DOMINICK'S FINER FOODS

Consulting Report 4 - BI Report Design and Implementation for DFF

ISTM 637 – 601
Group 7
Udhav Chandel
Sandheep Sridar
Aishwarya Muralidhar
Email ID: aishmurali24@tamu.edu

Credentials for accessing the data warehouse, reporting and analysis services

SQL Server Authentication

Server name: infodata16.mbs.tamu.edu

Username: sr7205

Password: Mays7205

The above credentials can be used to access:

1. Database Engine in the Microsoft SQL Server Management Studio

   Staging area: group7-staging-area
   Datawarehouse: group07_DataWarehouse

2. Analysis Services of the SQL Server

   Cube deployed for SSAS: group07_qsn2

3. http://infodata.tamu.edu/ReportServer

   Project Folders:

   Question 1: Group_07_SSRS_SSAS

   Question 2: group07-FEC

   Question 3: group07-cheese

Contents

1. Introduction

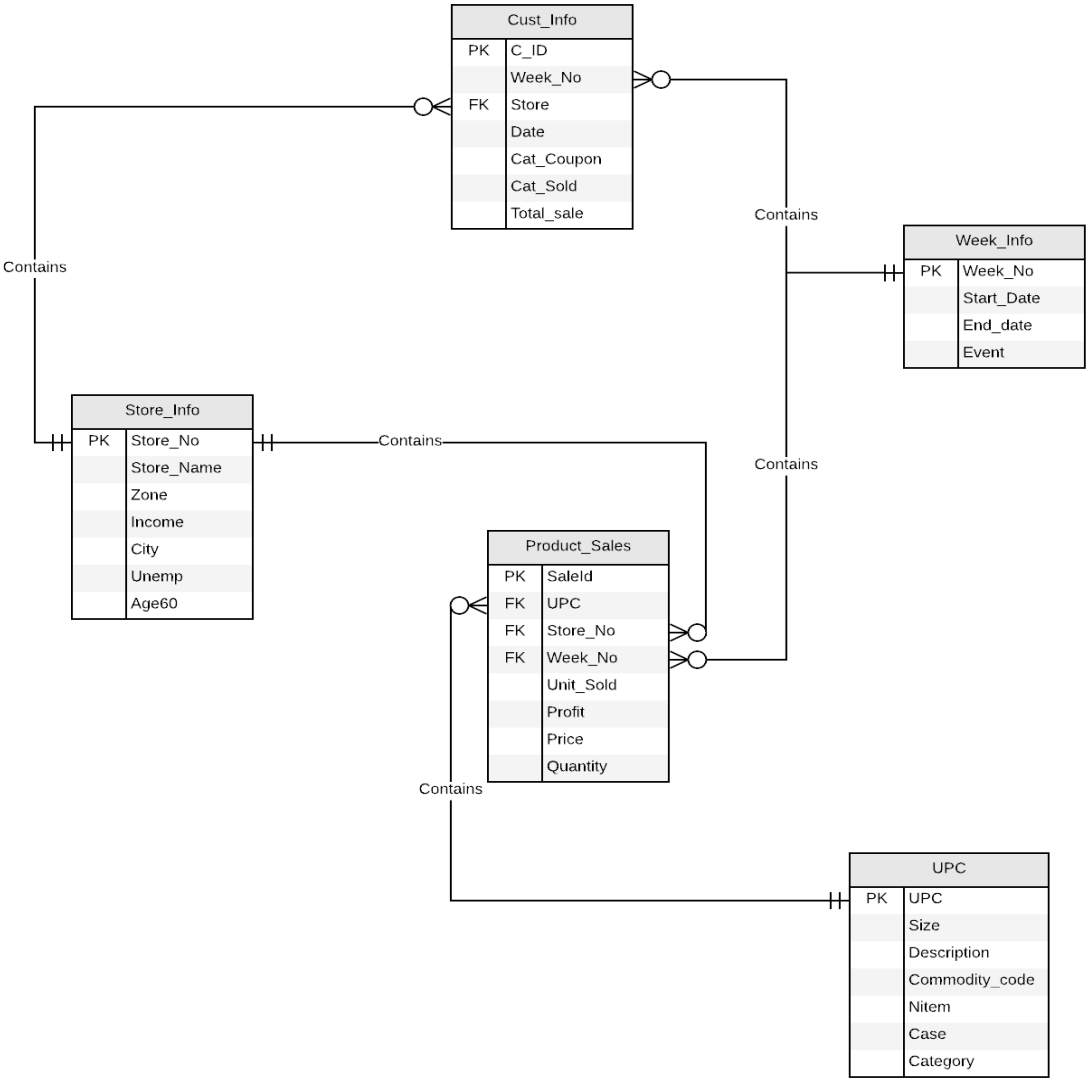
Dominick's Finer Food is a retail store chain with over 100 stores in various parts of Chicago. Founded in 1918 by Dominick Di Matteo, Dominick's Finer Foods, a family business, was a subsidiary of Safeway Inc and was headquartered at Oakbrook, Illinois. In 1950, DFF opened up their first supermarket store, the first of its kind. DFF carried products like bakery, dairy, deli, frozen foods, produce and liquor, among others. In 1986, DFF started a discount grocery store concept by the name of Jerry's Deep Discount Centers but this venture was discontinued soon after. DFF employed unique in-store communication methods like Redwood digital telephone systems, walkie talkie radios and music/paging. DFF began the Omni Superstore having a low-cost structure in 1987 which carried non-food items. Owing to subpar performance and low sales, DFF did not thrive.

Our biggest problem till date has been making sense of the several huge dirty datasets and trying to locate the desired values in the many tables in order to perform our analysis. Leaning on the metadata available in the manual simplified the process but not quite. The awfully slow network connection, lag in system speed and continuous Excel crashes resulted in loss of time and reduced efficiency. Design of charts using Excel and JMP was a tricky task as we were not adept at handling complex and dirty datasets which required cleaning prior to analysis. Investment of time and resources on understanding the data and the domain (retail, in our case) helped to make informed decisions and reach sensible conclusions.

I would like to acknowledge the contribution of the James M. Kilts Center, University of Chicago Booth School of Business in providing this data to us. It has helped numerous students like us, understand the concepts of data warehousing. Dominick's Finer Foods and Chicago Booth partnered for store-level research. They accumulated data spanning over a period of 1989 to 1994. It is historic data. The data size is about 4.76 Gigabytes. The data contains information covering products sold, sales, stores, demographics and movement. It is multifaceted, due to the sheer size of data that has been obtained. A large portion of the data is dirty and needs to be cleaned before analysis can take place. Domain knowledge is important in understanding the relationship between different entities.

2. Data Description

2.1. Entity-Relationship Diagram



2.2. Metadata

Information pertaining to Dominick's files are as follows:

- **Customer Count File – Ccount**
  This file contains information of customers purchasing products at their store on a daily basis. It makes available data such as store traffic related information, coupons which have been redeemed and total dollar sales.

- **Store-Specific Demographics**
  Store specific demographic information helps understand financial, age, ethnic related information of customers visiting their store. It provides information such as percent of college graduates, household size and percentage of telephone users. This data has been collected during census by the US government.

- **UPC files**
  UPC is an acronym for Universal Product Code. Every record which is stored per UPC is stored within this file. Details such as the name, size are found. The file names are named in an upcxxx format, where xxx is the three-letter acronym for the category.

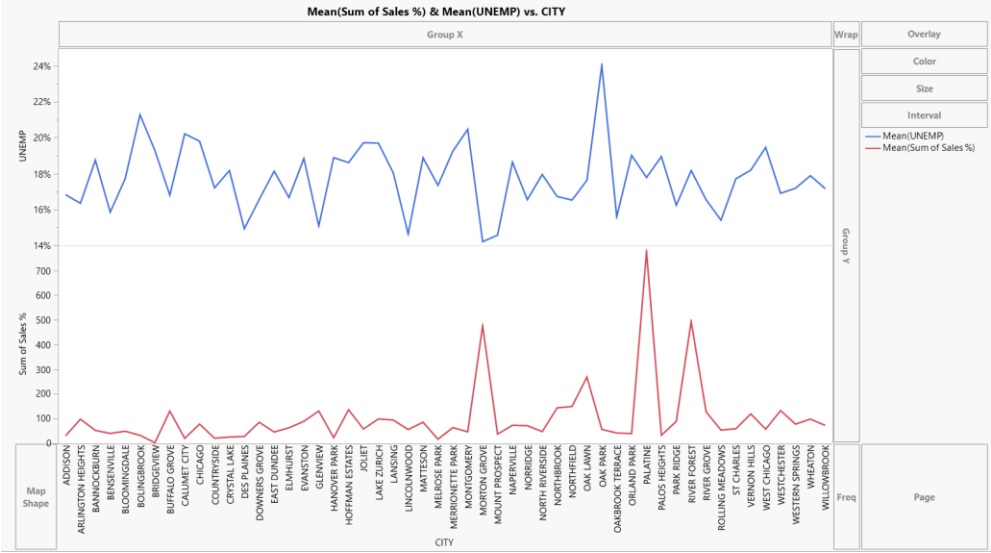
- **Movement files**
  Sales data pertinent to a store for every UPC is stored within the movement files. It contains weekly data with information such as the Number of units sold, retail price, sale code and gross margin

3. Business Questions

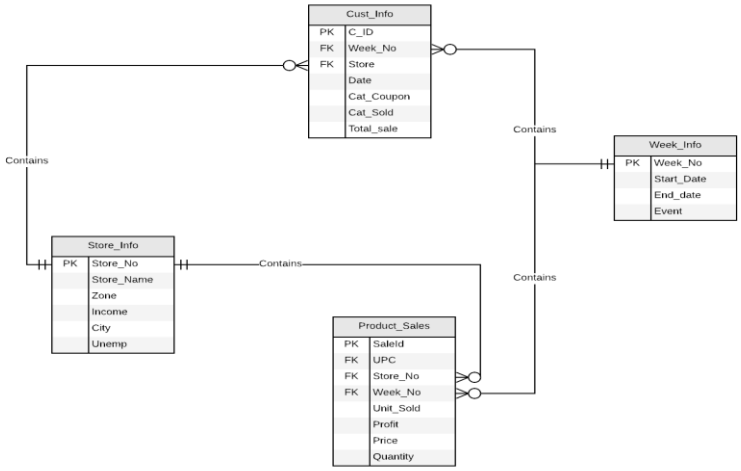
What is the effect of unemployment on sales in a particular city over time?

Rationalization

The employment scenario in a city impacts the business of firms operating in the vicinity. Unemployment inevitably results in low income margins and from a corporate strategy standpoint, the information pertaining to the employment rate in a particular city will help DFF expand into cities with higher income markets.



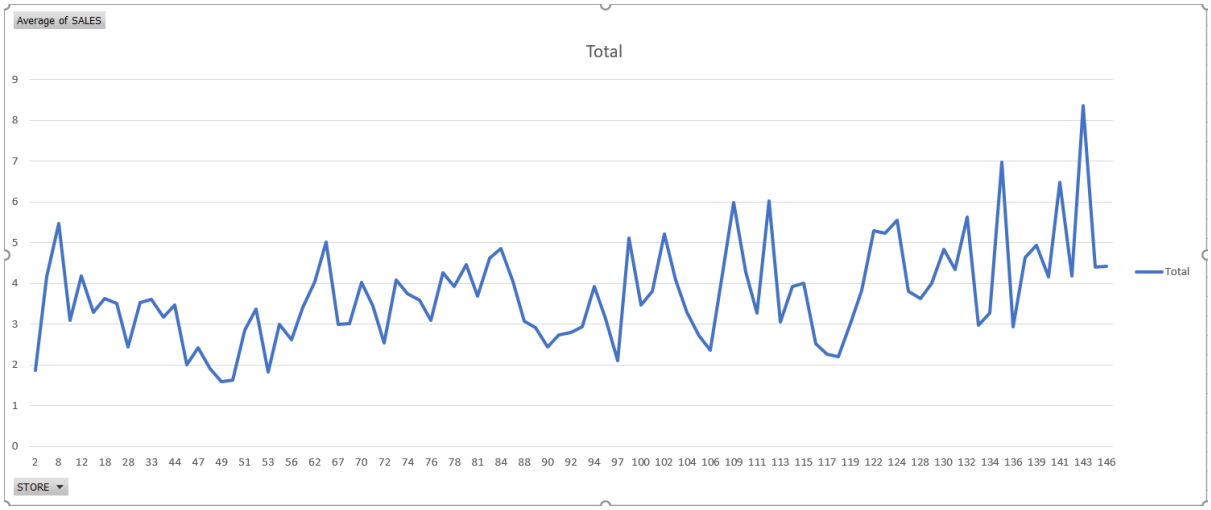
Entity Relationship Diagram



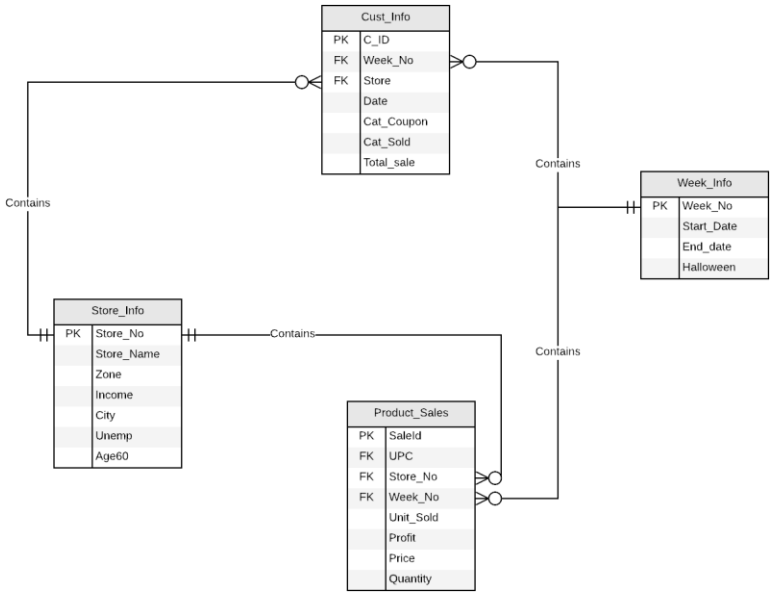
What is the trend of front-end candy sales during Halloween?

Rationalization

Trend of front-end candy sales is confirmed to increase during Halloween as people go trick-or-treating. This information can help DFF prepare to meet the increased demand by increasing capacity and production. Also, they can charge a premium price in order to gain a considerable profit margin



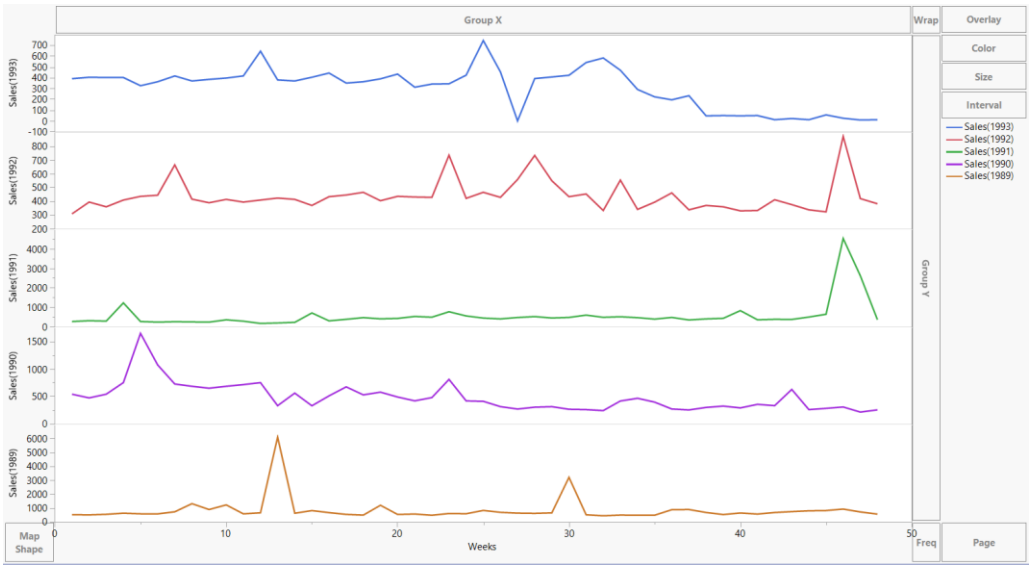
Entity Relationship Diagram



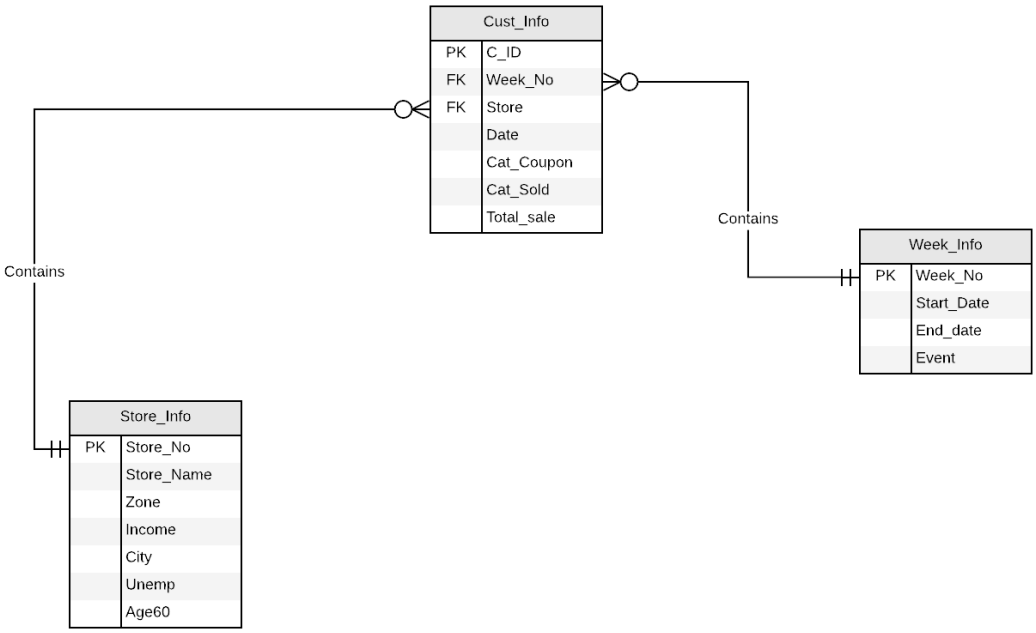
What is the trend of cheese sales from the year 1989 to 1993?

Rationalization

This data depicts cheese sales in a specific time period, yearly, in our case. This information can be used by DFF to decide whether to continue carrying cheese in their stores or remove it. Further, DFF can determine the improvements required in the product to boost sales.



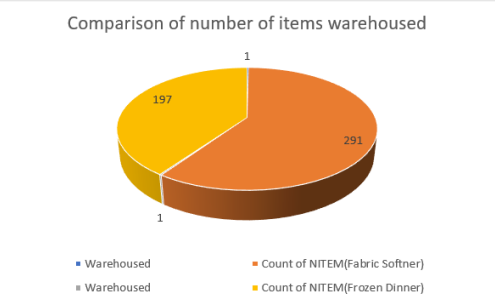
Entity Relationship Diagram

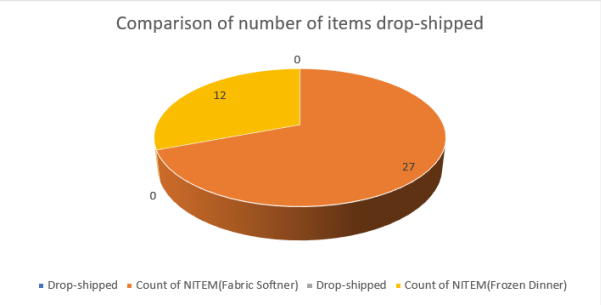


How many Fabric softeners and frozen dinners were drop-shipped vs warehoused? (0 or 1 at the last digit of the item code) according to each individual item? (Priority – 9)

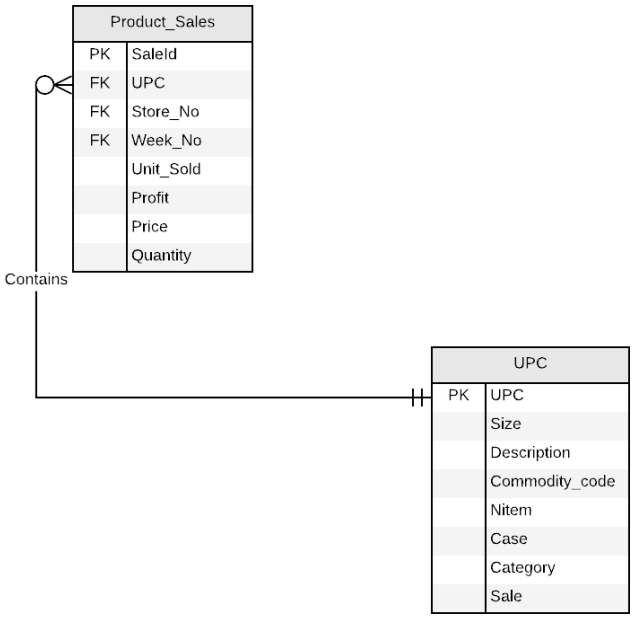
Rationalization

Since frozen dinners are perishable while fabric softeners are non-perishable, drop-shipping the former is the viable option as they tend to expire early resulting in losses to DFF whereas the latter can be warehoused. Thus, this data helps DFF in inventory management.





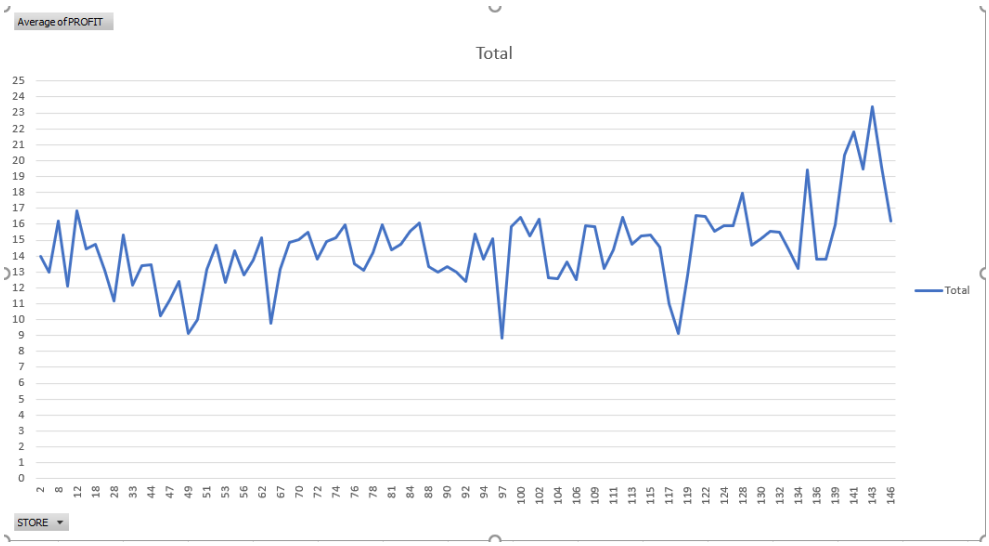
Entity Relationship Diagram



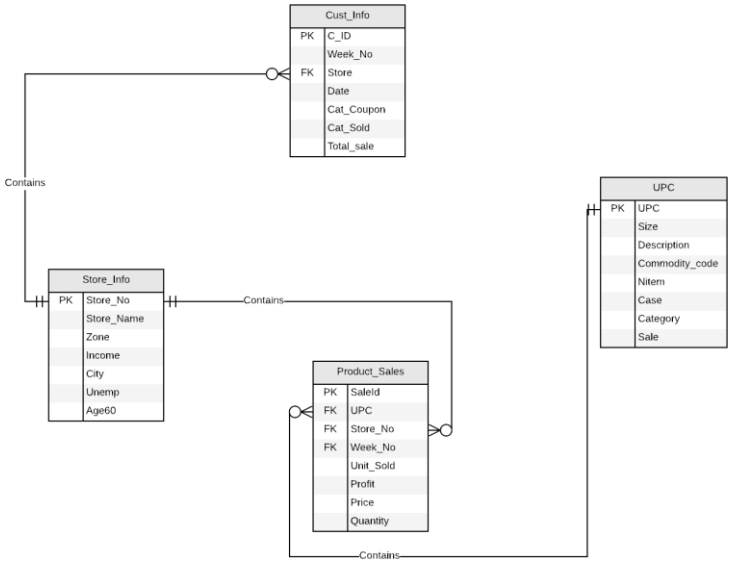
Find the average gross margin for oatmeal across all stores and determine what stores are performing below average

Rationalization

On knowing which stores sell maximum quantities of oatmeal and the revenue generated from it, DFF can invest on R&D and consider venturing into other organic foods such as oatmeal cookies, breads etc. in the best-selling store locations



Entity Relationship Diagram



4. Independent Data Marts using Kimball's approach

By using the STAR schema approach for designing a data warehouse, we will create dimension models and fact tables using Kimball's approach. Questions to be analyzed for Dominic's finer foods will be represented in the form of data marts.

Our analysis involves a total of three DMs and three fact tables:

4.1 Dimensional Modeling

The three Dimension Tables which fit our business requirements are as follows:

- ***DimProduct***

  The product dimension table focuses on individual products which exist in the Dominic's finer foods store.

Attributes of the *DimProduct* table has been described as follows:

Product_Key - It is a unique identifier which identifies our products individuality. It's generated in the form of a surrogate key in the schema model
Product_Name – Provides a description of the product and helps identify the product being analyzed, as contained in the DFF store
UPC_Number – The number essentially helps in identifying the manufacturer and product. The numbers first five digits identify the product and remaining digits identify the manufacturer
Item_Code – The last digit of the Item_Code provides information of whether the product was drop-shipped or warehoused

Category – Acts as an indicator in generic dimension models and provides information pertaining to the specific category a product belongs to

- ***DimStore***

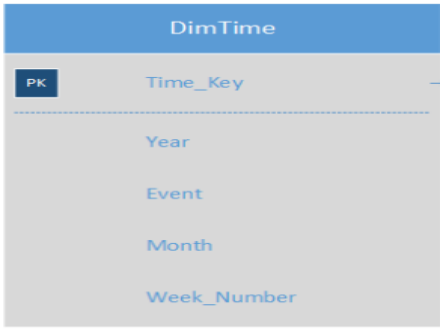
The dimension table for store identifies several stores which exist as part of the DFF store chain. This table will help in analysis of store-wise data which would need to be presented as part of our analysis.

Attributes of *DimStore* are described as follows:

Store_Key – Uniquely identifies each store in DFF store chain. It is generated as a surrogate key in the schema
Store_Name – Describes the location of each store which is located
Store_City – Identifies the city within which the store is located. This attribute allows analysis to be carried out among cities
Store_Zone – Each store falls into a particular zone

- ***DimTime***

Time is the medium of comparison. The Time dimension model provides information with respect to the time aspect of events, sales of products



Following attributes describe the *DimTime* dimension table:

Time_Key – A surrogate key which will uniquely identify the occurrence of an event
Year – The year during which the even has taken place in. Essential for analysis across several years
Month – Indicates the month during which sales have taken place
Week_Number – Extremely essential in the warehouse as it exists across several tables. Indicates the week of occurrence of the sale
Event – Occurrence of events such as Halloween, Thanksgiving during a year are indicated by this attribute

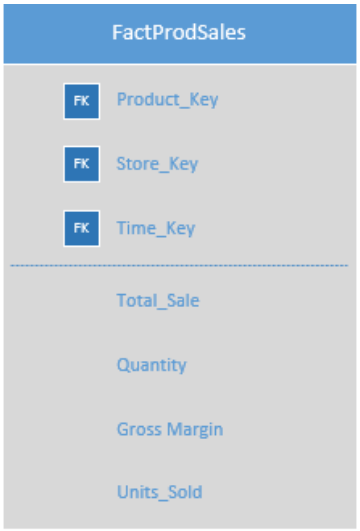
4.2 Fact Tables

Based on our business questions we've identified three fact tables which would help us provide an effective analysis of our data. These fact tables refer the surrogate keys and other metrics from DM's and provide insight to the data which is being analyzed.

Following are the fact tables which have been used by us:

- **FactProdSales**

  The fact table FactProdSales has the metrics of product sales. It references the product, store and time dimension tables



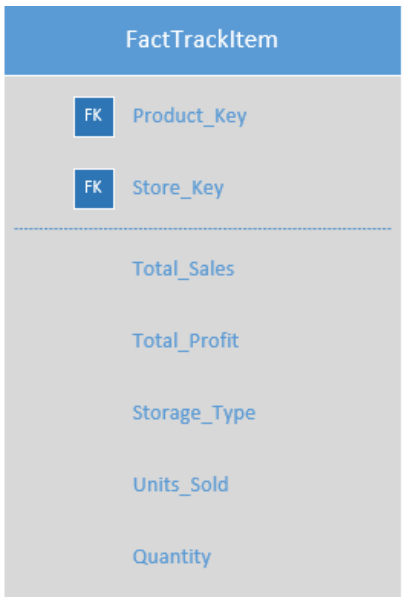
Following attributes describe the *FactProdSales* fact table:

Product_Key - It is a unique identifier which identifies our products individuality.

Store_Key – Uniquely identifies each store in DFF store chain. It is generated as a surrogate key in the schema.
Time_Key – A surrogate key which will uniquely identify the occurrence of an event
Total_Sale – A metric of the fact tables which is an additive measure. It will contain the total sales of the product
Units_Sold – A metric of the fact tables which indicates the number of products which have been sold by the store
Quantity – It is a metric of the fact table which holds the number of products which exist
GrossMargin – It is a quantifiable metric which represents the amount of revenue the product made

- **FactTrackItem**

The fact table FactTrackItem has the product tracking metrics. This table is useful for monitoring drop shipped and warehoused product.



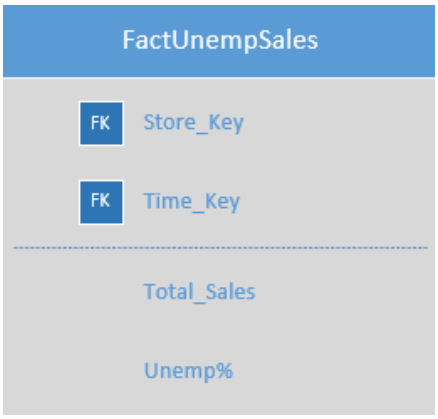
Following attributes describe the *FactTrackItem* fact table:

Product_Key - It is a unique identifier which identifies our products individuality.
Store_Key – Uniquely identifies each store in DFF store chain. It is generated as a surrogate key in the schema.
Total_Sale – A metric of the fact tables which is an additive measure. It will contain the total sales of the product

Units_Sold – A metric of the fact tables which indicates the number of products which have been sold by the store
Total_Profit – A metric of the fact tables which is an additive measure. It will contain the total profit made by the product
Storage_Type – It is a non-additive measure in the fact table which identifies the product tracking
Quantity - It is a metric of the fact table which holds the number of products which exist

- **FactUnempSales**

The fact table FactTrackItem has the unemployment metrics. This table is useful for monitoring sales in low employment regions over a period of time.



Following attributes describe the *FactUnempSales* fact table:

Store_Key – Uniquely identifies each store in DFF store chain. It is generated as a surrogate key in the schema.
Time_Key – A surrogate key which will uniquely identify the occurrence of an event
Total_Sales – A metric of the fact tables which is an additive measure. It will contain the total sales of the product
Unemp% - The purpose of this attribute is to store the unemployment metric. This value is calculated by converting UNEMP column in DEMO file into a more representable format.

4.3 Dimension Matrix

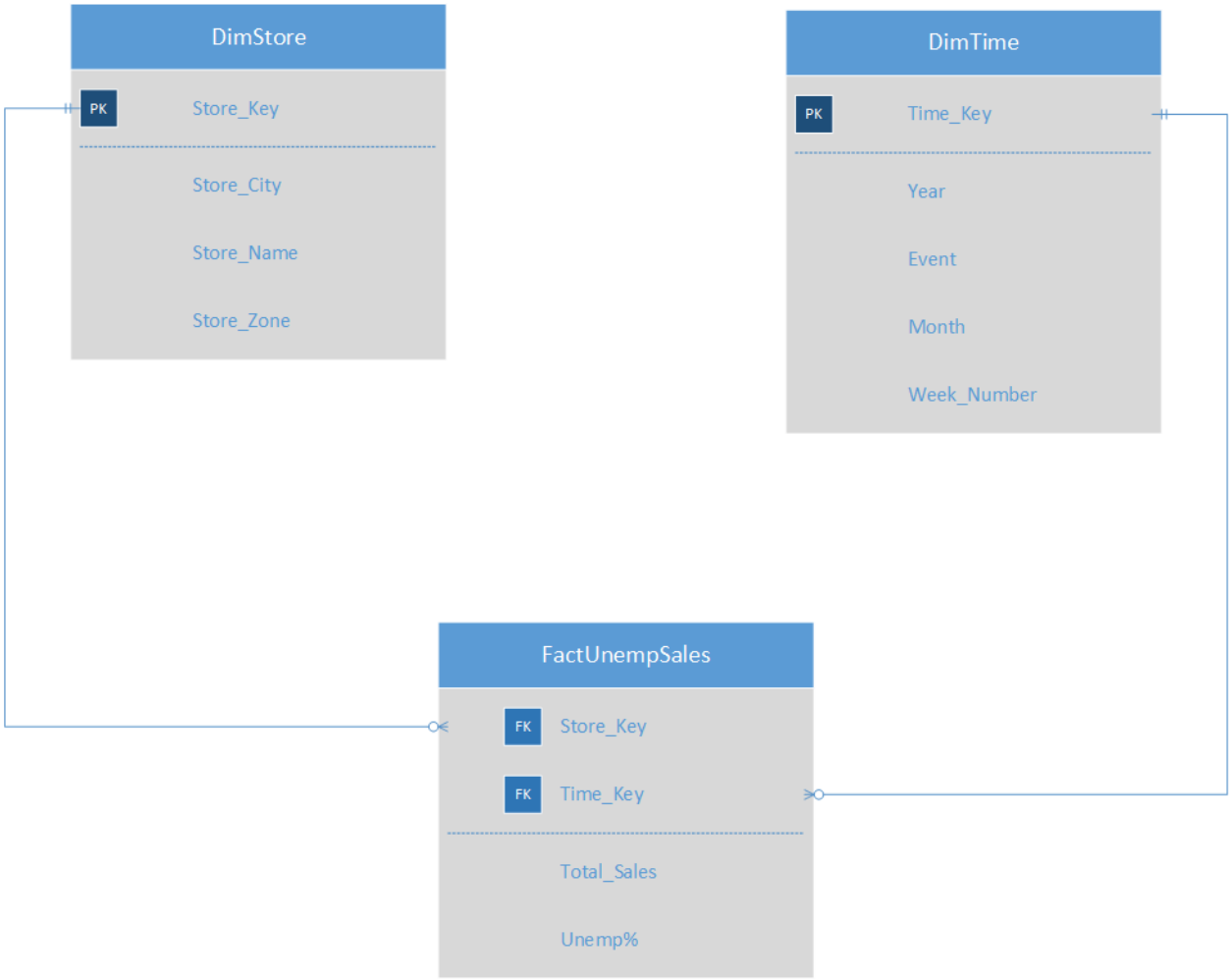
Dimension Tables/ Fact Tables	DimProduct	DimStore	DimTime
FactProdSales	✓	✓	✓
FactUnempSales		✓	✓
FactTrackItem	✓	✓	

4.4 Star Schemas

4.4.1 Star Schema for effect on unemployment on sales

This data mart answers the following question

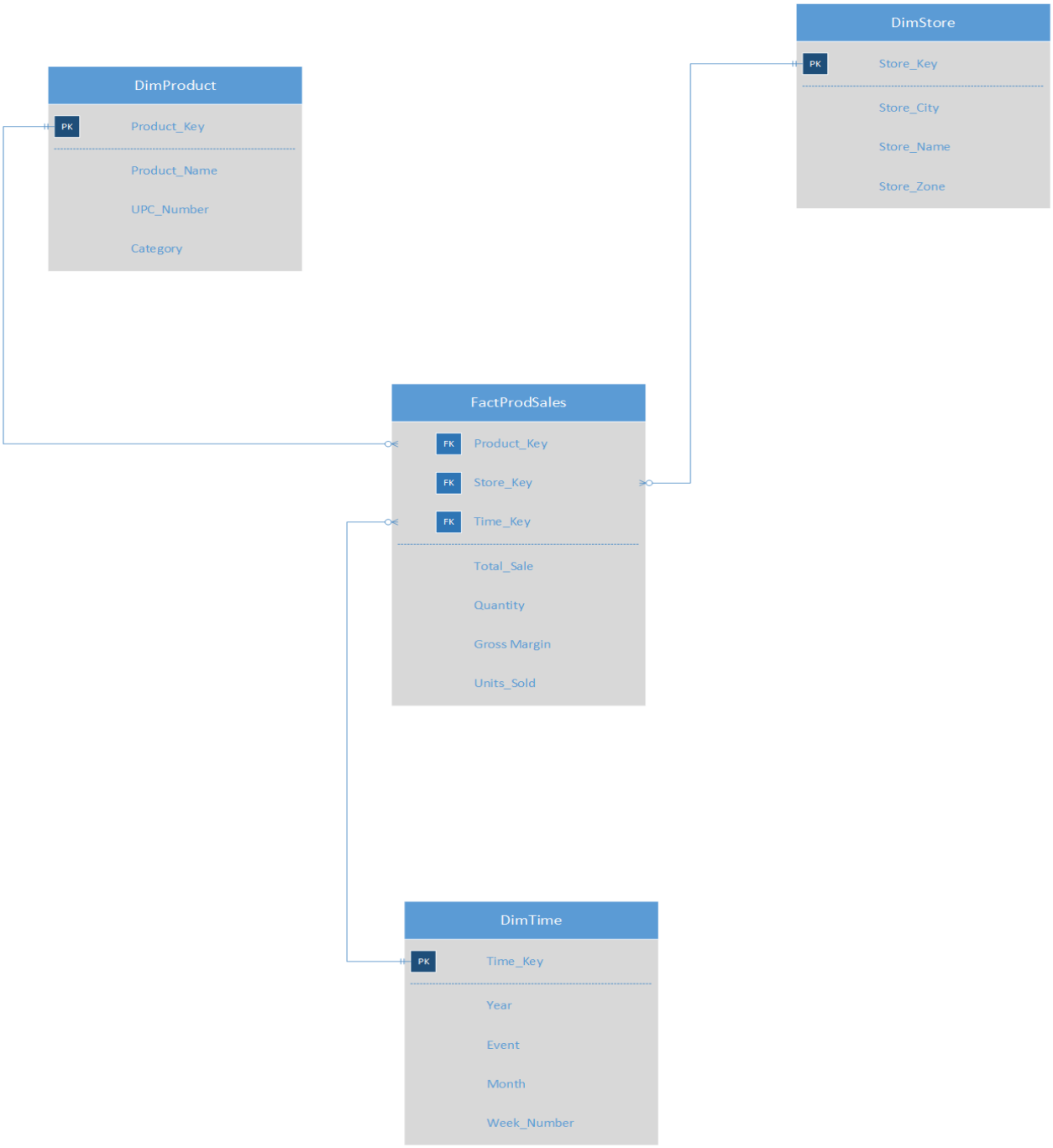
- What is the effect of unemployment on sales in a particular city over time?



4.4.2 Star Schema for Product sales

The Data mart answers the following questions:

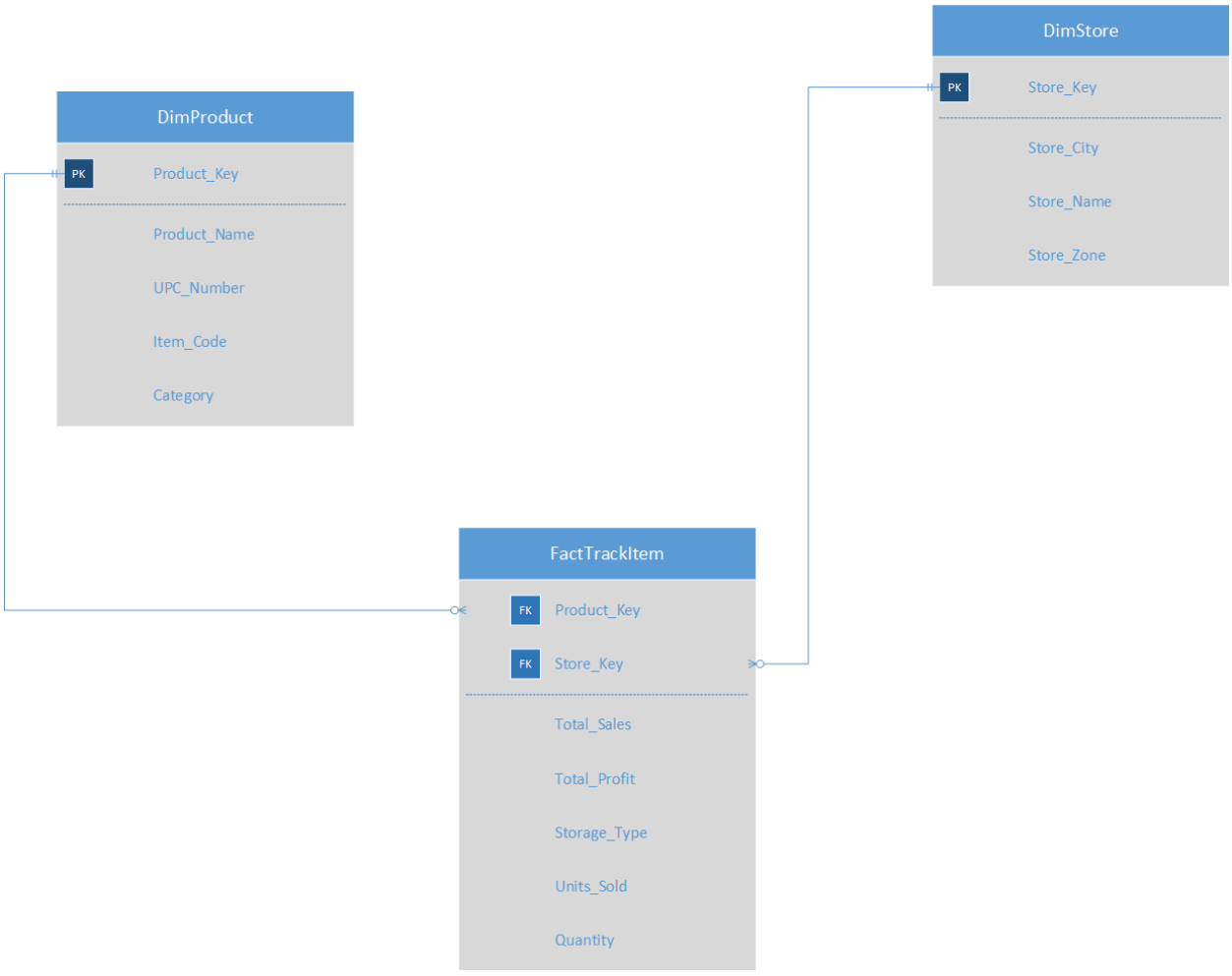
- What is the trend of front-end candy sales during Halloween?
- What is the trend of cheese sales from the year 1989 to 1993?
- Find the average gross margin for oatmeal across all stores and determine what stores are performing below average



4.4.3 Star Schema for Tracking Products

The Data Mart answers the following question:

- How many Fabric softeners and Frozen dinners were drop-shipped vs warehoused? (0 or 1 at the last digit of the item code) according to each individual item?



4.5 Mapping Table

4.5.1 Mapping table for Product Sales Data Mart

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Product	UPC Table			Product_Key(Row Number)
		Description		Product_Name
		UPC		UPC_Number
	OLTP Source			Category

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Store	Dominicks Store and Store Specific Demographics	Store		Store_Key(Row Number)
		City		Store_City
		Name		Store_Name
		Zone		Store_Zone

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Time	Week Decode Table			Time_Key(Row Number)
		Start End	By using Start and End from Week Decode Table, compute Year	Year
		UPC	By using Start and End from Week Decode Table, compute month. Divide value by 12	Month
		Week #		Week_Number
		Special Events		Event

DW Fact Table	Source Table	Source Table Attribute	Mapping Functions	DW Fact table Attributes
FactProdSales	Movement table		Primary key of product dimension	Product_Key
			Primary key of store dimension	Store_Key
			Primary key of time dimension	Time_Key
		sale	Formula: Sales = (price*move)/quantity	Total_Sales
		qty		Quantity
		profit		GrossMargin
		move		Units_Sold

4.5.2 Mapping table for Unemployment Sales Data Mart

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Store	Dominicks Store and Store Specific Demographics	Store		Store_Key(Row Number)
		City		Store_City
		Name		Store_Name
		Zone		Store_Zone

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Time	Week Decode Table			Time_Key(Row Number)
		Start End	By using Start and End from Week Decode Table, compute Year	Year
		UPC	By using Start and End from Week Decode Table, compute month. Divide value by 12	Month
		Week #		Week_Number
		Special Events		Event

DW Fact Table	Source Table	Source Table Attribute	Mapping Functions	DW Fact table Attributes
FactUnempSales	Movement table		Primary key of store dimension	Store_Key
			Primary key of time dimension	Time_Key
		sale		Total_Sales
		UNEMP	UNEMP * 100	Unemp%

4.5.3 Mapping table for Tracking Items

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Product	UPC Table			Product_Key(Row Number)
		Description		Product_Name
		UPC		UPC_Number
		NITEM		Item_Code
	OLTP Source			Category

DW Dimension Table	Source Table	Source Table Attribute	Mapping Functions	DW Dimension Attribute
Store	Dominicks Store and Store Specific Demographics	Store		Store_Key(Row Number)
		City		Store_City
		Name		Store_Name
		Zone		Store_Zone

DW Fact Table	Source Table	Source Table Attribute	Mapping Functions	DW Fact table Attributes
FactTrackItem	Movement table		Primary key of product dimension	Product_Key
			Primary key of store dimension	Store_Key
		sale	Formula: Sales = (price*move)/quantity	Total_Sales
		profit		Total_Profit
		Itemcode		Storage_Type
		move		Units_Sold
		qty		Quantity

4.6 Justification of business questions corresponding to data marts

4.6.1 What is the effect of unemployment on sales in a particular city over time?

This business question is answered by the UnemploymentSales data mart. The dimension table for store data consists of attributes Store_Key, Store_Name and Store_Zone. The Store_City along with Store_Zone will help us pin point to a specific geographic site where unemployment is prevalent. The DimTime dimension table highlights the Time_Key, Week_Number, the Month and Year to which the said week pertains and the Event (s) that occurred during that particular week and these attributes are used to divide data over time. The fact table FactUnempSales has Time_Key, Store_Key, Total_Sales and the percentage of unemployment in a specific city. Unemp% and Total_Sales help in understanding the store-wise impact of unemployment on sales during a time span. We have a star schema with these attributes as they are pivotal in determining the potential income markets for DFF.

4.6.2 What is the trend of front-end candy sales during Halloween?

This business question is answered by the ProductSales data mart. The DimTime dimension table consists of Time_Key, Week_Number, Month, Year and Event, DimStore dimension table consists of attributes Store_Key, Store_Name, Store_City and Store_Zone and DimProduct dimension table

consists of attributes Product_Key, Product_Name, UPC_Number and Item_Code. The Week_Number and Event attributes are used to identify the time periods pertaining to Halloween and the measure Total_Sales from the FactProdSales fact table is used to find the trend of front-end candy sales across the years which is a useful piece of information to DFF as it helps them meet the increasing demand for candies during the holiday season, by increasing capacity, production and maintaining sufficient inventory.

4.6.3 What is the trend of cheese sales from the year 1989-1993?

This business question is answered by the TrackingItem data mart. The DimTime dimension table consists of Time_Key, Week_Number, Month, Year and Event, DimStore dimension table consists of attributes Store_Key, Store_Name, Store_City and Store_Zone and DimProduct dimension table consists of attributes Product_Key, Product_Name, UPC_Number and Item_Code. The Year attribute is used to divide the data by year (in our case, 1989-1993). Then the measure Total_Sales from the FactProdSales fact table is used to find the trend of cheese sales across the above mentioned years thus enabling DFF to decide on whether to continue carrying cheese or remove it from their stores. Further, this data can help DFF predict the improvements required in the cheese variant they sell in order to boost sales.

4.6.4 How many Fabric softeners and Frozen dinners were drop-shipped vs warehoused? (0 or 1 at the last digit of the item code) according to each individual item?

This business question is answered by the data mart. The last digit of the Item_Code attribute of the DimProduct dimension table decides whether the product is drop-shipped (0) or warehoused (1). The Storage_Type attribute of the FactTrackItem fact table is used to determine whether the fabric softener and frozen dinner should be drop-shipped or warehoused and the Quantiy_Sold attribute provides information pertaining to the number of fabric softeners or frozen dinners sold by DFF. Since fabric softeners are non-perishable and have a relatively longer shelf life, they could be warehoused whereas frozen dinners being perishable should be drop-shipped. Thus, this data helps DFF in better inventory management.

4.6.5 Find the average gross margin for oatmeal across all stores and determine what stores are performing below average

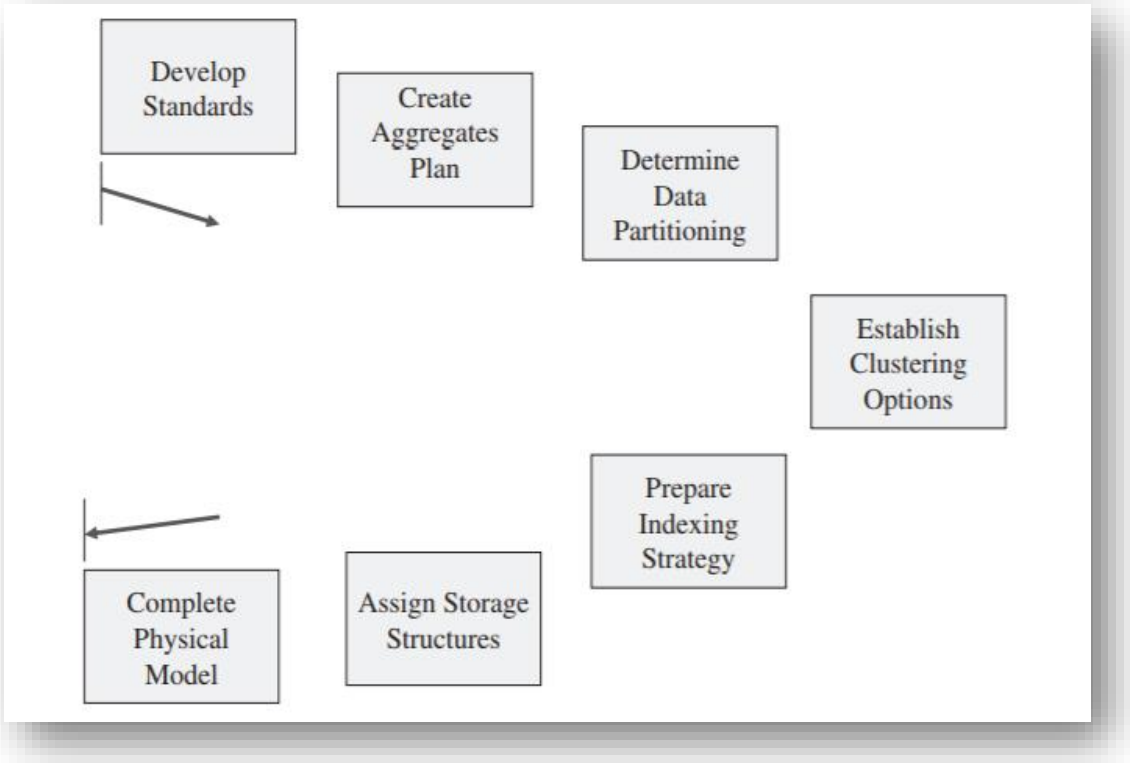
This business question is answered by the data mart. The DimTime dimension table consists of Time_Key, Week_Number, Month, Year and Event, DimStore dimension table consists of attributes Store_Key, Store_Name, Store_City and Store_Zone and DimProduct dimension table consists of attributes Product_Key, Product_Name, UPC_Number and

Item_Code. The Store_Key attribute is used from the DimStore dimension table to divide the data by stores. Then measure Total_Profit from the FactProdSales fact table is used to plot and compare sale of oatmeal across the stores. This data will provide DFF information pertaining to the quantity of oatmeal sold at the various store locations and the revenue generated from each.

4.7 Physical design plans

Physical models represent a data design in a database management system. They help in understanding the data requirements within a given context or environment and are used to add physical details such as indexing style, tables, columns and the underlying relationships between the various tables. Physical models are generally derived from logical models. It focusses on how a model is going to function and its major objectives are performance improvement, management of stored data, ensure scalability and flexibility, and provision of easy administration.

A typical physical design process follows the below trajectory of seven steps.



i. **Data standardization plan –** Standards are set to determine how a data warehouse is going to look, how the tables should be named, what information is to be elicited from the interviews with the end users, to name a few. Standards check consistency across various areas of a data warehouse and thus eliminate

ambiguity. Since both end users and the employees of an organization interact with the data warehouse, standards will simplify the lives of layman when they run their own queries.

Standards are employed in naming database objects, files and tables in the staging area and physical files. The key rule is to maintain the same names for both the logical and physical models. The names of database objects must be clear enough to convey its composition. Word separators such as underscore or dashes are used in the standardization process.

Staging area is where the action takes place and thus losing track of the files is a frequent mishap. Due to constant extraction, loading and transformation of data in the staging area, data is moved around here and thus should be named in accordance with the purpose it serves and the process it indicates.

Standards must cover data files, index files, physical files, application documents and source codes.

In case of DFF, we have named dimension tables starting with Dim, fact tables starting with Fact and primary keys ending with _Key in order to provide ease of identity to the parties involved. For example, the dimension tables are titled DimProduct, DimTime, DimStore. The fact tables are FactTrackItem, FactUnempSales, FactProdSales and the primary keys are Time_Key, Store_Key and Product_Key. Further, we have used camel case in our schema nomenclature.

ii. **Data aggregate plan –** This plan reviews the possibilities for constructing aggregate tables or summary tables. If data is stored at the lowest level of granularity, in the event of a query demanding summarized information, the query will have to traverse all the stored detailed records and then perform multiple calculations which is a cumbersome, time-consuming and inefficient process. Thus, data storage at high levels of granularity can help counter this issue but the concern is to determine the limit for the number of summary tables required. In order to design a comprehensive aggregate plan, consideration of the dimensional models and its hierarchies helps determine which levels require to be aggregated.

In case of DFF, storing data related to effect of unemployment on sales over time at a low level of granularity is permissible, as this information will not be queried in real-time. However, data related to the number of fabric softeners and frozen dinners that need to be drop-shipped or warehoused must be stored in an aggregated manner as this information is required by the management to perform inventory management. Rolling summary structures may be implemented wherein data can be stored at a low level of granularity and automatically rolled into the higher level of granularity, if need arises.

iii. **Indexing plan –** This is the most critical step in a physical design process as data warehouse revolves around queries. A solid indexing plan helps in enhancing performance and it should outline which columns in each table are to be indexed and the order in which the indexing must be performed.

Some of the issues in indexing are that the large number of indexes reduces speed of a data warehouse especially during the initial loading and this can be countered by recreating the indexes post loading or by splitting tables prior to index creation. Index provides a summary of the warehouse and thus adding columns in the index helps avoiding the traversal of redundant records. Examination of queries helps in understanding which ones are frequently used.

Commonly used indexing techniques are sequential indexing, bitmapped indexing and B-Tree indexing. B-tree indexing is the default method used and is superior owing to its data retrieval speed, simplicity and ease of maintenance. The lowest level index records refer to the rows in the table. If a column in a table contains many unique values, it is said to be highly selective. If a column is not highly selective, it can be combined with another column in order to increase its selectivity. Bitmapped indexing is used for data with low selectivity. They tend to occupy lesser space than B-tree indexes and are well-suited for data warehouse environments.

In case of DFF, we have decided to employ B-tree indexing technique as fact tables lack low-selectivity and thus will not work with Bitmapped indexing. Dimension tables will be indexed using B-tree indexing by determining the columns which are frequently used to limit queries and columns that are often accessed together will be combined to form multicolumn indexes in large dimension tables. For instance, Store_City aggregated with Store_Zone acts as the index in order to enhance selectivity and this information will be used in the query pertaining to effect of unemployment on sales in a city. Event can be the index in order to query information related to the trend of front-end candy sales during Halloween.

iv. **Data storage plan –** This plan details the physical files, their storage location, the tables to be assigned to each file and how to divide the file into data blocks for files related to front-end applications and the staging area.

The data structures used to store data are data warehouse tables (data base files and index files), OLAP system files and staging area files such as RDBMS files, RDBMS index files and data extract files.

In order to optimize storage, it is essential to set the appropriate block size and block usage parameters (block percent free or block percent used), manage data migration and block utilization efficiently.

Storage size can be estimated by determining the number of rows and length of each row in every table as well as the expected increase per month. The number of indexes and the space occupied per index is estimated in addition to the temporary work space required for sorting and merging data in the staging area.

RAID technology is useful in disk mirroring, disk duplexing, parity checking and disk striping. It has multiple levels .i.e. RAID 0, RAID 1, RAID2, RAID 3, RAID 4 and RAID 5.

In case of the large datasets of DFF, we plan on using Data Warehouse tables as the data structures to hold the data pertaining to DFF. We plan on estimating the size required to store the physical files beforehand by determining the contents of each table in every file and the desired location for each file. Further, we will employ RAID technology particularly RAID, RAID 3 and RAID 5 to ensure parity checking and data striping.

5. Data Cleaning and Integration – ETL Plan and Implementation

ETL stands for extract, transform, load which is a combined procedure used to copy data from source (s) to destination.

• **Extract** – Function used to read data from a database

• **Transform** – Function used to sanitize data and convert it into a usable form which is suitable for storage and analysis

• **Load** – Function used to insert data into an operational data store, data mart or data warehouse

Issues with the DFF data set

Parameter	Quality issue
Format	Formatting standards were inconsistent
Consistency	Presence of duplicates leading to misunderstanding of data
Validity	Presence of invalid values such as negative sales, special characters for date
Referential integrity	Owing to repetitive data, establishing relationships and enforcing referential integrity was a challenge
Precision	Blanks, dots in place of relevant data required intense data cleaning

The steps involved in the ETL plan for implementing a data warehouse are detailed below –

5.1 Determining target data needed in the data warehouse

Data from sources	Data in staging area	Data in data warehouse
Ccount	CCOUNT_FINAL	DimStore, FactProdSales
Demographics	DEMO_FINAL	DimStore, FactUnempSales
Product data	UPC<productacronym>	DimProduct
Time data	Week_Decode	DimTime
Movement data	W<productacronym>	DimProduct, FactTrackItem

5.2 Determining data sources

Target data needed	Data source
Ccount	CCOUNT
Demographics	DEMO
Product data	UPC<productacronym>
Time data	Week_Decode (created from manual)
Movement data	W<productacronym>

5.3 Preparing data mappings for data elements from sources in csv to staging and then data mapping from staging to data warehouse

Source File Name	Source File Attributes	Staging area table name	Staging area table attributes
DEMO	NAME	DEMO_FINAL	NAME
	CITY		CITY
	ZIP		ZIP
	STORE		STORE
	ZONE		ZONE
	UNEMP		UNEMP(converted unemp value in data source to percentage value)
UPC<productacronym>	COMM CODE	UPC<productacronym>	
	UPC		UPC
	DESCRIP		DESCRIP
	SIZE		
	CASE		
	NITEM		NITEM
			WH_DS (used RIGHT() to extract last digit from NITEM)
Week_Decode	WEEK#	Week_Decode	WEEK#
	START		START
	END		END
	SPECIAL EVENTS		SPECIAL EVENTS
W<productacronym>	STORE	W<productacronym>	STORE
	UPC		UPC
	WEEK		WEEK
	MOVE		MOVE
	QTY		QTY
	PRICE		PRICE
	SALE		SALE
	PROFIT		PROFIT
	OK		OK
			CATEGORY
			GROSS_MARGIN

Note: Product_<productname> files are created in the staging area by joining UPC<productacronym> and W<producta cronym> files

Staging area table	Staging table attributes	Data warehouse table	Data warehouse attributes
Week Decode		Dim Time	Time_key
	WEEK#		Week#
	START		Month
	END		Year
	SPECIAL EVENTS		Special Events
PRODUCT<product acronym>		Dim Product	Product_Key
	UPC		UPC
	DESCRIP		DESCRIP
	NITEM		NITEM
	STORE		STORE
	WEEK		WEEK
	MOVE		UNITS_SOLD
	CATEGORY		CATEGORY
	GROSS_MARGIN		GROSS_MARGIN
	QTY		QTY
STORE		Dim Store	Store_key
	STORE		STORE
	NAME		NAME
	CITY		CITY
	ZIP		ZIP
	ZONE		ZONE
	UNEMP		UNEMP
	TOTAL_SALES		TOTAL_WEEKLY_SALES
			Week
	Week		
Product_<productacronym>		FactProdSales	Product_Key
DEMO_final			Store_Key
CCOUNT_FINAL			Time-Key
Time	QTY		QTY
Store	MOVE		UNITS_SOLD
	GROSS_MARGIN		GROSS_MARGIN
Product_<productacronym>		FactTrackItem	Product_Key
DEMO_final			Store_key
CCOUNT_FINAL	WH_DS		STORAGE_TYPE
Store	QTY		QTY
	MOVE		UNITS_SOLD
	GROSS_MARGIN		GROSS_MARGIN
DEMO_final		FactUnempSales	Store_Key
CCOUNT_FINAL			Time_Key
Time	UNEMP		UNEMP
Store	TOTAL_WEEKLY_SALES		TOTAL_WEEKLY_SALES

5.4 Establishing comprehensive data extraction rules

Data extraction is the process involving retrieval of data from multiple sources for migration to a data warehouse or data lake or for further data analysis. The process of data extraction from the source files is the most crucial and the initial step involved in the ETL process of designing and implementing a data warehouse. A common byproduct of extraction is transformation of data.

The extraction of data involves combining all the data sources to a single format i.e tables in Microsoft SQL Server Studio, that can be later used for transformation and loading. The data from the sources are in the Comma Separated Value (.csv) formats. Data pertaining to our business questions was filtered for and extracted from the data source. The data for Week was extracted from the data manual for Dominick Finer Foods. The data from the source files has been loaded into the data staging area 'group7-staging-area'.

a) *Maintaining datatypes*

Data Consistency is a major issue that's faced while operating on data on SSIS. We decided to use specific data types like numeric, float for Sales and varchar for text files. Attributes were loaded as nvarchar initially from the source files and were then changed to their respective data types

b) *Naming convention*

All files were named with a common theme to allow easy understanding of tables. All products were named with their respective UPC files, movement with W. This helped us identify the difference between them.

c) *Operations during joins*

Prior to performing joins sorting was carried out to the key on which the joins would take place. Merge joins were used on tables with the rule of using a left join on the product.

5.5 Determining data transformation and loading rules

Once data is extracted from the appropriate sources, it is subjected to transformation and loading. Post extraction of data and its storage in the staging area, we applied necessary transformation rules to sanitize the data and eliminate redundant, inconsistent and invalid records. The output of the transformation process is used to create data warehouse tables. i.e dimension and fact tables which support the decision making processes.

a) *Removal of null values*

The extracted data contained blanks and '.' which would cause issues during loading thus we deleted those rows in the CCOUNT and DEMO tables.

b) *Removal of dirty data*

Only attributes of each table pertaining to our business questions were retained and the others were filtered out prior to loading into the staging area. Records containing blanks, '.', junk values in the SIZE column of UPC files and incorrect negatives such as negative sales records were deleted.

c) *Data conversion*

Date attribute in the source data is stored as varchar but is converted to Date datatype in the staging area using SQL string manipulation methods. Day, month and year are extracted from Date using date manipulation functions. All tables were loaded into the staging area as nvarchar datatype but were later converted to their appropriate data types.

d) *Creation of surrogate keys*

Surrogate keys are data warehouse keys which are incremented by 1 automatically per record. They were created as primary keys for every dimension table and referenced in every fact table to form the concatenated foreign key.

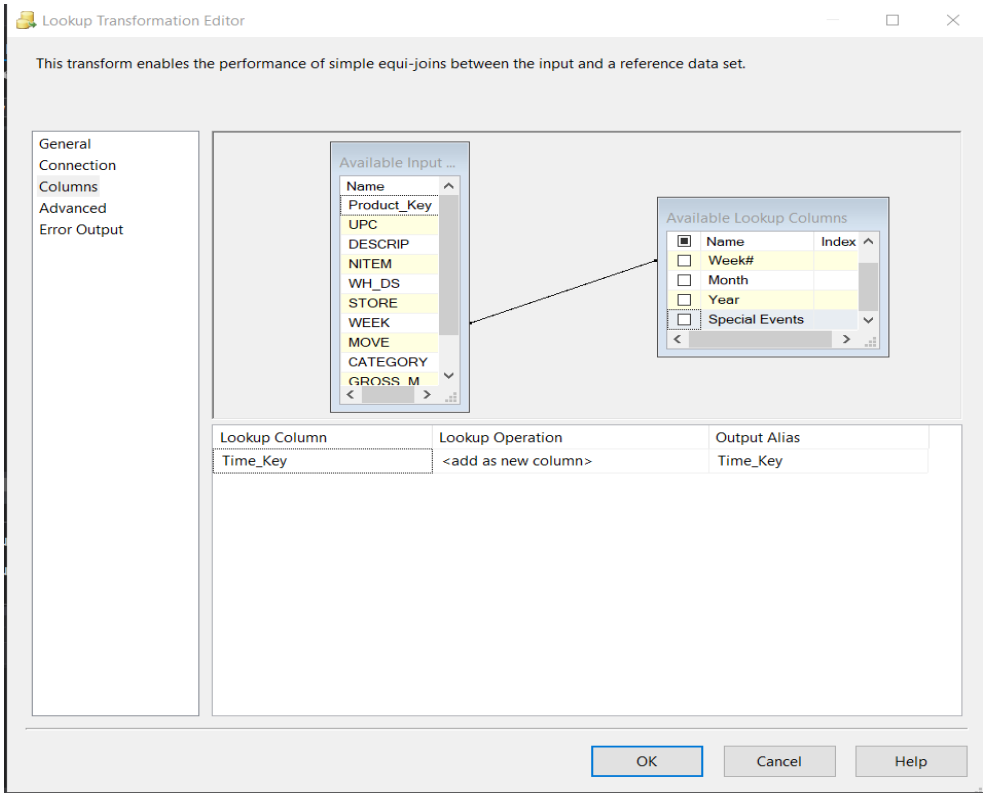
e) *Derived attributes*

1. Total_Sales per store was determined by calculating the sum of all product sales and difference of the coupons in the CCOUNT table
2. Gross Margin was determined by the function Gross margin = Price*Move/Qty
3. Unemp% was obtained by converting Unemp values to percentage
4. Year and month in DimTime were obtained from YEAR (date) and MONTH (date) respectively

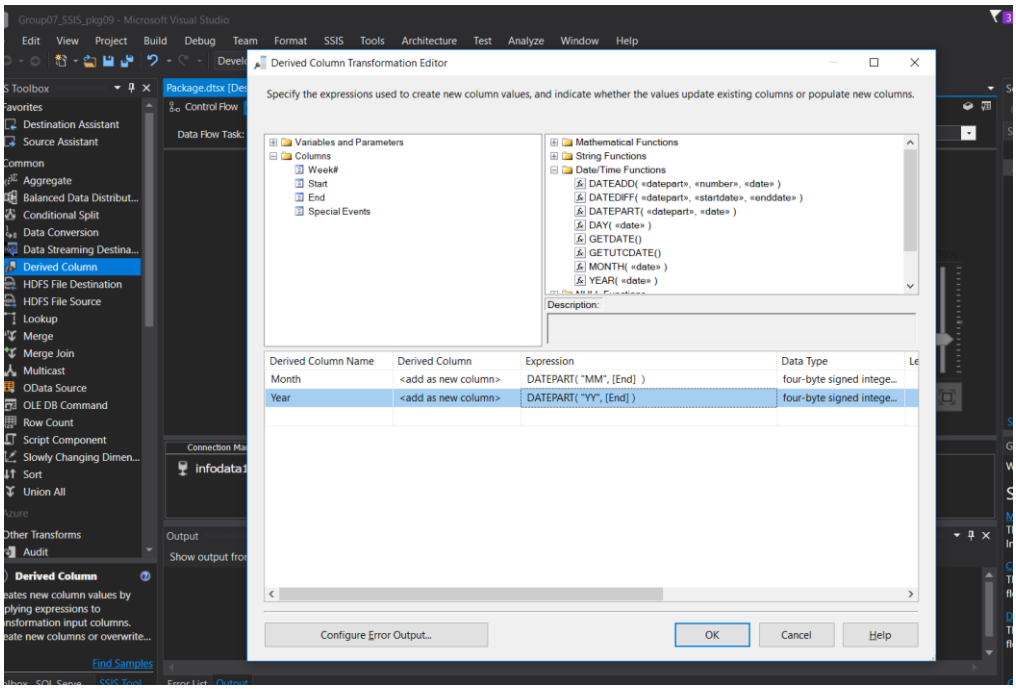
5.6 SSIS functions

The SSIS functions used in the transformation and cleaning processes are:

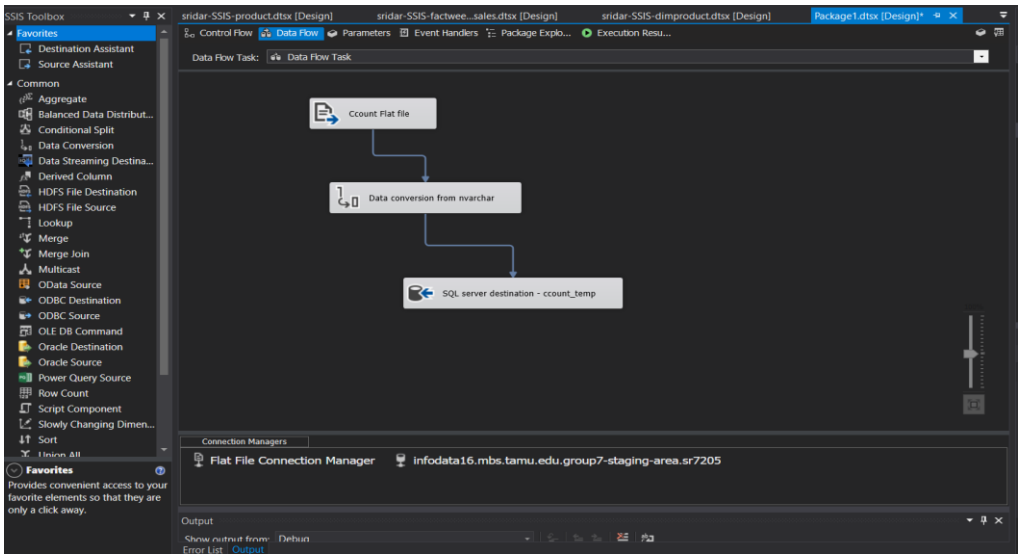
1. **LOOKUP**: As the term suggests, this function joins additional columns to the data flow by looking up values in a table. We implemented lookups to acquire supplementary information in related tables which is based on values in common columns.



2. **DERIVED COLUMN**: Derived columns create new column values by applying expressions to input columns. We calculated YEAR and MONTH using derived columns.



3. **DATA CONVERSION**: Converts data from one data type to another. For example, we converted data of datatype nvarchar to float, int, etc.



5.7 Plan for aggregate tables

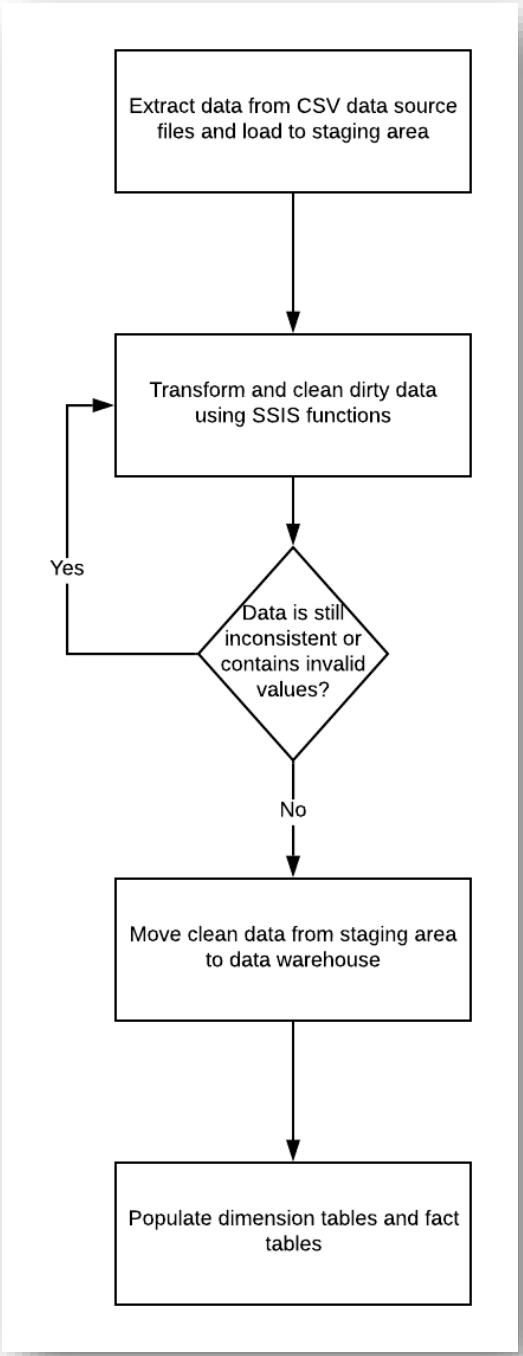
Aggregate function aggregates data with functions. We implemented count, sum, group by functions to aid in our transformation process. We calculated *Total_Weekly_Sales* by grouping the stores and their corresponding weekly sales. We summed the product sales per store and then calculated the difference of coupons from the result.

5.8 Organization of data staging area

Staging area is the center stage of the data warehousing process given ETL is performed here and large amount of data is constantly moved to and from the staging area thus increasing the chances of loss of data. In such a scenario, organization of the staging area becomes essential. 'Group7staging area' is organized as shown below where CCOUNT_FINAL is the customer count file, DEMO_final is the demographics file, Store, Time and Week_Decode files are created from the manual, UPC<productacronym> files are the UPC files, W<productacronym> files are the movement files and Product_<productacronym> files are the mergers between UPC and movement files.

5.9 Procedure for data extraction and loading

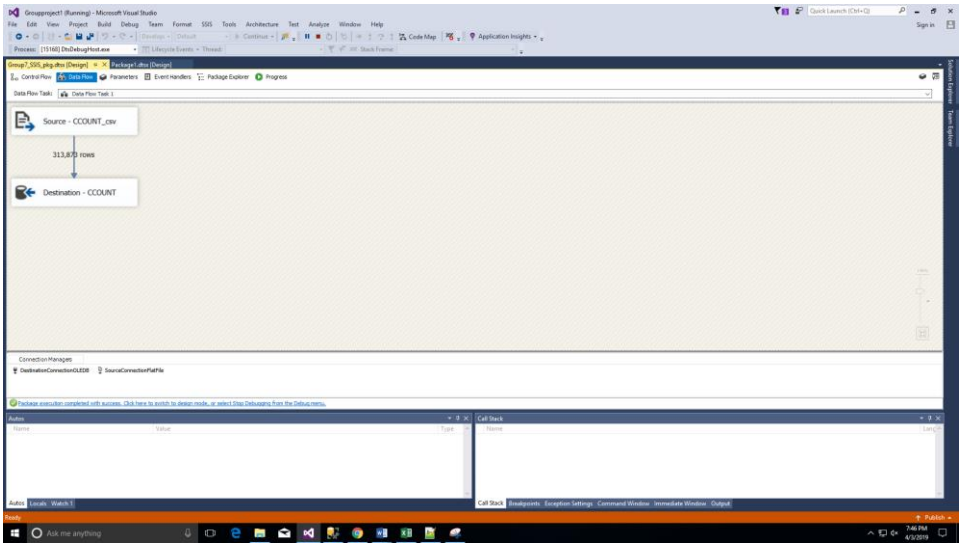
The sequence of events that comprise the data extraction and loading process is depicted by the flowchart below –



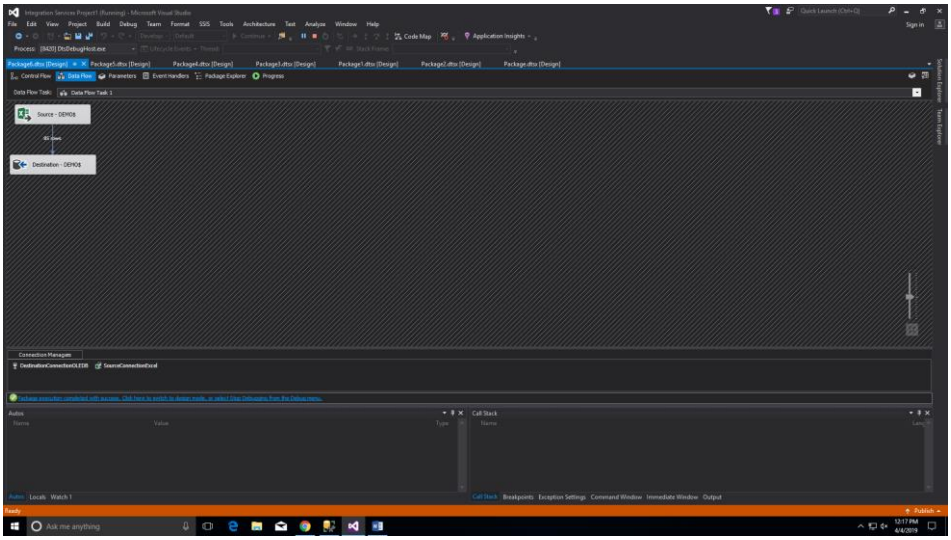
5.10 ETL Implementation

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

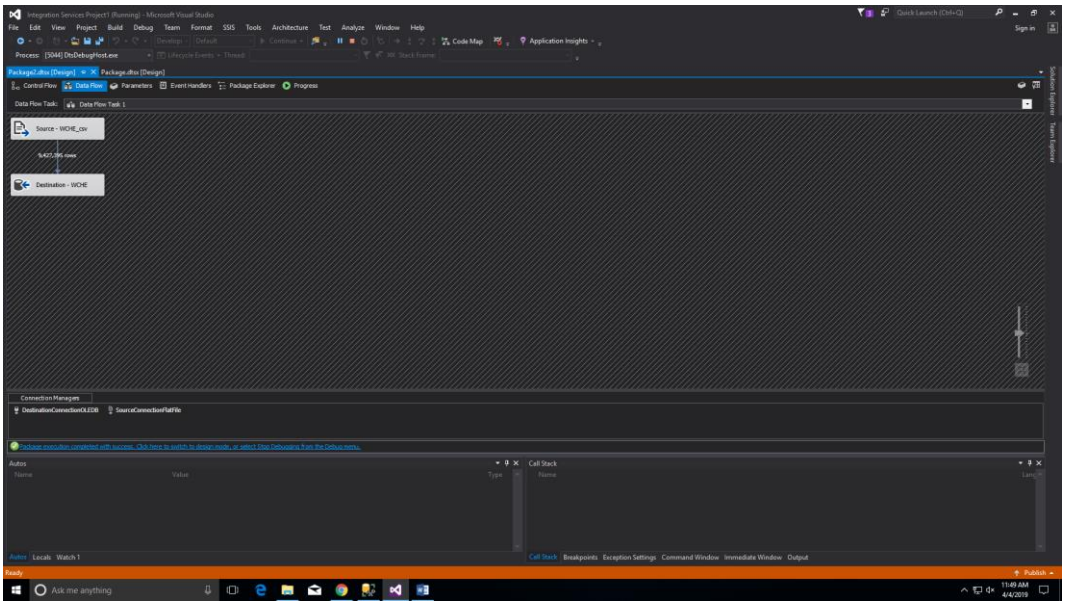
5.10.1 Extraction and transformation of source data into staging area



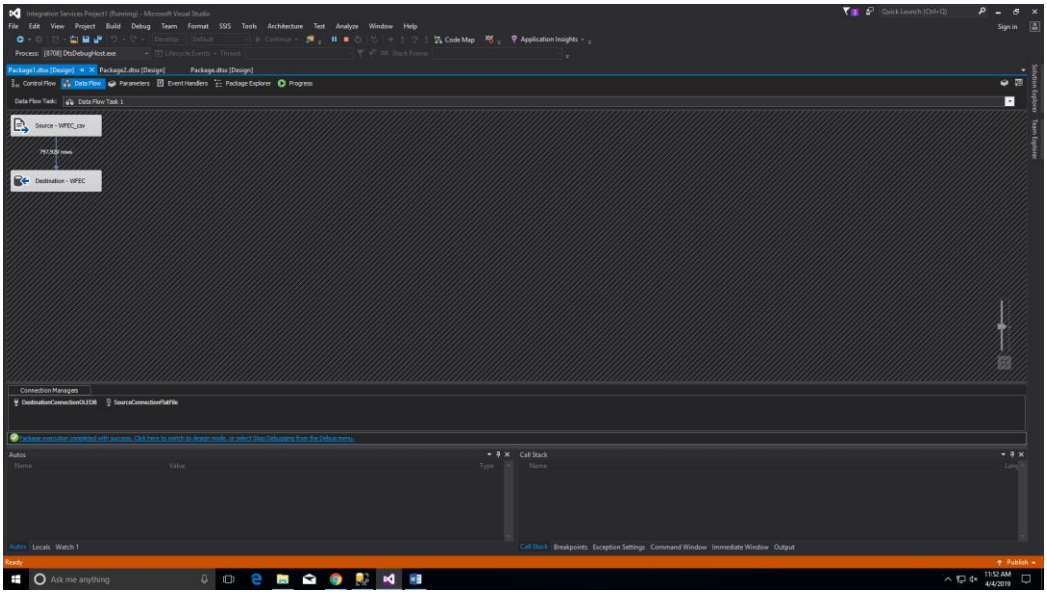
Extraction of CCOUNT from source to staging area



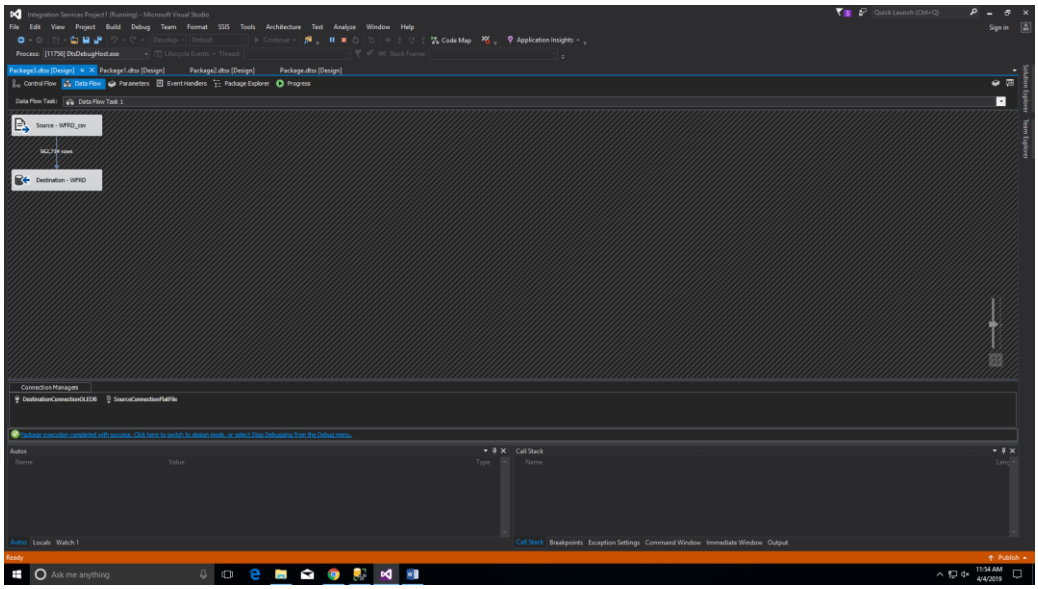
Extraction of DEMO from source to staging area



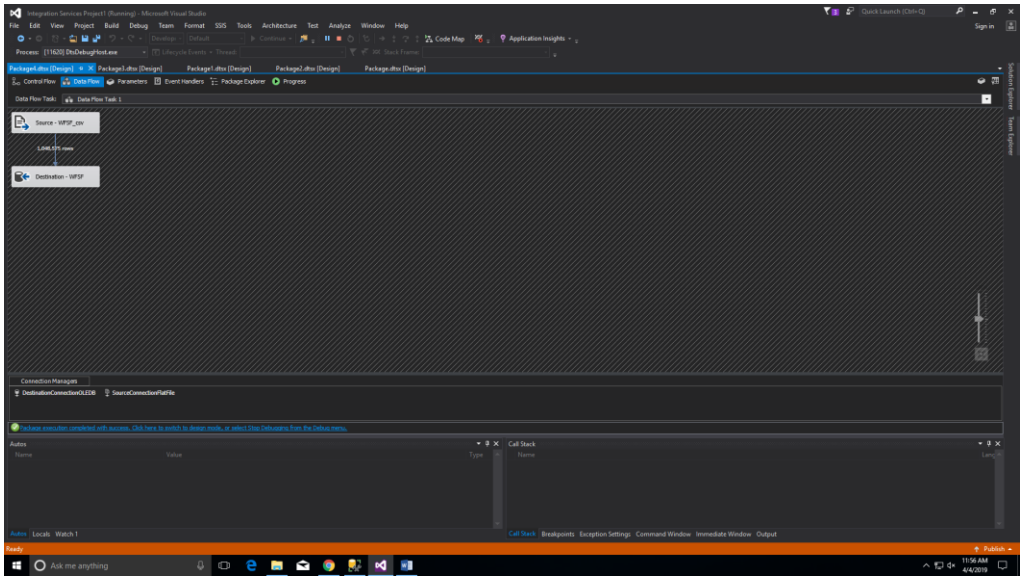
Extraction of WCHE from source to staging area



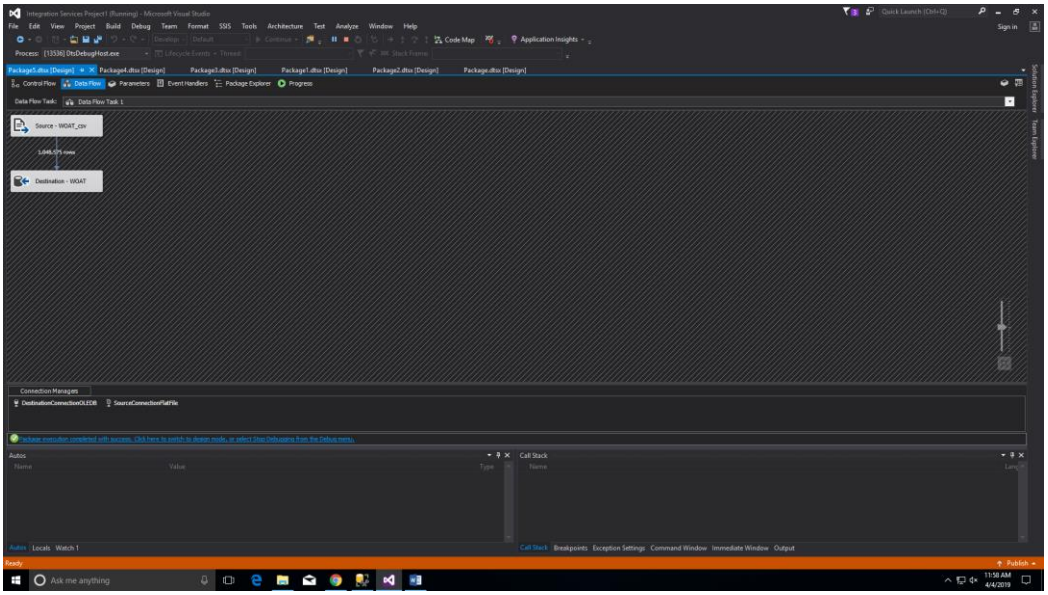
Extraction of WFEC from source to staging area



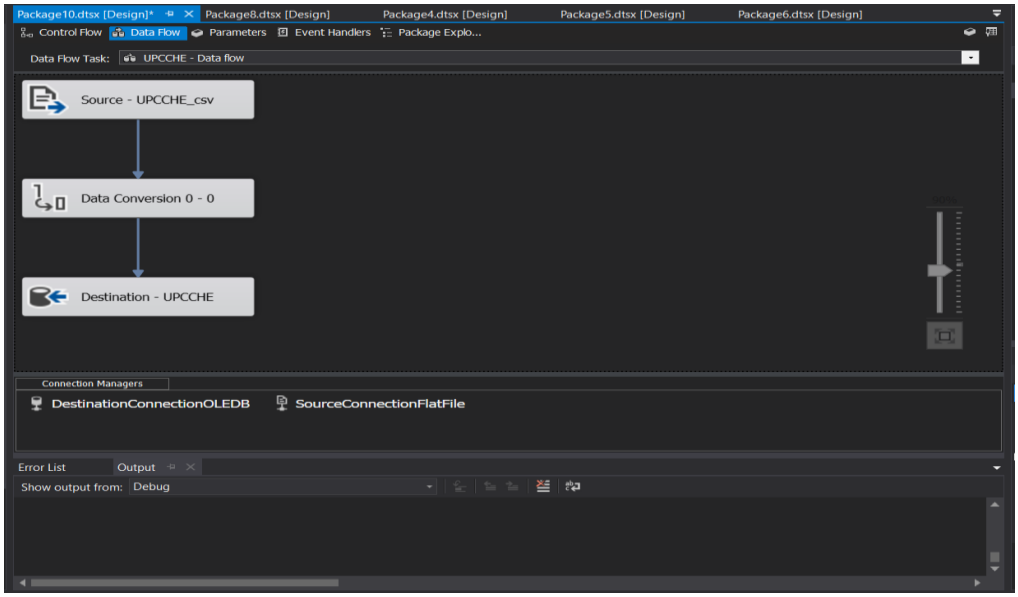
Extraction of WFRD from source to staging area



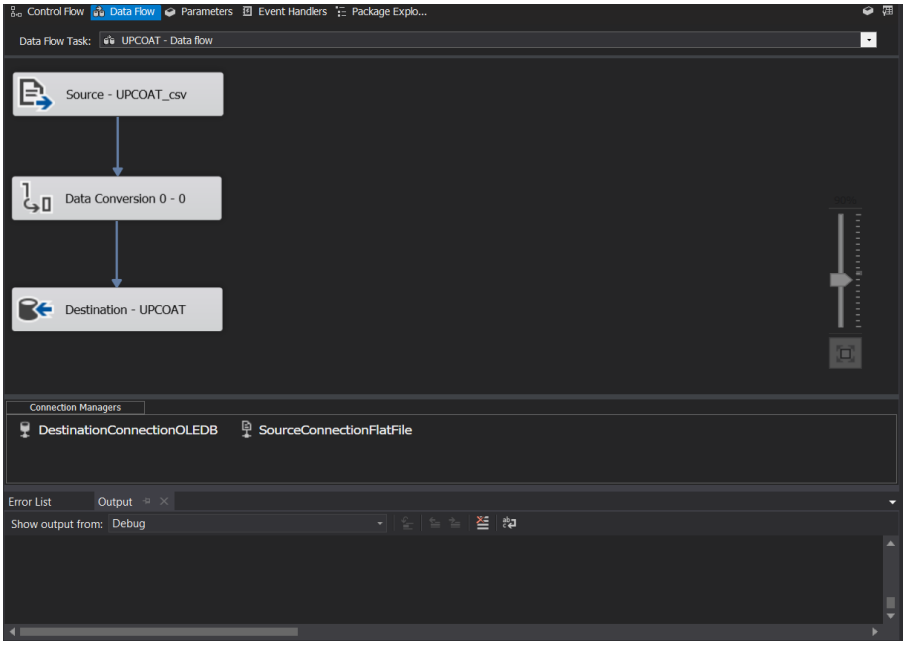
Extraction of WFSF from source to staging area



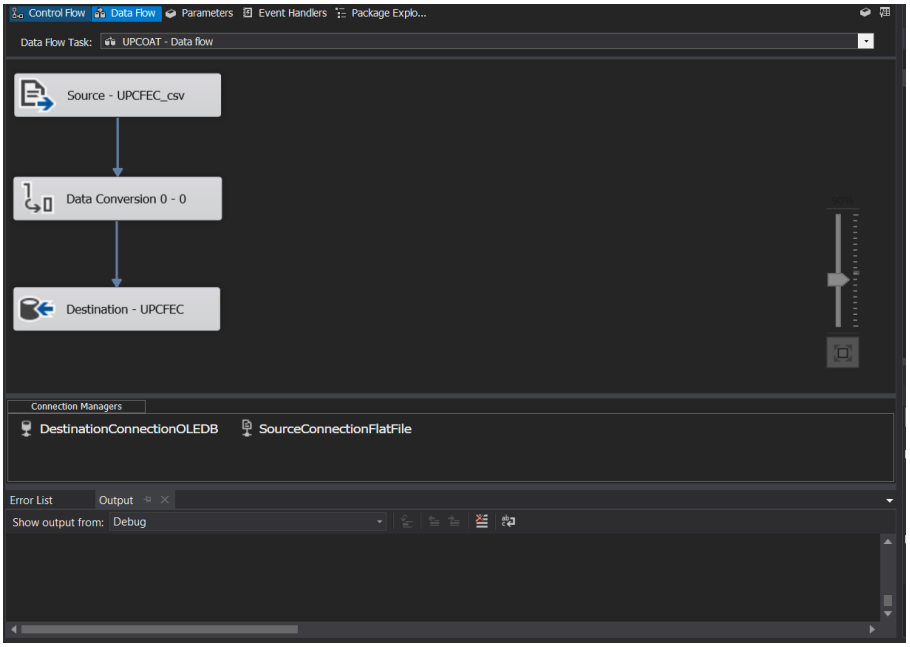
Extraction of WOAT from source to staging area



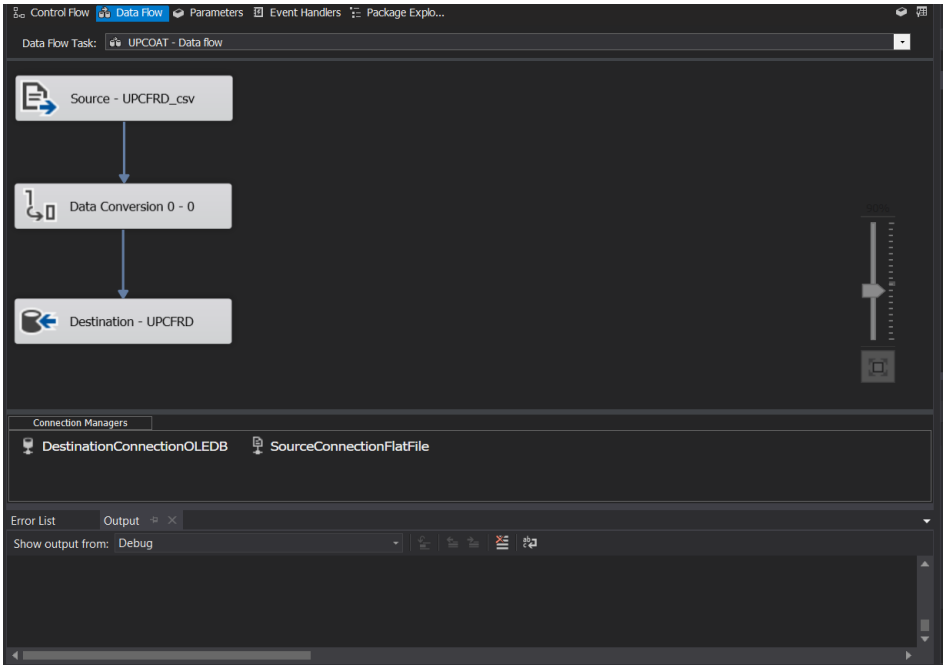
Extraction of UPCCHE from source to staging area



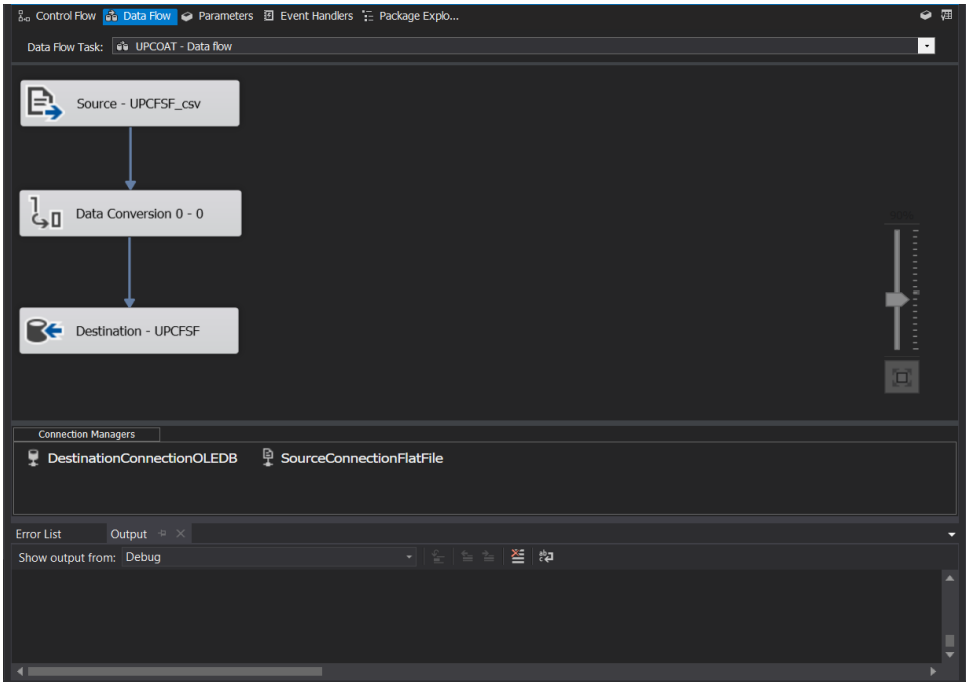
Extraction of UPCOAT from source to staging area



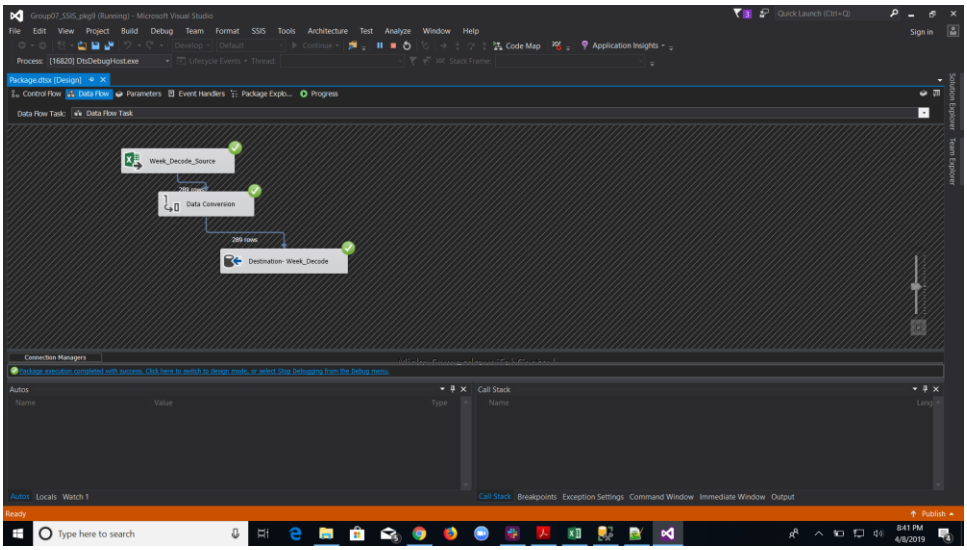
Extraction of UPCFEC from source to staging area



Extraction of UPCFRD from source to staging area



Extraction of UPCFSF from source to staging area



Extraction of Week_Decode from source to staging area

5.10.2 Transformation and cleaning of data

5.10.2.1 Data cleaning

SSIS SQL task functionalities were used to clean data in staging area before performing transformation operations. Below are the list of SQL statements and snap shots of SQL tasks created –

SQL for Creating CCount

```
SELECT * INTO CCOUNT_FINAL FROM CCOUNT
```

SQL for Cleaning Negative week Values

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

SQL for removing “” characters from columns

```
UPDATE [dbo].[WCHE]
SET ["SALE"] = SUBSTRING(["SALE"], 2, LEN(["SALE"]))
WHERE LEFT(["SALE"], 1) = "";

UPDATE [dbo].[WCHE]
SET ["SALE"] = SUBSTRING(["SALE"], 1, LEN(["SALE"])-1)
WHERE RIGHT(["SALE"], 1) = "";
```

SQL for extracting storage type value from Nitem code

```
UPDATE [dbo].[UPCFRD_Final]
SET    WH_DS = Cast(Right([NITEM], 1) AS Int)
WHERE  [NITEM] IS NOT NULL;
```

SQL for Cleaning Invalid Store Values:

```
DELETE FROM CCOUNT_FINAL
WHERE (["STORE"] NOT IN
(SELECT Store_number FROM dimStore))
```

SQL for Cleaning Invalid Values:

```
DELETE FROM CCOUNT_FINAL
WHERE (["WEEK"] IS NULL)
```

SQL for Cleaning Date Values

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

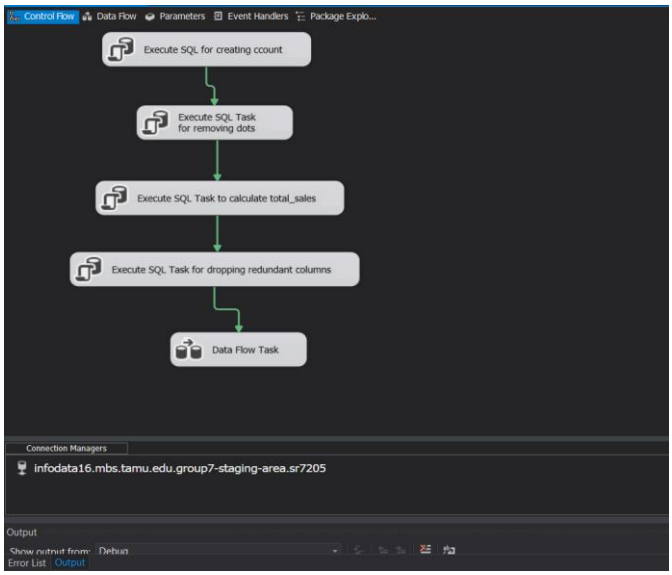
SQL for removing invalid store number

```
DELETE from CCOUNT_FINAL
WHERE [“STORE”] NOT IN (SELECT [Store_number] from dimStore);
```

SQL for calculating total sales

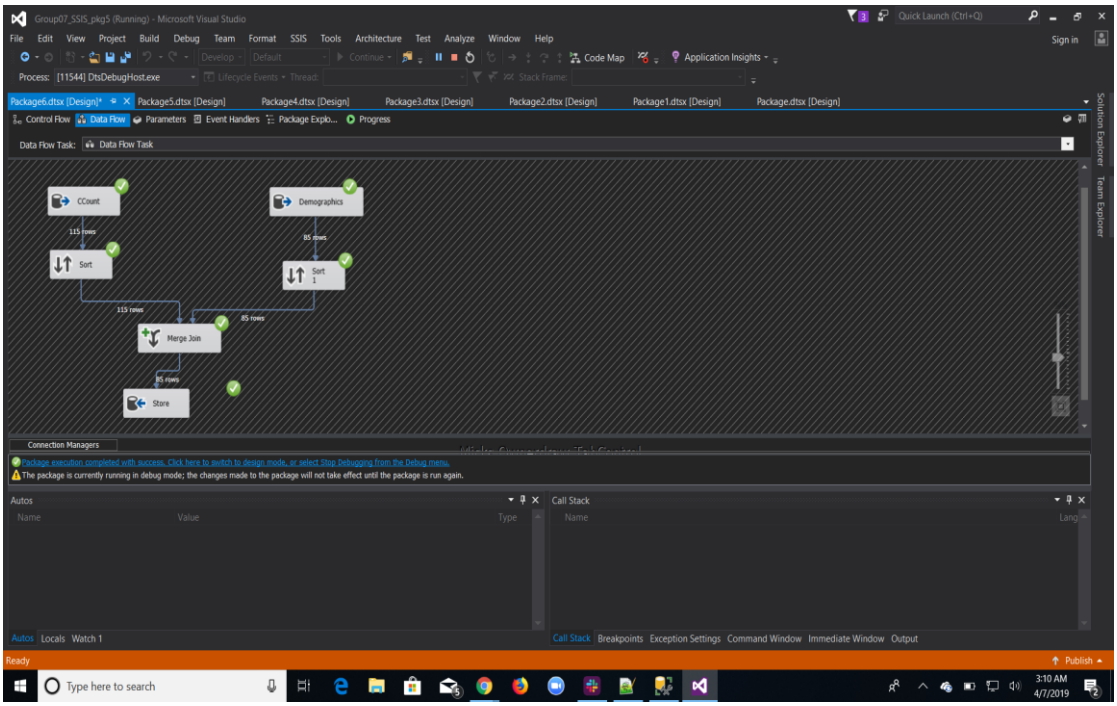
```
ALTER TABLE CCOUNT_FINAL add column total_sales float;

UPDATE TABLE CCOUNT_temp_1 set total_sales = [GROCERY] +
[DAIRY] + [FROZEN] + [BOTTLE]+[MVPCLUB] + [GROCCOUP] +
[MEAT] + [MEATFROZ] + [MEATCOUP] +[FISH] + [FISHCOUP] +
[PROMO] + [PROMCOUP] +  [PRODUCE] + [BULK] +  [SALADBAR]
+ [PRODCOUP] + [BULKCOUP] + [SALCOUP] + [FLORAL] +
[FLORCOUP] + [DELI] + [DELISELF] + [DELIEXPR] + [CONVFOOD] +
[CHEESE] + [DELICOUP] +  [BAKERY] + [PHARMACY] + [PHARCOUP]
+ [GM] + [JEWELRY] + [COSMETIC] + [HABA] + [GMCOUP]  [CAMERA]
+ [VIDEO] + [VIDEOREN] + [VIDCOUP] + [BEER] + [WINE] + [SPIRITS]
+ [MISCSCP] +  [MANCOUP] + [FTGCHIN] + [FTGCCOUP] + [FTGITAL]
+ [FTGICOUP] + [DAIRCOUP] + [FROZCOUP] +  [HABACOUP] +
```

[PHOTCOUP] + [COSMCOUP] + [SSDELICP] + [BAKCOUP] + [LIQCOUP]

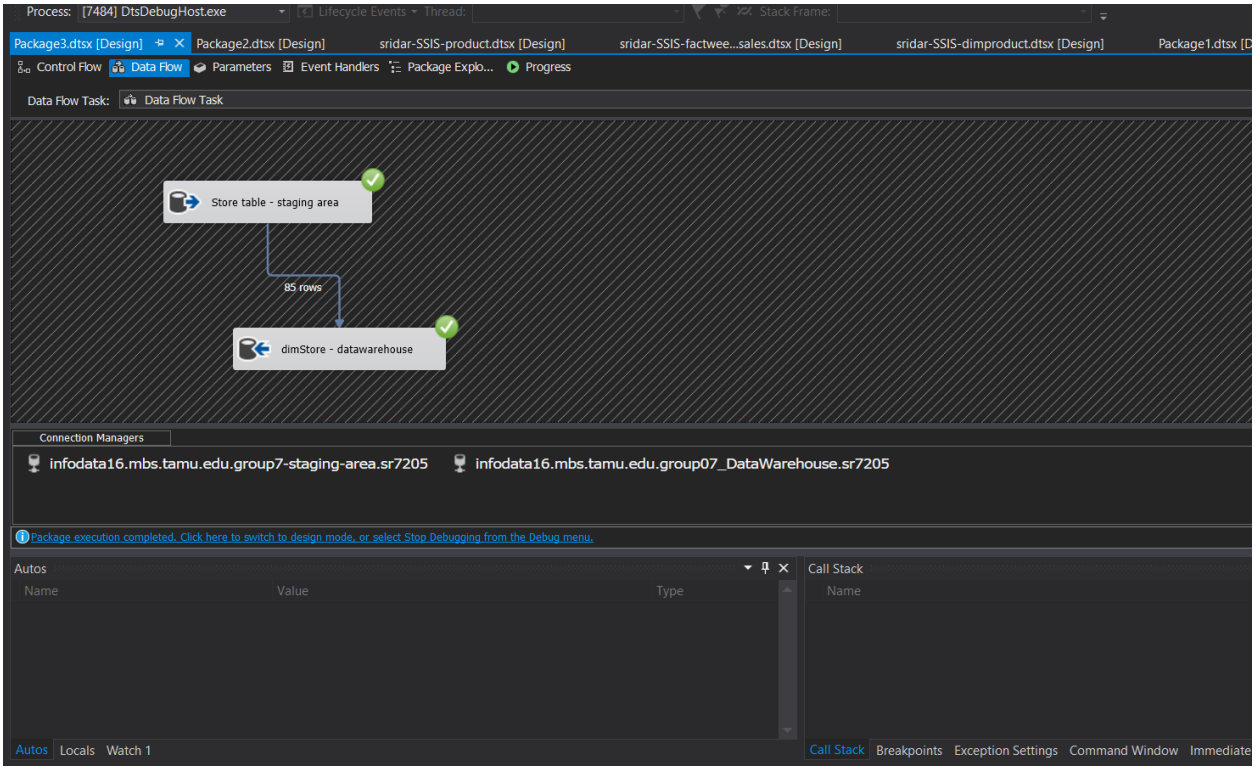


5.10.2.2 Transformation of source data into dimension tables

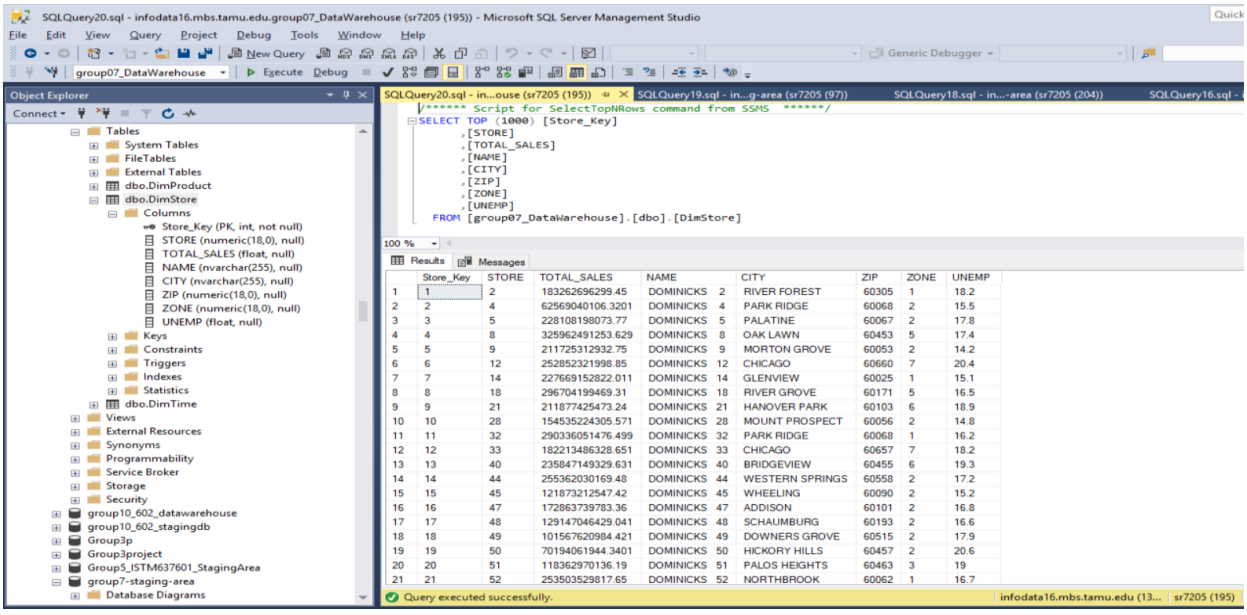
- DimStore dimension table creation



Creating a Merge Join between CCount and Store data in Staging

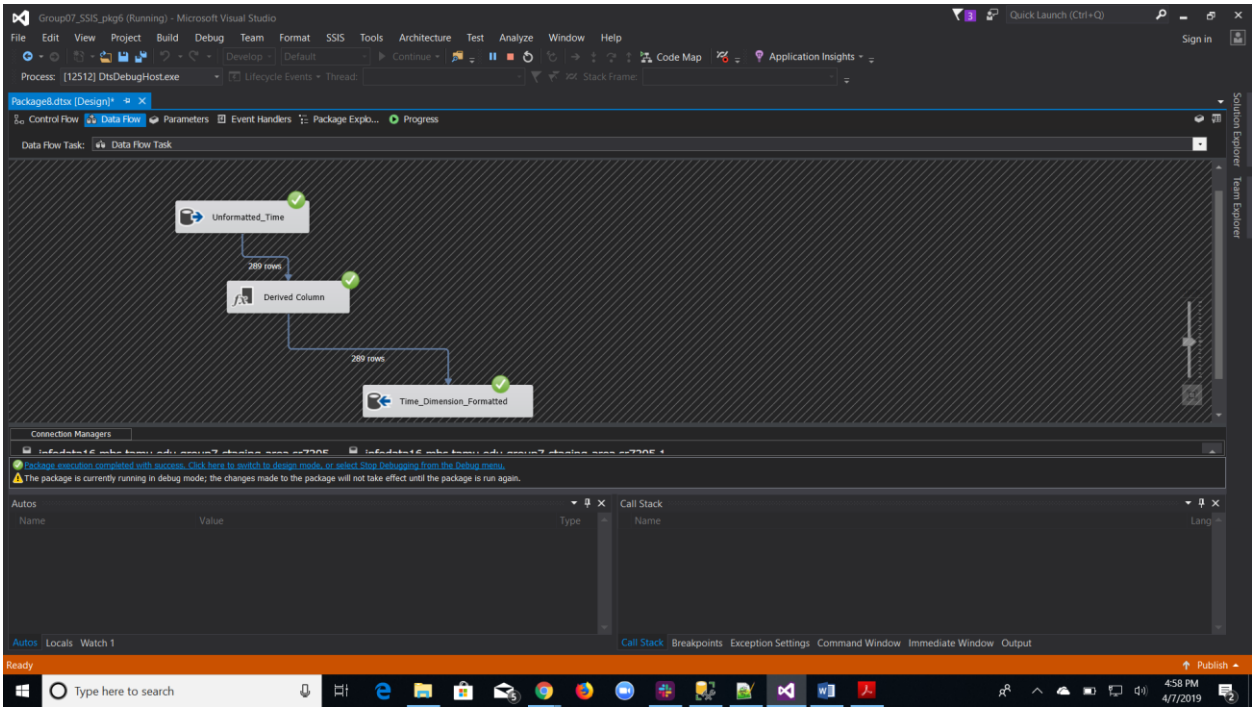


Data Moved to Staging

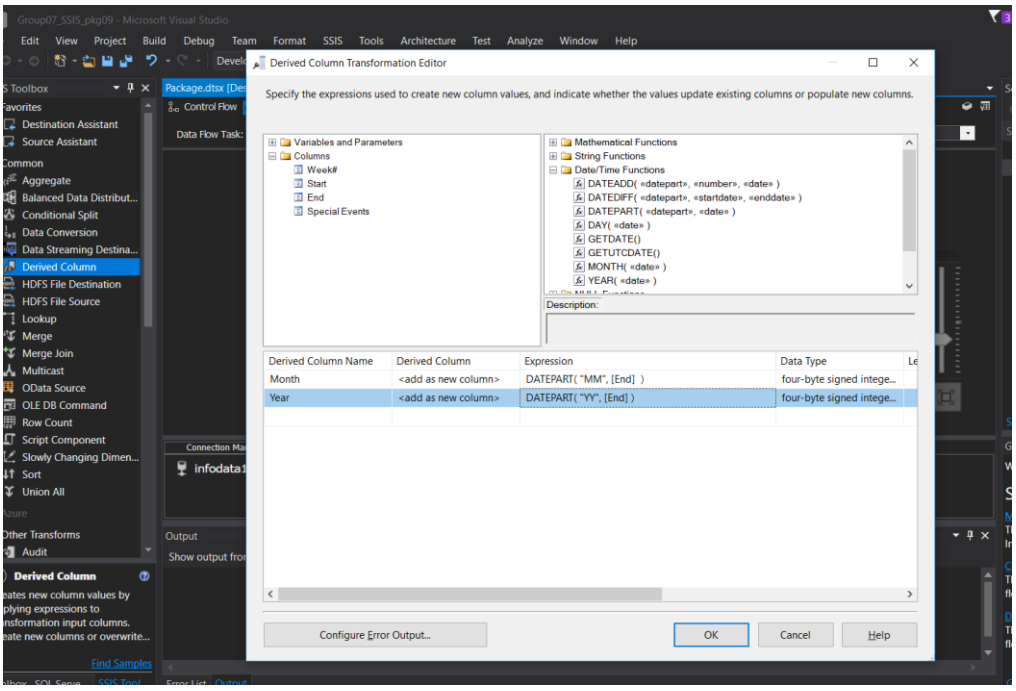


Snapshot of DimStore dimension table in data warehouse

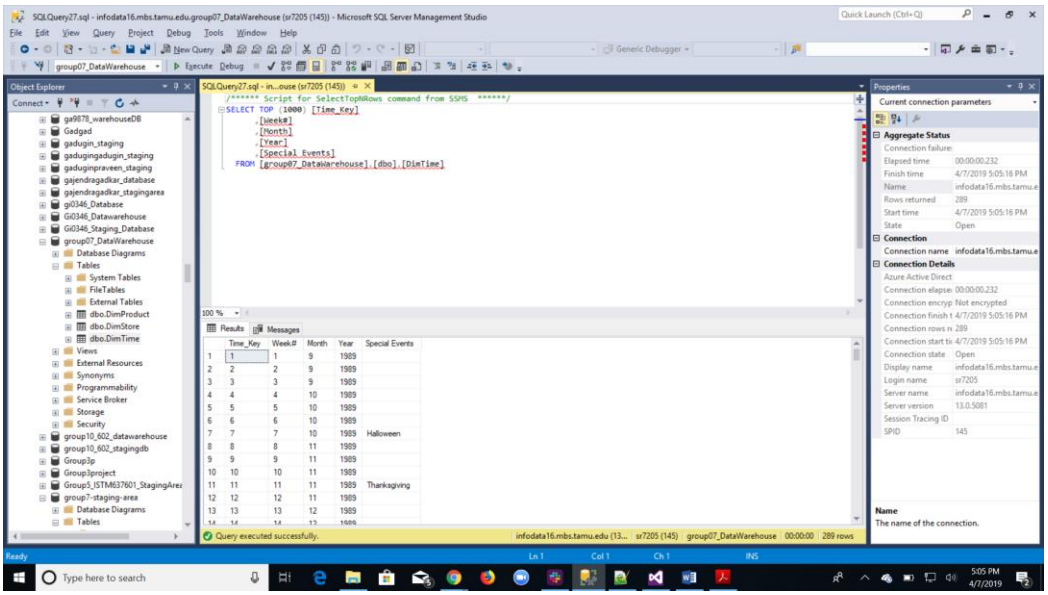
- DimTime dimension table creation



Formatting time as per our requirements

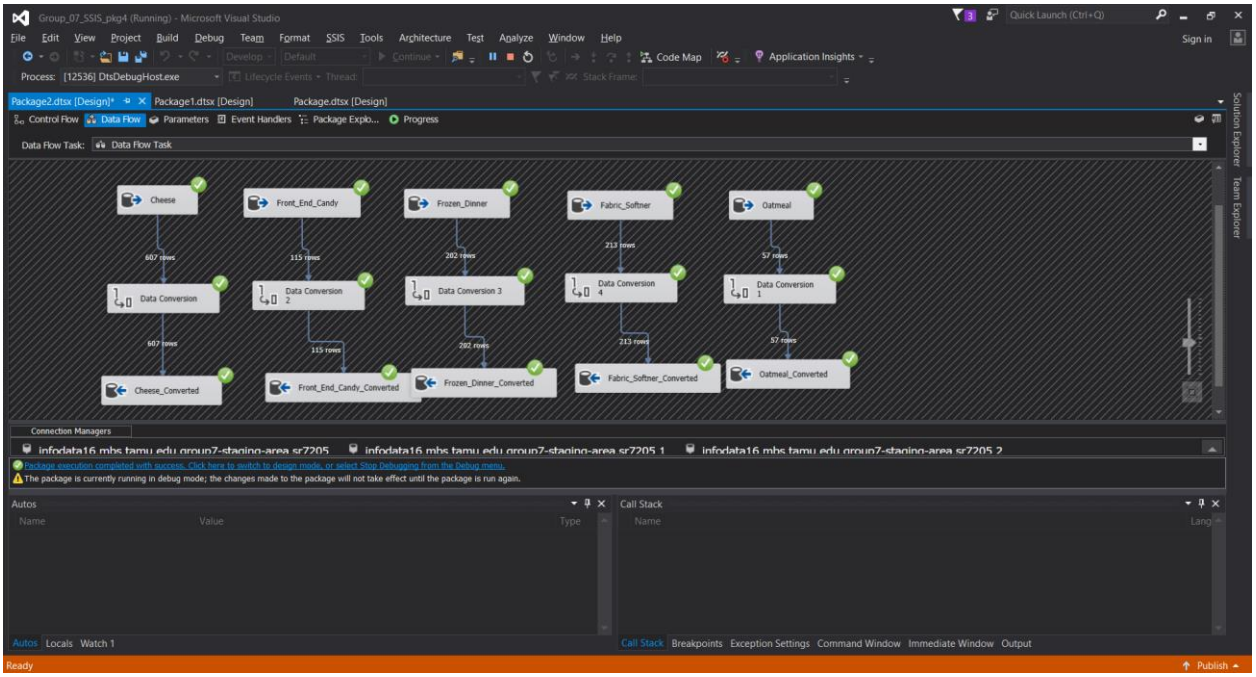


Using DatePart to segregate the contents within the Time table

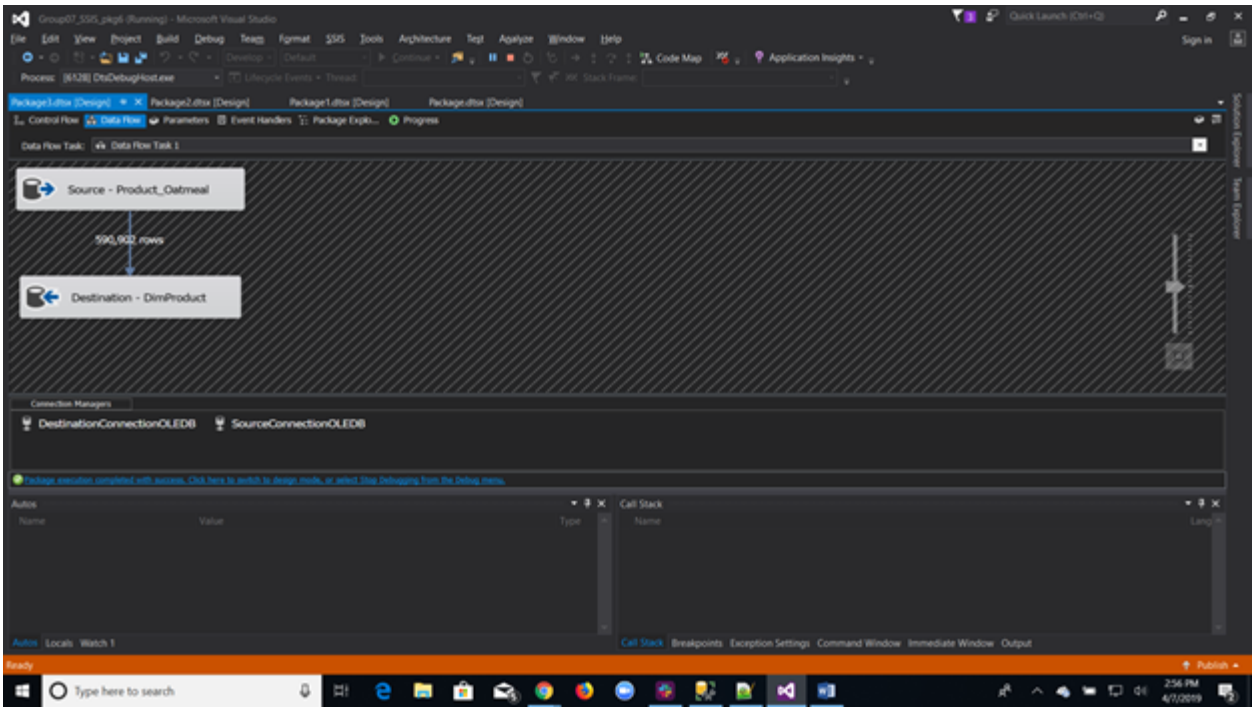


Snapshot of DimTime dimension table in data warehouse

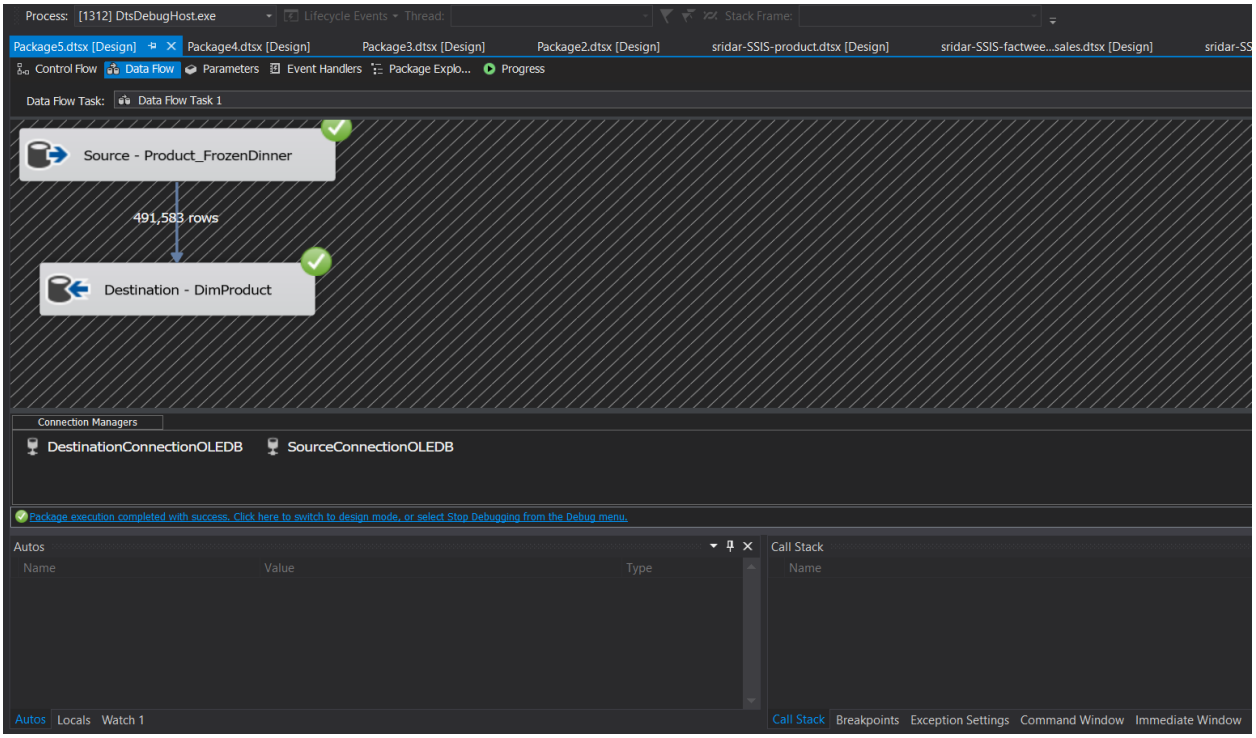
- DimProduct dimension table creation



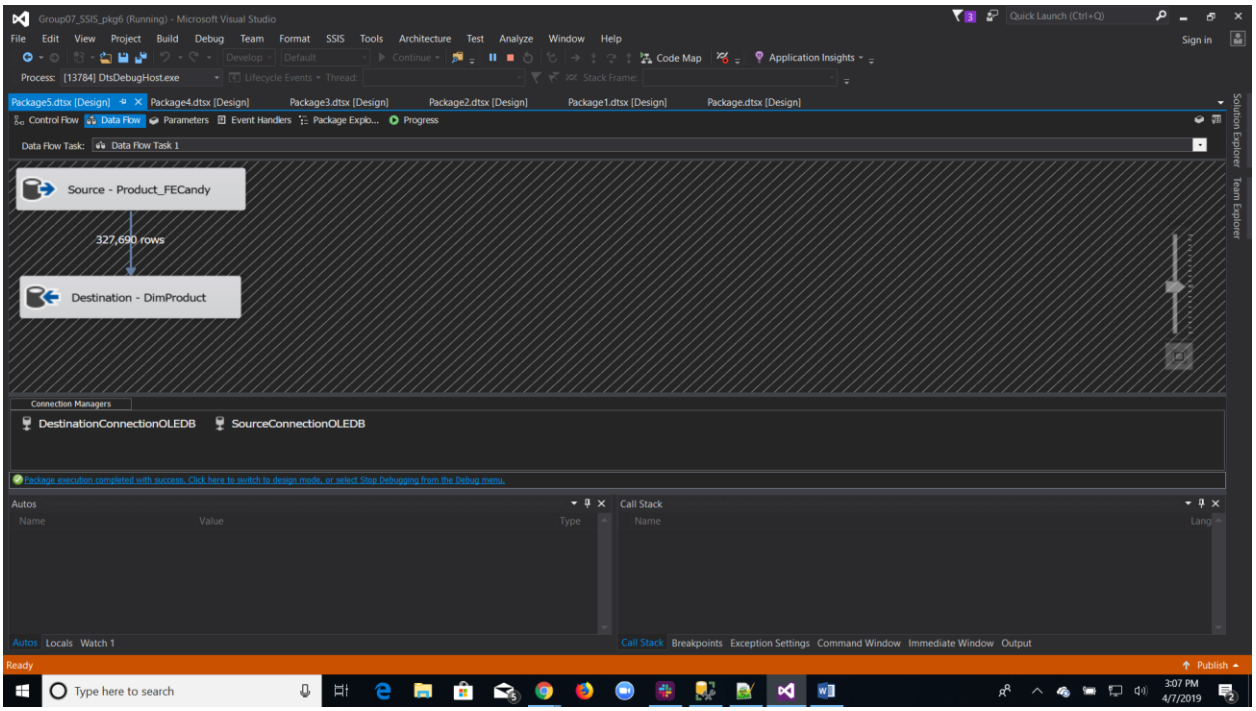
Products loaded into staging area



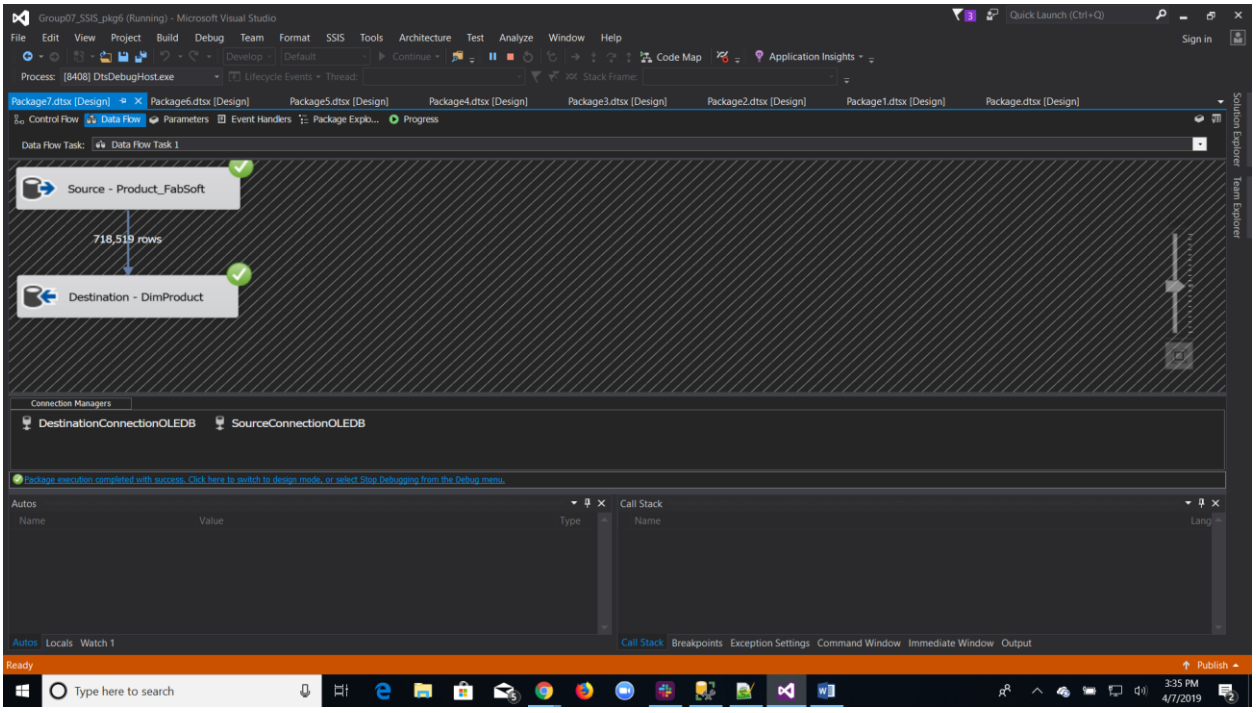
Products appended into the DimProduct dimension table



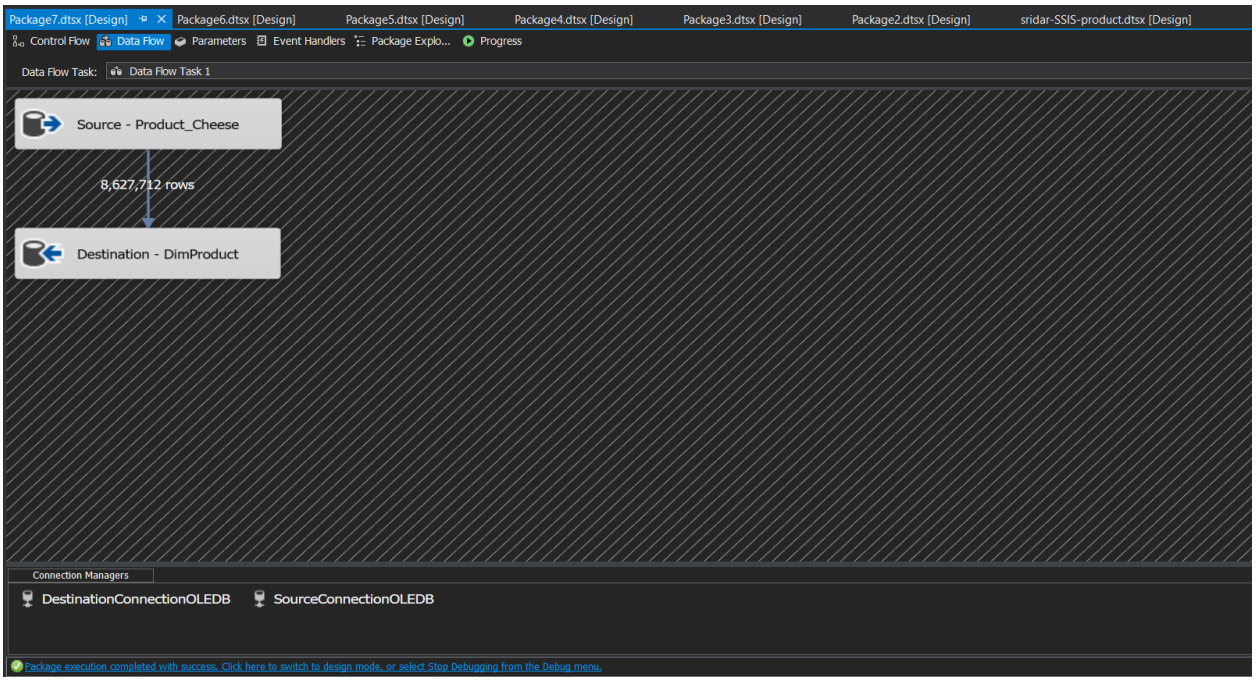
Products appended into the DimProduct dimension table

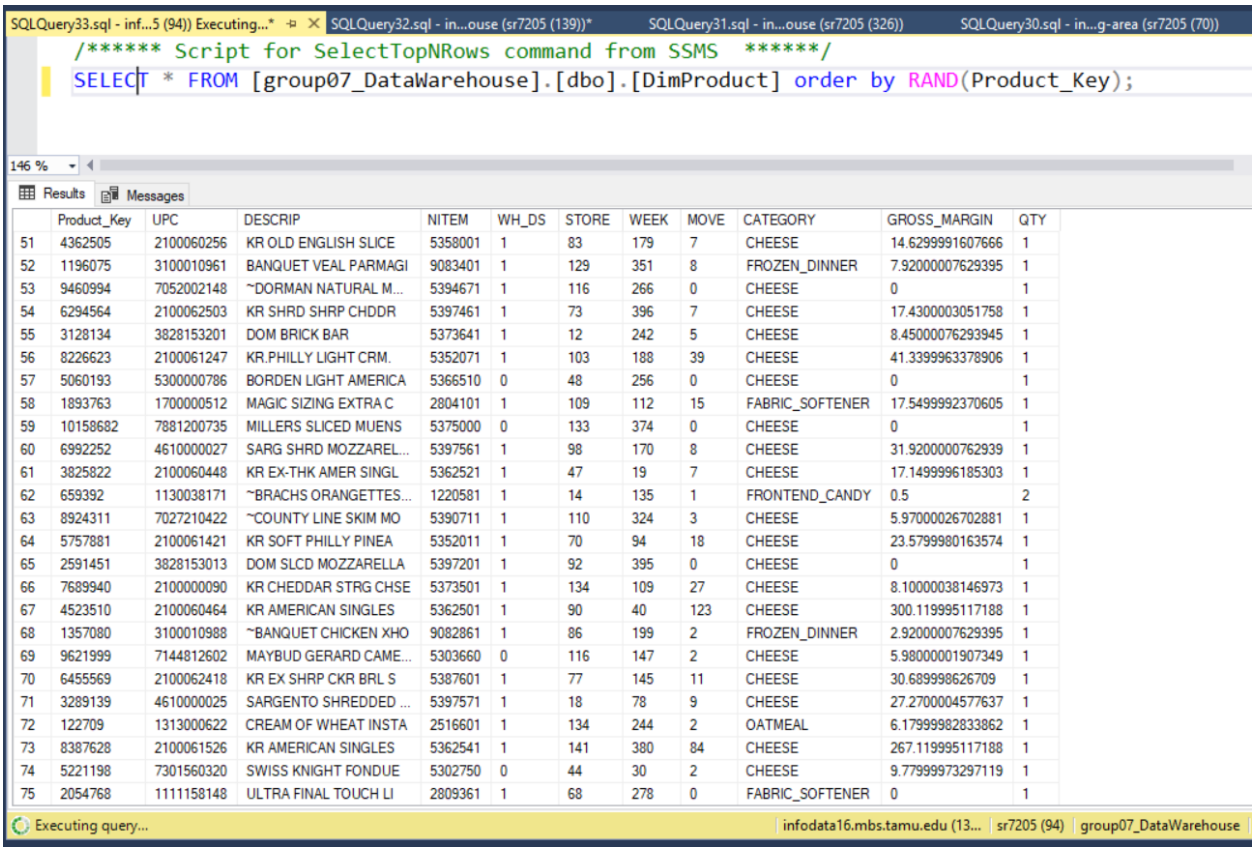


Products appended into the DimProduct dimension table



Products appended into the DimProduct dimension table





	Product_Key	UPC	DESCRIP	NITEM	WH_DS	STORE	WEEK	MOVE	CATEGORY	GROSS_MARGIN	QTY
51	4362505	2100060256	KR OLD ENGLISH SLICE	5358001	1	83	179	7	CHEESE	14.6299991607666	1
52	1196075	3100010961	BANQUET VEAL PARMAGI	9083401	1	129	351	8	FROZEN_DINNER	7.92000007629395	1
53	9460994	7052002148	~DORMAN NATURAL M...	5394671	1	116	266	0	CHEESE	0	1
54	6294564	2100062503	KR SHRD SHRP CHDDR	5397461	1	73	396	7	CHEESE	17.4300003051758	1
55	3128134	3828153201	DOM BRICK BAR	5373641	1	12	242	5	CHEESE	8.45000076293945	1
56	8226623	2100061247	KR.PHILLY LIGHT CRM.	5352071	1	103	188	39	CHEESE	41.3399963378906	1
57	5060193	5300000786	BORDEN LIGHT AMERICA	5366510	0	48	256	0	CHEESE	0	1
58	1893763	1700000512	MAGIC SIZING EXTRA C	2804101	1	109	112	15	FABRIC_SOFTENER	17.5499992370605	1
59	10158682	7881200735	MILLERS SLICED MUENS	5375000	0	133	374	0	CHEESE	0	1
60	6992252	4610000027	SARG SHRD MOZZAREL...	5397561	1	98	170	8	CHEESE	31.9200000762939	1
61	3825822	2100060448	KR EX-THK AMER SINGL	5362521	1	47	19	7	CHEESE	17.1499996185303	1
62	659392	1130038171	~BRACHS ORANGETTES...	1220581	1	14	135	1	FRONTEND_CANDY	0.5	2
63	8924311	7027210422	~COUNTY LINE SKIM MO	5390711	1	110	324	3	CHEESE	5.97000026702881	1
64	5757881	2100061421	KR SOFT PHILLY PINEA	5352011	1	70	94	18	CHEESE	23.5799980163574	1
65	2591451	3828153013	DOM SLCD MOZZARELLA	5397201	1	92	395	0	CHEESE	0	1
66	7689940	2100000090	KR CHEDDAR STRG CHSE	5373501	1	134	109	27	CHEESE	8.10000038146973	1
67	4523510	2100060464	KR AMERICAN SINGLES	5362501	1	90	40	123	CHEESE	300.119995117188	1
68	1357080	3100010988	~BANQUET CHICKEN XHO	9082861	1	86	199	2	FROZEN_DINNER	2.92000007629395	1
69	9621999	7144812602	MAYBUD GERARD CAME...	5303660	0	116	147	2	CHEESE	5.98000001907349	1
70	6455569	2100062418	KR EX SHRP CKR BRL S	5387601	1	77	145	11	CHEESE	30.689998626709	1
71	3289139	4610000025	SARGENTO SHREDDED ...	5397571	1	18	78	9	CHEESE	27.2700004577637	1
72	122709	1313000622	CREAM OF WHEAT INSTA	2516601	1	134	244	2	OATMEAL	6.17999982833862	1
73	8387628	2100061526	KR AMERICAN SINGLES	5362541	1	141	380	84	CHEESE	267.119995117188	1
74	5221198	7301560320	SWISS KNIGHT FONDUE	5302750	0	44	30	2	CHEESE	9.77999973297119	1
75	2054768	1111158148	ULTRA FINAL TOUCH LI	2809361	1	68	278	0	FABRIC_SOFTENER	0	1

Snapshot of DimProduct dimension table in data warehouse

5.10.2.3 Granularity of data marts

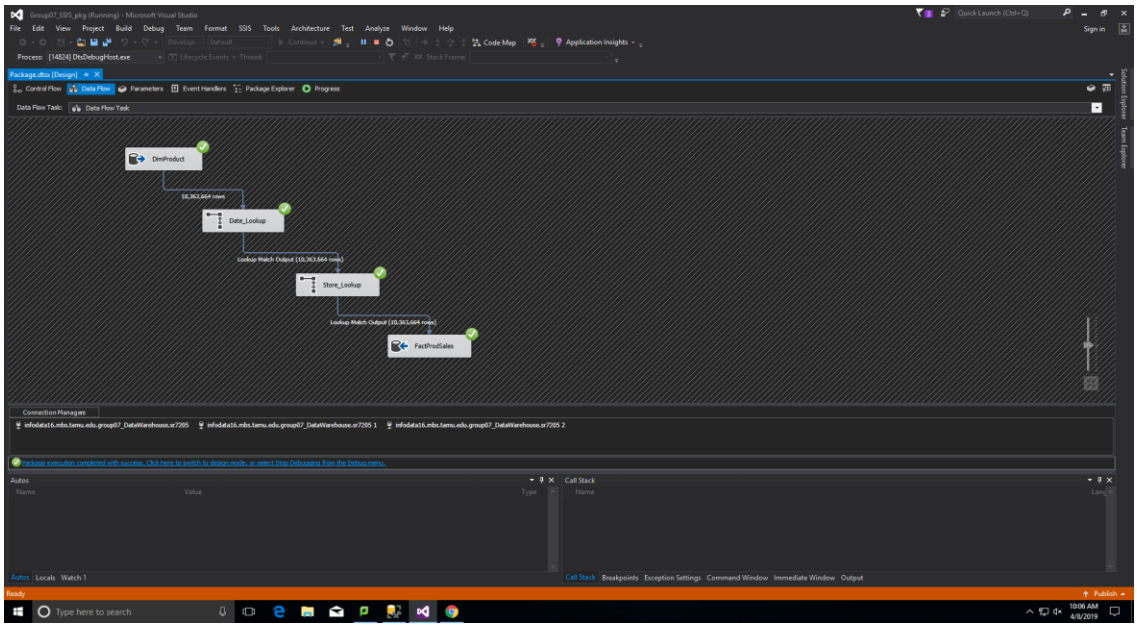
We have implemented a drill-down approach to facilitate a low-level granularity in our data marts. As a result, our processed data can answer business questions of varied depth. We adopted this technique to get different data patterns to guide our analysis in our five business questions and beyond. Since granularity has a major impact on query execution, we spent significant time and resources collecting the requisite details of the data under consideration to achieve low level granularity.

FactProdSales: In FactProdSales we use three dimension tables which provide us a drill-down of the *weekly sales, product name and the store* it is sold at. This provides the business user a detailed overview of a top to bottom picture of the process.

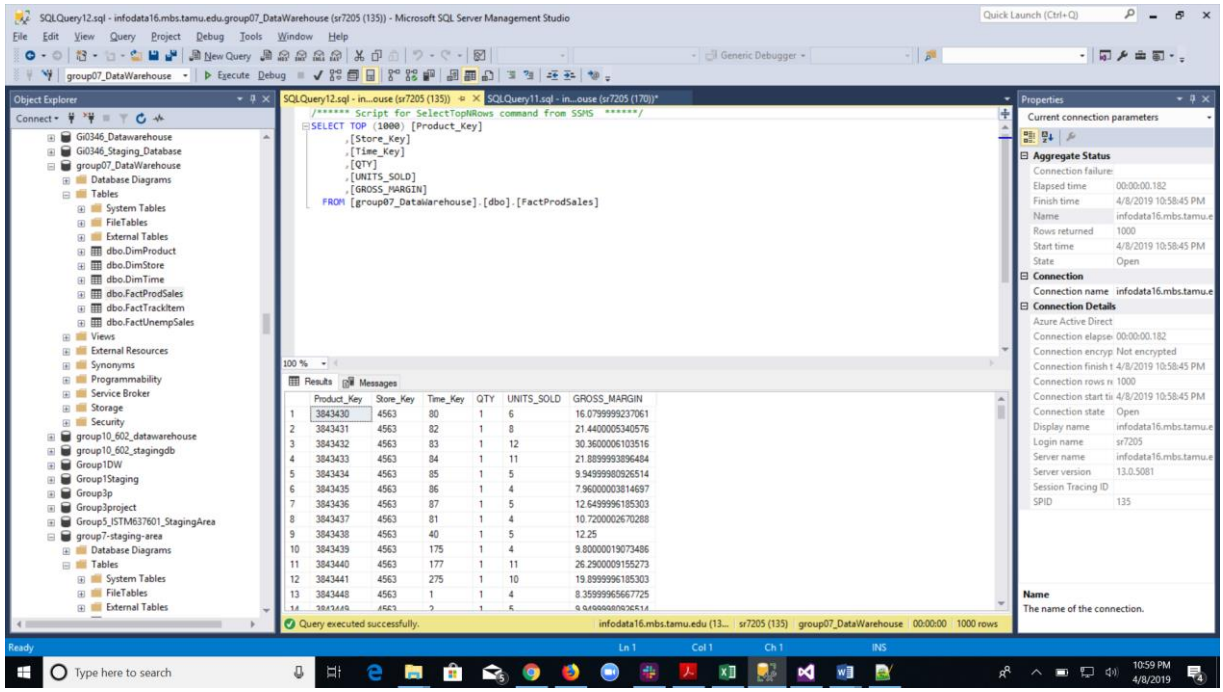
FactTrackItem: In FactTrackItem, two dimension tables help us obtain a drill down on the *storage type* that is used to store the product along with the corresponding *store* that carries it. This allows the user to specifically identify which product was stored in which store.

FactUnempSales: FactUnempSales provides an example of a drill down at different approaches. The lowest being the *weekly* sales of a particular store to yearly sales across different *cities*. This helps the user to identify the impact of unemployment across stores.

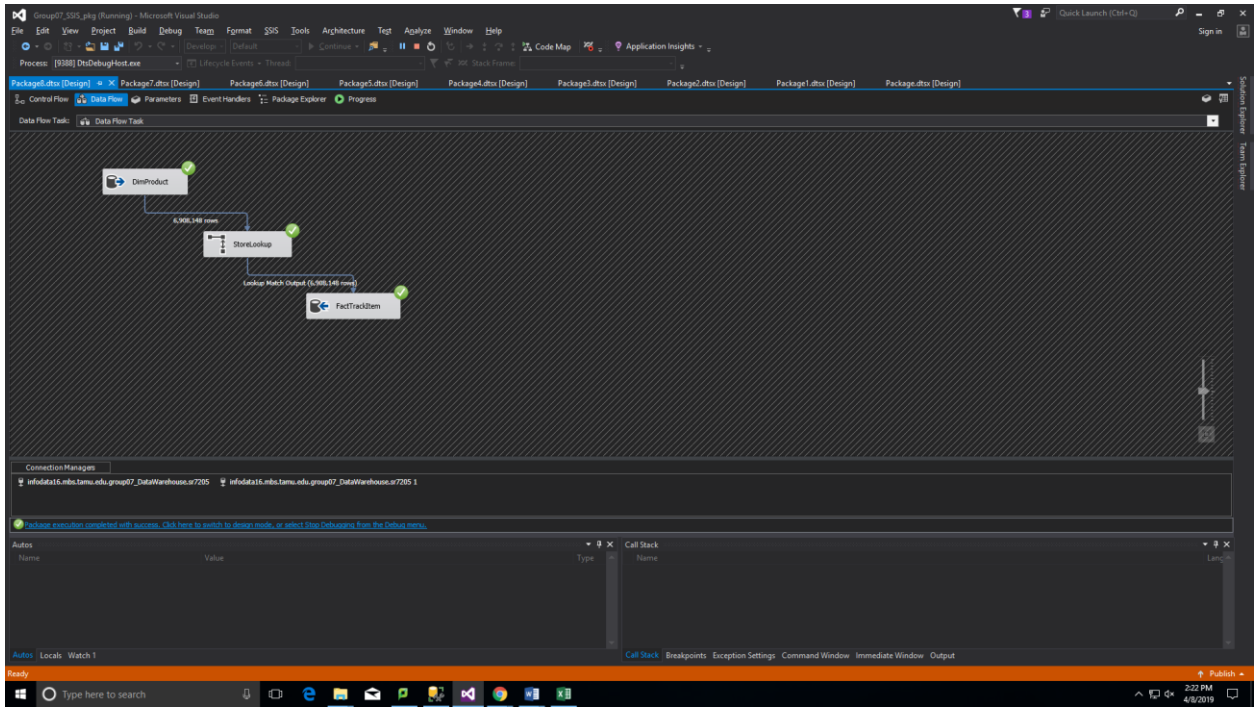
5.10.2.4 Loading data into fact tables



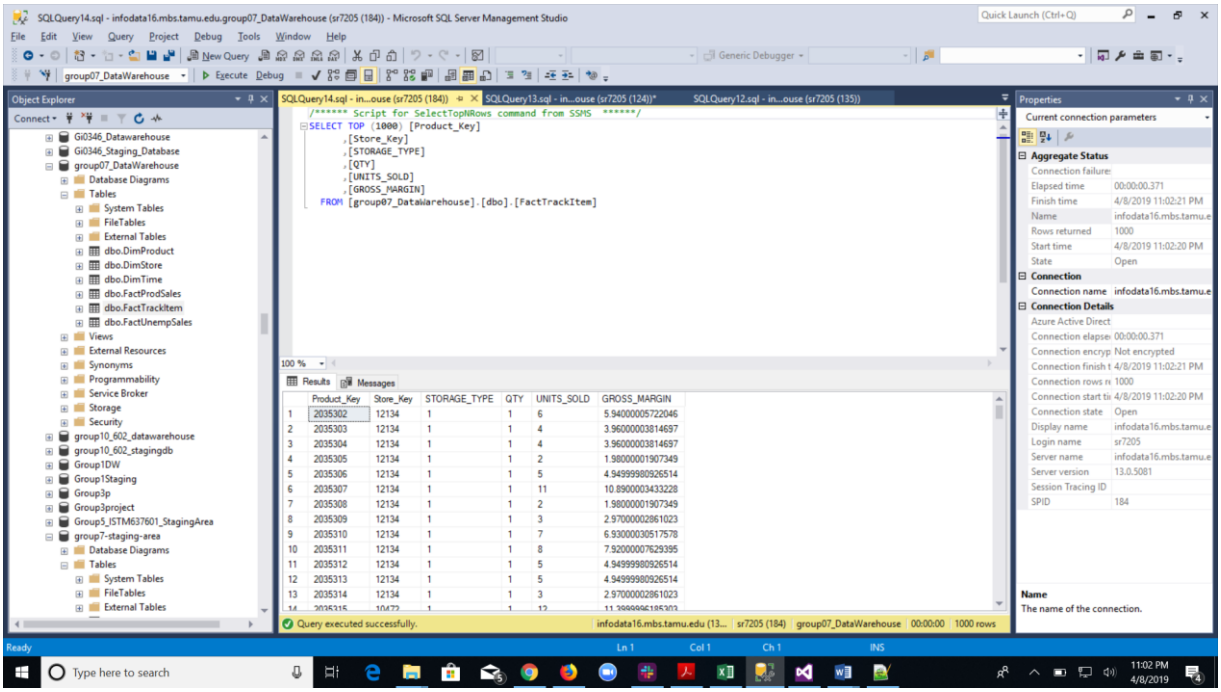
Loading data into FactProdSales



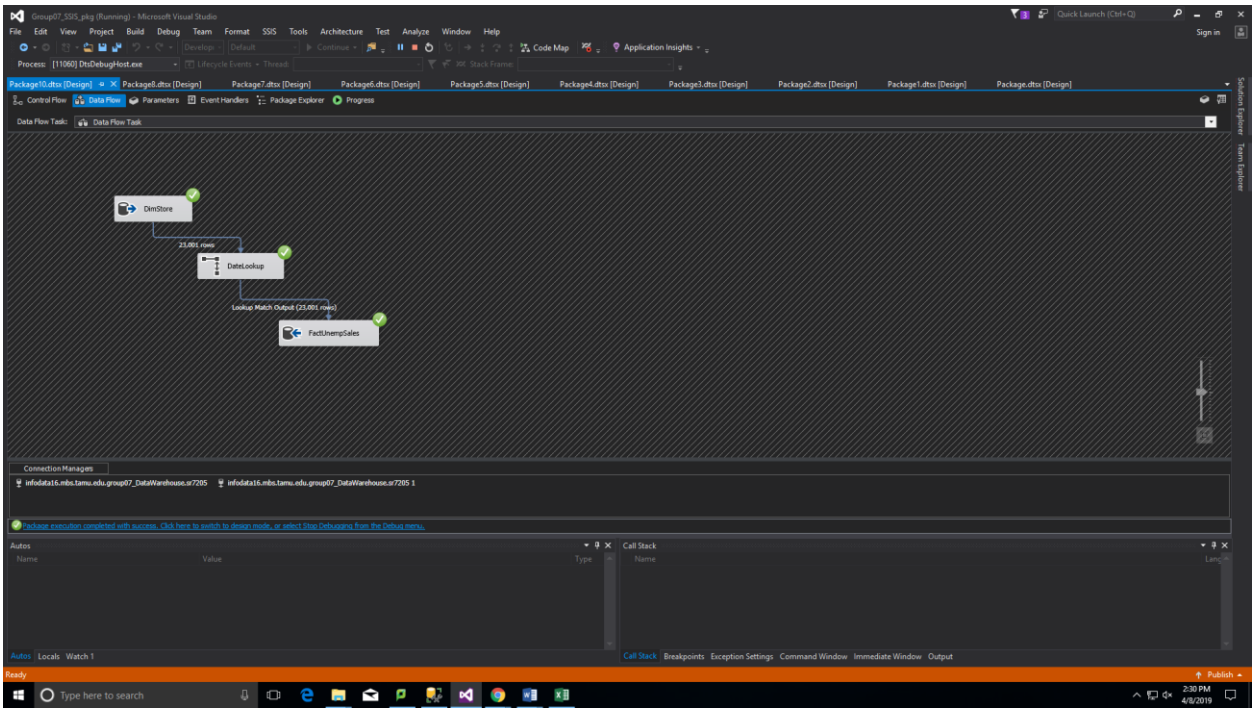
Snapshot of FactProdSales fact table in data warehouse



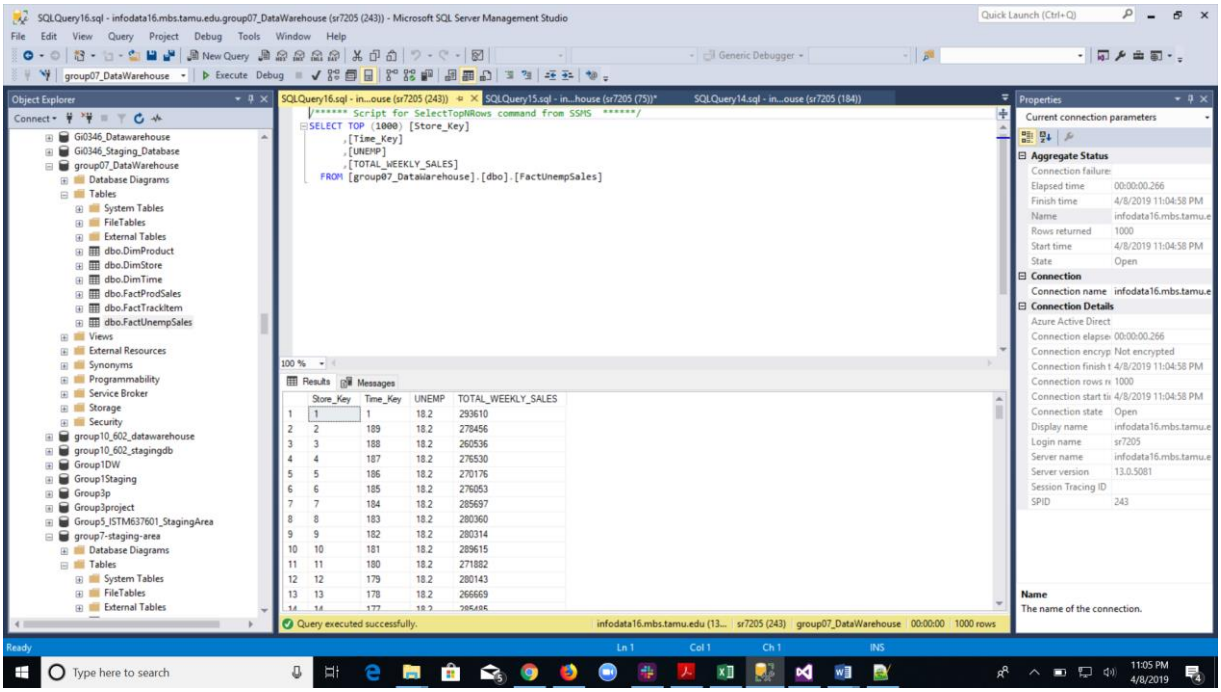
Loading data into FactTrackItem



Snapshot of FactTrackItem fact table in data warehouse



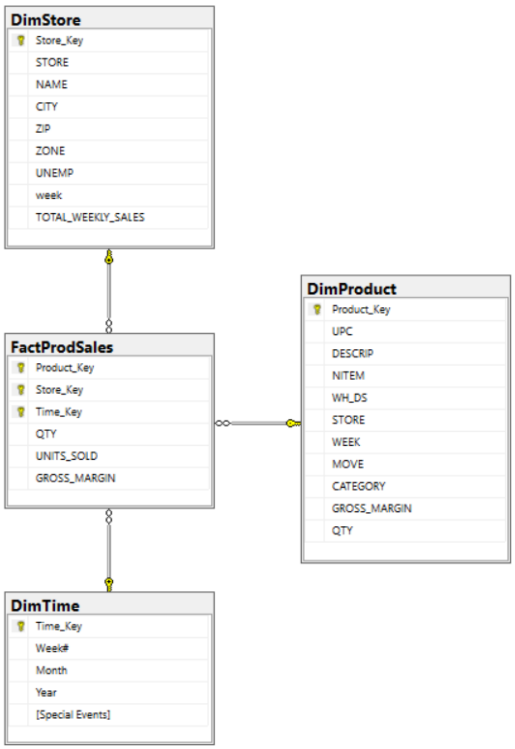
Loading data into FactUnempSales



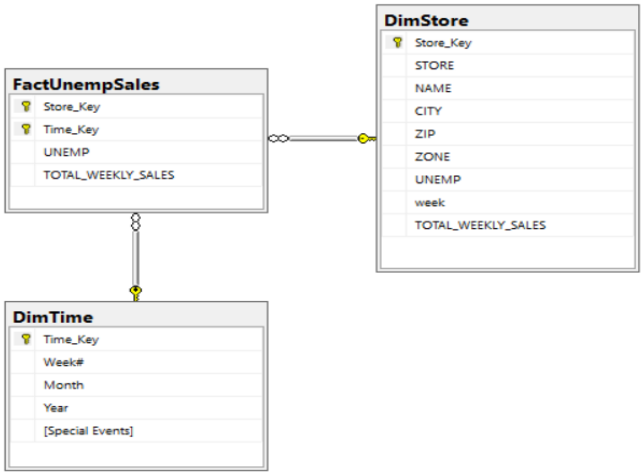
Snapshot of FactUnempSales fact table in data warehouse

5.10.3 Table structures and relationships

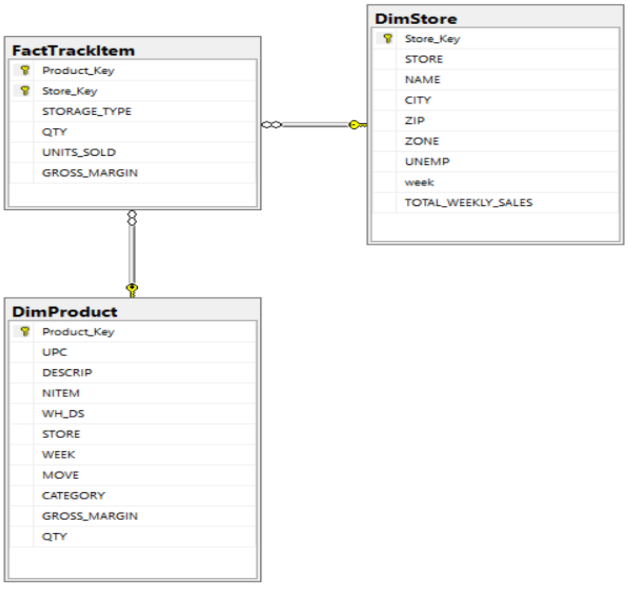
FactProdSales



FactUnempSales



FactTrackItem



5.10.4 SQL statements to create staging area and data warehouse

DimTime

```
CREATE TABLE dbo.DimTime
(
[Time_Key] INT IDENTITY(1,1) PRIMARY KEY NOT NULL,
[Start_Year] datetime NULL,
[End_Year] datetime NULL,
[Event] nvarchar(255) NULL,
[Week_Number] float NULL
) ;
```

DimProduct

```
CREATE TABLE dbo.DimProduct
(
[Product_Key] INT IDENTITY(1,1) PRIMARY KEY NOT NULL,
   [UPC] [numeric](18, 0) NULL,
   [DESCRIP] [nvarchar](255) NULL,
   [NITEM] [numeric](18, 0) NULL,
   [WH_DS] [int] NULL,
   [STORE] [numeric](18, 0) NULL,
   [WEEK] [numeric](18, 0) NULL,
   [MOVE] [numeric](18, 0) NULL,
   [CATEGORY] [nvarchar](255) NULL,
   [GROSS_MARGIN] [float] NULL,
   [QTY] [numeric](18, 0) NULL
) ;
```

DimStore

```
CREATE TABLE dbo.DimStore
(
```

```
    [Store_Key] INT IDENTITY(1,1) PRIMARY KEY NOT NULL,
    [STORE] [numeric](18, 0) NULL,
    [TOTAL_SALES] [float] NULL,
    [NAME] [nvarchar](255) NULL,
    [CITY] [nvarchar](255) NULL,
    [ZIP] [numeric](18, 0) NULL,
    [ZONE] [numeric](18, 0) NULL,
    [UNEMP] [float] NULL
);
```

FactProdSales

```
CREATE TABLE [dbo].[FactProdSales](
    [Product_Key] [int] NULL,
    [Store_Key] [int] NULL,
    [Time_Key] [int] NULL,
    [QTY] [numeric](18, 0) NULL,
    [UNITS_SOLD] [numeric](18, 0) NULL,
    [GROSS_MARGIN] [float] NULL,
    CONSTRAINT FactProdSales_PK PRIMARY KEY
(
[PRODUCT_KEY],[STORE_KEY],[TIME_KEY]
)
);
```

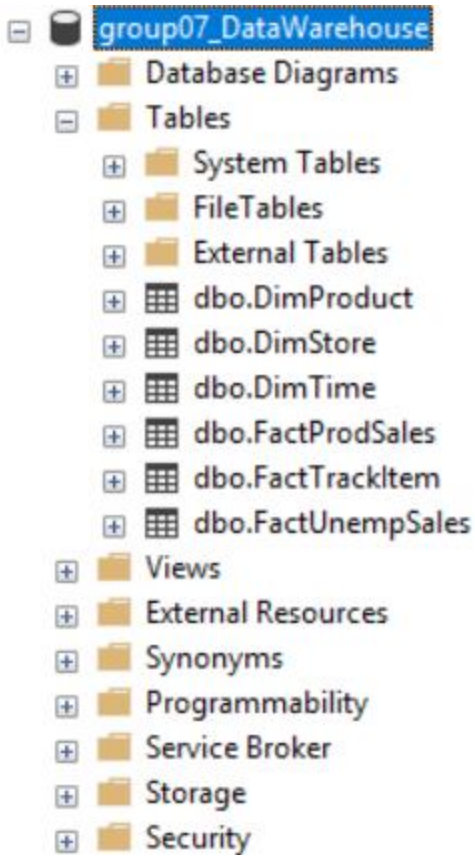
FactTrackItem

```
CREATE TABLE [dbo].[FactTrackItem](
    [Product_Key] [int] NULL,
    [Store_Key] [int] NULL,
    [STORAGE_TYPE] [int] NULL,
    [QTY] [numeric](18, 0) NULL,
```

```
    [UNITS_SOLD] [numeric](18, 0) NULL,
    [GROSS_MARGIN] [float] NULL
    CONSTRAINT FactTrackItem_PK PRIMARY KEY
(
[PRODUCT_KEY],[STORE_KEY]
)
);
```

FactUnempSales

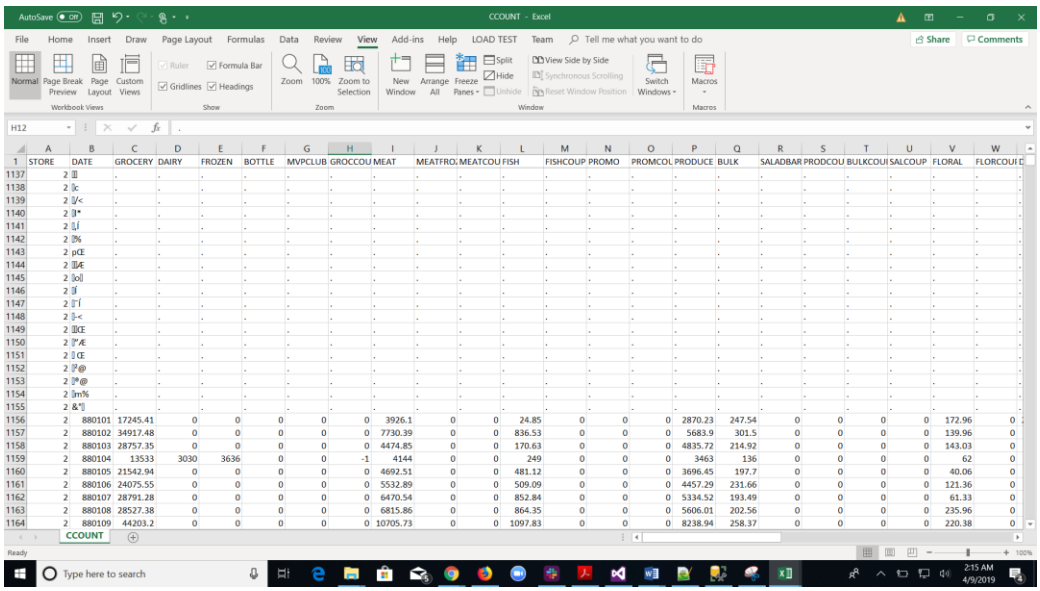
```
CREATE TABLE [dbo].[FactUnempSales](
    [Store_Key] [int] NULL,
    [Time_Key] [int] NULL,
    [UNEMP] [float] NULL,
    [TOTAL_WEEKLY_SALES] [numeric](18, 0) NULL
    CONSTRAINT FactUnempSales_PK PRIMARY KEY
(
[STORE_KEY],[TIME_KEY]
)
);
```



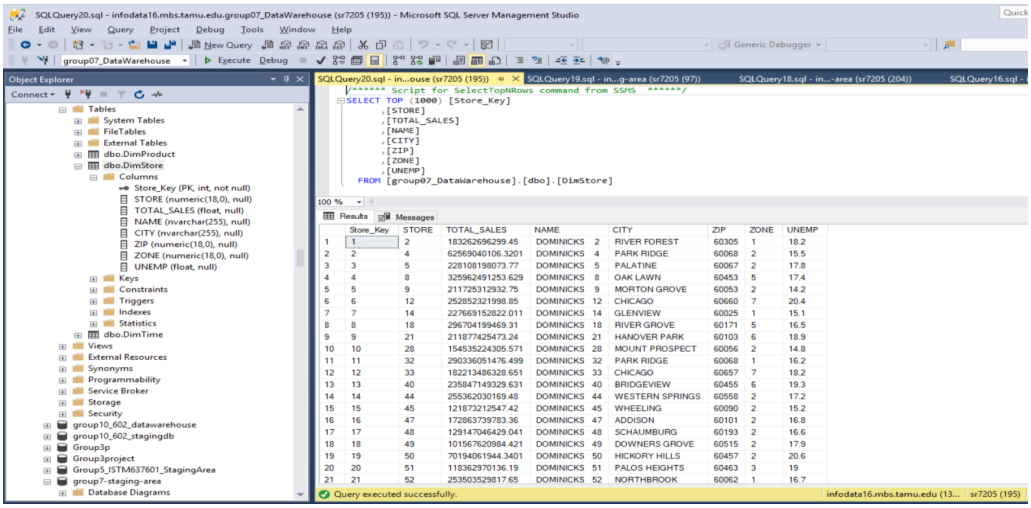
Snapshot of DataWarehouse tables existing

5.10.5 Snapshots of before-after table contents

- **CCount Files**

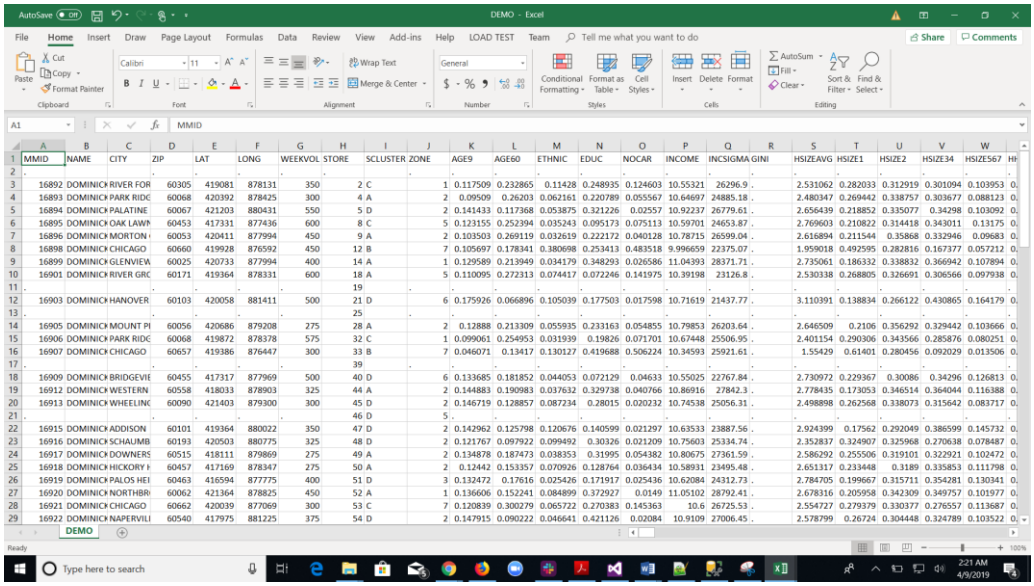


CCOUNT before ETL Run

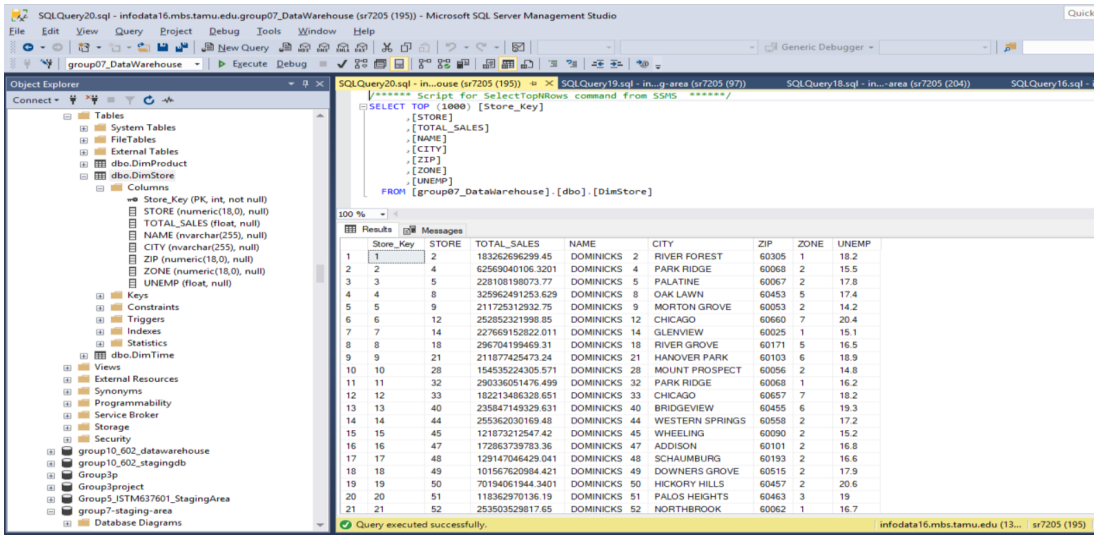


CCOUNT data transformed and used in DimStore

- **Demo Files**

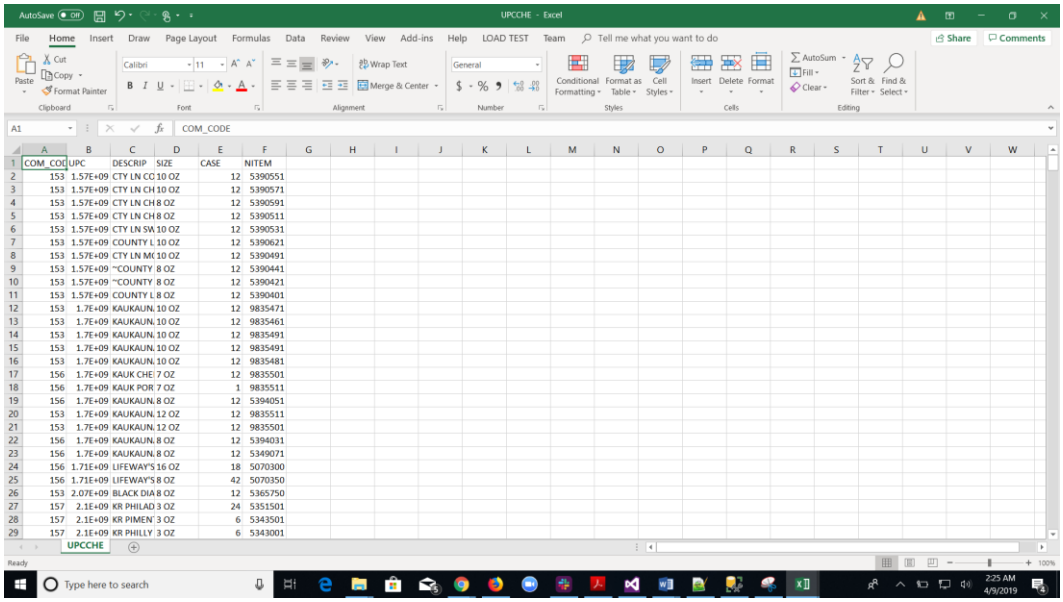


DEMO before ETL runs

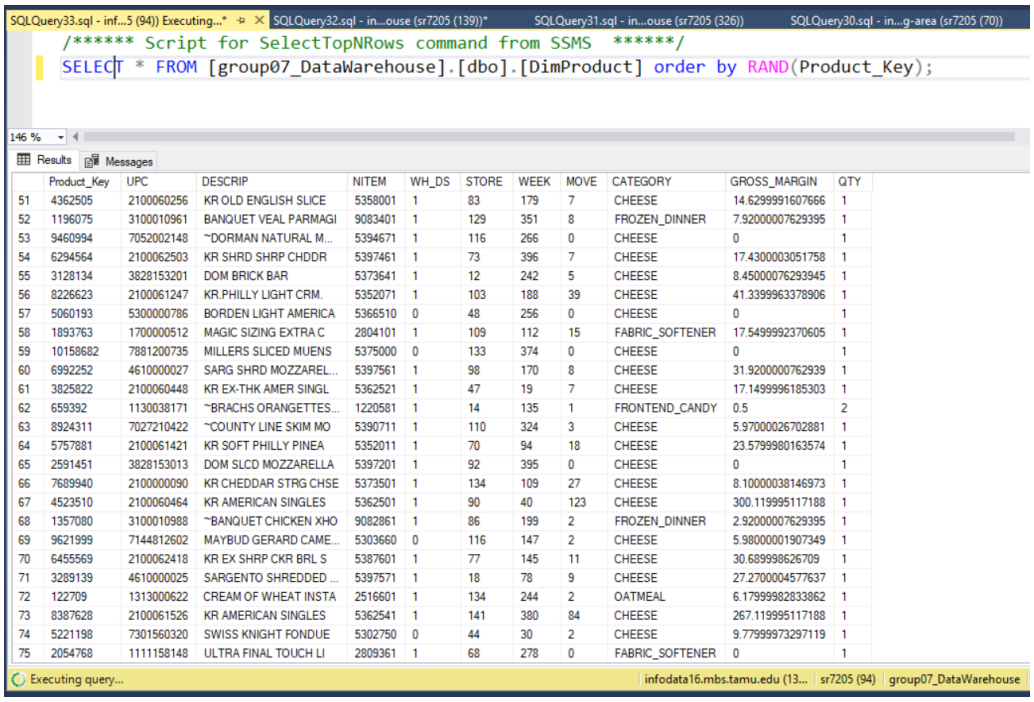


DEMO data transformed and used in DimStore

- **UPC Files**



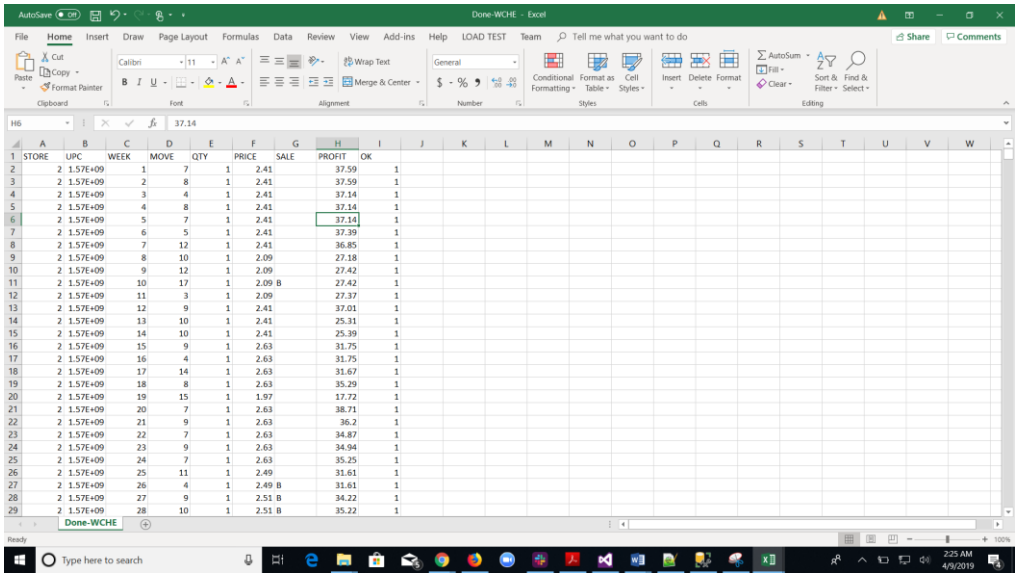
UPC<productacronym> before ETL runs



	Product_Key	UPC	DESCRIP	NITEM	WH_DS	STORE	WEEK	MOVE	CATEGORY	GROSS_MARGIN	QTY
51	4362505	2100060256	KR OLD ENGLISH SLICE	5358001	1	83	179	7	CHEESE	14.6299991607666	1
52	1196075	3100010961	BANQUET VEAL PARMAGI	9083401	1	129	351	8	FROZEN_DINNER	7.92000007629395	1
53	9460994	7052002148	~DORMAN NATURAL M...	5394671	1	116	266	0	CHEESE	0	1
54	6294564	2100062503	KR SHRD SHRP CHDDR	5397461	1	73	396	7	CHEESE	17.4300003051758	1
55	3128134	3828153201	DOM BRICK BAR	5373641	1	12	242	5	CHEESE	8.45000076293945	1
56	8226623	2100061247	KR.PHILLY LIGHT CRM.	5352071	1	103	188	39	CHEESE	41.3399963378906	1
57	5060193	5300000786	BORDEN LIGHT AMERICA	5366510	0	48	256	0	CHEESE	0	1
58	1893763	1700000512	MAGIC SIZING EXTRA C	2804101	1	109	112	15	FABRIC_SOFTENER	17.5499992370605	1
59	10158682	7881200735	MILLERS SLICED MUENS	5375000	0	133	374	0	CHEESE	0	1
60	6992252	4610000027	SARG SHRD MOZZAREL...	5397561	1	98	170	8	CHEESE	31.9200000762939	1
61	3825822	2100060448	KR EX-THK AMER SINGL	5362521	1	47	19	7	CHEESE	17.1499996185303	1
62	659392	1130038171	~BRACHS ORANGETTES...	1220581	1	14	135	1	FRONTEND_CANDY	0.5	2
63	8924311	7027210422	~COUNTY LINE SKIM MO	5390711	1	110	324	3	CHEESE	5.97000026702881	1
64	5757881	2100061421	KR SOFT PHILLY PINEA	5352011	1	70	94	18	CHEESE	23.5799980163574	1
65	2591451	3828153013	DOM SLCD MOZZARELLA	5397201	1	92	395	0	CHEESE	0	1
66	7689940	2100000090	KR CHEDDAR STRG CHSE	5373501	1	134	109	27	CHEESE	8.10000038146973	1
67	4523510	2100060464	KR AMERICAN SINGLES	5362501	1	90	40	123	CHEESE	300.119995117188	1
68	1357080	3100010988	~BANQUET CHICKEN XHO	9082861	1	86	199	2	FROZEN_DINNER	2.92000007629395	1
69	9621999	7144812602	MAYBUD GERARD CAME...	5303660	0	116	147	2	CHEESE	5.98000001907349	1
70	6455569	2100062418	KR EX SHRP CKR BRL S	5387601	1	77	145	11	CHEESE	30.689998626709	1
71	3289139	4610000025	SARGENTO SHREDDED ...	5397571	1	18	78	9	CHEESE	27.2700004577637	1
72	122709	1313000622	CREAM OF WHEAT INSTA	2516601	1	134	244	2	OATMEAL	6.17999982833862	1
73	8387628	2100061526	KR AMERICAN SINGLES	5362541	1	141	380	84	CHEESE	267.119995117188	1
74	5221198	7301560320	SWISS KNIGHT FONDUE	5302750	0	44	30	2	CHEESE	9.77999973297119	1
75	2054768	1111158148	ULTRA FINAL TOUCH LI	2809361	1	68	278	0	FABRIC_SOFTENER	0	1

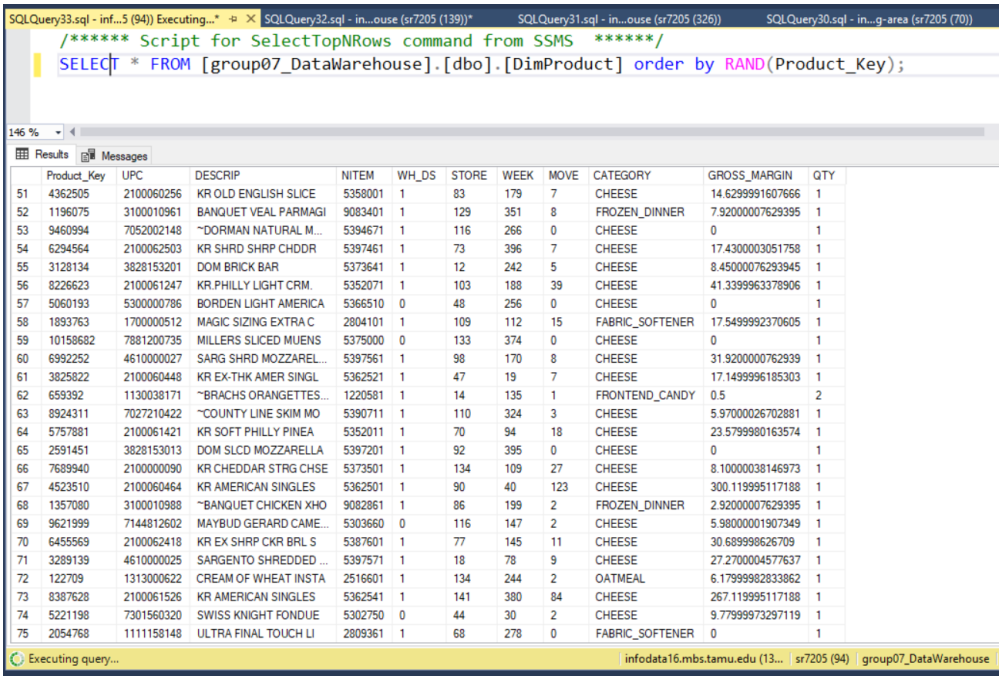
UPC<productacronym> data transformed and used in DimProduct

- **Product Files**



	A	B	C	D	E	F	G	H	I
1	STORE	UPC	WEEK	MOVE	QTY	PRICE	SALE	PROFIT	OK
2	2	1.57E+09	1	7	1	2.41		37.59	1
3	2	1.57E+09	2	8	1	2.41		37.59	1
4	2	1.57E+09	3	4	1	2.41		37.14	1
5	2	1.57E+09	4	8	1	2.41		37.14	1
6	2	1.57E+09	5	7	1	2.41		37.14	1
7	2	1.57E+09	6	5	1	2.41		37.39	1
8	2	1.57E+09	7	12	1	2.41		36.85	1
9	2	1.57E+09	8	10	1	2.09		27.18	1
10	2	1.57E+09	9	12	1	2.09		27.42	1
11	2	1.57E+09	10	17	1	2.09	B	27.42	1
12	2	1.57E+09	11	3	1	2.09		27.37	1
13	2	1.57E+09	12	9	1	2.41		37.01	1
14	2	1.57E+09	13	10	1	2.41		25.31	1
15	2	1.57E+09	14	10	1	2.41		25.39	1
16	2	1.57E+09	15	9	1	2.63		31.75	1
17	2	1.57E+09	16	4	1	2.63		31.75	1
18	2	1.57E+09	17	14	1	2.63		31.67	1
19	2	1.57E+09	18	8	1	2.63		35.29	1
20	2	1.57E+09	19	15	1	1.97		17.72	1
21	2	1.57E+09	20	7	1	2.63		38.71	1
22	2	1.57E+09	21	9	1	2.63		36.2	1
23	2	1.57E+09	22	7	1	2.63		34.87	1
24	2	1.57E+09	23	9	1	2.63		34.94	1
25	2	1.57E+09	24	7	1	2.63		35.25	1
26	2	1.57E+09	25	11	1	2.49		31.61	1
27	2	1.57E+09	26	4	1	2.49	B	31.61	1
28	2	1.57E+09	27	9	1	2.51	B	34.22	1
29	2	1.57E+09	28	10	1	2.51	B	35.22	1

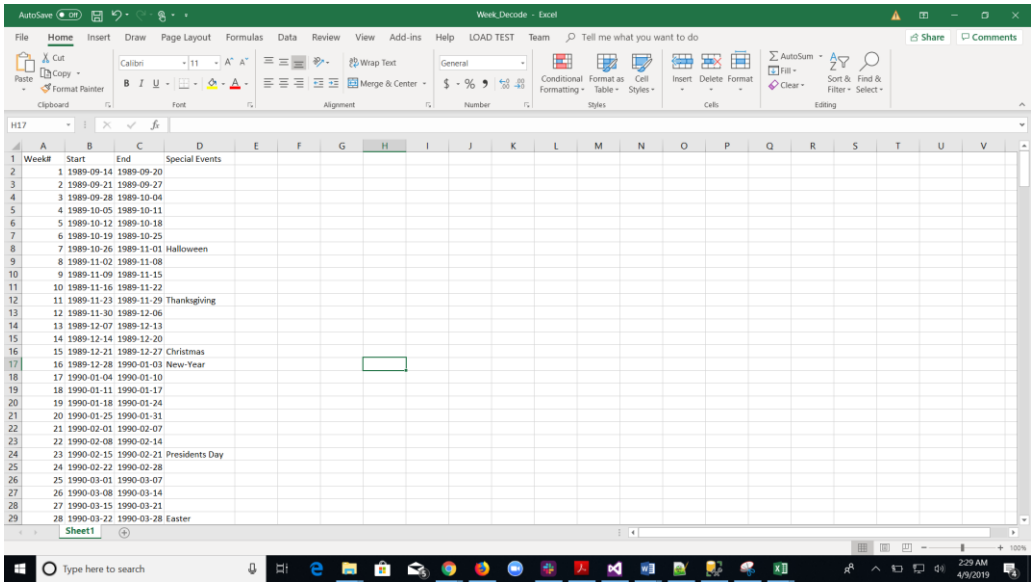
W<productacronym> data transformed and used in DimProduct



	Product_Key	UPC	DESCRIP	NITEM	WH_DS	STORE	WEEK	MOVE	CATEGORY	GROSS_MARGIN	QTY
51	4362505	2100060256	KR OLD ENGLISH SLICE	5358001	1	83	179	7	CHEESE	14.6299991607666	1
52	1196075	3100010961	BANQUET VEAL PARMAGI	9083401	1	129	351	8	FROZEN_DINNER	7.92000007629395	1
53	9460994	7052002148	~DORMAN NATURAL M...	5394671	1	116	266	0	CHEESE	0	1
54	6294564	2100062503	KR SHRD SHRP CHDDR	5397461	1	73	396	7	CHEESE	17.4300003051758	1
55	3128134	3828153201	DOM BRICK BAR	5373641	1	12	242	5	CHEESE	8.45000076293945	1
56	8226623	2100061247	KR PHILLY LIGHT CRM.	5352071	1	103	188	39	CHEESE	41.3399963378906	1
57	5060193	5300000786	BORDEN LIGHT AMERICA	5366510	0	48	256	0	CHEESE	0	1
58	1893763	1700000512	MAGIC SIZING EXTRA C	2804101	1	109	112	15	FABRIC_SOFTENER	17.5499992370605	1
59	10158682	7881200735	MILLERS SLICED MUENS	5375000	0	133	374	0	CHEESE	0	1
60	6992252	4610000027	SARG SHRD MOZZAREL...	5397561	1	98	170	8	CHEESE	31.9200000762939	1
61	3825822	2100060448	KR EX-THK AMER SINGL	5362521	1	47	19	7	CHEESE	17.1499996185303	1
62	659392	1130038171	~BRACHS ORANGETTES...	1220581	1	14	135	1	FRONTEND_CANDY	0.5	2
63	8924311	7027210422	~COUNTY LINE SKIM MO	5390711	1	110	324	3	CHEESE	5.97000026702881	1
64	5757881	2100061421	KR SOFT PHILLY PINEA	5352011	1	70	94	18	CHEESE	23.5799980163574	1
65	2591451	3828153013	DOM SLCD MOZZARELLA	5397201	1	92	395	0	CHEESE	0	1
66	7689940	2100000090	KR CHEDDAR STRG CHSE	5373501	1	134	109	27	CHEESE	8.10000038146973	1
67	4523510	2100060464	KR AMERICAN SINGLES	5362501	1	90	40	123	CHEESE	300.119995117188	1
68	1357080	3100010988	~BANQUET CHICKEN XHO	9082861	1	86	199	2	FROZEN_DINNER	2.92000007629395	1
69	9621999	7144812602	MAYBUD GERARD CAME...	5303660	0	116	147	2	CHEESE	5.98000001907349	1
70	6455569	2100062418	KR EX SHRP CKR BRL S	5387601	1	77	145	11	CHEESE	30.689998626709	1
71	3289139	4610000025	SARGENTO SHREDDED ...	5397571	1	18	78	9	CHEESE	27.2700004577637	1
72	122709	1313000622	CREAM OF WHEAT INSTA	2516601	1	134	244	2	OATMEAL	6.17999982833862	1
73	8387628	2100061526	KR AMERICAN SINGLES	5362541	1	141	380	84	CHEESE	267.119995117188	1
74	5221198	7301560320	SWISS KNIGHT FONDUE	5302750	0	44	30	2	CHEESE	9.77999973297119	1
75	2054768	1111158148	ULTRA FINAL TOUCH LI	2809361	1	68	278	0	FABRIC_SOFTENER	0	1

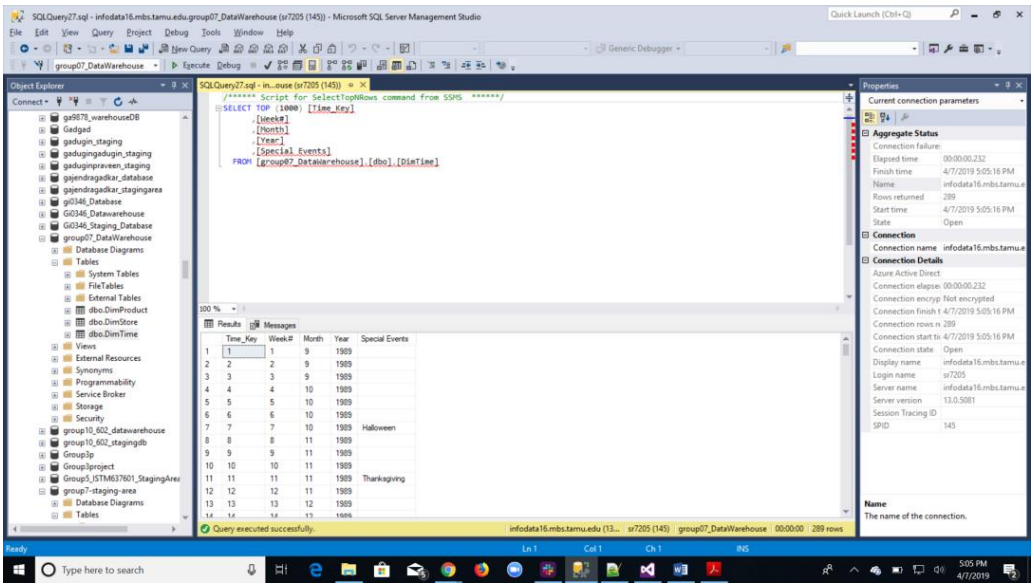
W<productacronym> before ETL runs

- **Week Decode Files**



	A	B	C	D
1	Week#	Start	End	Special Events
2	1	1989-09-14	1989-09-20	
3	2	1989-09-21	1989-09-27	
4	3	1989-09-28	1989-10-04	
5	4	1989-10-05	1989-10-11	
6	5	1989-10-12	1989-10-18	
7	6	1989-10-19	1989-10-25	
8	7	1989-10-26	1989-11-01	Halloween
9	8	1989-11-02	1989-11-08	
10	9	1989-11-09	1989-11-15	
11	10	1989-11-16	1989-11-22	
12	11	1989-11-23	1989-11-29	Thanksgiving
13	12	1989-11-30	1989-12-06	
14	13	1989-12-07	1989-12-13	
15	14	1989-12-14	1989-12-20	
16	15	1989-12-21	1989-12-27	Christmas
17	16	1989-12-28	1990-01-03	New-Year
18	17	1990-01-04	1990-01-10	
19	18	1990-01-11	1990-01-17	
20	19	1990-01-18	1990-01-24	
21	20	1990-01-25	1990-01-31	
22	21	1990-02-01	1990-02-07	
23	22	1990-02-08	1990-02-14	
24	23	1990-02-15	1990-02-21	Presidents Day
25	24	1990-02-22	1990-02-28	
26	25	1990-03-01	1990-03-07	
27	26	1990-03-08	1990-03-14	
28	27	1990-03-15	1990-03-21	
29	28	1990-03-22	1990-03-28	Easter

Week_Decode before ETL runs



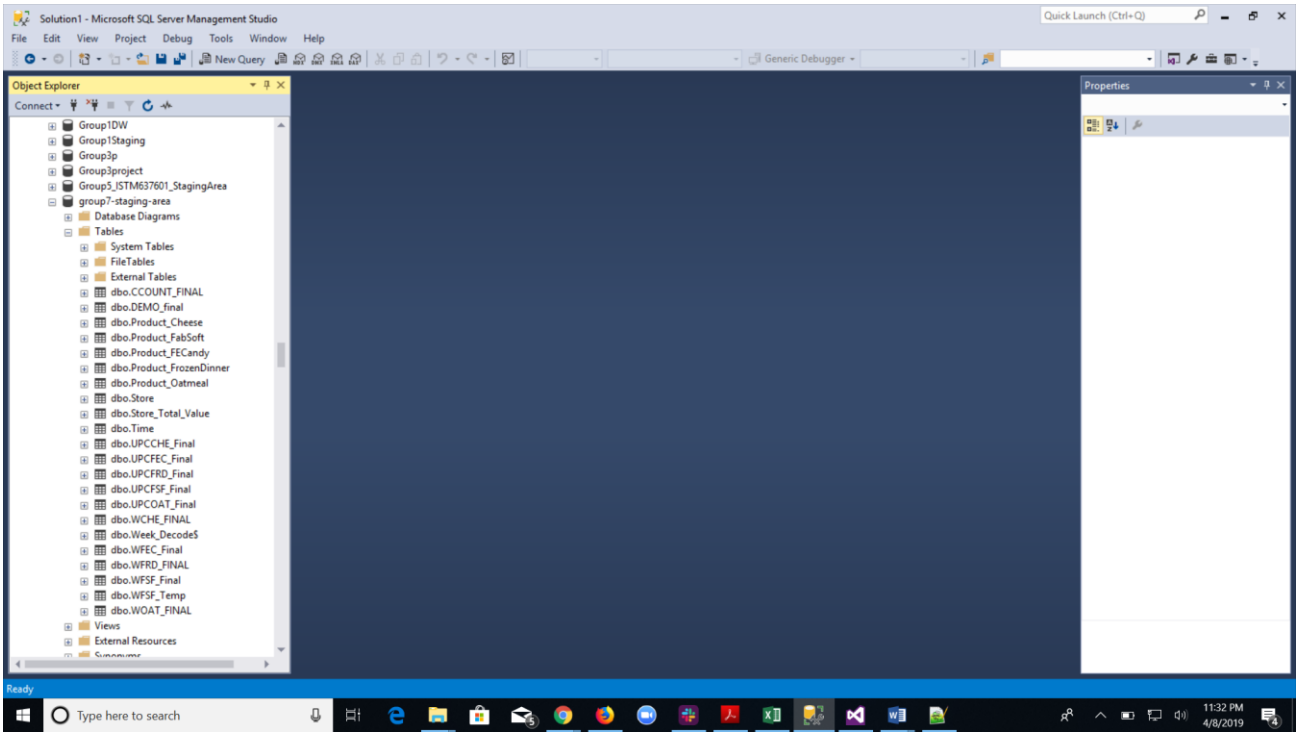
Week_Decode data transformed and used in DimTime

5.10.6 Removal of temporary files from Staging

The temp files have been removed using the below query and the staging area only includes the transformed and clean files ready to be moved to the data warehouse.

SQL Query:

```
DROP TABLE CCOUNT_FINAL,DEMO$,UPCCHE,UPCFEC,
UPCFRD,UPCFSF,UPCOAT,WCHE,WCHE_FINAL_Store_1_to_25,
WCHE_FINAL_Store_101_to_150,WCHE_FINAL_Store_1_to_67,
WCHE_FINAL_Store_26_to_50,WCHE_FINAL_Store_51_to_75,
WCHE_FINAL_Store_68_to_175,WCHE_FINAL_Store_76_to_100,
WFEC,WFRD,WFSF;
```



5.10.7 Special tasks performed as part of the ETL process

1. DATEPART() was used to derive the date literal of our WEEK_DECODE table. This provided us the ability to split the date into week, month and year
2. RIGHT() was used to extract the last digit of NITEM in the UPC<productacronym> files in order to determine if the product should be drop-shipped or warehoused
3. The Union of UPCCHE and WCHE is Product_Cheese which was split into five separate Product_Cheese_(1-150) files in order to expedite the loading of data into the staging area.

6. BI Reporting (using SSRS, SSAS and Report builder)

6.1 Reporting Plan

"Business Intelligence or BI is the accumulation, analysis, reporting, budgeting and presentation of business data in order to improve visibility of organizational operations and financial status to manage the business better"

Tools used: SSRS, SSAS, Report builder 3.0, SSRS+SSAS

Reporting tool	Question #
SSRS	2
SSAS	3
Report builder 3.0	4,5
SSRS+SSAS	1

6.1.1 Target reports satisfying business questions

Question 1: What is the effect of unemployment on sales in a particular city over time?

Report generated from SSRS on top of SSAS

In order to solve this business question, we used the analysis services of the SQL tools. We used the Unemployment Sales data mart, DimStore and DimTime dimensions. By using an SQL query, we were able to choose the store details, unemployment rate and corresponding sales values. We created a cube and installed it to the server. In order to visualize this tool, we used the SSRS tool on top of SSAS. We choose the SSAS cube through the SSRS tool and populated the graph. We used a line graph to display the effect of unemployment on sales. The report also has a drill-down capability.

Question 2: What is the trend of front-end candy sales during Halloween?

Report generated from SSRS alone

To solve this question, we used the DimProduct table from our warehouse. This contained an attribute "category" which allowed us to analyze front-end candy products. The Halloween event was obtained from the DimTime table which mapped the week values of corresponding Halloween weeks. The units sold from the fact table helps us analyze the trend in sales during the week. We have used a bar chart for this question which shows an increasing trend of sales during the thanksgiving period.

Question 3: What is the trend of cheese sales from the year 1989 to 1993?

Report generated from SSAS alone

We used SSAS to analyze the trend of cheese sales across the years spanning 1989 to 1993. We deployed a cube which consists of FactProdSales table, DimProduct dimension, DimTime dimension and DimStore dimension. The cube was able to group the cheese sales by the year and provide a detailed drill down analysis across several stores.

Question 4: How many Fabric softeners and frozen dinners were drop-shipped vs warehoused? (0 or 1 at the last digit of the item code) according to each individual item?

Report generated from Report Builder 3.0

In order to answer this question, we used the independent data mart FactTrackItem table, DimProduct dimension and DimStore dimension. We used a SQL query to identify the number of units which were drop shipped and warehoused. This information was present in the form of two codes- 0 is drop-shipped and 1 is warehoused. We then represented the data obtained in a report builder. We used a pie chart to show the number of units which were drop shipped and warehoused based on the category it belonged to.

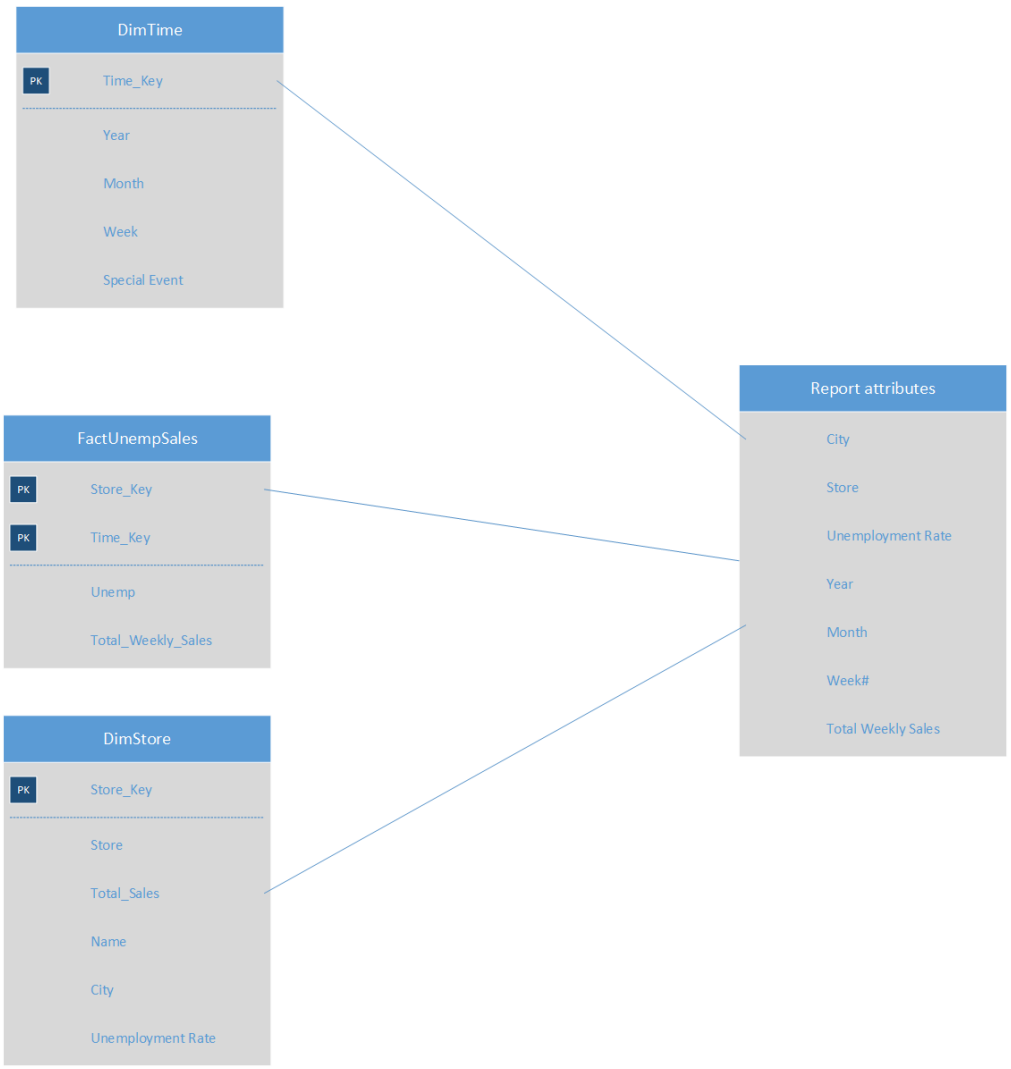
Question 5*:* Find the average gross margin for oatmeal across all stores and determine what stores are performing below average

Report generated from Report Builder 3.0

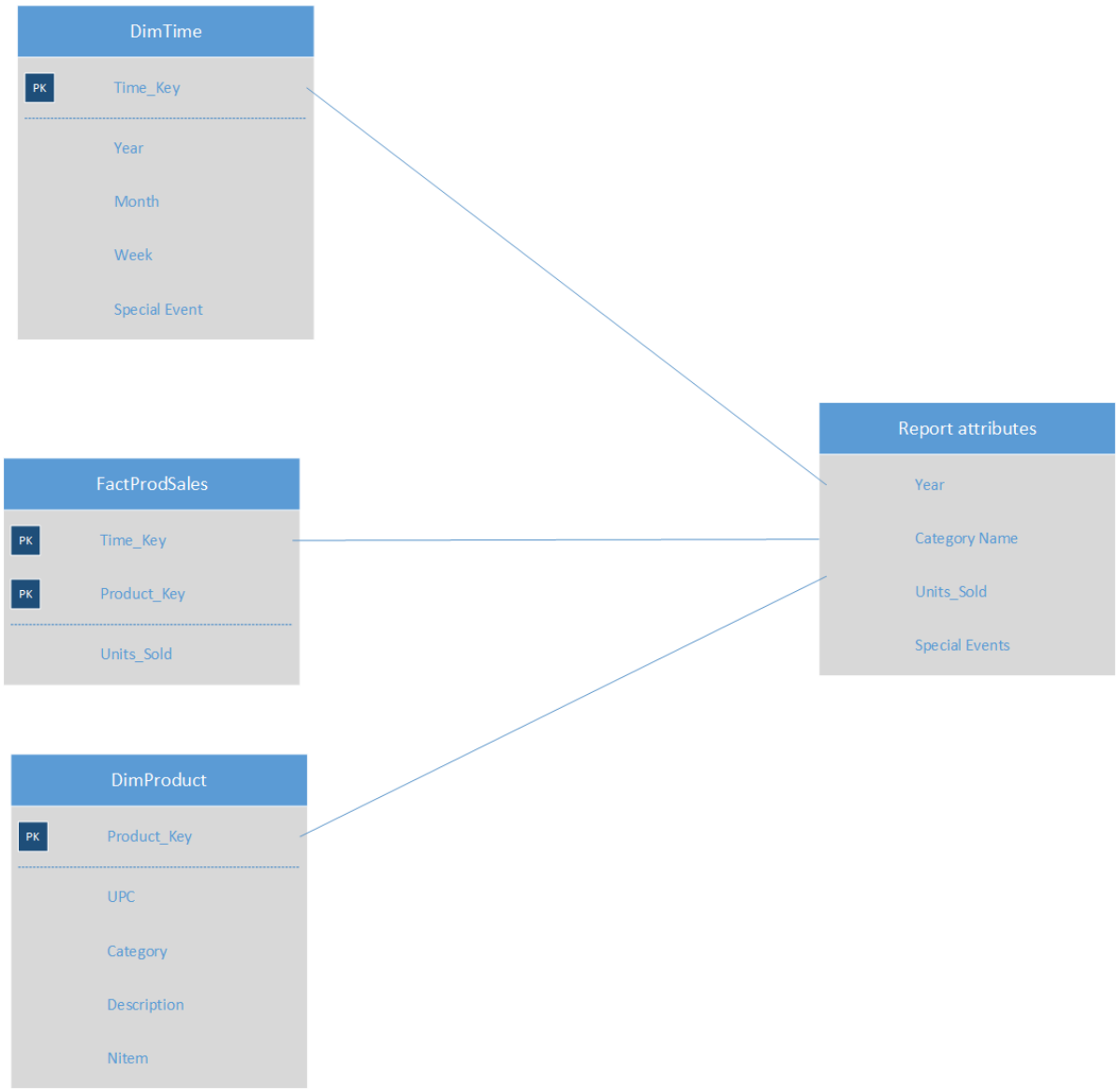
We used FactProdSales data mart consisting of the DimProduct, Dimstore dimensions in order to analyze which products where performing below average. By using an SQL query, we placed a baseline of the average sales across all products and then grouped each store by their respective gross margin. By using Report Builder, we displayed a line graph which showed in detail the performance of Oatmeal in each store when compared to the average across all.

6.2 Mappings from the tables in the data marts to the attributes in the report

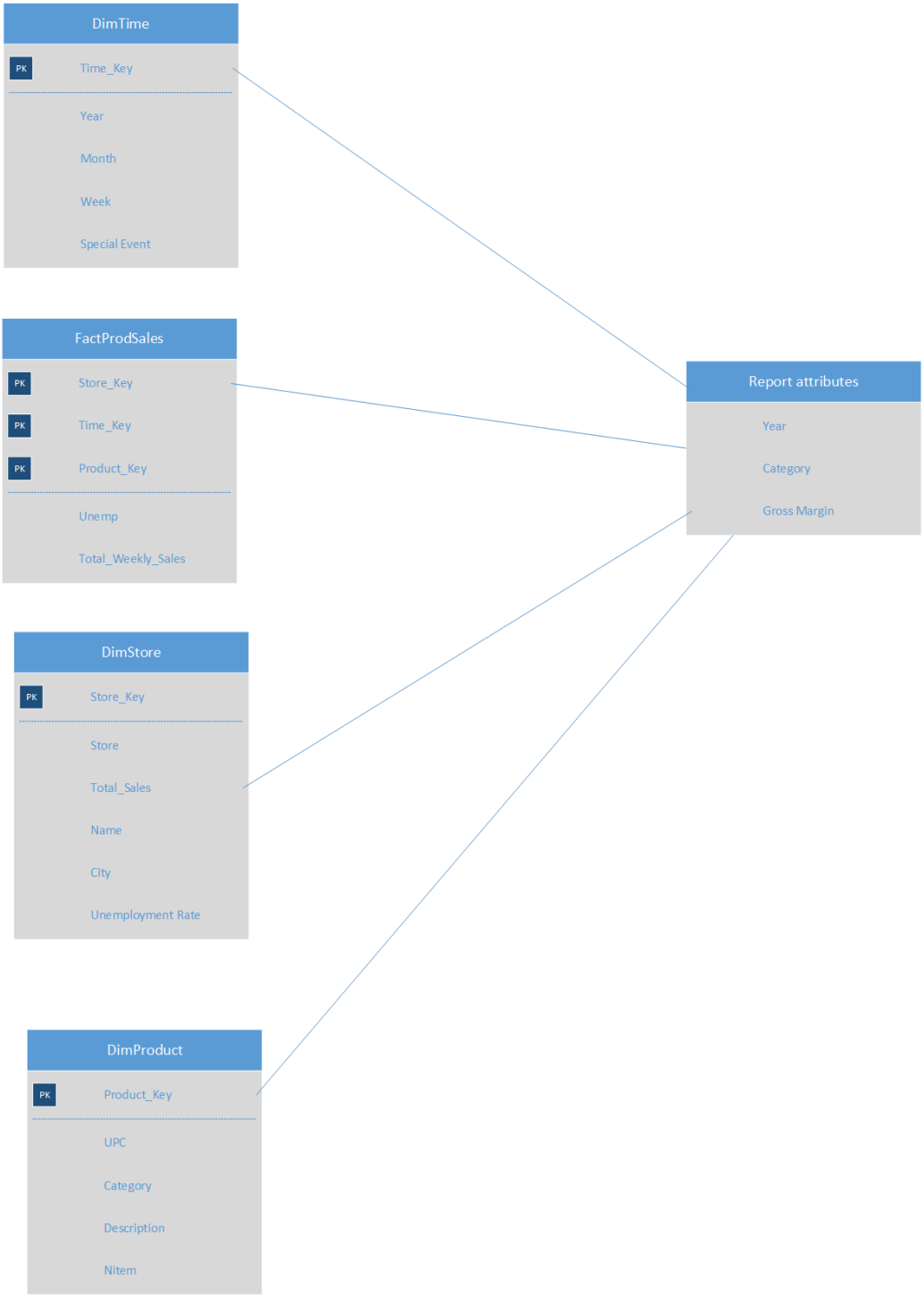
Question 1: What is the effect of unemployment on sales in a particular city over time?



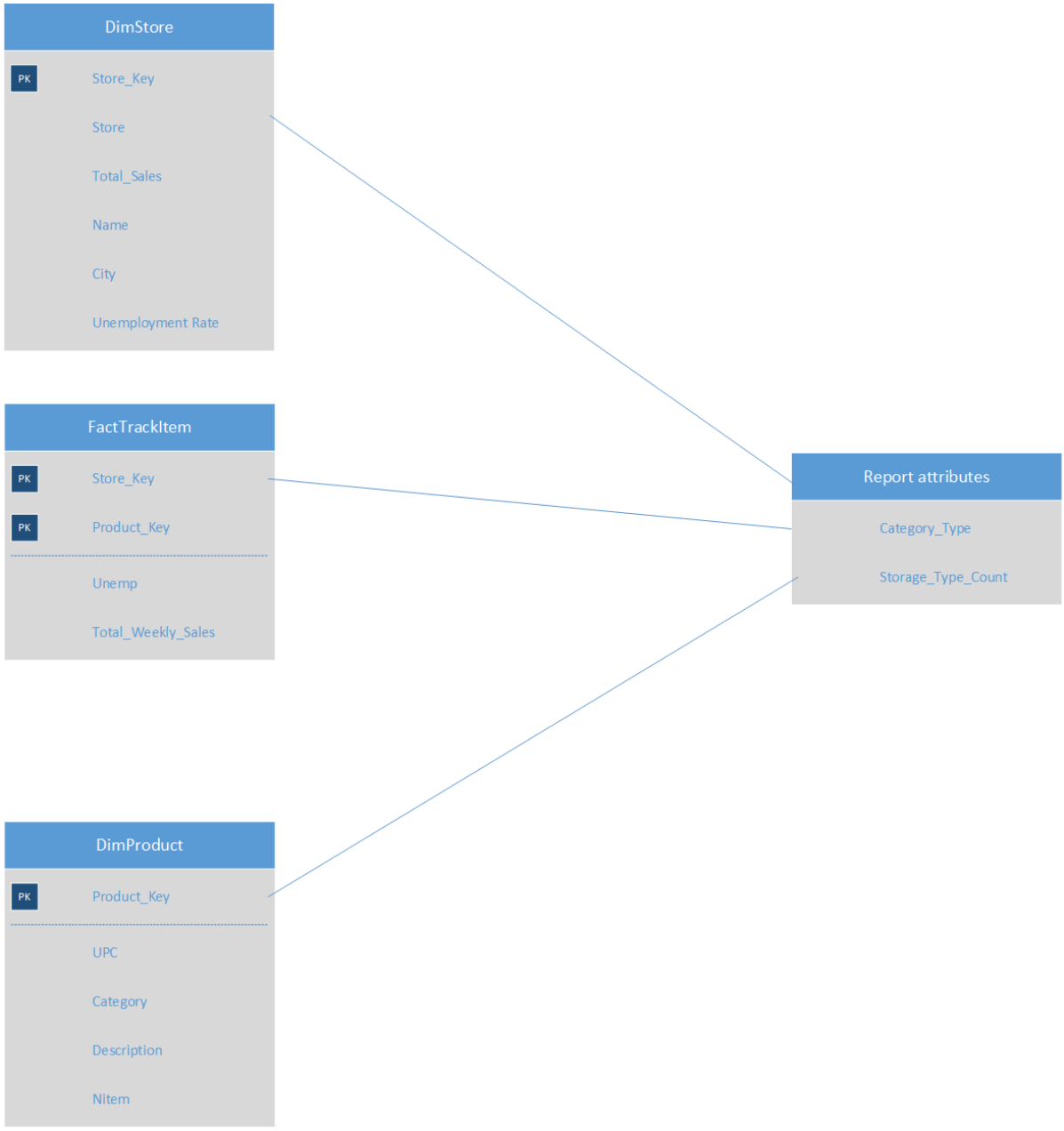
Question 2: What is the trend of front-end candy sales during Halloween?



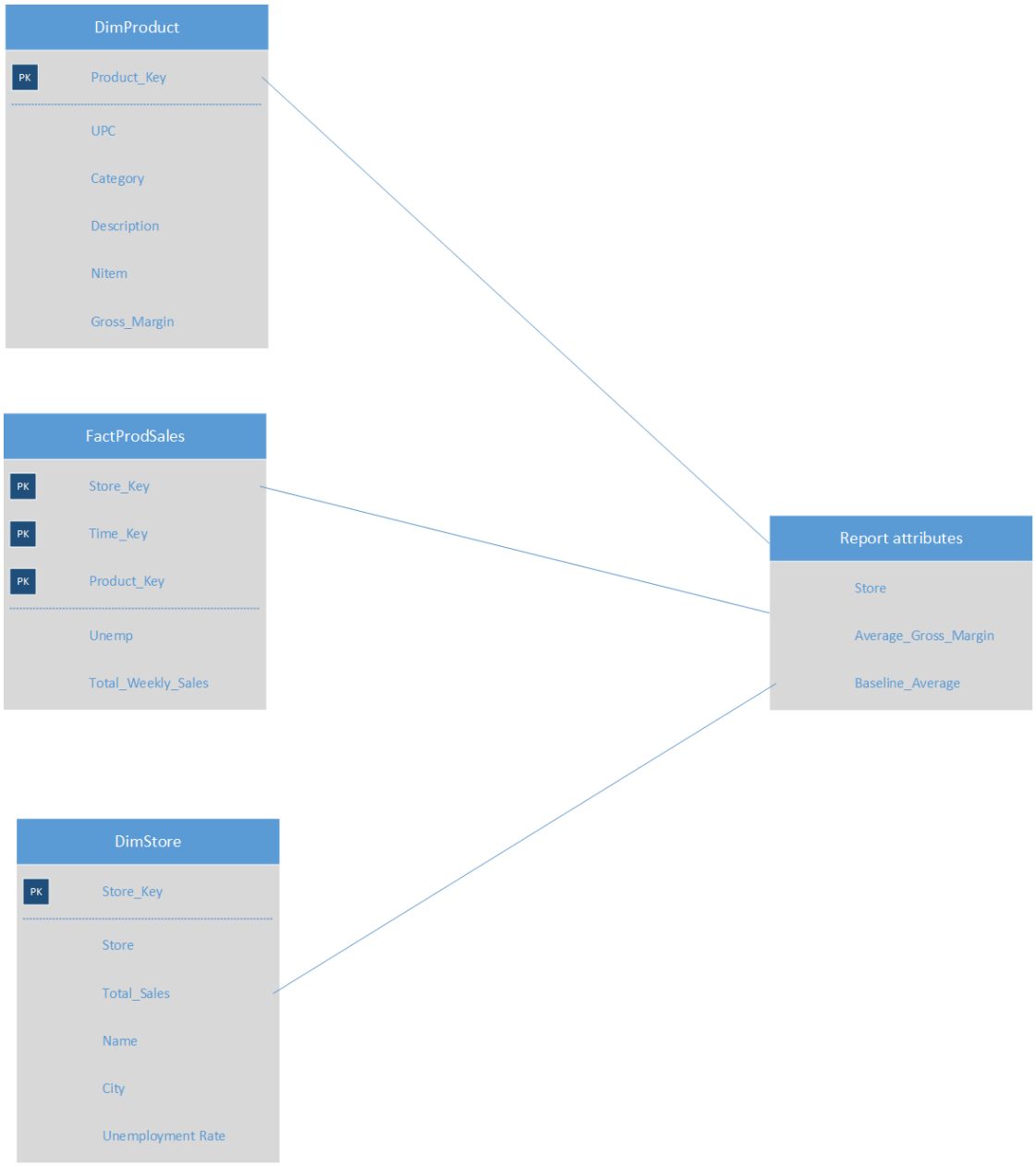
Question 3: What is the trend of cheese sales from the year 1989 to 1993?



Question 4: How many Fabric softeners and frozen dinners were drop-shipped vs warehoused? (0 or 1 at the last digit of the item code) according to each individual item?



Question 5*:* Find the average gross margin for oatmeal across all stores and determine what stores are performing below average

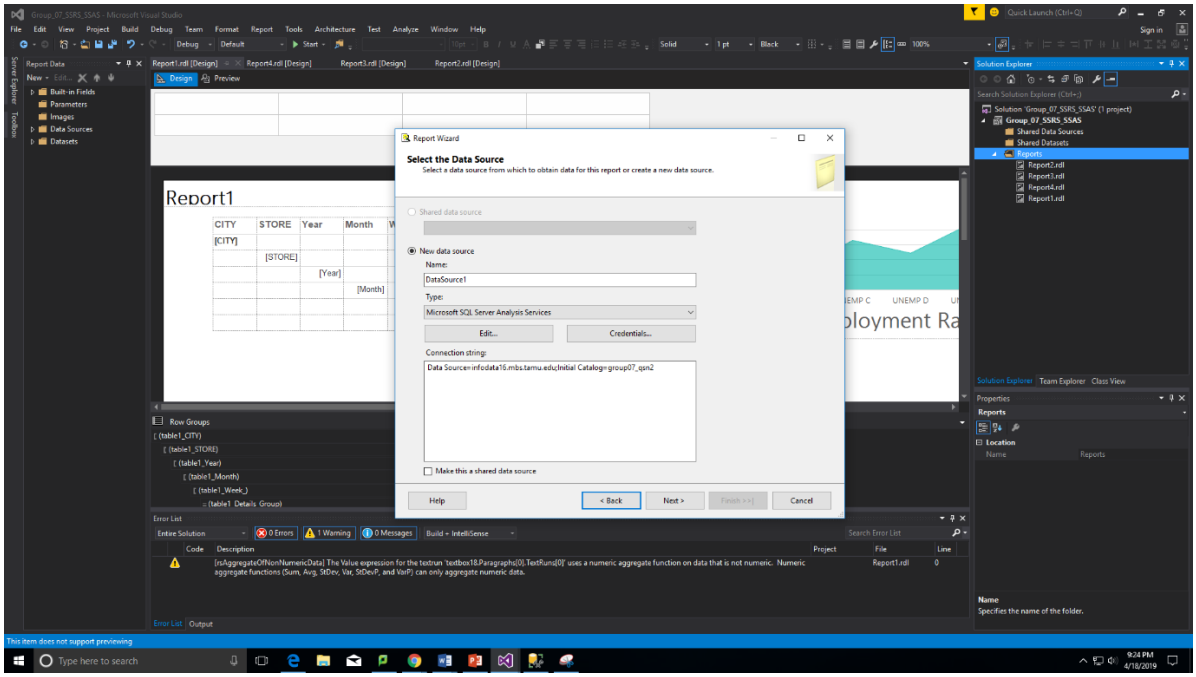


6.3 Report building from individual data marts

What is the effect of unemployment on sales in a particular city over time?

Based on the reporting plan discussed in the report earlier, we are using report building from individual data marts using *SSAS on top of SSRS* approach to create dashboards for this question

SSRS data source



SSRS query design

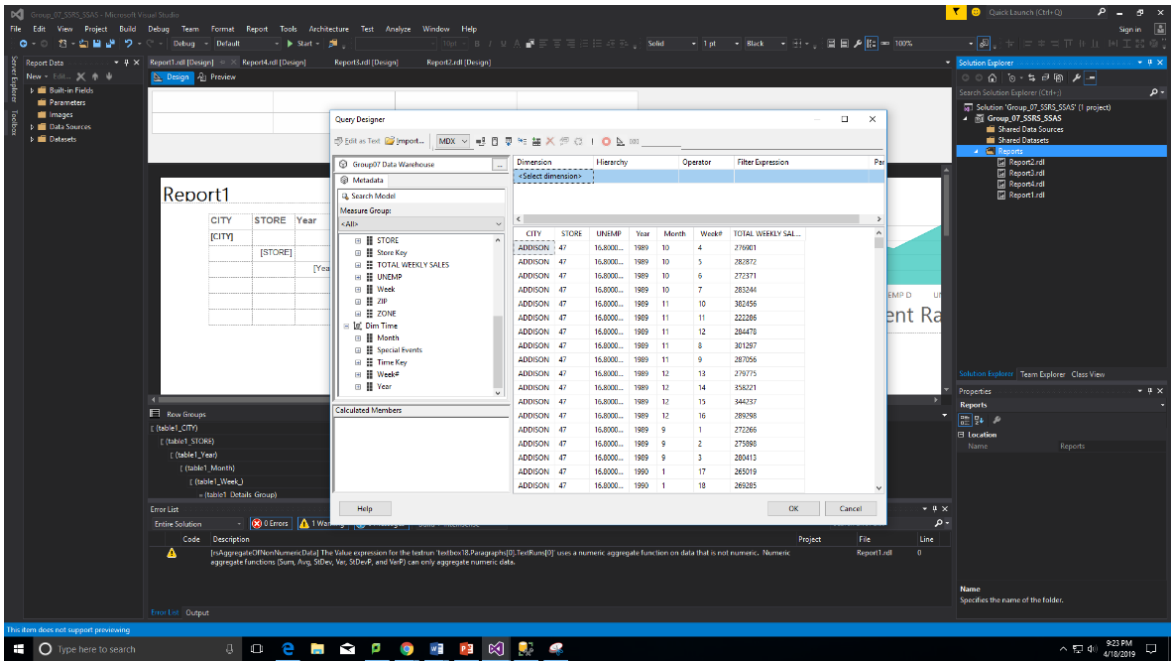
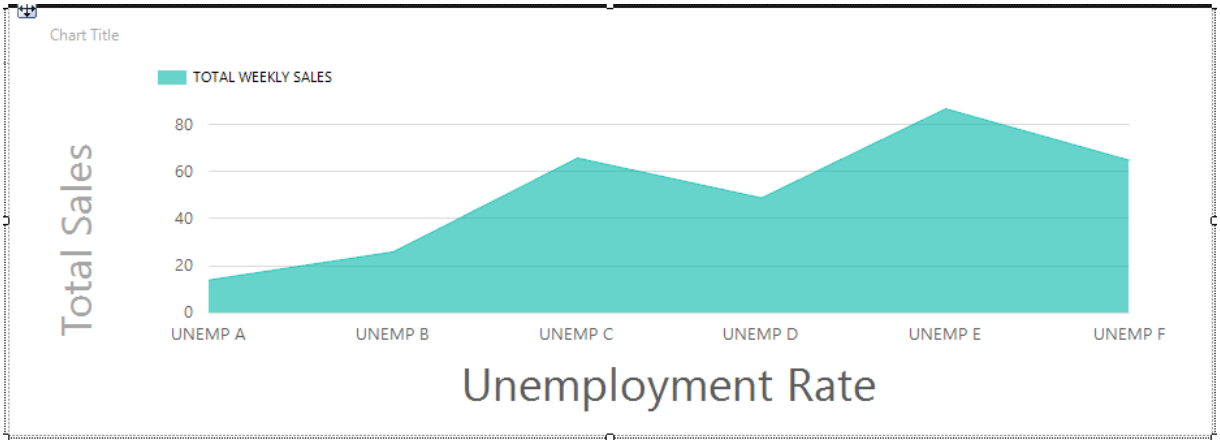


Table design

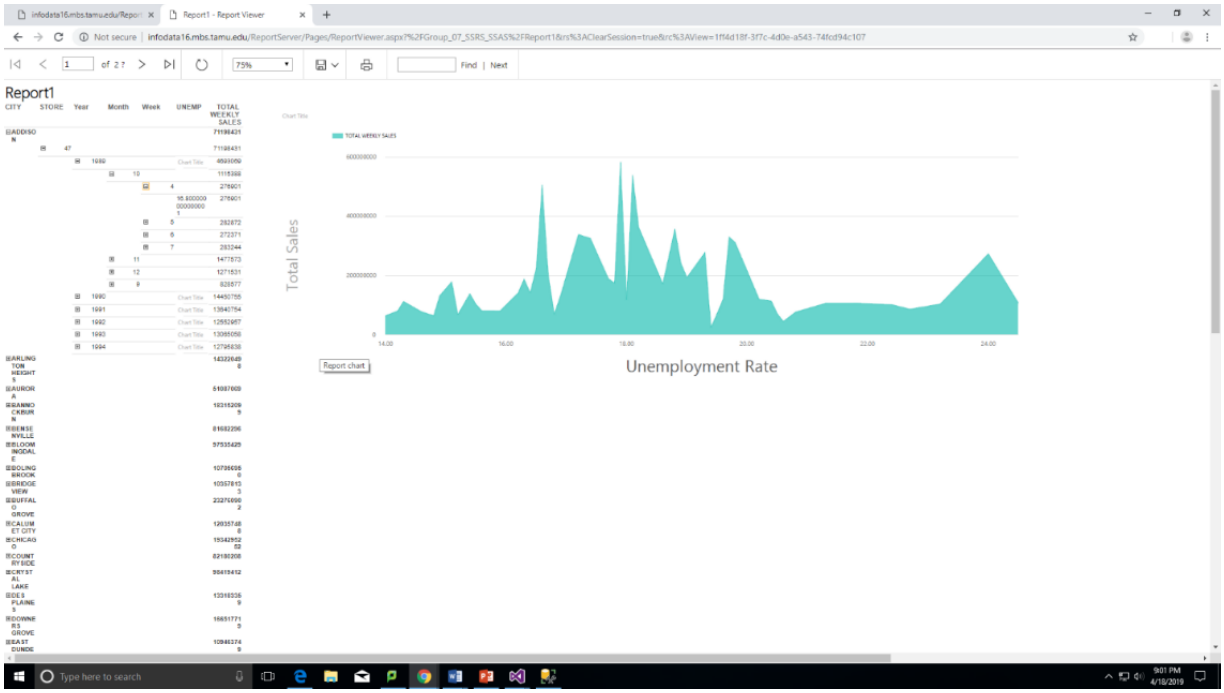
CITY	STORE	Year	Month	Week	UNEMP	TOTAL WEEKLY SALES
[CITY]						[Sum(TOTAL_WEEKLY_SALES)]
	[STORE]					[Sum(TOTAL_WEEKLY_SALES)]
		[Year]			Chart Title	[Sum(TOTAL_WEEKLY_SALES)]
			[Month]			[Sum(TOTAL_WEEKLY_SALES)]
				[Week_]		[Sum(TOTAL_WEEKLY_SALES)]
					[UNEMP]	[TOTAL_WEEKLY_SALES]

Chart design

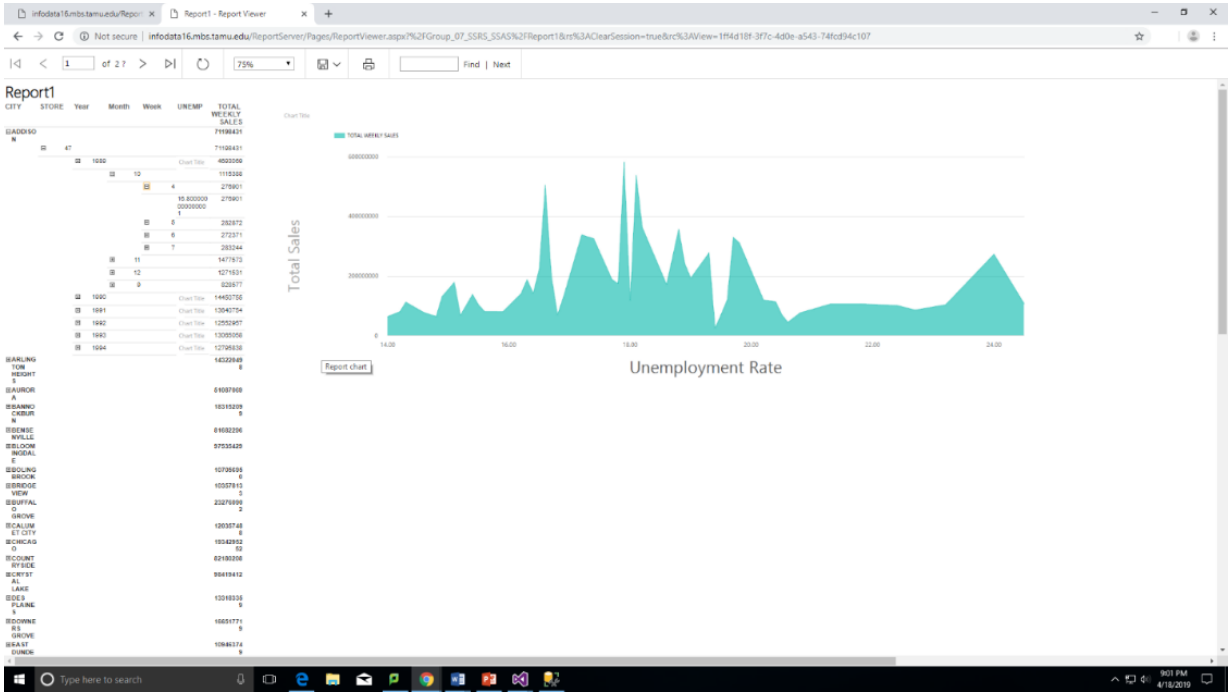


Deployment of report on infodata16.mbs.tamu.edu

Report deployed before drilldown



Report deployed after drilldown



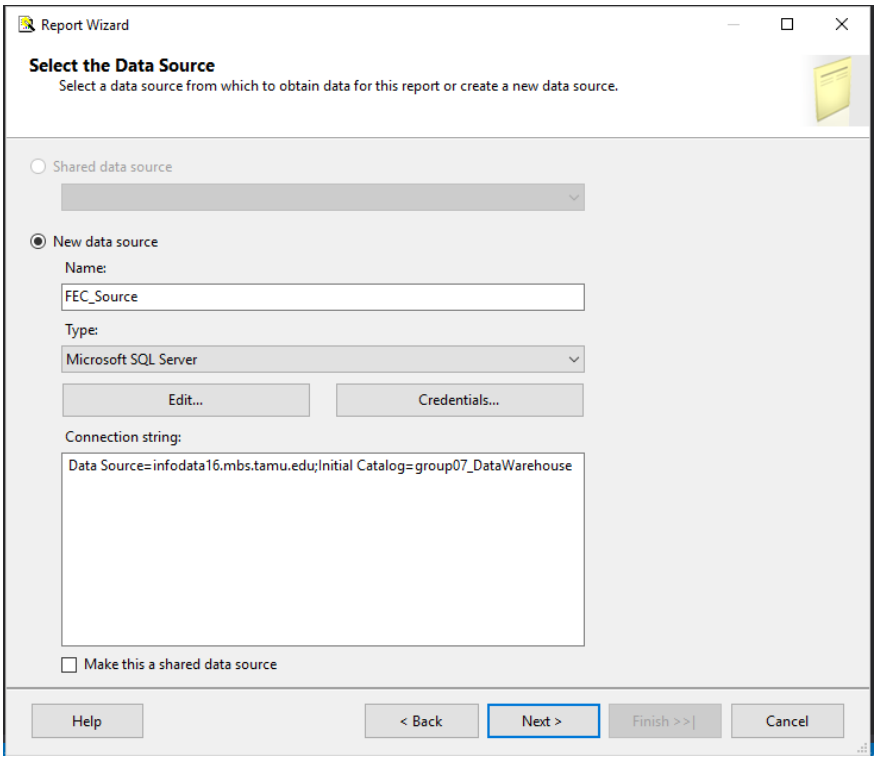
Conclusion

From the above graphs created using SSAS on top of SSRS, we notice that as the percentage of unemployment goes beyond 20%, the total weekly sales falls considerably. This graph provides data to DFF using which they can expand into cities with higher income margins as it is a known fact that a city impacted by the unemployment wave tends to be a low income market.

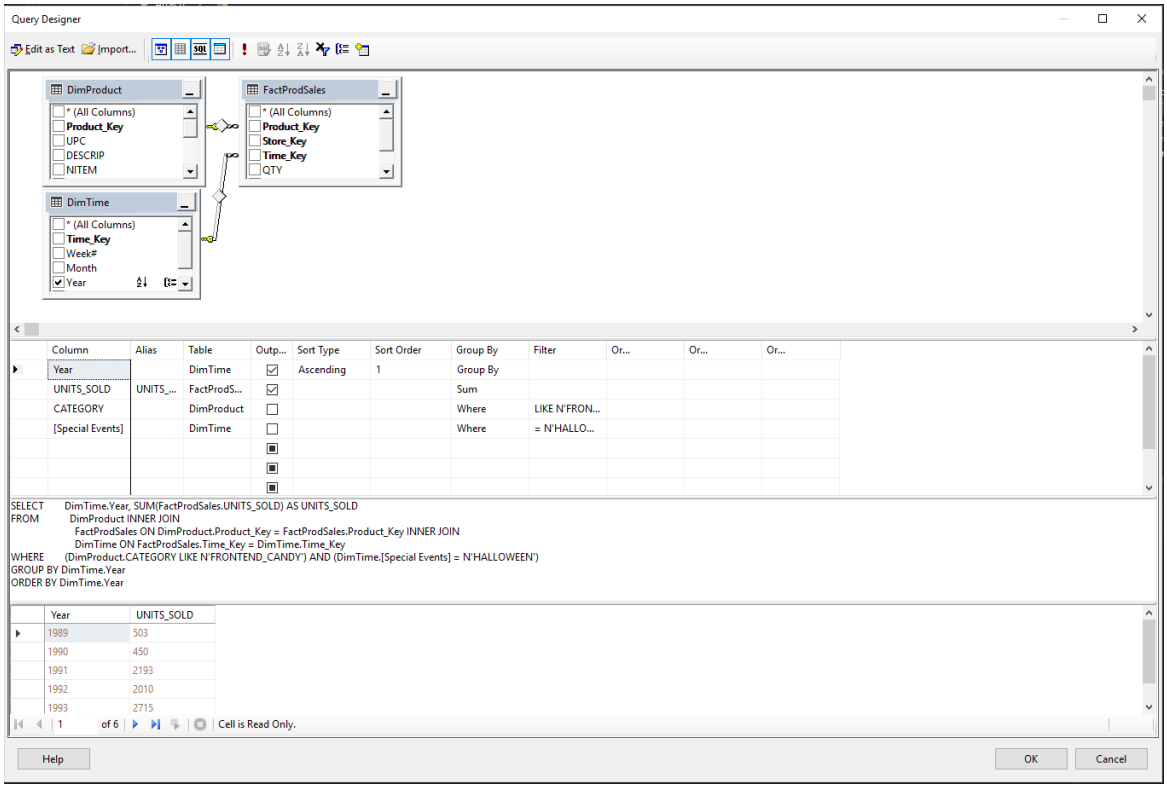
What is the trend of front-end candy sales during Halloween?

Based on the reporting plan discussed in the report earlier, we are using report building from individual data marts using *SSRS approach* to create dashboards for this question

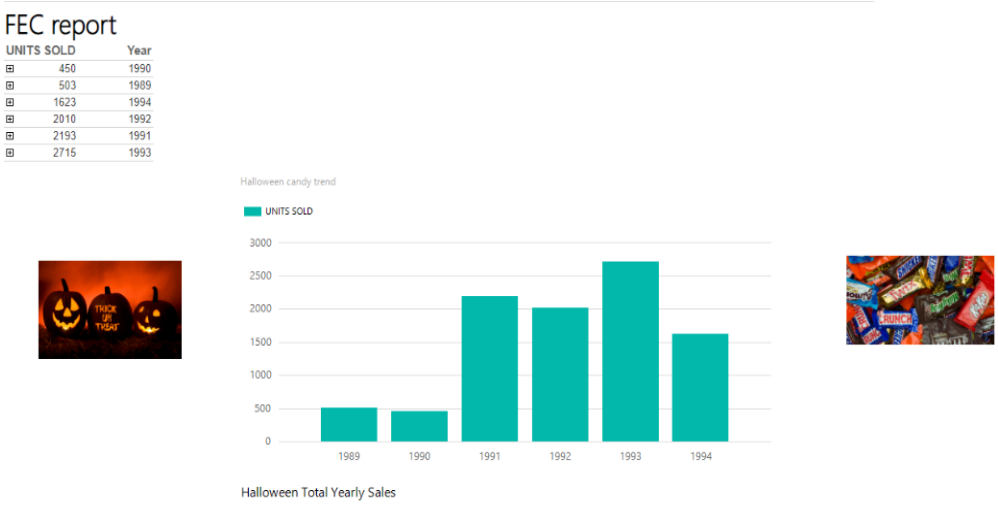
Data source creation



Query designer



Final report and bar chart design



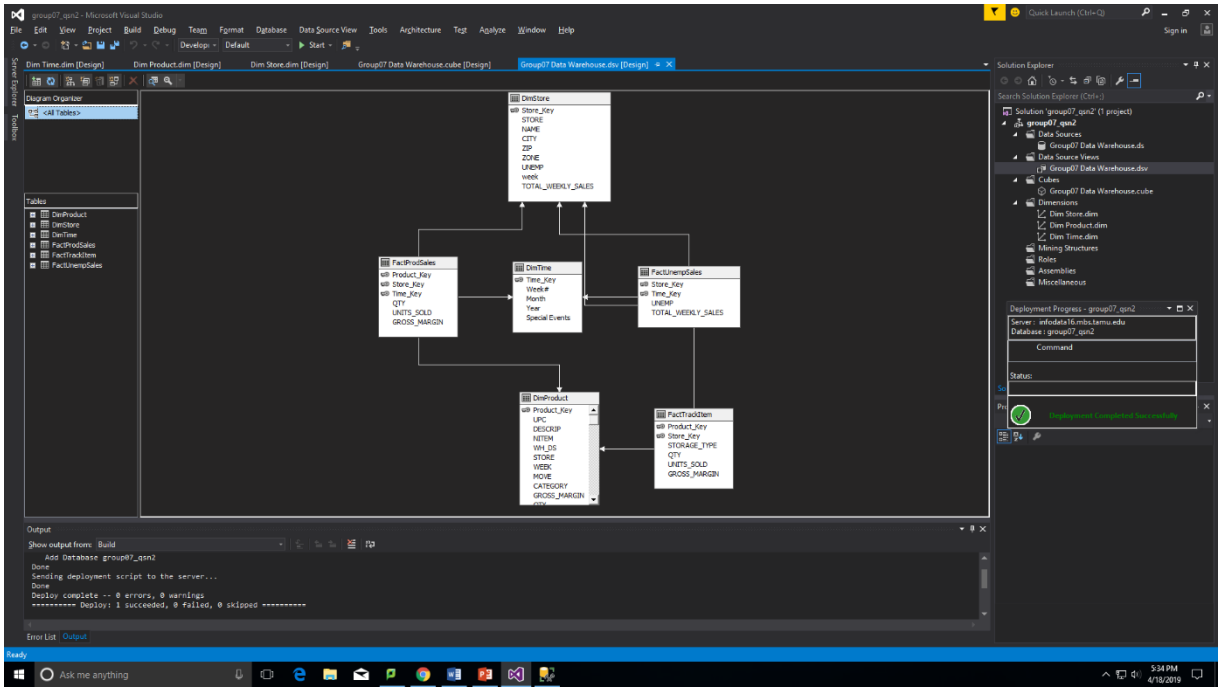
Conclusion

Based on the graph created using SSRS, it is evident that the front-end candy sales during Halloween is at its peak during the holiday season of 1993 and for the most part, the units sold increases as the years progress. DFF can use this information to perform inventory management so that they can increase capacity and production during Halloween in order to meet the demands of candies. They can also charge a premium during this time of the year so as to boost sales and increase profit margins.

What is the trend of cheese sales from the year 1989 to 1993?

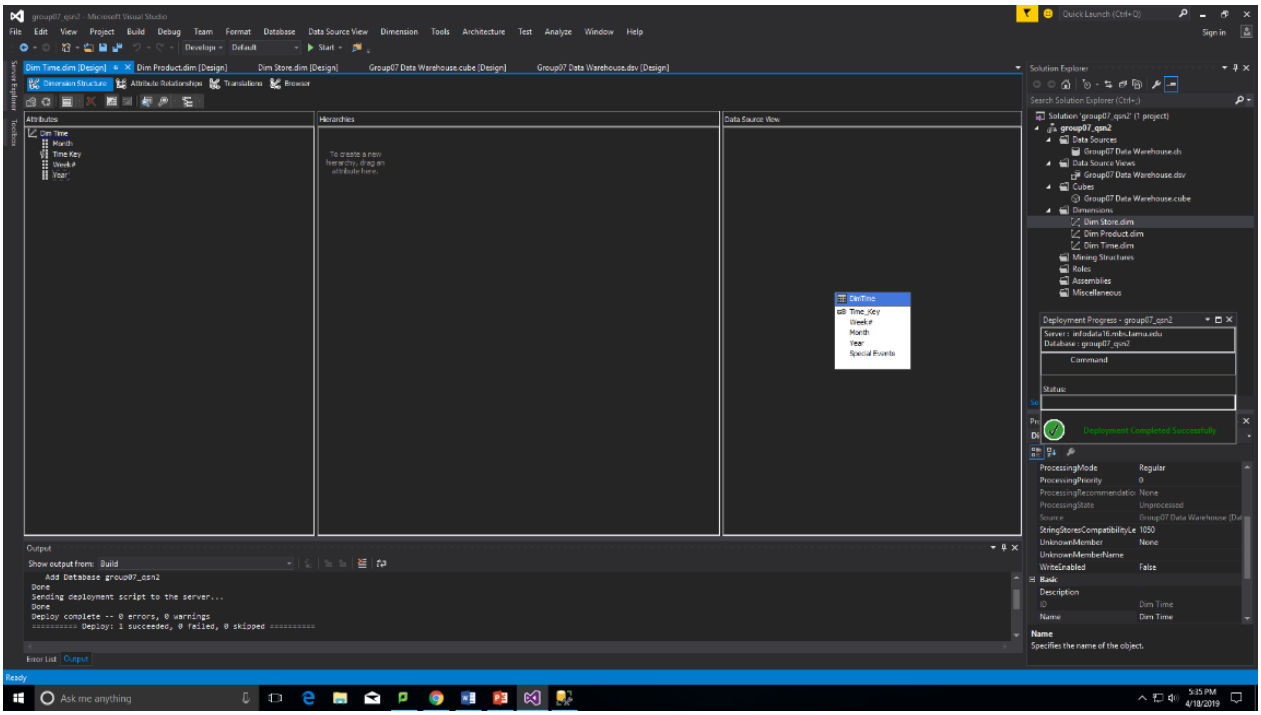
Based on the reporting plan discussed in the report earlier, we are using report building from individual data marts using *SSAS approach* to create dashboards for this question

Data source view

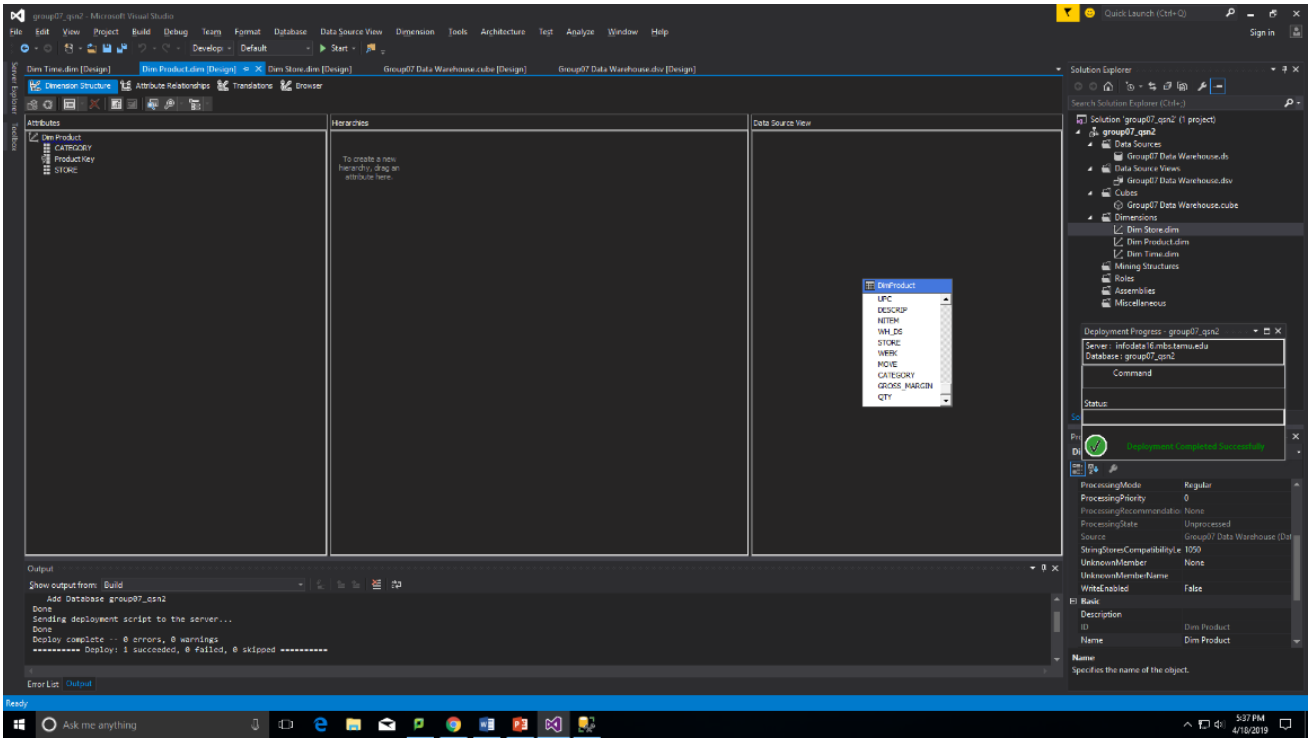


Dimensions

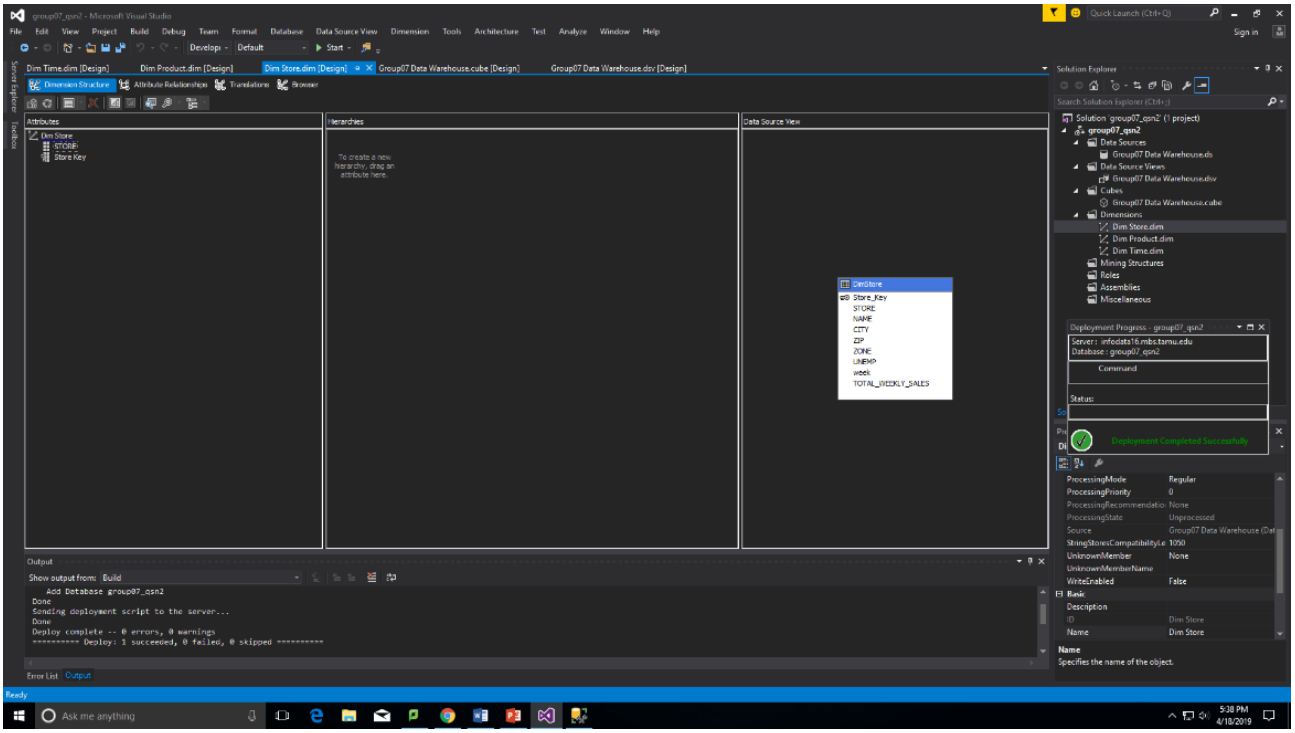
Time dimension



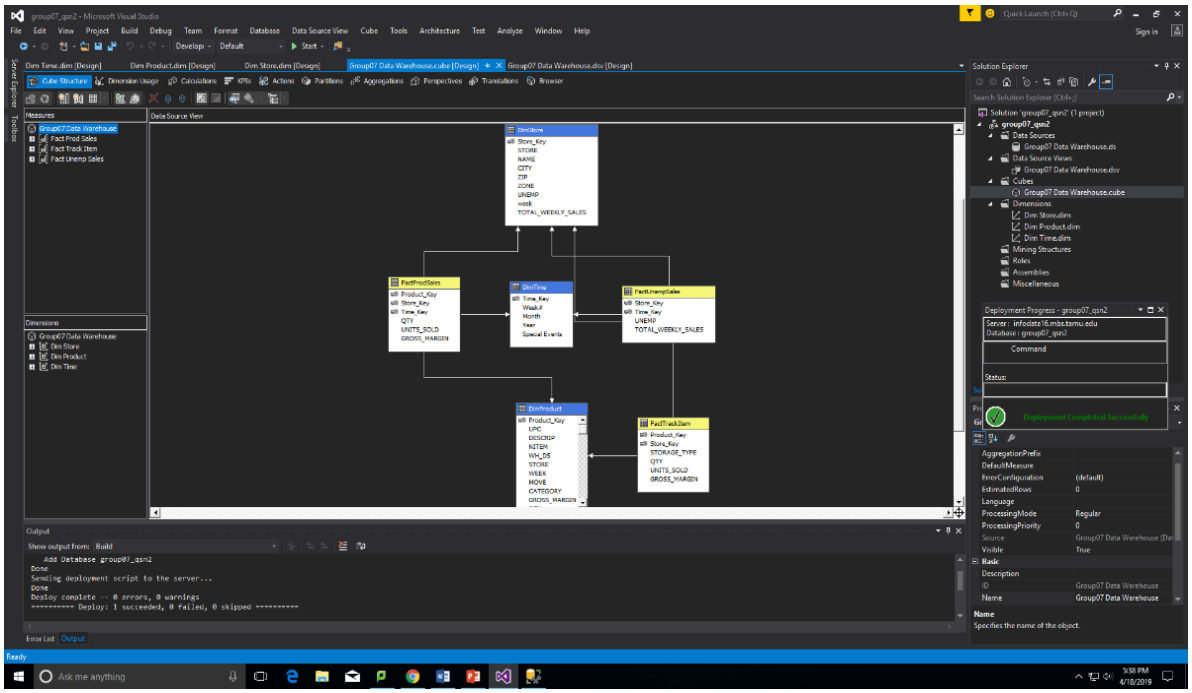
Product dimension



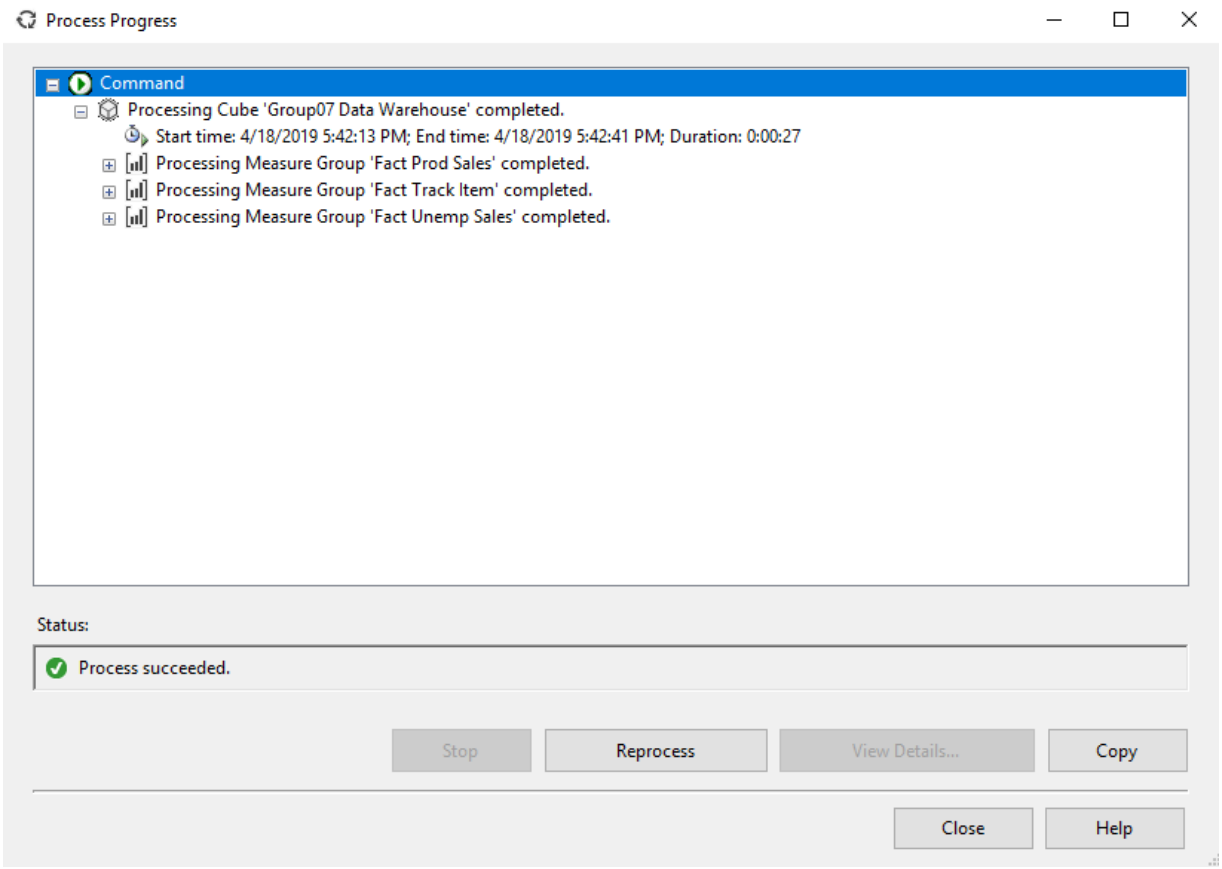
Store dimension



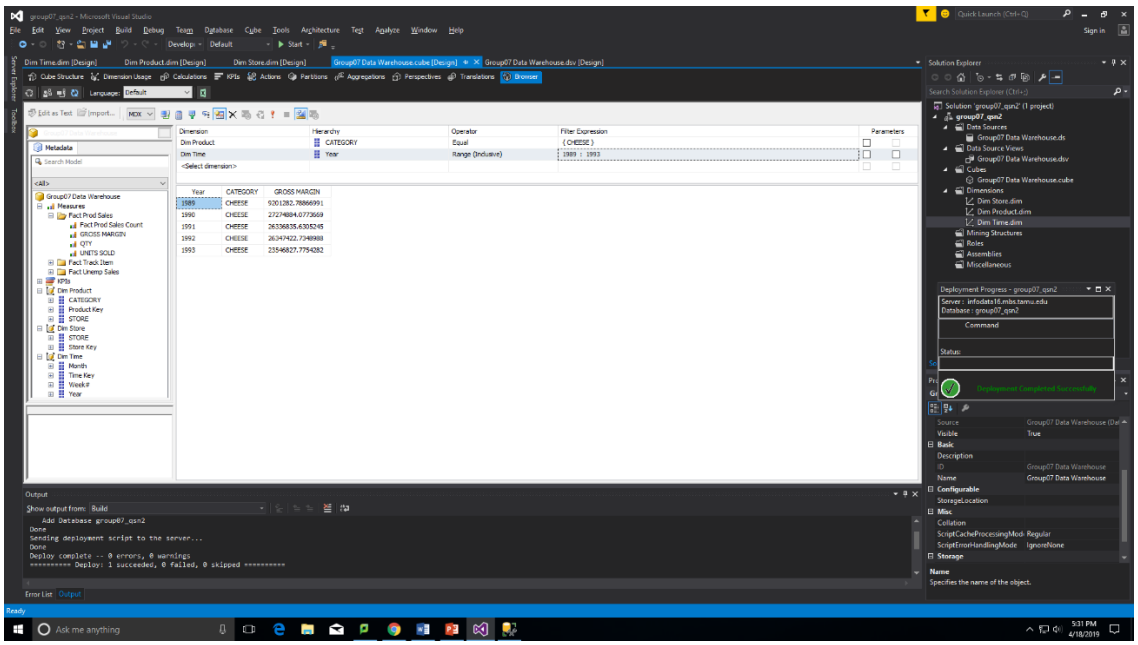
Cube structure



Successful process



Cube browser



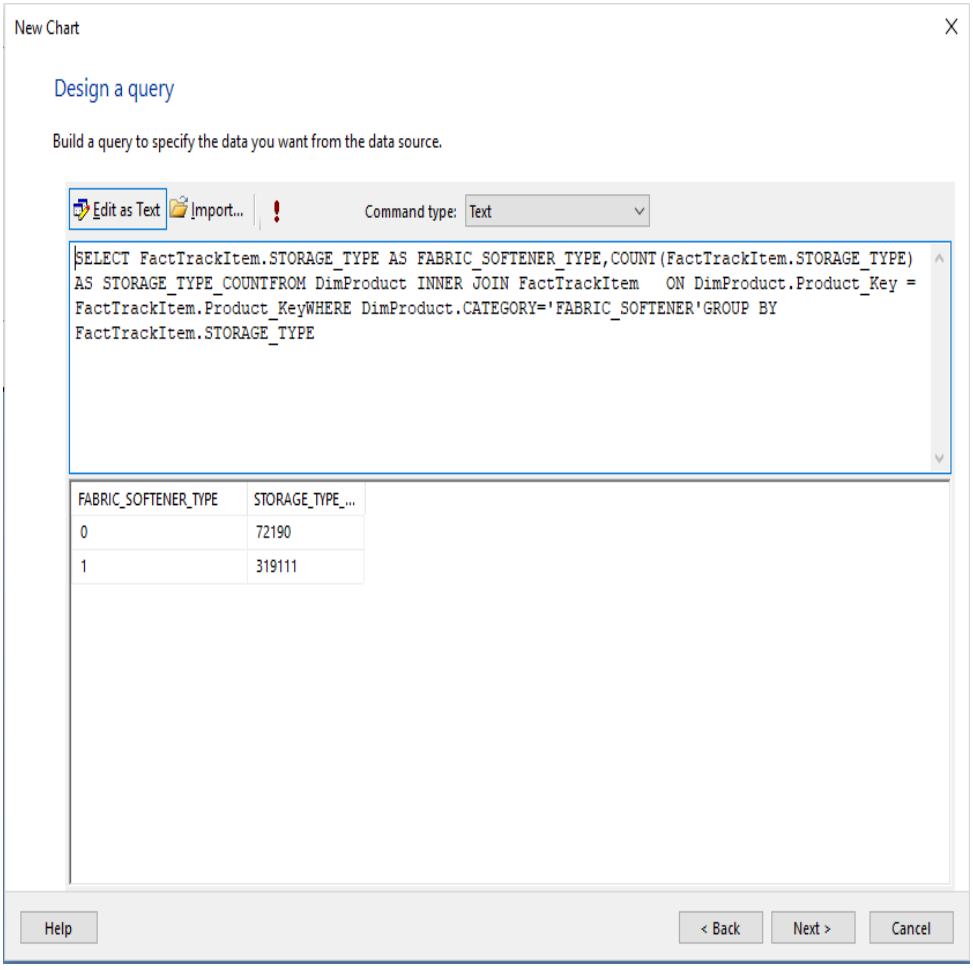
Conclusion

The cube designed using SSAS depicts the gross margin of DFF owing to sale of cheese from the year 1989 to 1993. It shows that the sale of sale was highest during 1990 and then reduced till 1993. This information can be leveraged by DFF to study how cheese is faring in the market and will help in deciding if it should be discontinued in the event of losses. DFF can also determine product improvements in order to boost sales.

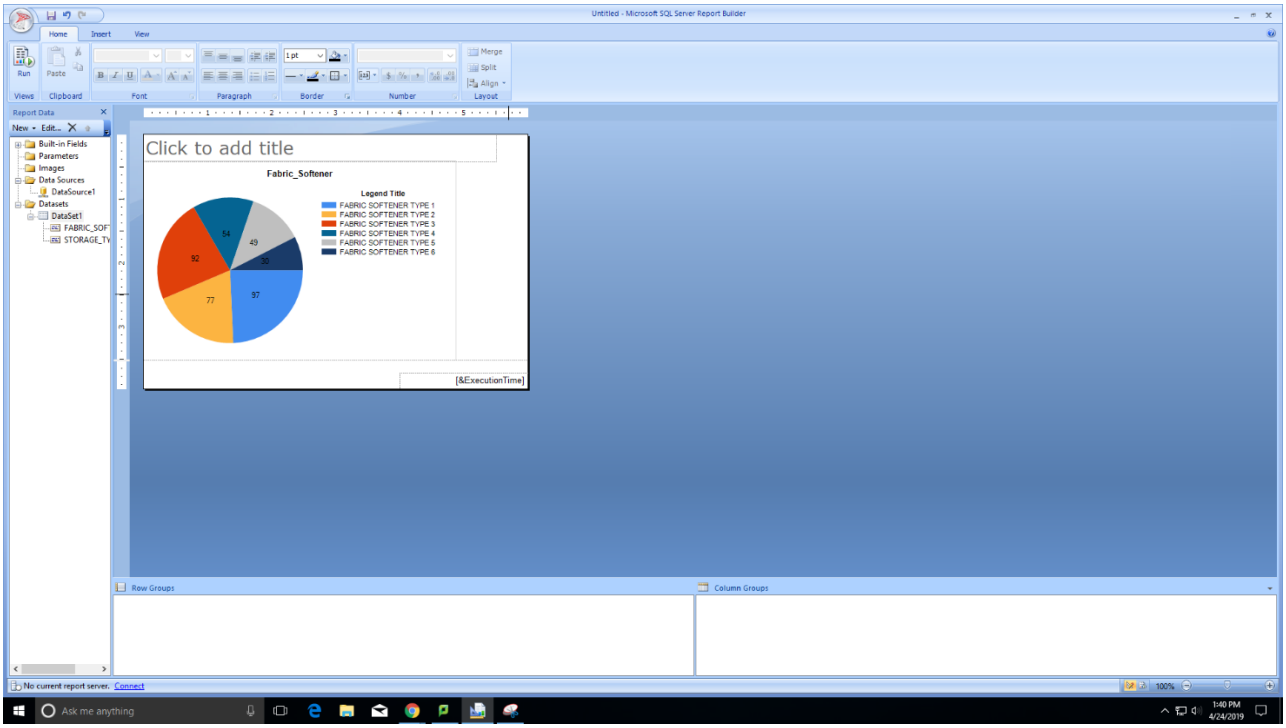
How many Fabric softeners and frozen dinners were drop-shipped vs warehoused? (0 or 1 at the last digit of the item code) according to each individual item?

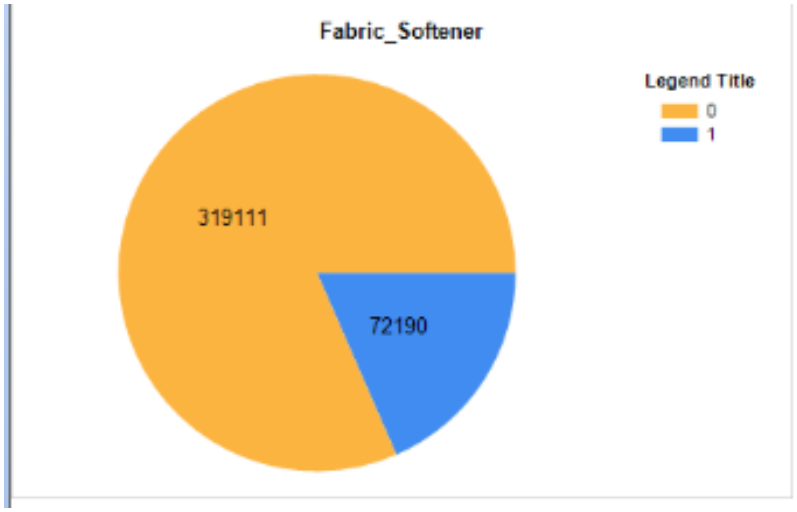
Based on the reporting plan discussed in the report earlier, we are using report building from individual data marts using *Report Builder 3.0 approach* to create dashboards for this question

Query design for fabric softener

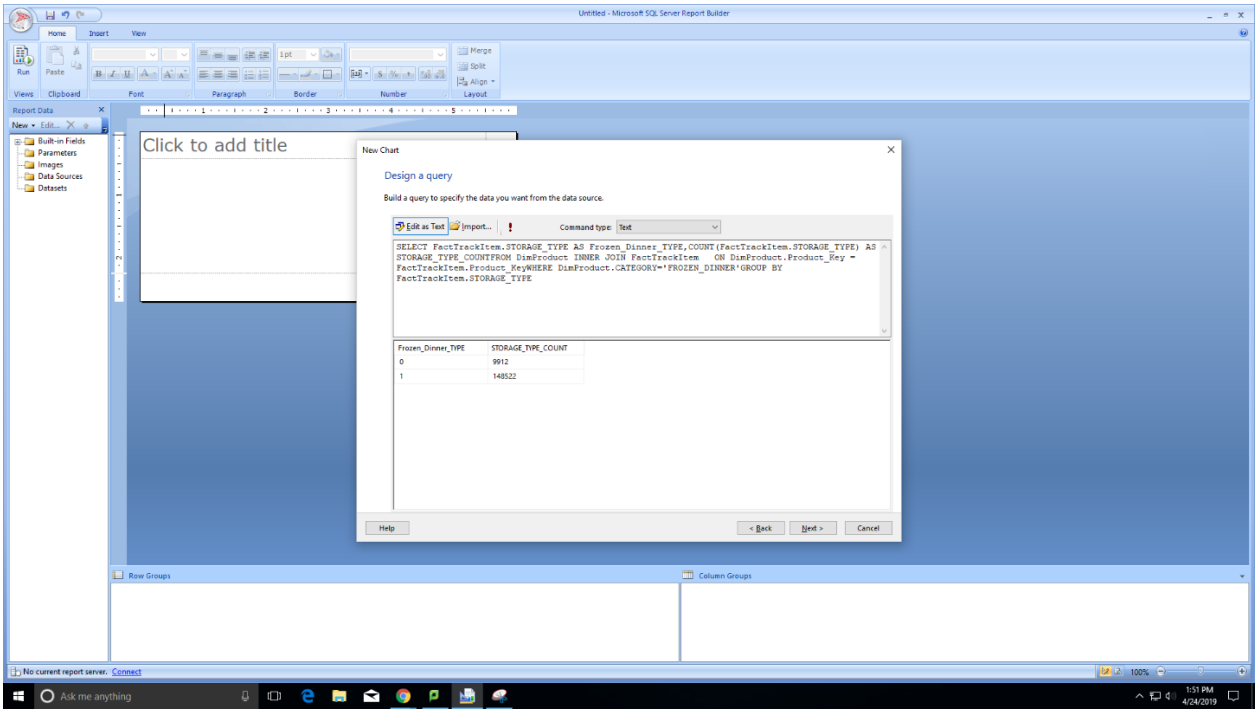


Pie chart design for fabric softener

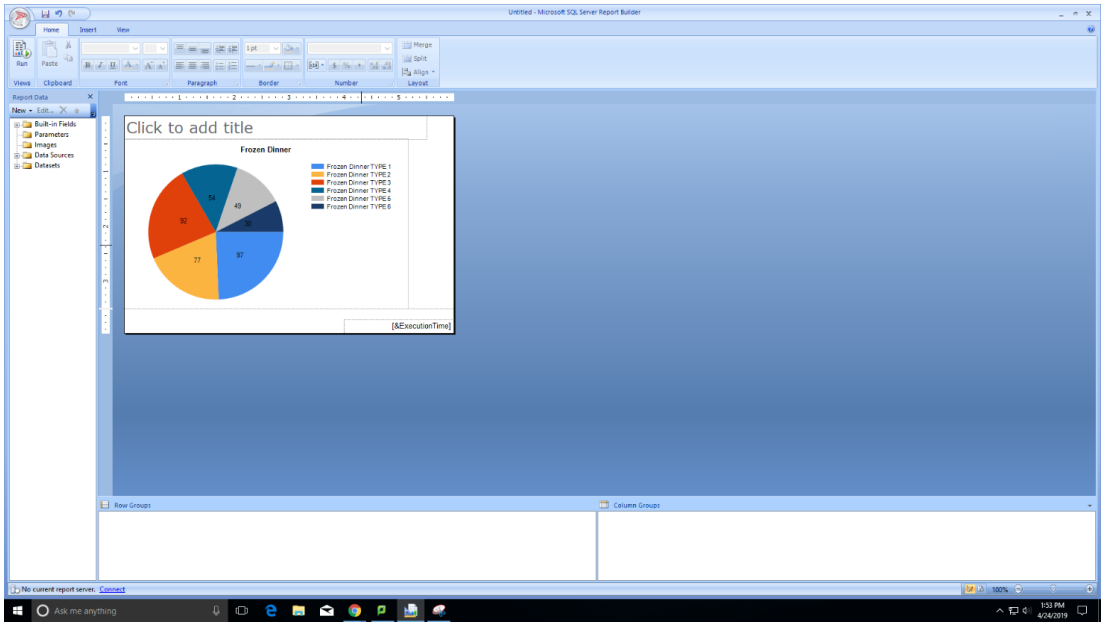


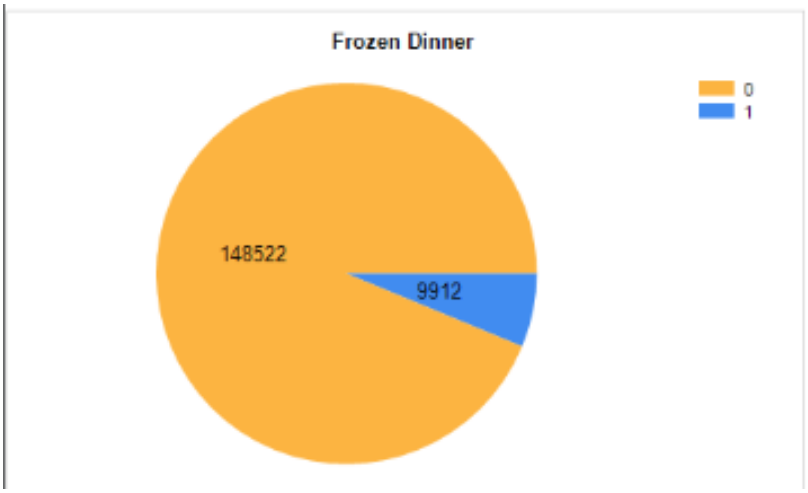


Query design for frozen dinner



Pie chart design for frozen dinner





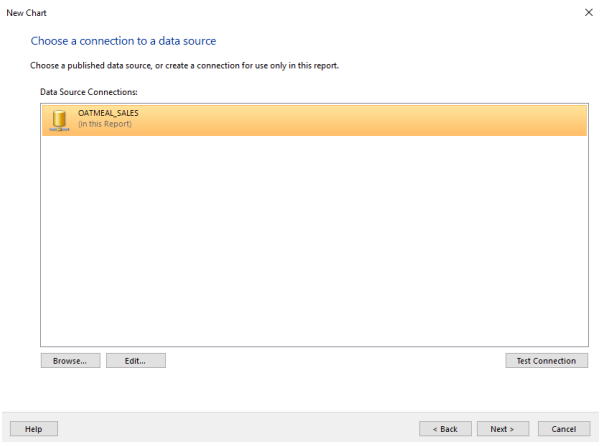
Conclusion

The pie graphs created using Report Builder 3.0 represent how many fabric softeners and frozen dinners were drop-shipped or warehoused. It is seen that more frozen dinners are drop-shipped in comparison to fabric softeners as the former is perishable and the latter has a long shelf life. This information can be used by DFF for inventory management and to avoid losses due to expiry of products.

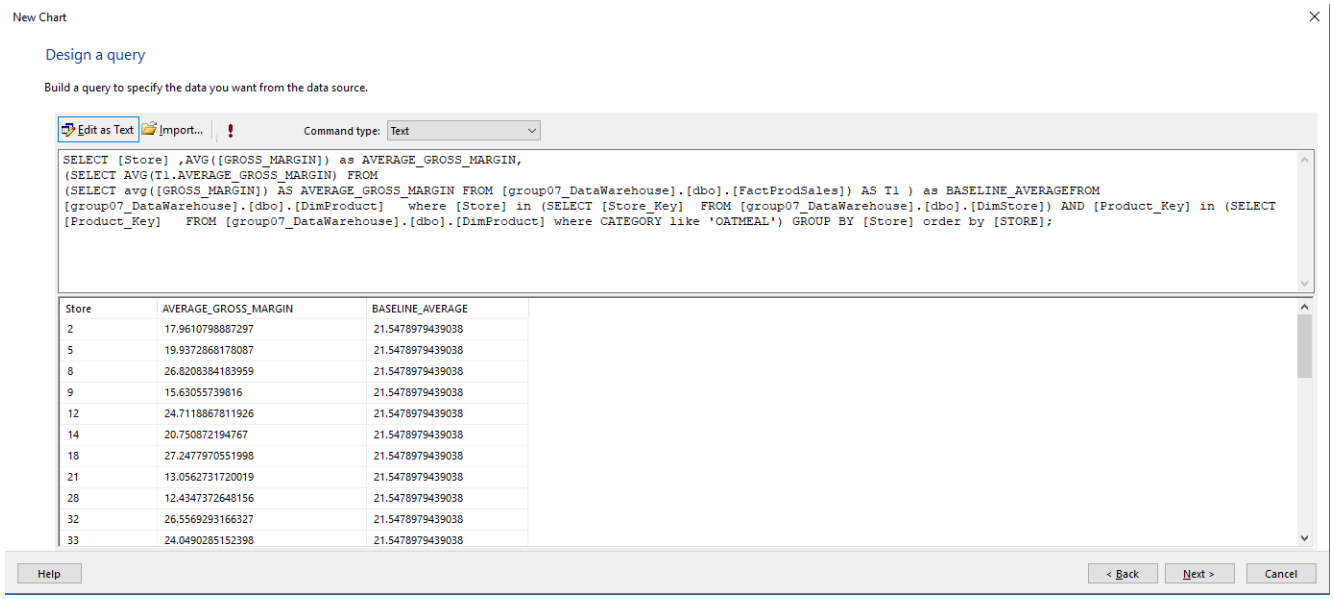
Find the average gross margin for oatmeal across all stores and determine what stores are performing below average

Based on the reporting plan discussed in the report earlier, we are using report building from individual data marts using *Report Builder 3.0 approach* to create dashboards for this question

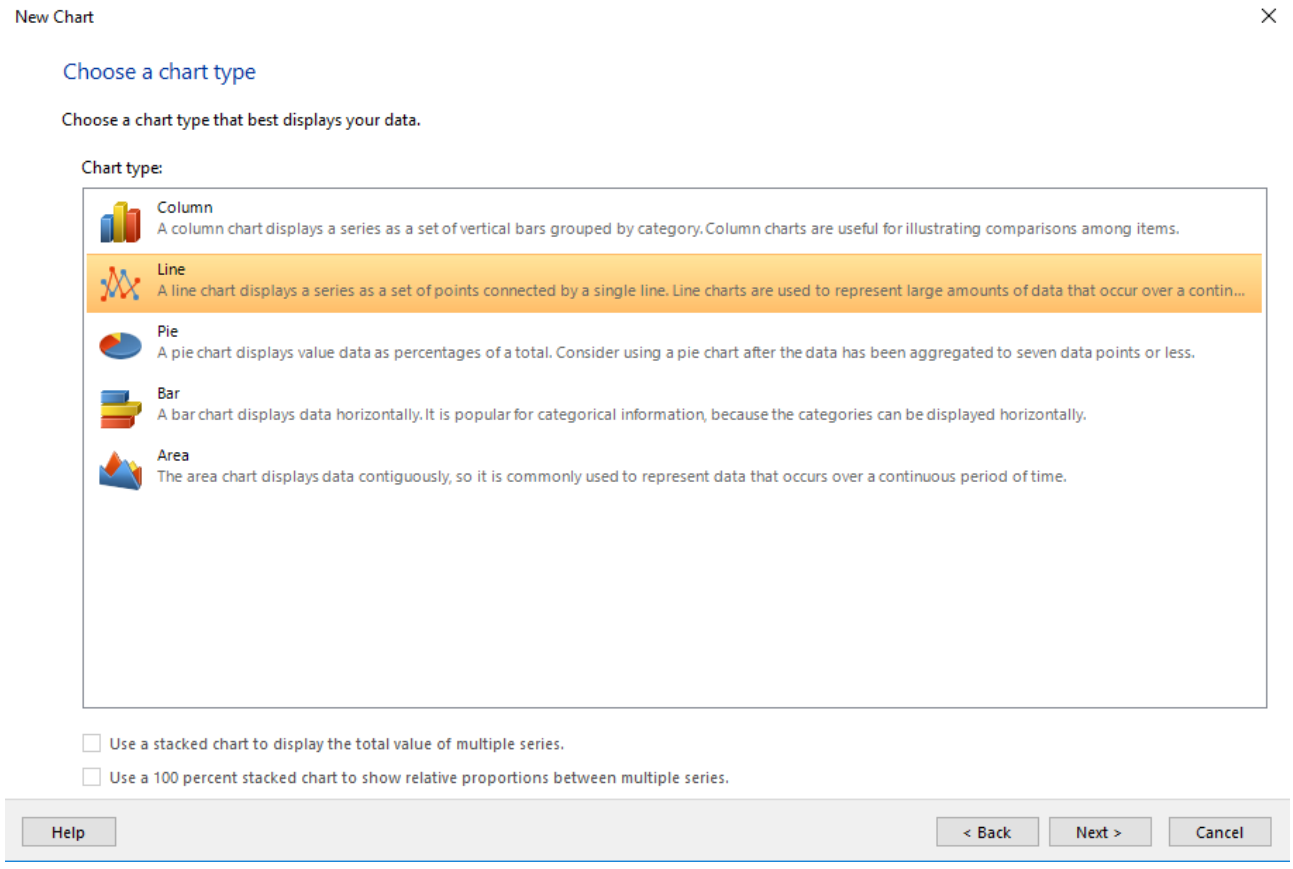
Data source creation



Query design



Choice of chart type



Arrange chart fields

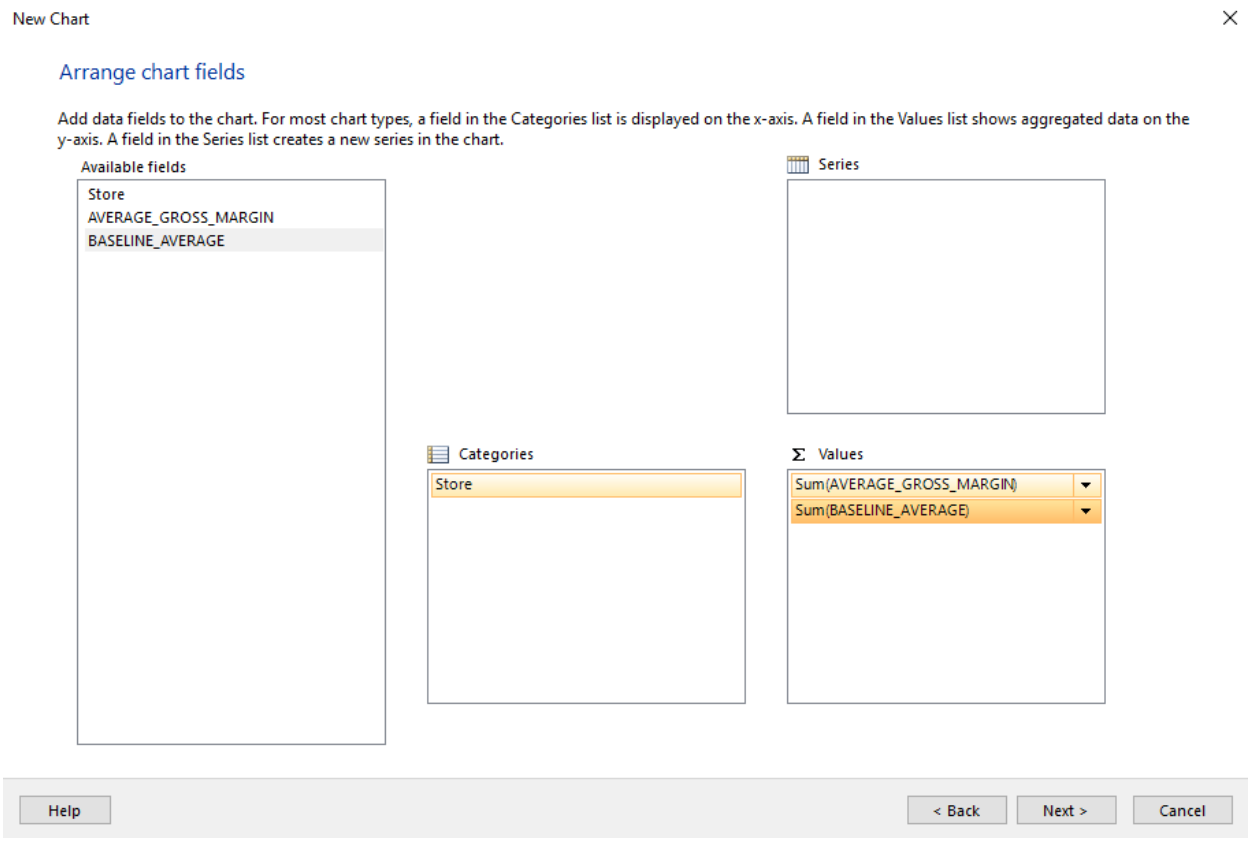
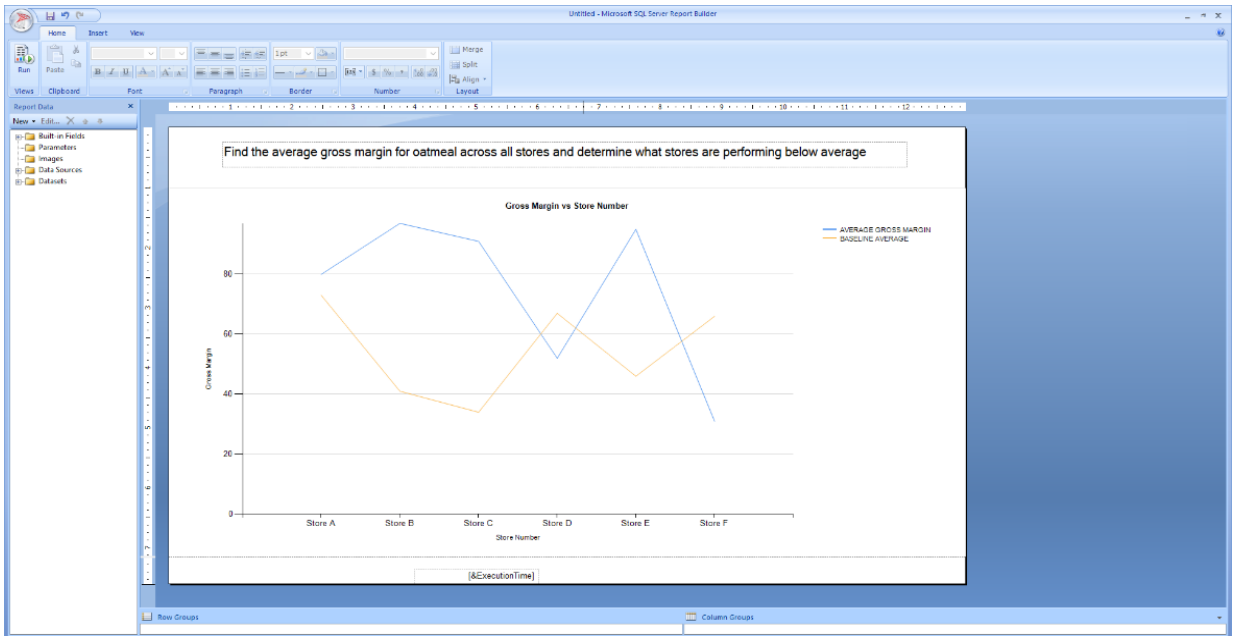
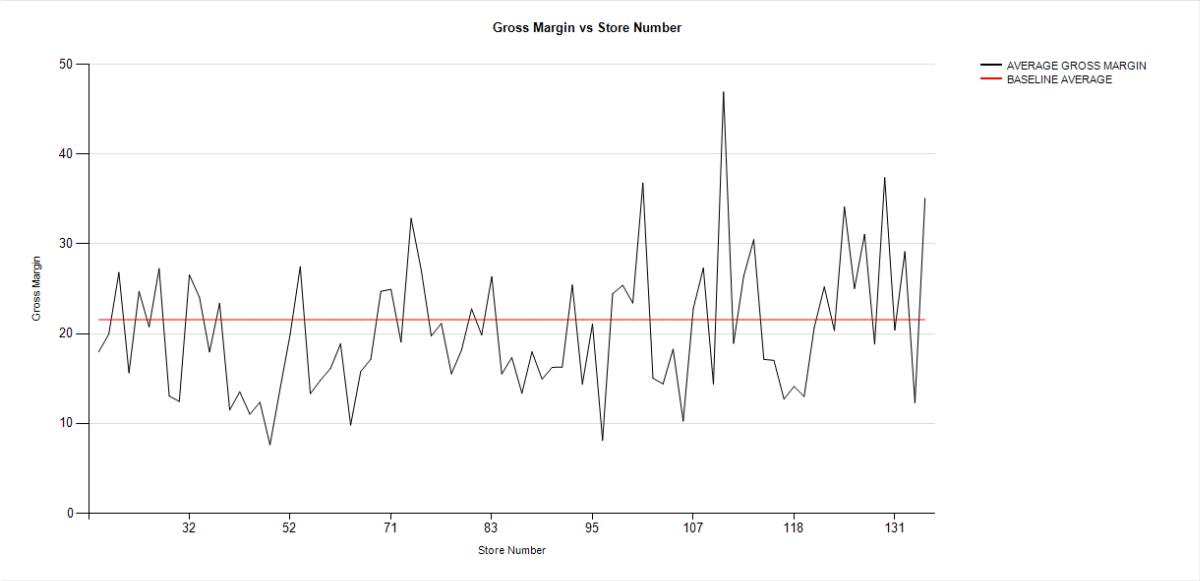


Chart design



Find the average gross margin for oatmeal across all stores and determine what stores are performing below average



4/18/2019 8:58:58 PM

Conclusion

Using Report Builder 3.0 the above graph was created which depicts the average gross margin for oatmeal across the various DFF stores with reference to a baseline average. We see that store number 112 (approximately) has the highest gross margin and store number 25 (approximately) is performing below average. DFF can leverage this information to invest in R&D on other organic foods such as oatmeal cookies, cereals etc and also determine which stores to increase production and sales at

7. References

- https://www.lifewire.com/facts-vs-dimensions-1019646
- https://www.vertabelo.com/blog/technical-articles/data-warehouse-modeling-star-schema-vs-snowflake-schema
- https://en.wikipedia.org/wiki/Physical_data_model
- https://sesamesoftware.com/white-papers/physical-database-design/
- http://business.baylor.edu/gina_green/teaching/dw/spr16/Ponniah_data-warehousing-fundamentals-for-it-professionalsSecondEdition.pdf
- https://www.chicagobooth.edu/research/kilts/datasets/dominicks
- https://www.chicagobooth.edu//media/enterprise/centers/kilts/datasets/dominicks-dataset/dominicks-manual-and-codebook_kiltscenter.aspx
- https://www.chicagobooth.edu/research/kilts/datasets/dominicks
- https://www.chicagobooth.edu//media/enterprise/centers/kilts/datasets/dominicks-dataset/dominicks-manual-and-codebook_kiltscenter.aspx
- http://www.jmp.com/
- https://repository.upenn.edu/cgi/viewcontent.cgi?article=1441&context=marketing_papers
- http://publications.dyson.cornell.edu/research/researchpdf/sp/1994/Cornell_Dyson_sp9410.pdf
- https://pdfs.semanticscholar.org/ef88/03928a6e1f613b4df63871ee6846efcad82a.pdf
- http://www.learnmsbitutorials.net/etl-process-ssis-example.php
- https://en.wikipedia.org/wiki/SQL_Server_Integration_Services