NVIDIA

BI System Specification Document

Version 1.0

Janan Ghanadri

Content

1. General

1. Project Objective

The main goal of this project is to create a comprehensive Business Intelligence (BI) solution using data from the PriorityERP Database for Nvidia, a prominent American technology company recognized for its graphics processing units (GPUs) and system-on-a-chip units (SoCs).

The project focuses on developing a detailed BI system that includes consolidated data tables, particularly emphasizing sales data, customer details, employee records, product information, and store-related data.

Moreover, the project aims to present data visually through dashboards and reporting mechanisms tailored to meet the needs of Nvidia's management, department heads, and sales managers. The objective is to provide valuable insights into customer preferences, buying patterns, popular products, and the top-performing stores. By analyzing this information, the BI solution seeks to assist Nvidia in refining product development strategies for optimal results.

1.2. Project Contents

In this project, I will build a Data Mart that will contain information about sales data of Nvidia company.

- Data Cleaning and Preparation: Prior to analysis, we will need to perform thorough data cleaning and preparation to ensure their quality and consistency.

- Main summary tables to be built for the company's needs :
  - FactSales – Information about all the orders. Data loading process for this table will be incremental
  - Dim_Products – Information about the products sold by the company.
  - Dim_Customers – Information about the company's customers
  - Dim_Employees – Information about the company's employees.
  - Dim_Stores – Information about the stores that sell the company's products.
  - DimProductsHistory – Provides information about changes in Dim Products.
  - Transfertable – Provides information about each package start/end time and number of rows that entered the table.

  Source To Target Link

- I will also build reports that contain measure for the company and customers' usage to achieve the project's main goal.

1. Sales Department Analysis:

The Sales Department will analyze sales data to evaluate sales performance, recognize product category trends, customize sales strategies for specific countries, and assess goal achievement by comparing current results with the previous year. Top 5 products will be acknowledged and incentivized. Special attention will be given to the comparison between online sales and physical stores sales. There they can see the amount of purchases, the average per order and more.

2. Customer Department Analysis:

The Customer Department aims to provide the data on the customers purchases and satisfaction. The customers department can access information such as total sales, number of customers by country and period, customers sales average by country, number of online and physical store customers. This empowers customers department to make informed decisions, discover desired products in specific countries, and compare their purchasing behavior with other regions.

3. Executive Dashboard:

The Dashboard will contain measures from both departments. The information provided will increase the overall understanding of the sales movement in the company.

2. work plan, distribution of tasks and schedules for execution :

GANTT

3. Technical Specifications:

3.1. Perquisites:

• SQL Server: ERP system in the operational DB (PriorityERP) tables, data (SQL files)
• SSIS: ETL processes using SSIS in Visual Studio
• Data refresh processes through the definition of JOBS in SSMS
• Power BI: Creating reports and dashboards using Power B

3.2. Solution Architecture

HLD:

The report for the Customer Department consists of:

1. Number of customers by period.
2. Number of customers who made purchases in each country.
3. Average of customers purchases by country.
4. Customers who purchased online vs. in physical stores.
5. Top 5 Customers.

The report for the Sales Department consists of:

1. Sales conversion rate by country.
2. Sales Amount by category.
3. Sales performance trend over time (Sales growth).
4. Top 5 products
5. Comparison between Online and Physical stores.

4. Functional Specification

4.1. Creation of final Source to Target and ERD models.

4.1.1 Source to Target

Source To Target Link

4.1.2 ERD model

ERD Link

4.2. Detailed description of all ETL processes:

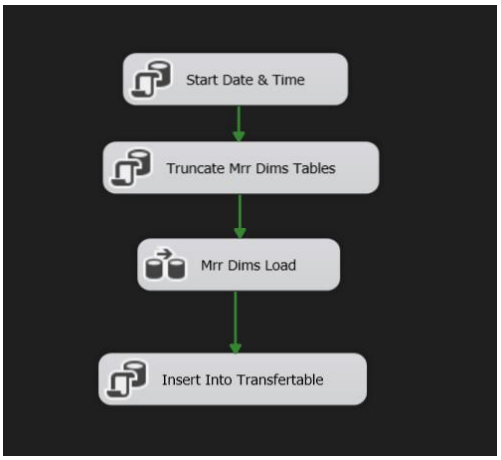
The document describes the ETL processes for creating a SnowFlake-modeled Data Mart through the implementation of three stages; MRR, STG tables, Dim and FactSales tables.

- The Project Contains 6 solutions, 13 packages.

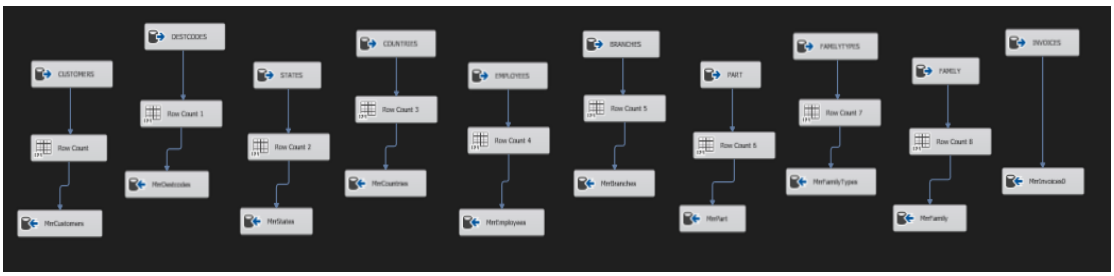
1. MRR Dims :

   In this package happens the extraction of the data from the PriorityERP database into the MRR tables.

   First we Truncate the Mrr Tables then we extract the fields we want to extract for our usage.



   Data flow:



   From the part table we take only products that has participated in sales in the last 2 years before the last sales by checking the maximum order date in the fact sales.

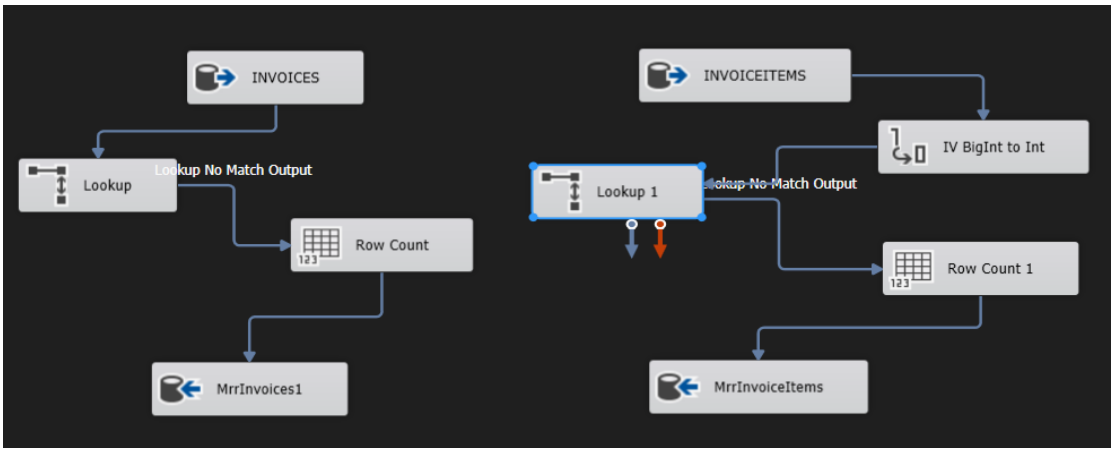
   We also want to insert the number of rows that entered each table into table Transfertable.

   The count is calculated by using the Row Count tool each count is put in a variable, then inserted into the table with the run start time of the package and the end time, with an EXECUTE SQL TASK.

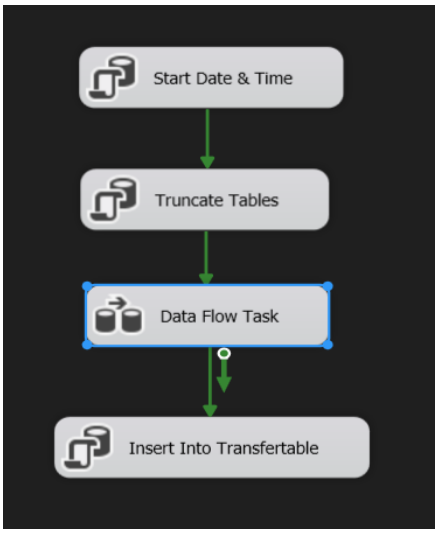
2. Mrr Sales :

In this package happens the load from the sales tables, INVOICES and INVOICEITEMS into the MrrINVOICES and MrrINVOICEITEMS.

In the initial insertion, we load all the values. For each subsequent run, we only insert records that do not exist in our destination table (FactSales table) using LOOKUP.

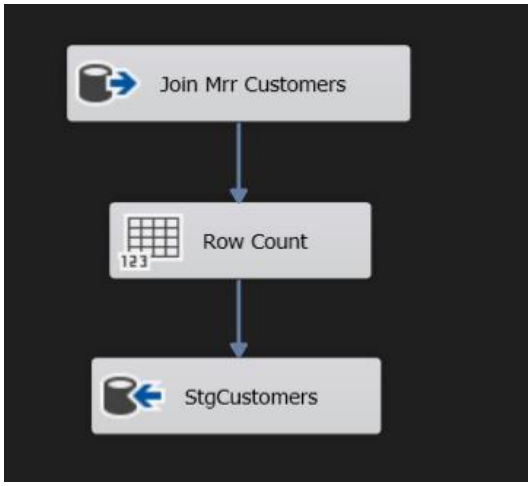


We also insert the count of rows and the start,end date and time into table Transfertable.



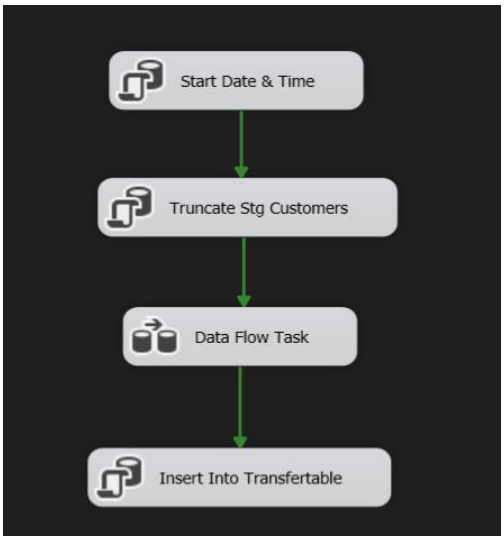
3. Stg Customers :

   In this package we join the customers related table to create the StgCustomers table, MrrCUSTOMERS, MrrDESTCODES, MrrSTATES, MrrCOUNTRIES, MrrINVOICES, in order to get the proper fields.



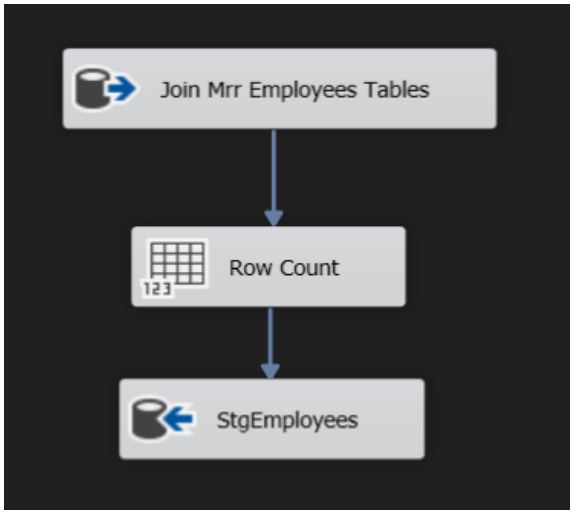
   We also insert to Transfertable table the row count and the start,end date and time of the package.

   In Control flow:



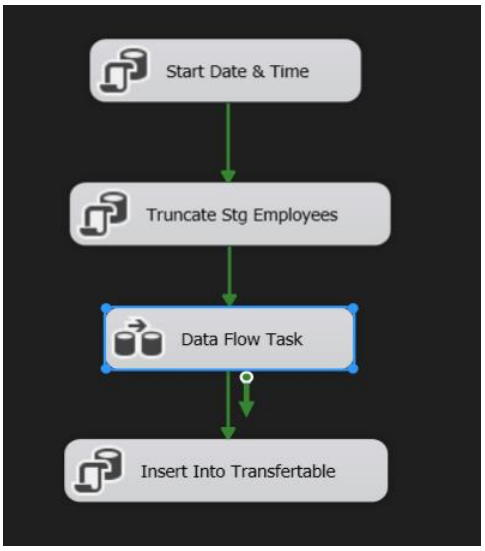
4. Stg Employees :

   In this package we join the Employees related table to create the StgEmployees table, MrrEMPLOYEES, MrrINVOICES, in order to get the proper fields.



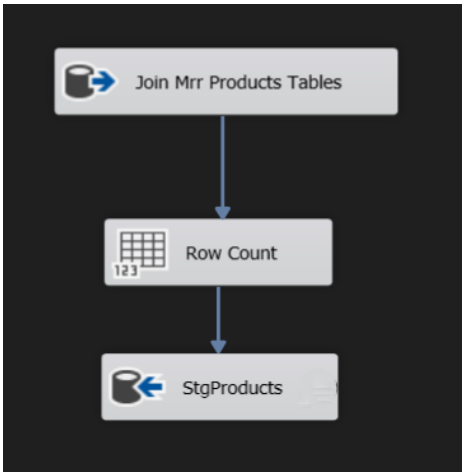
   We also insert to Transfertable table the row count and the start,end date and time of the package.

   In Control flow:



5. Stg Products :

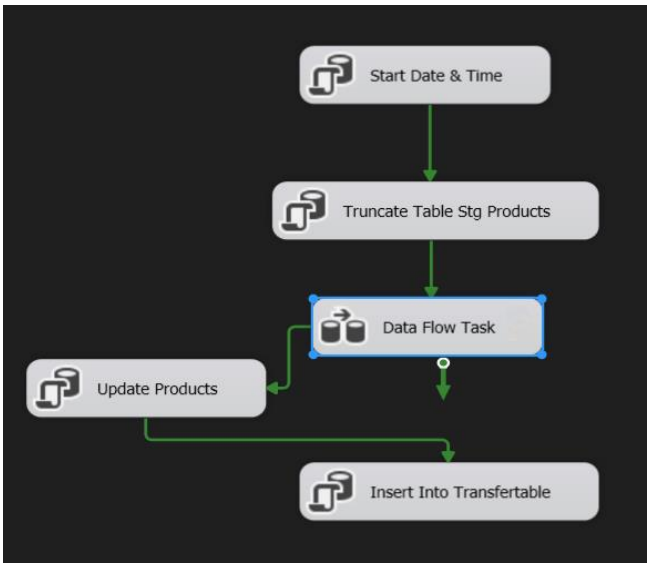
In this package we join the Products related table to create the StgProducts table, MrrPART, MrrFAMILY ,MrrFAMILYTYPES , in order to get the proper fields.



We also insert to Transfertable table the row count and the start,end date and time of the package.

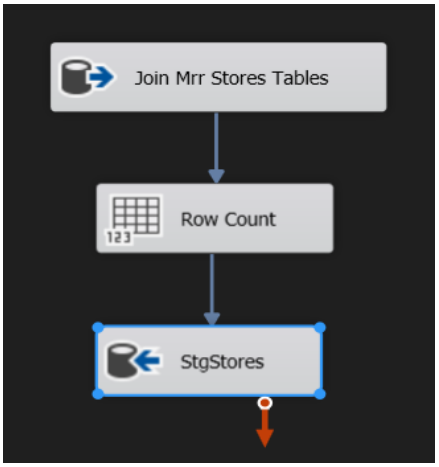
NOTICE : Before inserting into Transfertable I updated the StgProducts with the proper names and categories of the Nvidia products with an SQL Task.

In Control flow:



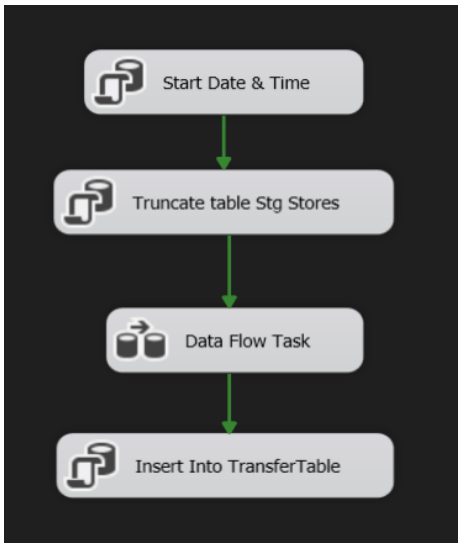
6. Stg Stores :

In this package we join the Stores related tables to create the StgStores table, MrrBRANCHES, MrrINVOICES, MrrSTATES, MrrDESTCODES and MrrCUSTOMERS in order to extract the proper fields out of them.



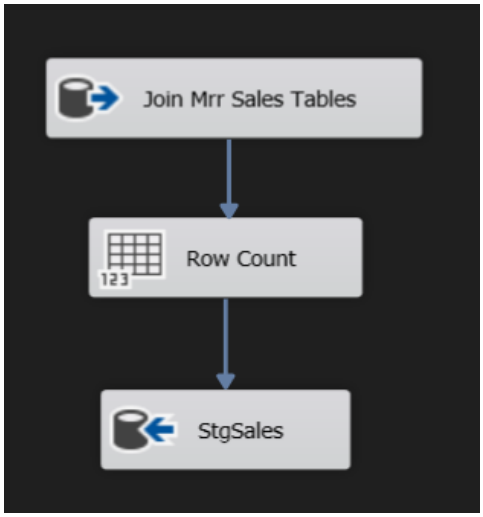
We also insert to Transfertable table the row count and the start,end date and time of the package.

In Control flow:



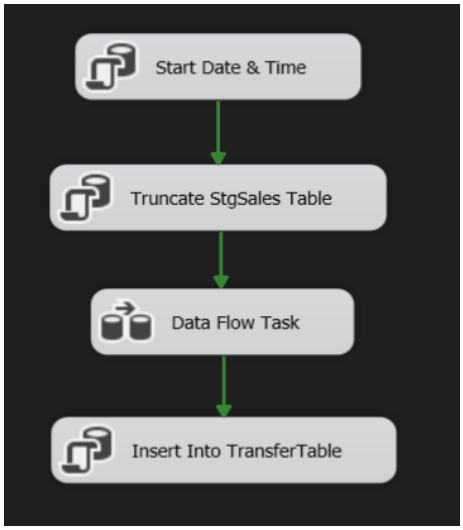
7. Stg Sales :

In this package we join the Sales related tables to create the StgSales table, MrrINVOICES, MrrINVOICEITEMS in order to extract the proper fields out of them.



We also insert to Transfertable table the row count and the start,end date and time of the package.

In Control flow:

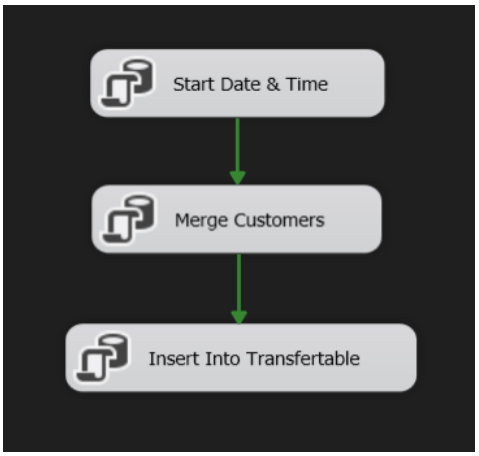


8. Dim Customers:

In this package we keep the DimCustomers table synchronized with the data in the StgCustomers table by using the merge method.

1. If a row in StgCustomers has a CustomerID that doesn't exist in DimCustomers, a new row is inserted into DimCustomers with values from the corresponding columns in StgCustomers.
2. If a row in StgCustomers matches a row in DimCustomers based on CustomerID, and there are differences in certain columns (Name, StoreID, Address, City, Region, Country), the row in DimCustomers is updated with values from the corresponding columns in StgCustomers.
3. If a row exists in DimCustomers but not in StgCustomers, this means that the row has been deleted, so the IsActive column in DimCustomers is set to 'N', and the UpdateDate column is updated to the current date and time.

The UpdateDate column is updated with the current date each time the row has changes.
The SELECT @@rowcount AS rownums; statement returns the number of rows affected by the merge operation.



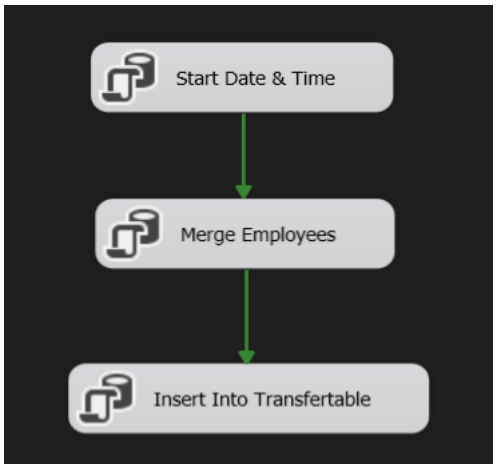
We also insert to Transfertable table the row count and the start,end date and time of the package.

9. Dim Employees:

In this package, we ensure that the DimEmployees table reflects the current state of the StgEmployees table by handling insertions, updates, and marking deletions as inactive using Merge method.

1. If a row in StgEmployees has an EmployeeID that doesn't exist in DimEmployees, a new row is inserted into DimEmployees with values from the corresponding columns in StgEmployees.
2. If a row in StgEmployees matches a row in DimEmployees based on EmployeeID, and there are differences in certain columns (FirstName, LastName, JobTitle, HireDate, PhoneNumber, EmailAddress, TerritoryName), the row in DimEmployees is updated with values from the corresponding columns in StgEmployees.
3. If a row exists in DimEmployees but not in StgEmployees, indicating that the row has been deleted, the IsActive column in DimEmployees is set to 'N'.

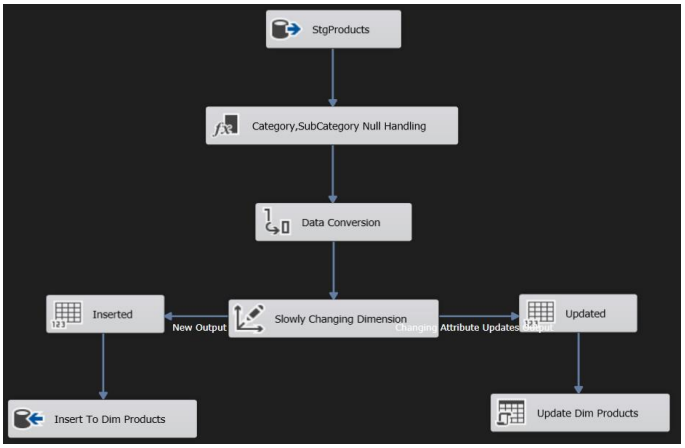
The UpdateDate column is updated with the current date each time the row has changes.
The SELECT @@rowcount AS rownums; statement returns the number of rows affected by the merge operation.



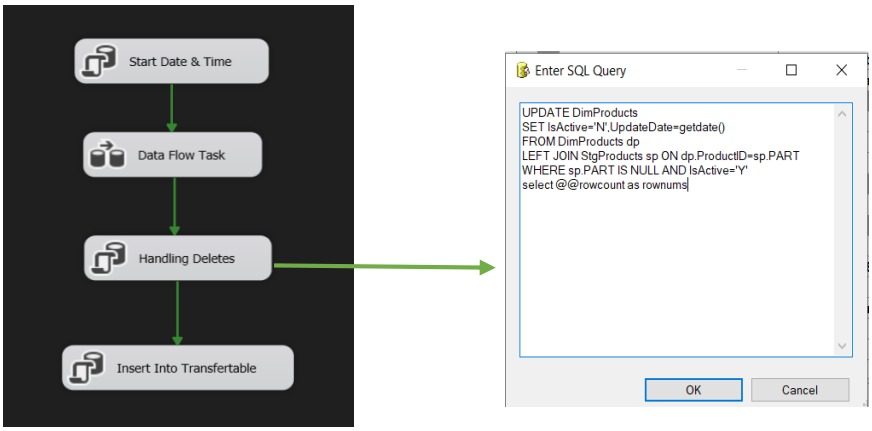
We also insert into Transfertable table the row count and the start,end date and time of the package.

10. Dim Products

This package employs SCD techniques to synchronize `DimProducts` with `StgProducts`. New rows are inserted when a `ProductID` in `StgProducts` is absent in `DimProducts`. Existing rows are updated when a matching `ProductID` has differences in specified columns.
Also, with aderived column, we update the Category and SubCategory field to 'Unknown' If it includes a Null value .

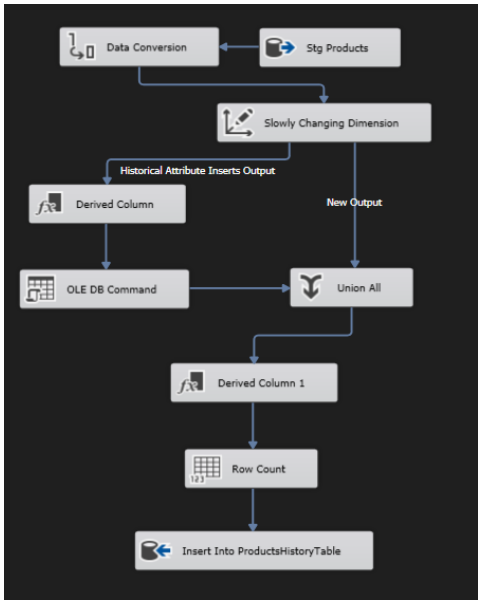


Handling deleted rows happens in the control flow, where we check if a ProductID exists in DimProducts but is absent from StgProducts, in this case we update the product to be Inactive by setting IsActive field to 'N' and get the current date to be the UpdateDate.
Then insert into Transfertable the row count and the start,end date and time of the package.

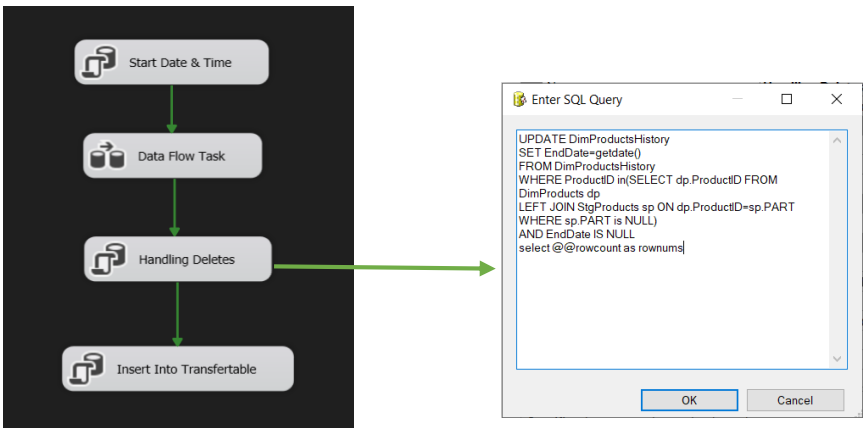


11. Dim Products History

This package employs SCD techniques to save the updates that happened in `DimProducts`. New rows are inserted when a `ProductID` in `StgProducts` is absent in `DimProducts`. When existing rows are updated with a differences specified columns values, a new row with the new values is inserted into the DimProductsHistoryTable.

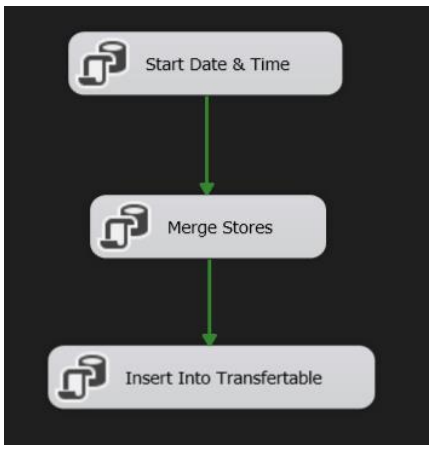


Handling deleted rows happens in the control flow, where we check if a ProductID exists in DimProducts but is absent from StgProducts and the EndDate in DimProductsHistory is Null (so we don't update it twice), in this case we update the product to be Inactive by setting EndDate field to current date
Then insert into Transfertable the row count and the start,end date and time of the package.



12. Dim Stores:

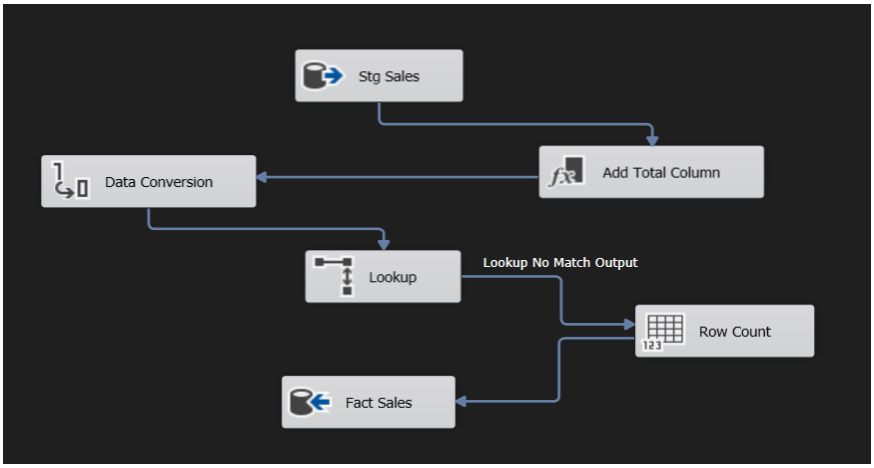
This package synchronizes the DimStores table with StgStores using the merge method. It inserts new rows or updates existing ones based on StoreID matches. If a row in StgStores doesn't exist in DimStores, a new row is inserted; if there are differences in StoreName or Location, the row is updated. Deleted rows are marked as inactive. The SELECT @@rowcount AS rownums; statement returns the affected row count.



Then the row count and the start,end date and time of the package are inserted into Transfertable.

13. Fact Sales:

In this package we build the Fact Sales table, which contains all the details of a sale in the Nvidia stores. Using the lookup transform we only insert the new rows added into the StgSales table by checking the OrderID matches.



Total: computed using the derived column.

OrderQty * UnitPrice * (1 - UnitPriceDiscount) * 1.17

This package includes data conversion, for example: OrderDate is converted to DATE with two numbers after the decimal point.

In the end the row count and the start,end date and time of the package are inserted into Transfertable.

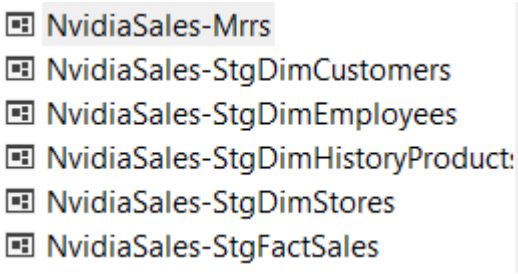
❖ Production – Development environment:

Once the SSIS process is completed, we replicate the database into a new one, essentially creating a duplicate that we can manipulate separately. This duplication allows us to perform various operations and analyses on the data without affecting the original database.
This is done using the task 'Transfer Sql Server Objects Task'.

❖ SQL Server agent:

I created two environments in SQL server agent, one environment for each database, in order to choose on which environment to run my project.
Then, comes the stage where we should deploy the project to SQL server agent.

After deploying the project:
I created 6 jobs that will run my solutions in order. I had one job for each solution.



Every job contains the solution's packages in order, and an additional step which was created to run the next job, this helps to run the solutions one after another in ONE CLICK only.

The package running order:
MrrDims ➡ MrrSales ➡ StgStores ➡ DimStores ➡ StgEmployees ➡ DimEmployees ➡ StgCustomers ➡ DimCustomers ➡ StgProducts ➡ DimProducts ➡ DimProductsHistory ➡ StgSales ➡ FactSales.

The next step was scheduling the jobs:

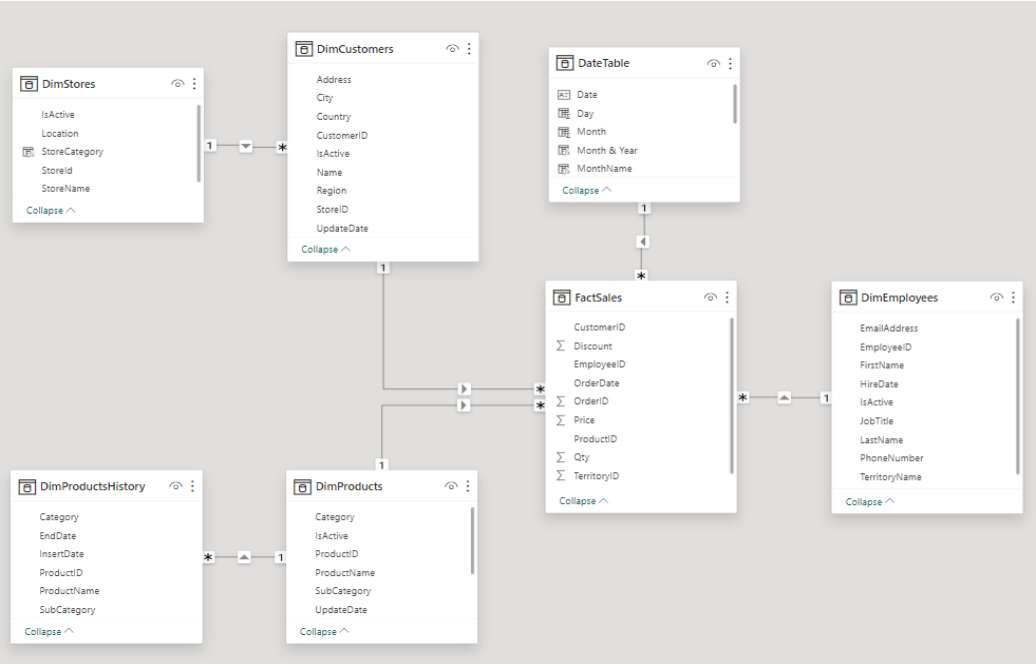
ID	Name	Enabl...	Description
1020	NvidiaDailyMrr	Yes	Occurs every day at 8:00:00 AM. Schedule will be ...

I scheduled the NvidiaDailyMrr Job (which contains the first two packages; MrrDims & MrrSales), to run daily at 8AM.
The last step at every job helps the job to go to the next one after completing, therefore I only scheduled the firestone and the rest will follow automatically.

4.3. Specification of measures and tables for reports in PowerBI:

After building the Data Mart, we will generate reports for the sales department managers in order to draw conclusions from the data and thereby lead to organizational change that supports the organization's strategy and increases company profits. The reporting system is built around a central dashboard for Nvidia's sales department and two secondary reports that analyze data for the company's customers' department and sales.

❖ SnowFlake DataMart :



❖ Measures:

In DAX Syntax

- Sales Amount = SUM(FactSales[Total])
- Sales Amount Pervious Month = CALCULATE([Sales Amount], DATEADD('DateTable'[Date], -1, MONTH))
- Delta Sales Month = IF(AND([Sales Amount],[Sales Amount Pervious Month]), 1-([Sales Amount Pervious Month])/[Sales Amount])
- Sales Growth = DIVIDE([Sales Amount]- [LY Sales],[LY Sales])
- LY Sales = CALCULATE([Sales Amount], SAMEPERIODLASTYEAR(DateTable[Date]))
- Total Orders = DISTINCTCOUNT(FactSales[OrderID])
- Total Quantity = SUM(FactSales[Qty])
- TotalCustomersWithPurchases = CALCULATE(DISTINCTCOUNT(DimCustomers[CustomerID]), FILTER(VALUES(DimCustomers[CustomerID]), CALCULATE(COUNTROWS(FactSales), USERELATIONSHIP(FactSales[OrderDate], DateTable[Date])) > 0))
- NvidiaOnlineStoresCustomers = CALCULATE(DISTINCTCOUNT(DimCustomers[CustomerID]), DimCustomers[StoreID] = 1995)
- NvidiaStoresCustomers=

  CALCULATE(DISTINCTCOUNT(DimCustomers[CustomerID]), DimCustomers[StoreID] <> 1995)
- Avg Sale per Customer = DIVIDE([Sales Amount], [TotalCustomersWithPurchases])
- Avg Sales Per Order = DIVIDE([Sales Amount],[Total Orders])
- YTD Sales = TOTALYTD([Sales Amount],'DateTable'[Date])
- Sales Trend KPI =

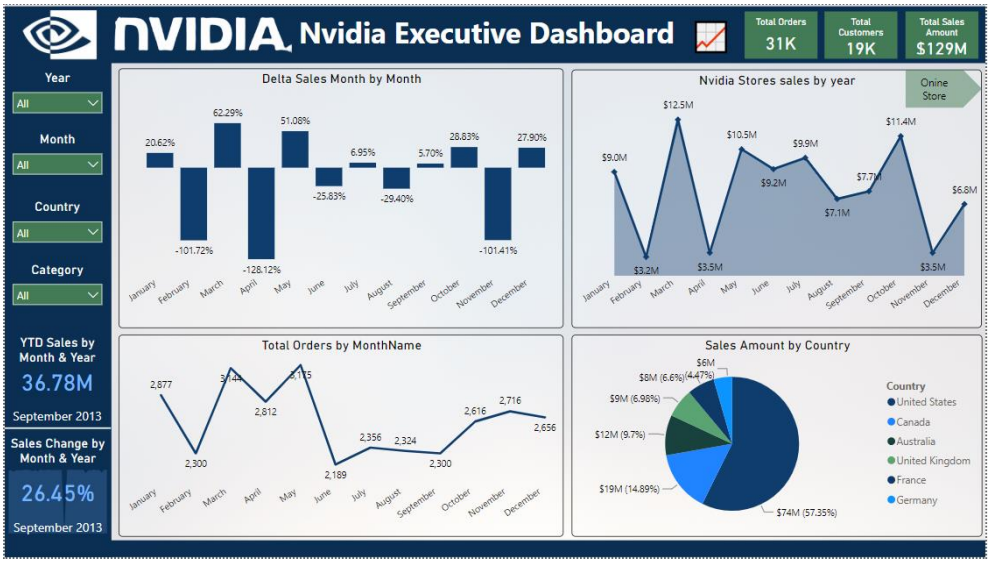
```
VAR ChartIncrease = UNICHAR(128200)
VAR ChartDecrease = UNICHAR(128201)
VAR SixMonthTrend = [delta sales month]
RETURN

SWITCH(TRUE(),
SixMonthTrend>0,ChartIncrease,SixMonthTrend<0,ChartDecrease)
```

❖ Dashboard:

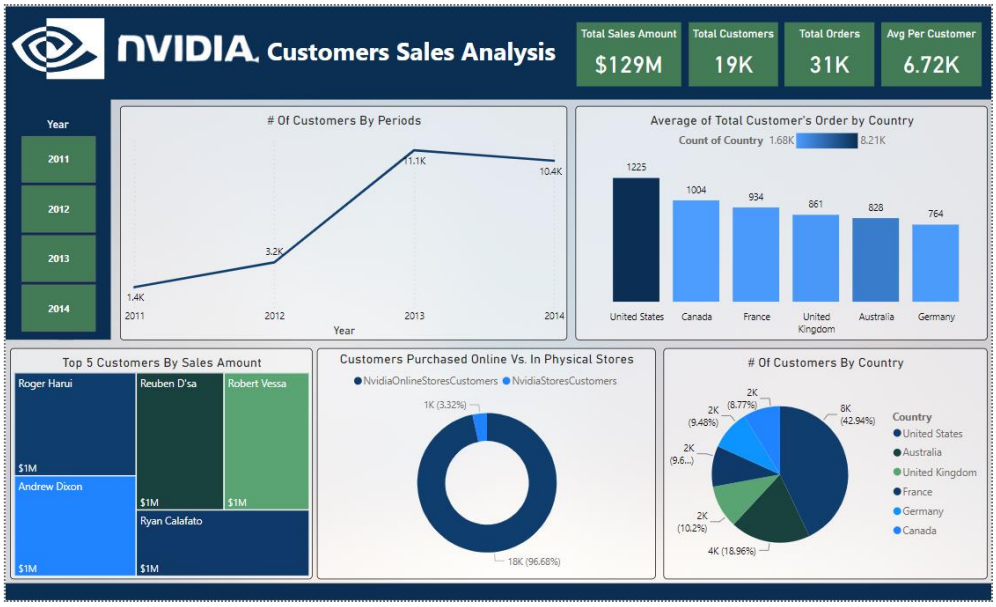
Slicers : Year, Month, Country and Category, which allows selection of the desired content.

Buttons : Online Stores and Back buttons to move between the graph of Nvidia online stores sales and Nvidia physical stores sale, which are separated with two bookmarks, one for every visual.

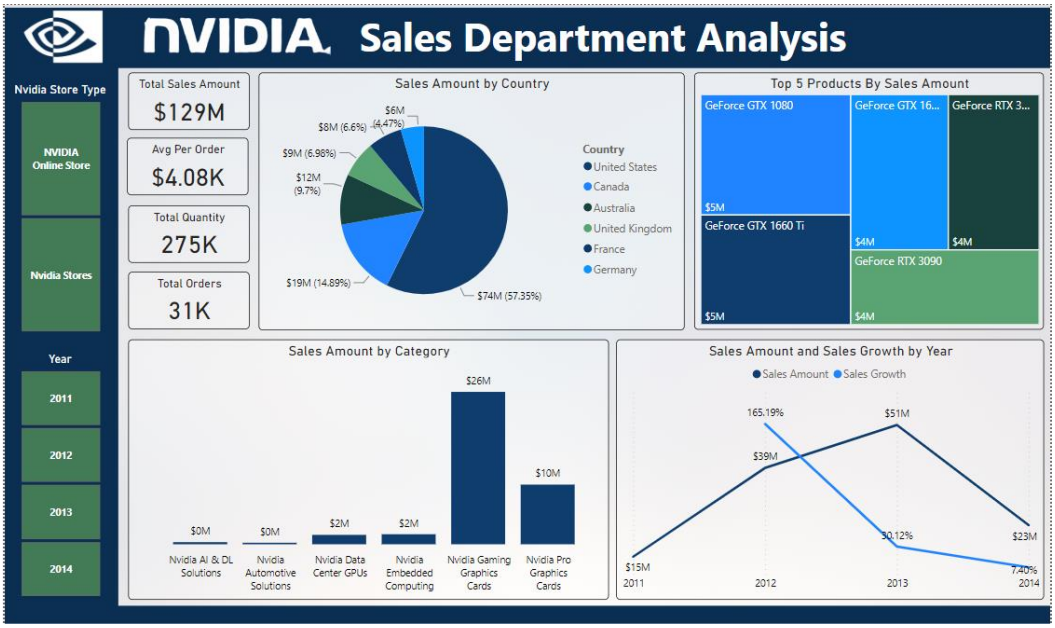
Orientation KPI : Shows the directions of the sales, are they going up or down in the selected period, category and country.



❖ Customers Department Analysis Report:



❖ Sales Department Analysis Report



4.4. Appendix

After creating the reports, I uploaded them to the Power BI Service Workspace. Then, I made the Nvidia App, which includes both the reports and the dashboard. I also set up a daily report refresh for 9 am. This ensures that the data stays up-to-date without manual intervention.
I also created a GitHub Repository that contains the project specification document, the power BI reports and the summary of my work.

App Link : NvidiaSalesDataMartApp

GitHub Repository link: NvidiaDataMart-EXPERIS