Western Digital

SALES DATA MART

BI System Specifications Document

Date: 01/08/2024
Version: 2.0
Written by: Odelia Hochman

Content

1. General

This project's objective is to create a full BI solution for Western Digital's sales department, to support Western Digital's growth in device sales. The project was designed according to Western Digital's sales department KPIs.

Western Digital Corporation (WDC) is an American company known for designing, manufacturing, and selling data storage solutions. They offer a wide range of products such as hard disk drives (HDDs), solid-state drives (SSDs), and data center systems. Western Digital serves consumer and enterprise markets, providing reliable storage solutions for various applications, including personal computing, data centers, and cloud storage. The company is headquartered in San Jose, California. Western Digital's devices are sold worldwide, through Western Digital's physical stores, Western Digital's online store, and authorized resellers (via physical and online stores).
This project will focus only on sales performed by Western Digital's stores (physical and online).

The Data Mart creation will be done using information derived from the PriorityERP database (Western Digital's operational database). The solution will include summarized data tables, focus on device sales data, and data regarding Western Digital's customers, employees, products, and stores. In addition, the BI solution will include costumed reports containing sales analysis, customer analysis, and executive dashboard. These reports will be tailored to the sales department's needs and will contribute to Western Digital's device sales growth.

2. Content

The Data Mart will include transactional and informative focused data regarding Company employees, customers, products, and Sales. The data in the data mart will be according to the S.M.A.R.T method to accomplish management goals.
The data will be transferred through an ETL process from the PriorityERP operational database to the Data Mart –SalesDM and will be displayed in Power BI reports.

ERD model of the SalesDM database: ERD Link

2.1 Tables
The Data Mart will include 1 Fact table, 4 Dimension tables, 2 History tables and 1 transfer table:
•**FACT_Sales** – Data regarding all sales, including the order ID, products bought, quantities, and prices. The data-loading process for this table will be incremental.
•**DIM_Customers** – Data regarding the company's customers.
•**DIM_Stores** – Data regarding the company's stores.
•**DIM_Agents** – Data regarding the company's agents.
•**DIM_Products** - Data regarding the company's products.

History Management Tables:

•**DIM_CustomersHistory** - Historical data regarding the company's customers. The customer history table will be included to track changes in customers over time using Slowly Changing Dimensions (SCD) Type 4.

•**DIM_ProductsHistory** – Historical data regarding the company's products. The product history table will be included to track changes in products over time using Slowly Changing Dimensions (SCD) Type 4.

•**Transfers** - Information about all the updates in the tables.

[Source To Target Link](#)

The tables will be updated daily at 05:00:00 using an automated process configured in the SQL Server Management Studio.

2.2 visualizations
The project will contain a dashboard and reports in Power BI that will contribute to the achievement of the project's goals:

Executive Dashboard:
The dashboard will include key visuals from the reports. The dashboard will allow a wider perspective on the data and will integrate measures from sales and customer analysis.

Customer Analysis:
The customer analysis report will include data regarding Western Digital's customers by date, country, store, product, subcategory, and category. This report is aimed to help Western Digital's customer department to better understand their customers' behavior, like what products they buy, where (countries and stores), and when they shop. This is vital to retain current customers and reach new ones.

Sales Analysis:
The sales report will include data about sales (revenue, number of orders, and number of units) by date, country, product, subcategory, category, store (online vs. physical), and employee which will help the department assess the performance of all the parts needed for sales growth.
The reports will help to identify sale trends like seasonality and trending product categories, analyze product orders and revenue, spot top-performing salespersons, and analyze the differences in behavior between the online store and physical stores. All of these will support data-driven strategic decision making which can lead to growth in sales and revenue.
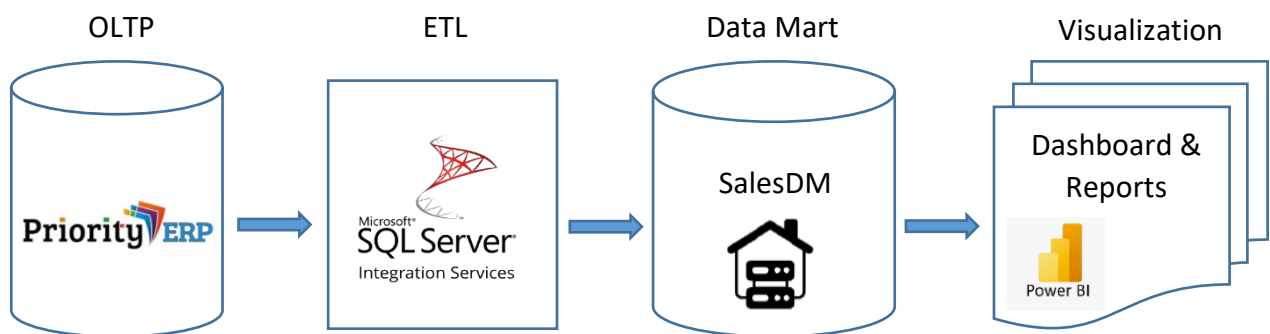
3. Gantt

[Gantt Link](#)

4.Technical Specification

4.1 Prerequisites

SQL Server	PriorityERP database (Western Digital's operational database)
SSIS	ETL processes using SSIS in Visual Studio
Data refresh processes	Definition of JOBS in SSMS
Power BI	Creating dashboard and reports using Power BI

4.2 Solution Architecture

4.2.1 High Level Design

The ETL process, which includes arranging the data into a Data Mart will be performed in SQL Server using SSIS. After the Data Mart creation, reports will be created using Power BI.

4.2.2 Power BI Dashboard and Reports

Executive Dashboard:
- Number of Active Agents
- Number of Active Customers
- New Customers %
- Average Monthly Revenue
- YTD Orders
- Average Revenue per Month
- Best Store by Total Sales
- Best Seller Product
- Highest Sales Country
- YTD Revenue vs Previous Year Revenue
- YOY Growth Revenue by Year
- Total Revenue by Country

• % Orders by Store Type

Sales Report:
• Average Monthly by Orders/Revenue/Units
• YTD by Orders/Revenue/Units
• Average Revenue per Orders
• Orders/Revenue/Units by Store Type
• Orders/Revenue/Units by Brand
• Orders/Revenue/Units by Category/Sub Category
• Orders/Revenue/Units by Year, Quarter and Store Type
• Previous Year Orders/Revenue/Units vs Current Orders/Revenue/Units by Year,
Quarter.

Customers Report:
• Number of Active Customers
• Number of New Customers
• New Customers %
• Average Orders per Customer
• Average Revenue per Customer
• Average Units per Customer
• Number of Customers by Store Type
• Number of Customers by Number of Orders
• Top 5 Selling Products by Number of Customers
• % Customers by Countries

5. Functional Specification

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

* [Source to Target](#)- 11 tables will be used from the operational database.
* [ERD model](#) of the SalesDM database - 1 Fact table, 4 Dimension tables and 2
  History tables.

5.2. ETL processes

The ETL process was done in SSIS using 13 packages.
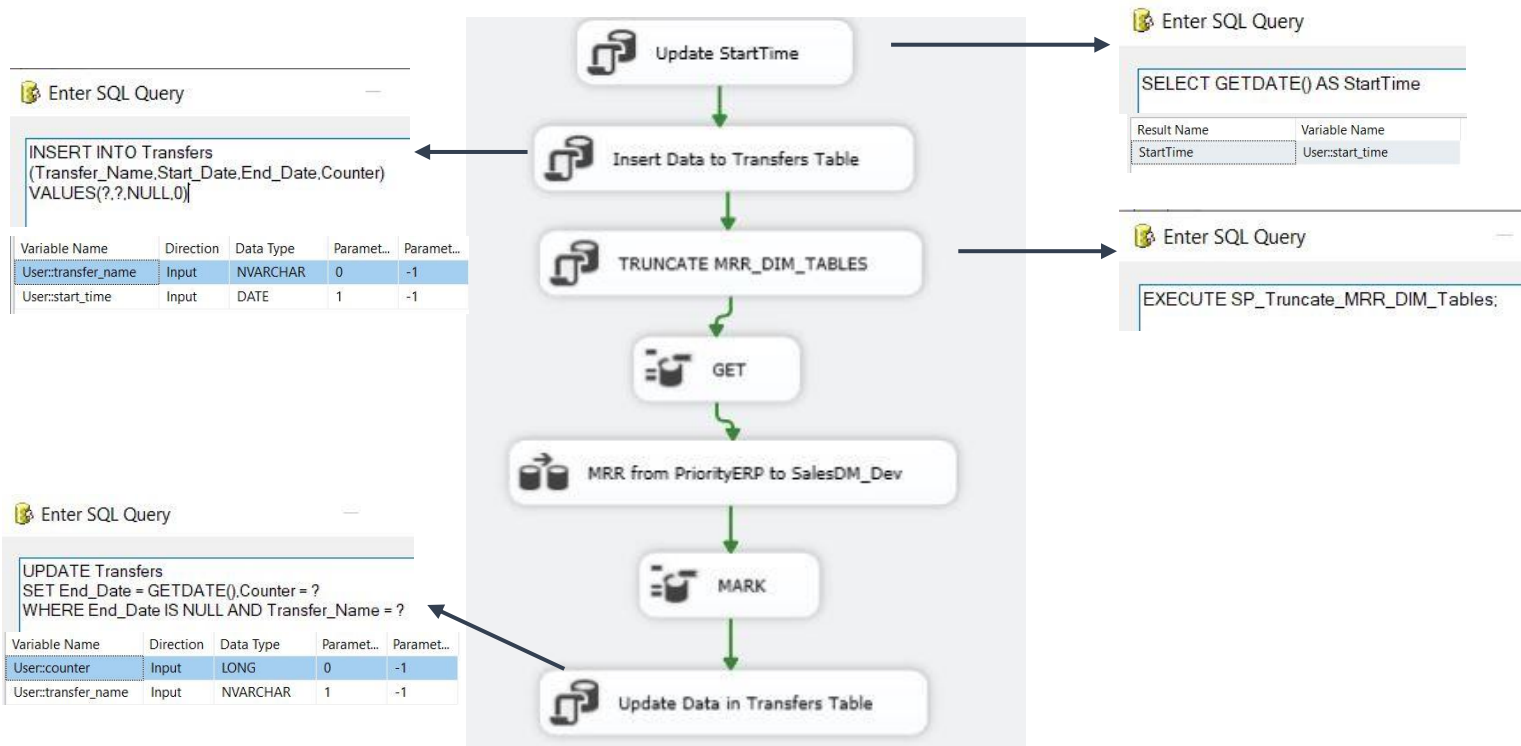All the packages include 3 reoccurring Execute SQL tasks:
Update StartTime Var, Insert Data to Transfers table and Update transfers table with
End Date and Row Count which oversee updating the Transfers table. These will be
explained later in the Transfers table section.

* ### MRR_DIM_Tables Package:
  This package is responsible for loading data from PriorityERP tables to all mirror
  tables relevant for the dim tables (9 tables in total).
 All mirror tables are truncated using a stored procedure.

Enter SQL Query

```
SELECT GETDATE() AS StartTime
```

Result Name	Variable Name
StartTime	User::start_time

Enter SQL Query

```
INSERT INTO Transfers
(Transfer_Name,Start_Date,End_Date,Counter)
VALUES(?,?,NULL,0)
```

Variable Name	Direction	Data Type	Paramet...	Paramet...
User::transfer_name	Input	NVARCHAR	0	-1
User::start_time	Input	DATE	1	-1

Enter SQL Query

```
EXECUTE SP_Truncate_MRR_DIM_Tables;
```

Enter SQL Query

```
UPDATE Transfers
SET End_Date = GETDATE(),Counter = ?
WHERE End_Date IS NULL AND Transfer_Name = ?
```

Variable Name	Direction	Data Type	Paramet...	Paramet...
User::counter	Input	LONG	0	-1
User::transfer_name	Input	NVARCHAR	1	-1

1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **TRUNCATE MRR_DIM_Tables** Store Procedure in SSMS -

```
CREATE PROCEDURE SP_Truncate_MRR_DIM_Tables
AS
BEGIN
    TRUNCATE TABLE [dbo].[MRR_Agents];
    TRUNCATE TABLE [dbo].[MRR_Branches];
    TRUNCATE TABLE [dbo].[MRR_Countries];
    TRUNCATE TABLE [dbo].[MRR_Customers];
    TRUNCATE TABLE [dbo].[MRR_DestCodes];
    TRUNCATE TABLE [dbo].[MRR_Family];
    TRUNCATE TABLE [dbo].[MRR_FamilyTypes];
    TRUNCATE TABLE [dbo].[MRR_Part];
    TRUNCATE TABLE [dbo].[MRR_States];
END;
```

4. **GET** – get from cdc_states table the datetime of the last time data was update in the Agents table.
5. **MRR from PriorityERP to SalesDM_Dev** - transfer data from tables in PriorityERP DB to tables in SalesDM_Dev DB.
6. **MARK** – update the datetime for the next CDC.
7. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
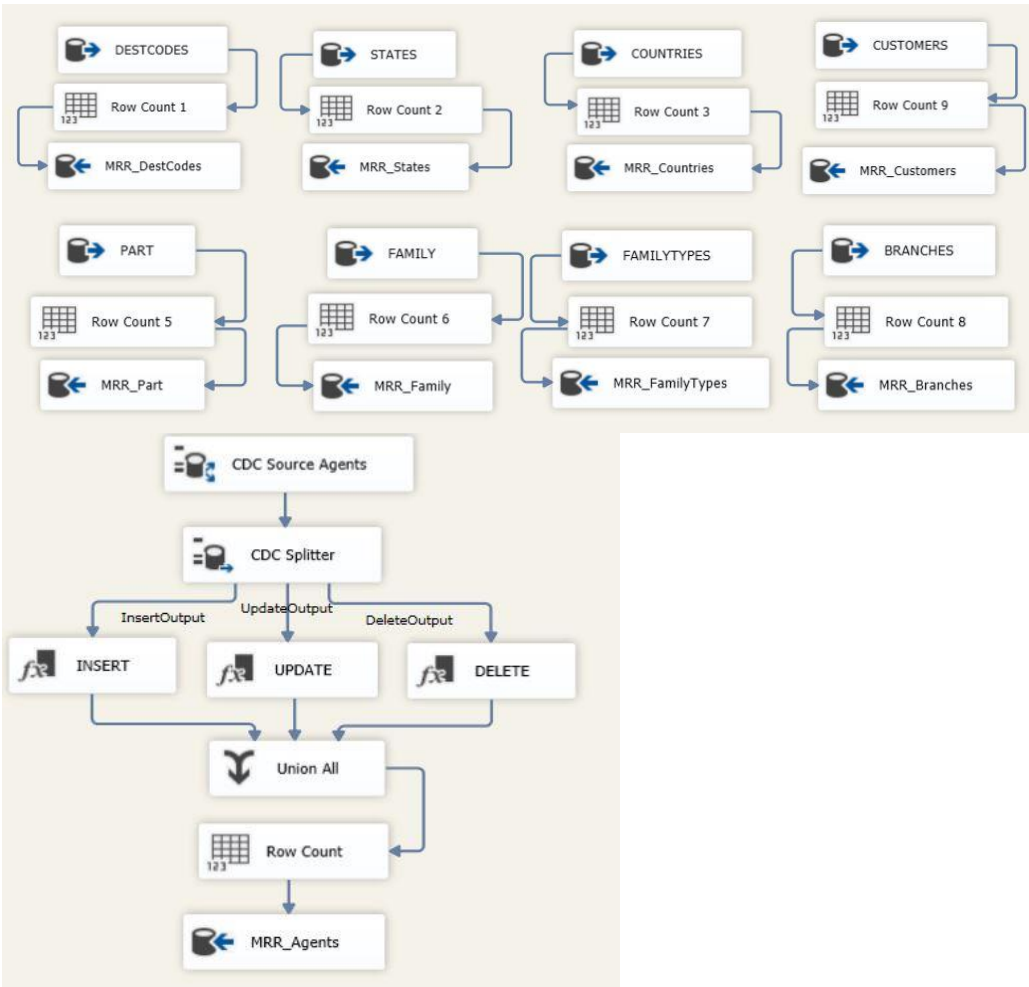
Row_Count – count the rows that pass.
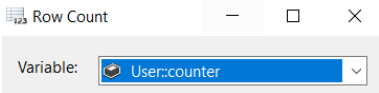
CDC is on Agents table in PriorityERP Database.

CDC capture the changes in Agents table and bring only them.

CDC Splitter divide the operations to 3 types: Insert, Update and Delete.

We mark every operation type by adding Status column:

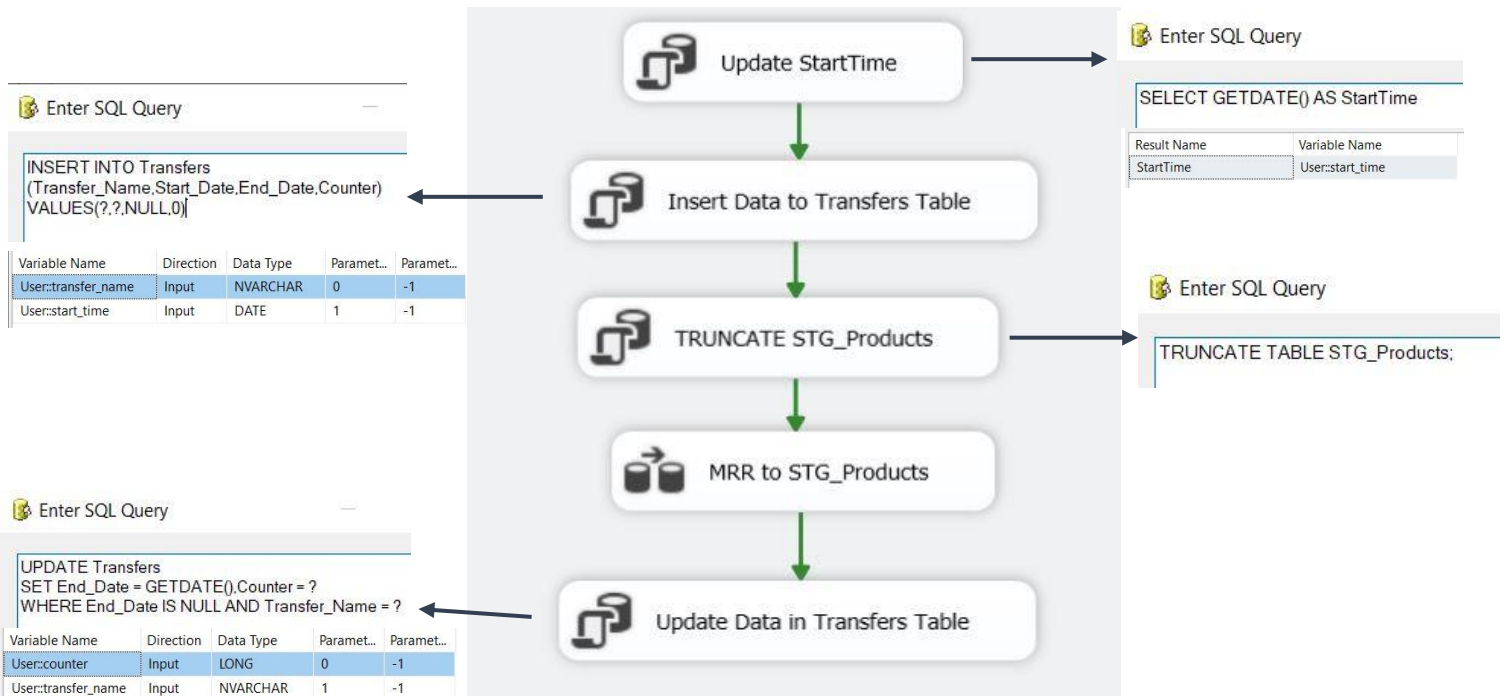Status = 1 -> INSERT, Status = 2 -> UPDATE and Status = 3 -> DELETE.

Package Variables:

📦	CDC_State	MRR_DIM_Tables	String	
📦	counter	MRR_DIM_Tables	Int32	0
📦	start_time	MRR_DIM_Tables	DateTime	21/07/2024 11:13
📦	transfer_name	MRR_DIM_Tables	String	MRR_DIM_Tables

STG_Products table is truncated, and the mirror tables are joined and loaded using a data flow task.
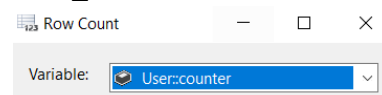
In Control Flow:

1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **TRUNCATE STG_ Products**
4. **MRR to STG_ Products** - transfer data from MRR tables to STG_Products table.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
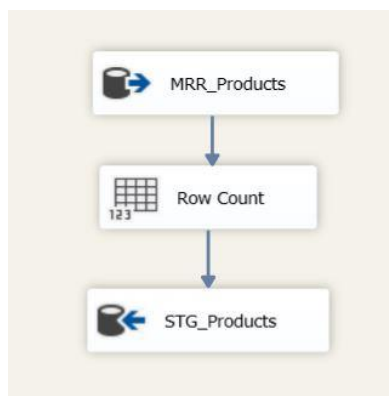
Row_Count – count the rows that pass.


Package Variables:

Name	Scope	Data type	Value
counter	STG_Products	Int32	0
start_time	STG_Products	DateTime	21/07/2024 16:57
transfer_name	STG_Products	String	STG_Products

* DIM_Products Package:

Incremental load and update data to the Dim_Products table is done using the Slowly Changing Dimension transformation (change type: Changing Attribute).

In Control Flow:
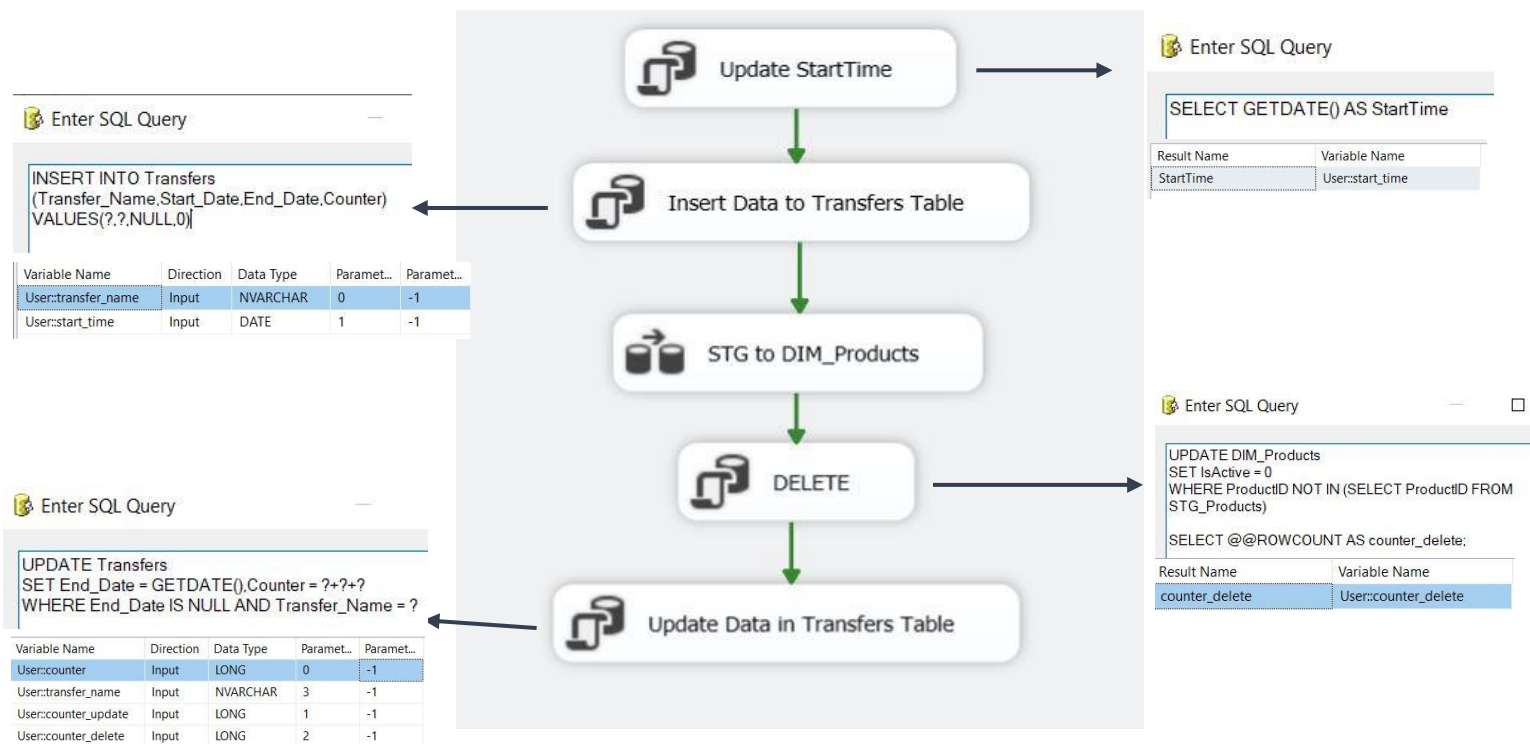
1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **STG to DIM_Products** - transfer data from STG_Products table to DIM_Products table.
4. **DELETE**- Deleted records in STG_Products are updated in DimProducts by changing IsActive value from 1 to 0.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:

Derived Column Name	Derived Column	Expression
IsActive	\<add as new column\>	1

```
UPDATE [dbo].[DIM_Products] SET [Brand] = ?,[Capacity] = ?,
[Category] = ?,[IsActive] = ?,[ProductName] = ?,[SubCategory] = ?,
[UnitPrice] = ?,[UpdateDate] = GETDATE()
 WHERE [ProductID] = ?|
```
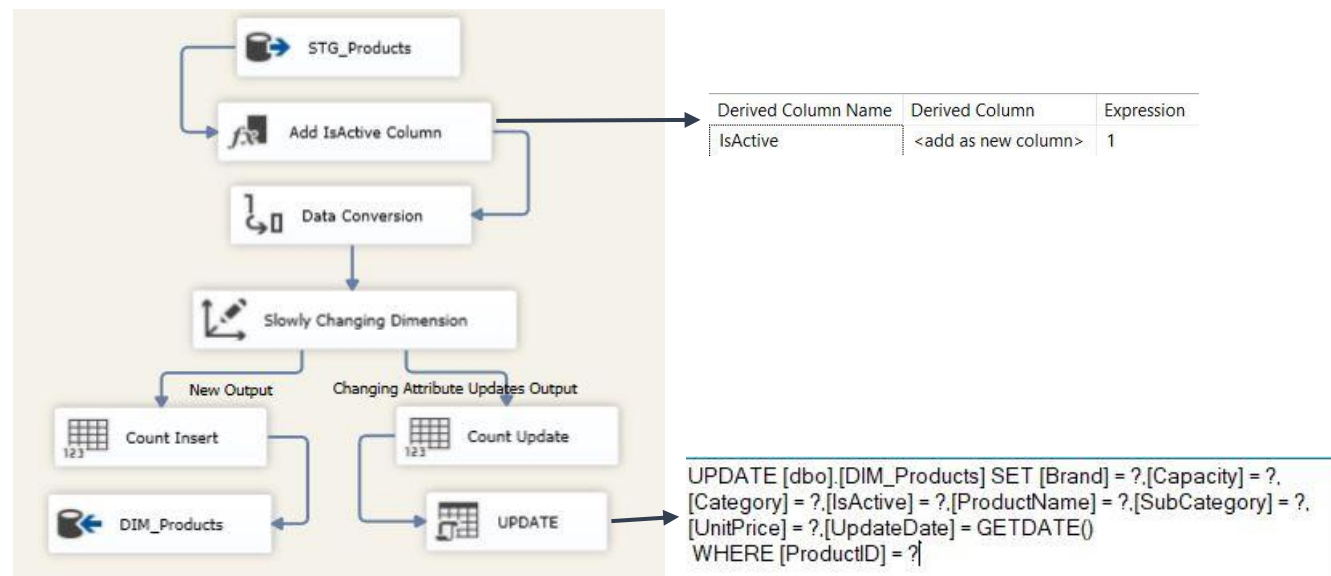
Row_Count – count the rows that pass.

Package Variables:

Name	Scope	Data type	Value
counter	DIM_Products	Int32	0
counter_delete	DIM_Products	Int32	0
counter_update	DIM_Products	Int32	0
start_time	DIM_Products	DateTime	21/07/2024 17:07
transfer_name	DIM_Products	String	DIM_Products

* DIM_ProductsHistory Package:
  Incremental load and update data to the Dim_ProductsHistory table is done using
  the Slowly Changing Dimension transformation (change type: Historical Attribute).
  Deleted records are located in the same manner as in the DIM_Products package but
  while using EndDate column instead of IsActive.
  Every row get StartDate and if there is change in the row details the SCD will add a
  new row to the DIM_ProductsHistory and update the EndDate with current datetime
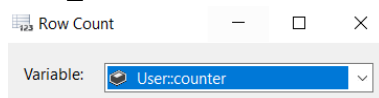 for the changed row.
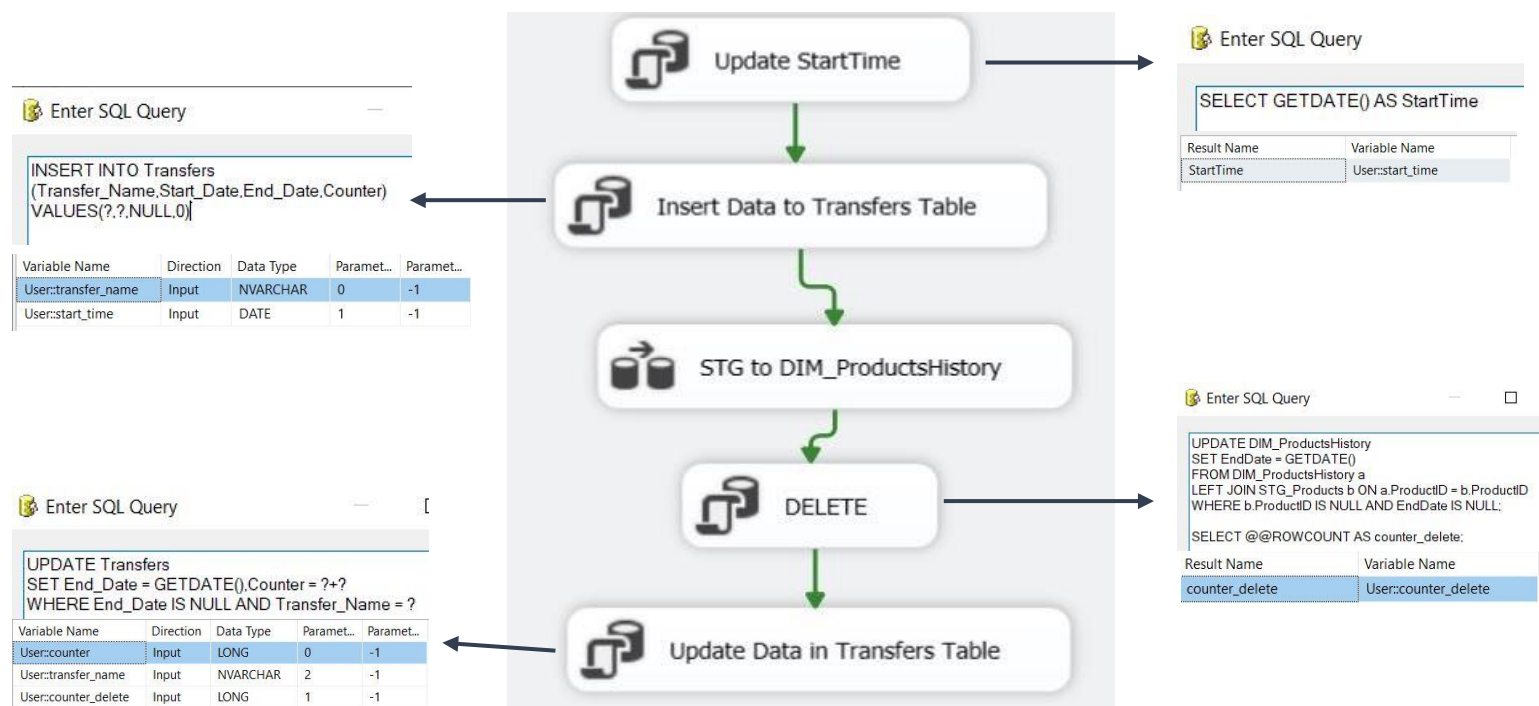
In Control Flow:

1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **STG to DIM_ProductsHistory** - transfer data from STG_Products table to DIM_ProductsHistory table.
4. **DELETE**- Deleted records in STG_Products are updated in Dim_ProductsHistory by changing EndDate value from NULL to current date.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
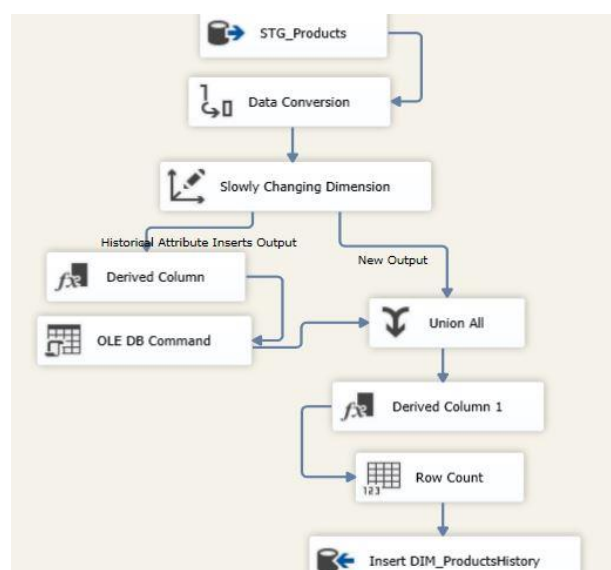
Row_Count – count the rows that pass.

Package Variables:

Name	Scope	Data type	Value
counter	DIM_ProductsHistory	Int32	0
counter_delete	DIM_ProductsHistory	Int32	0
start_time	DIM_ProductsHistory	DateTime	22/07/2024 09:31
transfer_name	DIM_ProductsHistory	String	DIM_ProductsHistory

* **STG_Customers Package:**
  STG_Customers table is truncated, and the mirror tables are joined and loaded using a data flow task.

In Control Flow:
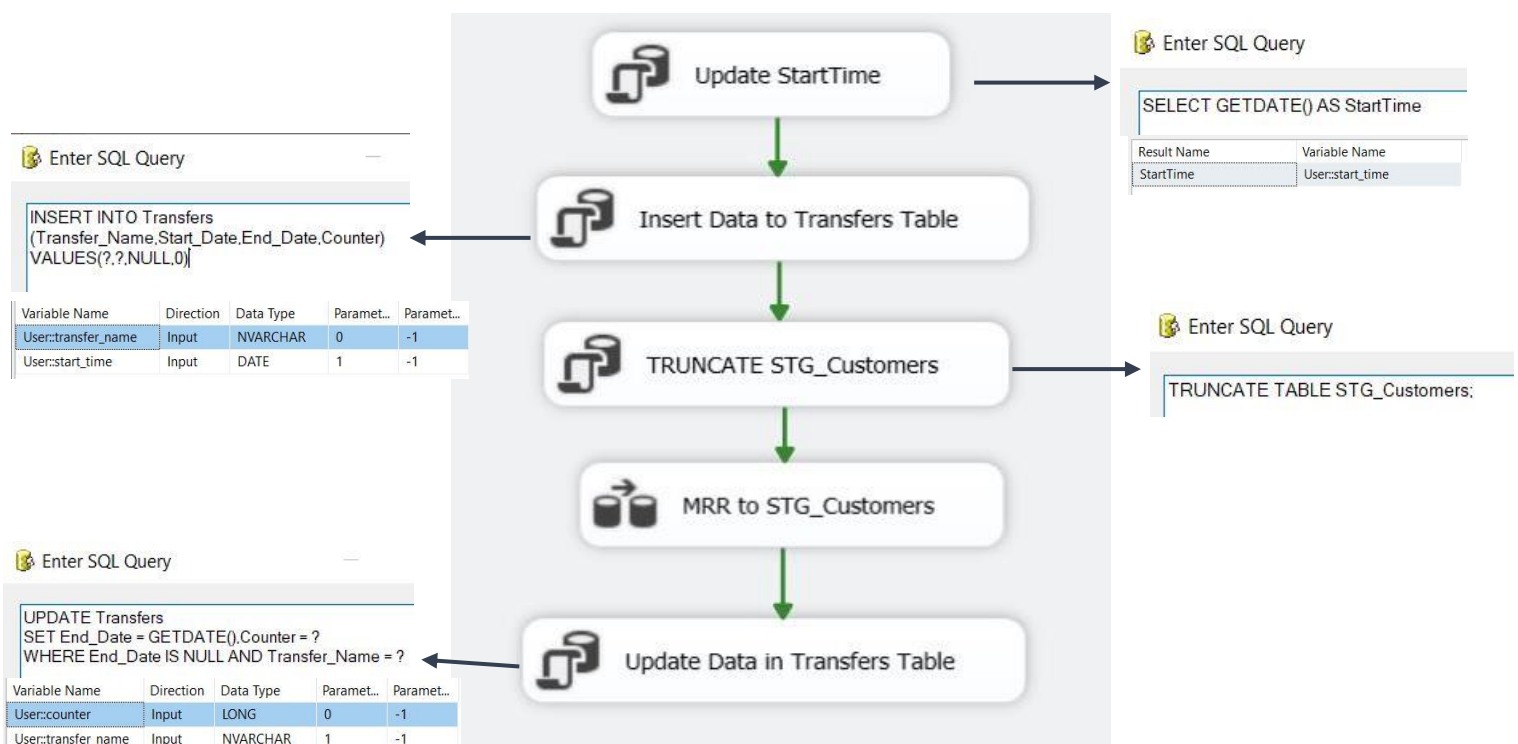
1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **TRUNCATE STG_Customers**
4. **MRR to STG_Customers** - transfer data from MRR tables to STG_Customers table.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
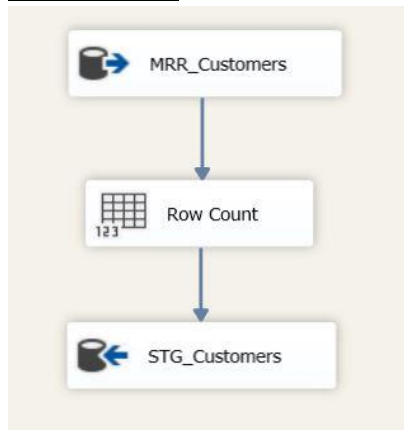
Row_Count – count the rows that pass.

Package Variables:

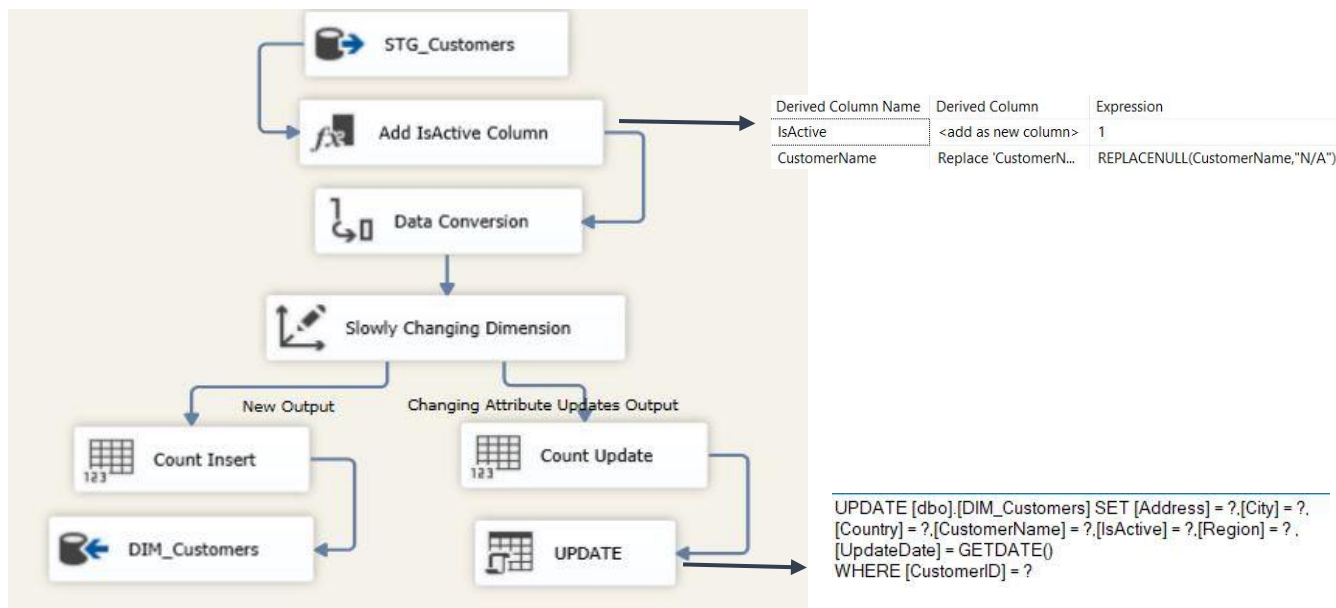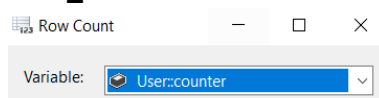| Name	Scope	Data type	Value
counter	STG_Customers	Int32	0
start_time	STG_Customers	DateTime	21/07/2024 16:48
transfer_name	STG_Customers	String	STG_Customers

* DIM_Customers Package:
  Incremental load and update data to the Dim_Customers table is done using the Slowly Changing Dimension transformation (change type: Changing Attribute).
  In Control Flow:
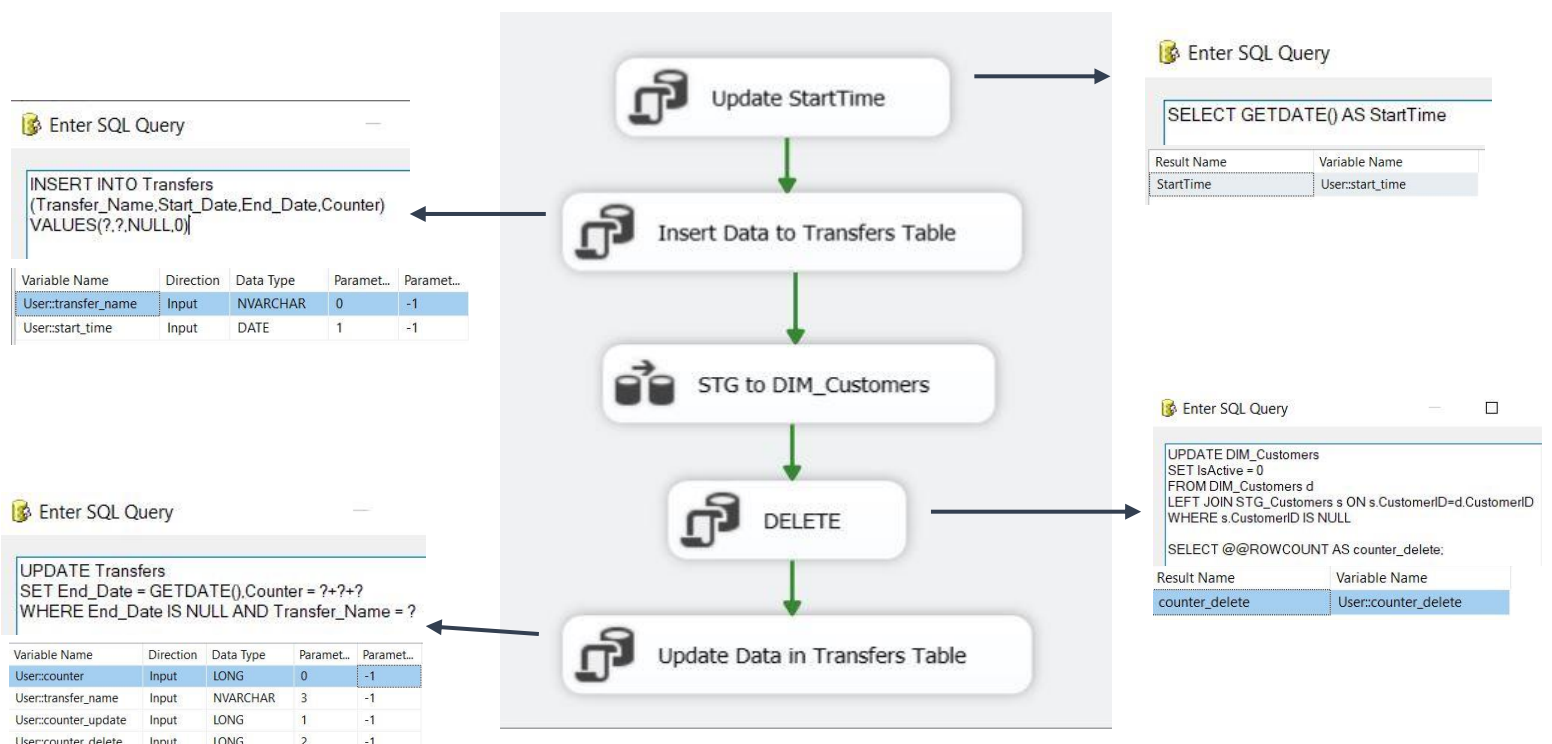
1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **STG to DIM Customers** - transfer data from STG_Customers table to DIM_Customers table.
4. **DELETE**- Deleted records in STG_Customers are updated in Dim_Customers table by changing IsActive value 1 to 0.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:

Derived Column Name	Derived Column	Expression
IsActive	<add as new column>	1
CustomerName	Replace 'CustomerN...	REPLACENULL(CustomerName,"N/A")

```
UPDATE [dbo].[DIM_Customers] SET [Address] = ?,[City] = ?,
[Country] = ?,[CustomerName] = ?,[IsActive] = ?,[Region] = ? ,
[UpdateDate] = GETDATE()
WHERE [CustomerID] = ?
```

Row_Count – count the rows that pass.

Package Variables:

Name	Scope	Data type	Value
counter	DIM_Customers	Int32	0
counter_delete	DIM_Customers	Int32	0
counter_update	DIM_Customers	Int32	0
start_time	DIM_Customers	DateTime	22/07/2024 10:47
transfer_name	DIM_Customers	String	DIM_Customers

Incremental load and update data to the Dim_CustomersHistory table is done using the Slowly Changing Dimension transformation (change type: Historical Attribute). Deleted records are located in the same manner as in the DIM_Customers package but while using EndDate column instead of IsActive.

Every row get StartDate and if there is change in the row details the SCD will add a new row to the DIM_CustomersHistory and update the EndDate with current datetime for the changed row.
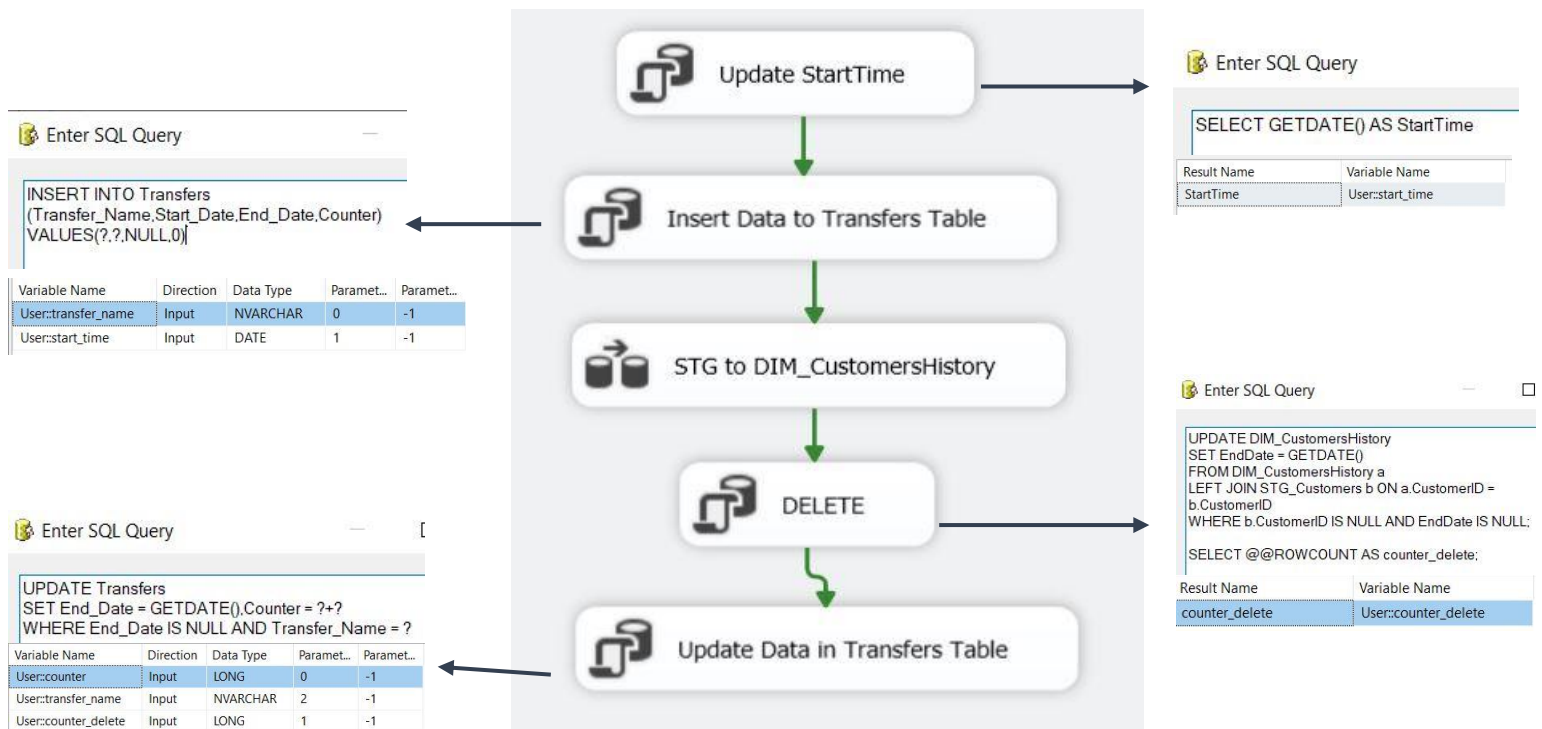
In Control Flow:

1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **STG to DIM_CustomersHistory** - transfer data from STG_Customers table to DIM_CustomersHistory table.
4. **DELETE**- Deleted records in STG_Customers are updated in Dim_CustomersHistory by changing EndDate value from NULL to current date.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
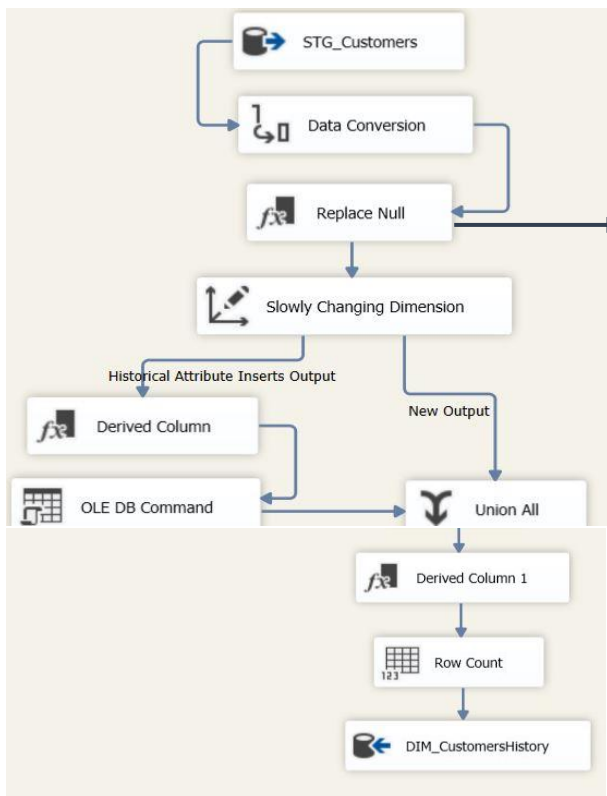
Derived Column Name	Derived Column	Expression
Copy of CustomerName	Replace 'Copy of CustomerName'	REPLACENULL([Copy of CustomerName],"N/A")

Row_Count – count the rows that pass.
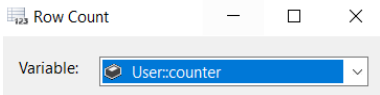
Package Variables:

Name	Scope	Data type	Value
counter	DIM_CustomersHistory	Int32	0
counter_delete	DIM_CustomersHistory	Int32	0
start_time	DIM_CustomersHistory	DateTime	22/07/2024 14:29
transfer_name	DIM_CustomersHistory	String	DIM_CustomersHistory

* DIM_Agents Package:
  DIM Agents table Use CDC and get only the rows that have been changed in the operating database from the MRR_Agents table.

In Control Flow:
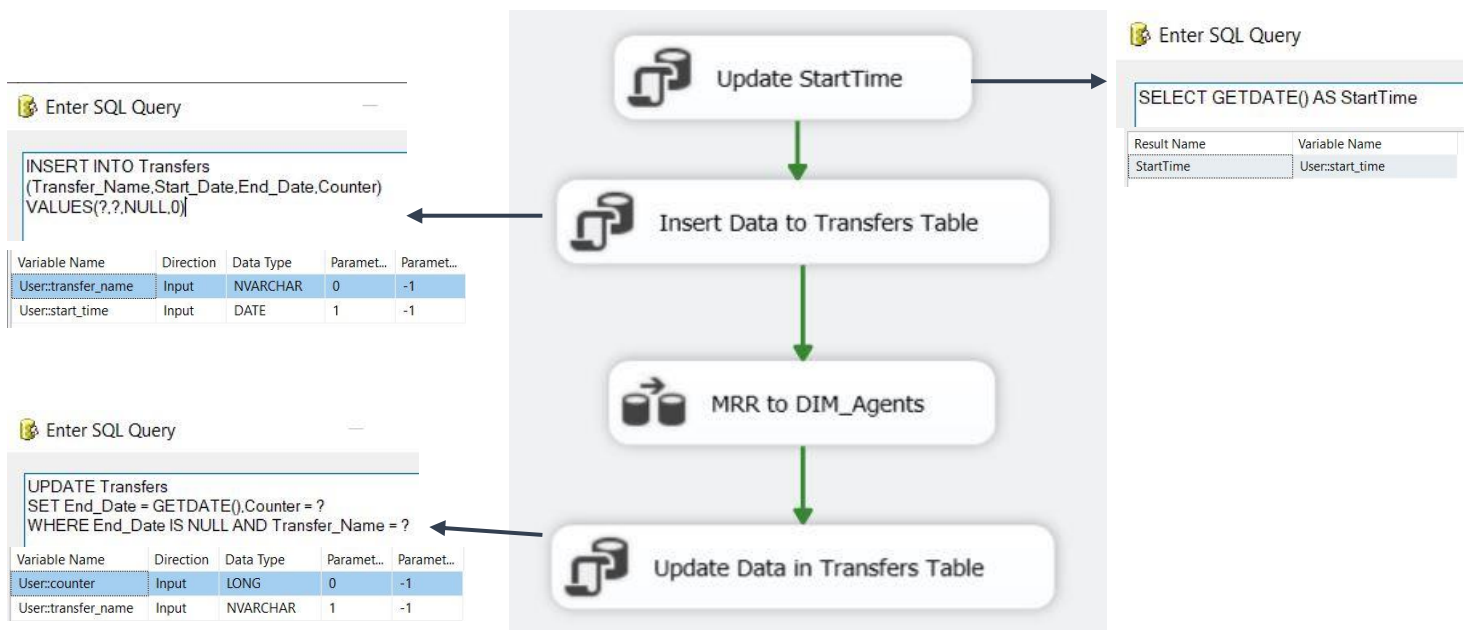
1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **MRR to DIM_Agents**- transfer data from MRR_Agents table to DIM_Agents table.
4. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
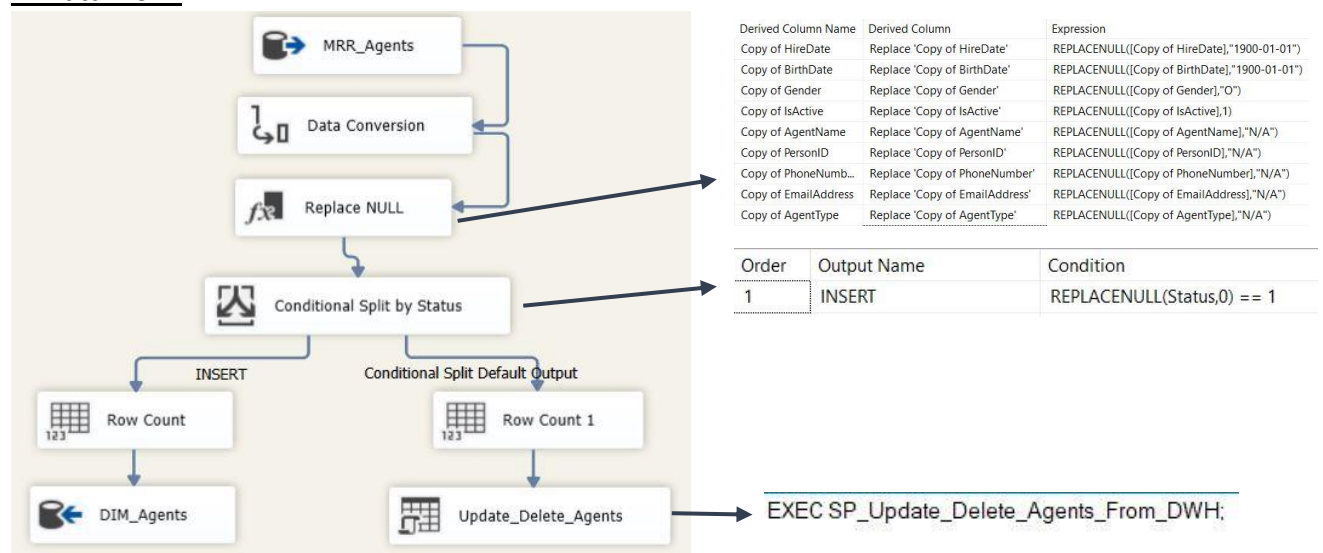
Row_Count – count the rows that pass.

If Status = 1 then Insert new row to DIM_Agents.

If Status = 2 then Update row by Merge query.

If Status = 3 then Delete – Update IsActive value to 0 in current row by Merge query.

Store Procedure Update_Delete_Agents:

```sql
CREATE PROCEDURE SP_Update_Delete_Agents_From_DWH
AS
    MERGE [dbo].[DIM_Agents] AS Target
    USING [dbo].[MRR_Agents] AS Source
    ON [Target].[AgentID] = [Source].[AGENT]
    WHEN MATCHED THEN UPDATE
    SET [Target].[AgentName] = [Source].[AGENTNAME],
    [Target].[PersonID] = [Source].[PERSON_ID],
    [Target].[PhoneNumber] = [Source].[PHONE],
    [Target].[EmailAddress] = [Source].[EMAIL],
    [Target].[Gender] = [Source].[GENDER],
    [Target].[HireDate] = [Source].[HIREDATE],
    [Target].[BirthDate] = [Source].[BIRTHDATE],
    [Target].[AgentType] = [Source].[AGENT_TYPE],
    [Target].[IsActive] = IIF([Source].[Status]=2,1,0);
```

Package Variables:

Name	Scope	Data type	Value
counter	DIM_Agents	Int32	0
counter_update	DIM_Agents	Int32	0
start_time	DIM_Agents	DateTime	21/07/2024 13:14
transfer_name	DIM_Agents	String	DIM_Agents

DIM Stores Package:
Incremental load, Update and Delete data in the Dim_Stores table is done using
Merge query.

In Control Flow:
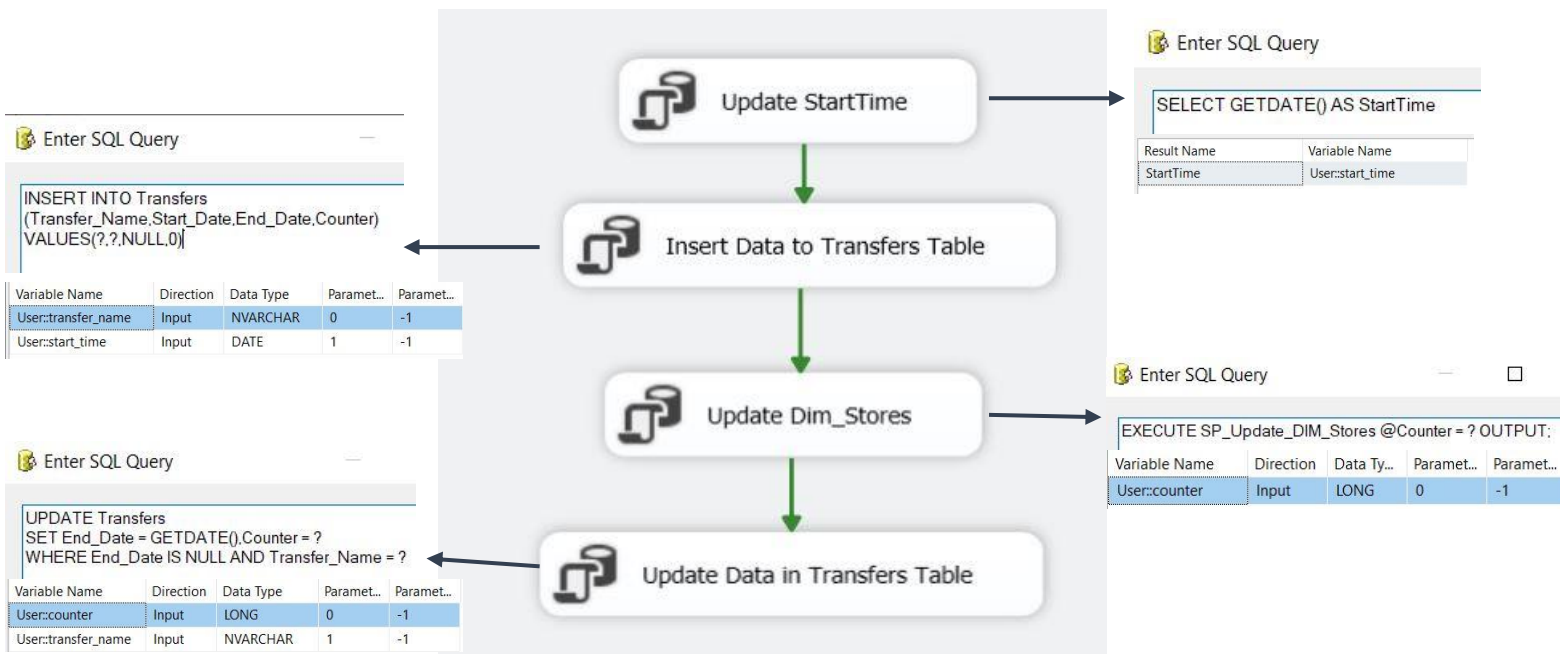
1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with
3. current datetime and package name.
4. **Update DIM Stores**- Insert/Update/Delete data in DIM_Stores table from
   MRR_Branches table using Merge query.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with
   end datetime and rows number that changed.

Store Procedure Update DIM Stores:
Insert row – if storeID is in MRR_Branches table but not in DIM_Stores table then
insert new row to DIM_Stores table.
Update row – if storeID is in both MRR_Branches and DIM_Stores tables then update
row in DIM_Stores table.
Delete row – if storeID not exist in MRR_Branches but exist in DIM_Stores then
replace IsActive value from 1 to 0.

```sql
CREATE PROCEDURE SP_Update_DIM_Stores
@Counter INT OUTPUT
AS
BEGIN
    MERGE [dbo].[DIM_Stores]  AS Target
    USING [dbo].[MRR_BRANCHES]  AS Source
    ON [Target].[StoreID]= [Source].[BRANCH]

    WHEN MATCHED AND [Target].[IsActive] = 1 AND
    (ISNULL([Target].[StoreName],'') <> ISNULL([Source].[BRANCHNAME],'')
    OR ISNULL([Target].[City],'') <> ISNULL([Source].[CITY],'')
    OR ISNULL([Target].[Country],'') <> ISNULL([Source].[COUNTRY],''))
    THEN UPDATE
    SET [Target].[StoreName] = CONVERT(NVARCHAR(40), ISNULL([Source].[BRANCHNAME],'N/A')),
    [Target].[City] = CONVERT(NVARCHAR(20), ISNULL([Source].[CITY],'N/A')),
    [Target].[Country] =  CONVERT(NVARCHAR(25), ISNULL([Source].[COUNTRY],'N/A'))

    WHEN NOT MATCHED BY Target THEN INSERT
    VALUES([Source].[BRANCH],
    CONVERT(NVARCHAR(40), ISNULL([Source].[BRANCHNAME],'N/A')),
    CONVERT(NVARCHAR(20), ISNULL([Source].[CITY],'N/A')),
    CONVERT(NVARCHAR(25), ISNULL([Source].[COUNTRY],'N/A')),1)

    WHEN NOT MATCHED BY Source AND [Target].[IsActive] = 1 THEN UPDATE SET [Target].[IsActive] = 0;

    SET @Counter = @@ROWCOUNT;
END
```

Package Variables:

| Name	Scope	Data type	Value
counter	DIM_Stores	Int32	0
start_time	DIM_Stores	DateTime	21/07/2024 16:28
transfer_name	DIM_Stores	String	DIM_Stores

MRR Sales Package:

This package is responsible for loading data from PriorityERP tables to all mirror tables relevant for the fact table (2 tables in total).
All mirror tables are truncated using a stored procedure.
After one run, Lookup is used to perform incremental loading and load only the new rows.

In Control Flow:
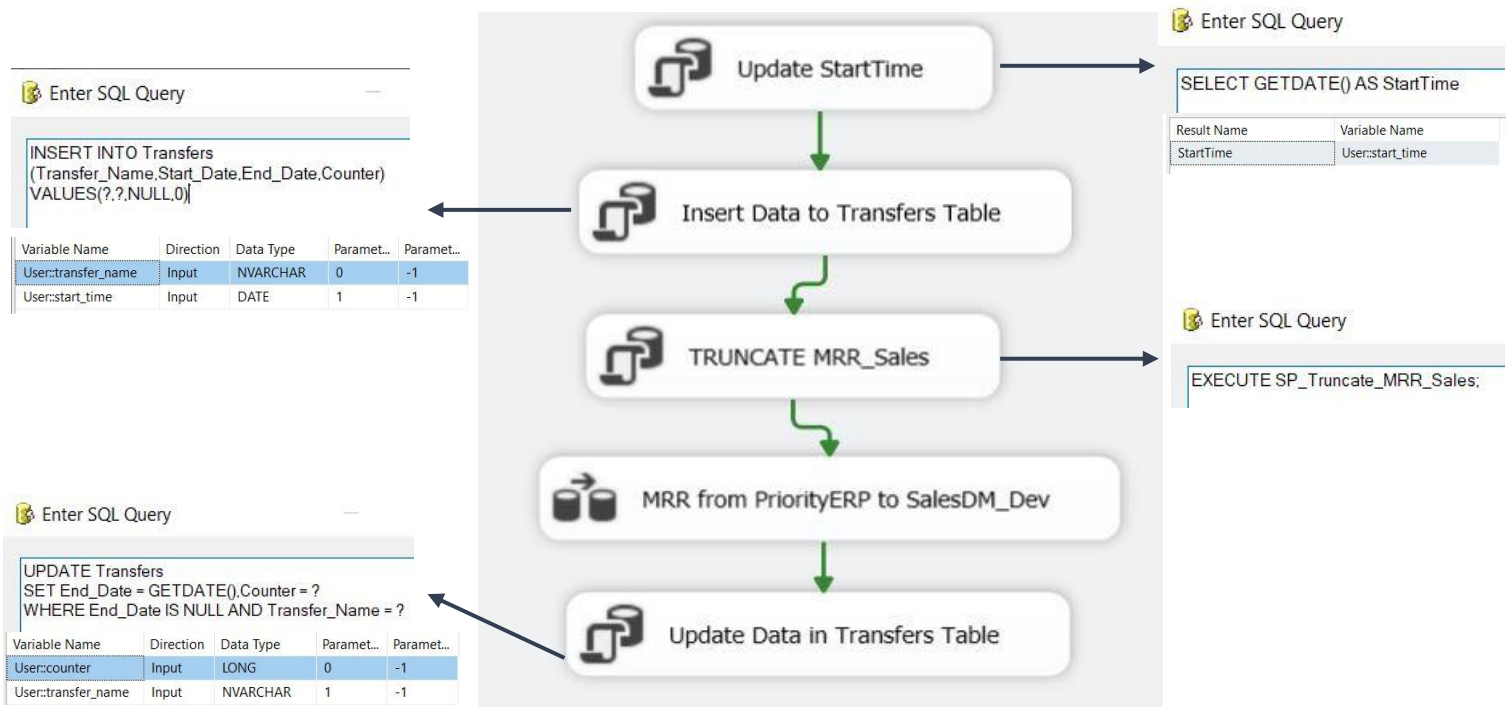
1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **TRUNCATE MRR Sales** Store Procedure in SSMS -

```
CREATE PROCEDURE SP_Truncate_MRR_Sales
AS
BEGIN
    TRUNCATE TABLE [dbo].[MRR_Invoices];
    TRUNCATE TABLE [dbo].[MRR_InvoiceItems];

END;
```

4. **MRR from PriorityERP to SalesDM Dev** - transfer data from tables in PriorityERP DB to tables in SalesDM_Dev DB.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
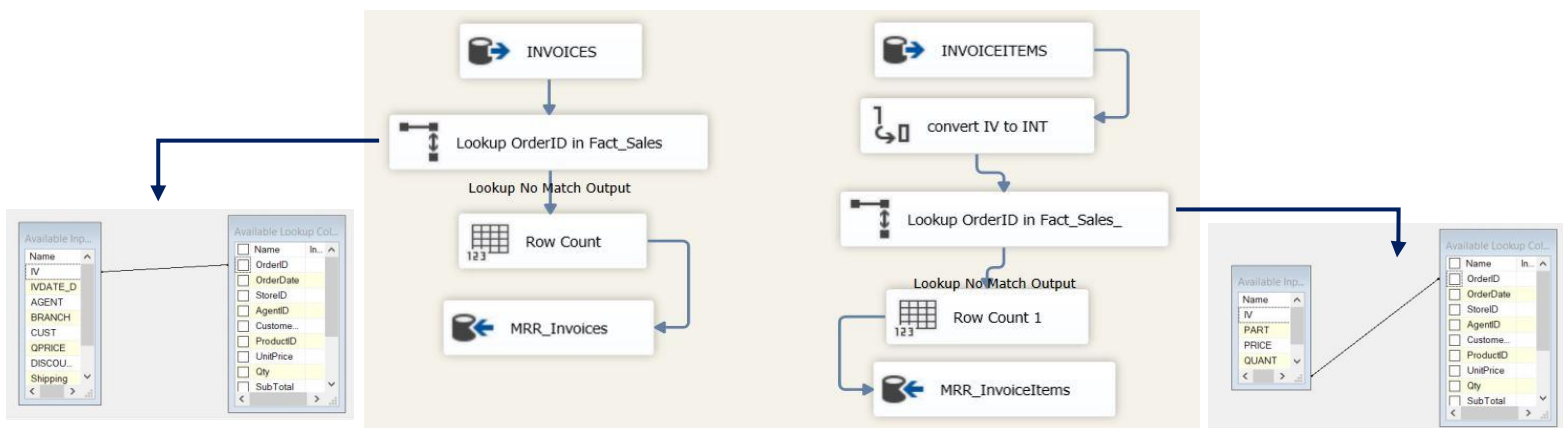
Row_Count – count the rows that pass.
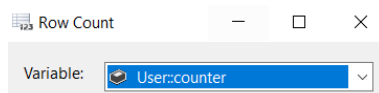
Data is incrementally loaded using lookup transformation. That means only new transactions that can't be found in the Fact_Sales table are loaded.
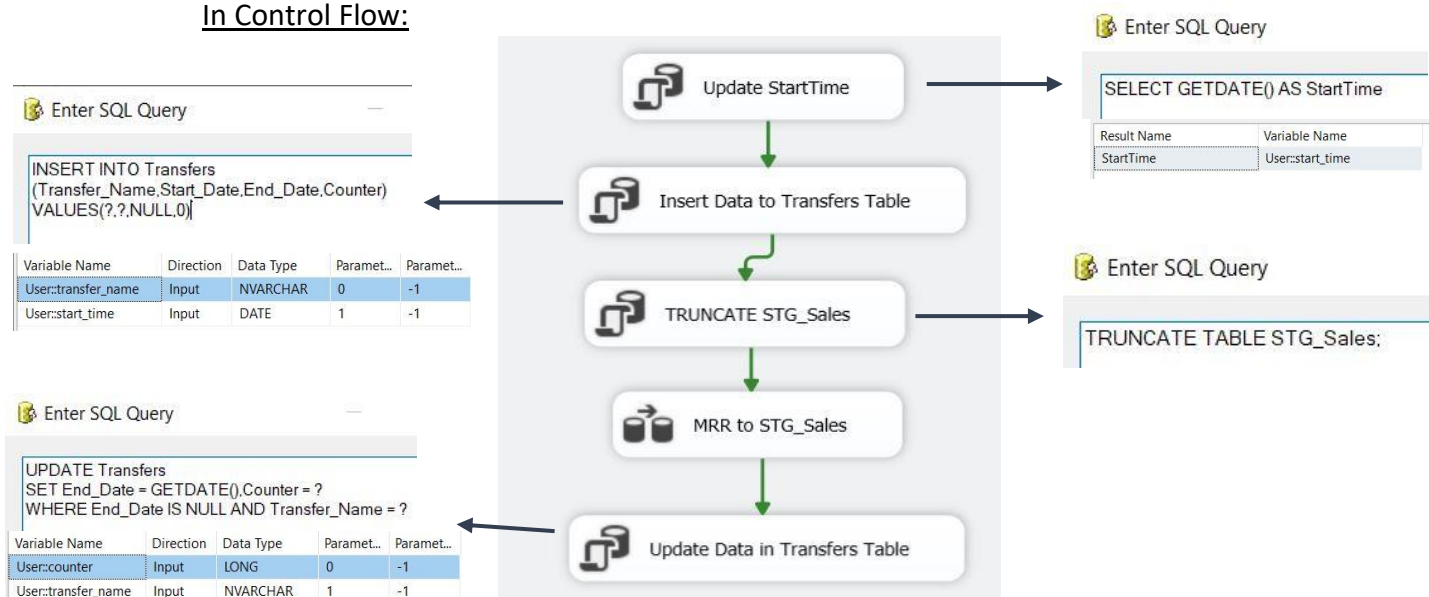
Package Variables:

Name	Scope	Data type	Value
counter	MRR_Sales	Int32	0
start_time	MRR_Sales	DateTime	21/07/2024 11:22
transfer_name	MRR_Sales	String	MRR_Sales

* STG_Sales Package:
  STG_Sales table is truncated, and the mirror tables are joined and loaded using a data flow task.
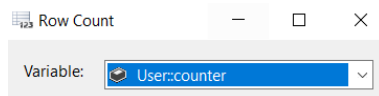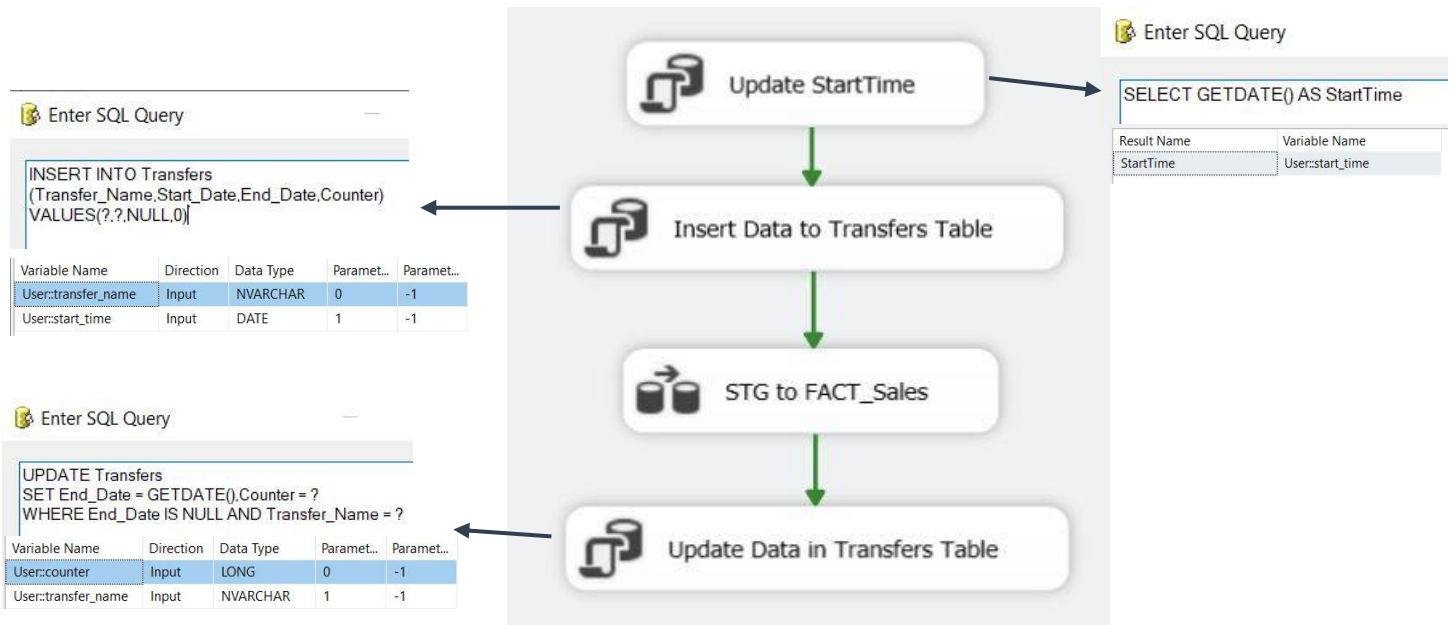 In Control Flow:

22

1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **TRUNCATE STG_Sales**
4. **MRR to STG_Sales** - transfer data from MRR tables to STG_Sales table.
5. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:
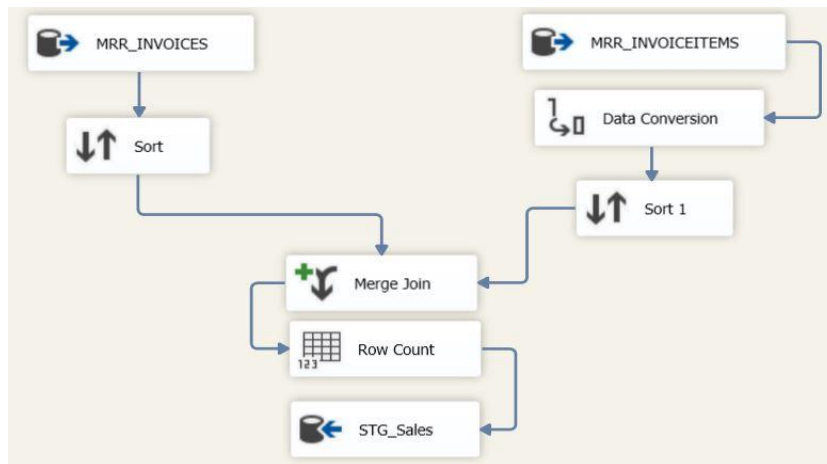
Row_Count – count the rows that pass.

Package Variables:

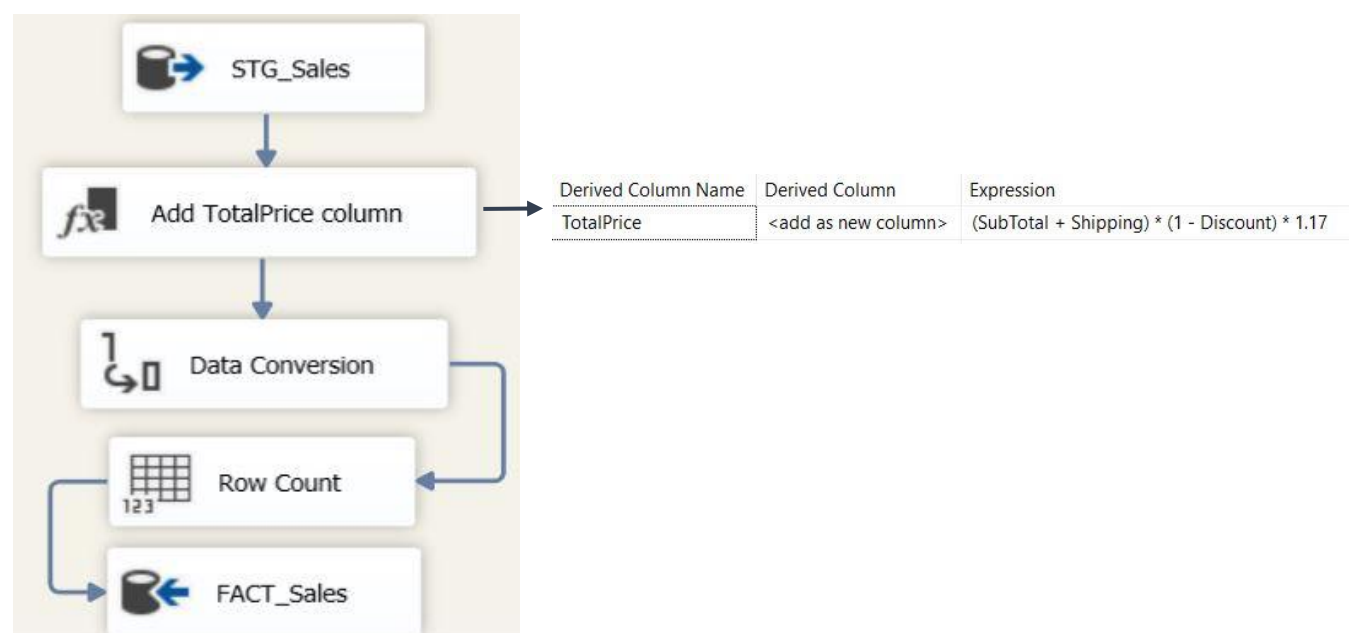| Name	Scope	Data type	Value
counter	STG_Sales	Int32	0
start_time	STG_Sales	DateTime	23/07/2024 16:25
transfer_name	STG_Sales	String	STG_Sales

* FACT_Sales Package:

Data is loaded from STG_Sales to FACT_Sales table, and a TotalPrice column is added. Only new rows are loaded to the FACT_Sales table.

In Control Flow:

1. **Update StartTime** - set current datetime.
2. **Insert Data to Transfers table** - insert new row into transfers table with current datetime and package name.
3. **STG to FACT_Sales** - transfer data from STG_Sales to FACT_Sales table.
4. **Update Data in Transfers table** - Update the inserted row in transfers table with end datetime and rows number that changed.

In Data Flow:

Package Variables:

Name	Scope	Data type	Value
counter	FACT_Sales	Int32	0
start_time	FACT_Sales	DateTime	23/07/2024 16:43
transfer_name	FACT_Sales	String	FACT_Sales

* **Transfers table:**

To monitor the ETL process, a Transfers table was created documenting each data insert: which table was updated, how many rows were inserted (counter), start datetime, end datetime and total time. The tasks and transformation in charge of the updates are included in all of the packages.

In the control flow user variable StartTime is updated in the first task,then in insert task StartTime and transfer name are inserted to Transfers table. After the operations are done on the data then an update statement is executed in the last task, inserting counter and GETDATE() as EndTime values.
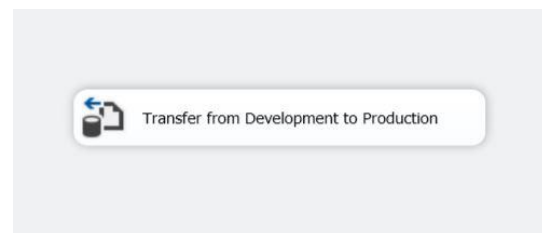
In the data flow task the user variable counter is updated using a Row Count transformation.

Transfer_ID	Transfer_Name	Start_Date	End_Date	Total	Counter
1	MRR_DIM_Tables	2024-07-21 11:38:35.000	2024-07-21 11:39:09.433	34433	41271

* **Transfer to Production:**

Copy all Development database to Production database.

In Control Flow:

Transfer from Development to Production

* **Deploy to SSMS:**

Deploy all packages to SSMS.

- Integration Services Catalogs
  - SSISDB
    - SalesDM
      - Projects
        - WesternDigitalSalesDM_Dev
          - Packages
            - Copy to production.dtsx
            - DIM_Agents.dtsx
            - DIM_Customers.dtsx
            - DIM_CustomersHistory.dtsx
            - DIM_Products.dtsx
            - DIM_ProductsHistory.dtsx
            - DIM_Stores.dtsx
            - FACT_Sales.dtsx
            - MRR_DIM_Tables.dtsx
            - MRR_Sales.dtsx
            - STG_Customers.dtsx
            - STG_Orders.dtsx
            - STG_Products.dtsx
          - Environments
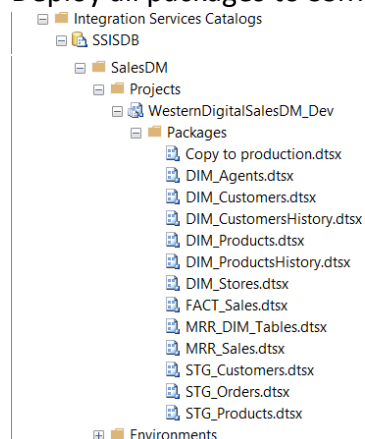
Automatic Processing:

The data is automatically refreshed daily at 5:00:00 using SQL Agent jobs, the first job executed is the SalesDM-MRR_DIM this job executes the next job and so on.

6. Visualization in Power BI

6.1 Model

The reports were created using Power BI Desktop and were published to Power BI Service. The model in the Power BI includes the Fact table and the 4 Dimension tables (not including the products and customers history tables). To these tables, a DimDate table was added.

Segment Table – create to divided number of customers by number of orders.

total by Table - contains parameters for Orders/Revenue/Units.
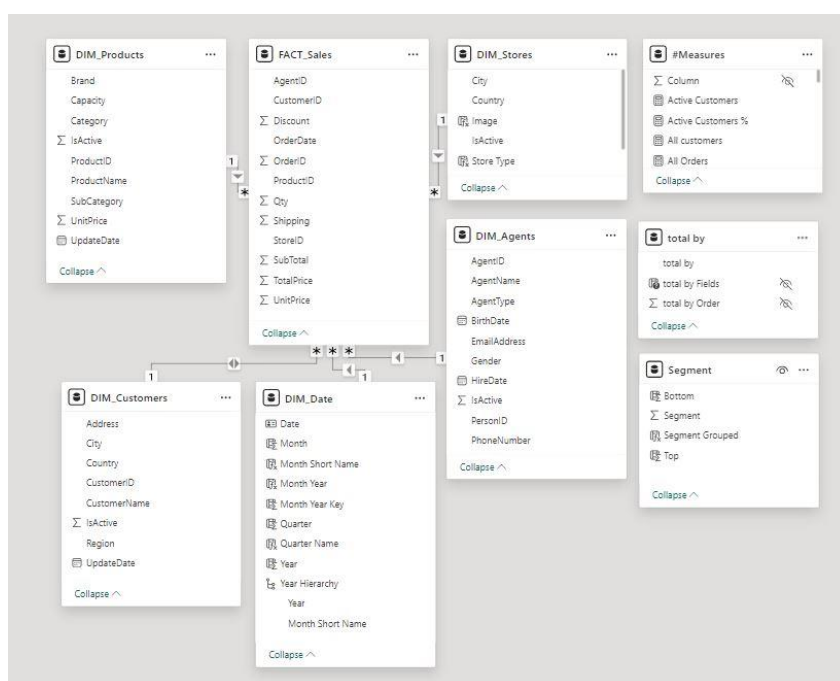
6.2 Dashboard & Reports

The project includes 1 Executive Dashboard and 2 reports: Sales Analysis and Customer Analysis. The reports load data from SalesDM database for years 2021-2024. The years 2021 and 2024 have partial data.

Executive Dashboard:

The dashboard was created to provide a wider perspective on the company's status, it includes the main KPIs and general graphs.
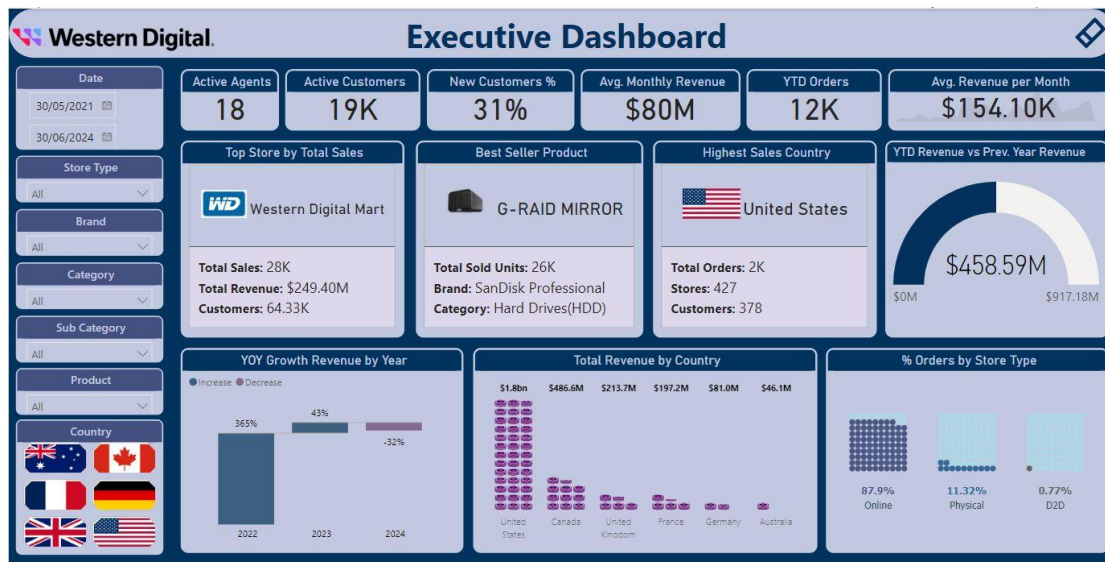
KPI Cards:
- Active Agents
- Active Customers
- New Customers %
- Average Monthly Revenue
- YTD Orders
- Average Revenue per Month

Information Cards:
- Best Store by Total Sales
- Best Seller Product
- Highest Sales Country

Graphs:
- YTD Revenue vs Previous Year Revenue
- YOY Growth Revenue by Year
- Total Revenue by Country
- % Orders by Store Type

Slicers:
- Date
- Store Type
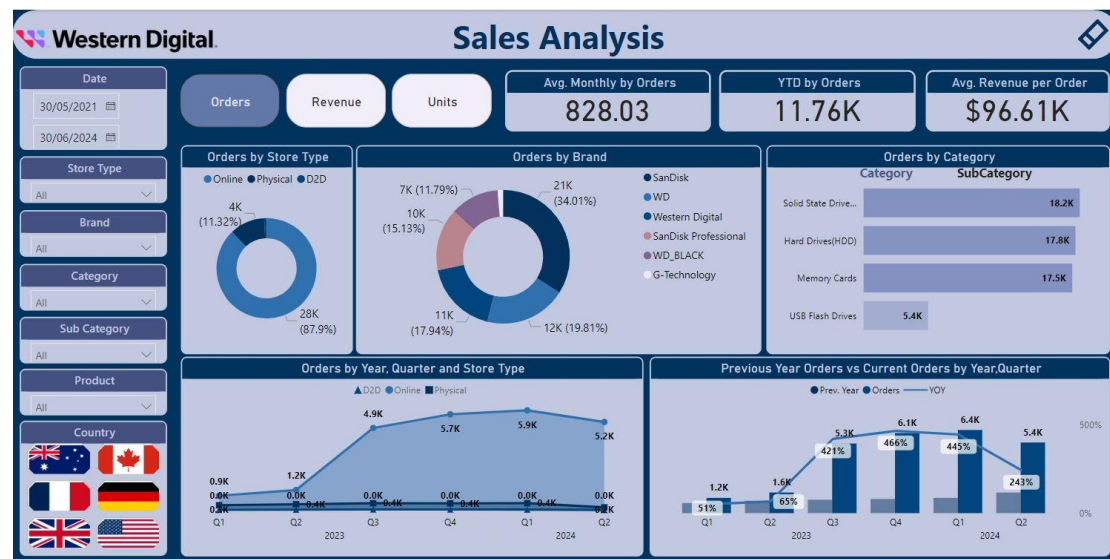- Brand
- Category
- Sub Category
- Product Name
- Country

Sales Analysis:

This report was created for the sales department to follow and understand sales performance to achieve the department's goals.

In its initial state, the graphs present orders data. Using the three buttons on the top left, the user can control the data shown in the graphs and change it - Orders/Revenue/Units data.

In category graph, the user can switch between category and sub category data.

KPI Cards:
- Average Monthly by Orders/Revenue/Units
- YTD by Orders/Revenue/Units
- Average Revenue per Orders

Graphs:
- Orders/Revenue/Units by Store Type
- Orders/Revenue/Units by Brand
- Orders/Revenue/Units by Category/Sub Category
- Orders/Revenue/Units by Year, Quarter and Store Type
- Previous Year Orders/Revenue/Units vs Current Orders/Revenue/Units by Year, Quarter.

Slicers:
- Date
- Store Type
- Brand
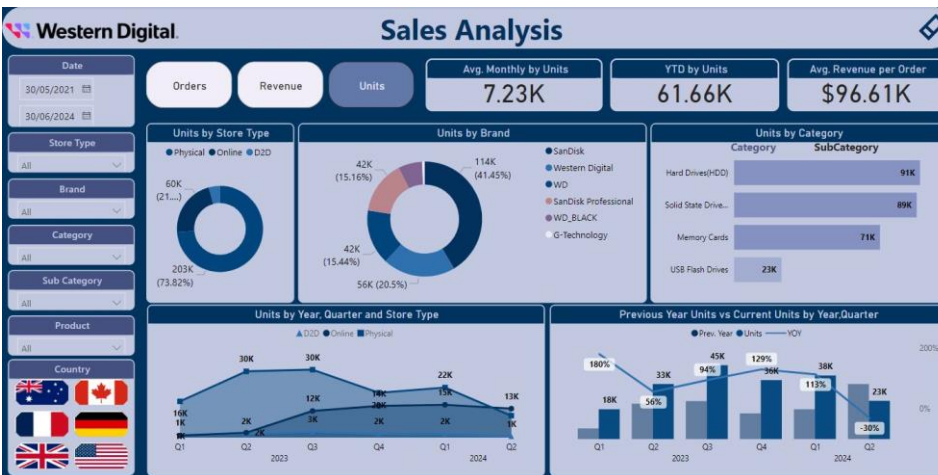- Category
- Sub Category
- Product Name
- Country

Orders State:
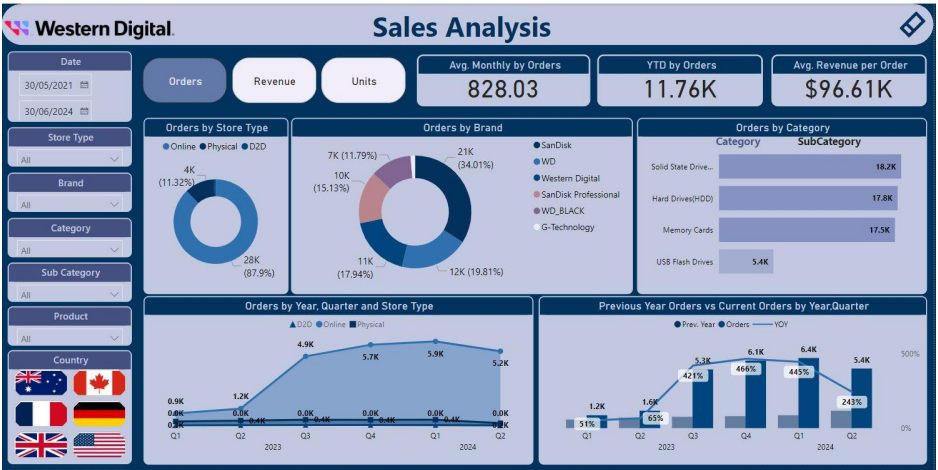
Revenue State:
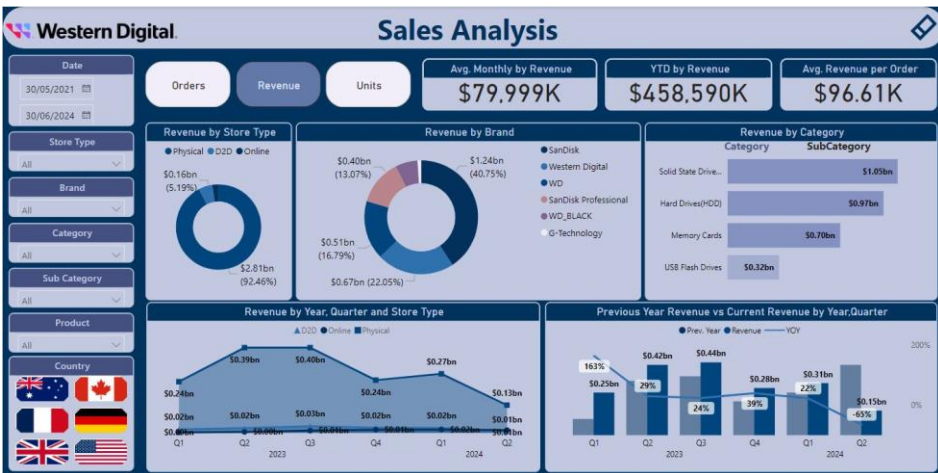
Units State:

Customers Analysis:

This report was created for the customers department to better understand Western Digital's customer behavior to achieve the department's goals.

KPI Cards:
- Active Customers
- New Customers
- New Customers %
- Average Orders per Customer
- Average Revenue per Customer
- Average Units per Customer

Graphs:
- Number of Customers by Store Type
- Number of Customers by Number of Orders
- Top 5 Selling Products by Number of Customers
- % Customers by Countries

Slicers:
- Date
- Store Type
- Brand
- Category
- Sub Category
- Product Name
- Country

6.3 Published to Power BI Service

After creating the dashboard and the reports in Power BI Desktop, they were published to Power BI Service, and an app was created -

Western Digital Sales DM Final Project– Odelia Hochman - Link.
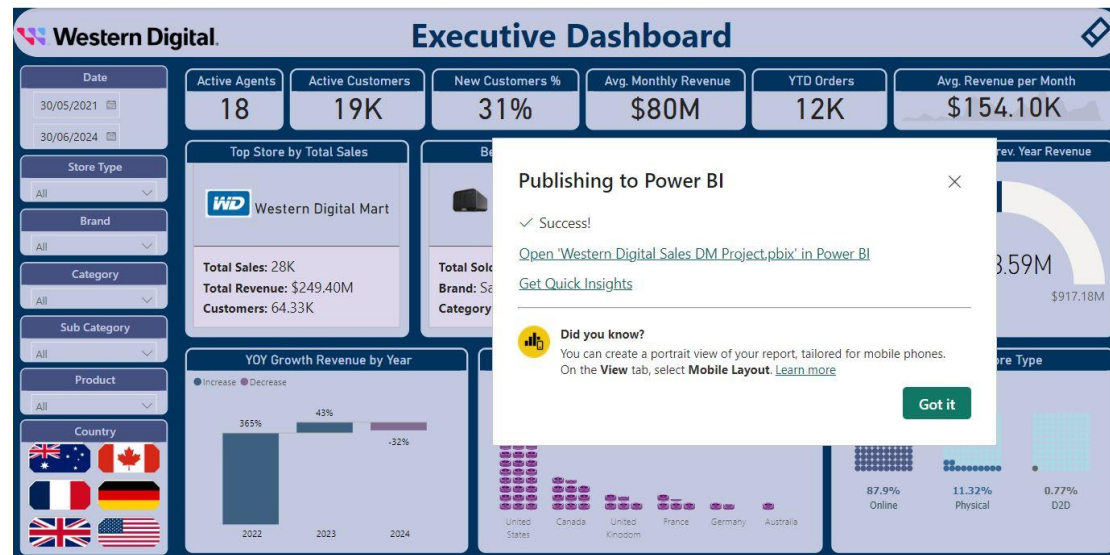
6.4 Data refresh processes

The data is refreshed daily at 5:00:00 (after the refresh of the data mart occurs), for this purpose, a personal gateway was created: