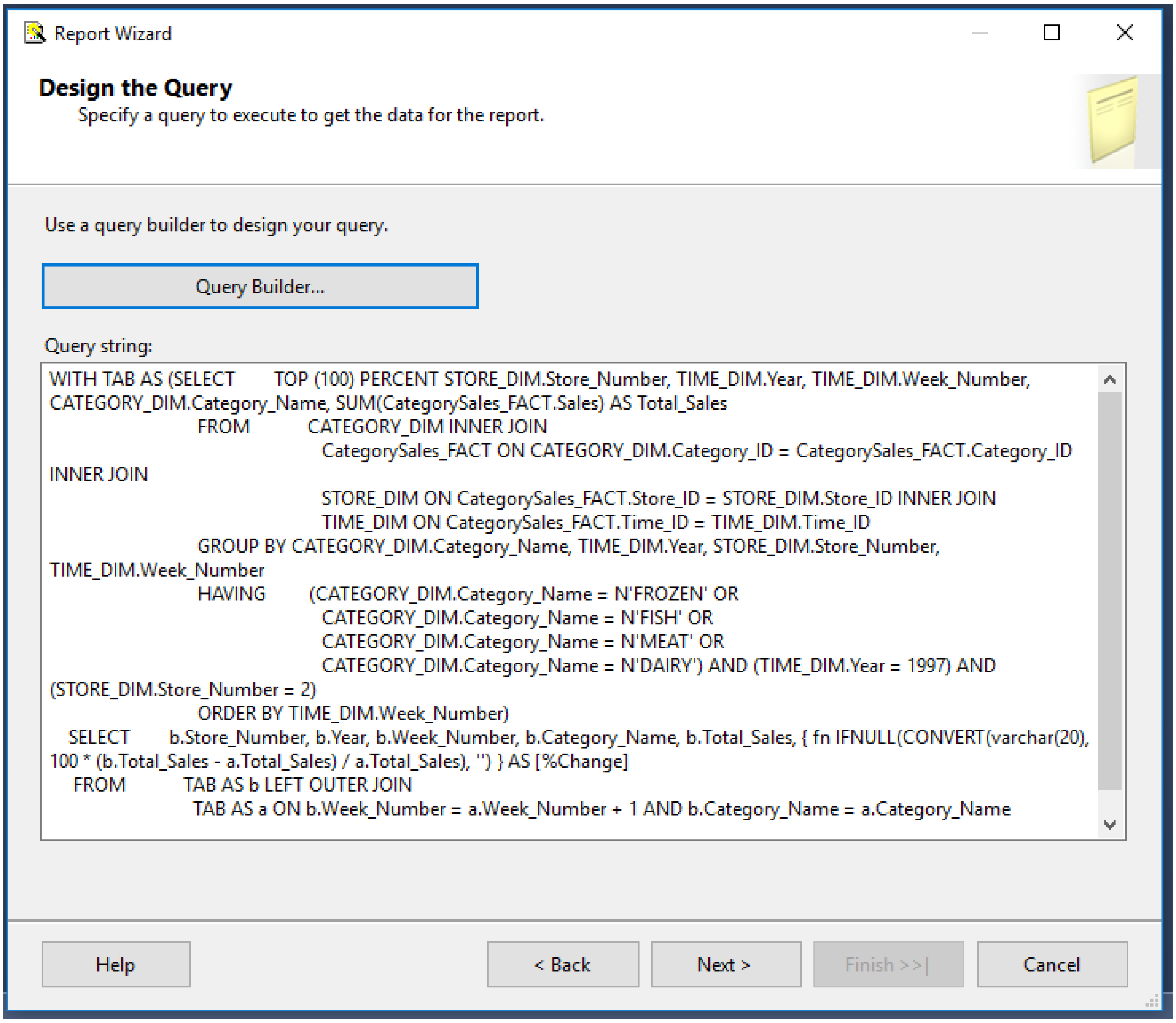


Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...	Or...	Or...
Store_Number		b	☑						
Year		b	☑						
Week_Number		b	☑	Ascending	1				
Category_Name		b	☑	Ascending	2				
Total_Sales		b	☑						
{ fn IFNULL(C...	[%Cha...		☑						

```
WITH TAB AS (SELECT      TOP (100) PERCENT STORE_DIM.Store_Number, TIME_DIM.Year, TIME_DIM.Week_Number, CATEGORY_DIM.Category_Name, SUM(Category
                FROM          CATEGORY_DIM INNER JOIN
                                   CategorySales_FACT ON CATEGORY_DIM.Category_ID = CategorySales_FACT.Category_ID INNER JOIN
                                   STORE_DIM ON CategorySales_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
                                   TIME_DIM ON CategorySales_FACT.Time_ID = TIME_DIM.Time_ID
                GROUP BY CATEGORY_DIM.Category_Name, TIME_DIM.Year, STORE_DIM.Store_Number, TIME_DIM.Week_Number
                HAVING        (CATEGORY_DIM.Category_Name = N'FROZEN' OR
                                   CATEGORY_DIM.Category_Name = N'FISH' OR
```

Store_Number	Year	Week_Number	Category_Name	Total_Sales	%Change
2	1997	381	DAIRY	28697.6300	
2	1997	381	FISH	8641.2800	
2	1997	381	FROZEN	15867.2400	
2	1997	381	MEAT	24639.8900	
2	1997	382	DAIRY	27795.6300	-3.143116
2	1997	382	FISH	3518.1200	-59.287050
2	1997	382	FROZEN	16646.7100	4.912448

1 of 72 Cell is Read Only.



Design the Query

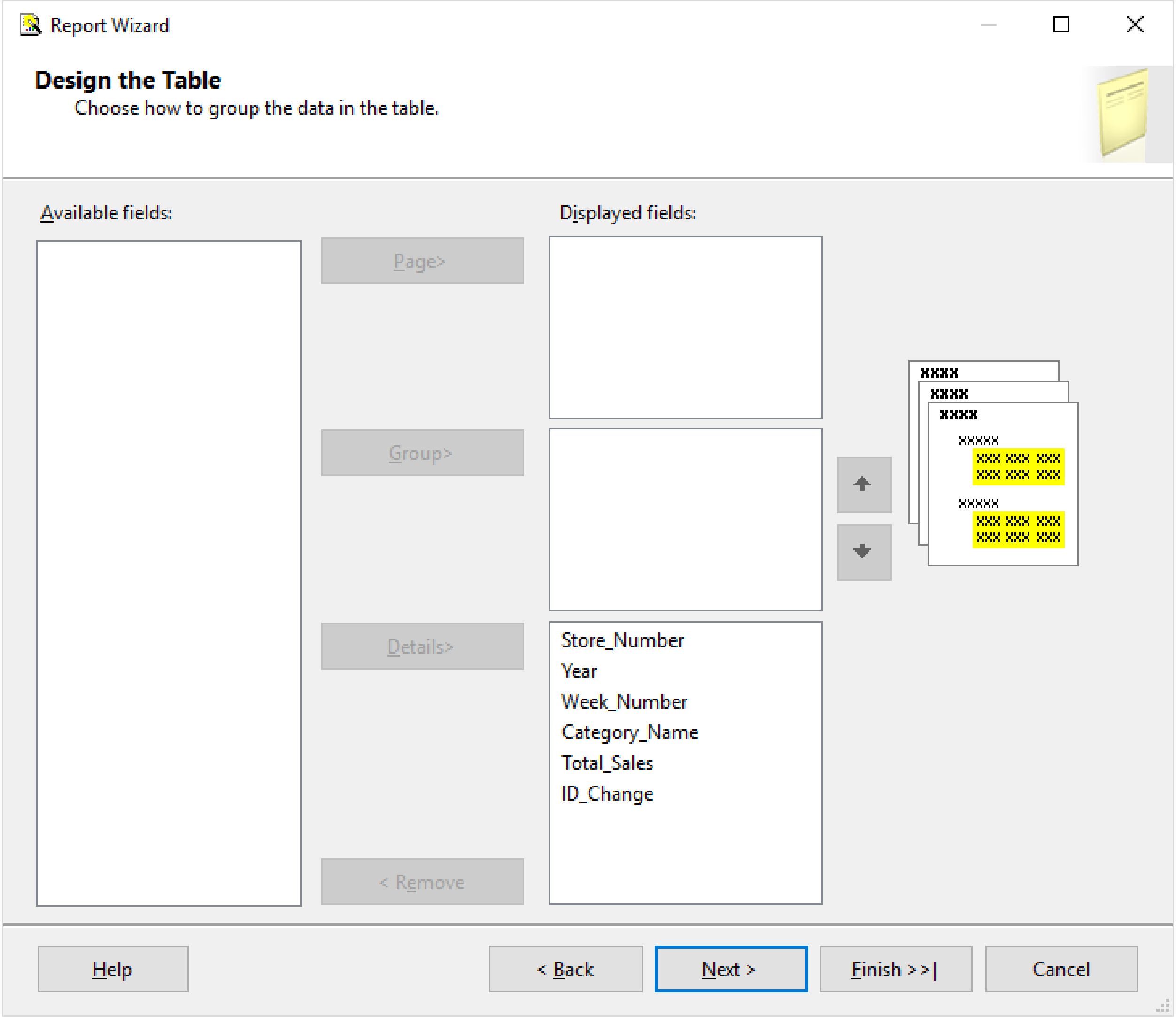
Specify a query to execute to get the data for the report.

Use a query builder to design your query.

Query string:

```
WITH TAB AS (SELECT      TOP (100) PERCENT STORE_DIM.Store_Number, TIME_DIM.Year, TIME_DIM.Week_Number,
CATEGORY_DIM.Category_Name, SUM(CategorySales_FACT.Sales) AS Total_Sales
                    FROM         CATEGORY_DIM INNER JOIN
                                   CategorySales_FACT ON CATEGORY_DIM.Category_ID = CategorySales_FACT.Category_ID
INNER JOIN
                                   STORE_DIM ON CategorySales_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
                                   TIME_DIM ON CategorySales_FACT.Time_ID = TIME_DIM.Time_ID
                    GROUP BY CATEGORY_DIM.Category_Name, TIME_DIM.Year, STORE_DIM.Store_Number,
TIME_DIM.Week_Number
                    HAVING       (CATEGORY_DIM.Category_Name = N'FROZEN' OR
                                   CATEGORY_DIM.Category_Name = N'FISH' OR
                                   CATEGORY_DIM.Category_Name = N'MEAT' OR
                                   CATEGORY_DIM.Category_Name = N'DAIRY') AND (TIME_DIM.Year = 1997) AND
(STORE_DIM.Store_Number = 2)
                    ORDER BY TIME_DIM.Week_Number)
  SELECT       b.Store_Number, b.Year, b.Week_Number, b.Category_Name, b.Total_Sales, { fn IFNULL(CONVERT(varchar(20),
100 * (b.Total_Sales - a.Total_Sales) / a.Total_Sales), '') } AS [%Change]
   FROM         TAB AS b LEFT OUTER JOIN
                   TAB AS a ON b.Week_Number = a.Week_Number + 1 AND b.Category_Name = a.Category_Name
```

Group Data in the Table



Report Wizard

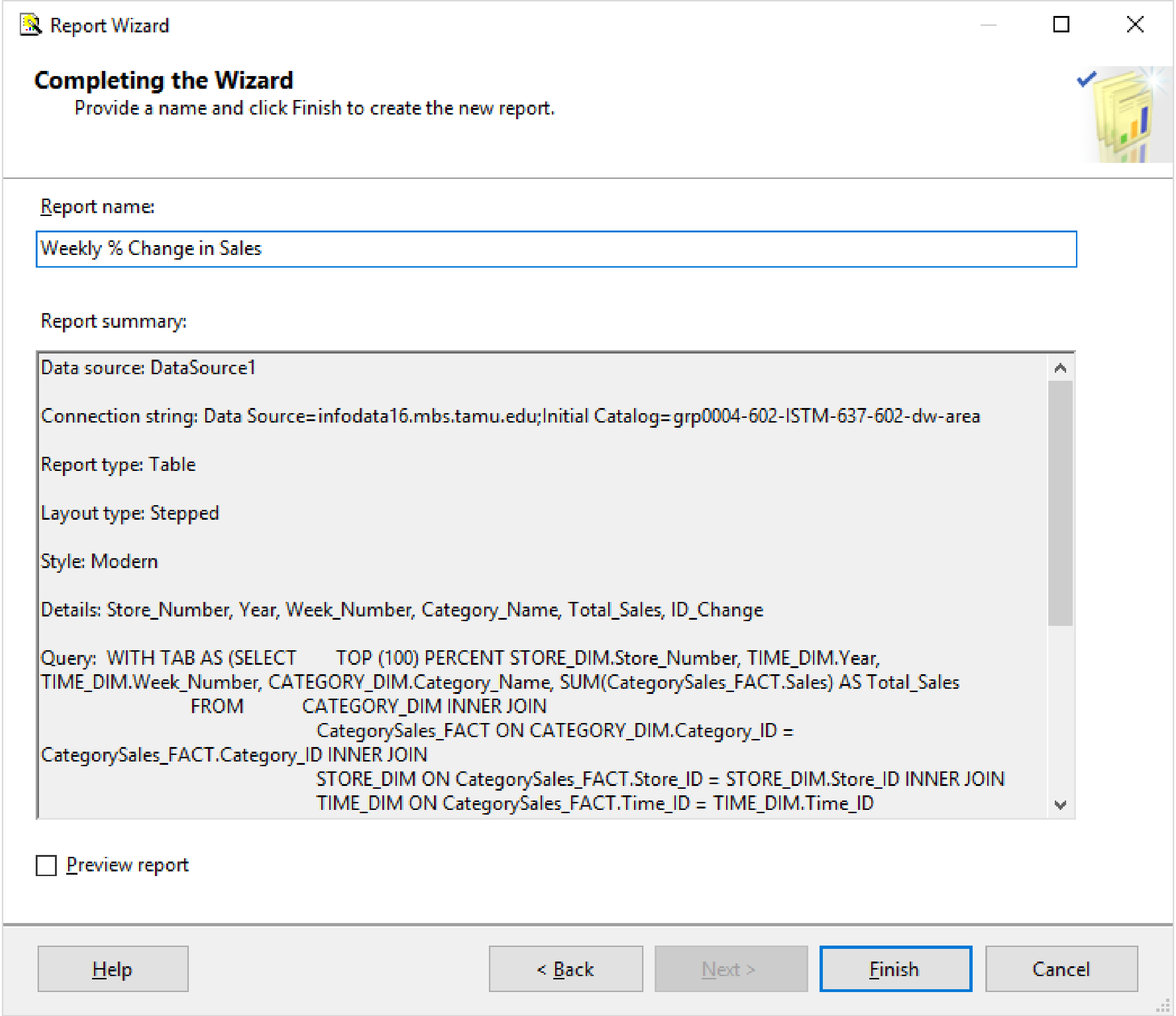
Design the Table

Choose how to group the data in the table.

Available fields:

Displayed fields:

Store_Number
Year
Week_Number
Category_Name
Total_Sales
ID_Change



Completing the Wizard

Provide a name and click Finish to create the new report.

Report name:

Weekly % Change in Sales

Report summary:

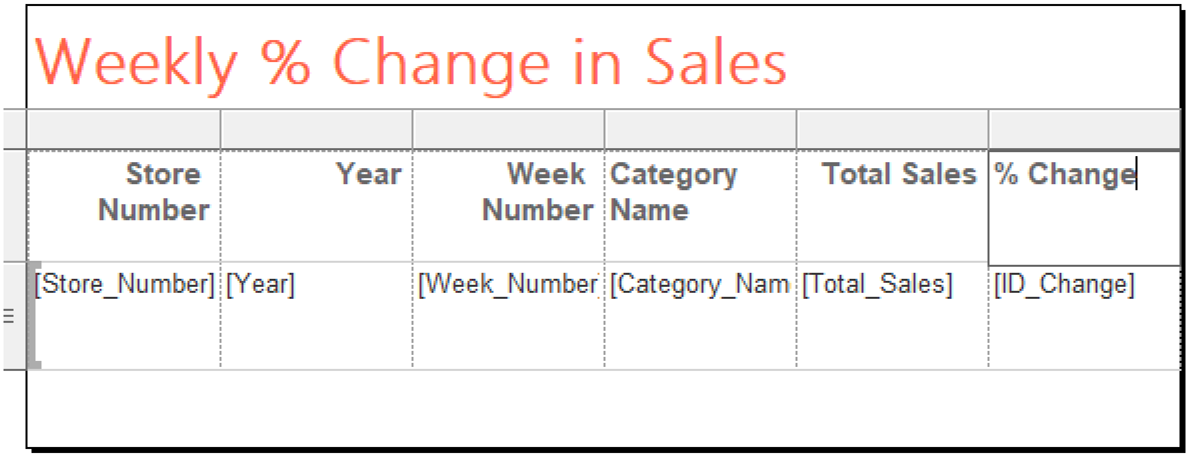
```
Data source: DataSource1

Connection string: Data Source=infodata16.mbs.tamu.edu;Initial Catalog=grp0004-602-ISTM-637-602-dw-area

Report type: Table

Layout type: Stepped

Style: Modern

Details: Store_Number, Year, Week_Number, Category_Name, Total_Sales, ID_Change

Query:  WITH TAB AS (SELECT       TOP (100) PERCENT STORE_DIM.Store_Number, TIME_DIM.Year,
TIME_DIM.Week_Number, CATEGORY_DIM.Category_Name, SUM(CategorySales_FACT.Sales) AS Total_Sales
                      FROM          CATEGORY_DIM INNER JOIN
                                      CategorySales_FACT ON CATEGORY_DIM.Category_ID =
CategorySales_FACT.Category_ID INNER JOIN
                                      STORE_DIM ON CategorySales_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
                                      TIME_DIM ON CategorySales_FACT.Time_ID = TIME_DIM.Time_ID
```

☐ Preview report

SQL Query

```
WITH TAB AS (SELECT        TOP (100) PERCENT STORE_DIM.Store_Number, TIME_DIM.Year, 
TIME_DIM.Week_Number, CATEGORY_DIM.Category_Name, SUM(CategorySales_FACT.Sales) AS 
Total_Sales
                               FROM            CATEGORY_DIM INNER JOIN
                                                         CategorySales_FACT ON 
CATEGORY_DIM.Category_ID = CategorySales_FACT.Category_ID INNER JOIN
                                                         STORE_DIM ON 
CategorySales_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
                                                         TIME_DIM ON 
CategorySales_FACT.Time_ID = TIME_DIM.Time_ID
                               GROUP BY CATEGORY_DIM.Category_Name, TIME_DIM.Year, 
STORE_DIM.Store_Number, TIME_DIM.Week_Number
                               HAVING         (CATEGORY_DIM.Category_Name = 
N'FROZEN' OR
                                                         CATEGORY_DIM.Category_Name 
= N'FISH' OR
                                                         CATEGORY_DIM.Category_Name 
= N'MEAT' OR
                                                         CATEGORY_DIM.Category_Name 
= N'DAIRY') AND (TIME_DIM.Year = 1997) AND (STORE_DIM.Store_Number = 2)
                               ORDER BY TIME_DIM.Week_Number)
    SELECT        b.Store_Number, b.Year, b.Week_Number, b.Category_Name, 
b.Total_Sales, { fn IFNULL(CONVERT(varchar(20), 100 * (b.Total_Sales - 
a.Total_Sales) / a.Total_Sales), '') } AS [%Change]
     FROM            TAB AS b LEFT OUTER JOIN
                              TAB AS a ON b.Week_Number = a.Week_Number + 1 AND 
b.Category_Name = a.Category_Name
     ORDER BY b.Week_Number
```

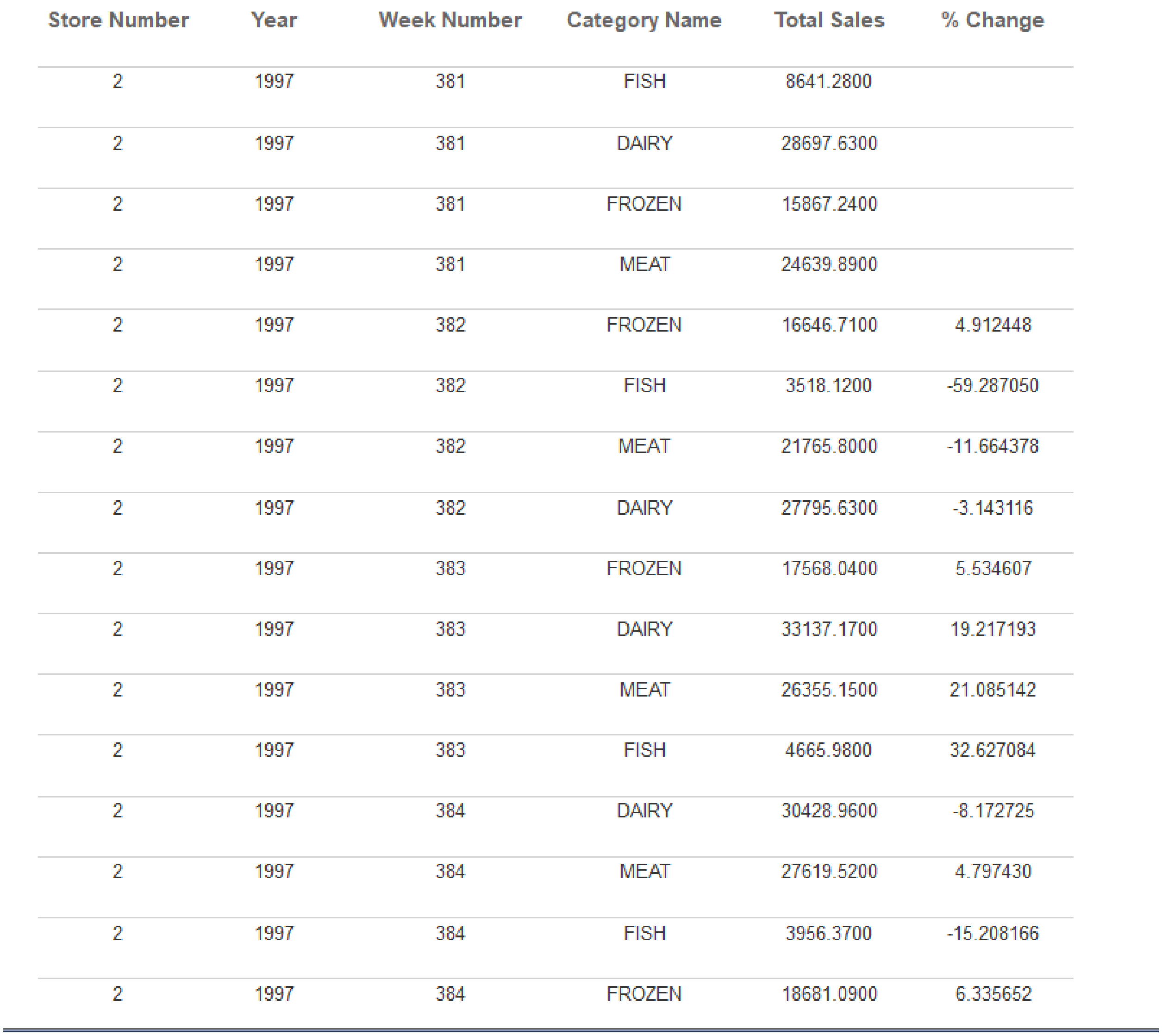
Final Report



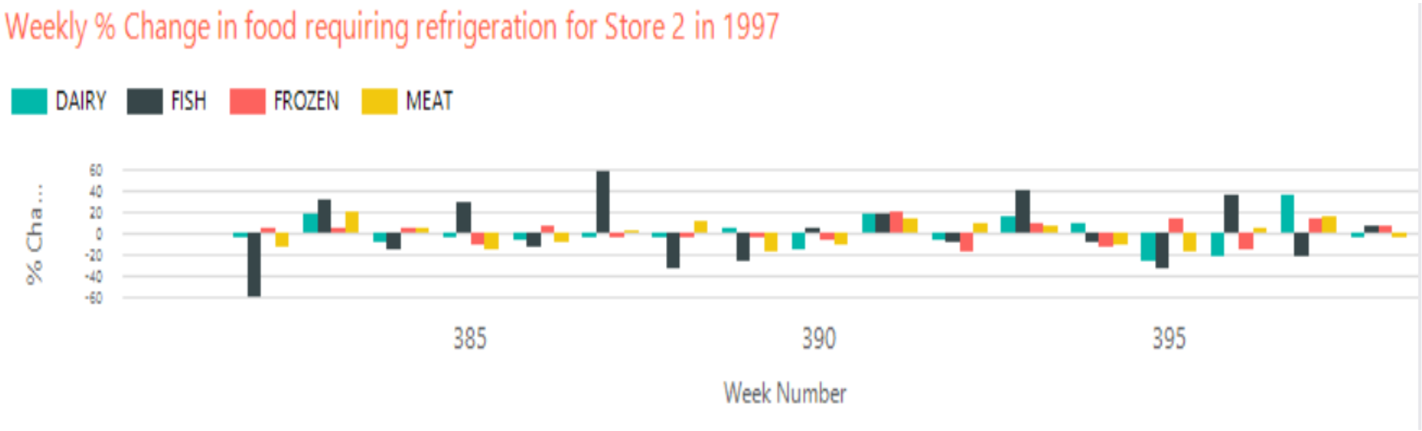
Weekly % Change in Sales

Store Number	Year	Week Number	Category Name	Total Sales	% Change
[Store_Number]	[Year]	[Week_Number]	[Category_Nam	[Total_Sales]	[ID_Change]

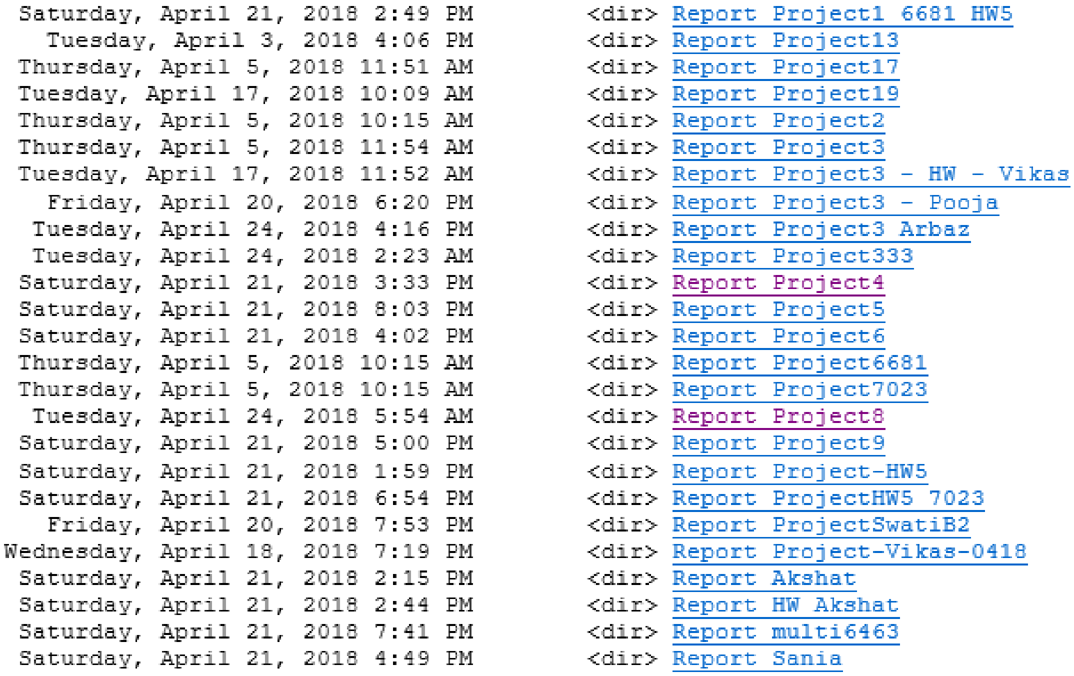
Weekly % Change in Fish, Frozen, Dairy & Meat Sales



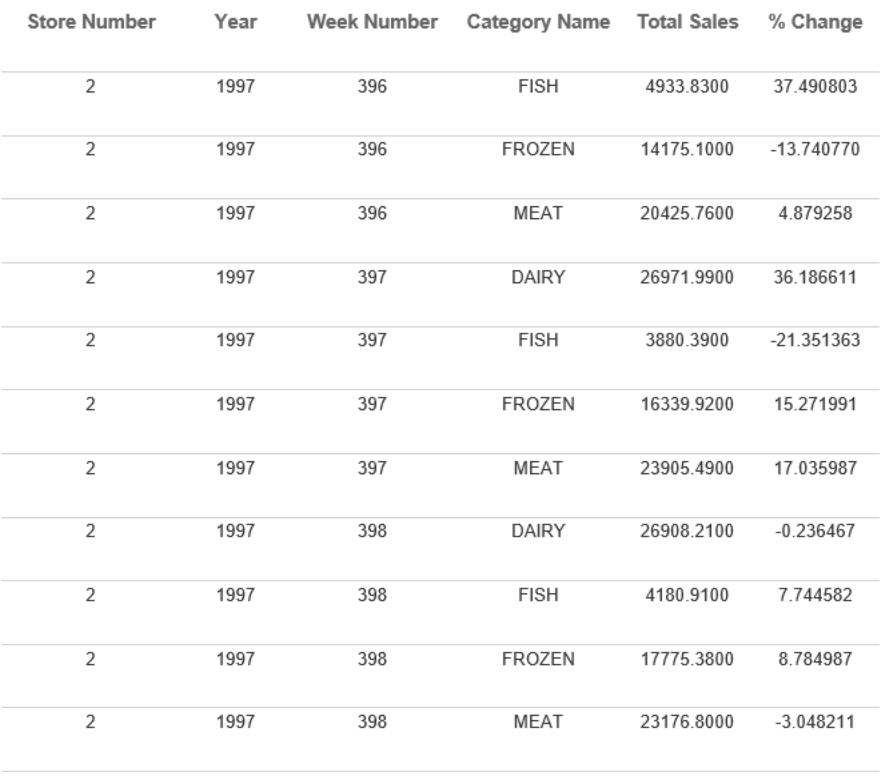
Store Number	Year	Week Number	Category Name	Total Sales	% Change
2	1997	381	FISH	8641.2800	
2	1997	381	DAIRY	28697.6300	
2	1997	381	FROZEN	15867.2400	
2	1997	381	MEAT	24639.8900	
2	1997	382	FROZEN	16646.7100	4.912448
2	1997	382	FISH	3518.1200	-59.287050
2	1997	382	MEAT	21765.8000	-11.664378
2	1997	382	DAIRY	27795.6300	-3.143116
2	1997	383	FROZEN	17568.0400	5.534607
2	1997	383	DAIRY	33137.1700	19.217193
2	1997	383	MEAT	26355.1500	21.085142
2	1997	383	FISH	4665.9800	32.627084
2	1997	384	DAIRY	30428.9600	-8.172725
2	1997	384	MEAT	27619.5200	4.797430
2	1997	384	FISH	3956.3700	-15.208166
2	1997	384	FROZEN	18681.0900	6.335652


Weekly % Change in food requiring refrigeration for Store 2 in 1997

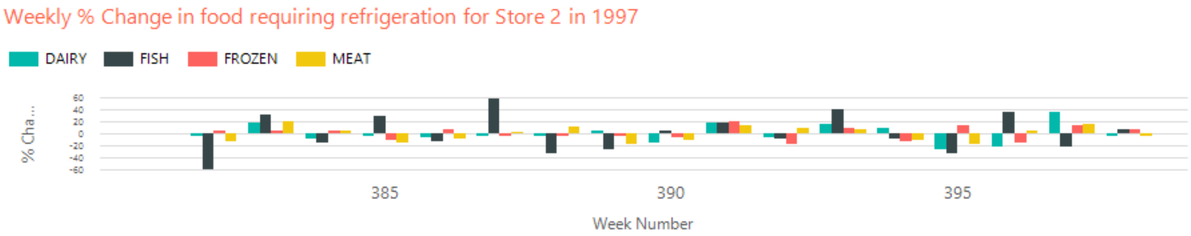
<u>Deployment Information</u>



```
  Saturday, April 21, 2018 2:49 PM        <dir> Report Project1 6681 HW5
    Tuesday, April 3, 2018 4:06 PM        <dir> Report Project13
  Thursday, April 5, 2018 11:51 AM        <dir> Report Project17
  Tuesday, April 17, 2018 10:09 AM        <dir> Report Project19
  Thursday, April 5, 2018 10:15 AM        <dir> Report Project2
  Thursday, April 5, 2018 11:54 AM        <dir> Report Project3
  Tuesday, April 17, 2018 11:52 AM        <dir> Report Project3 - HW - Vikas
    Friday, April 20, 2018 6:20 PM        <dir> Report Project3 - Pooja
   Tuesday, April 24, 2018 4:16 PM        <dir> Report Project3 Arbaz
   Tuesday, April 24, 2018 2:23 AM        <dir> Report Project333
  Saturday, April 21, 2018 3:33 PM        <dir> Report Project4
  Saturday, April 21, 2018 8:03 PM        <dir> Report Project5
  Saturday, April 21, 2018 4:02 PM        <dir> Report Project6
  Thursday, April 5, 2018 10:15 AM        <dir> Report Project6681
  Thursday, April 5, 2018 10:15 AM        <dir> Report Project7023
   Tuesday, April 24, 2018 5:54 AM        <dir> Report Project8
  Saturday, April 21, 2018 5:00 PM        <dir> Report Project9
  Saturday, April 21, 2018 1:59 PM        <dir> Report Project-HW5
  Saturday, April 21, 2018 6:54 PM        <dir> Report ProjectHW5 7023
    Friday, April 20, 2018 7:53 PM        <dir> Report ProjectSwatiB2
 Wednesday, April 18, 2018 7:19 PM        <dir> Report Project-Vikas-0418
  Saturday, April 21, 2018 2:15 PM        <dir> Report Akshat
  Saturday, April 21, 2018 2:44 PM        <dir> Report HW Akshat
  Saturday, April 21, 2018 7:41 PM        <dir> Report multi6463
  Saturday, April 21, 2018 4:49 PM        <dir> Report Sania
```



Store Number	Year	Week Number	Category Name	Total Sales	% Change
2	1997	396	FISH	4933.8300	37.490803
2	1997	396	FROZEN	14175.1000	-13.740770
2	1997	396	MEAT	20425.7600	4.879258
2	1997	397	DAIRY	26971.9900	36.186611
2	1997	397	FISH	3880.3900	-21.351363
2	1997	397	FROZEN	16339.9200	15.271991
2	1997	397	MEAT	23905.4900	17.035987
2	1997	398	DAIRY	26908.2100	-0.236467
2	1997	398	FISH	4180.9100	7.744582
2	1997	398	FROZEN	17775.3800	8.784987
2	1997	398	MEAT	23176.8000	-3.048211



Weekly % Change in food requiring refrigeration for Store 2 in 1997

Conclusion

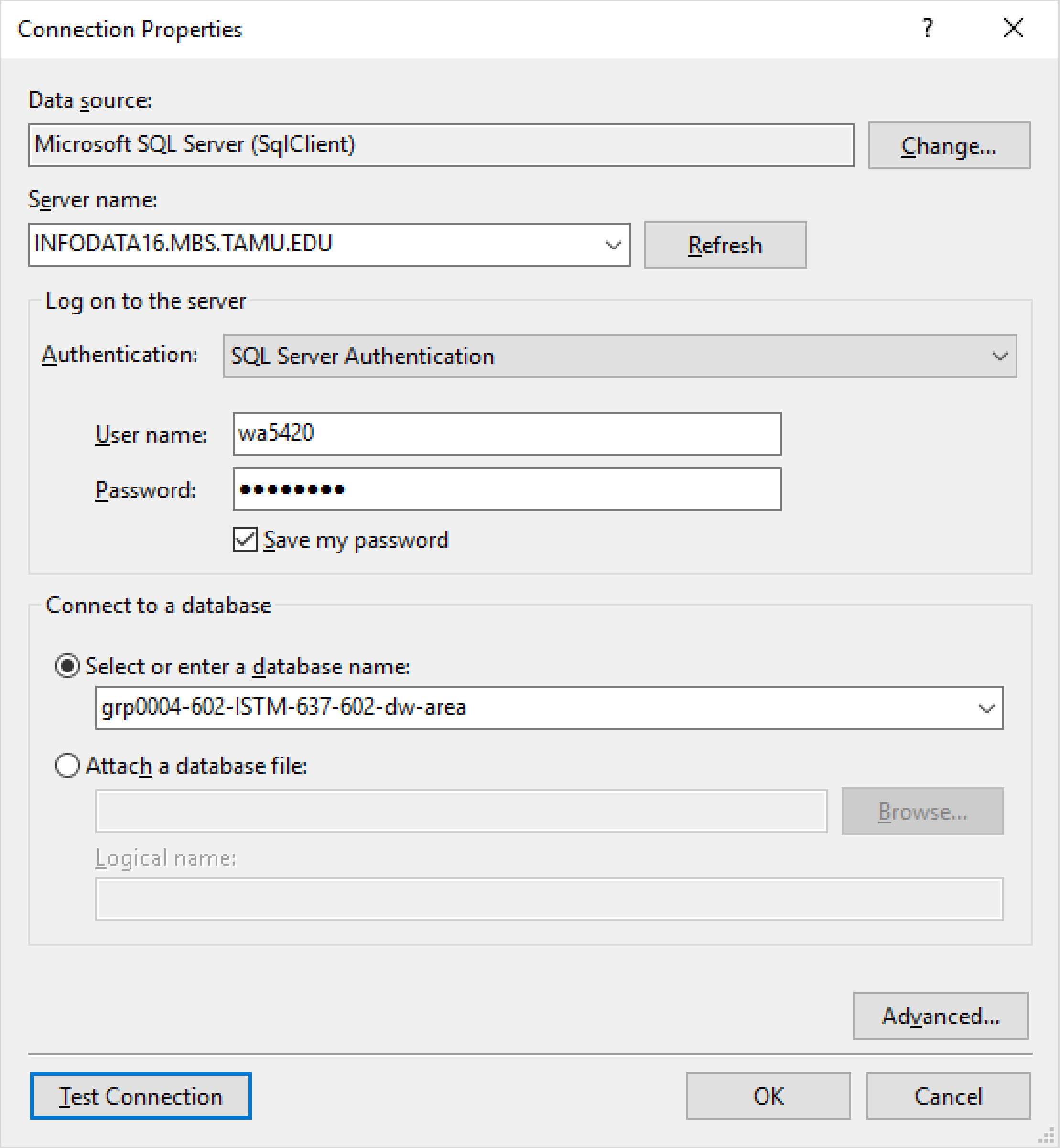
In view of the bar diagram we made utilizing SSRS, we can presume that there is huge measure of change in offers of different classifications of food items which requires refrigeration particularly in the idea of fish utilization. This offer pattern can give Dominick Finer Foods indicates about how to build up an incorporated system of strategically pitching in different meat and dairy items in coming year. A combo bundle construct marking down offer in light of mix of above items could prompt higher and unsurprising volumes. The store chief can design the stock to increase or down their amount without additional acquisition of any expected spike in utilization. Subsequently, the client-side push in volumes could be utilized to get provider side refunds and rebates to enhance edges.

6.3 SSRS for BQ-2

Based on the report plan, SSRS is adopted to create visualization for business question 2.

BQ-2: Which are the top 10 products with respect to profitability in Medium Tier stores w.r.t "Canned Soup" category of in year 1991?

Connection to Data Source



Connection Properties

Data source: Microsoft SQL Server (SqlClient)

Server name: INFODATA16.MBS.TAMU.EDU

Log on to the server

Authentication: SQL Server Authentication

User name: wa5420

Password: ●●●●●●●●●

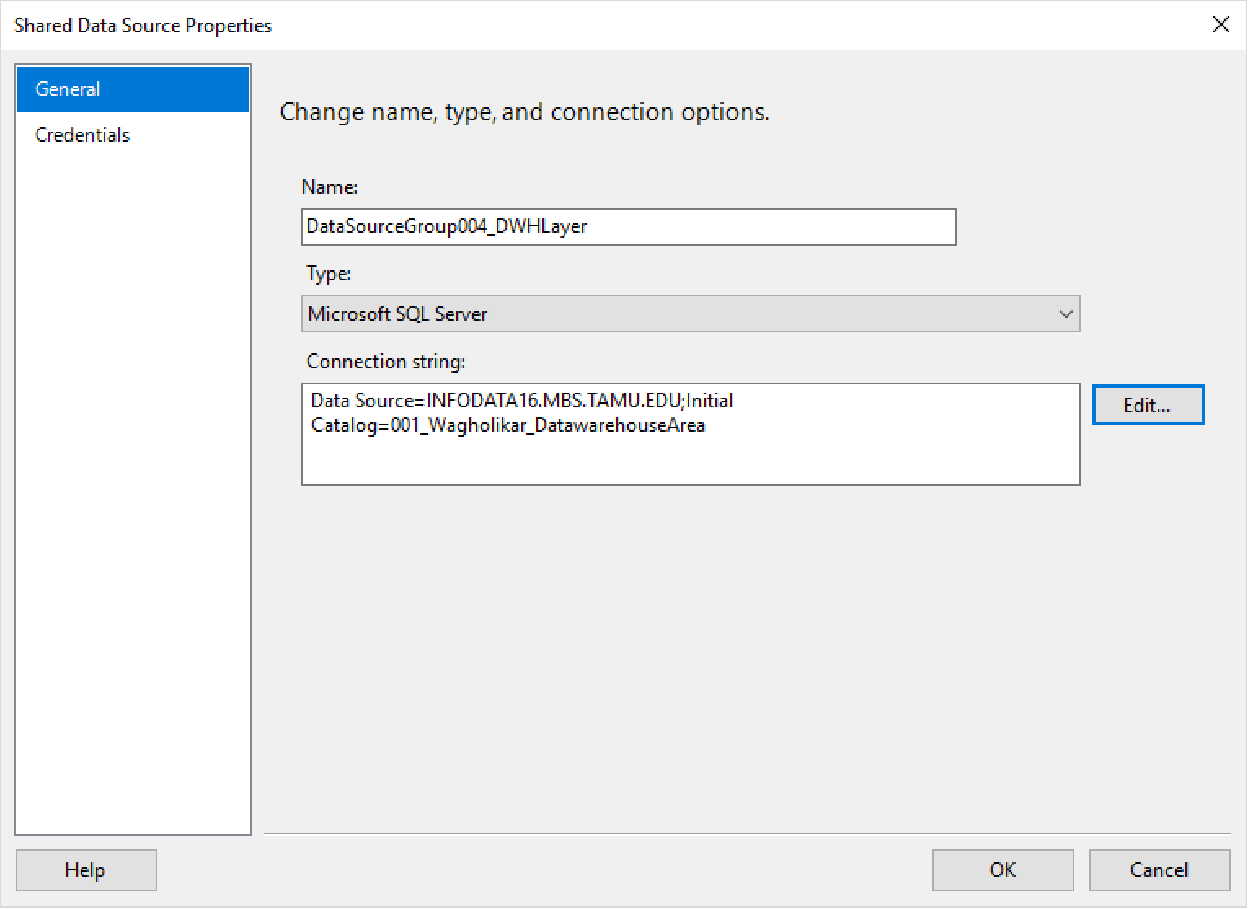
☑ Save my password

Connect to a database

● Select or enter a database name: grp0004-602-ISTM-637-602-dw-area

○ Attach a database file:

Logical name:



Shared Data Source Properties

Change name, type, and connection options.

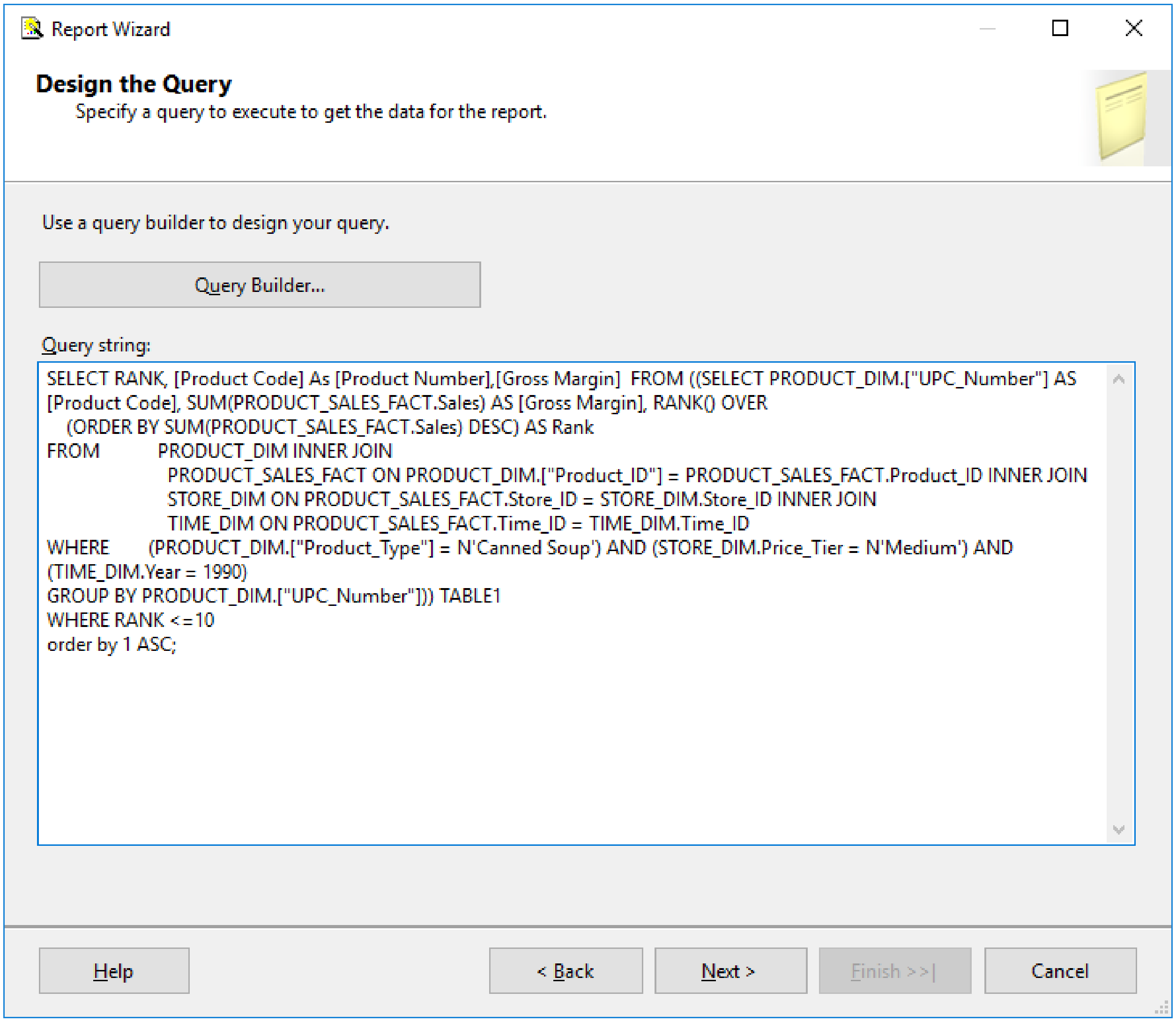
Name: DataSourceGroup004_DWHLayer

Type: Microsoft SQL Server

Connection string:

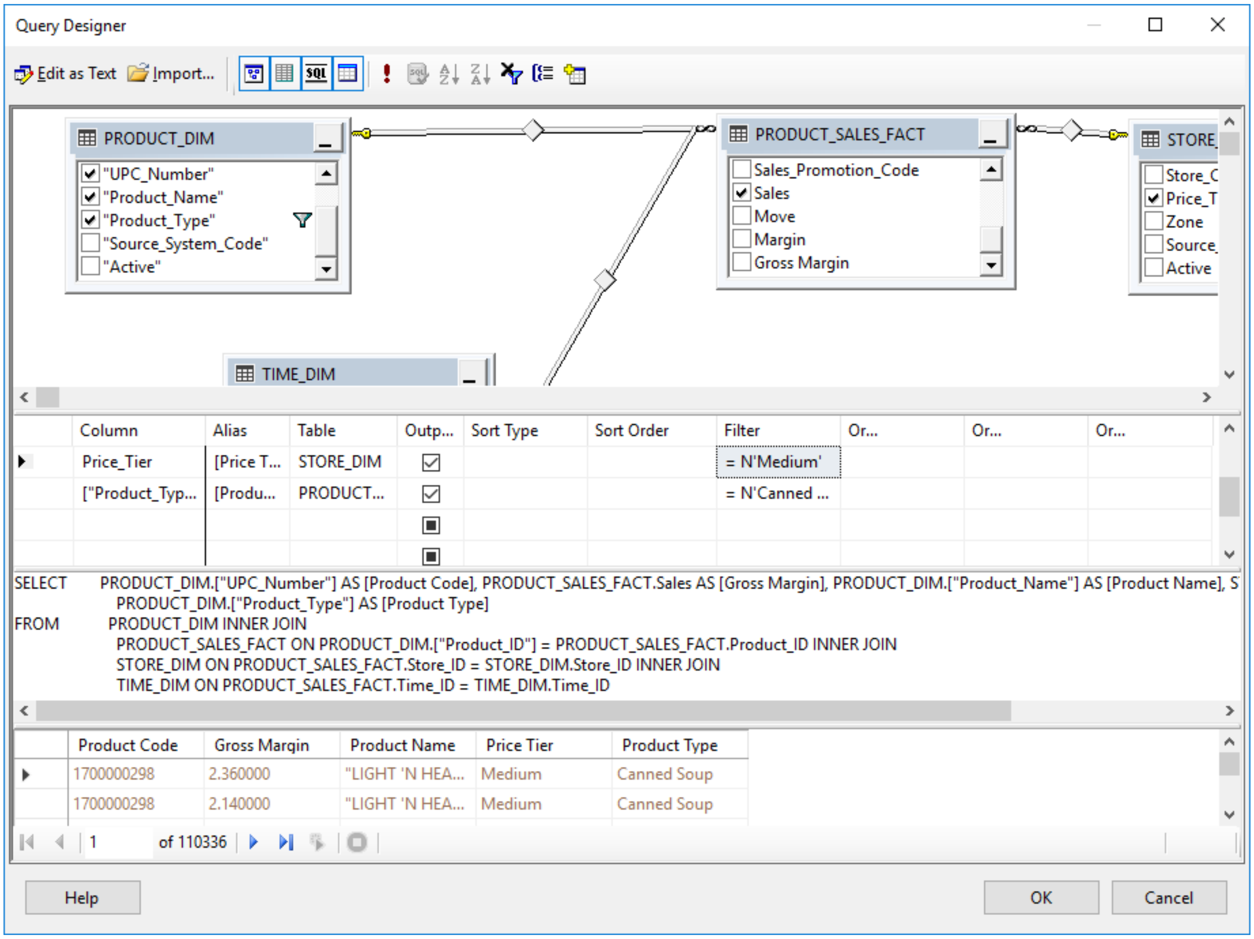
```
Data Source=INFODATA16.MBS.TAMU.EDU;Initial
Catalog=001_Wagholikar_DatawarehouseArea
```

Query Builder


Report Wizard
Design the Query
Specify a query to execute to get the data for the report.
Use a query builder to design your query.
Query Builder...
Query string:
Help
< Back
Next >
Finish >>|
Cancel

```
SELECT RANK, [Product Code] As [Product Number],[Gross Margin]  FROM ((SELECT PRODUCT_DIM.["UPC_Number"] AS
[Product Code], SUM(PRODUCT_SALES_FACT.Sales) AS [Gross Margin], RANK() OVER
   (ORDER BY SUM(PRODUCT_SALES_FACT.Sales) DESC) AS Rank
FROM         PRODUCT_DIM INNER JOIN
                  PRODUCT_SALES_FACT ON PRODUCT_DIM.["Product_ID"] = PRODUCT_SALES_FACT.Product_ID INNER JOIN
                  STORE_DIM ON PRODUCT_SALES_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
                  TIME_DIM ON PRODUCT_SALES_FACT.Time_ID = TIME_DIM.Time_ID
WHERE      (PRODUCT_DIM.["Product_Type"] = N'Canned Soup') AND (STORE_DIM.Price_Tier = N'Medium') AND
(TIME_DIM.Year = 1990)
GROUP BY PRODUCT_DIM.["UPC_Number"])) TABLE1
WHERE RANK <=10
order by 1 ASC;
```

Query Design



	Column	Alias	Table	Outp...	Sort Type	Sort Order	Filter	Or...	Or...	Or...
▶	Price_Tier	[Price T...	STORE_DIM	☑			= N'Medium'			
	["Product_Typ...	[Produ...	PRODUCT...	☑			= N'Canned ...			
				■						
				■						

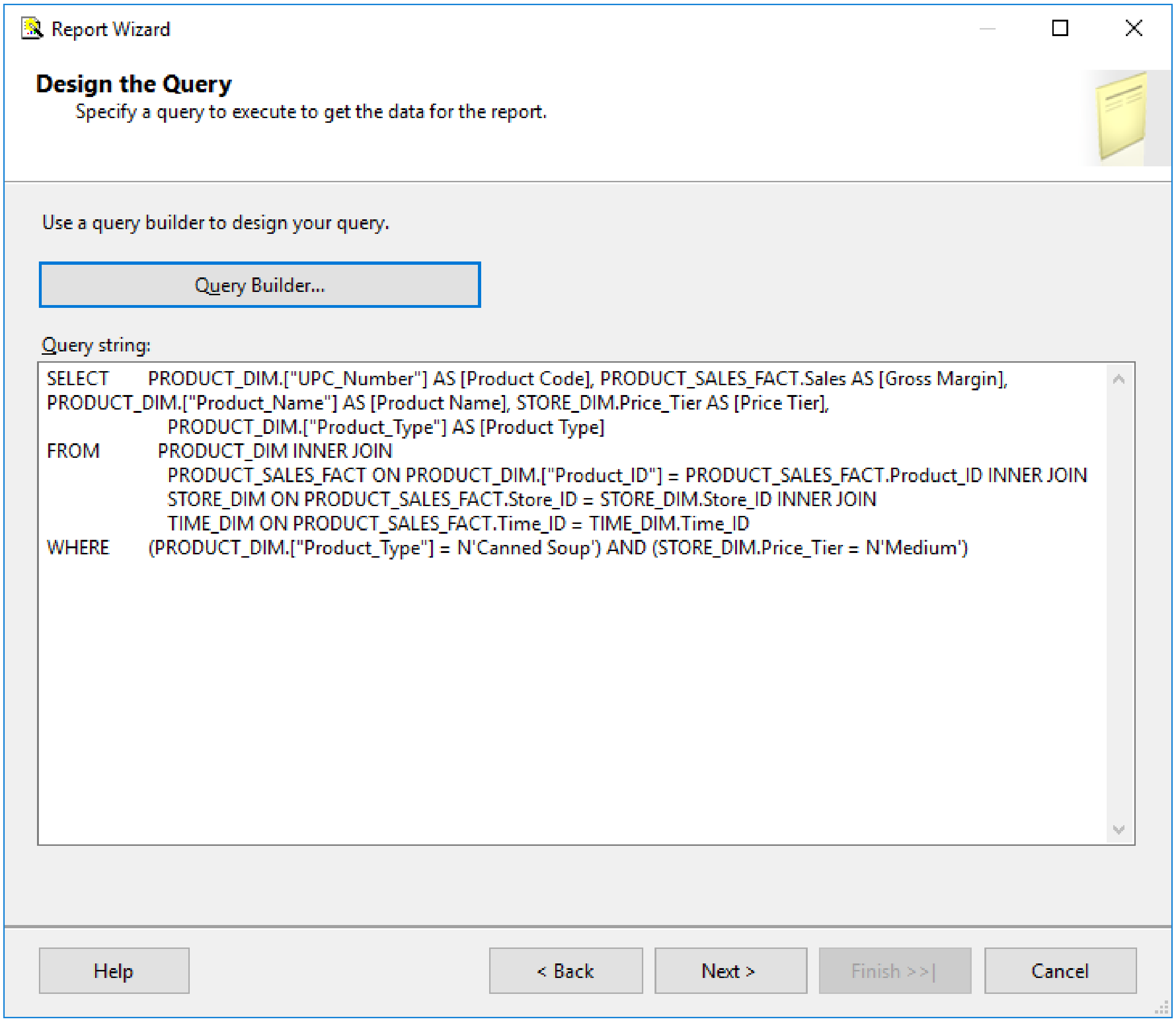
```
SELECT    PRODUCT_DIM.["UPC_Number"] AS [Product Code], PRODUCT_SALES_FACT.Sales AS [Gross Margin], PRODUCT_DIM.["Product_Name"] AS [Product Name], S
            PRODUCT_DIM.["Product_Type"] AS [Product Type]
FROM       PRODUCT_DIM INNER JOIN
            PRODUCT_SALES_FACT ON PRODUCT_DIM.["Product_ID"] = PRODUCT_SALES_FACT.Product_ID INNER JOIN
            STORE_DIM ON PRODUCT_SALES_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
            TIME_DIM ON PRODUCT_SALES_FACT.Time_ID = TIME_DIM.Time_ID
```

	Product Code	Gross Margin	Product Name	Price Tier	Product Type
▶	1700000298	2.360000	"LIGHT 'N HEA...	Medium	Canned Soup
	1700000298	2.140000	"LIGHT 'N HEA...	Medium	Canned Soup

1 of 110336

SQL Query

```
SELECT RANK, [Product Code] As [Product Number],[Gross Margin]  FROM ((SELECT
PRODUCT_DIM.["UPC_Number"] AS [Product Code], SUM(PRODUCT_SALES_FACT.Sales) AS [Gross
Margin], RANK() OVER
    (ORDER BY SUM(PRODUCT_SALES_FACT.Sales) DESC) AS Rank
FROM                PRODUCT_DIM INNER JOIN
                         PRODUCT_SALES_FACT ON PRODUCT_DIM.["Product_ID"] =
PRODUCT_SALES_FACT.Product_ID INNER JOIN
                         STORE_DIM ON PRODUCT_SALES_FACT.Store_ID = STORE_DIM.Store_ID
INNER JOIN
                         TIME_DIM ON PRODUCT_SALES_FACT.Time_ID = TIME_DIM.Time_ID
WHERE         (PRODUCT_DIM.["Product_Type"] = N'Canned Soup') AND (STORE_DIM.Price_Tier =
N'Medium') AND (TIME_DIM.Year = 1990)
GROUP BY PRODUCT_DIM.["UPC_Number"])) TABLE1
WHERE RANK <=10
order by 1 ASC;
```



Design the Query

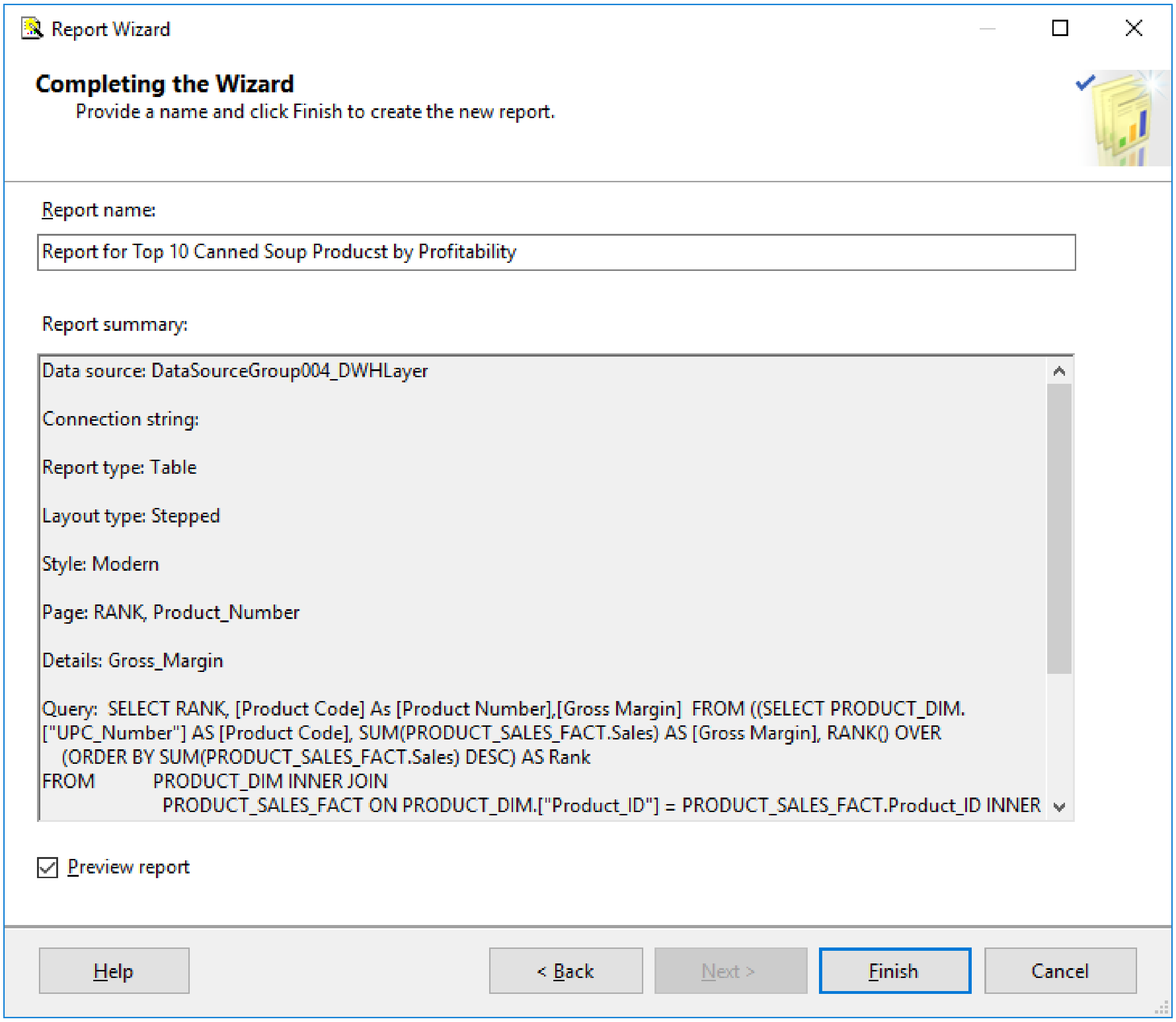
Specify a query to execute to get the data for the report.

Use a query builder to design your query.

Query string:

```
SELECT      PRODUCT_DIM.["UPC_Number"] AS [Product Code], PRODUCT_SALES_FACT.Sales AS [Gross Margin],
PRODUCT_DIM.["Product_Name"] AS [Product Name], STORE_DIM.Price_Tier AS [Price Tier],
                  PRODUCT_DIM.["Product_Type"] AS [Product Type]
FROM         PRODUCT_DIM INNER JOIN
                  PRODUCT_SALES_FACT ON PRODUCT_DIM.["Product_ID"] = PRODUCT_SALES_FACT.Product_ID INNER JOIN
                  STORE_DIM ON PRODUCT_SALES_FACT.Store_ID = STORE_DIM.Store_ID INNER JOIN
                  TIME_DIM ON PRODUCT_SALES_FACT.Time_ID = TIME_DIM.Time_ID
WHERE      (PRODUCT_DIM.["Product_Type"] = N'Canned Soup') AND (STORE_DIM.Price_Tier = N'Medium')
```

Final Report



Completing the Wizard

Provide a name and click Finish to create the new report.

Report name:

Report for Top 10 Canned Soup Producst by Profitability

Report summary:

Data source: DataSourceGroup004_DWHLayer

Connection string:

Report type: Table

Layout type: Stepped

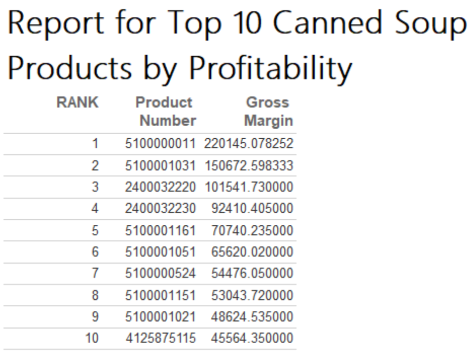
Style: Modern

Page: RANK, Product_Number

Details: Gross_Margin

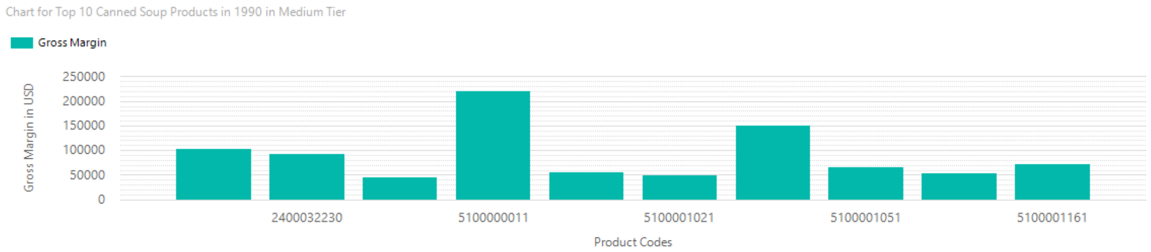
```
Query:  SELECT RANK, [Product Code] As [Product Number],[Gross Margin]  FROM ((SELECT PRODUCT_DIM.
["UPC_Number"] AS [Product Code], SUM(PRODUCT_SALES_FACT.Sales) AS [Gross Margin], RANK() OVER
    (ORDER BY SUM(PRODUCT_SALES_FACT.Sales) DESC) AS Rank
FROM          PRODUCT_DIM INNER JOIN
                   PRODUCT_SALES_FACT ON PRODUCT_DIM.["Product_ID"] = PRODUCT_SALES_FACT.Product_ID INNER
```

☑ Preview report



Report for Top 10 Canned Soup Products by Profitability

RANK	Product Number	Gross Margin
1	5100000011	220145.078252
2	5100001031	150672.598333
3	2400032220	101541.730000
4	2400032230	92410.405000
5	5100001161	70740.235000
6	5100001051	65620.020000
7	5100000524	54476.050000
8	5100001151	53043.720000
9	5100001021	48624.535000
10	4125875115	45564.350000



Conclusion

Based on the bar chart we created using SSRS, we can conclude that the top 10 profitable "Canned Soup" product in Medium Tier stores in 1991 is 5100000011, 5100001031, 2400032220, 2400032230, 5100001161, 5100001051, 5100000524, 5100001151, 5100001021 and 4125875115. This sell trend can give Dominick Finer Foods hints about how to promote Canned Soup related products in Medium Tier store. Also, same method can be used on other products to determine the sell trend, thus to facilitate better inventory and enhance sale revenue.

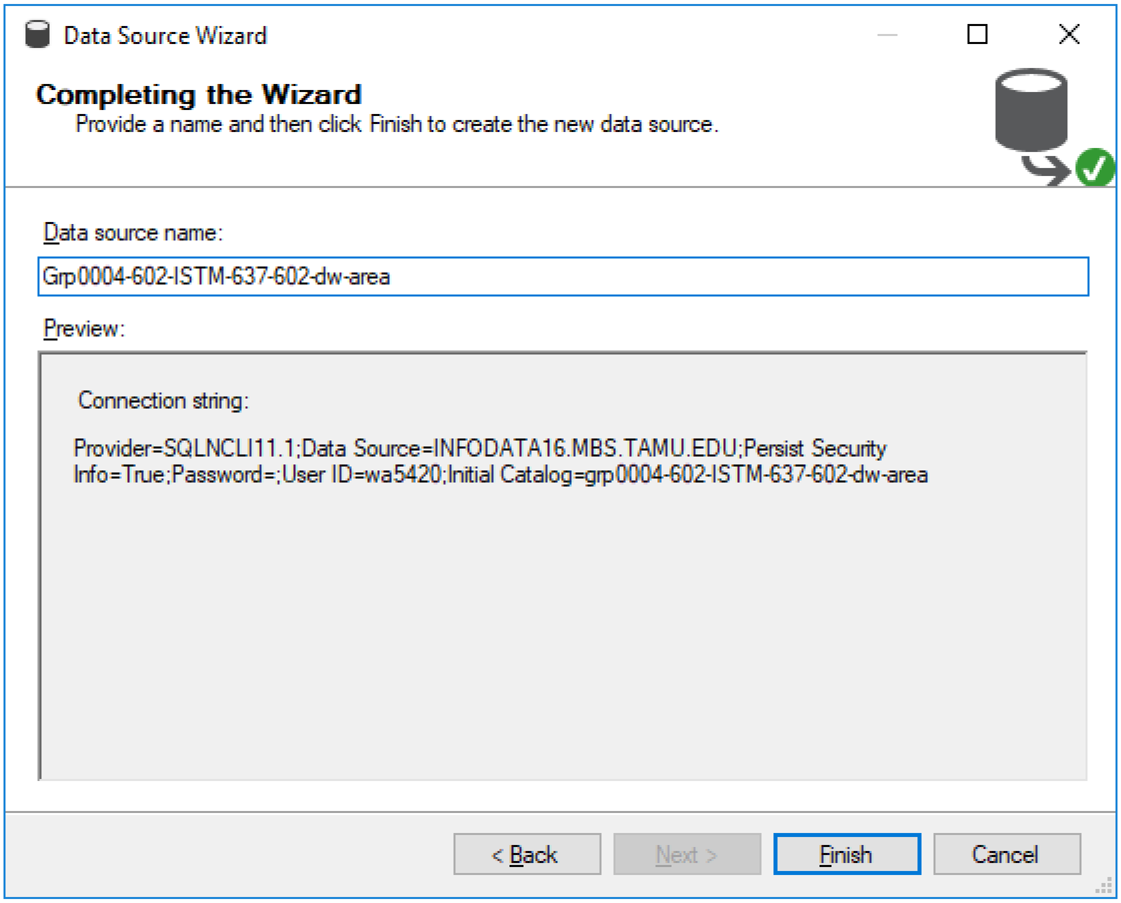
6.4 SSAS+SSRS for BQ-3

Based on the report plan, SSAS+SSRS is adopted to create visualization for business question 3.

BQ-3: What is the growth trend of Soft Drinks versus Bottled Juice for year 1990 and 1991? Which of these has negative growth rate? What is the trend for bottled juice especially in Low Tier?

6.4.1 SSAS Part

Connect to data source



Completing the Wizard

Provide a name and then click Finish to create the new data source.

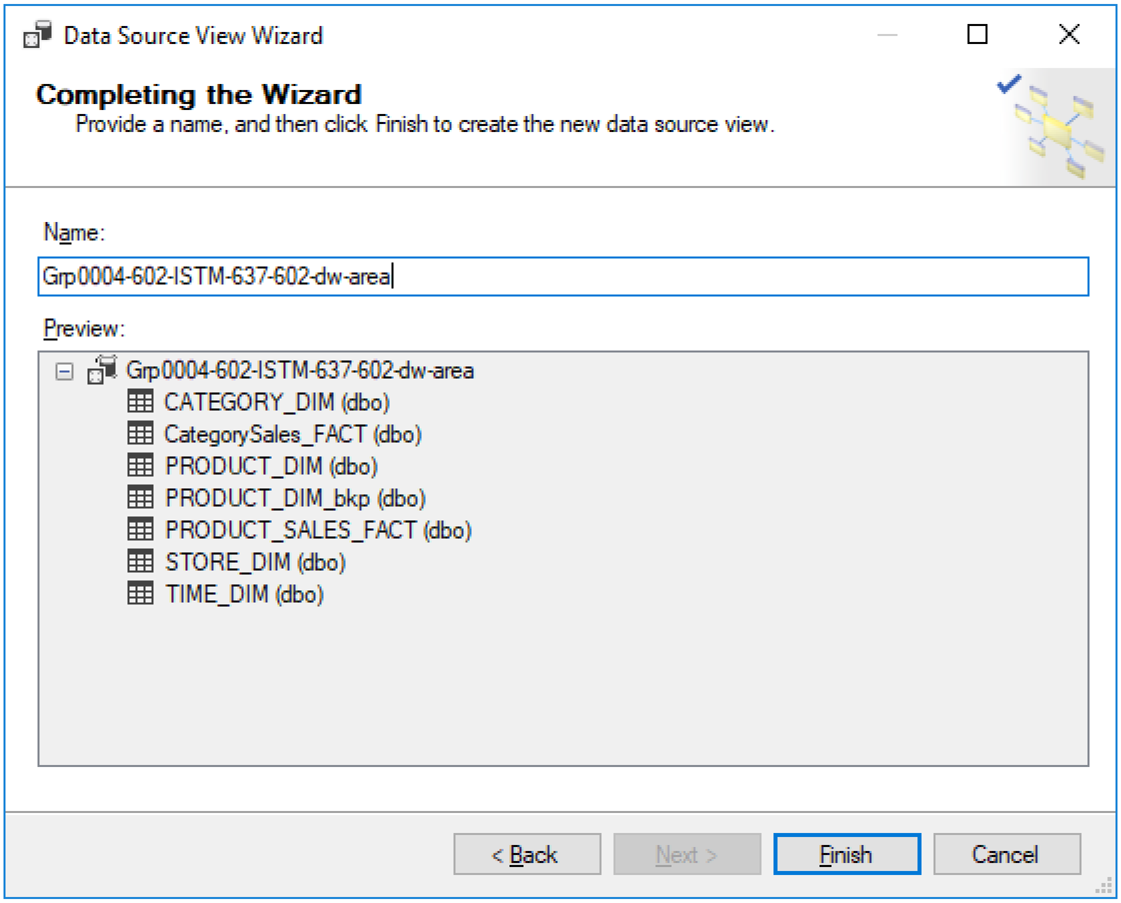
Data source name:

Grp0004-602-ISTM-637-602-dw-area

Preview:

Connection string:

```
Provider=SQLNCLI11.1;Data Source=INFODATA16.MBS.TAMU.EDU;Persist Security
Info=True;Password=;User ID=wa5420;Initial Catalog=grp0004-602-ISTM-637-602-dw-area
```



Data Source View Wizard

Completing the Wizard

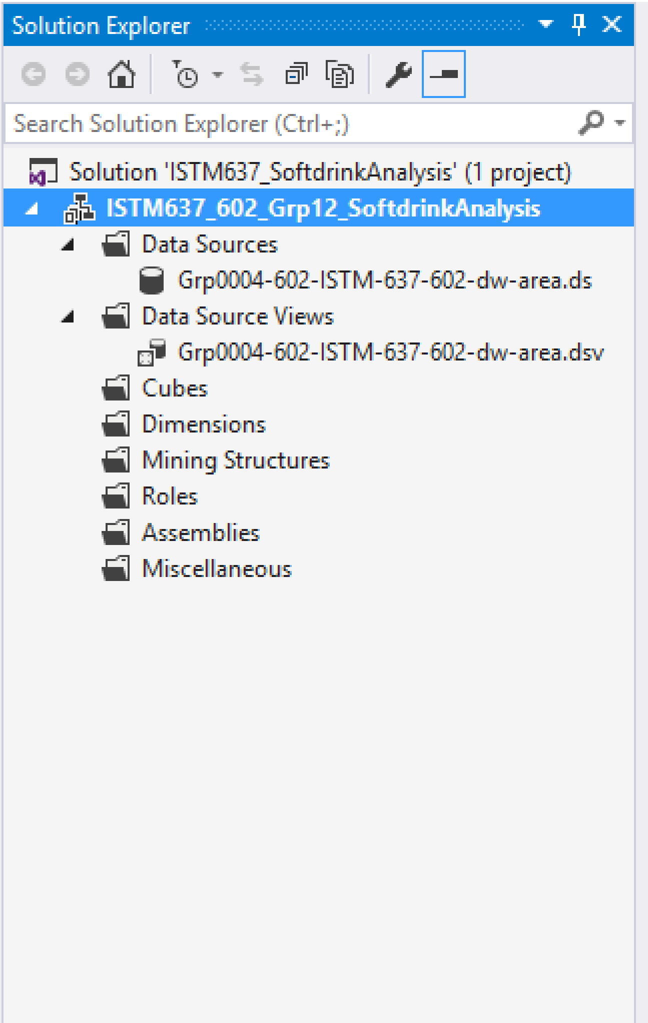
Provide a name, and then click Finish to create the new data source view.

Name:

Grp0004-602-ISTM-637-602-dw-area

Preview:

- Grp0004-602-ISTM-637-602-dw-area
  - CATEGORY_DIM (dbo)
  - CategorySales_FACT (dbo)
  - PRODUCT_DIM (dbo)
  - PRODUCT_DIM_bkp (dbo)
  - PRODUCT_SALES_FACT (dbo)
  - STORE_DIM (dbo)
  - TIME_DIM (dbo)

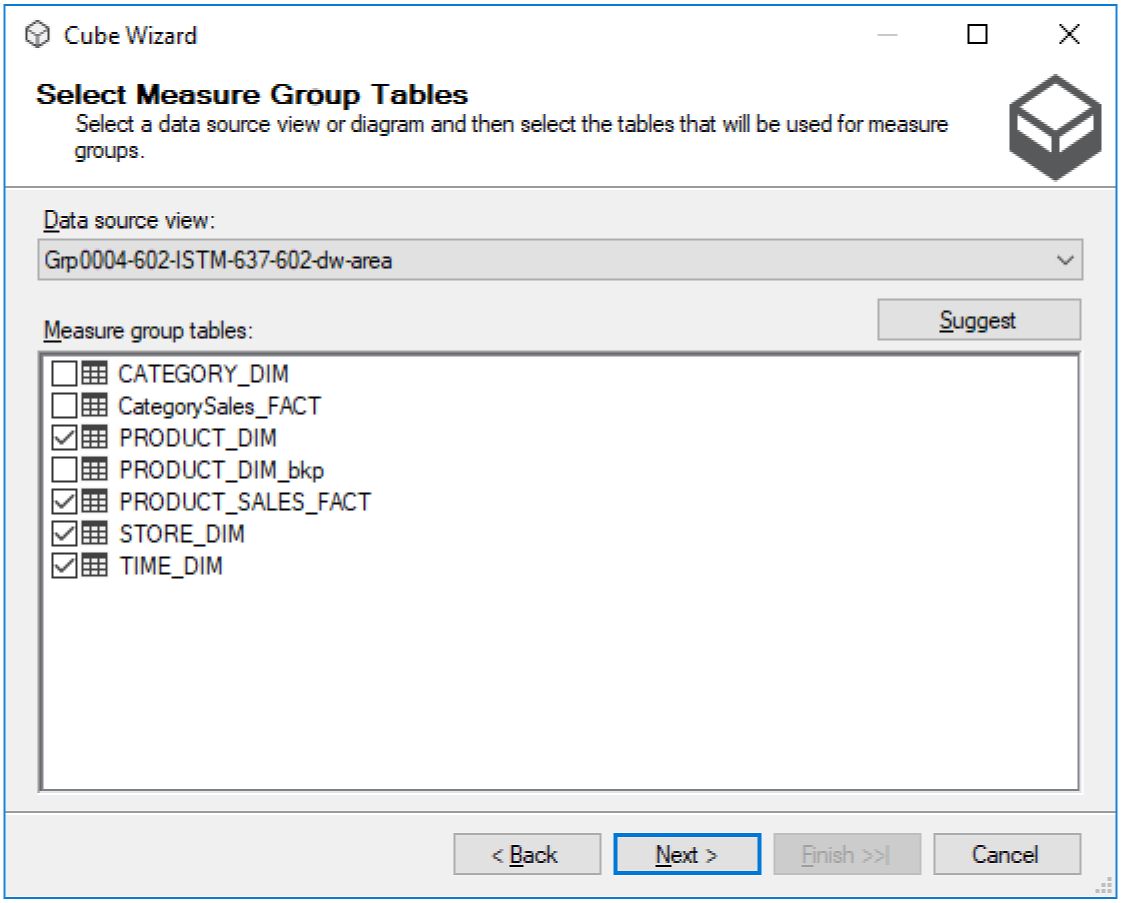


Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'ISTM637_SoftdrinkAnalysis' (1 project)
  - ISTM637_602_Grp12_SoftdrinkAnalysis
    - Data Sources
      - Grp0004-602-ISTM-637-602-dw-area.ds
    - Data Source Views
      - Grp0004-602-ISTM-637-602-dw-area.dsv
    - Cubes
    - Dimensions
    - Mining Structures
    - Roles
    - Assemblies
    - Miscellaneous

<u>Cube creation</u>



Select Measure Group Tables

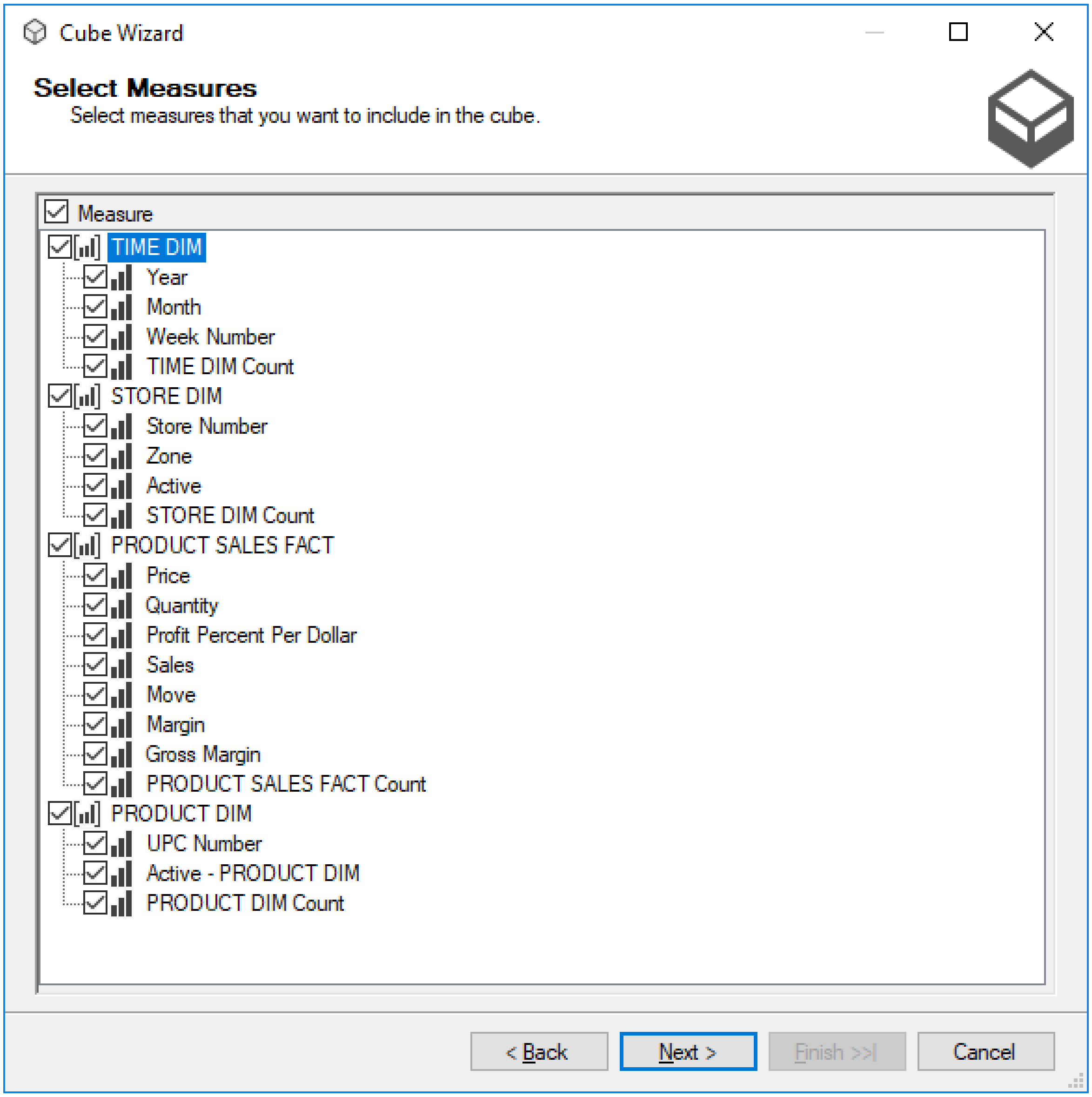
Select a data source view or diagram and then select the tables that will be used for measure groups.

Data source view:

Grp0004-602-ISTM-637-602-dw-area

Measure group tables:

- ☐ CATEGORY_DIM
- ☐ CategorySales_FACT
- ☑ PRODUCT_DIM
- ☐ PRODUCT_DIM_bkp
- ☑ PRODUCT_SALES_FACT
- ☑ STORE_DIM
- ☑ TIME_DIM



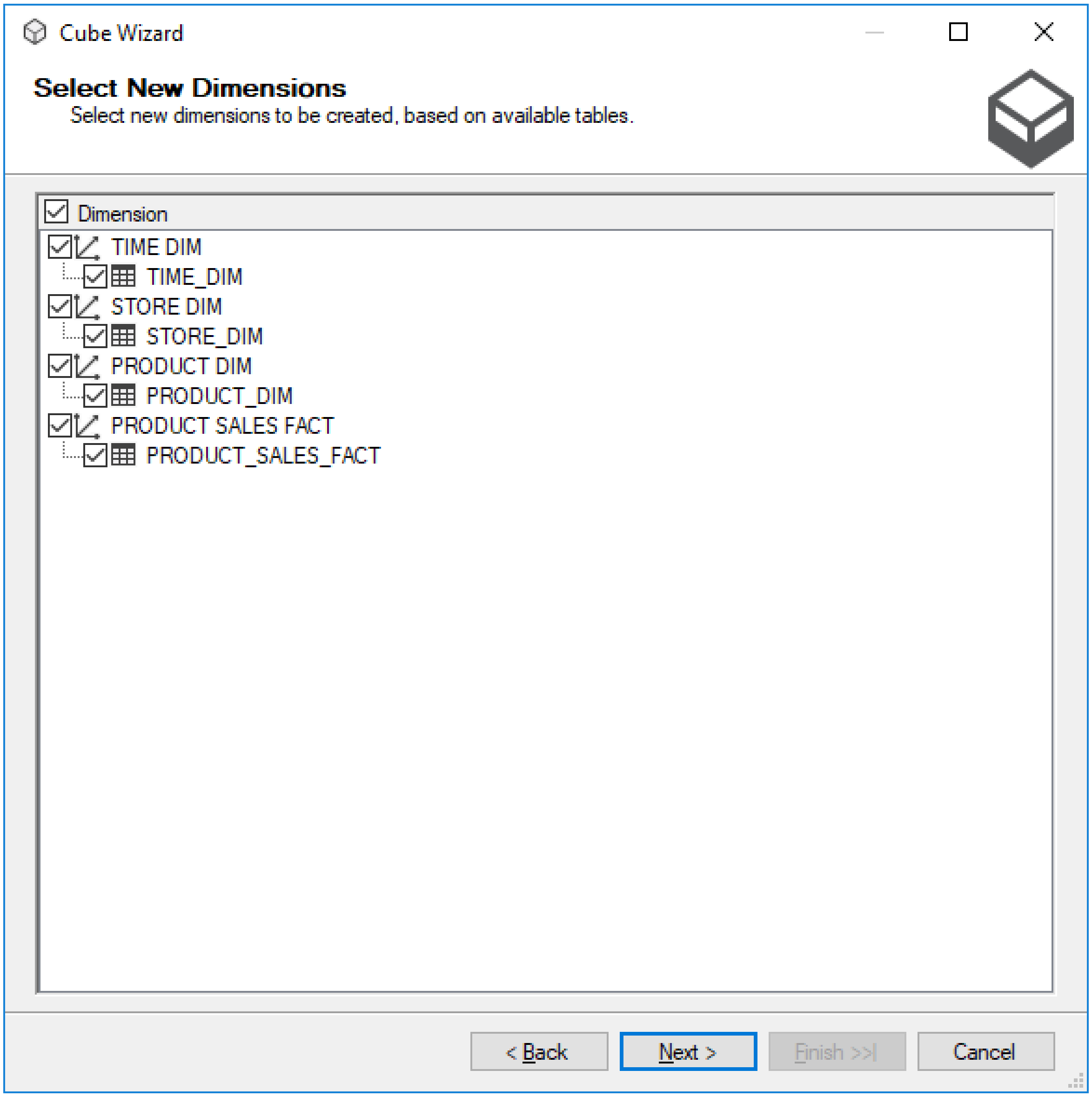
Cube Wizard

Select Measures

Select measures that you want to include in the cube.

☑ Measure

- ☑ TIME DIM
  - ☑ Year
  - ☑ Month
  - ☑ Week Number
  - ☑ TIME DIM Count
- ☑ STORE DIM
  - ☑ Store Number
  - ☑ Zone
  - ☑ Active
  - ☑ STORE DIM Count
- ☑ PRODUCT SALES FACT
  - ☑ Price
  - ☑ Quantity
  - ☑ Profit Percent Per Dollar
  - ☑ Sales
  - ☑ Move
  - ☑ Margin
  - ☑ Gross Margin
  - ☑ PRODUCT SALES FACT Count
- ☑ PRODUCT DIM
  - ☑ UPC Number
  - ☑ Active - PRODUCT DIM
  - ☑ PRODUCT DIM Count

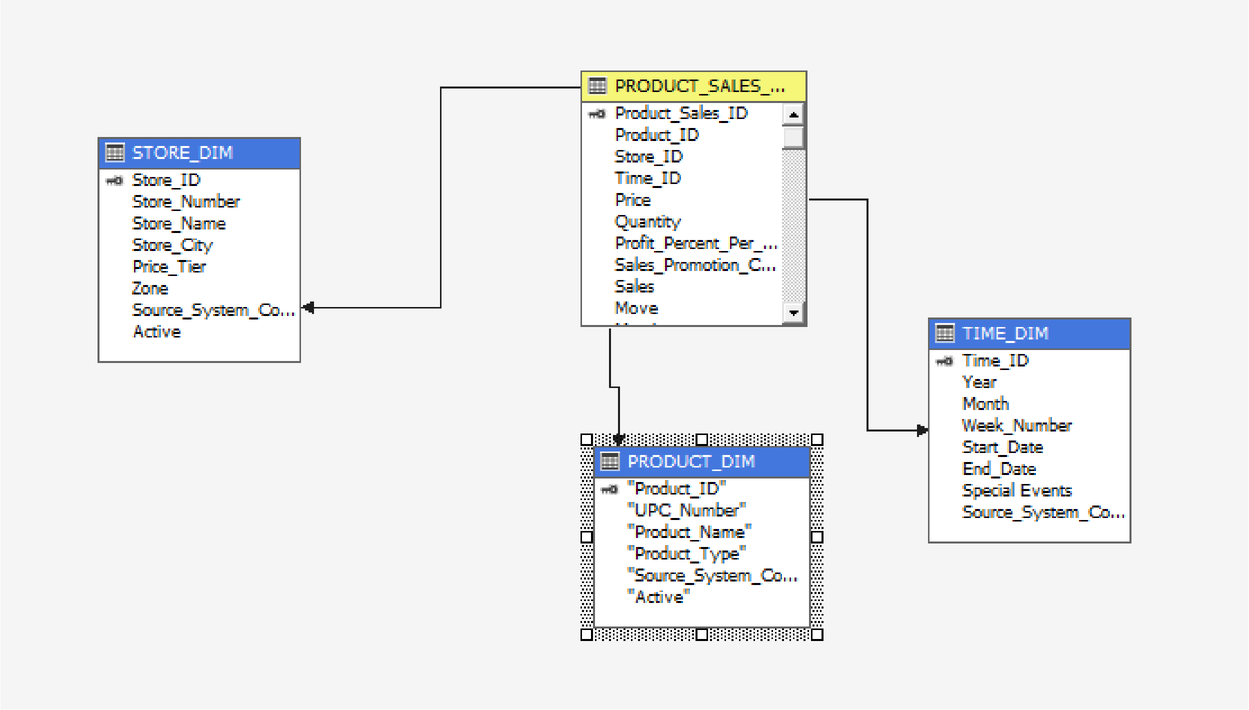


Select New Dimensions

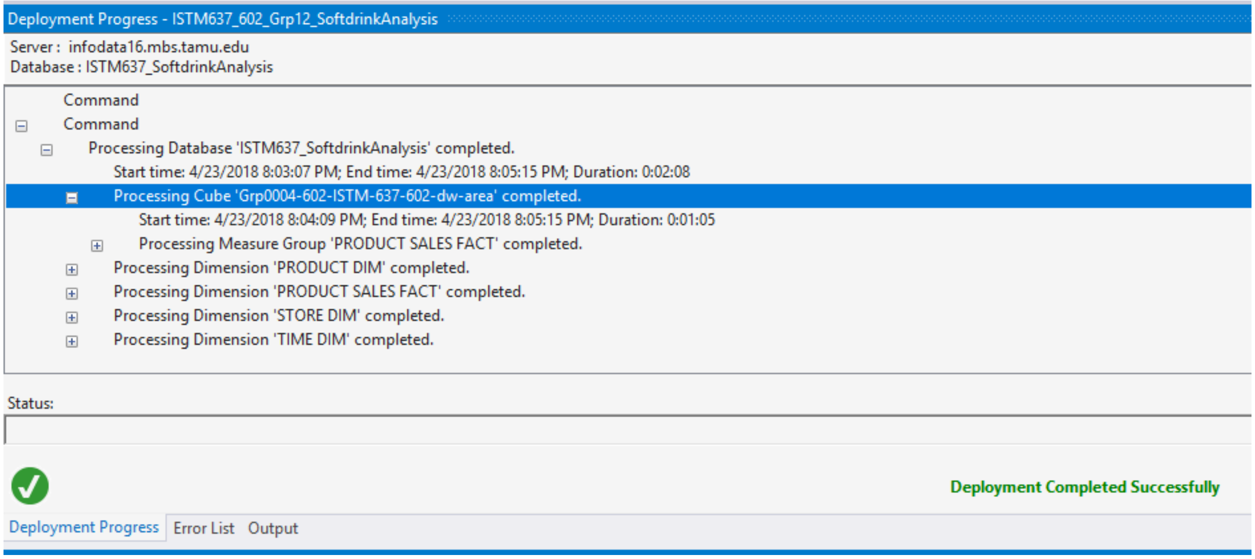
Select new dimensions to be created, based on available tables.

☑ Dimension
- ☑ TIME DIM
  - ☑ TIME_DIM
- ☑ STORE DIM
  - ☑ STORE_DIM
- ☑ PRODUCT DIM
  - ☑ PRODUCT_DIM
- ☑ PRODUCT SALES FACT
  - ☑ PRODUCT_SALES_FACT

Cube Structure



Success Deploy



Deployment Progress - ISTM637_602_Grp12_SoftdrinkAnalysis

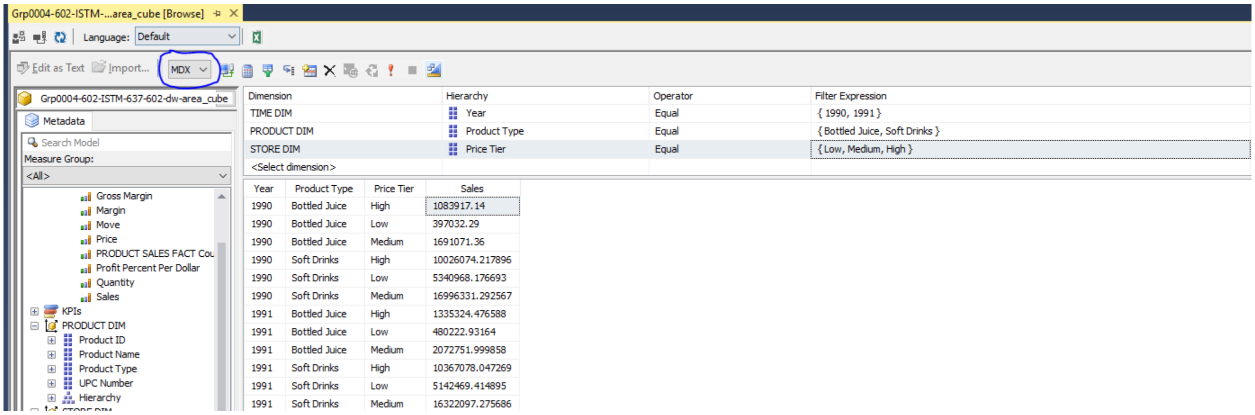
Server : infodata16.mbs.tamu.edu
Database : ISTM637_SoftdrinkAnalysis

```
    Command
⊟   Command
  ⊟   Processing Database 'ISTM637_SoftdrinkAnalysis' completed.
          Start time: 4/23/2018 8:03:07 PM; End time: 4/23/2018 8:05:15 PM; Duration: 0:02:08
    ⊟   Processing Cube 'Grp0004-602-ISTM-637-602-dw-area' completed.
            Start time: 4/23/2018 8:04:09 PM; End time: 4/23/2018 8:05:15 PM; Duration: 0:01:05
      ⊞   Processing Measure Group 'PRODUCT SALES FACT' completed.
    ⊞   Processing Dimension 'PRODUCT DIM' completed.
    ⊞   Processing Dimension 'PRODUCT SALES FACT' completed.
    ⊞   Processing Dimension 'STORE DIM' completed.
    ⊞   Processing Dimension 'TIME DIM' completed.
```

Status:

✓ Deployment Completed Successfully

Generate Excel

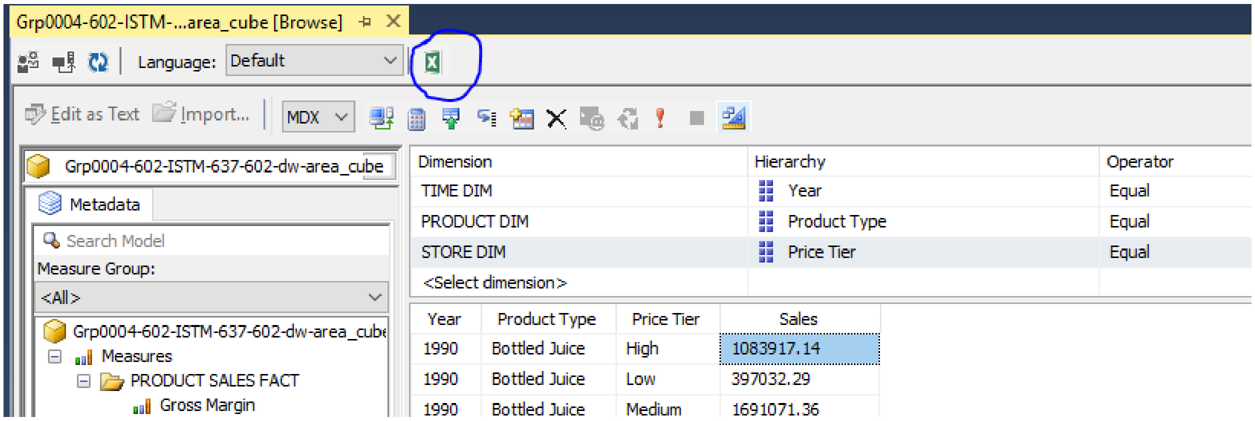


Language: Default

Measure Group: <All>

Dimension	Hierarchy	Operator	Filter Expression
TIME DIM	Year	Equal	{ 1990, 1991 }
PRODUCT DIM	Product Type	Equal	{ Bottled Juice, Soft Drinks }
STORE DIM	Price Tier	Equal	{ Low, Medium, High }
<Select dimension>			

Year	Product Type	Price Tier	Sales
1990	Bottled Juice	High	1083917.14
1990	Bottled Juice	Low	397032.29
1990	Bottled Juice	Medium	1691071.36
1990	Soft Drinks	High	10026074.217896
1990	Soft Drinks	Low	5340968.176693
1990	Soft Drinks	Medium	16996331.292567
1991	Bottled Juice	High	1335324.476588
1991	Bottled Juice	Low	480222.93164
1991	Bottled Juice	Medium	2072751.999858
1991	Soft Drinks	High	10367078.047269
1991	Soft Drinks	Low	5142469.414895
1991	Soft Drinks	Medium	16322097.275686

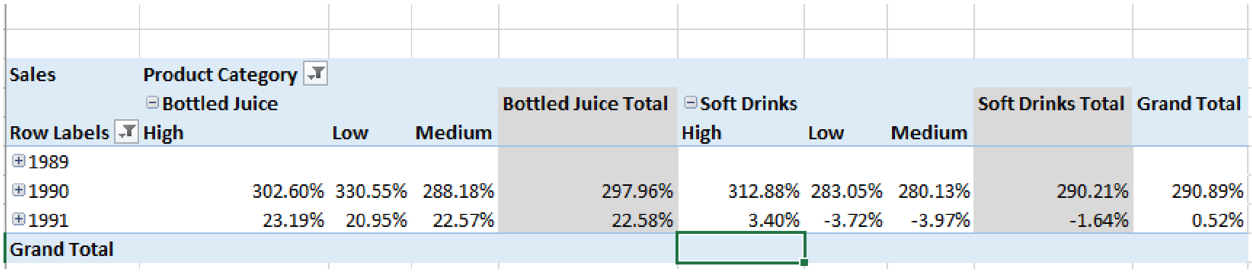

Grp0004-602-ISTM-...area_cube [Browse]
Edit as Text
Import...
MDX
Grp0004-602-ISTM-637-602-dw-area_cube
Metadata
Search Model
Grp0004-602-ISTM-637-602-dw-area_cube
Measures
PRODUCT SALES FACT
Gross Margin

Language: Default

Measure Group: <All>

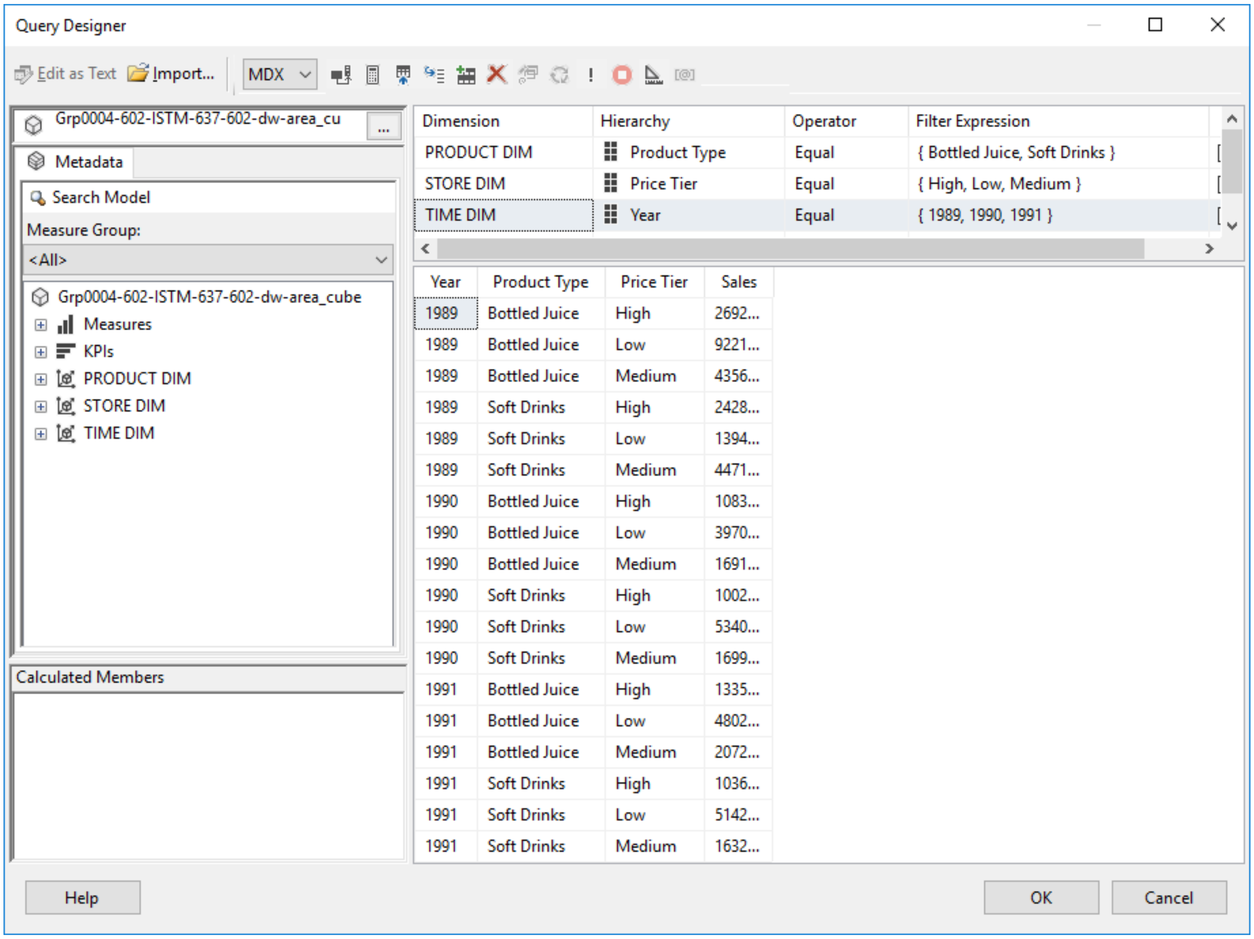
Dimension	Hierarchy	Operator
TIME DIM	Year	Equal
PRODUCT DIM	Product Type	Equal
STORE DIM	Price Tier	Equal
<Select dimension>		

Year	Product Type	Price Tier	Sales
1990	Bottled Juice	High	1083917.14
1990	Bottled Juice	Low	397032.29
1990	Bottled Juice	Medium	1691071.36



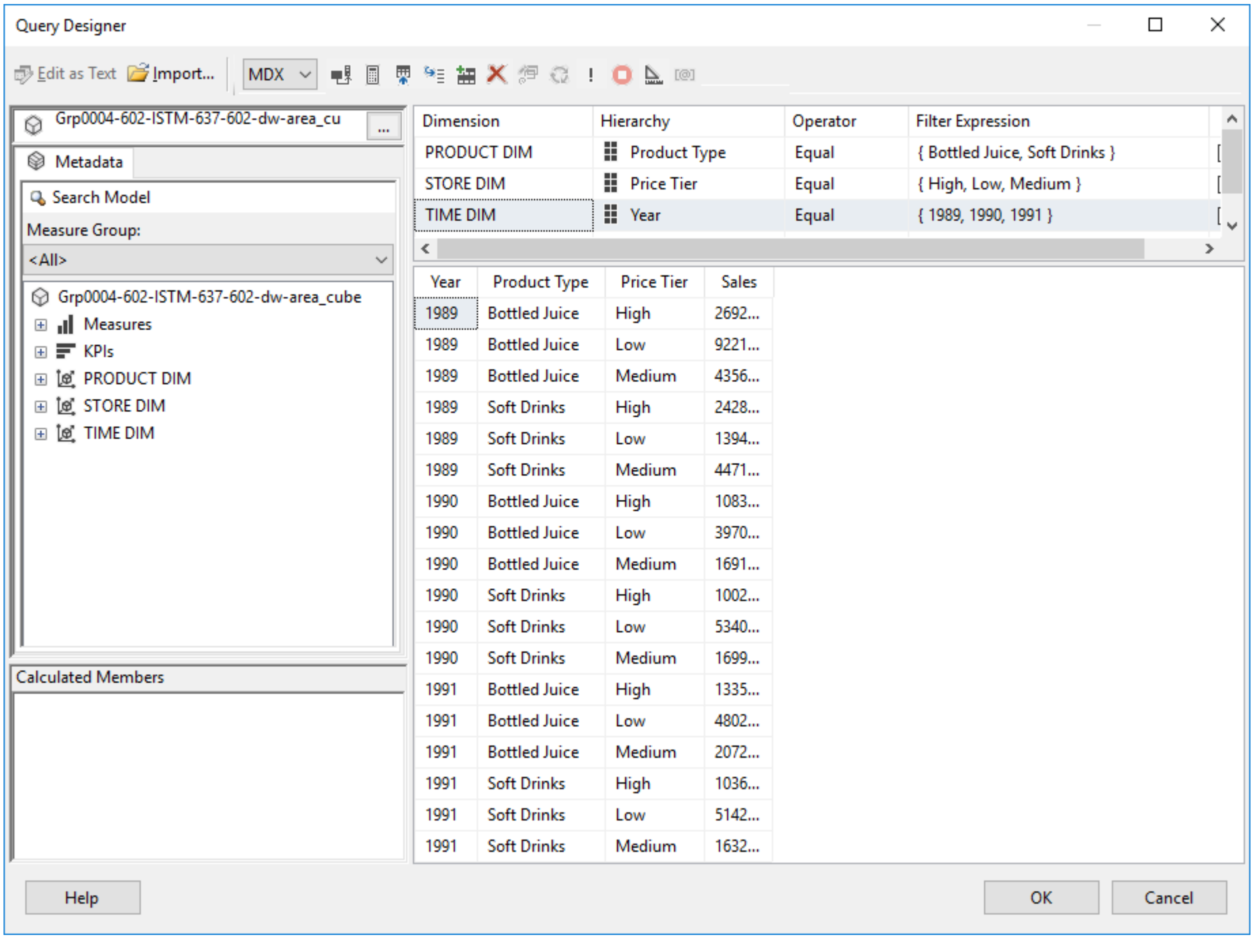
Sales	Product Category								
	⊟ Bottled Juice			Bottled Juice Total	⊟ Soft Drinks			Soft Drinks Total	Grand Total
Row Labels	High	Low	Medium		High	Low	Medium		
⊞ 1989									
⊞ 1990	302.60%	330.55%	288.18%	297.96%	312.88%	283.05%	280.13%	290.21%	290.89%
⊞ 1991	23.19%	20.95%	22.57%	22.58%	3.40%	-3.72%	-3.97%	-1.64%	0.52%
Grand Total									

Query Design



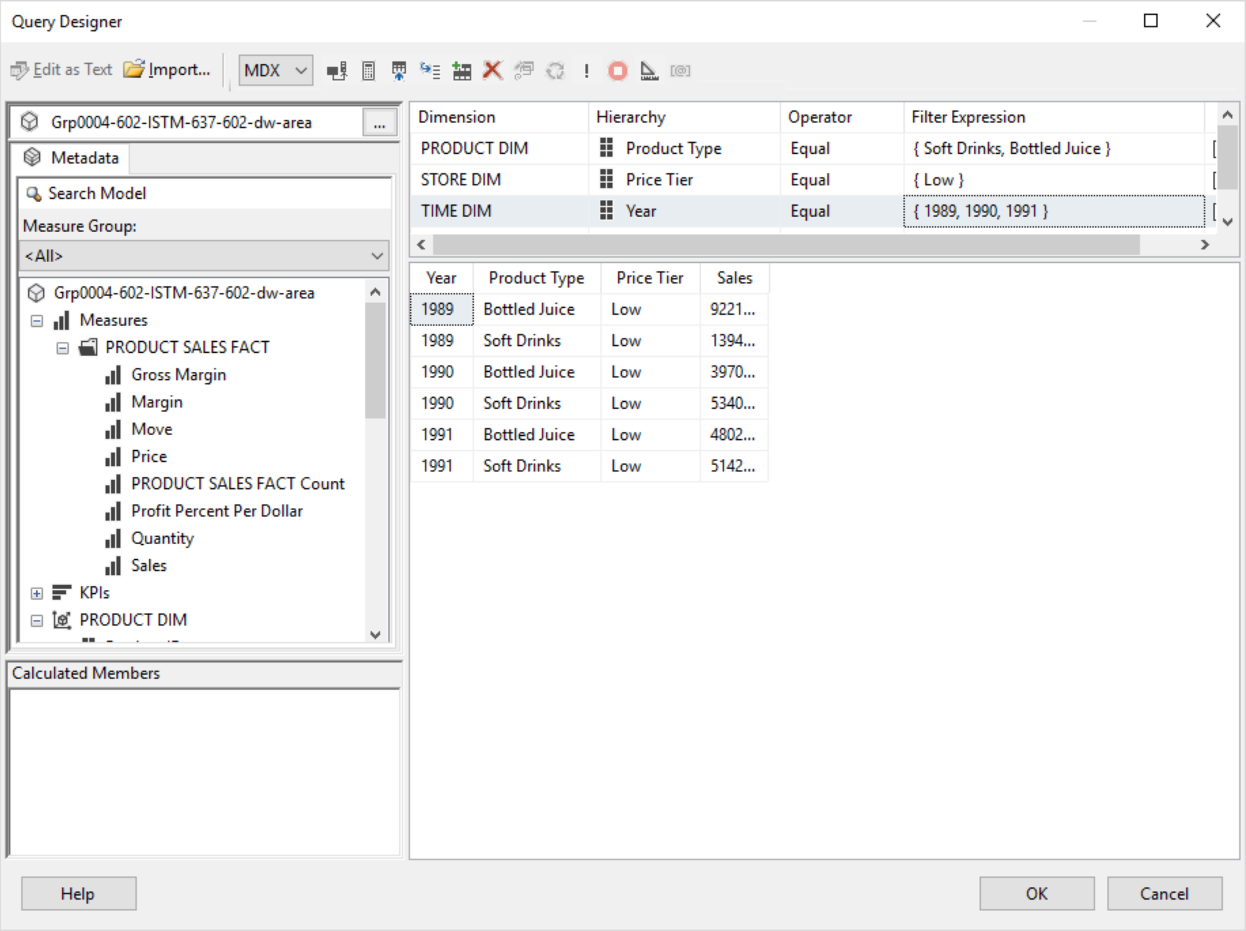
Dimension	Hierarchy	Operator	Filter Expression
PRODUCT DIM	Product Type	Equal	{ Bottled Juice, Soft Drinks }
STORE DIM	Price Tier	Equal	{ High, Low, Medium }
TIME DIM	Year	Equal	{ 1989, 1990, 1991 }

Year	Product Type	Price Tier	Sales
1989	Bottled Juice	High	2692...
1989	Bottled Juice	Low	9221...
1989	Bottled Juice	Medium	4356...
1989	Soft Drinks	High	2428...
1989	Soft Drinks	Low	1394...
1989	Soft Drinks	Medium	4471...
1990	Bottled Juice	High	1083...
1990	Bottled Juice	Low	3970...
1990	Bottled Juice	Medium	1691...
1990	Soft Drinks	High	1002...
1990	Soft Drinks	Low	5340...
1990	Soft Drinks	Medium	1699...
1991	Bottled Juice	High	1335...
1991	Bottled Juice	Low	4802...
1991	Bottled Juice	Medium	2072...
1991	Soft Drinks	High	1036...
1991	Soft Drinks	Low	5142...
1991	Soft Drinks	Medium	1632...



Dimension	Hierarchy	Operator	Filter Expression
PRODUCT DIM	Product Type	Equal	{ Bottled Juice, Soft Drinks }
STORE DIM	Price Tier	Equal	{ High, Low, Medium }
TIME DIM	Year	Equal	{ 1989, 1990, 1991 }

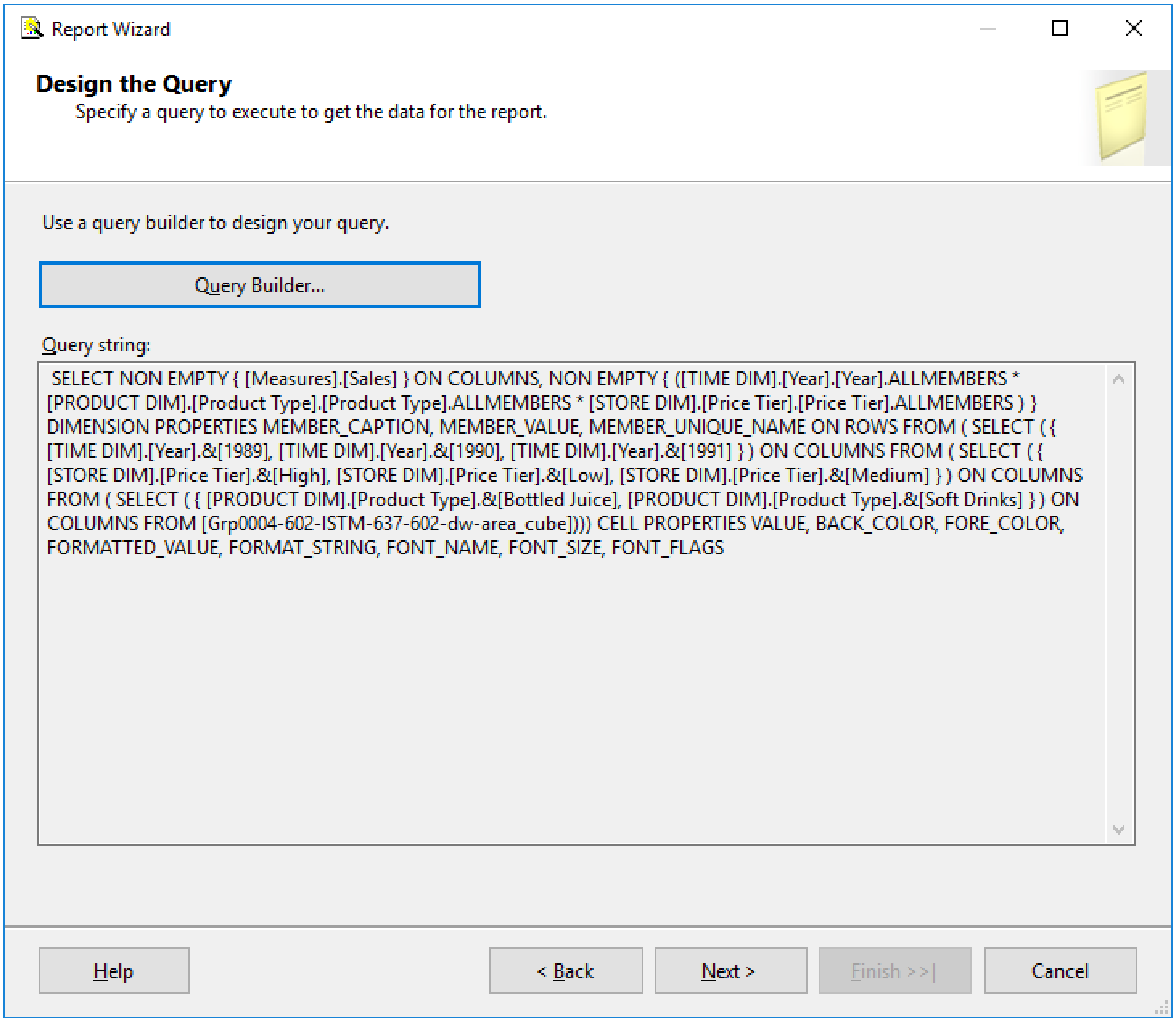
Year	Product Type	Price Tier	Sales
1989	Bottled Juice	High	2692...
1989	Bottled Juice	Low	9221...
1989	Bottled Juice	Medium	4356...
1989	Soft Drinks	High	2428...
1989	Soft Drinks	Low	1394...
1989	Soft Drinks	Medium	4471...
1990	Bottled Juice	High	1083...
1990	Bottled Juice	Low	3970...
1990	Bottled Juice	Medium	1691...
1990	Soft Drinks	High	1002...
1990	Soft Drinks	Low	5340...
1990	Soft Drinks	Medium	1699...
1991	Bottled Juice	High	1335...
1991	Bottled Juice	Low	4802...
1991	Bottled Juice	Medium	2072...
1991	Soft Drinks	High	1036...
1991	Soft Drinks	Low	5142...
1991	Soft Drinks	Medium	1632...



Measure Group: <All>

Dimension	Hierarchy	Operator	Filter Expression
PRODUCT DIM	Product Type	Equal	{ Soft Drinks, Bottled Juice }
STORE DIM	Price Tier	Equal	{ Low }
TIME DIM	Year	Equal	{ 1989, 1990, 1991 }

Year	Product Type	Price Tier	Sales
1989	Bottled Juice	Low	9221...
1989	Soft Drinks	Low	1394...
1990	Bottled Juice	Low	3970...
1990	Soft Drinks	Low	5340...
1991	Bottled Juice	Low	4802...
1991	Soft Drinks	Low	5142...



Design the Query

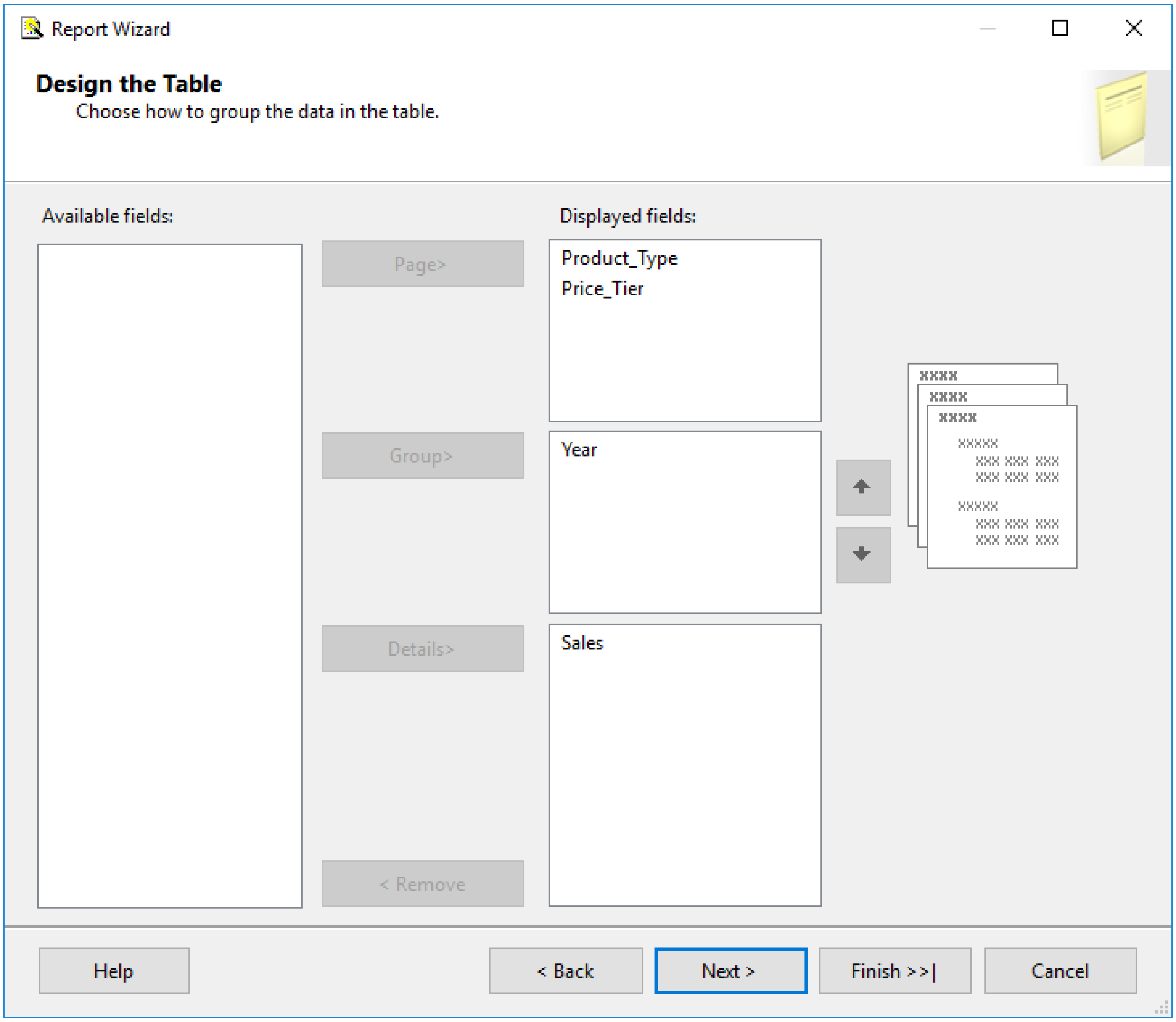
Specify a query to execute to get the data for the report.

Use a query builder to design your query.

Query string:

```
SELECT NON EMPTY { [Measures].[Sales] } ON COLUMNS, NON EMPTY { ([TIME DIM].[Year].[Year].ALLMEMBERS *
[PRODUCT DIM].[Product Type].[Product Type].ALLMEMBERS * [STORE DIM].[Price Tier].[Price Tier].ALLMEMBERS ) }
DIMENSION PROPERTIES MEMBER_CAPTION, MEMBER_VALUE, MEMBER_UNIQUE_NAME ON ROWS FROM ( SELECT ( {
[TIME DIM].[Year].&[1989], [TIME DIM].[Year].&[1990], [TIME DIM].[Year].&[1991] } ) ON COLUMNS FROM ( SELECT ( {
[STORE DIM].[Price Tier].&[High], [STORE DIM].[Price Tier].&[Low], [STORE DIM].[Price Tier].&[Medium] } ) ON COLUMNS
FROM ( SELECT ( { [PRODUCT DIM].[Product Type].&[Bottled Juice], [PRODUCT DIM].[Product Type].&[Soft Drinks] } ) ON
COLUMNS FROM [Grp0004-602-ISTM-637-602-dw-area_cube]))) CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR,
FORMATTED_VALUE, FORMAT_STRING, FONT_NAME, FONT_SIZE, FONT_FLAGS
```

Table Design



Report Wizard

Design the Table

Choose how to group the data in the table.

Available fields:

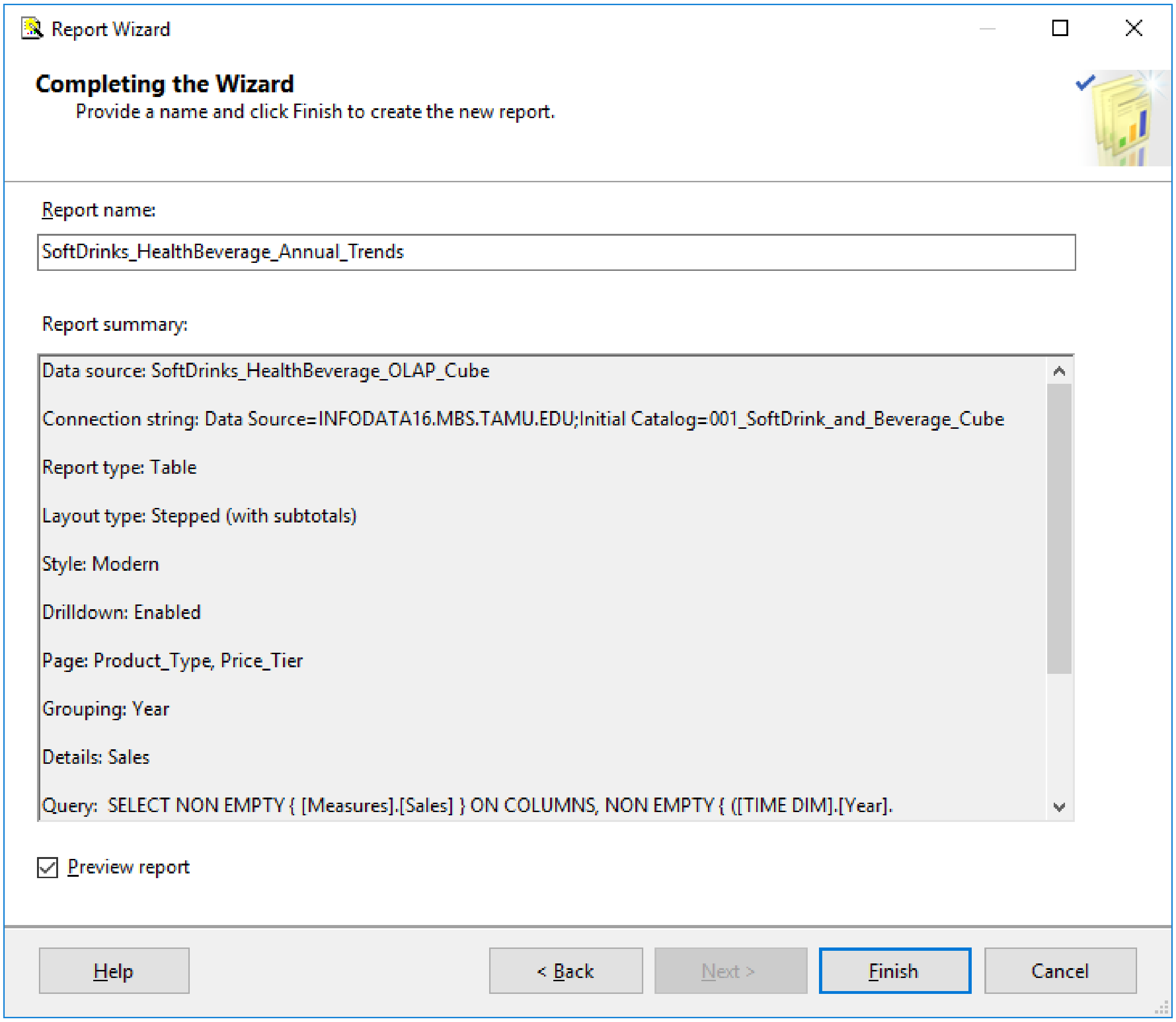
Displayed fields:

Page>: Product_Type, Price_Tier

Group>: Year

Details>: Sales

Final Report



Completing the Wizard

Provide a name and click Finish to create the new report.

Report name:

SoftDrinks_HealthBeverage_Annual_Trends

Report summary:

Data source: SoftDrinks_HealthBeverage_OLAP_Cube

Connection string: Data Source=INFODATA16.MBS.TAMU.EDU;Initial Catalog=001_SoftDrink_and_Beverage_Cube

Report type: Table

Layout type: Stepped (with subtotals)

Style: Modern

Drilldown: Enabled

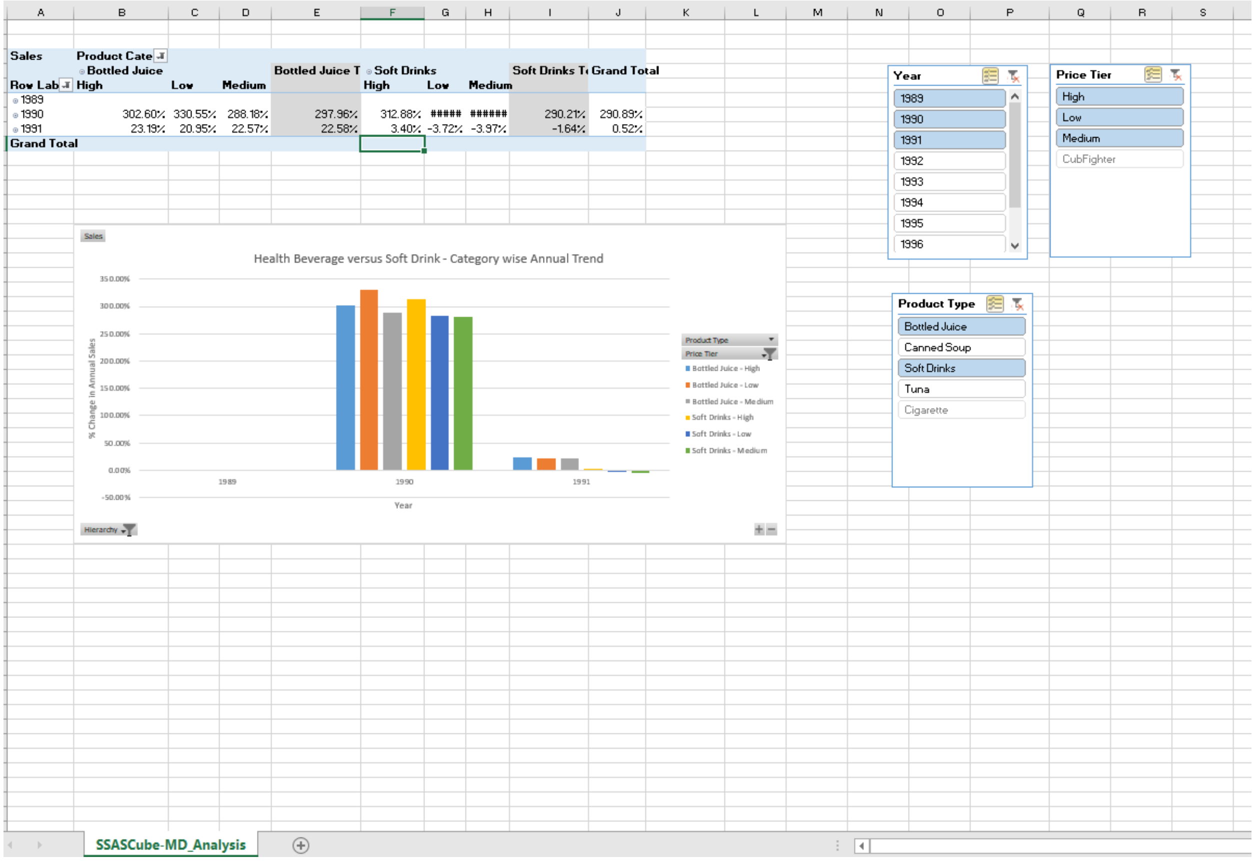
Page: Product_Type, Price_Tier

Grouping: Year

Details: Sales

Query: SELECT NON EMPTY { [Measures].[Sales] } ON COLUMNS, NON EMPTY { ([TIME DIM].[Year].

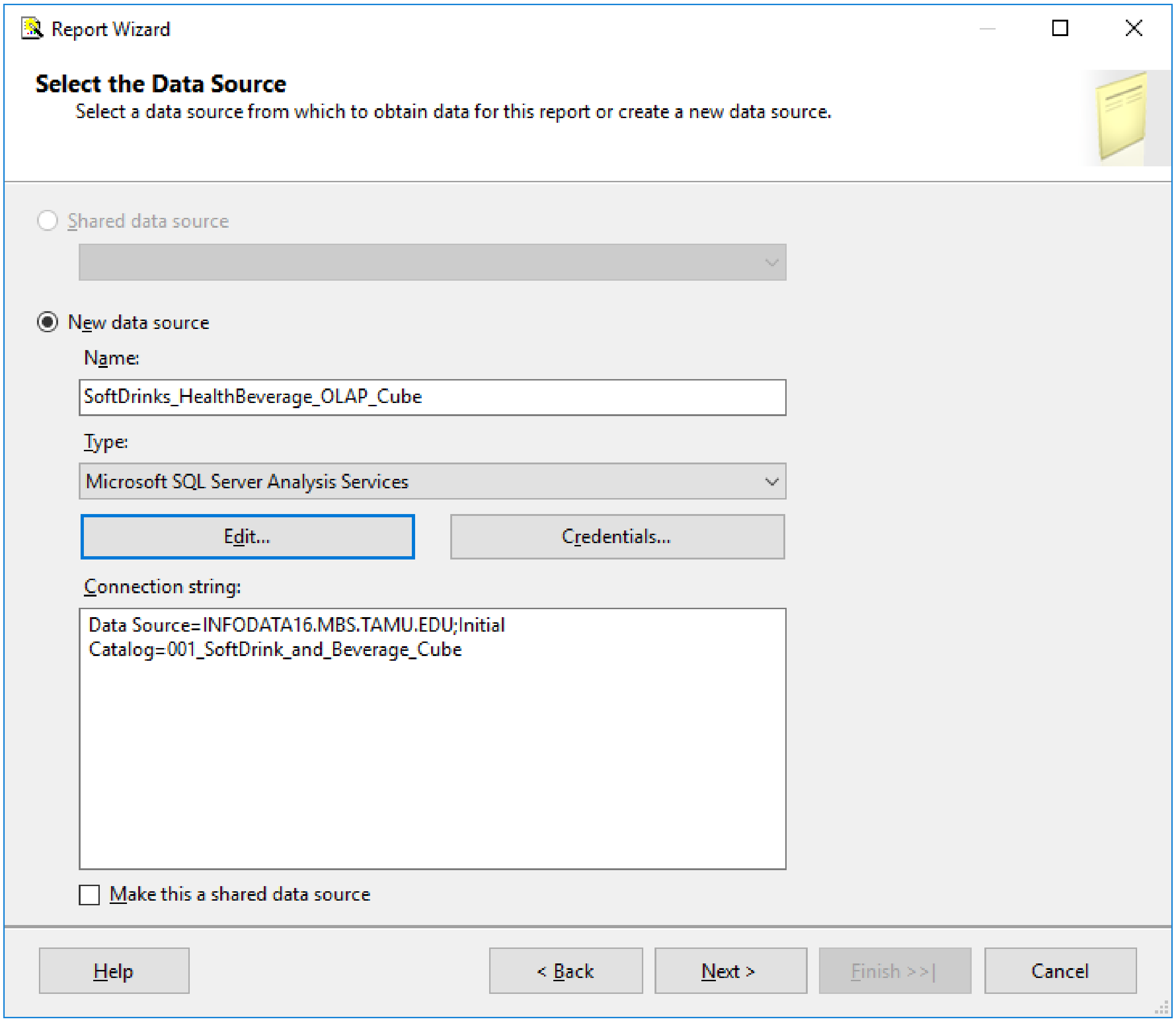
☑ Preview report



Sales	Product Cate								
	Bottled Juice			Bottled Juice T	Soft Drinks			Soft Drinks T	Grand Total
Row Lab	High	Low	Medium		High	Low	Medium		
1989									
1990	302.60%	330.55%	288.18%	297.96%	312.88%	#####	######	290.21%	290.89%
1991	23.19%	20.95%	22.57%	22.58%	3.40%	-3.72%	-3.97%	-1.64%	0.52%
Grand Total									

6.4.2 SSRS on top of SSAS

Connect to Cube



Select the Data Source

Select a data source from which to obtain data for this report or create a new data source.

○ Shared data source

● New data source

Name:

SoftDrinks_HealthBeverage_OLAP_Cube

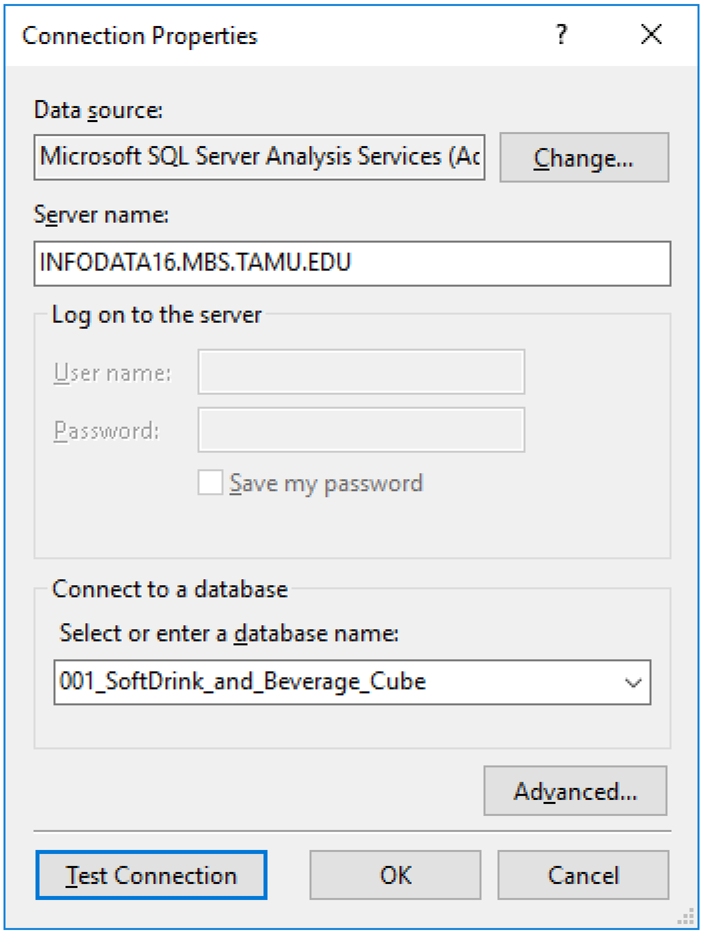
Type:

Microsoft SQL Server Analysis Services

Connection string:

```
Data Source=INFODATA16.MBS.TAMU.EDU;Initial
Catalog=001_SoftDrink_and_Beverage_Cube
```

□ Make this a shared data source



Data source: Microsoft SQL Server Analysis Services (Ac

Server name: INFODATA16.MBS.TAMU.EDU

User name:

Password:

□ Save my password

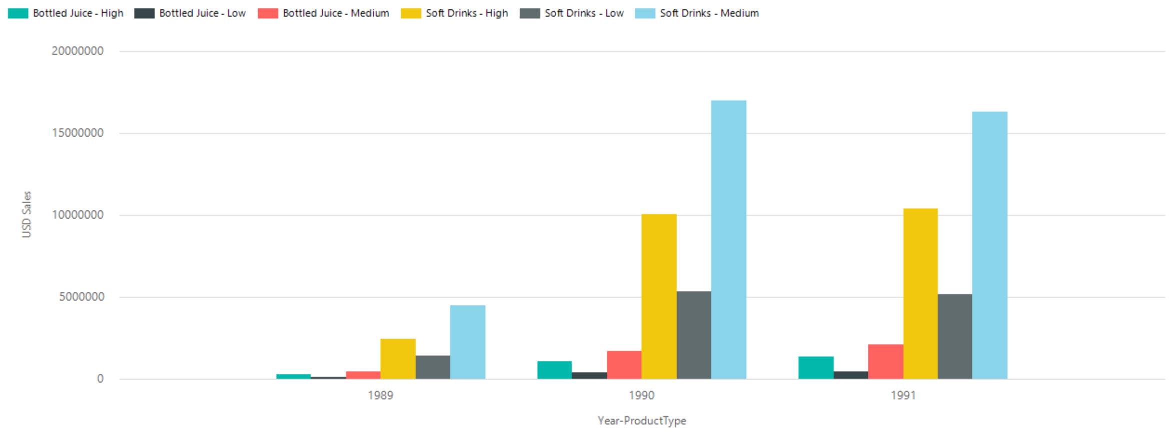
Select or enter a database name: 001_SoftDrink_and_Beverage_Cube

Exported Graph

SoftDrinks HealthBeverage Analysis

Year	Price Tier	Product Type	Sales
1989	High	Bottled Juice	
			269226.93
			92214.83
			435638.16
1990	High	Bottled Juice	
			1083917.14
			397032.29
			1691071.36
1991	High	Bottled Juice	
			1335324.48
			480222.93
			2072752.00

Health Beverage versus Softdrink - Category Wise Annual Trend



Conclusion

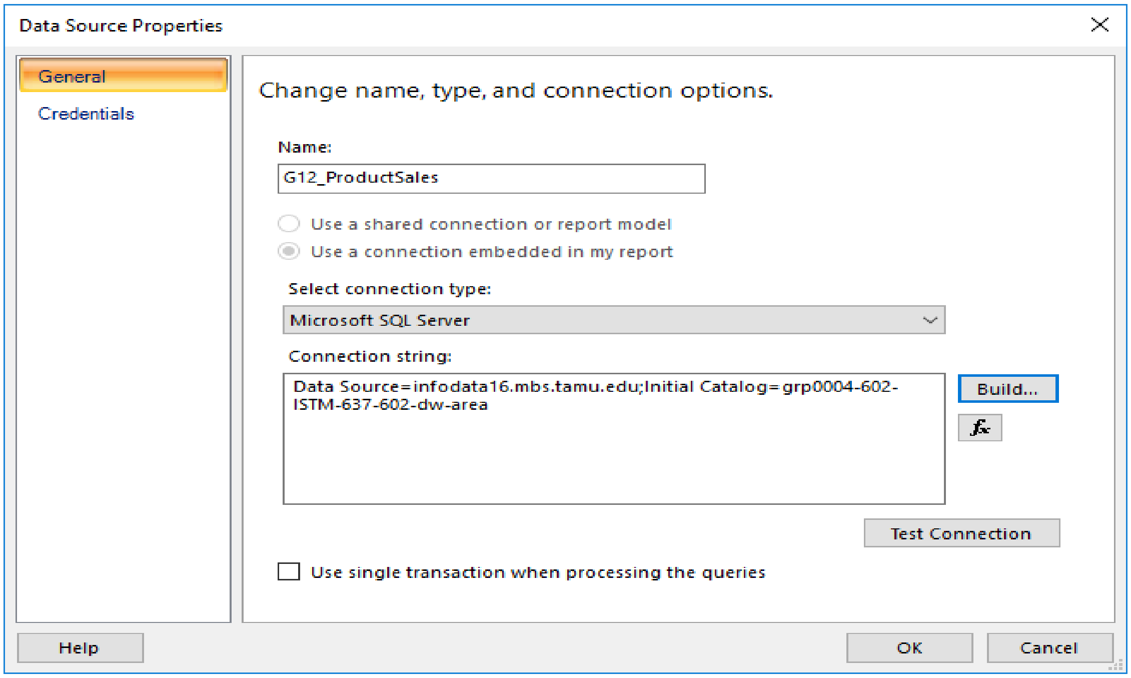
Based on the graphs we created using SSAS and SSRS on top of SSRS, generally the sales of Soft Drinks and Bottle Juice are both increasing from year 1989 to 1991, yet the increase ratio from 1989 to 1990 is larger than that from 1990 to 1991. Besides, the sale of Soft Drinks is much higher than Bottle Juice in each tire-level of stores. Only Soft Drinks in Medium Tire Store has slightly negative growth rate from 1990 to 1991. About the sales of Bottle Juice in Low Tire, it is increasing from 1989 to 1991. Comparatively, the obvious increase happens in 1990 and the sale slightly increases from 1990 to 1991. Such contrast sale trend analysis in the close category enables Dominick Finer Foods to adjust the inventory according to customers' need. Like in this case, the sale of Soft Drinks clearly beats the sale of Bottled Juice, thus DFF can provide more Soft Drinks and also initiate some promotion to enhance the sales of Bottle Juice.

6.5 ReportBuilder3.0 for BQ-4

Based on the report plan, ReportBuilder3.0 is adopted to create visualization for business question 4.

BQ-4: What effect does the time of the year have on sales of each Tuna? Determine the impact of seasonality on Average Sales of Tuna from December-1990 to September 1992?

Data Source Connection



Data Source Properties

Change name, type, and connection options.

Name: G12_ProductSales

○ Use a shared connection or report model
● Use a connection embedded in my report

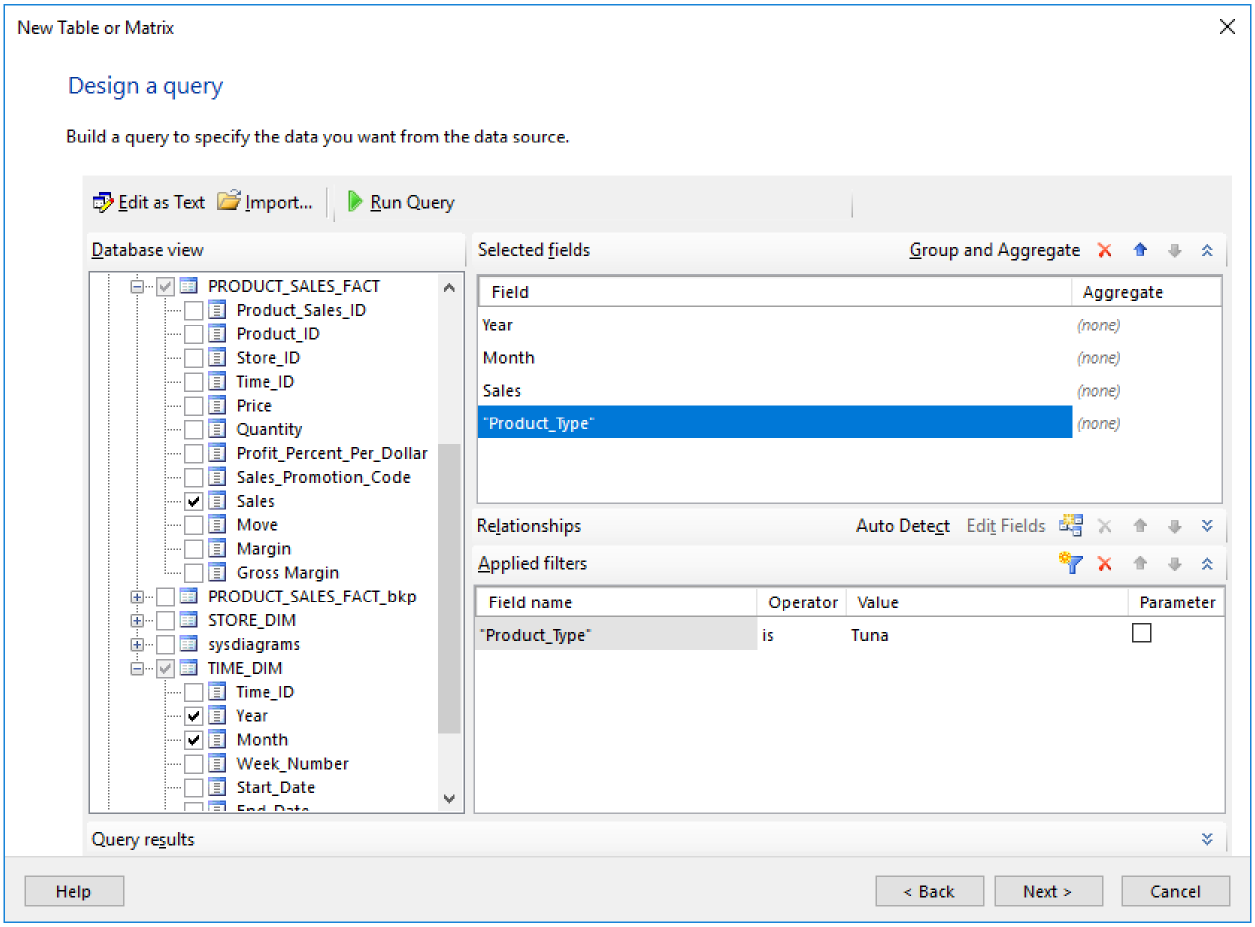
Select connection type: Microsoft SQL Server

Connection string:

```
Data Source=infodata16.mbs.tamu.edu;Initial Catalog=grp0004-602-
ISTM-637-602-dw-area
```

□ Use single transaction when processing the queries

Query Design



New Table or Matrix

Design a query

Build a query to specify the data you want from the data source.

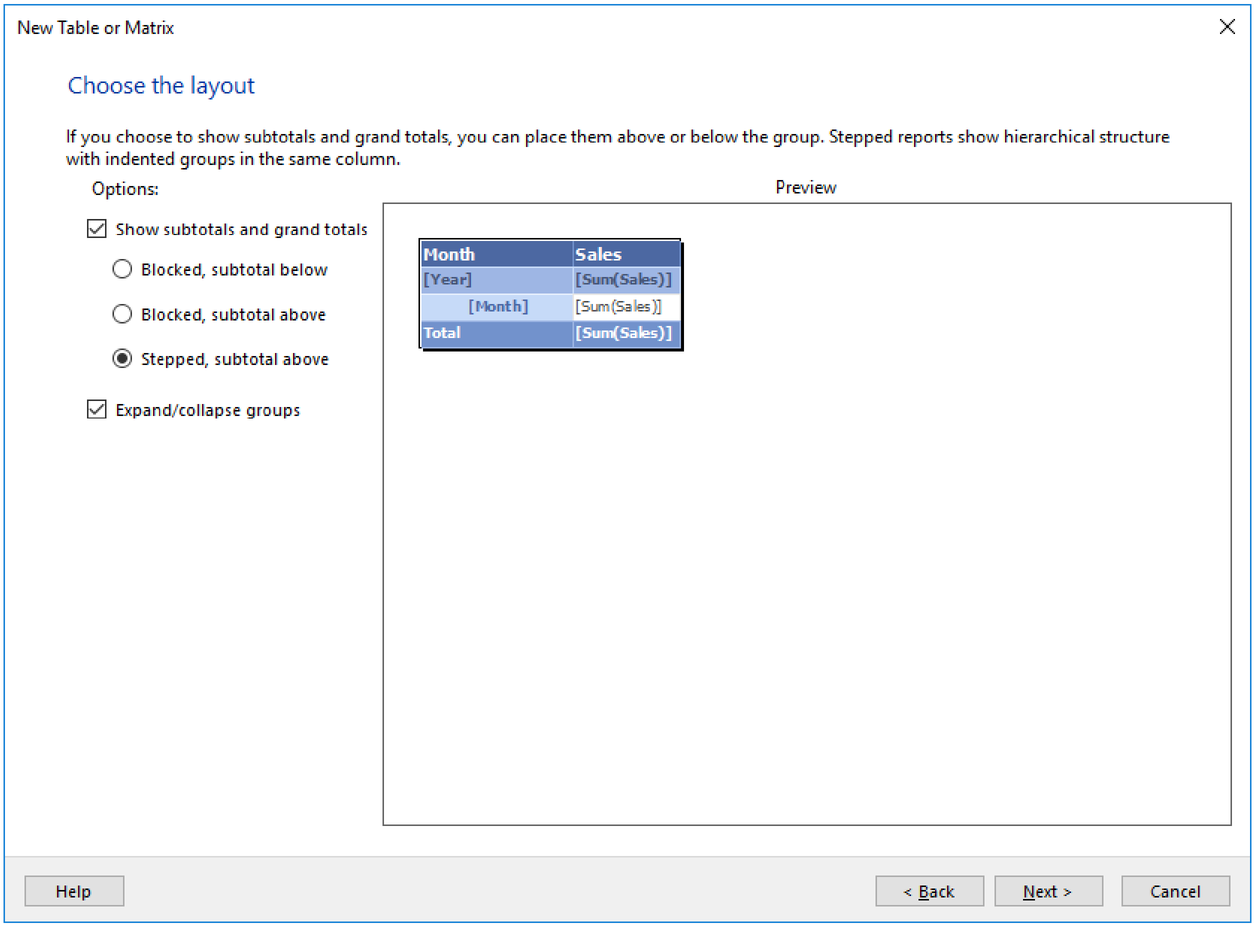
Selected fields:

Field	Aggregate
Year	(none)
Month	(none)
Sales	(none)
"Product_Type"	(none)

Applied filters:

Field name	Operator	Value	Parameter
"Product_Type"	is	Tuna	□

Choose Layout



New Table or Matrix

Choose the layout

If you choose to show subtotals and grand totals, you can place them above or below the group. Stepped reports show hierarchical structure with indented groups in the same column.

Options:

☑ Show subtotals and grand totals

○ Blocked, subtotal below

○ Blocked, subtotal above

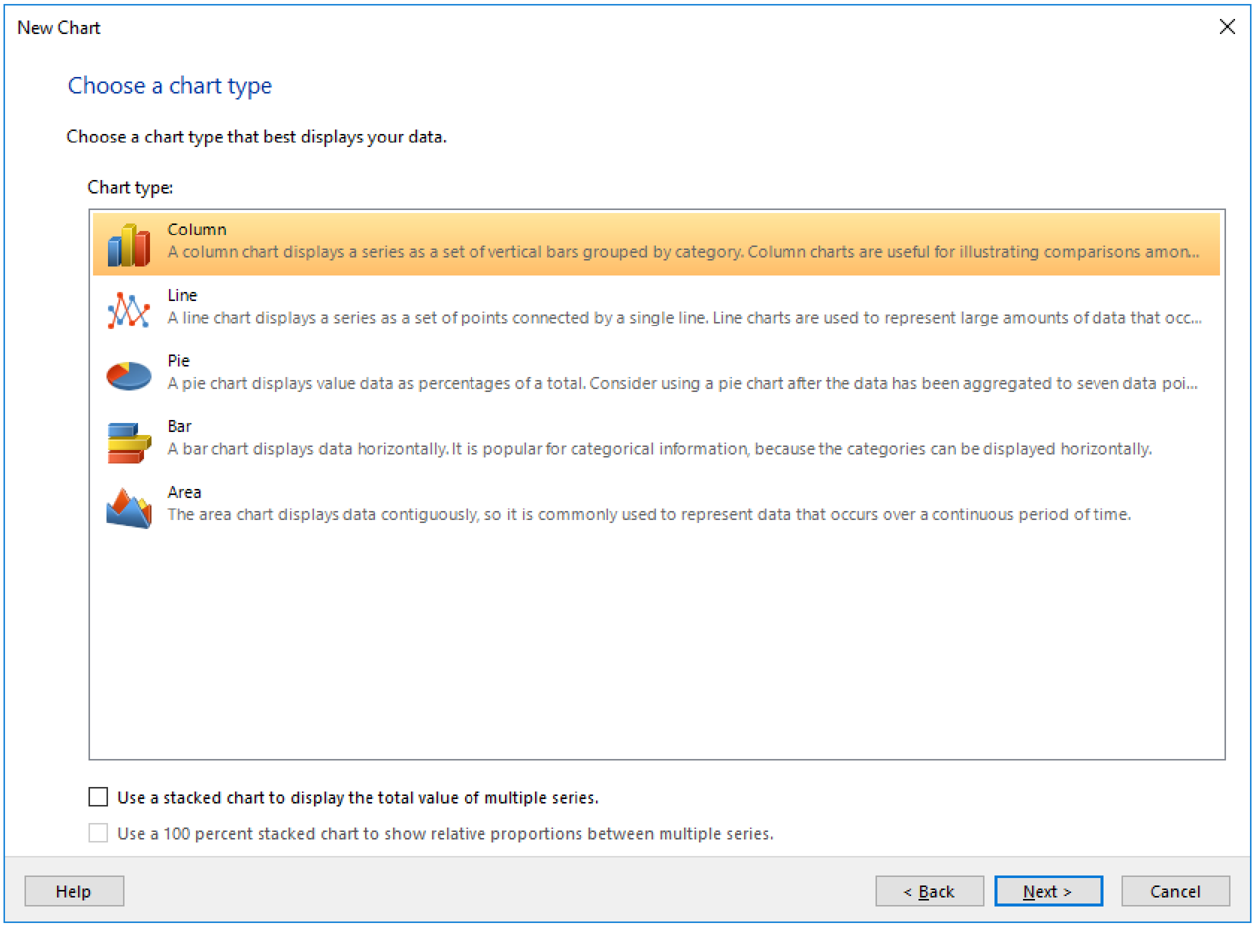
● Stepped, subtotal above

☑ Expand/collapse groups

Preview

Month	Sales
[Year]	[Sum(Sales)]
[Month]	[Sum(Sales)]
Total	[Sum(Sales)]

Chart Design



New Chart

Choose a chart type

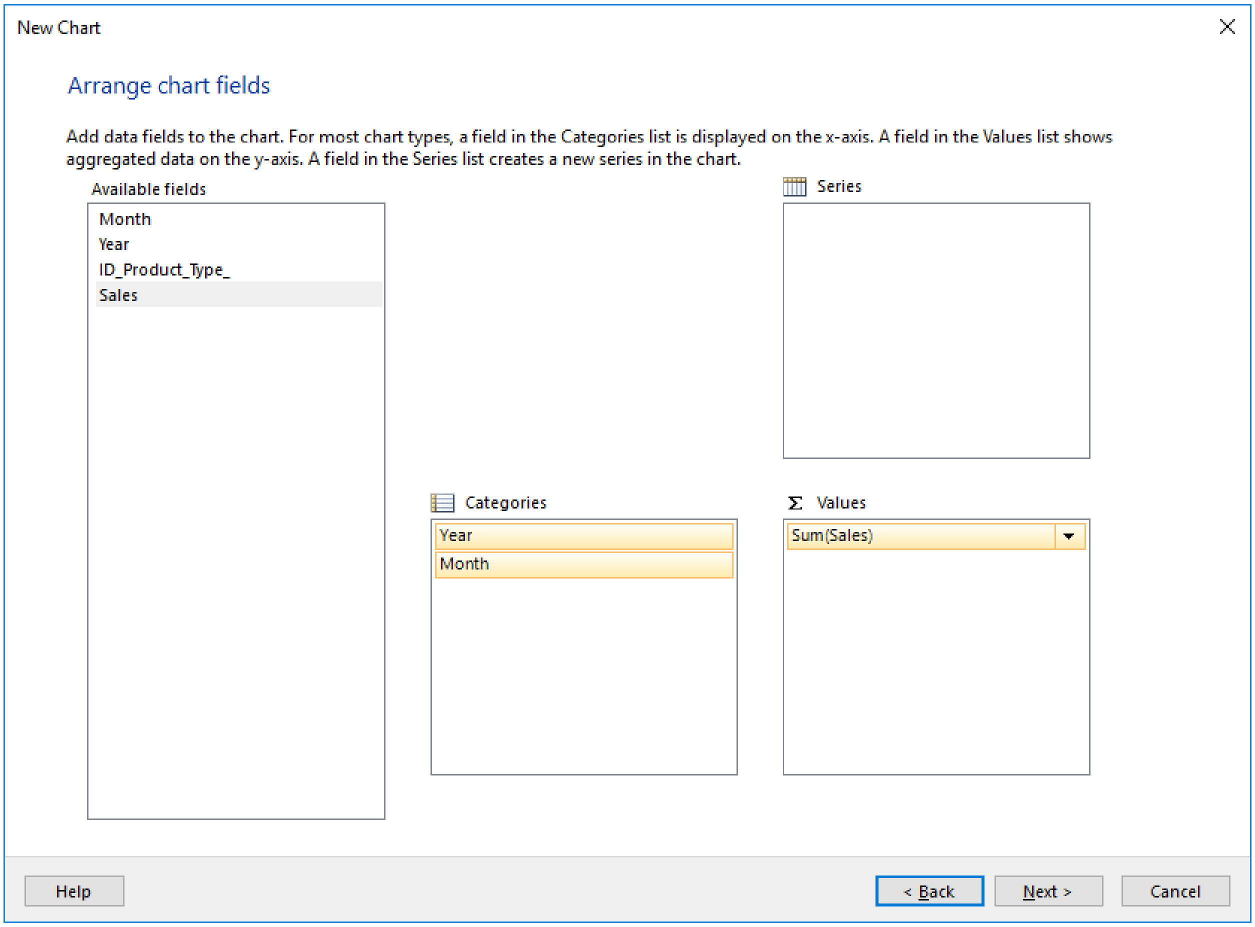
Choose a chart type that best displays your data.

Chart type:

- Column
  A column chart displays a series as a set of vertical bars grouped by category. Column charts are useful for illustrating comparisons amon...
- Line
  A line chart displays a series as a set of points connected by a single line. Line charts are used to represent large amounts of data that occ...
- Pie
  A pie chart displays value data as percentages of a total. Consider using a pie chart after the data has been aggregated to seven data poi...
- Bar
  A bar chart displays data horizontally. It is popular for categorical information, because the categories can be displayed horizontally.
- Area
  The area chart displays data contiguously, so it is commonly used to represent data that occurs over a continuous period of time.

☐ Use a stacked chart to display the total value of multiple series.

☐ Use a 100 percent stacked chart to show relative proportions between multiple series.



New Chart

Arrange chart fields

Add data fields to the chart. For most chart types, a field in the Categories list is displayed on the x-axis. A field in the Values list shows aggregated data on the y-axis. A field in the Series list creates a new series in the chart.

Available fields

- Month
- Year
- ID_Product_Type_
- Sales

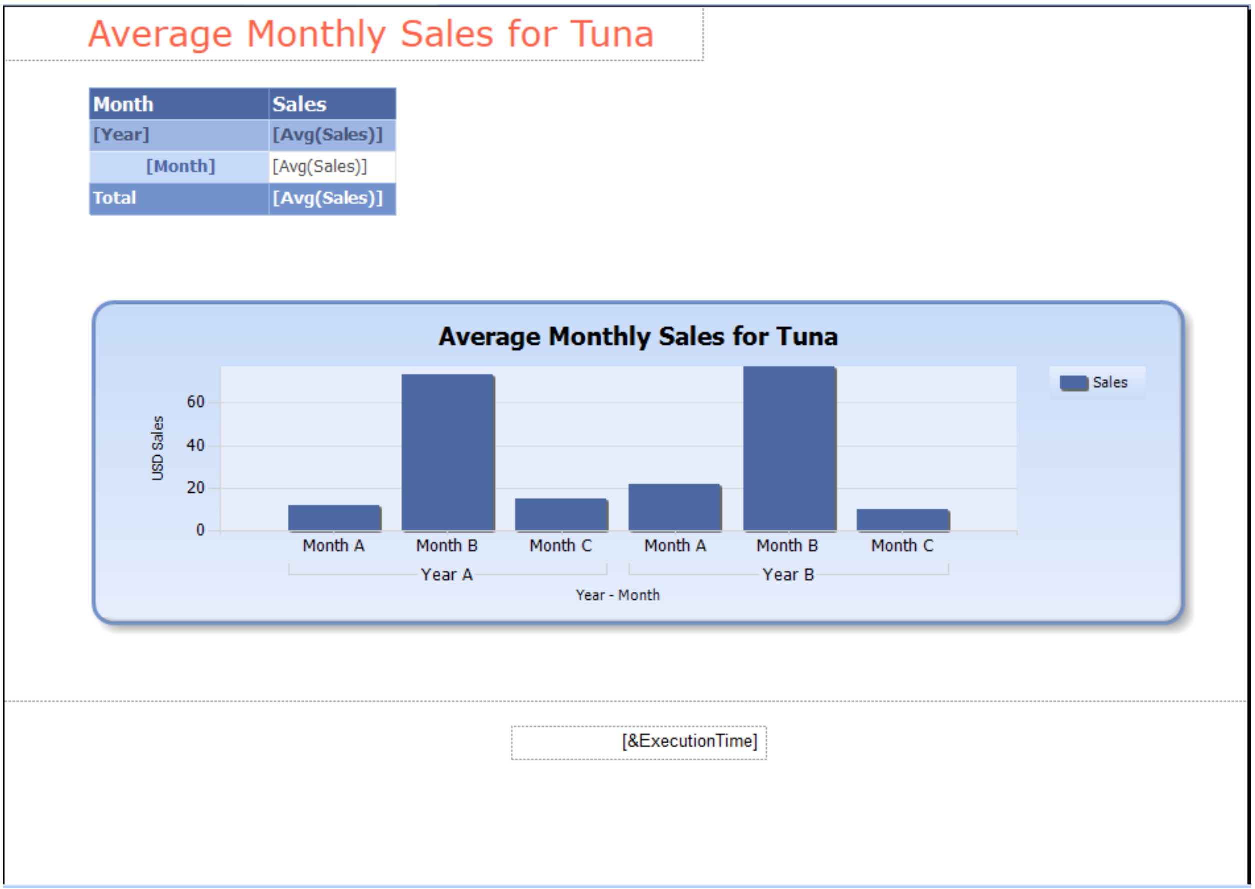
Series

Categories

- Year
- Month

Values

- Sum(Sales)

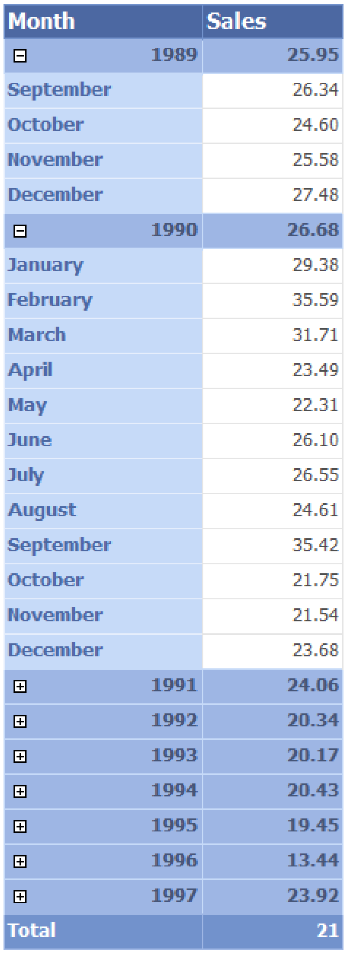


Average Monthly Sales for Tuna

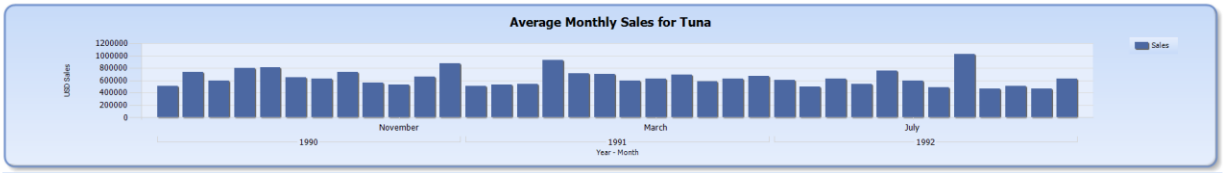
Month	Sales
[Year]	[Avg(Sales)]
[Month]	[Avg(Sales)]
Total	[Avg(Sales)]

[&ExecutionTime]

Average Monthly Sales for Tuna



Month	Sales
⊟ 1989	**25.95**
September	26.34
October	24.60
November	25.58
December	27.48
⊟ 1990	**26.68**
January	29.38
February	35.59
March	31.71
April	23.49
May	22.31
June	26.10
July	26.55
August	24.61
September	35.42
October	21.75
November	21.54
December	23.68
⊞ 1991	**24.06**
⊞ 1992	**20.34**
⊞ 1993	**20.17**
⊞ 1994	**20.43**
⊞ 1995	**19.45**
⊞ 1996	**13.44**
⊞ 1997	**23.92**
Total	**21**



4/29/2018 9:21:46 PM

Conclusion

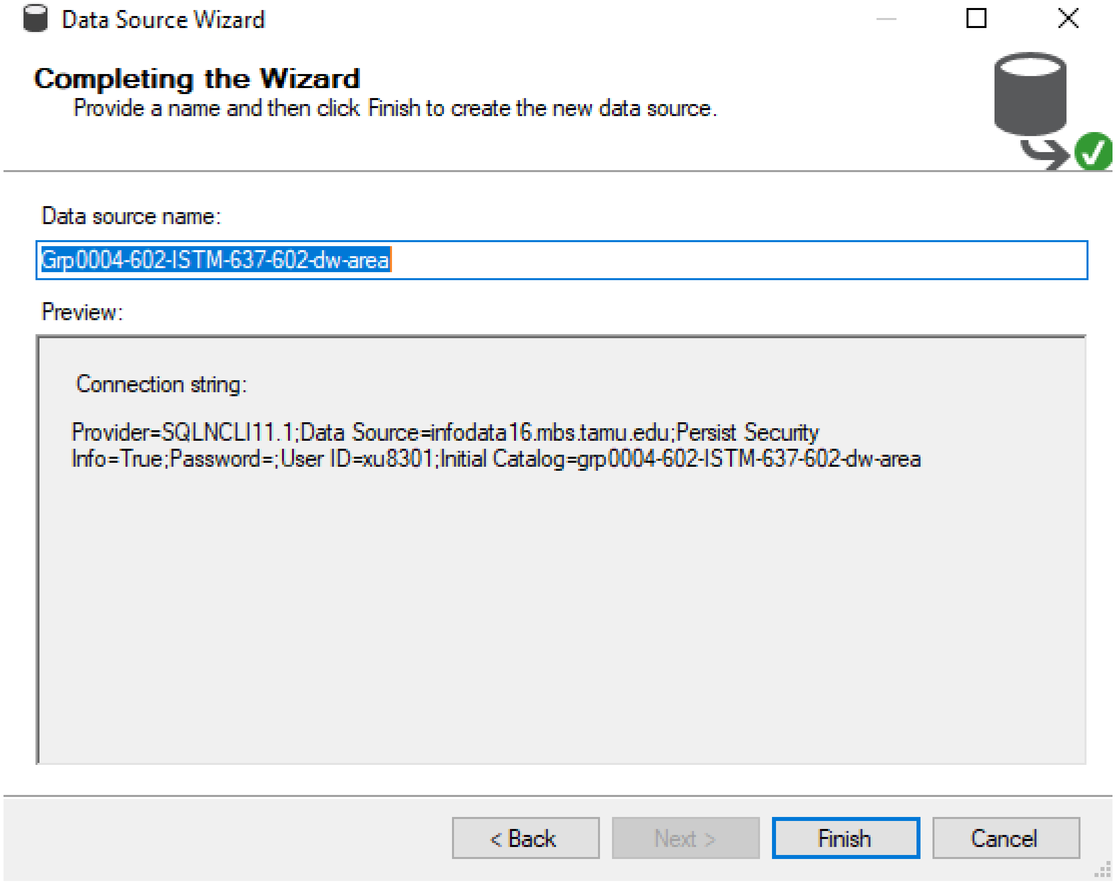
As we can see in the chart, there is a factor of seasonality and cyclicity for Tuna Sales. We see during the 1st quarter of year, especially during the months of January, February and March, there is a surge in the average sales price of Tuna. This pattern repeats itself across years. Knowledge of this trend can help DFF to plan their Tuna inventory, new product launch, discounts and channel rebates to increase the volumes, customer base and margins by adopting proper pricing and bundling strategy. We found Report Builder 3.0 to be a useful tool for this report. It aligned with the needs for drill down reports.

6.6 SSAS for BQ-5

Based on the report plan, SSAS is adopted to create visualization for business question 5.

BQ-5: What is annual growth rate of cigarette margins in low tier stores? What is the decreasing trend?

Connect to Database



Data Source Wizard

Completing the Wizard

Provide a name and then click Finish to create the new data source.

Data source name:

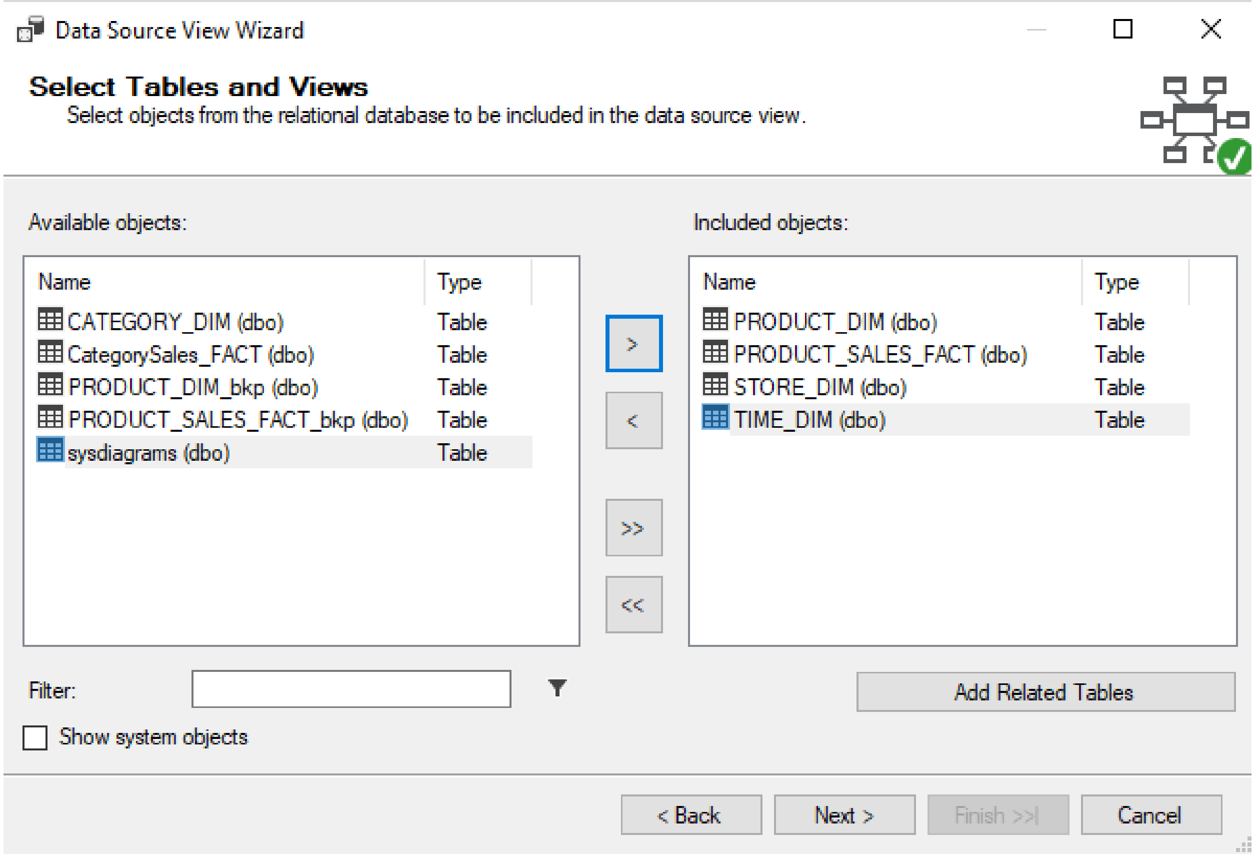
Grp0004-602-ISTM-637-602-dw-area

Preview:

Connection string:

Provider=SQLNCLI11.1;Data Source=infodata16.mbs.tamu.edu;Persist Security Info=True;Password=;User ID=xu8301;Initial Catalog=grp0004-602-ISTM-637-602-dw-area

Create Data Source View



Data Source View Wizard

Select Tables and Views

Select objects from the relational database to be included in the data source view.

Available objects:

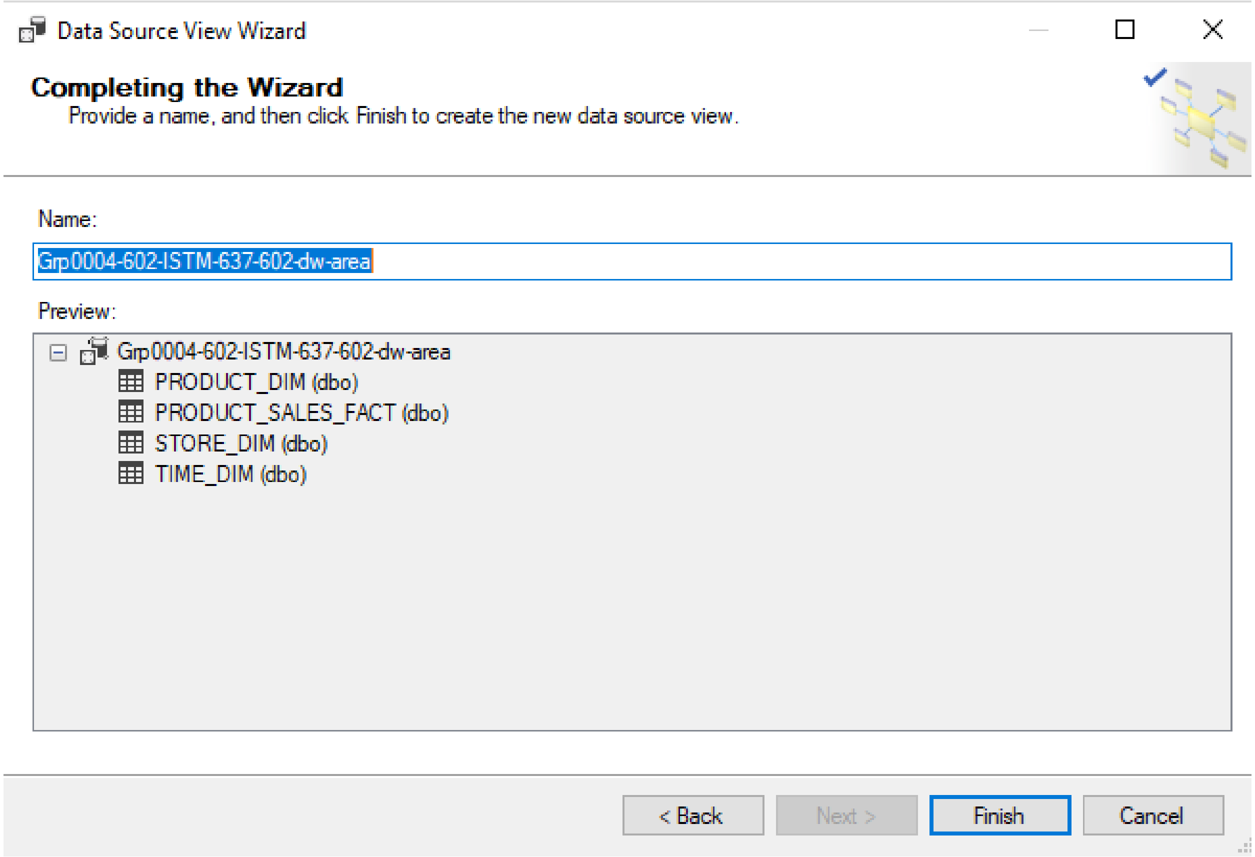
Name	Type
CATEGORY_DIM (dbo)	Table
CategorySales_FACT (dbo)	Table
PRODUCT_DIM_bkp (dbo)	Table
PRODUCT_SALES_FACT_bkp (dbo)	Table
sysdiagrams (dbo)	Table

Included objects:

Name	Type
PRODUCT_DIM (dbo)	Table
PRODUCT_SALES_FACT (dbo)	Table
STORE_DIM (dbo)	Table
TIME_DIM (dbo)	Table

Filter:

☐ Show system objects



Completing the Wizard

Provide a name, and then click Finish to create the new data source view.

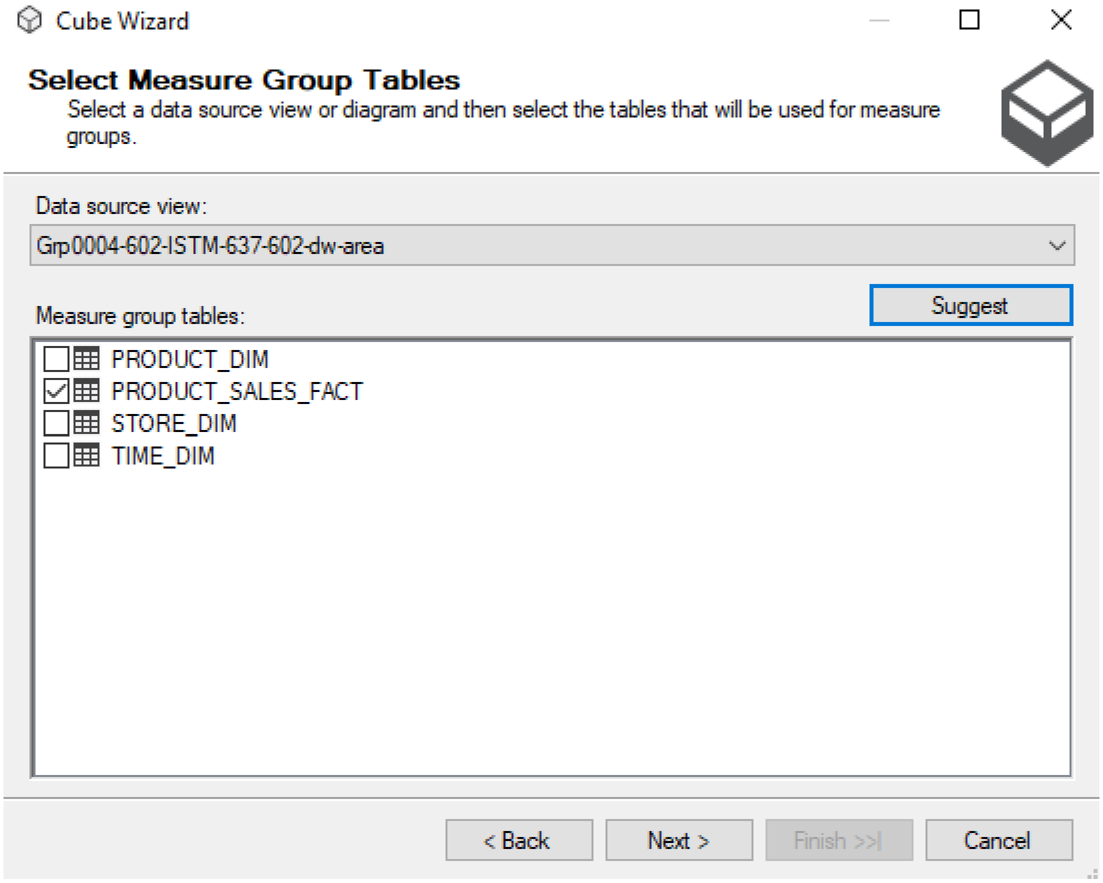
Name:

Grp0004-602-ISTM-637-602-dw-area

Preview:

- Grp0004-602-ISTM-637-602-dw-area
  - PRODUCT_DIM (dbo)
  - PRODUCT_SALES_FACT (dbo)
  - STORE_DIM (dbo)
  - TIME_DIM (dbo)

Create Cube



Cube Wizard

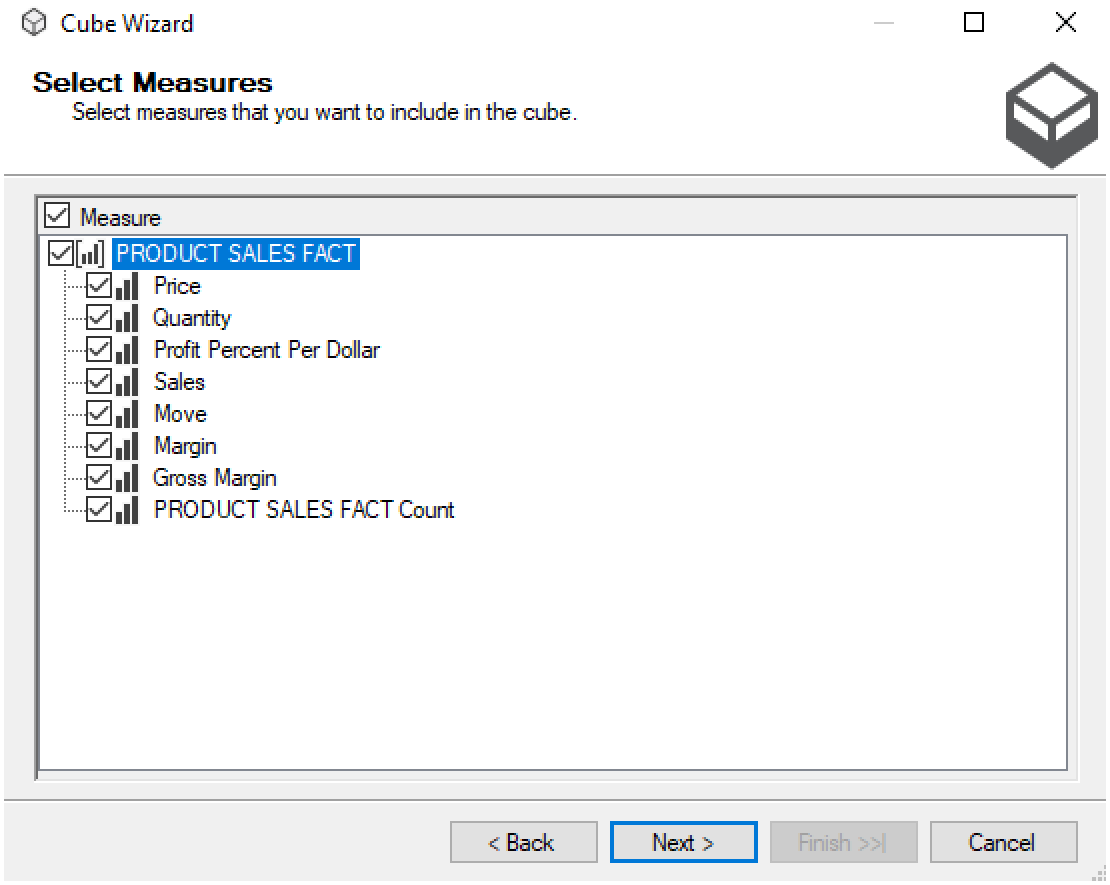
Select Measure Group Tables

Select a data source view or diagram and then select the tables that will be used for measure groups.

Data source view: Grp0004-602-ISTM-637-602-dw-area

Measure group tables:

- ☐ PRODUCT_DIM
- ☑ PRODUCT_SALES_FACT
- ☐ STORE_DIM
- ☐ TIME_DIM

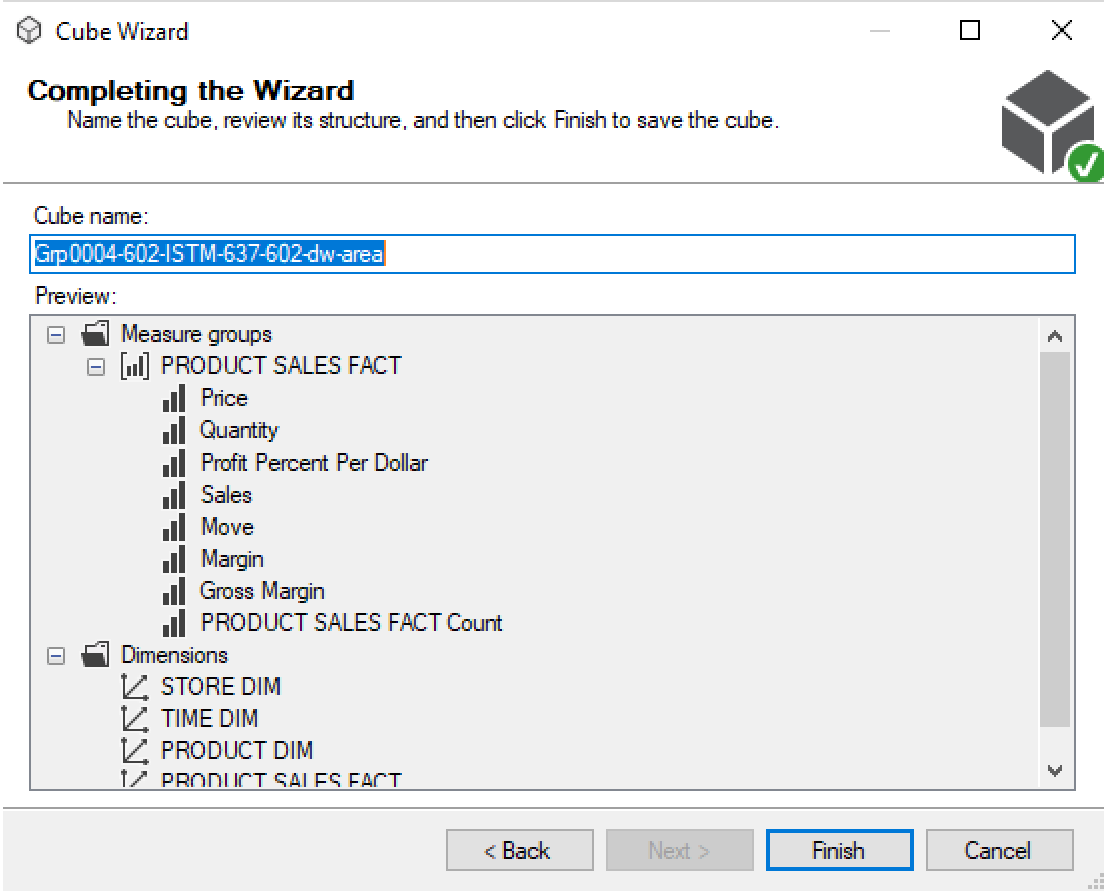


Select Measures

Select measures that you want to include in the cube.

☑ Measure

- ☑ PRODUCT SALES FACT
  - ☑ Price
  - ☑ Quantity
  - ☑ Profit Percent Per Dollar
  - ☑ Sales
  - ☑ Move
  - ☑ Margin
  - ☑ Gross Margin
  - ☑ PRODUCT SALES FACT Count



Cube Wizard

Completing the Wizard

Name the cube, review its structure, and then click Finish to save the cube.

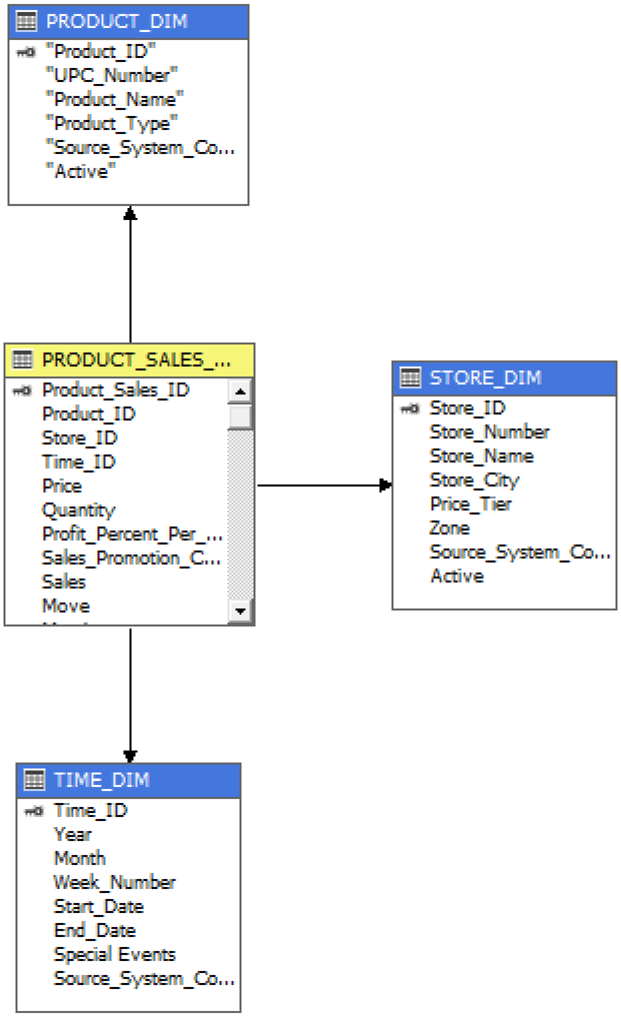
Cube name:

Grp0004-602-ISTM-637-602-dw-area

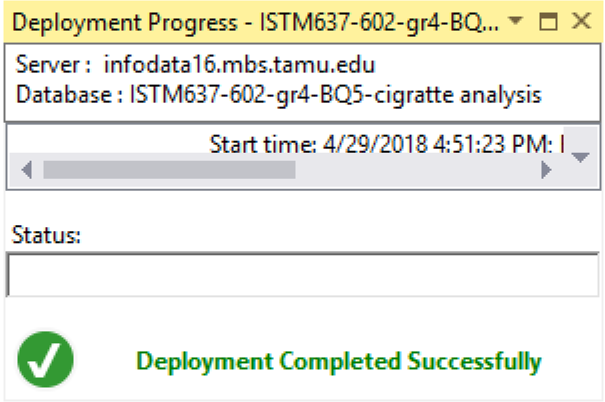
Preview:

- Measure groups
  - PRODUCT SALES FACT
    - Price
    - Quantity
    - Profit Percent Per Dollar
    - Sales
    - Move
    - Margin
    - Gross Margin
    - PRODUCT SALES FACT Count
- Dimensions
  - STORE DIM
  - TIME DIM
  - PRODUCT DIM
  - PRODUCT SALES FACT

Cube Structure



Deployment

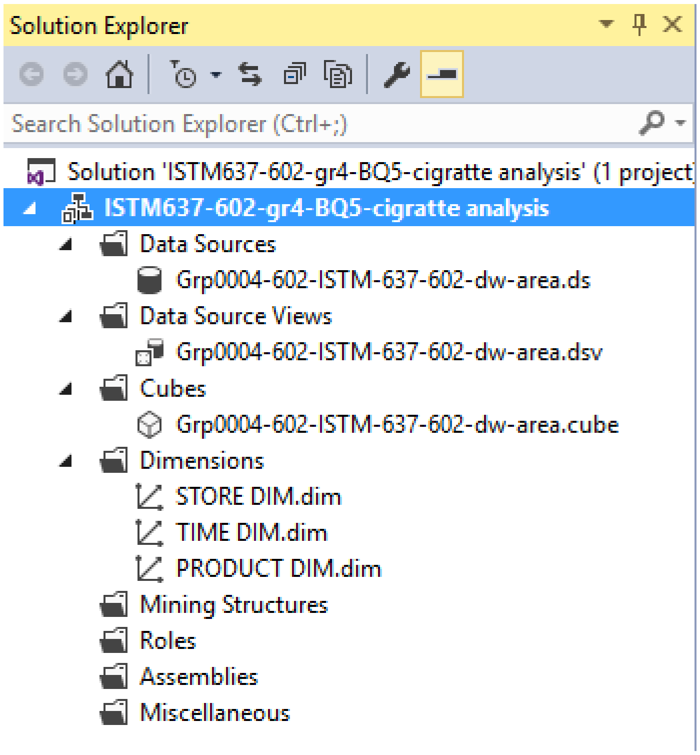

Deployment Progress - ISTM637-602-gr4-BQ...

Server : infodata16.mbs.tamu.edu
Database : ISTM637-602-gr4-BQ5-cigratte analysis

Start time: 4/29/2018 4:51:23 PM; I

Status:

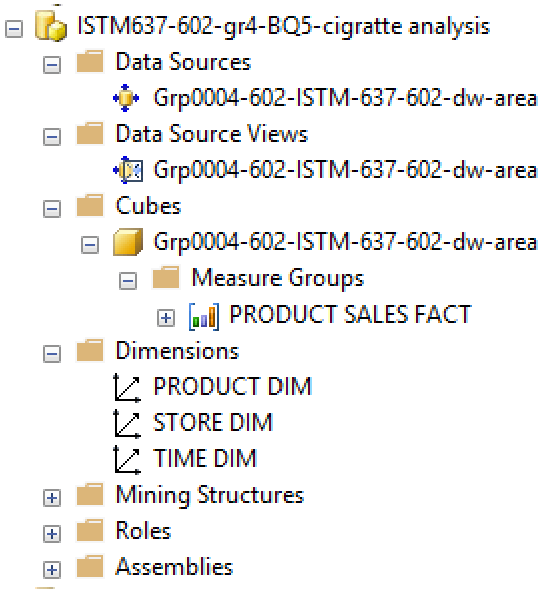
✓ Deployment Completed Successfully



Solution Explorer

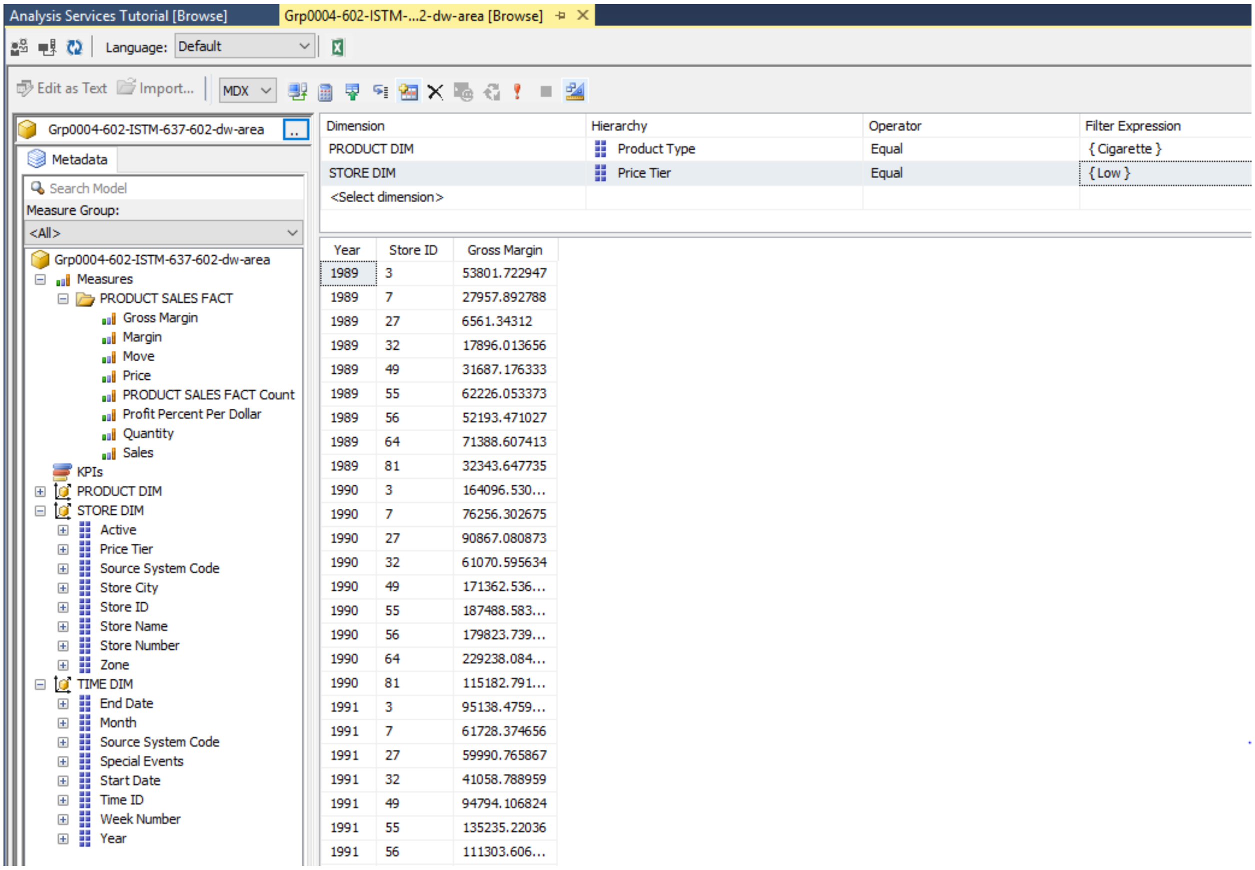
Search Solution Explorer (Ctrl+;)

- Solution 'ISTM637-602-gr4-BQ5-cigratte analysis' (1 project
  - ISTM637-602-gr4-BQ5-cigratte analysis
    - Data Sources
      - Grp0004-602-ISTM-637-602-dw-area.ds
    - Data Source Views
      - Grp0004-602-ISTM-637-602-dw-area.dsv
    - Cubes
      - Grp0004-602-ISTM-637-602-dw-area.cube
    - Dimensions
      - STORE DIM.dim
      - TIME DIM.dim
      - PRODUCT DIM.dim
    - Mining Structures
    - Roles
    - Assemblies
    - Miscellaneous



- ISTM637-602-gr4-BQ5-cigratte analysis
  - Data Sources
    - Grp0004-602-ISTM-637-602-dw-area
  - Data Source Views
    - Grp0004-602-ISTM-637-602-dw-area
  - Cubes
    - Grp0004-602-ISTM-637-602-dw-area
      - Measure Groups
        - PRODUCT SALES FACT
  - Dimensions
    - PRODUCT DIM
    - STORE DIM
    - TIME DIM
  - Mining Structures
  - Roles
  - Assemblies

Query Design



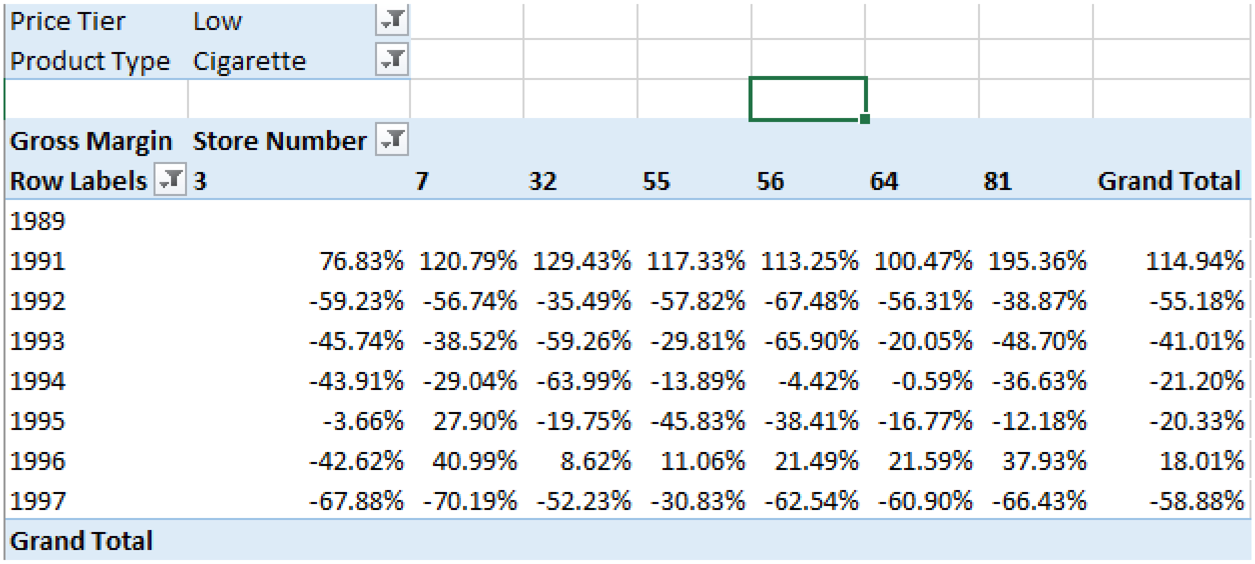
Analysis Services Tutorial [Browse]
Grp0004-602-ISTM-...2-dw-area [Browse]
Language: Default
Edit as Text
Import...
MDX
Grp0004-602-ISTM-637-602-dw-area
Metadata
Search Model
Measure Group:
<All>

```
Grp0004-602-ISTM-637-602-dw-area
  Measures
    PRODUCT SALES FACT
      Gross Margin
      Margin
      Move
      Price
      PRODUCT SALES FACT Count
      Profit Percent Per Dollar
      Quantity
      Sales
  KPIs
  PRODUCT DIM
  STORE DIM
    Active
    Price Tier
    Source System Code
    Store City
    Store ID
    Store Name
    Store Number
    Zone
  TIME DIM
    End Date
    Month
    Source System Code
    Special Events
    Start Date
    Time ID
    Week Number
    Year
```

Dimension	Hierarchy	Operator	Filter Expression
PRODUCT DIM	Product Type	Equal	{ Cigarette }
STORE DIM	Price Tier	Equal	{ Low }
<Select dimension>			

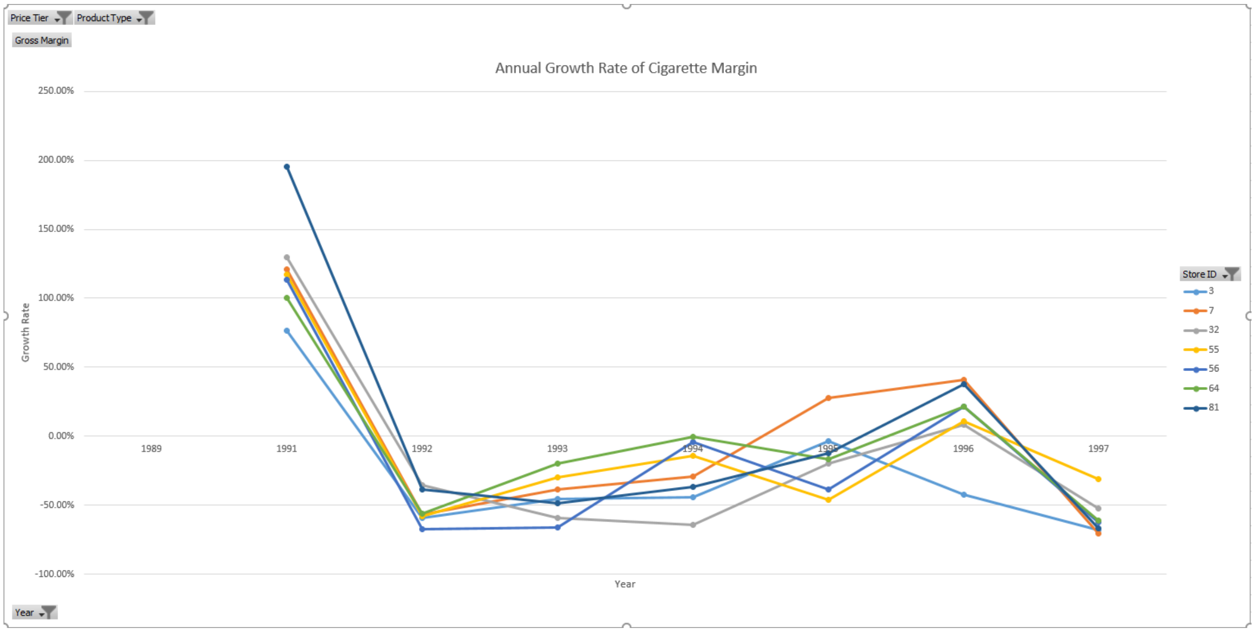
Year	Store ID	Gross Margin
1989	3	53801.722947
1989	7	27957.892788
1989	27	6561.34312
1989	32	17896.013656
1989	49	31687.176333
1989	55	62226.053373
1989	56	52193.471027
1989	64	71388.607413
1989	81	32343.647735
1990	3	164096.530...
1990	7	76256.302675
1990	27	90867.080873
1990	32	61070.595634
1990	49	171362.536...
1990	55	187488.583...
1990	56	179823.739...
1990	64	229238.084...
1990	81	115182.791...
1991	3	95138.4759...
1991	7	61728.374656
1991	27	59990.765867
1991	32	41058.788959
1991	49	94794.106824
1991	55	135235.22036
1991	56	111303.606...

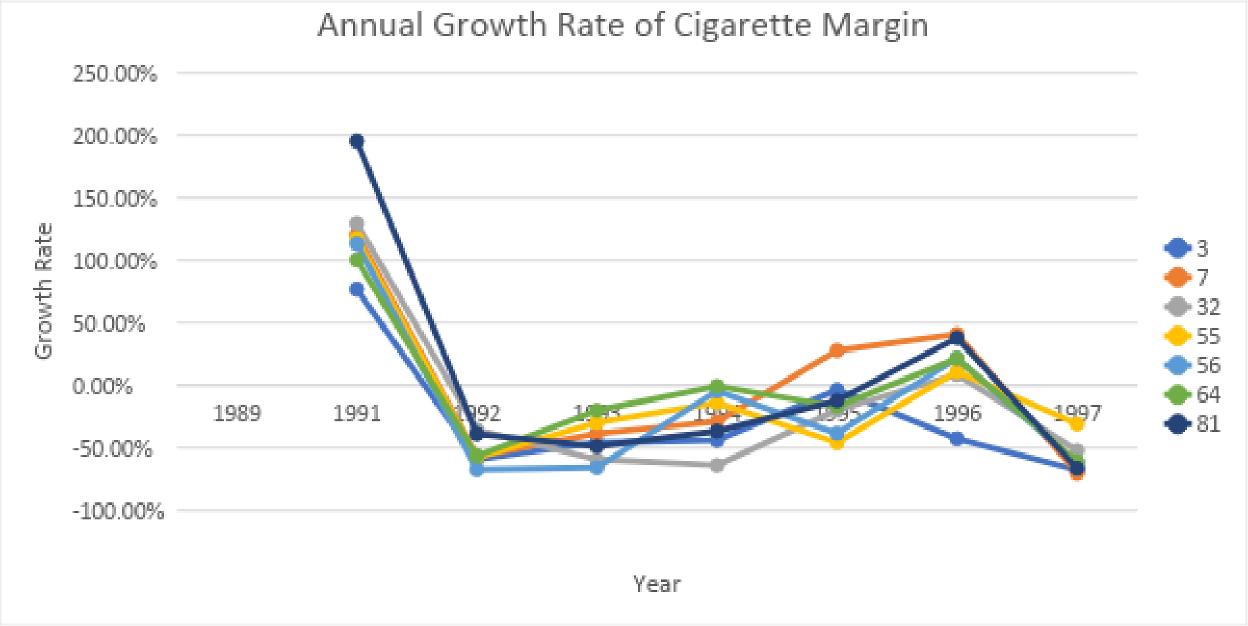
Exported Excel



Price Tier	Low							
Product Type	Cigarette							
Gross Margin	Store Number							
Row Labels	3	7	32	55	56	64	81	Grand Total
1989								
1991	76.83%	120.79%	129.43%	117.33%	113.25%	100.47%	195.36%	114.94%
1992	-59.23%	-56.74%	-35.49%	-57.82%	-67.48%	-56.31%	-38.87%	-55.18%
1993	-45.74%	-38.52%	-59.26%	-29.81%	-65.90%	-20.05%	-48.70%	-41.01%
1994	-43.91%	-29.04%	-63.99%	-13.89%	-4.42%	-0.59%	-36.63%	-21.20%
1995	-3.66%	27.90%	-19.75%	-45.83%	-38.41%	-16.77%	-12.18%	-20.33%
1996	-42.62%	40.99%	8.62%	11.06%	21.49%	21.59%	37.93%	18.01%
1997	-67.88%	-70.19%	-52.23%	-30.83%	-62.54%	-60.90%	-66.43%	-58.88%
Grand Total								

Analysis Report





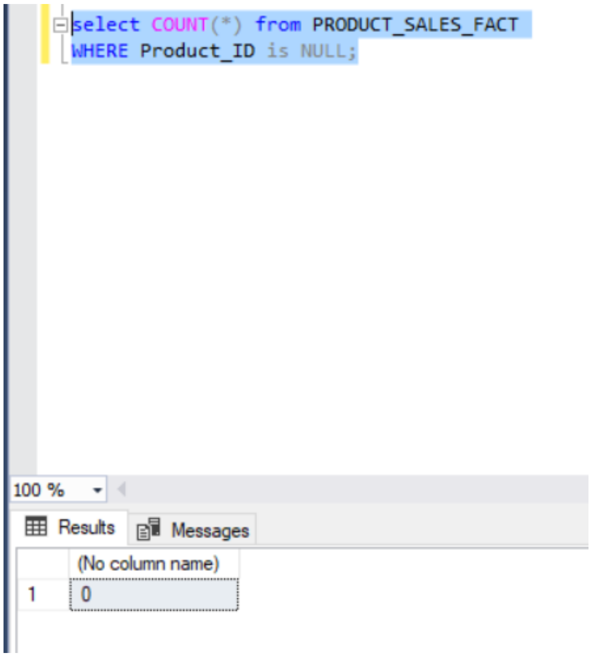
Conclusion

The graph above addresses the annual growth rate of cigarette margins in low tier stores well. All the listed numbers, like 3, 7, 32, 55, are the store IDs of low tire stores. A clear trend is that from 1991 to 1992, the cigarette sale in all low tire stores decreased sharply. From year 1992 to 1996, the sale was basically increasing. Like store 7 and store 81, the sale of cigarette was always increasing in the five years. And in stores like 55, the sales decreased from 1994 to 1995, but it continued to increase from 1995 to 1996. Overall, the trend was increasing. However, from 1996 to 1997, all stores' cigarette sales were decreasing once again. That trend analysis enables DFD react quickly to the market and adjust the inventory to the market needs. The same analysis method applies to other sale trend discoveries, which facilitates DFD to make better business decisions.

7. Group Special Work on Data Quality, Error Files

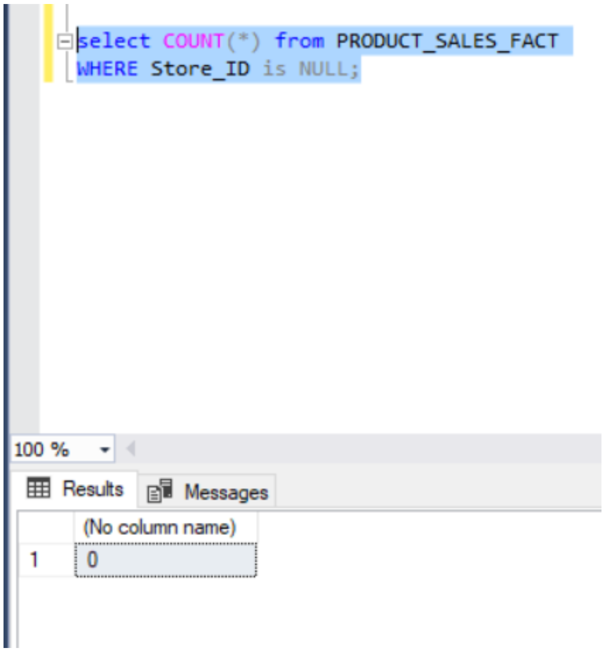
We have done the data quality checks at the data warehouse level for referential integrity, value and volume tests by using Industry standard ETL testing procedures. We have successfully tested the data for NULL value and Duplicate checks in Fact and Dimension tables thereby assuring highest data quality to facilitate single version of truth. We have re-directed the bad records to Error file which can be used to reconciliation and Audit purpose to periodically check the overall health of data warehouse.

7.1 NULL CHECK



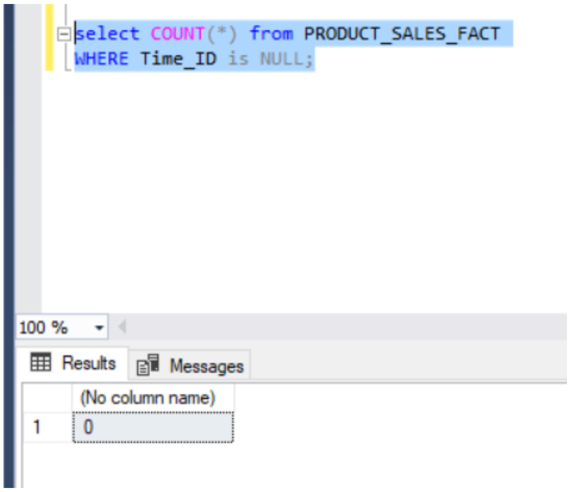
```
select COUNT(*) from PRODUCT_SALES_FACT
WHERE Product_ID is NULL;
```

```
   (No column name)
1  0
```



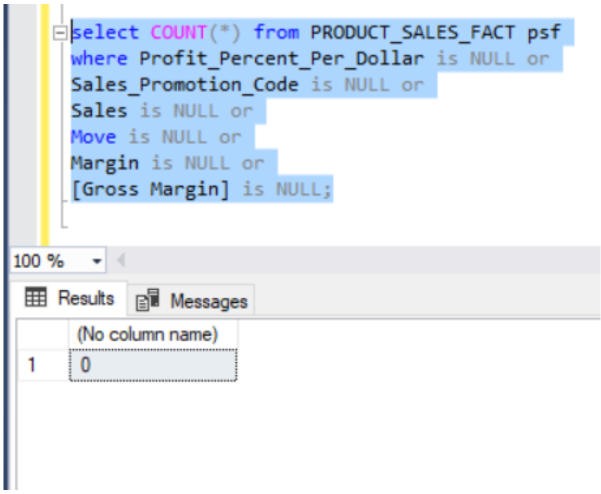
```
select COUNT(*) from PRODUCT_SALES_FACT
WHERE Store_ID is NULL;
```

```
   (No column name)
1  0
```



```
select COUNT(*) from PRODUCT_SALES_FACT
WHERE Time_ID is NULL;
```

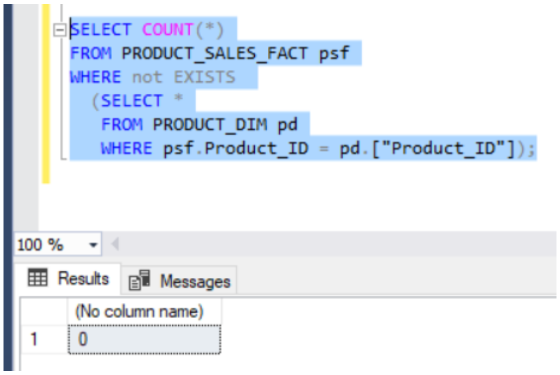
```
   (No column name)
1  0
```



```
select COUNT(*) from PRODUCT_SALES_FACT psf
where Profit_Percent_Per_Dollar is NULL or
Sales_Promotion_Code is NULL or
Sales is NULL or
Move is NULL or
Margin is NULL or
[Gross Margin] is NULL;
```

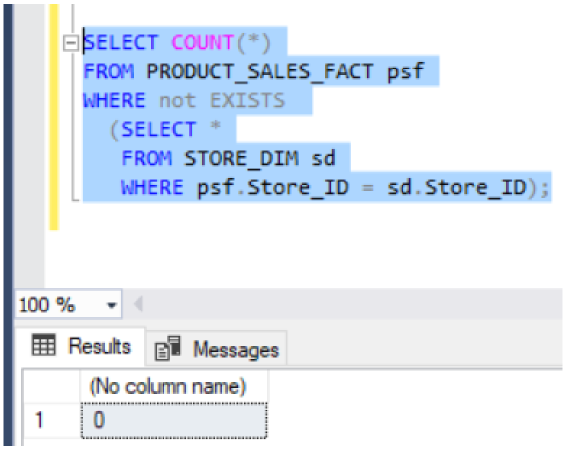
```
   (No column name)
1  0
```

7.2 REFERENTIAL INTEGRITY CHECK



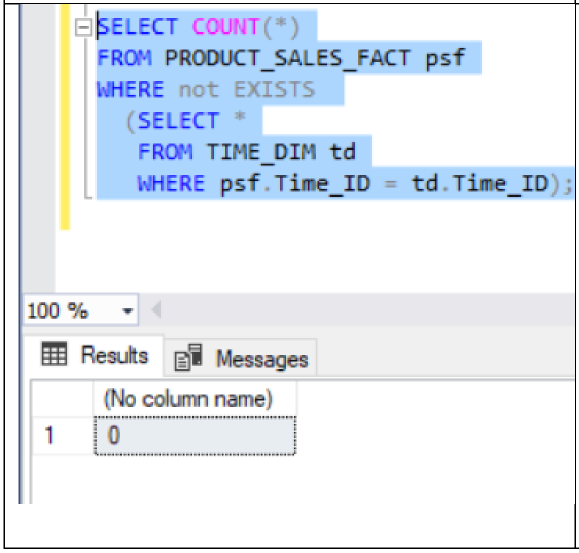
```
SELECT COUNT(*)
FROM PRODUCT_SALES_FACT psf
WHERE not EXISTS
  (SELECT *
   FROM PRODUCT_DIM pd
   WHERE psf.Product_ID = pd.["Product_ID"]);
```

```
   (No column name)
1  0
```



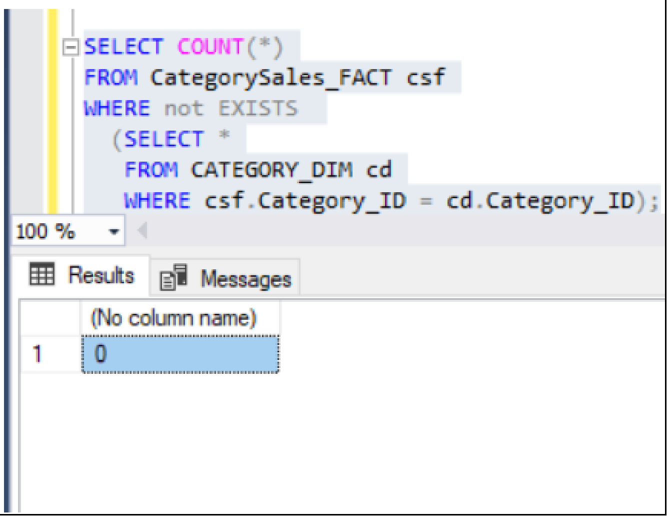
```
SELECT COUNT(*)
FROM PRODUCT_SALES_FACT psf
WHERE not EXISTS
  (SELECT *
   FROM STORE_DIM sd
   WHERE psf.Store_ID = sd.Store_ID);
```

```
   (No column name)
1  0
```



```
SELECT COUNT(*)
FROM PRODUCT_SALES_FACT psf
WHERE not EXISTS
  (SELECT *
   FROM TIME_DIM td
   WHERE psf.Time_ID = td.Time_ID);
```

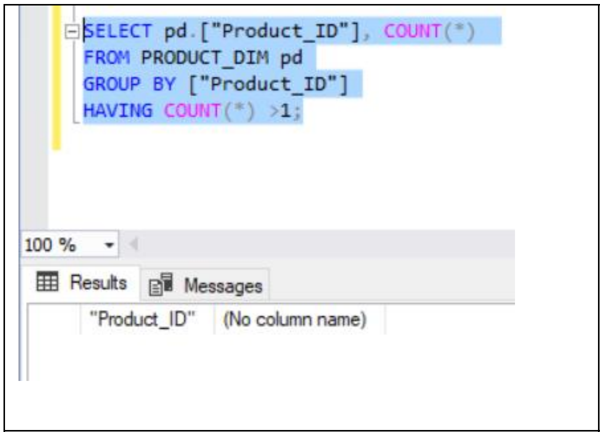
```
   (No column name)
1  0
```



```
SELECT COUNT(*)
FROM CategorySales_FACT csf
WHERE not EXISTS
  (SELECT *
   FROM CATEGORY_DIM cd
   WHERE csf.Category_ID = cd.Category_ID);
```

```
   (No column name)
1  0
```

7.3 DUPLICATES CHECK

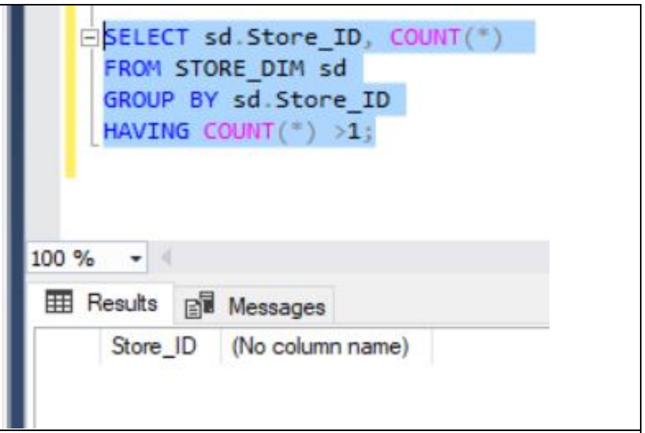


```
SELECT pd.["Product_ID"], COUNT(*)
FROM PRODUCT_DIM pd
GROUP BY ["Product_ID"]
HAVING COUNT(*) >1;
```

100 %

Results | Messages

"Product_ID"	(No column name)

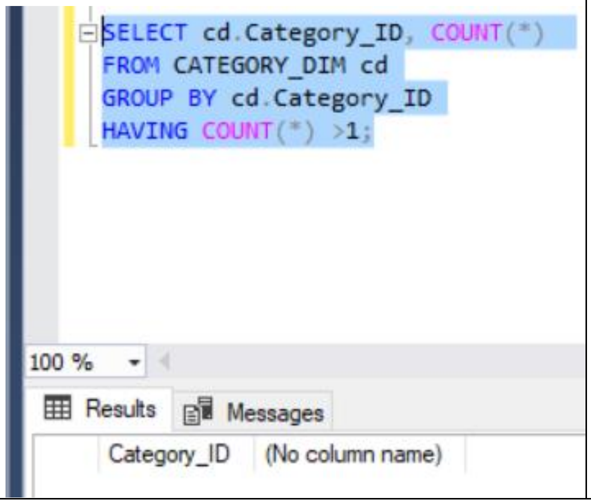


```
SELECT sd.Store_ID, COUNT(*)
FROM STORE_DIM sd
GROUP BY sd.Store_ID
HAVING COUNT(*) >1;
```

100 %

Results | Messages

Store_ID	(No column name)

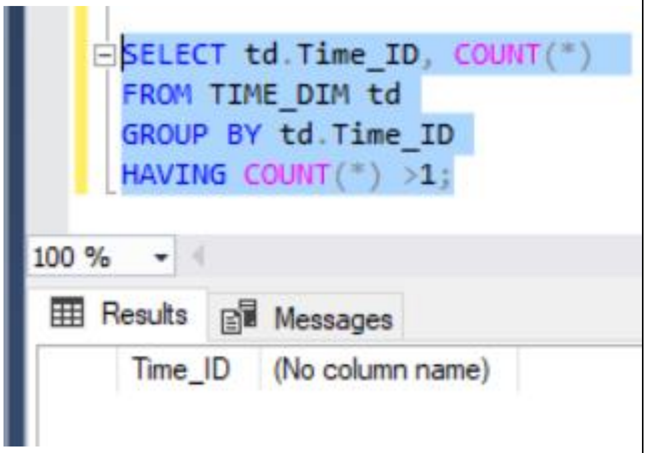


```
SELECT cd.Category_ID, COUNT(*)
FROM CATEGORY_DIM cd
GROUP BY cd.Category_ID
HAVING COUNT(*) >1;
```

100 %

Results | Messages

Category_ID	(No column name)



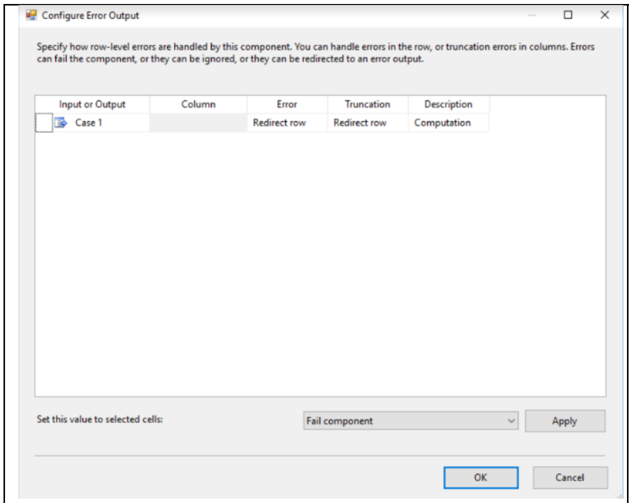
```
SELECT td.Time_ID, COUNT(*)
FROM TIME_DIM td
GROUP BY td.Time_ID
HAVING COUNT(*) >1;
```

100 %

Results | Messages

Time_ID	(No column name)

7.4 Error File Creation



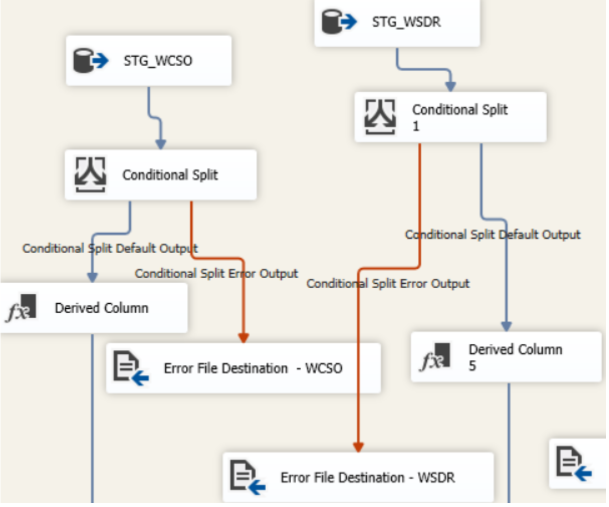
Configure Error Output

Specify how row-level errors are handled by this component. You can handle errors in the row, or truncation errors in columns. Errors can fail the component, or they can be ignored, or they can be redirected to an error output.

Input or Output	Column	Error	Truncation	Description
Case 1		Redirect row	Redirect row	Computation

Set this value to selected cells: Fail component | Apply

OK | Cancel


STG_WSDR
STG_WCSO
Conditional Split 1
Conditional Split
Conditional Split Default Output
Conditional Split Error Output
Conditional Split Default Output
Conditional Split Error Output
Derived Column
Error File Destination - WCSO
Derived Column 5
Error File Destination - WSDR

Appendix A

Figure 1 ERD for Dominick's Data Warehouse 6
Figure 2 Data for BQ-1 7
Figure 3 Pivot Table for BQ-1 8
Figure 4 Data for BQ-2 9
Figure 5 Pivot Table for BQ-2 9
Figure 6 Data for BQ-3 10
Figure 7 Pivot Table for BQ-3 10
Figure 8 Data for BQ-4 11
Figure 9 Pivot Table for BQ-4 12
Figure 10 Data for BQ-5 13
Figure 11 Pivot Table for BQ-5 13
Figure 12 Product Sales Information Package 14
Figure 13 Category Sales Information Package 14
Figure 16 - Table Structure for Product Sales Data Mart 51
Figure 17 - Table Structure for CategorySales Data Mart 52
Figure 18 - Snapshot for Data Warehouse Area 53