BI System Specifications Document

Version 1.0
Date: 22/01/2024
Or H.

Table of Contents

1. General

1.1. Project's Objective

The project's aim is to develop a comprehensive Business Intelligence (BI) solution by utilizing the PriorityERP Database for NVIDIA's sales department. NVIDIA is a leading technology company renowned for its innovations in graphics processing units (GPUs) and artificial intelligence, playing a pivotal role in shaping advancements in gaming, data centers, and autonomous vehicles.

This initiative seeks to implement an all-encompassing Business Intelligence (BI) solution utilizing data from the PriorityERP system tailored for NVIDIA. The solution will include condensed data tables, emphasizing sales data, in addition to customer details, employee records, product information, territories, stores, dates, and other relevant metrics.

Moreover, the project will enable the graphical representation of data using dashboards and a reporting system crafted to efficiently cater to the needs of the company's management, department heads, and sales managers.

By analyzing customer data, we can aid NVIDIA in gaining deeper insights into customer preferences, purchasing patterns, and brand loyalty. This insight can pave the way for more precise and effective marketing campaigns. Furthermore, having a clear understanding of the top-selling products enables optimization of production quantities for each model, resulting in cost reduction.

1.2. Project Content

1.2.1. Data Cleaning and Preparation

While exploring the data, we have noticed some cleanings that has to be made and will be addressed in the ETL processes:

- We're developing a Data Mart for the NVIDIA's sales department, therefore, we will keep only the salesmen in our employee dimension table.
- We don't have an online store to represent the online sales, so we will add one.
- For the customers that don't have a store assigned to them, we will assign the online store to them.

1.2.2.   Datawarehouse's tables

Fact_Sales - Information about all the sales.
Dim_Customers - Information about the customers.
Dim_Employees - Information about the employees.
Dim_Producs - Information about the products
Dim_Stores - Information about the stores

1.2.3.   Entity Relationship Diagram

[Link](#)

1.2.4.   Source to Target Tables

[Link](#)

1.2.5.   Power BI Reports and Dashboard
1.2.5.1.   Customer Sales Analysis Report

The Customer Sales Analysis Report will include measures and visualizations that will point out the customers purchase habits.
- Total Quantity Bought by Country in a specific year.
- Total Revenue by Region in a specific year.
- Total Quantity bought every month in a specific year compared to previous year.
- Quantity of Top 10 Products by Revenue in a specific year compared to previous year.

1.2.5.2.   Sales Department Report

The Sales Department Report will include measures and visualizations that will point out the employees performance and sales.
- Total Revenue by Sub-Category in a specific year.
- Monthly Revenue Change in a specific year.
- Top 5 Employees by Revenue in a specific year.
- Top 10 Products by Revenue in a specific year compared to previous year.

1.2.5.3.   Management Report

The Management report will include measures and visualizations that will give the higher executives a good overview about the yearly revenue and sales.

- Total Revenue/Quantity by Store's location (including online store) in a specific year.
- Revenue/Quantity sold of the newly released RTX 40 Series Graphics Cards.
- Total Revenue/Quantity by Months in a specific year.
- Total Revenue/Quantity by Categories in a specific year.

2. Gant
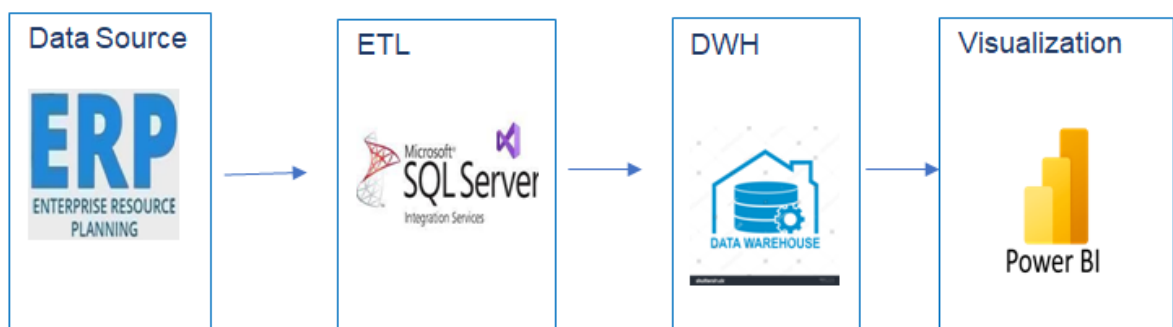
Link

3. Technical Specifications

3.1. Prerequisites

SQL Server: Data from the OLTP (Online Transaction Processing) server that includes PriorityERP database.

SSIS: ETL (Extract, Transform, Load) processes by utilizing SSMS projects that include different kinds of tasks.

Power BI: Utilizing Power BI to produce reports and dashboards.

3.2. Solution Architecture

Information extracted from the ERP system will be gathered and examined within SQL Server. Subsequently, an ETL process employing SSIS will be employed to structure the data within a Data Warehouse. The resulting metrics will be displayed in reports and visual representations through Power BI.
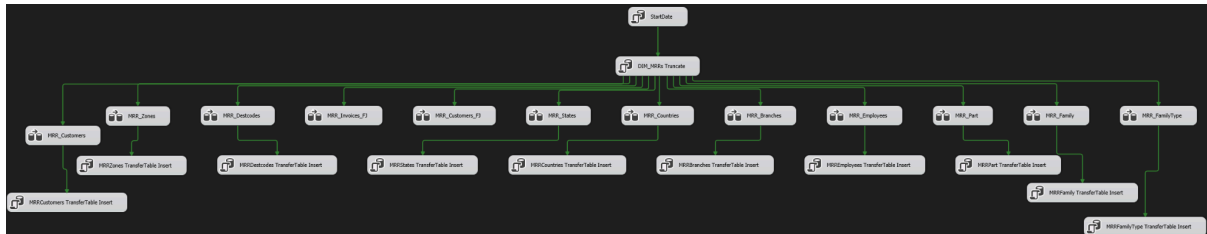
4. Functional Specifications

4.1. ETL Processes

We have 6 Solutions that contain 2 packages each, a total of 12 packages.
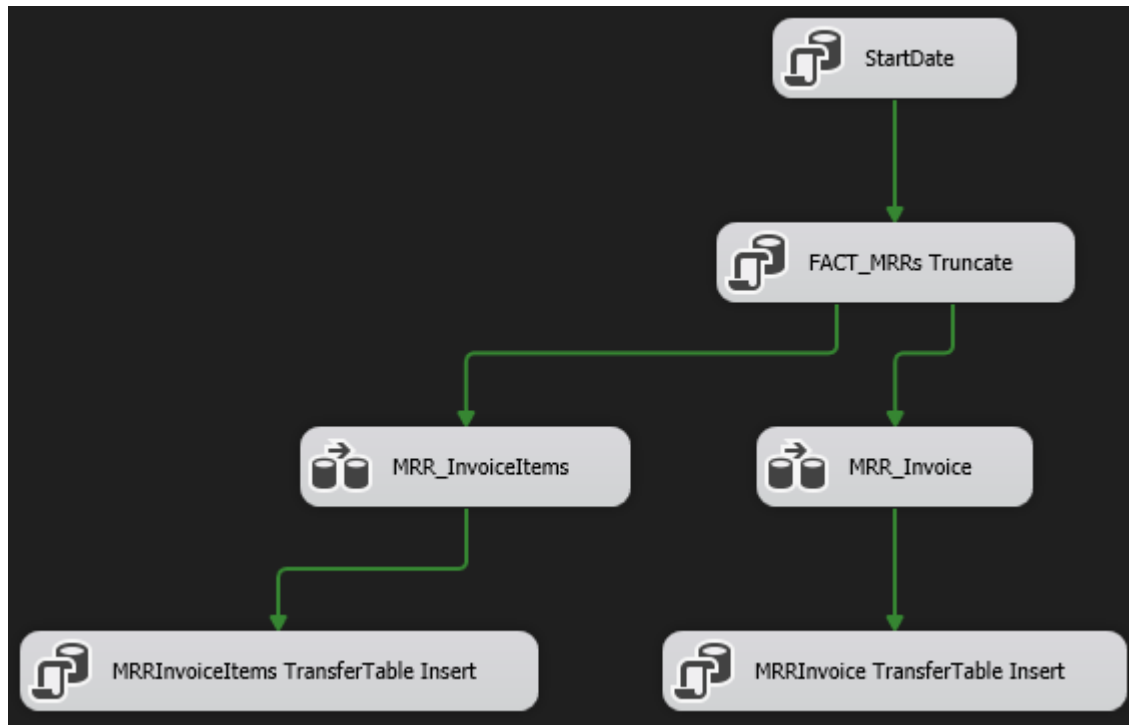
4.1.1. Mirror

Dim_MRRs:

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the mirror tables from previously loaded data.
- **Data Flow Tasks**:
    - **OLE DB Source** - The source table from the OLTP database.
    - **Row Count** - Assign the number of records to a variable.
    - **OLE DB Destination** - Mirror destination table.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.
- We've also implemented a business rule that takes only the records of customers that made an order in the last 3 years:
SELECT CUSTNAME , C.CUST , C.BRANCH, CONVERT(DATE,IVDATE)
IVDATE
    FROM CUSTOMERS C
    JOIN INVOICES S ON C.CUST = S.CUST
    WHERE DATEDIFF(yy,CONVERT(DATE,IVDATE),GETDATE()) < 3 order by
    CONVERT(DATE,IVDATE)
- We've also implemented a CDC (change data capture) process for employees, which inserts to the mirror only the new, changed or deleted data from the last update.
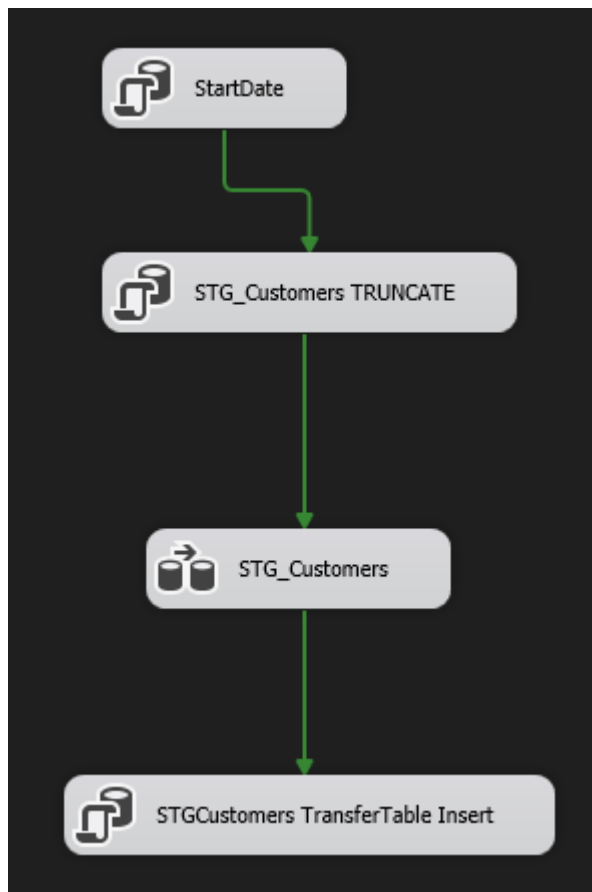
Fact_MRRs:

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the mirror tables from previously loaded data.
- **Data Flow Tasks**:
    - **OLE DB Source** - The source table from the OLTP database.
    - **Lookup** - Making sure that we don't load data that is already in the fact table.
    - **Row Count** - Assign the number of records to a variable.
    - **OLE DB Destination** - Mirror destination table.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

4.1.2. Customers

STG_Customers

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the stage tables from previously loaded data.
- **Data Flow Tasks**:
  - **OLE DB Source** - The mirror table from the OLAP database.
  - **SQL Query** - To merge the different mirror tables:
    SELECT DISTINCT C.CUSTNAME Name, C.CUST CustomerID, D.ADDRESS Address
    , D.CITY City, S.Region Region, S.Country Country, C.BRANCH StoreID
    FROM MRR_CUSTOMERS C
    LEFT JOIN MRR_INVOICES_FJ I ON C.CUST = I.CUST
    LEFT JOIN MRR_DESTCODES D ON I.DESTCODE = D.DESTCODE
    LEFT JOIN STG_Stores S ON C.BRANCH = S.StoreID
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Stage destination table.

- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.
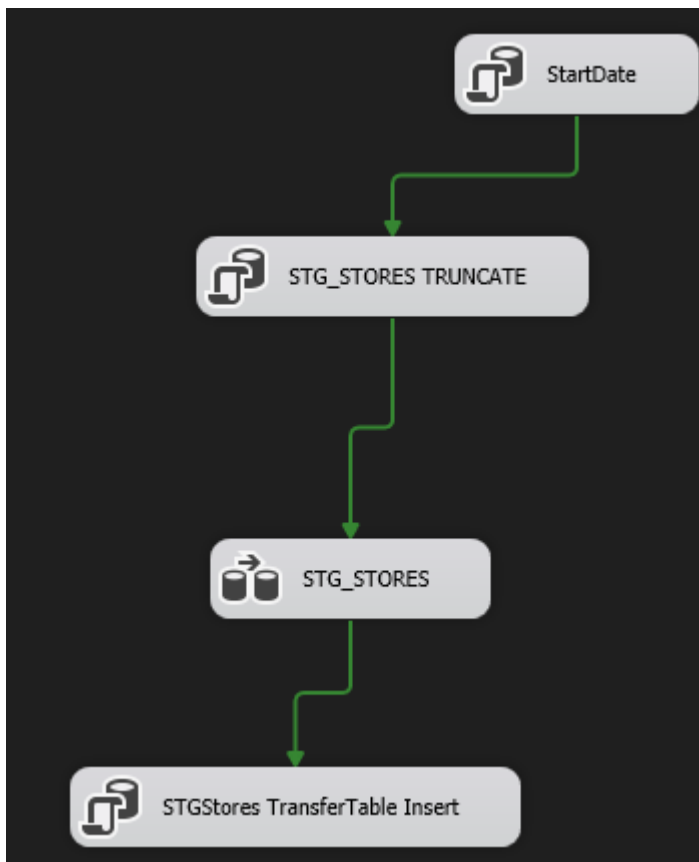
Dim_Customers
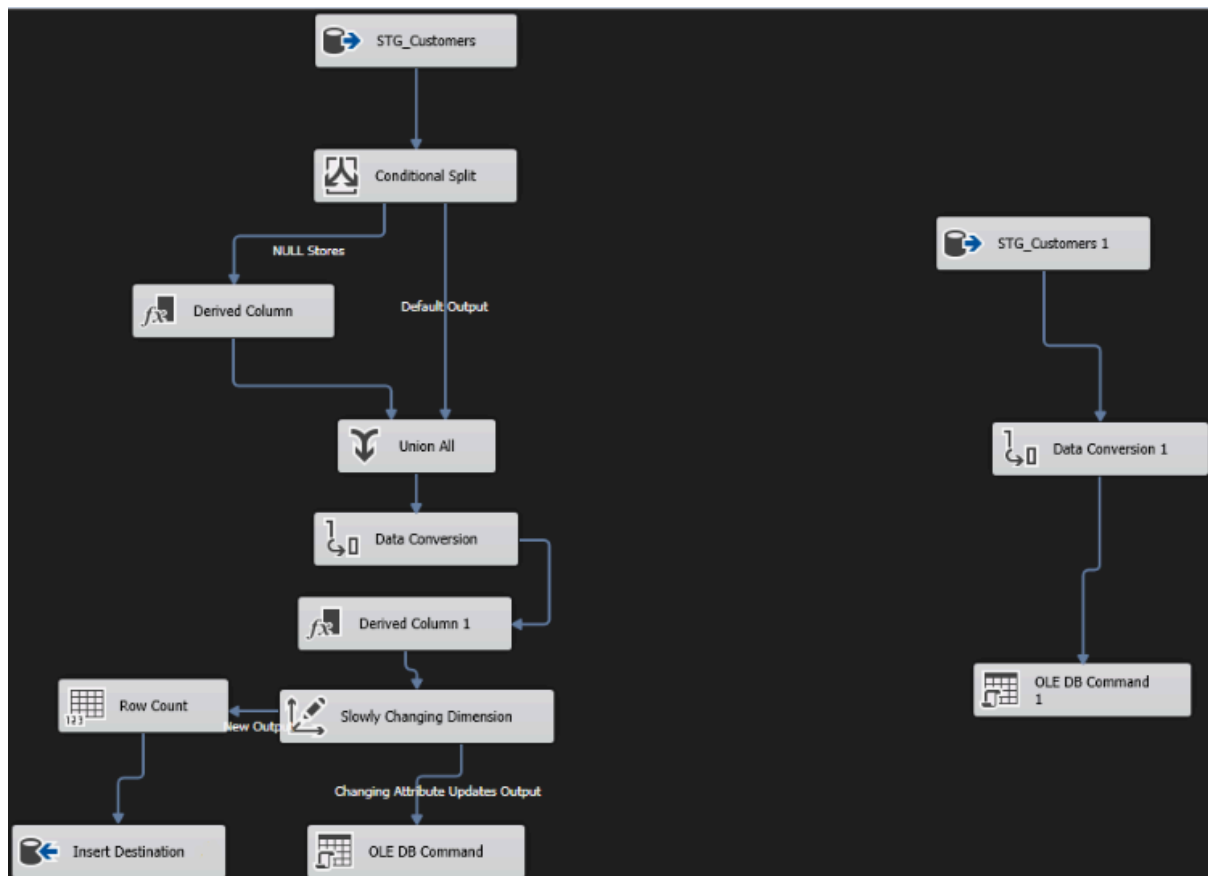
- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Data Flow Tasks**:
  - **OLE DB Source** - The stage table from the OLAP database.
  - **Conditional Split** - Splitting the store that has NULL value for location.
  - **Derived Column** - Assign the values for online store.
  - **Union All** - Union for the splitted data.
  - **Data Conversion** - Changing the different columns to unicode string to fit our destination table.
  - **Derived Column 1** - Adding column isActive with the value of "Y".
  - **Slowly Changing Dimension -** Making the split for new and changed records.
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Dimension destination table.
  - **OLE DB Command** - Changed records update.
  - **OLE DB Command 1** - Deleted records update.

- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.
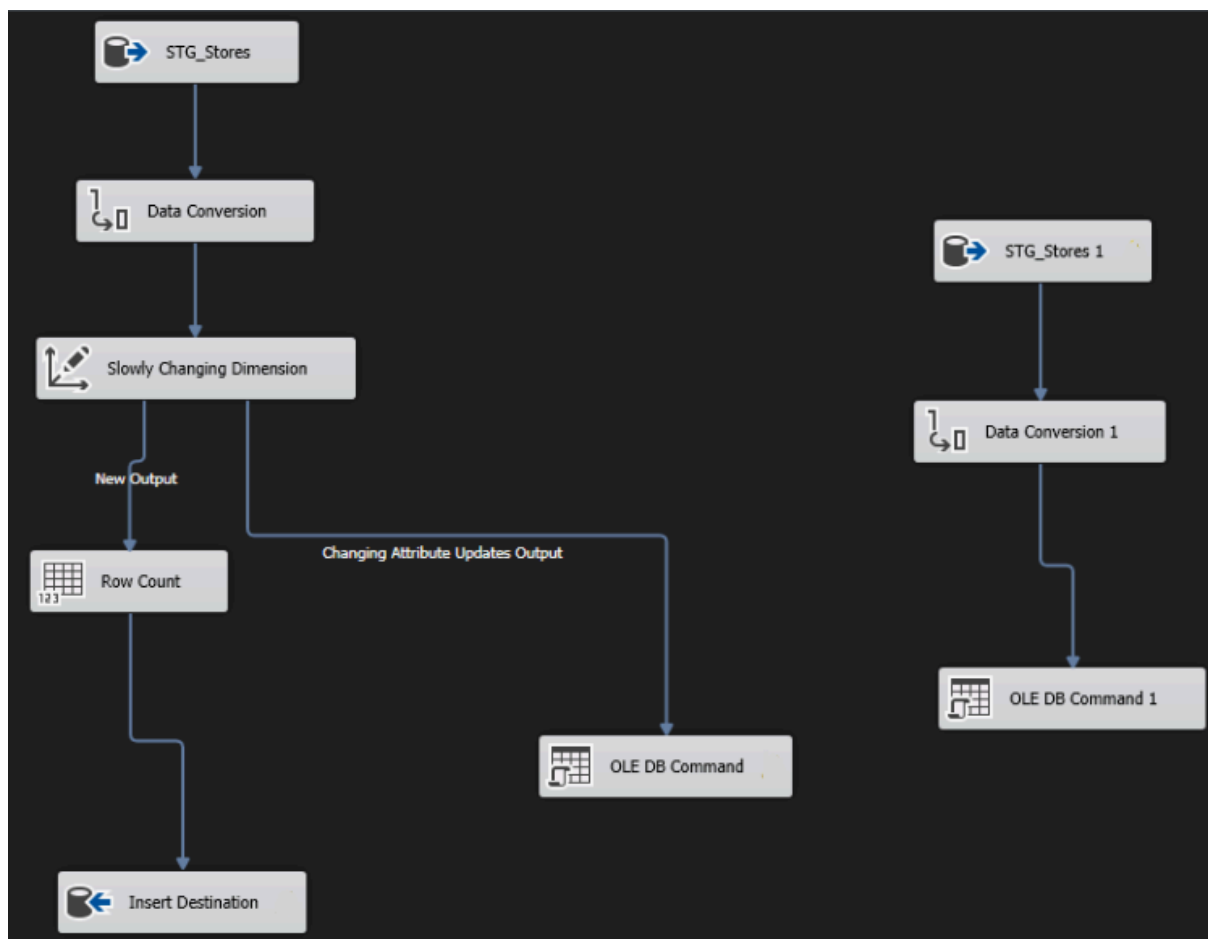
4.1.3.   Stores
STG_Stores

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the stage tables from previously loaded data.
- **Data Flow Tasks**:
    - **OLE DB Source** - The mirror table from the OLAP database.
    - **SQL Query** - To merge the different mirror tables:
      SELECT DISTINCT B.BRANCH StoreID, B.BRANCHNAME Name
      , S.STATENAME Region, CO.COUNTRY Country, D.CITY City
      FROM MRR_BRANCHES B
      LEFT JOIN MRR_CUSTOMERS_FJ C ON B.BRANCH = C.BRANCH
      LEFT JOIN MRR_INVOICES_FJ I ON C.CUST = I.CUST
      LEFT JOIN MRR_DESTCODES D ON I.DESTCODE = D.DESTCODE
      LEFT JOIN MRR_STATES S ON D.STATE = S.STATEID
      LEFT JOIN MRR_COUNTRIES CO ON S.COUNTRY = CO.COUNTRYCODE
    - **Row Count** - Assign the number of records to a variable.
    - **OLE DB Destination** - Stage destination table.

- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

Dim_Stores

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Data Flow Tasks**:
  - **OLE DB Source** - The stage table from the OLAP database.
  - **Data Conversio**n - Changing the different columns to unicode string to fit our destination table.
  - **Slowly Changing Dimension -** Making the split for new and changed records.
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Dimension destination table.
  - **OLE DB Command** - Changed records update.
  - **OLE DB Command 1** - Deleted records update.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.
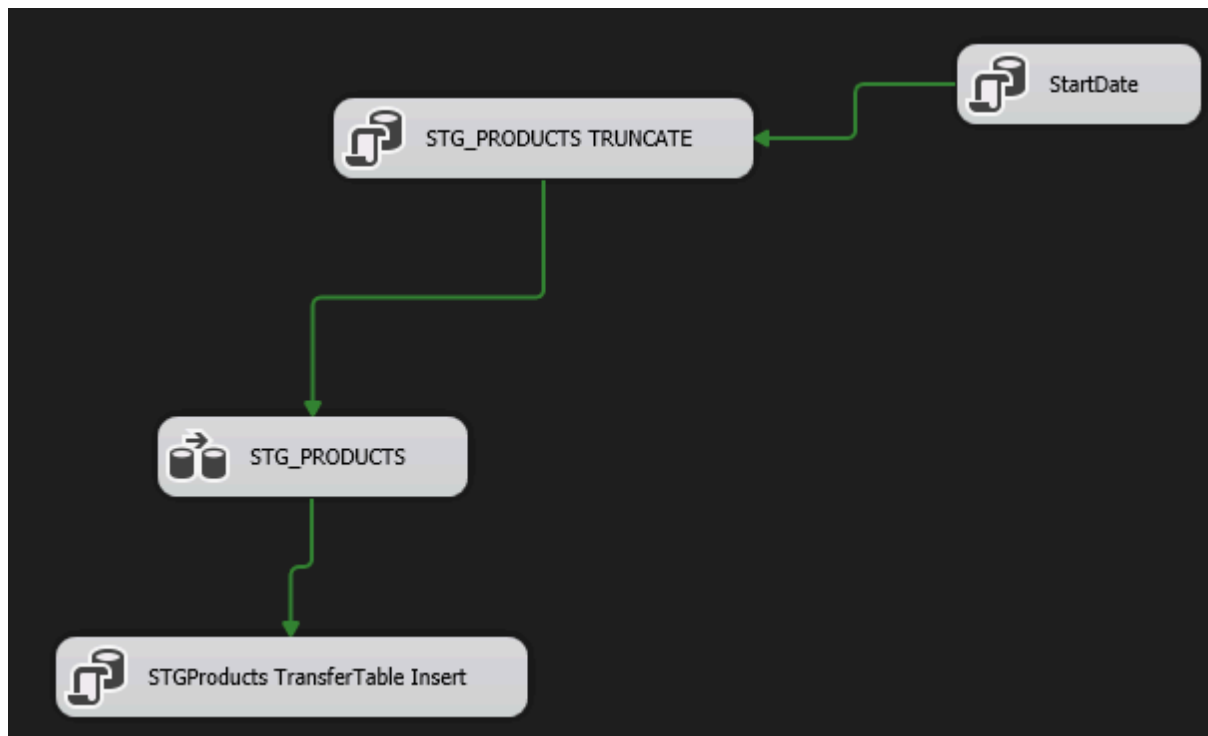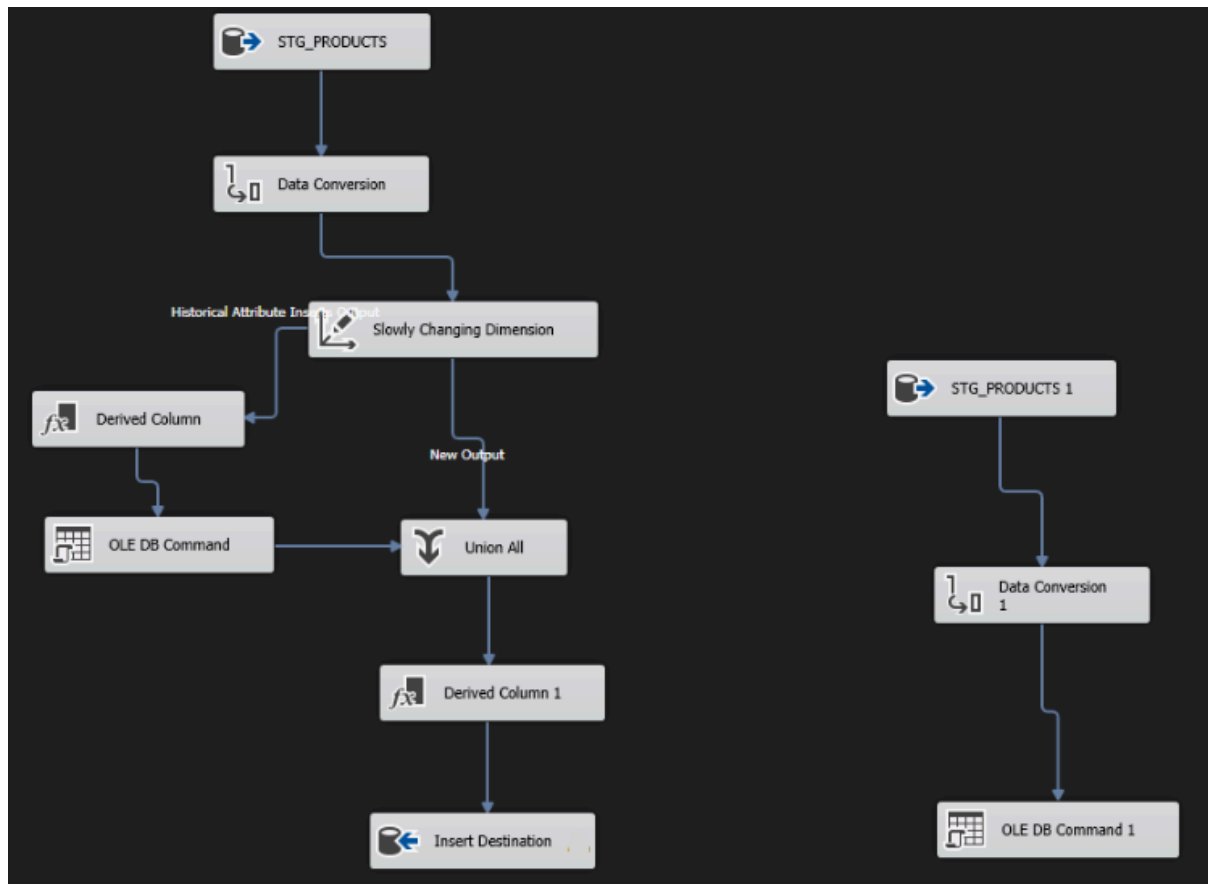
4.1.4. Products
STG_Products

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the stage tables from previously loaded data.
- **Data Flow Tasks**:
    - **OLE DB Source** - The mirror table from the OLAP database.
    - **SQL Query** - To merge the different mirror tables:
      SELECT DISTINCT P.PART ProductID, P.PARTNAME ProductName
      , F.FAMILYNAME SubCategoryName, FT.FTNAME CategoryName
      FROM MRR_PART P
      JOIN MRR_FAMILY F ON P.FAMILY = F.FAMILY
      JOIN MRR_FamilyType FT ON F.FAMILYTYPE = FT.FAMILYTYPE
    - **Row Count** - Assign the number of records to a variable.
    - **OLE DB Destination** - Stage destination table.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

Dim_Products
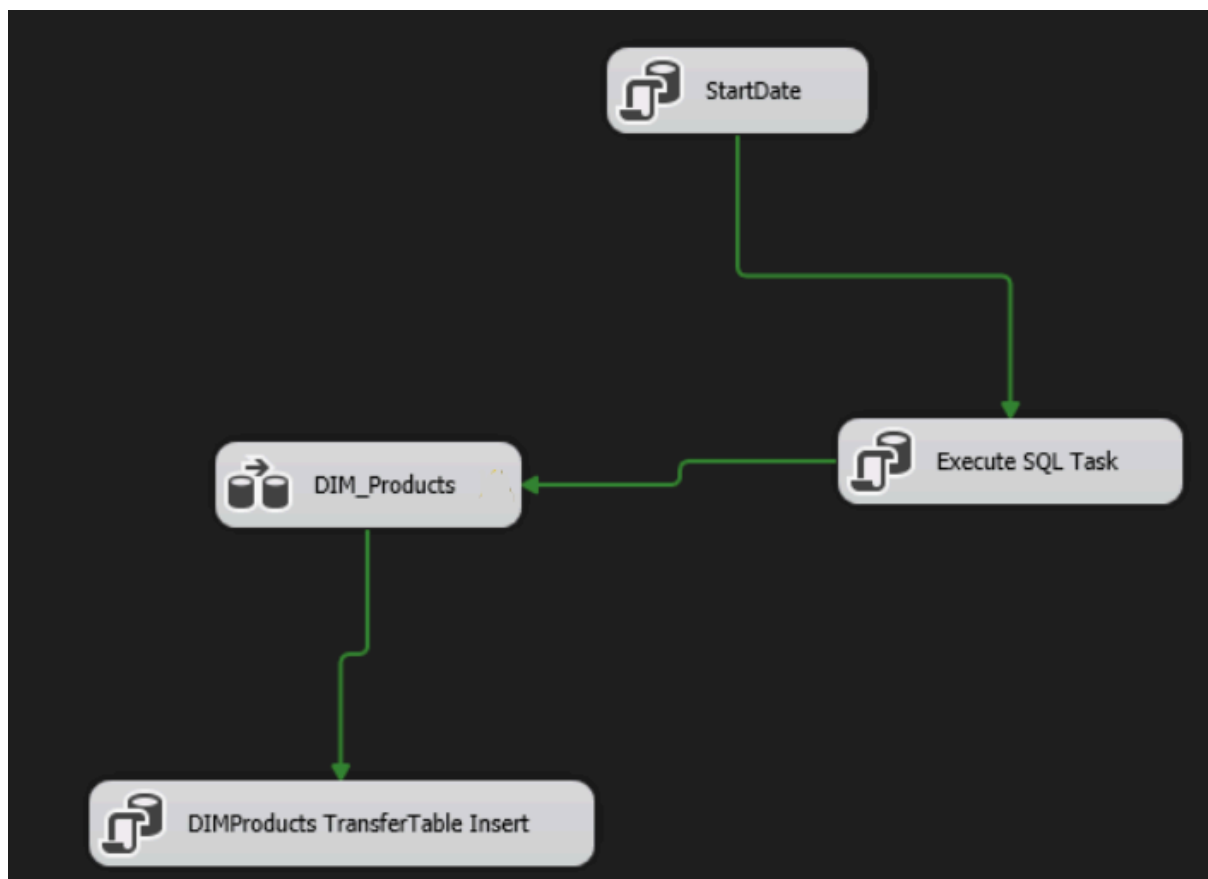
- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Execute SQL Task** - Merge query to update changed and deleted records.
- **Data Flow Tasks**:
    - **OLE DB Source** - The stage table from the OLAP database.
    - **Data Conversio**n - Changing the different columns to unicode string to fit our destination table.
    - **Lookup** - Making sure that we don't load data that is already in the destination table.
    - **Derived Column** - Adding column isActive with the value of "Y".
    - **Conditional Split** - Making a split for the different operations of new, changed and removed records.
    - **Row Count** - Assign the number of records to a variable.
    - **OLE DB Destination** - Dimension destination table.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

Dim_Products_History

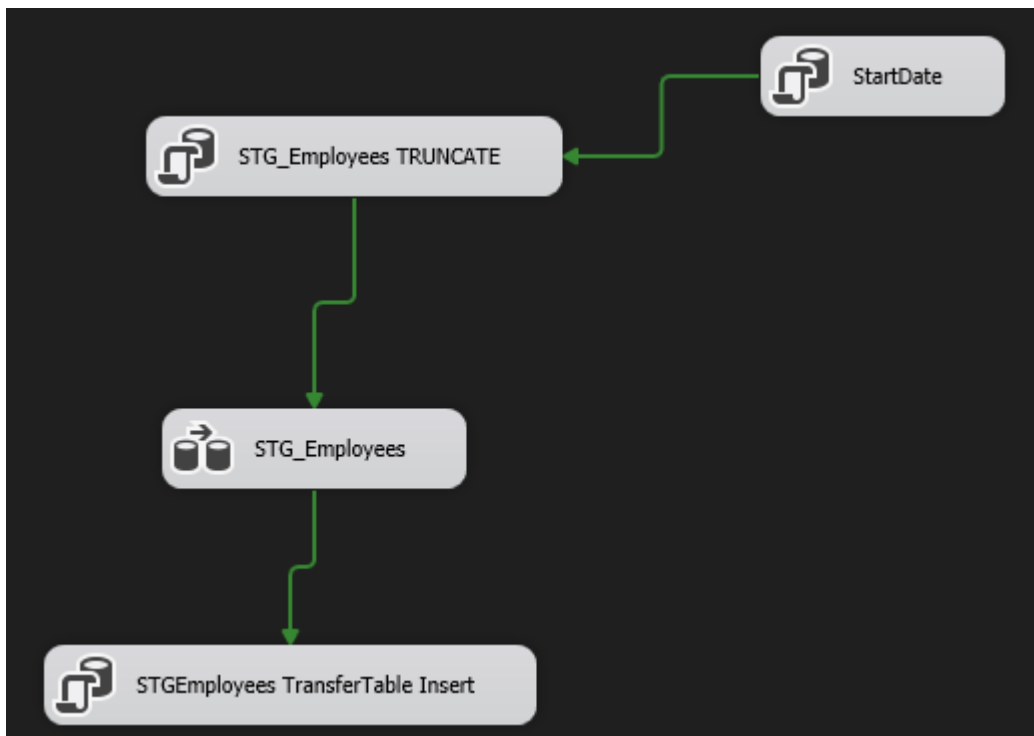- **Data Flow Tasks**:
    - **OLE DB Source** - The stage table from the OLAP database.
    - **Data Conversion** - Changing the different columns to unicode string to fit our destination table.
    - **Slowly Changing Dimension** - New output and Historical Attribute Insert.
    - **Derived Column** - Adding column InsertDate DATETIME.
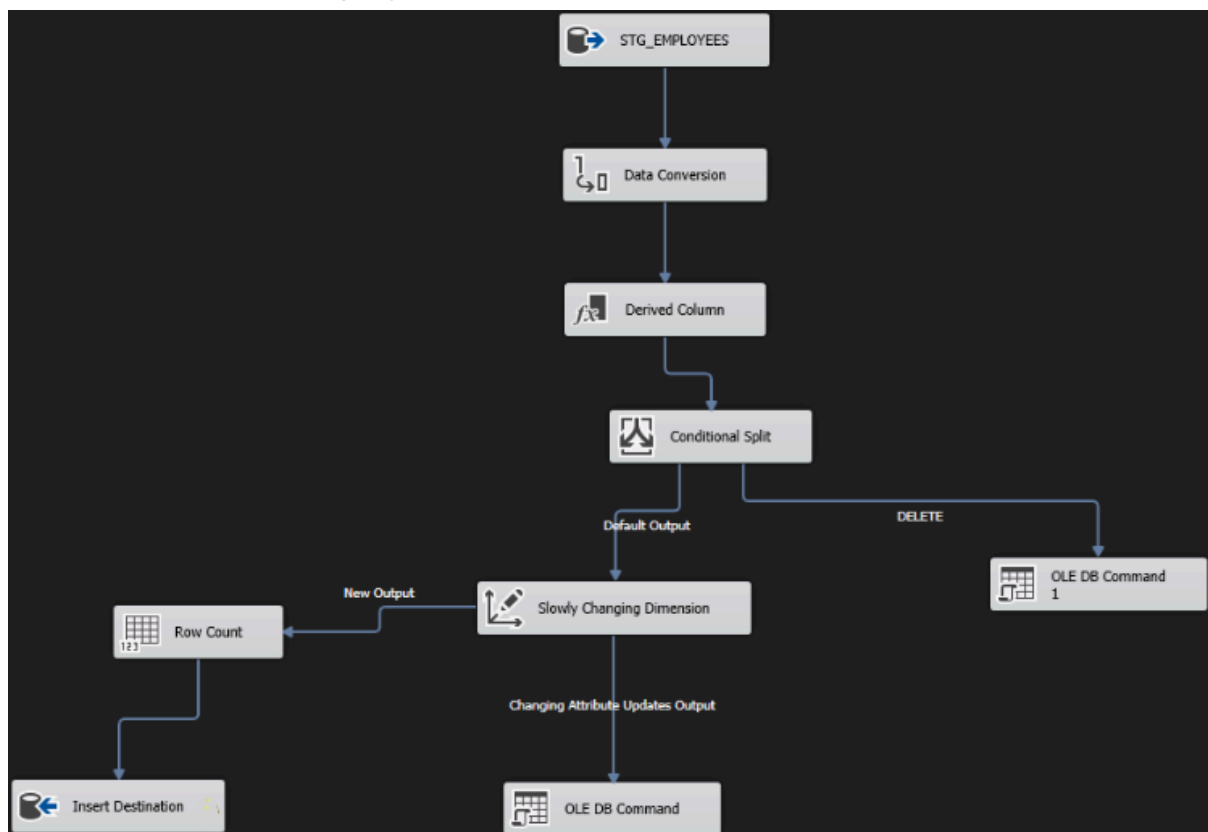   - **OLE DB Destination** - History destination table.

4.1.5. Employees
STG_Employees

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the stage tables from previously loaded data.
- **Data Flow Tasks**:
  - **OLE DB Source** - The mirror table from the OLAP database.
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Stage destination table.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.02.
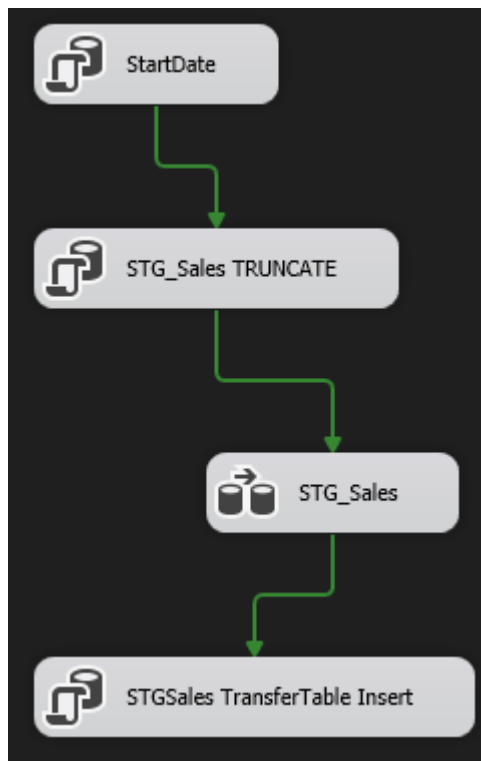
Dim_Employees

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Data Flow Tasks**:
  - **OLE DB Source** - The stage table from the OLAP database.

  - **Data Conversio**n - Changing the different columns to unicode string to fit our destination table.
  - **Derived Column** - Adding column isActive with the value of "Y".
  - **Conditional Split** - Making a split for the different operations of new, changed and removed records.
  - **Slowly Changing Dimension -** Making the split for new and changed records.
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Dimension destination table.
  - **OLE DB Command** - Changed records update.
  - **OLE DB Command 1** - Deleted records update.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

4.1.6. Sales

STG_Sales

- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Truncate Execute SQL Task** - Truncate the stage tables from previously loaded data.
- **Data Flow Tasks**:
  - **OLE DB Source** - The mirror table from the OLAP database.
  - **SQL Query** - To merge the different mirror tables:
    SELECT II.IV OrderID, II.PART ProductID, II.PRICE Price, II.PRICED Discount, II.QUANT Quantity
    , I.IVDATE OrderDate, I.CUST CustomerID, I.AGENT AgentID, I.ZONE TerritoryID
    FROM MRR_INVOICEITEMS II
    JOIN MRR_INVOICES I ON II.IV = I.IV
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Stage destination table.
- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

FACT_Sales
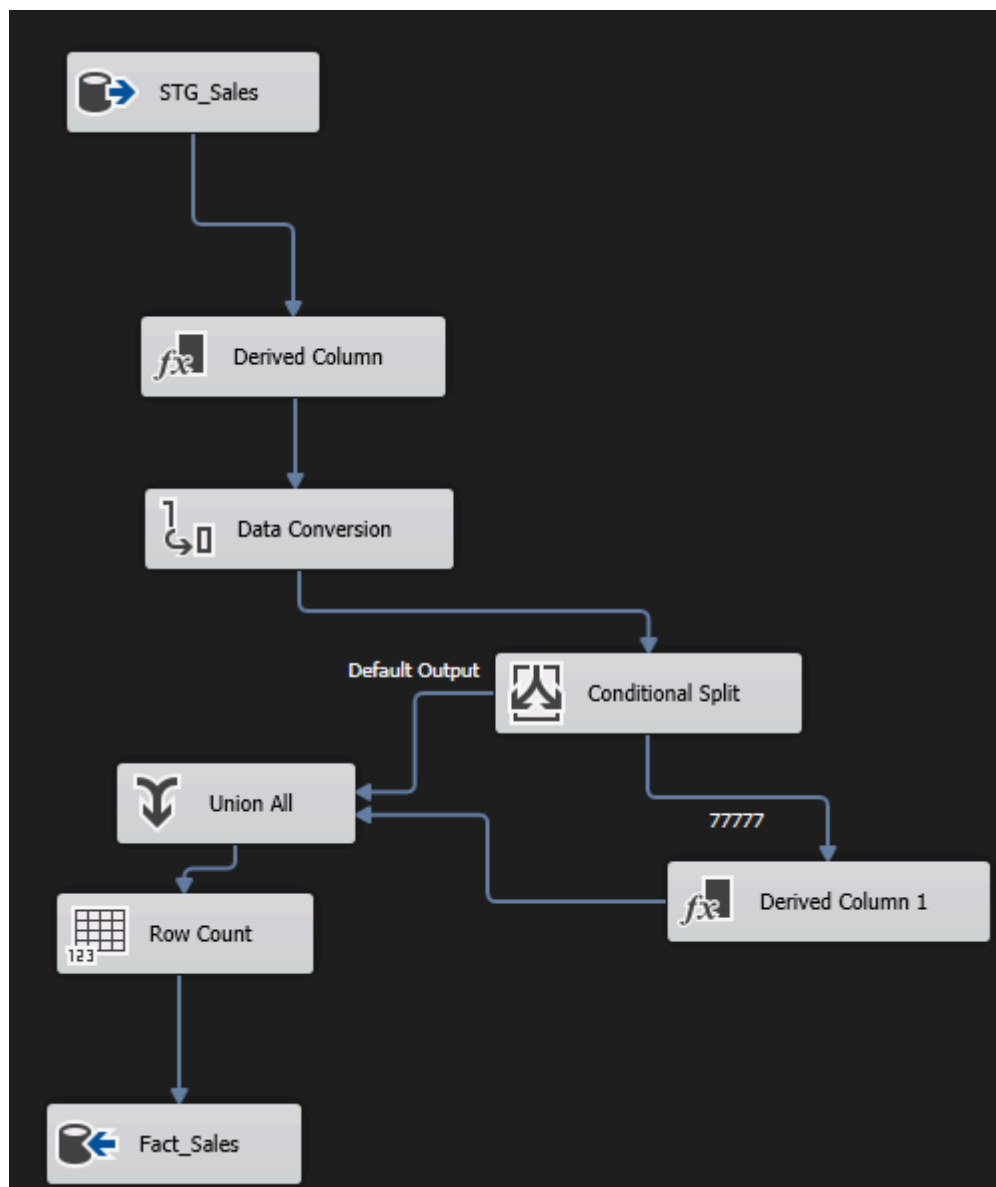
- **StartDate Execute SQL Task** - Assign the current DATETIME to a variable.
- **Data Flow Tasks**:
  - **OLE DB Source** - The stage table from the OLAP database.
  - **Derived Column** - Calculate the calculated column 'Total'.
  - **Data Conversio**n - Changing the OrderDate to DATETIME.
  - **Conditional Split** - Making a split for the Agents with the ID of '77777'.
  - **Derived Column 1** - Changing their ID to 777.
  - **Union All** - Union to the split.
  - **Row Count** - Assign the number of records to a variable.
  - **OLE DB Destination** - Fact destination table.

- **TransferTable Execute SQL Task** - Insert the name of the task, amount of new records, beginning of the process DATETIME and the end of the process DATETIME to the TransferTable.

4.1.7. Jobs and Schedule in SSMS

I've defined the jobs to work in the production environment.

4.2. Tables in the Data Warehouse

4.2.1. FactSales

- **OrderID** - The identifier of an order.
- **OrderDate** - The date of an order.
- **CustomerID** - The identifier of a customer.
- **EmployeeID** - The identifier of an employee.
- **TerritoryID** - The identifier of a territory.
- **ProductID** - The identifier of a product.
- **Quantity** - The amount of a product in the order.
- **Price** - The price of a single product.
- **Discount** - The discount for the product.
- **Total** - The sum of money for the total of a specific product in the order.

4.2.2. DimProducts

- **ProductID** - The identifier of the product.
- **ProductName** - The name of the product.
- **SubCategoryName** - The product's sub-category.
- **CategoryName** - The product's category.
- **IsAvailable** - Indicator for the availability of the product.

4.2.3. DimProductsHistory

- **ProductID** - The identifier of the product.
- **ProductName** - The name of the product.
- **SubCategoryName** - The product's sub-category.
- **CategoryName** - The product's category.
- **InsertDate** - The date time when the record was inserted.
- **EndDate** - The datetime when the following record was inserted.

4.2.4. DimEmployees

- **EmployeeID** - The employee identifier.
- **FirstName** - The employee's first name.
- **LastName** - The employee's last name.
- **JobTitle** - The employee's job title.
- **HireDate** - The employee's hire date.
- **PhoneNumber** - The employee's phone number.
- **EmailAddress** - The employee's email address.
- **TerritoryName** - The employee's territory name

- **isActive** - Flag that says if the employee still works at the company.
- **UpdateDate** - Date when an employee's data has been changed.

4.2.5. DimCustomers

- **CustomerID** - The identifier of a customer.
- **Name** - Customer's full name.
- **Address** - Customer's address.
- **City** - Customer's city.
- **Region** - Customer's region.
- **Country** - Customer's country.
- **StoreID** - The Identifier of a store.
- **IsActive** - Flag that says if the customer made a purchase in the last 3 years.

4.2.6. DimStores

- **StoreID** - The identifier of the store.
- **Name** - The name of the store.
- **Region** - Store's region.
- **Country** - Store's country.
- **IsActive** - Indication for shut down stores.

4.3.   Power BI Visualizations

4.3.1.   Measures

- Average Sales per Order = AVERAGE('Sales Aggregated'[Total])
- Average Sales Per Order Prev Year = CALCULATE(

AVERAGE('Sales Aggregated'[Total]),

SAMEPERIODLASTYEAR('Calendar'[FullDateAlternateKey])

)

- Delta Sales Months = IF(AND([Sum of Sales],[Total Sales Pervious Month]), ([Sum of Sales]-[Total Sales Pervious Month]))
- Last Update Text = DATE(2023,12,31)
- New Customers =

    VAR currentCustomers = VALUES(Sales[CustomerID])

    VAR pastCustomers =

CALCULATETABLE(VALUES(Sales[CustomerID]),ALL('Calendar'[FullDateAlternateKey]),'Calendar'[FullDateAlternateKey] < DATE(max('Calendar'[Year]),1,1))

    VAR newCustomers = EXCEPT(currentCustomers,pastCustomers)

    RETURN

    COUNTROWS(newCustomers)

- Repeat Customer % = [Repeat Customers]/([New Customers] + [Repeat Customers])
- Repeat Customers =

    VAR CurrentCustomers = VALUES(Sales[CustomerID])

    VAR PastCustomers = CALCULATETABLE(VALUES(Sales[CustomerID]),ALL('Calendar'[Month Name],'Calendar'[Month],'Calendar'[Year]),'Calendar'[FullDateAlternateKey] < DATE(max('Calendar'[Year]),1,1))

    VAR RepeatCustomers = INTERSECT(CurrentCustomers,PastCustomers)

    RETURN

    COUNTROWS(RepeatCustomers)

- Sum of Quantity = SUM(Sales[Quantity])
- Sum of Quantity Prev Year = CALCULATE(

SUM('Sales'[Quantity]),

SAMEPERIODLASTYEAR('Calendar'[FullDateAlternateKey])

)

- Sum of Sales = SUM(Sales[Total 80 Times])
- Sum of Sales Prev Year = CALCULATE(

SUM('Sales'[Total 80 Times]),

SAMEPERIODLASTYEAR('Calendar'[FullDateAlternateKey])

)

- Total Sales Pervious Month = CALCULATE([Sum of Sales], DATEADD('Calendar'[FullDateAlternateKey], -1, MONTH))

4.3.2.   Dynamic Titles Measures

- Monthly Revenue Change = "Monthly Revenue Change in " & SELECTEDVALUE('Calendar'[Year], "All Years")
- Top 10 Products by Revenue =

    VAR GetValues= CONCATENATEX(

```
VALUES('Calendar'[Year]), 'Calendar'[Year], ", ")
        VAR GetValues1 = CONCATENATEX(
VALUES('Calendar'[Year]), 'Calendar'[Year]-1, ", ")
        RETURN
        "Top 10 Products by Revenue in " & GetValues & " and " & GetValues1
```

There are more Dynamic Title measures, which are all similar.

4.3.3. Hierarchies

- **Date Hierarchy** - Year, Quarter, Month Name Short, Day.
- **Location Hierarchy** - Country, Region, City.
- **Product Hierarchy** - Category, Sub-Category, Product.

4.3.4. Management Dashboard
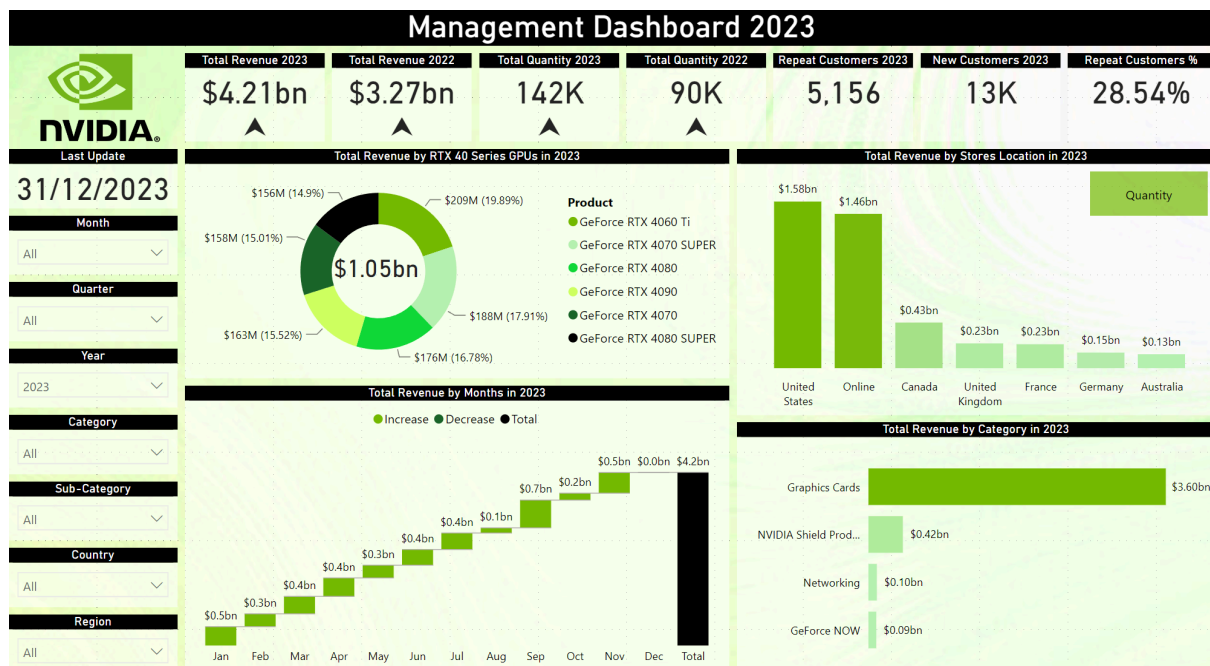
The management dashboard contains:

- Month, Quarter, Year, Category, Sub-Category, Country and Region slicers.
- Total Revenue, Total Quantity, Repeat Customers, New Customers and Repeat Customers % KPI Score Cards.
- Pie Chart of the revenue for the newly released Graphics Cards.
- Column Chart of the sales of the stores.
- Waterfall Chart of the yearly revenue.
- Bar Chart of the revenue from the different categories.
- A Button that changes the different visualizations to Quantity instead of Revenue.

Management Dashboard 2023

Total Revenue 2023	Total Revenue 2022	Total Quantity 2023	Total Quantity 2022	Repeat Customers 2023	New Customers 2023	Repeat Customers %
$4.21bn	$3.27bn	142K	90K	5,156	13K	28.54%

NVIDIA

Last Update
31/12/2023

Month
All

Quarter
All

Year
2023

Category
All

Sub-Category
All

Country
All

Region
All

Total Quantity by RTX 40 Series GPUs in 2023

6,578

- 1,370 (20.83%)
- 954 (14.5%)
- 957 (14.55%)
- 1,192 (18.12%)
- 1,007 (15.31%)
- 1,098 (16.69%)

Product
- GeForce RTX 4060 Ti
- GeForce RTX 4070 SUPER
- GeForce RTX 4080
- GeForce RTX 4090
- GeForce RTX 4080 SUPER
- GeForce RTX 4070

Total Quantity by Stores Location in 2023

Revenue

Online	United States	Canada	United Kingdom	France	Germany	Australia
53.45K	48.20K	15.67K	7.64K	7.16K	5.54K	4.41K

Total Quantity by Months in 2023

● Increase ● Decrease ● Total

Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Total
18.59K	11.55K	14.58K	14.98K	9.67K	11.05K	11.46K	4.29K	22.58K	5.31K	15.88K	2.13K	142.07K

Total Quantity by Category in 2023

Category	Value
GeForce NOW	48.34K
Graphics Cards	37.07K
Networking	36.37K
NVIDIA Shield Prod...	20.29K

4.3.5. Sales Department Report

The sales department report contains:
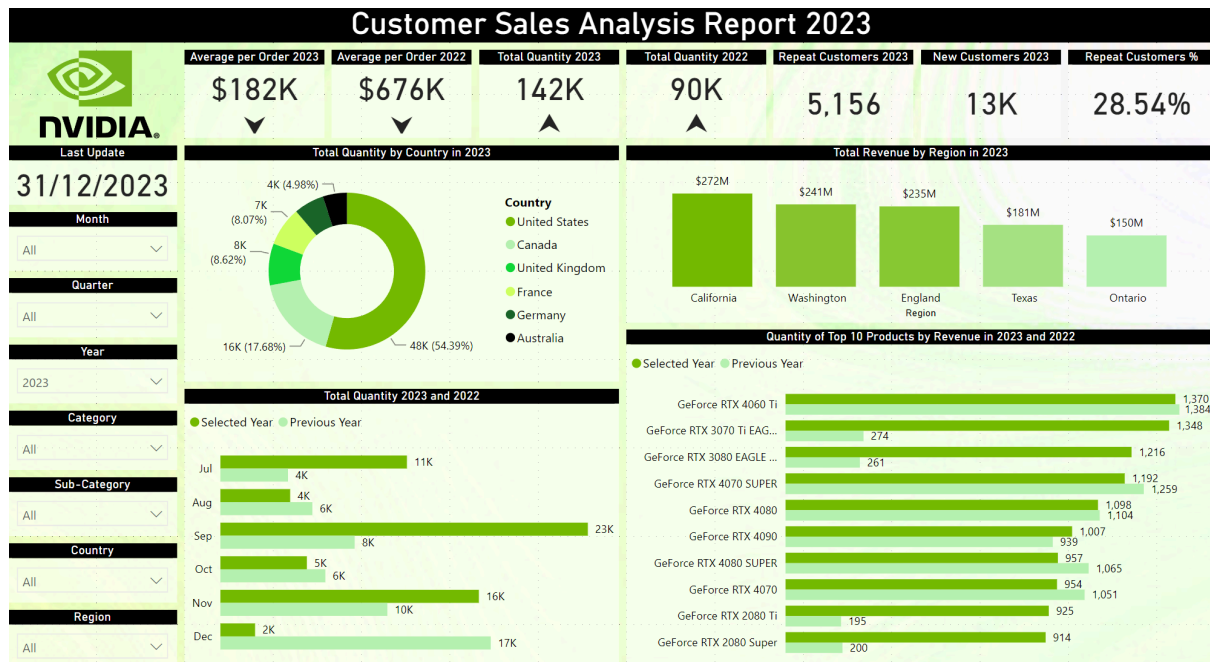- Employee, Average per Order Range, Month, Quarter, Year, Sub-Category, Country and Region slicers.
- Average per Order, Total Revenue, Total Quantity KPI Score Cards.
- Pie Chart of the revenue for the different sub-categories.
- Column Chart of change in the monthly sales.
- Bar Chart of the revenue from the different employees.
- Bar Chart of the revenue from the top 10 products with comparison to previous year.

4.3.6. Customer Sales Analysis Report

The customer sales analysis report contains:
- Month, Quarter, Year, Category, Sub-Category, Country and Region slicers.
- Average per Order, Total Quantity, Repeat Customers, New Customers and Repeat Customers % KPI Score Cards.
- Pie Chart of the quantity that has been bought from different countries.
- Column Chart of the revenue that has been bought from different regions.
- Bar Chart of the quantity that has been bought from each month.
- Bar Chart of the quantity of the top 10 products with comparison to previous year.

4.3.7.   Publish to Power BI Service

◁ Gateway and cloud connections

To use a data gateway, make sure the computer is online and the data source is added in Manage Connections and Gateways. If you're using an On-premises data gateway (standard mode), please select the corresponding data sources and then click apply.

Gateway connections

Use an On-premises or VNet data gateway

◯ On

	Gateway	Department	Contact information	Status	Actions
◉	☁ Personal Gateway			⊘ Running on DESKTOP-KKUDGF3	🗑

Cloud connections

No cloud connections

Apply   Discard

◁ Data source credentials

PROD_Sales_DM_NVIDIA-DESKTOP-KKUDGF3   Edit credentials   Show in lineage view ⧉

▷ Parameters

◁ Refresh

Configure a refresh schedule

Define a data refresh schedule to import data from the data source into the semantic model. Learn more

🔵 On

Refresh frequency

Daily ⌄

Time zone

(UTC+02:00) Jerusalem ⌄

Time

| 4 ⌄ | 00 ⌄ | AM ⌄ | ✕ |

Add another time

Send refresh failure notifications to

☑ Semantic model owner

☐ These contacts:

Enter email addresses

Apply   Discard