Somatic’s Sales Data Mart

Date: 14/07/2024

Version: 4.00

Written by: Yanai Bardosh

Content

1.General

1.1 Purpose of the Project

The objective of this project is to develop a comprehensive BI solution for Somatix's sales department to support the growth of SafeBeing™ sales. This project is tailored to meet the KPIs of Somatix's sales department and aims to enhance the company's overall ROI.

Somatix is an AI-powered digital health company founded on the belief that all patients deserve equitable, affordable, high-quality healthcare. Their technology is designed to detect early behavioral and emotional indicators of declining health and provide actionable clinical insights. Using its unique and patented gesture detection technology, Somatix's digital health platform enables providers to remotely and passively track for risk factors of adverse events, including poor medical compliance, falls, and dehydration. This novel, yet highly intuitive approach simplifies provider workflows and supports connected care through real-time remote patient monitoring and telemedicine capabilities.

The Data Mart will be created using data from the PriorityERP database, Somatix's operational database. This solution will include summarized data tables focusing on company sales and customer data. Additionally, the BI solution will feature customized reports on sales analysis, customer analysis, and an executive dashboard. Tailored to the needs of the sales department, these reports will contribute to the growth of Somatix's SafeBeing™ sales.

1.2 Project Contents

The project includes the building of a Data Mart which contains sales data. The data will be transferred through an ETL process from the PriorityERP operational database to the Data Mart – SomatixSalesDM.

1.2.1. The Data Mart will include 1 fact table and 4 dimension tables, and 1 history table:

• **Fact_Sales** – Data regarding all sales, including the id of the order, products bought, quantities, and prices.

• **Dim_Person_Customers** – Data regarding the company's Person customers.

• **Dim_Location_Customers** – Data regarding the company's Location customers(Hospitals, Research Centers and rehab Centers).

• **Dim_Sales_Person** – Data regarding the company's Salespersons.

• **Dim_Products** - Data regarding the company's products.

• **Dim_Customers_History** – Historic data regarding the company's Person Customers.

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

1.2.2. The project will include a Transfer Table. This table will allow us to monitor data movement by showing the number of rows transferred and the time taken for each package in the ETL process.

1.2.3. The reports will include data visualizations in Power BI that will support the project's objective in the following ways:

- **Sales Analysis:**

The Sales Analysis Dashboard will offer a detailed overview of the company's sales activities, providing insights into overall revenue generation, performance across different products, and sales trends over time. It will highlight the contributions of individual salespersons. The dashboard will also include Location Customers sales distribution, helping to identify strong and weak location's type. This view will support the sales team in understanding sales dynamics and optimizing their strategies to boost performance.

- **Customer Analysis:**

The Customer Analysis Dashboard will focus on understanding the customer base and their behaviors. It will present demographic data, purchasing patterns, and customer segmentation to reveal who the customers are and how they interact with the company's products. The dashboard will also track customer engagement levels, identifying active and inactive customers, and providing insights into customer acquisition trends. This analysis will help the sales team tailor their approaches to meet customer needs better and enhance customer satisfaction and loyalty.

- **Executive Dashboard:**

The Executive Dashboard will provide a high-level summary of the company's overall performance, focusing on strategic insights and trends. It will offer a snapshot of sales performance, key growth indicators, and ROI metrics, allowing executives to quickly assess the company's performance. This dashboard will support strategic planning and drive the company's growth and success.

2.Gantt

I used ClickUp Platform to create my Gant, if you don't have an account please use my account: Email - Yanaib1215@Gmail.com, Password - Tsv1860m. The Gantt is in the following **Link**

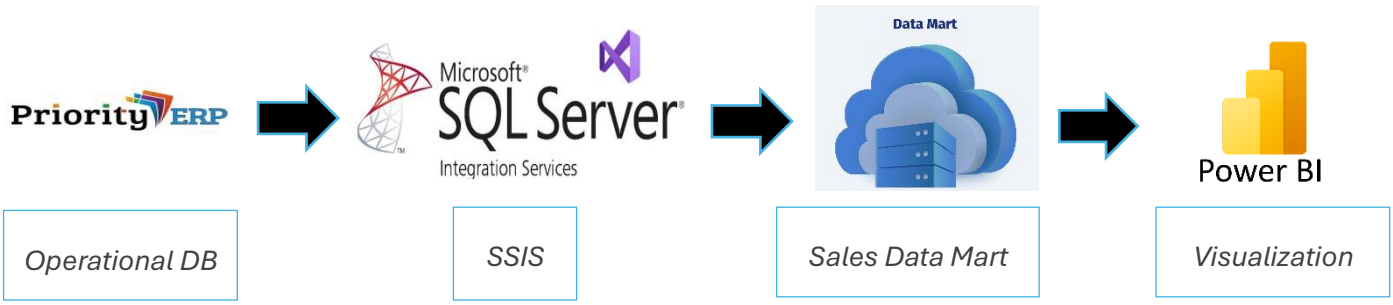
3.Technical Characterization

3.1. Prerequisites

SQL SERVER	ERP system in the operational DB (PriorityERP)
SSIS	ETL Process Using SSIS In Visual Studio
Data refresh processes	Definition of JOBS in SSMS
POWER BI	Creating repots and dashboard using Power BI

3.2. Solution Architecture

3.2.1. High Level Design:



4.Functional Characterization

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

4.1.1. Source To Target link

A total of 11 tables will be used from the operational database.

4.1.2. ERD link

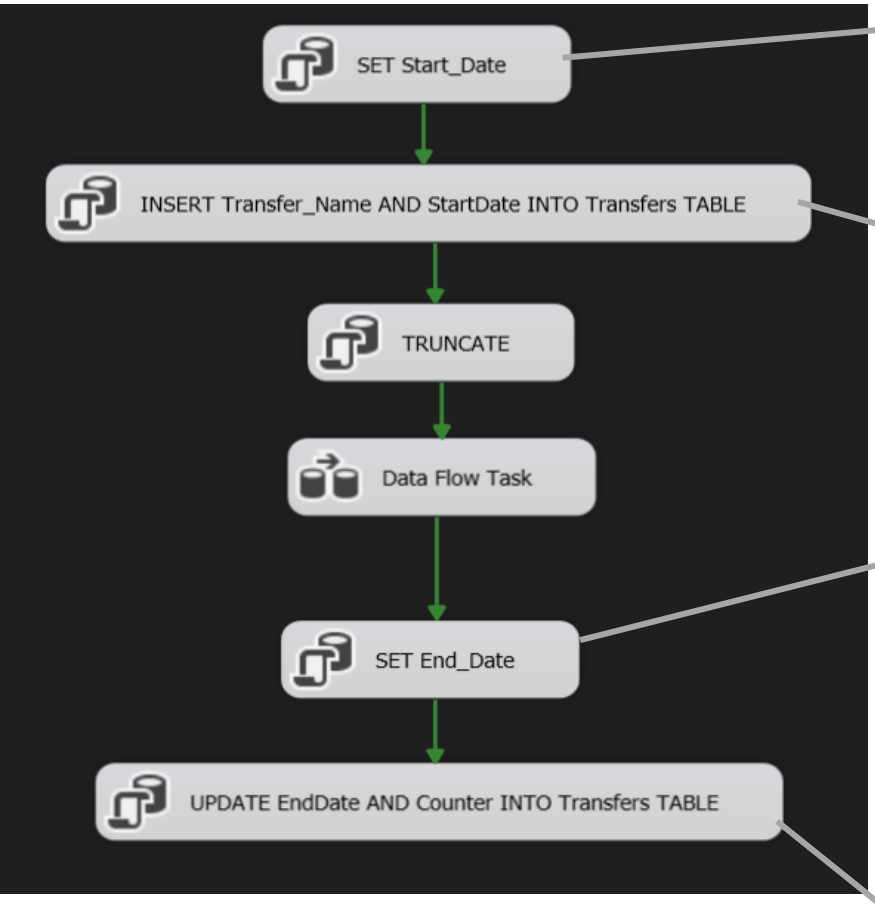
4.2. ETL processes

- **Transfer Table**

To monitor the ETL process, I created a Transfer table that documents each data insert, including which table was updated, how many rows were inserted, the start and end times, and the run duration of the package. The tasks and transformations responsible for the updates are included in all the packages.

Example from STG_Person_Customers package:

In the control flow user variable startdate is set in the first task with GETDATE() as startdate, an insert statement is executed in the second task, inserting the values of the user variables: name, startdate, counter (which is updated in the data flow), user variable enddate is set in the fifth task with GETDATE() as enddate. and an update statement is executed in the last task.




```
SELECT GETDATE() AS startdate
```

Result Name	Variable Name
startdate	User::startdate

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

Variable Name	Direction	Data Type	Parameter Name	Parameter Size
User::name	Input	VARCHAR	NewParameterName	0
User::startdate	Input	DATE	NewParameterName	1
User::counter	Input	LONG	NewParameterName	2

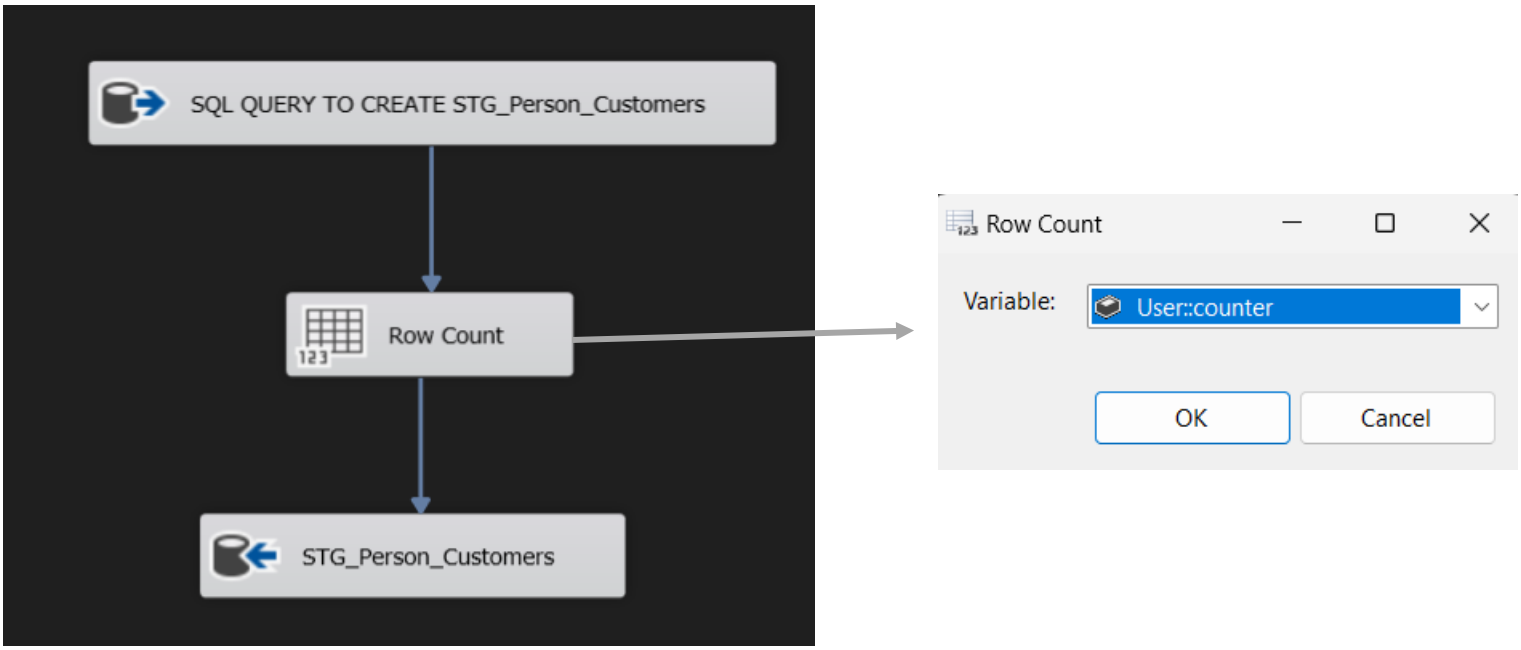
```
SELECT GETDATE() AS enddate
```

Result Name	Variable Name
enddate	User::enddate

```
UPDATE Transfers
SET End_Date = ?,
    Counter = ?,
 Total_Runing_Time = DATEDIFF(SECOND,?,?)
WHERE Transfer_Name = ?
AND Transfer_ID = (SELECT MAX(Transfer_ID)
                   FROM Transfers
                   WHERE Transfer_Name = ?)
```

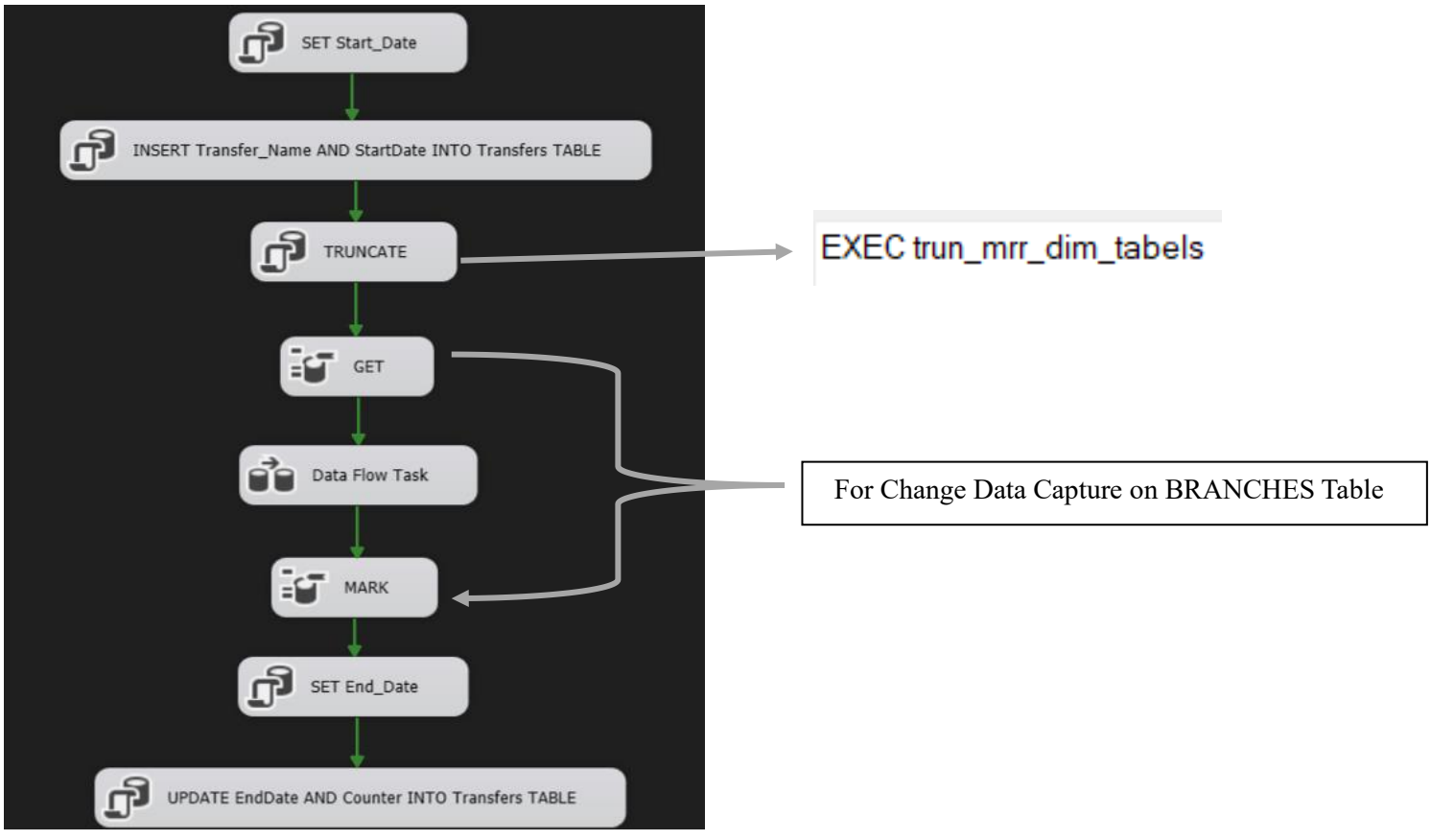
Variable Name	Direction	Data Type	Parameter Name	Parameter Size
User::enddate	Input	DATE	NewParameterName	0
User::counter	Input	LONG	NewParameterName	1
User::startdate	Input	DATE	NewParameterName	2
User::enddate	Input	DATE	NewParameterName	3
User::name	Input	NVARCHAR	NewParameterName	4
User::name	Input	NVARCHAR	NewParameterName	5

In the data flow the user variable counter is updated using a Row Count transformation.



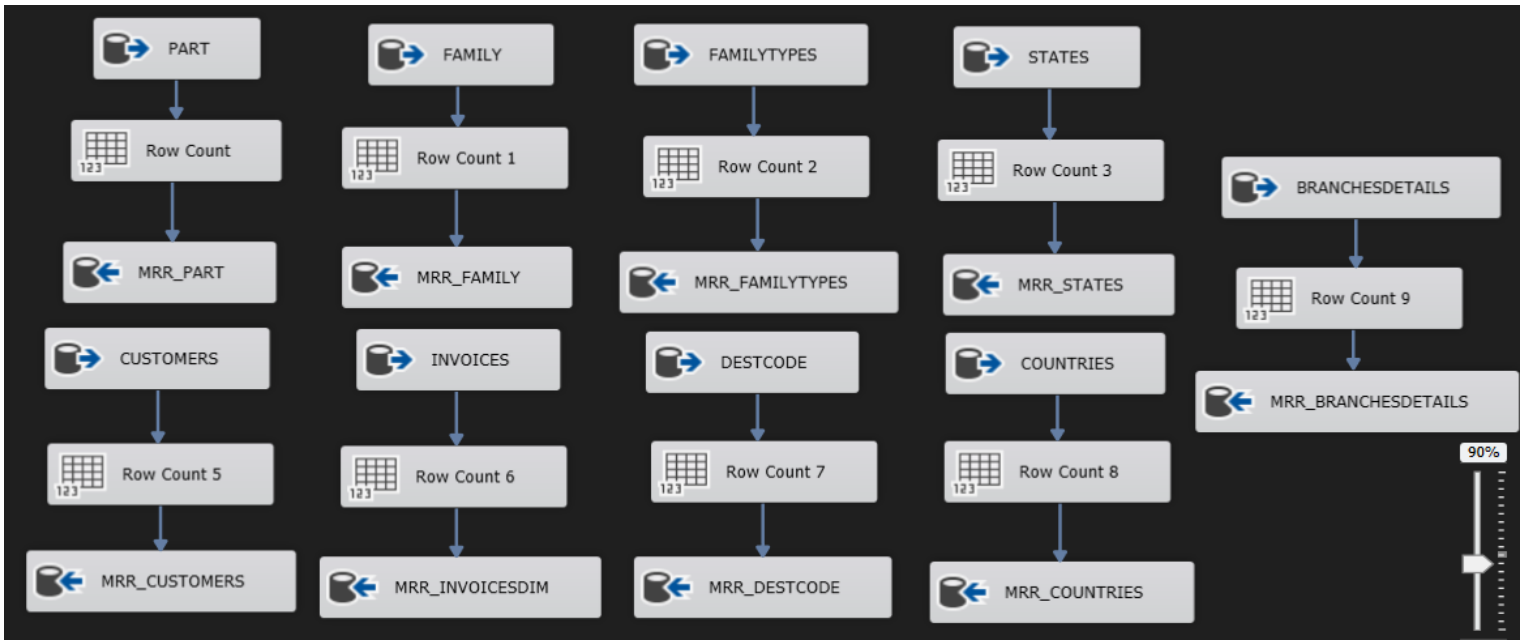
- **MRR_DIM_Tabels Package**

This package is responsible for loading data from PriorityERP tables to all mirror tables relevant for the dim tables (10 tables in total). All the tables in this package are truncated using a stored procedure.



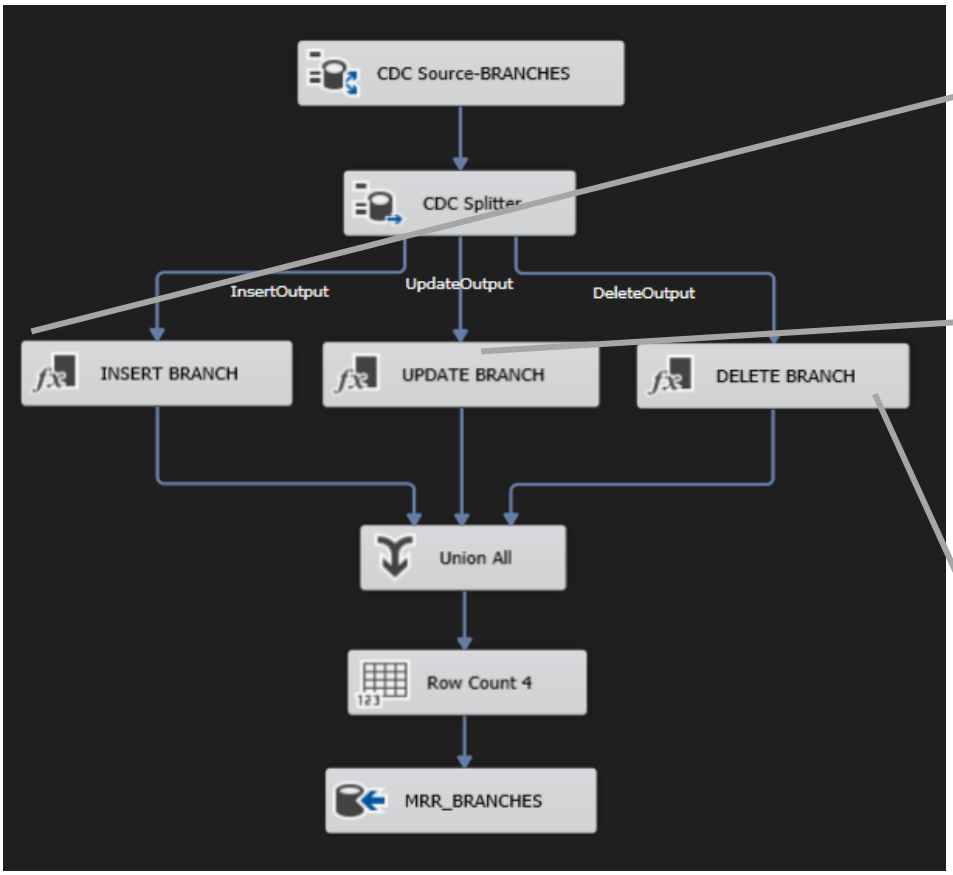
In the Data Flow Task:

Regular MRR



Change Data Capture on BRANCHES Table

Change data capture (CDC) refers to the process of identifying and capturing changes made to data in BRANCHES table and then delivering those changes in the ETL process.

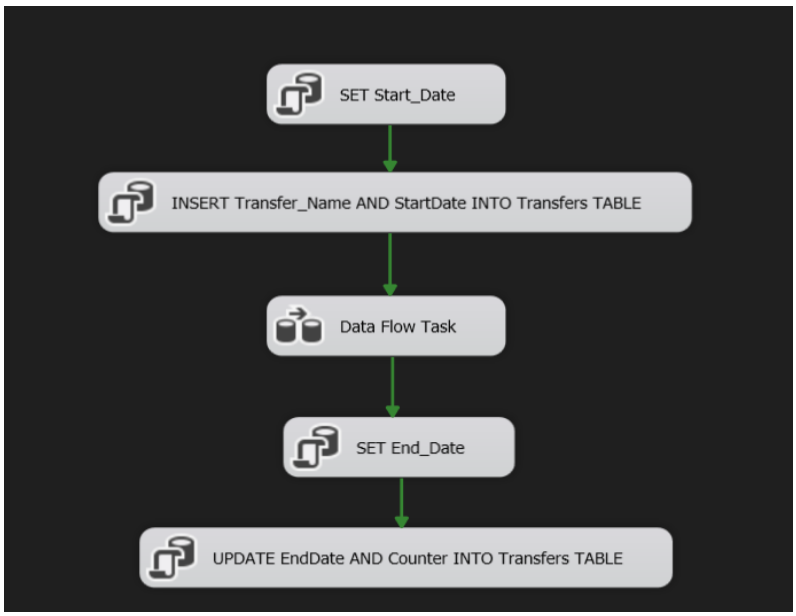


Derived Column Name	Derived Column	Expression
Status	<add as new column>	1

Derived Column Name	Derived Column	Expression
Status	<add as new column>	2

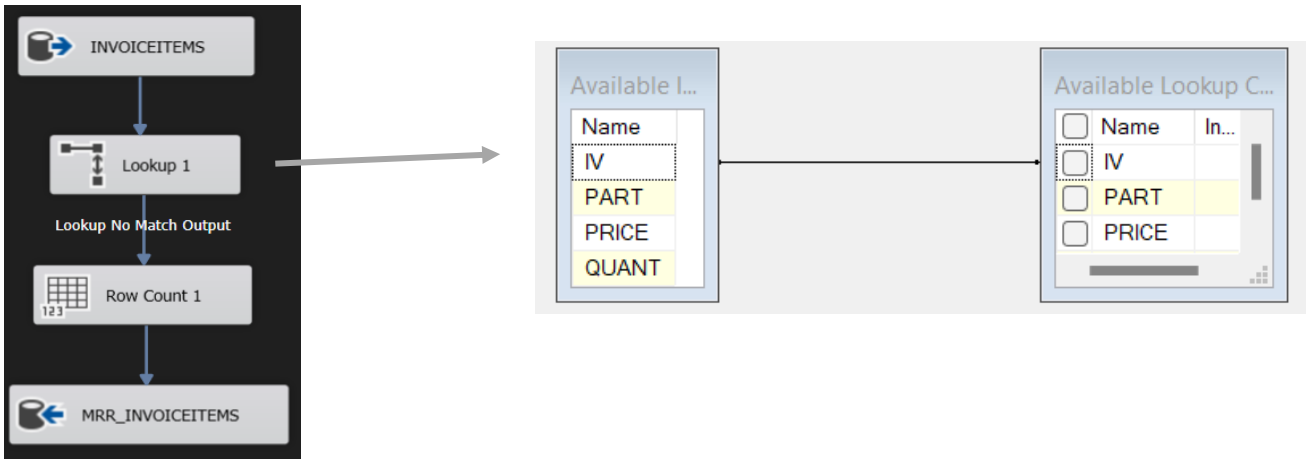
Derived Column Name	Derived Column	Expression
Status	<add as new column>	3

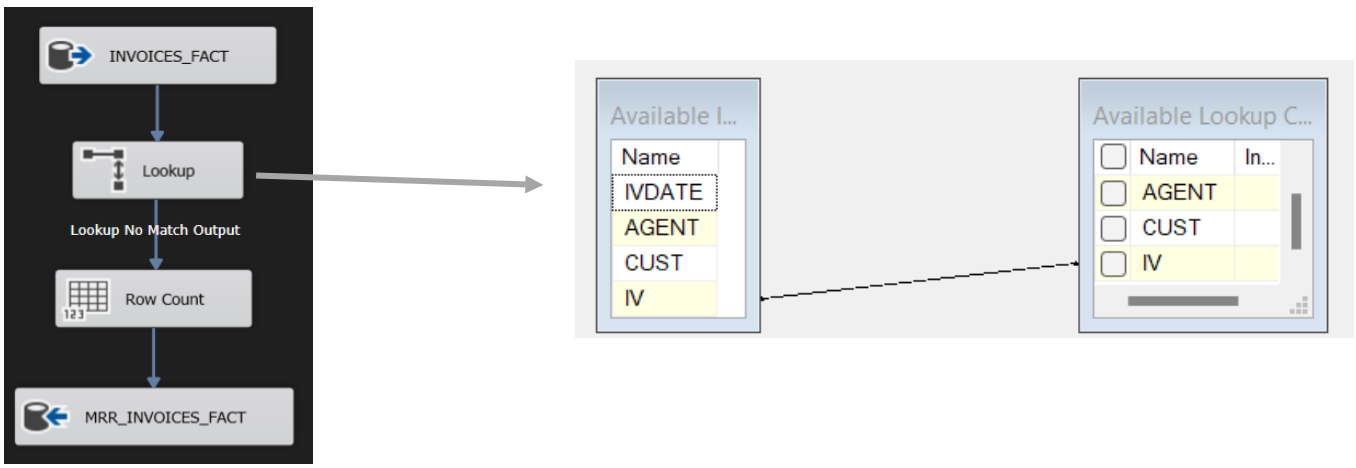
- **MRR_FACT_Tabels Package**



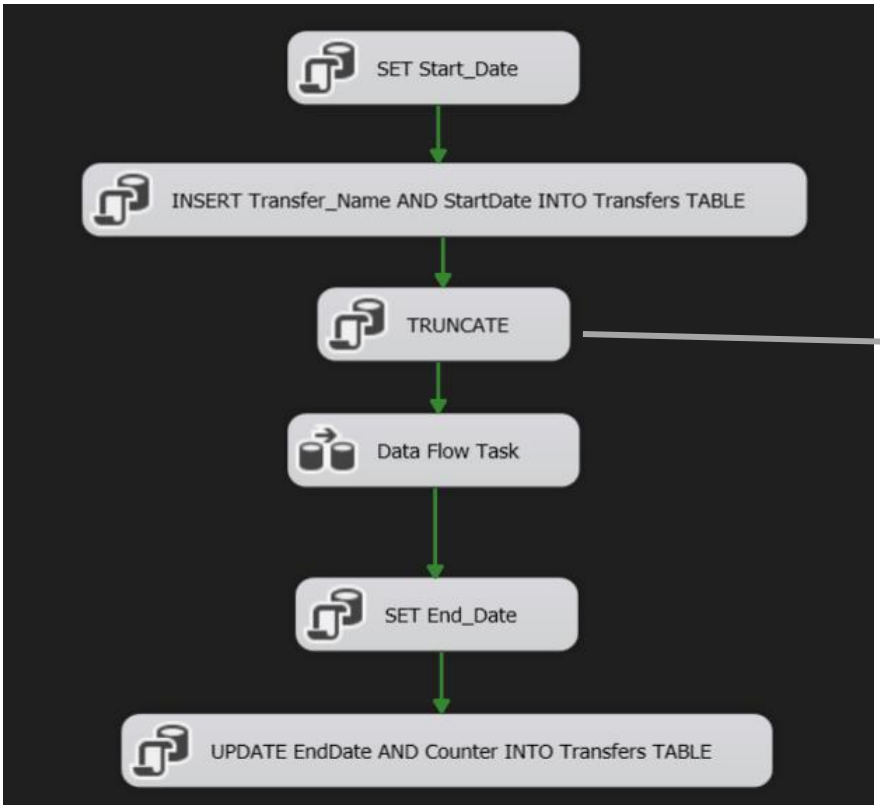
In the Data Flow Task:

Loading data into MRR_INVOICES_FACT and MRR_INVOICEITEMS is performed incrementally using the Lookup transformation, which compares records to existing entries in MRR_INVOICES_FACT and MRR_INVOICEITEMS. The tables are not truncated during this process.



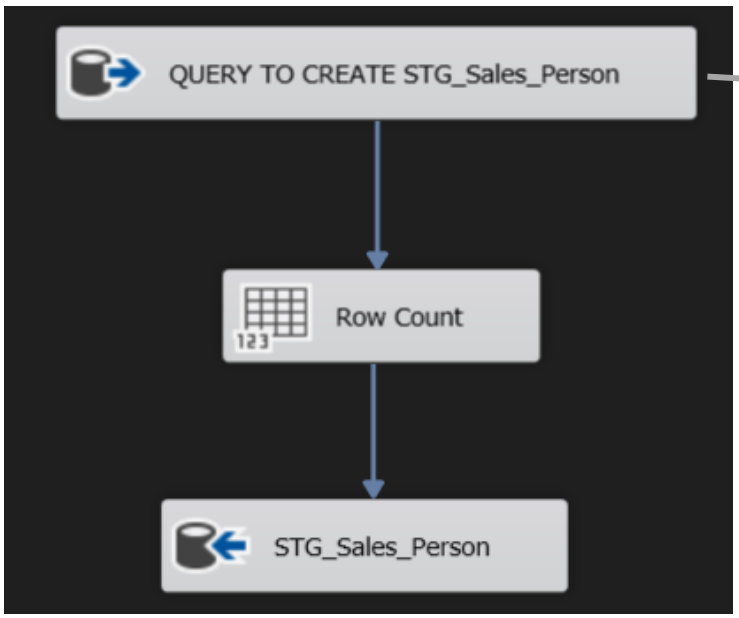


• **STG_Sales_Person Package**




```
TRUNCATE TABLE STG_Sales_Person
```

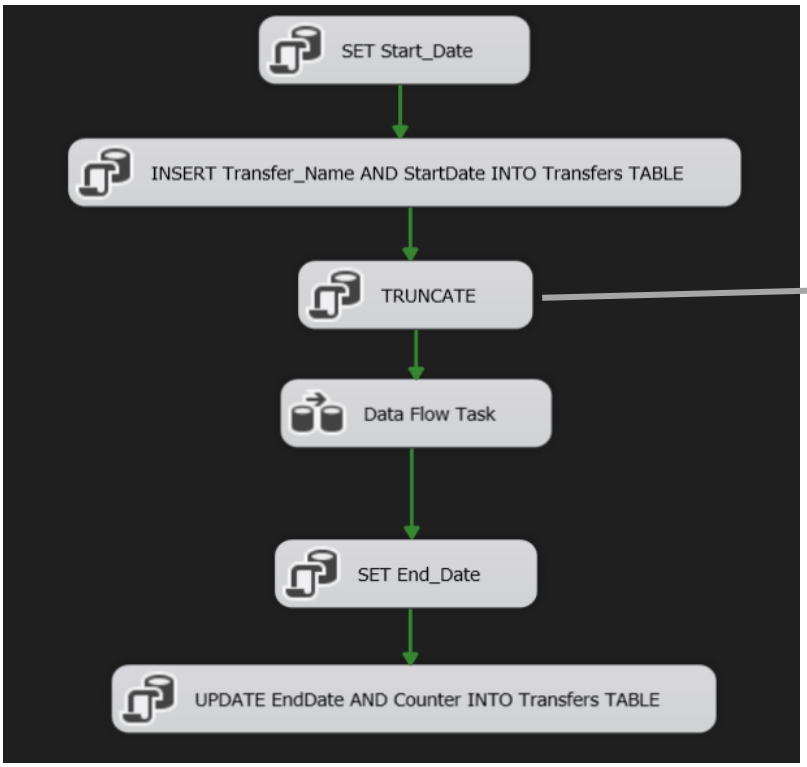
In the Data Flow Task:



SQL command text:

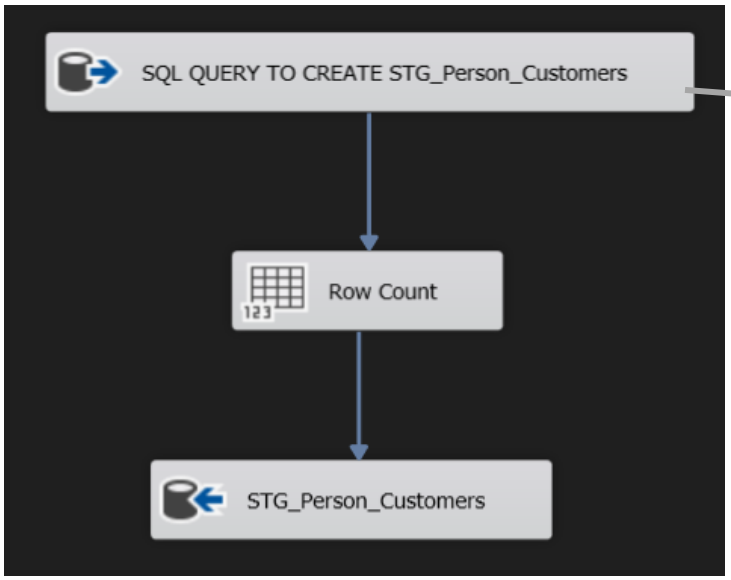
```
SELECT DISTINCT i.AGENT, c.CUSTNAME
FROM MRR_INVOICESDIM i
LEFT JOIN MRR_CUSTOMERS c ON c.CUST = i.AGENT
```

• **STG_Person_Customers Package**




```
TRUNCATE TABLE STG_Person_Customers
```

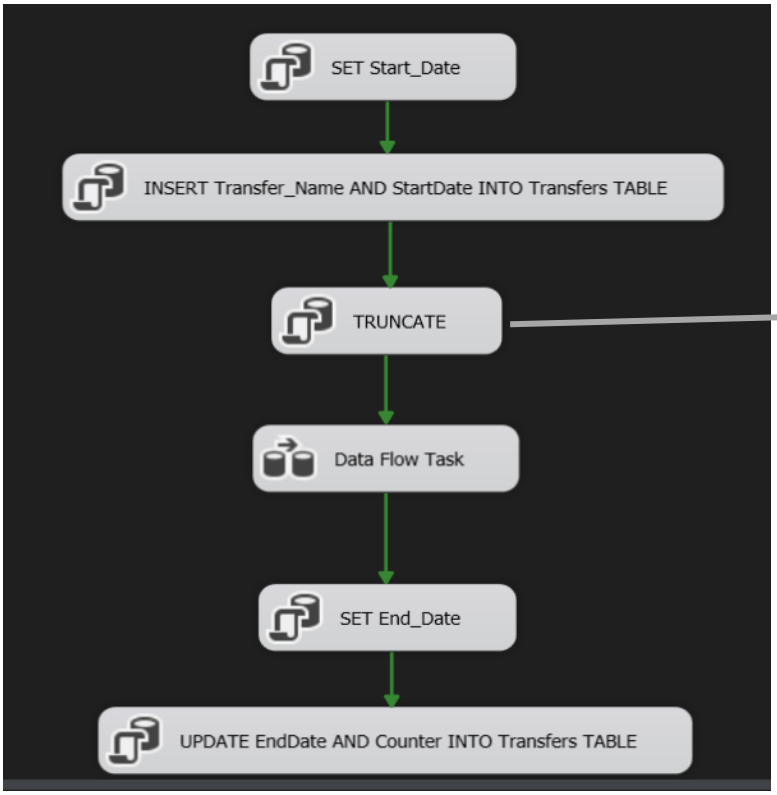
In the Data Flow Task:



SQL command text:

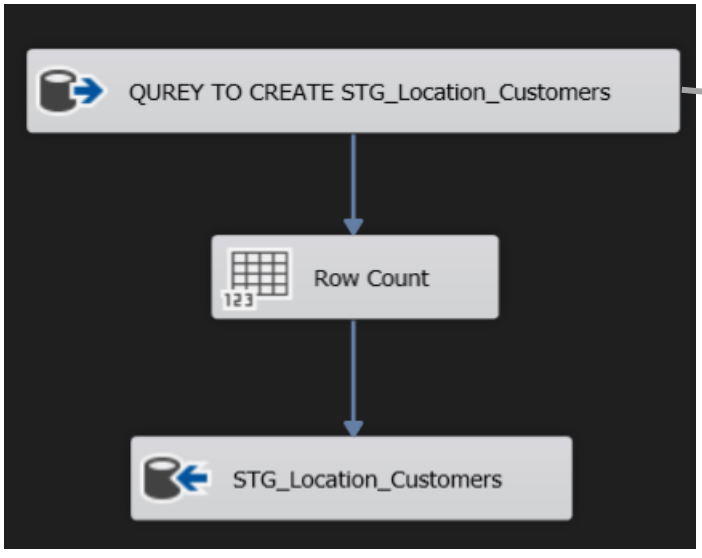
```
SELECT c.CUST, c.CUSTNAME,MIN(DATEADD
(YEAR,10,DATEADD(DAY,i.IVDATE,'1/1/1900'))) AS
Registration_Date, d.ADDRESS,
d.CITY,s.STATENAME,co.COUNTRY
FROM MRR_CUSTOMERS c
LEFT JOIN MRR_INVOICESDIM i ON c.CUST = i.CUST
LEFT JOIN MRR_DESTCODE d ON d.DESTCODE = i.DESTCODE
LEFT JOIN MRR_STATES s ON d.STATE = s.STATEID
LEFT JOIN MRR_COUNTRIES co ON s.COUNTRY =
co.COUNTRYCODE
WHERE c.CUST NOT IN(SELECT DISTINCT i1.AGENT
                    FROM MRR_INVOICESDIM i1)
GROUP BY c.CUST, c.CUSTNAME, d.ADDRESS,
d.CITY,s.STATENAME,co.COUNTRY
```

• **STG_Location_Customers Package**




```
TRUNCATE TABLE STG_Location_Customers
```

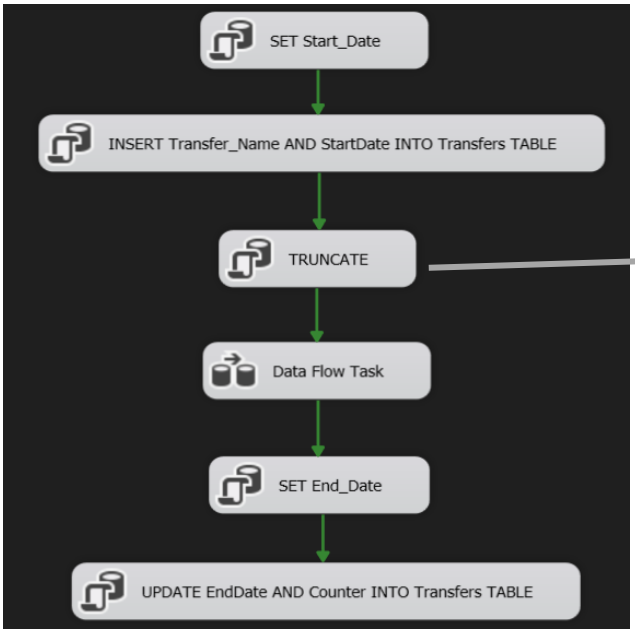
In the Data Flow Task:



SQL command text:

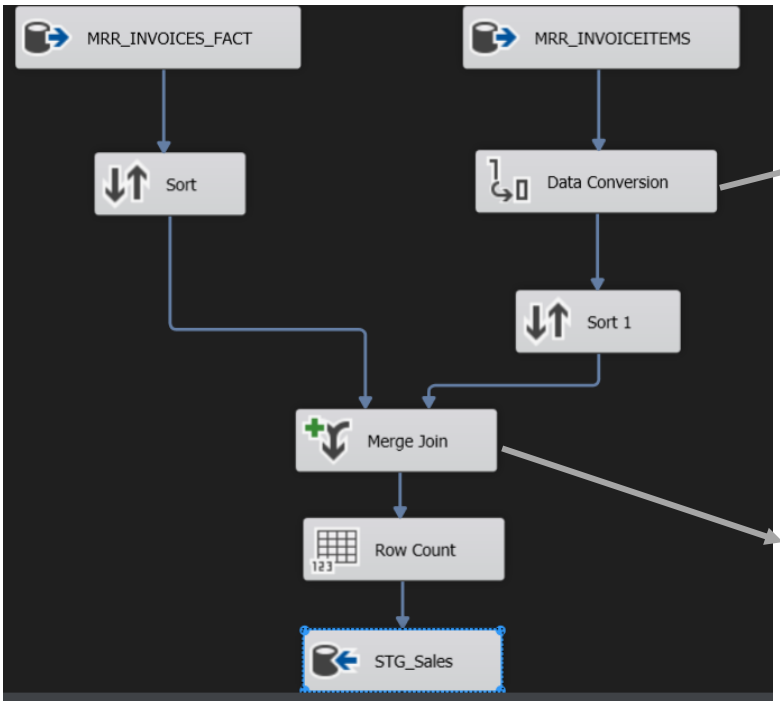
```
SELECT
b.BRANCH,bd.Name,bd.Address,bd.City,bd.State,bd.Country,b.A
GENT,bd.Monthly_Rate,bd.Registration_Date,bd.Location_Type
FROM MRR_BRANCHES b
INNER JOIN MRR_BRANCHESDETAILS bd ON b.BRANCH =
bd.BRANCH
```

• **STG_Sales Package**

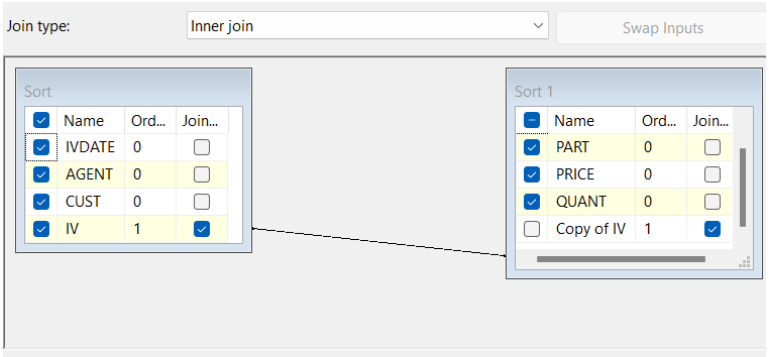


TRUNCATE TABLE STG_Sales

In the Data Flow Task:



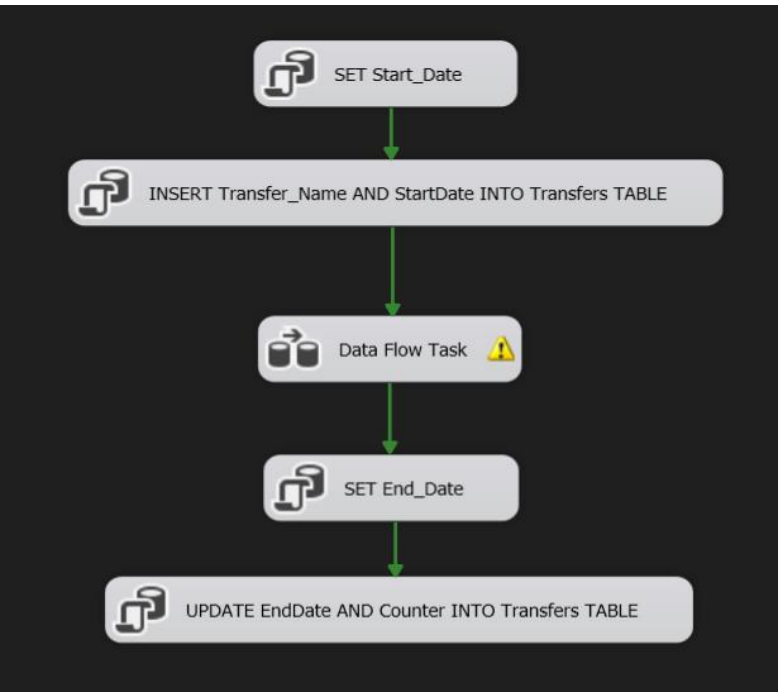
Input Column	Output Alias	Data Type
IV	Copy of IV	four-byte signed integer [DT_I4]



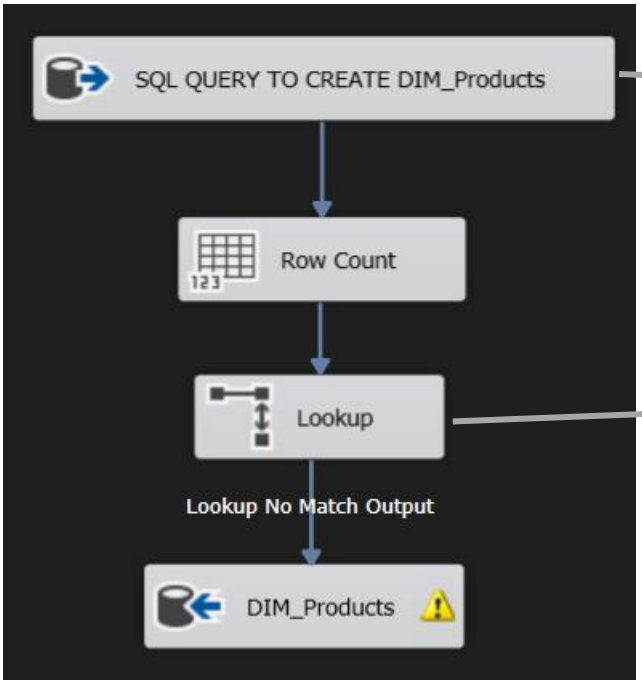
Input	Input Column	Output Alias
Sort	IV	IV
Sort	CUST	CUST
Sort	AGENT	AGENT
Sort	IVDATE	IVDATE
Sort 1	QUANT	QUANT
Sort 1	PRICE	PRICE
Sort 1	PART	PART

• **DM_Products Package**

Loading data into DIM_Products is performed incrementally using the Lookup transformation, which compares records to existing entries in DIM_Products. The table is not truncated during this process.

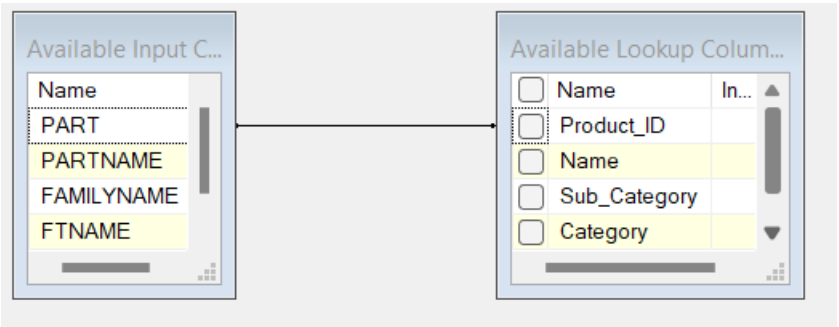


In the Data Flow Task:



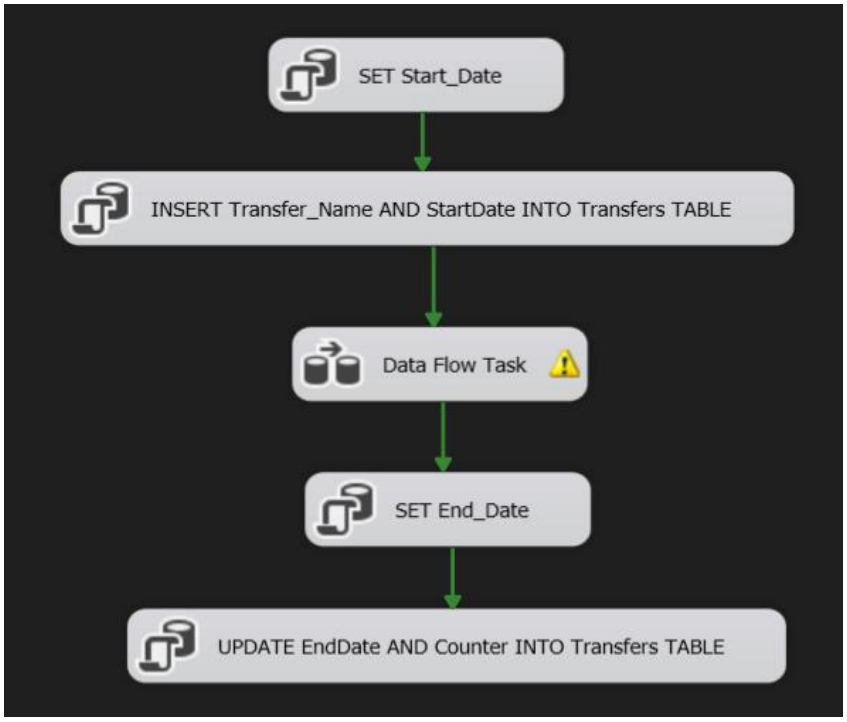
SQL command text:

```
SELECT p.PART, p.PARTNAME, f.FAMILYNAME, ft.FTNAME
,p.PRICE
FROM MRR_PART p
INNER JOIN MRR_FAMILY f ON p.FAMILY = f.FAMILY
INNER JOIN MRR_FAMILYTYPES ft ON f.FAMILYTYPE =
ft.FAMILYTYPE
```



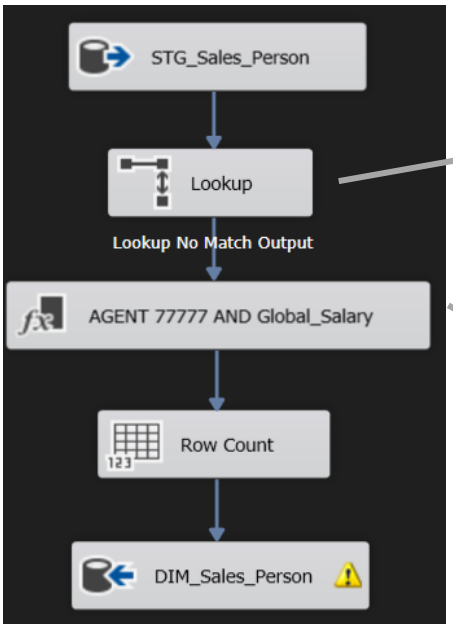
• **DM_Sales_Person Package**

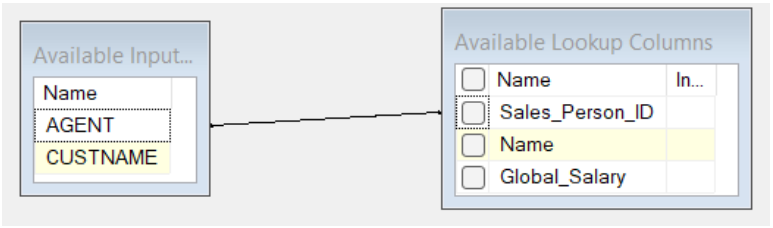
Loading data into DIM_ Sales_Person is performed incrementally using the Lookup transformation, which compares records to existing entries in DIM_ Sales_Person. The table is not truncated during this process.



In the Data Flow Task:

I used the Derived Column Transform to calculate the global salary for each worker and to assign a name to agent 77777.

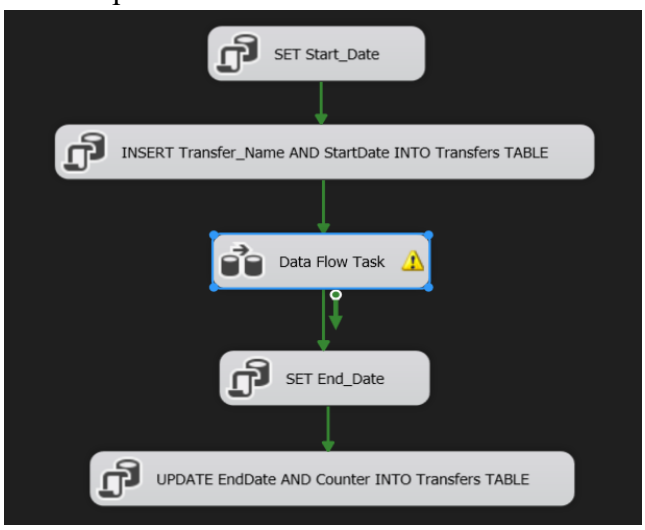




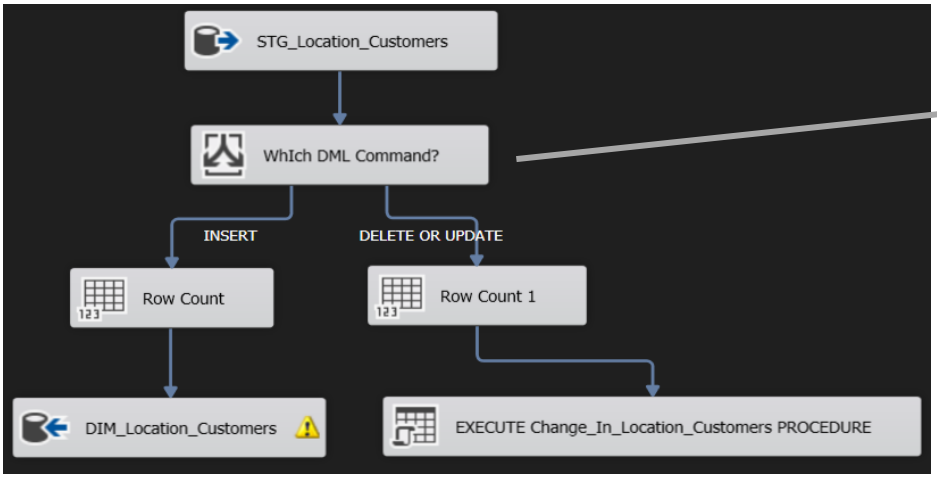
Derived Column Name	Derived Column	Expression
Global_Salary	<add as new column>	AGENT == 77777 ? AGENT * 0.05 : AGENT * 8.2
CUSTNAME	Replace 'CUSTNAME'	REPLACENULL(CUSTNAME,"James Bond")

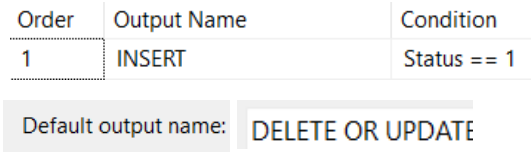
• **DM_Location_Customers Package**

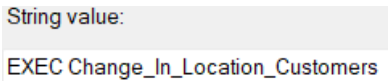
Loading data into DIM_Location_Customers is performed using CDC with the Split Condition transformation. When Status = 1, a regular insert is performed; otherwise, it is an update or delete. I used a stored procedure to execute this process. The table is not truncated during this operation.



In the Data Flow Task:





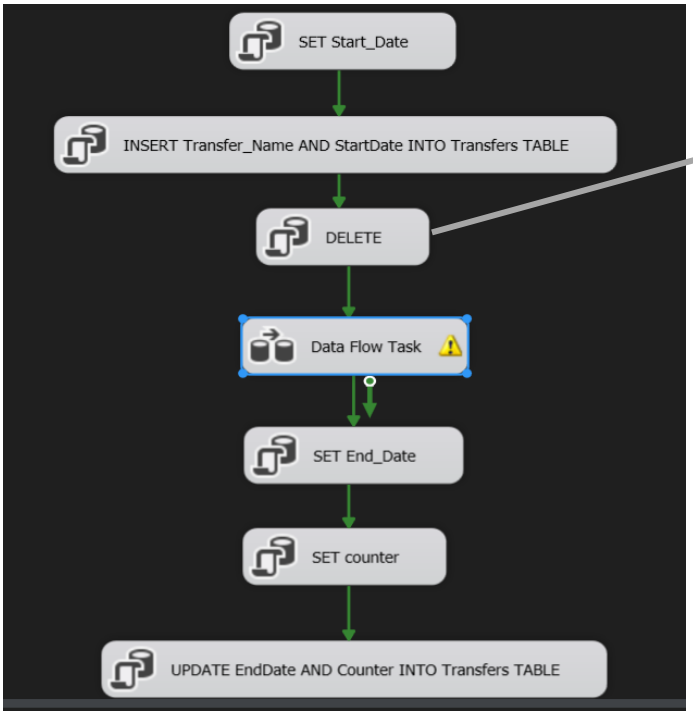



```
CREATE PROCEDURE Change_In_Location_Customers
AS
BEGIN
MERGE DIM_Location_Customers AS [Target]
USING  STG_Location_Customers AS [Source]
ON  [Target].[Location_Customers_ID] = [Source].[BRANCH]
WHEN MATCHED AND [Source].[Status] != 1
   THEN UPDATE
      SET [Target].[Sales_Person_ID] = [Source].[AGENT],
          [Target].[Name] = [Source].[Name],
          [Target].[Address] = [Source].[Address],
          [Target].[State] = [Source].[State],
          [Target].[Country] = [Source].[Country],
          [Target].[Monthly_Rate] = [Source].[Monthly_Rate],
          [Target].[Registration_Date] = [Source].[Registration_Date],
          [Target].[Location_Type] = [Source].[Location_Type],
          [Target].[IS_ACTIVE] = CASE
                                 WHEN [Source].[Status] = 2 THEN 1
                                 WHEN [Source].[Status] = 3 THEN 0
                                 END;

END;
```

• **DM_Person_Customers Package**

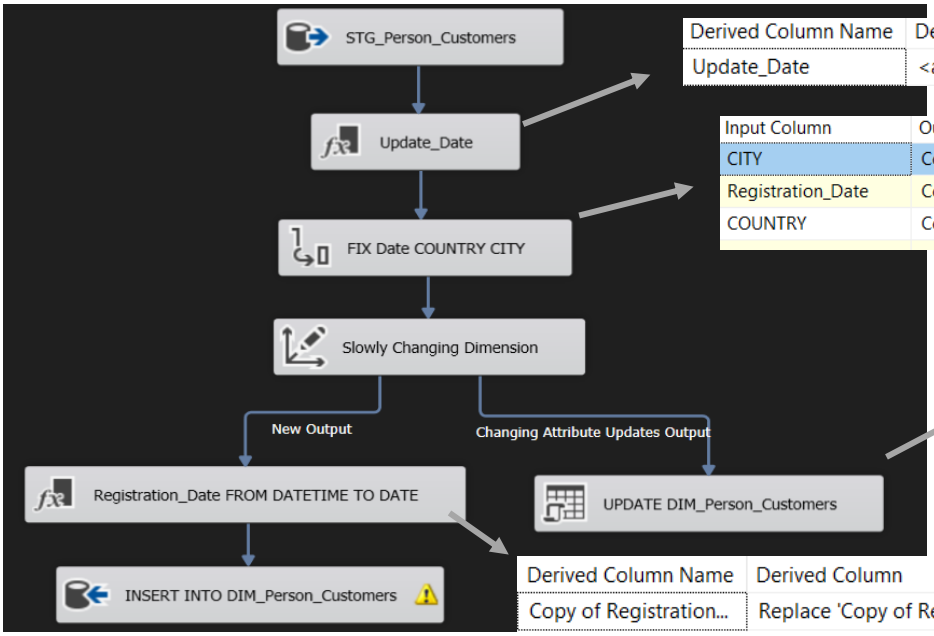
Loading data into DIM_Person_Customers is performed using an incremental load with the Slowly Changing Dimension transformation (change type: Changing Attribute). To deleted entries, I created a stored procedure that updates the IS_Active column in DIM_Person_Customers to 0.




```
EXEC DELETE_In_Person_Customers
```

```
CREATE PROCEDURE DELETE_In_Person_Customers
AS
BEGIN
  MERGE DIM_Person_Customers AS [Target]
  USING STG_Person_Customers  AS [Source]
  ON  [Target].[Person_Customer_ID] = [Source].[CUST]
  WHEN NOT MATCHED BY SOURCE AND [Target].[IS_Active] != 0
  THEN UPDATE
       SET [Target].[IS_Active] = 0,
       [Target].[Update_Date] = GETDATE();
END;
```

In the Data Flow Task, I created the Update_Date column using the Derived Column transformation and converted the Registration_Date, City, and Country column types using the Data Conversion transformation. In addition, Update statement created to update the new values in DIM_Person_Customers



Derived Column Name	Derived Column	Expression
Update_Date	<add as new column>	(DT_DBTIMESTAMP)@[User::startdate]

Input Column	Output Alias	Data Type	Length
CITY	Copy of CITY	Unicode string [DT_WSTR]	40
Registration_Date	Copy of Registration...	Unicode string [DT_WSTR]	50
COUNTRY	Copy of COUNTRY	Unicode string [DT_WSTR]	50

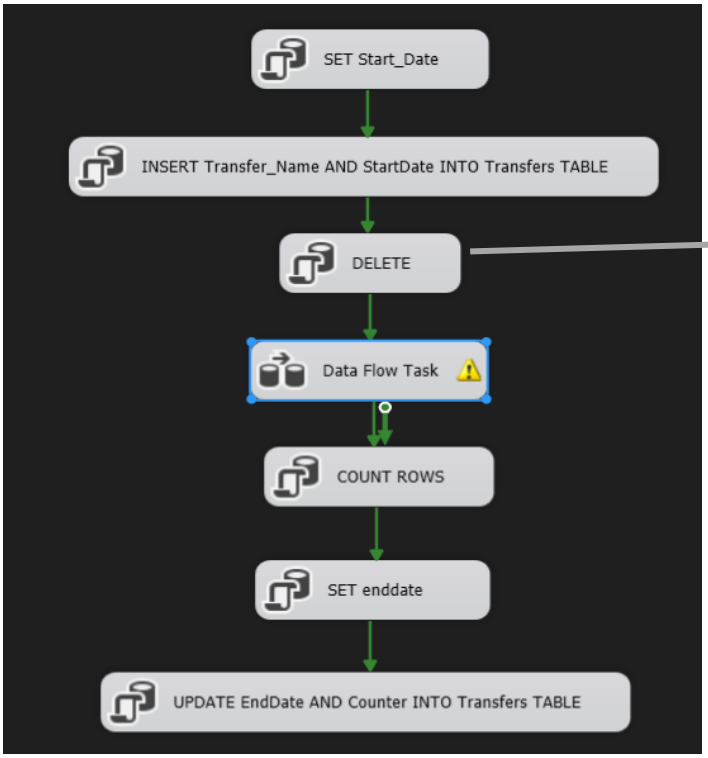
String value:

```
UPDATE [dbo].[DIM_Person_Customers]
SET [Address] = ?,
    [City] = ?,
    [Country] = ?,
    [Name] = ?,
    [Registration_Date] = ?,
    [State] = ? ,
    [Update_Date] = ?
WHERE [Person_Customer_ID] = ?
```

Derived Column Name	Derived Column	Expression
Copy of Registration...	Replace 'Copy of Re...	LEFT([Copy of Registration_Date],10)

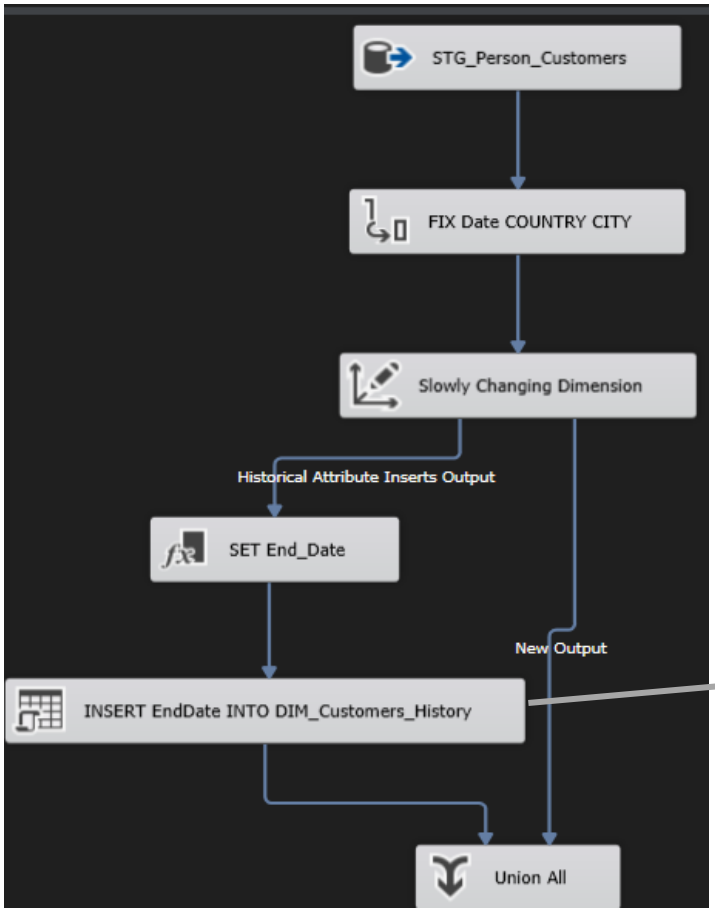
• **DM_Customers_History Package**

Loading data into DIM_Customers_History is performed using an incremental load with the Slowly Changing Dimension transformation (change type: Historical Attribute). To deleted entries, I created a update statement that updates the End_Date column in DIM_Customers_History to current datetime.



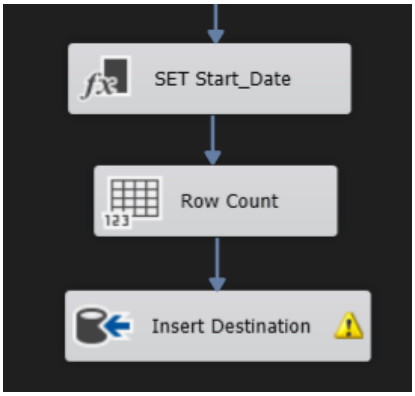

```
UPDATE DIM_Customers_History
SET End_Date = ?
FROM DIM_Customers_History ch
LEFT JOIN STG_Person_Customers c
ON ch.Person_Customer_ID  = c.CUST
WHERE c.CUST IS NULL AND End_Date IS NULL
```

In the Data Flow Task, I created the End_Date and Start_Date columns using the Derived Column transformation and converted the Registration_Date, City, and Country column types using the Data Conversion transformation. In addition, Update statement created to set the End_Date column in the old values.



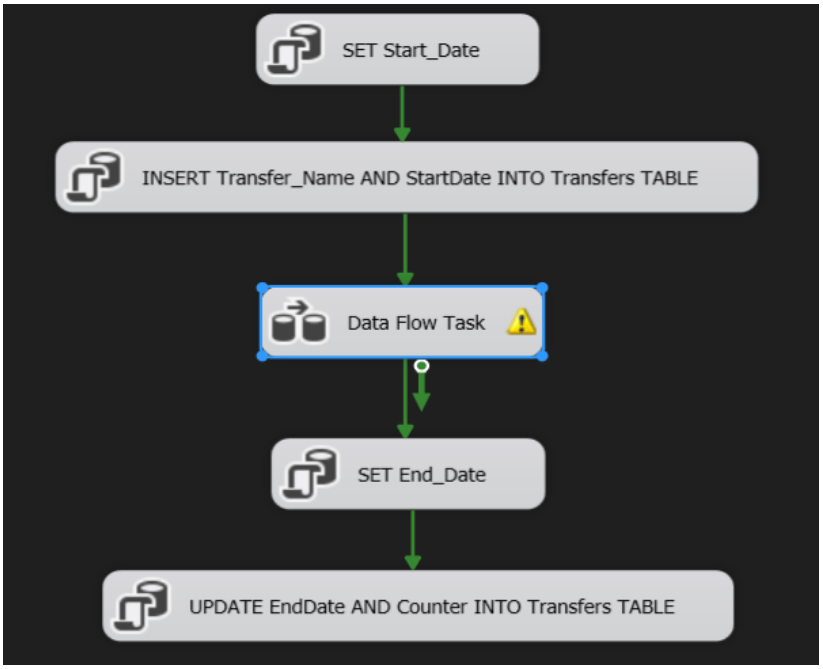
String value:

```
UPDATE [dbo].[DIM_Customers_History]
 SET [End_Date] = ?
WHERE [Person_Customer_ID] = ? AND [End_Date] IS NULL
```

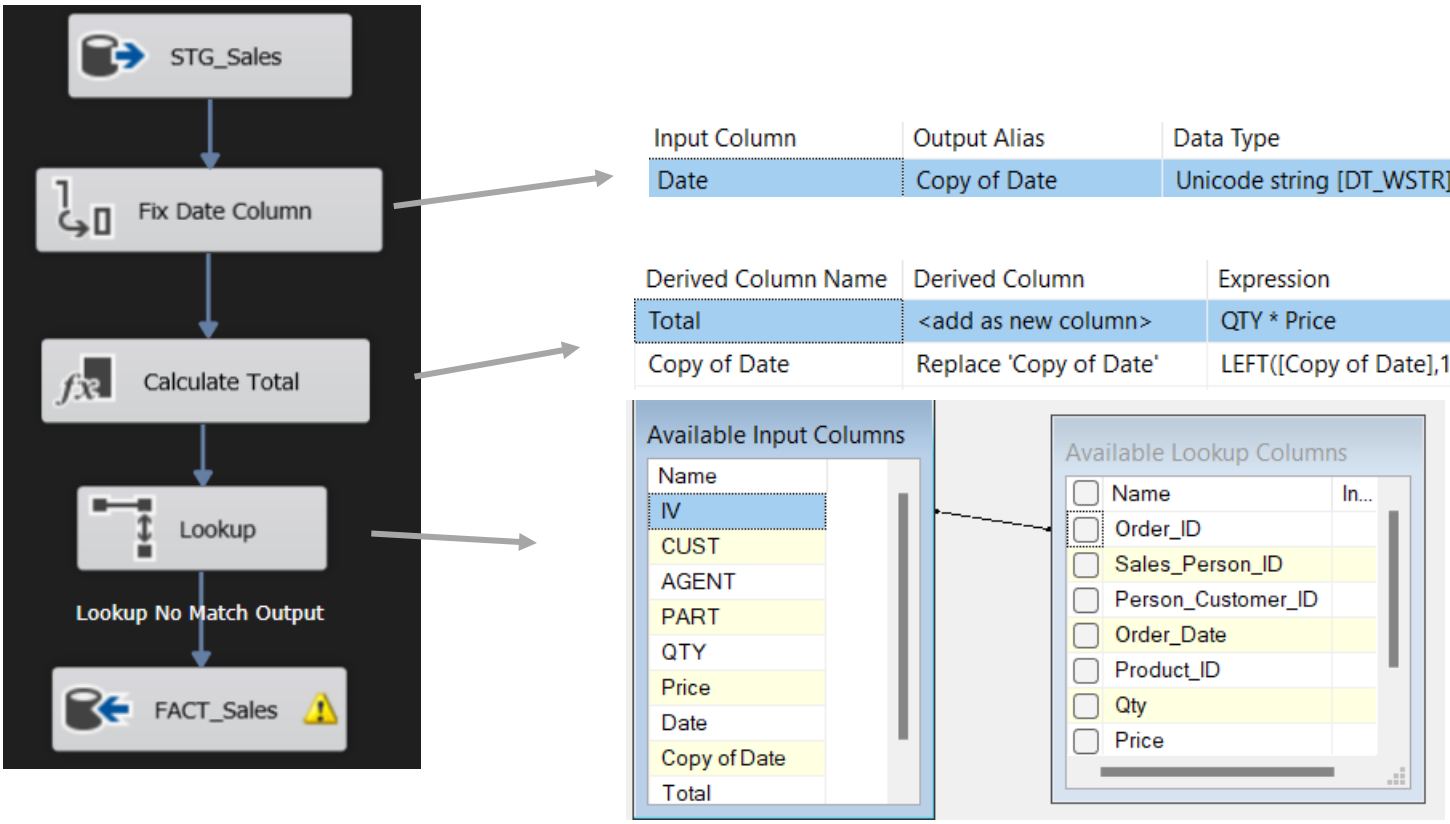


• **DM_Sales Package**

Loading data into FACT_ Sales is performed incrementally using the Lookup transformation, which compares records to existing entries in FACT_ Sales. The table is not truncated during this process. I calculated the Total column using the Derived Column transformation and converted the Date column type using the Data Conversion transformation**.**



In the Data Flow Task:



• **Automatic Processing**

The data is automatically refreshed daily at 00:00:00 using SQL Agent jobs, the first step executed is the RUN_MRR_DIM_TABLES, the second step executed is EUN_MRR_FACT_TABLES and so on.

Job step list:

Step	Name	Type	On Success	On Failure
1	RUN_MRR_DIM_TABLES	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
2	RUN_MRR_FACT_TABLES	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
3	RUN_STG_SALES_PERSON	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
4	RUN_STG_LOCATION_CUSTOME...	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
5	RUN_STG_PERSON_CUSTOMERS	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
6	RUN_STG_SALES	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
7	RUN_DM_PRODUCTS	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
8	RUN_DM_SALES_PERSON	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
9	RUN_DM_LOCATION_CUSTOMERS	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
10	RUN_DM_PERSON_CUSTOMERS	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
11	RUN_DM_CUSTOMERS_HISTORY	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...
12	RUN_DM_SALES	SQL Server Integration Services Package	Go to the next step	Quit the job reporting fail...

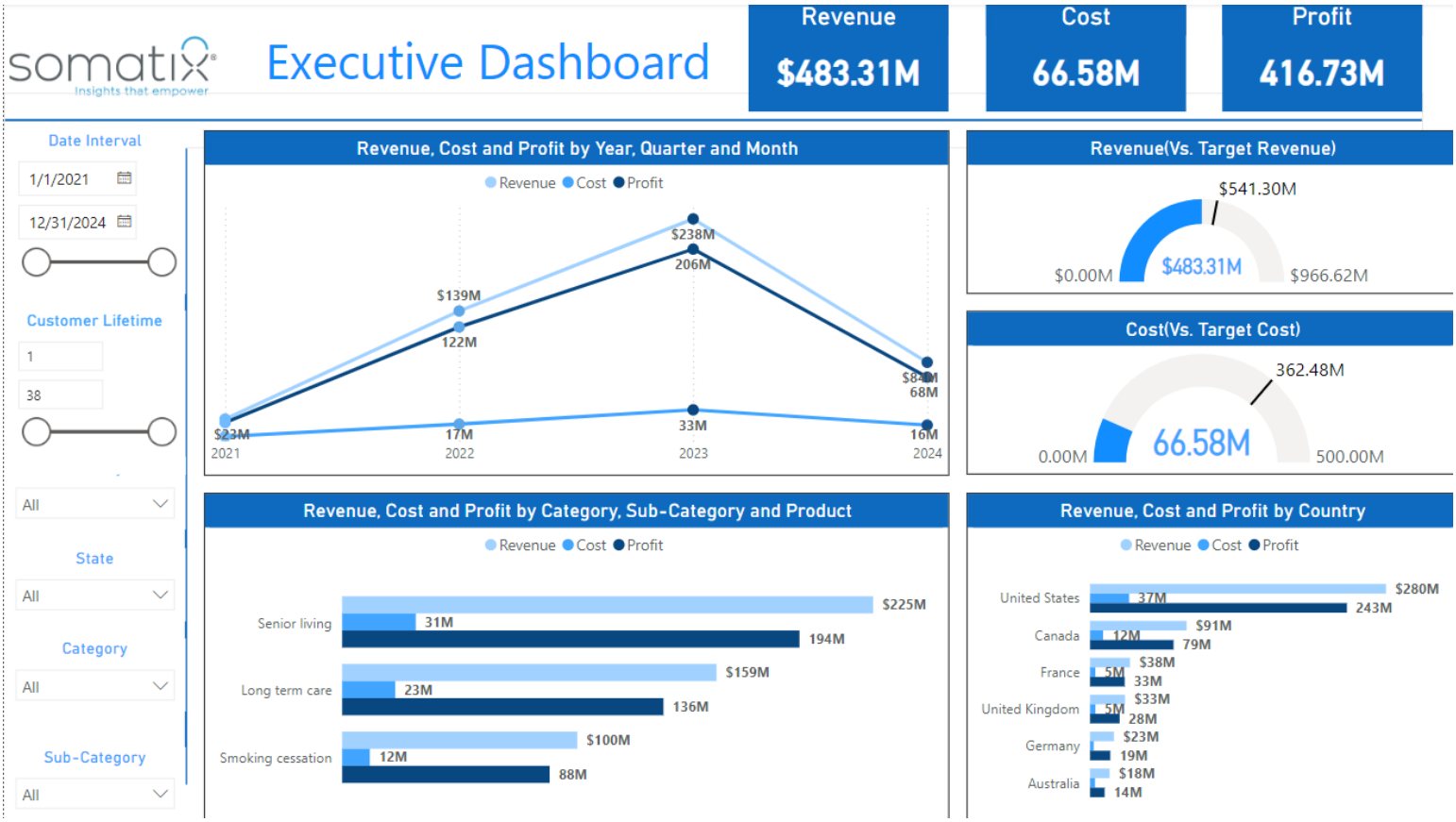
4.3. Visualization in Power BI:

• **Reports**

The project includes 3 reports: Executive Dashboard, Sales Analysis, Customer Analysis.

Executive Dashboard:

This dashboard provides a comprehensive overview of the company's financial performance, highlighting key metrics and performance against targets.



Key Metrics

- Revenue
- Cost
- Profit

Visuals

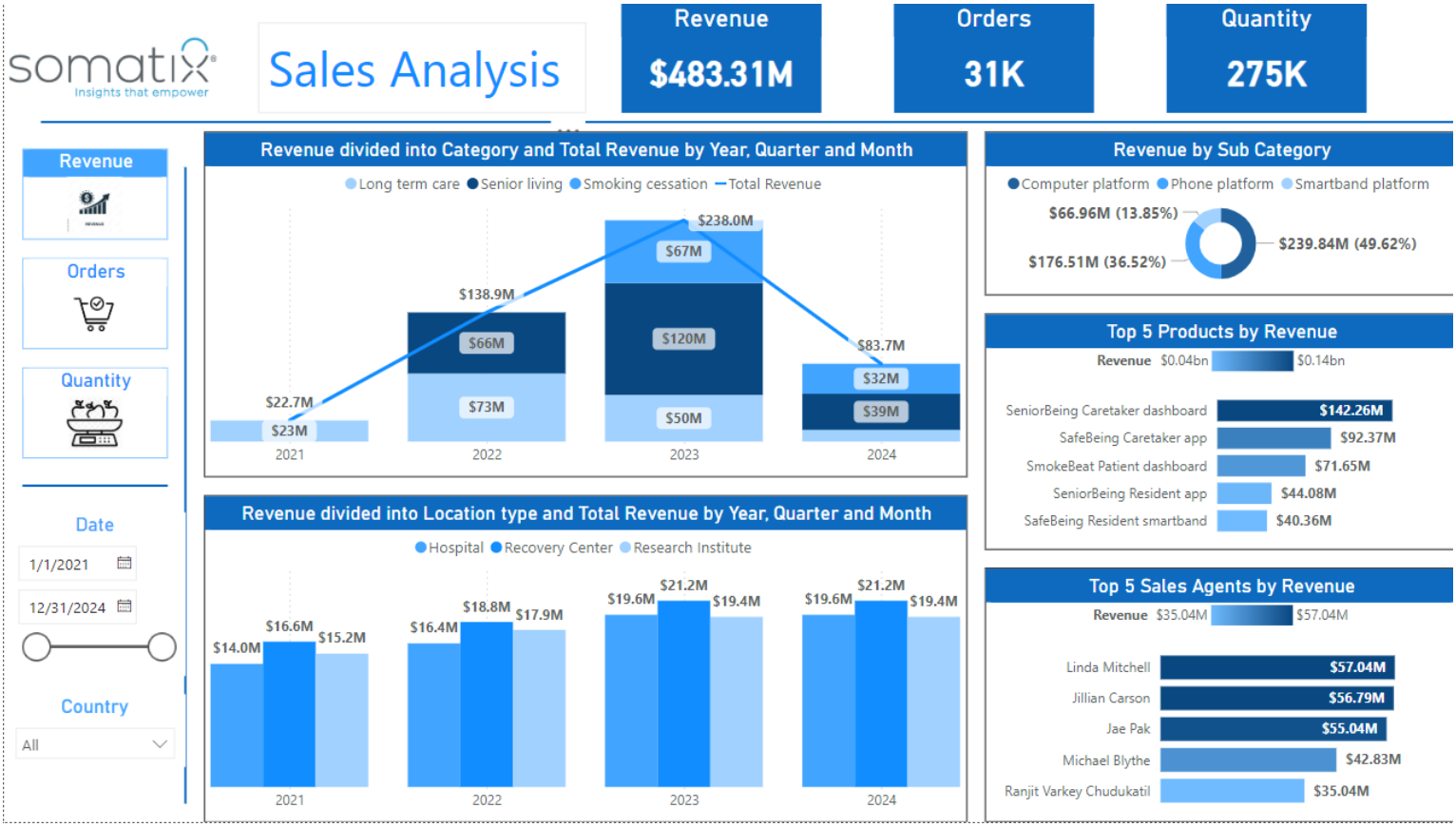
- Revenue, Cost, and Profit by Year, Quarter, and Month
- Revenue (Vs. Target Revenue)
- Cost (Vs. Target Cost)
- Revenue, Cost, and Profit by Category, Sub-Category, and Product
- Revenue, Cost, and Profit by Country

Filters

- Date Interval: Allows selection of data between 1/1/2021 and 12/31/2024.
- Customer Lifetime: Range from 1 to 38.
- State
- Category
- Sub-Category

Sales Analysis:

This dashboard provides an in-depth analysis of the company's sales performance, broken down by various categories, products, locations, and sales agents. It highlights key revenue figures and top-performing segments.



Same for al States(Revenue State, Orders State, Quantity State)

Key Metrics

- Revenue
- Orders
- Quantity

Filters

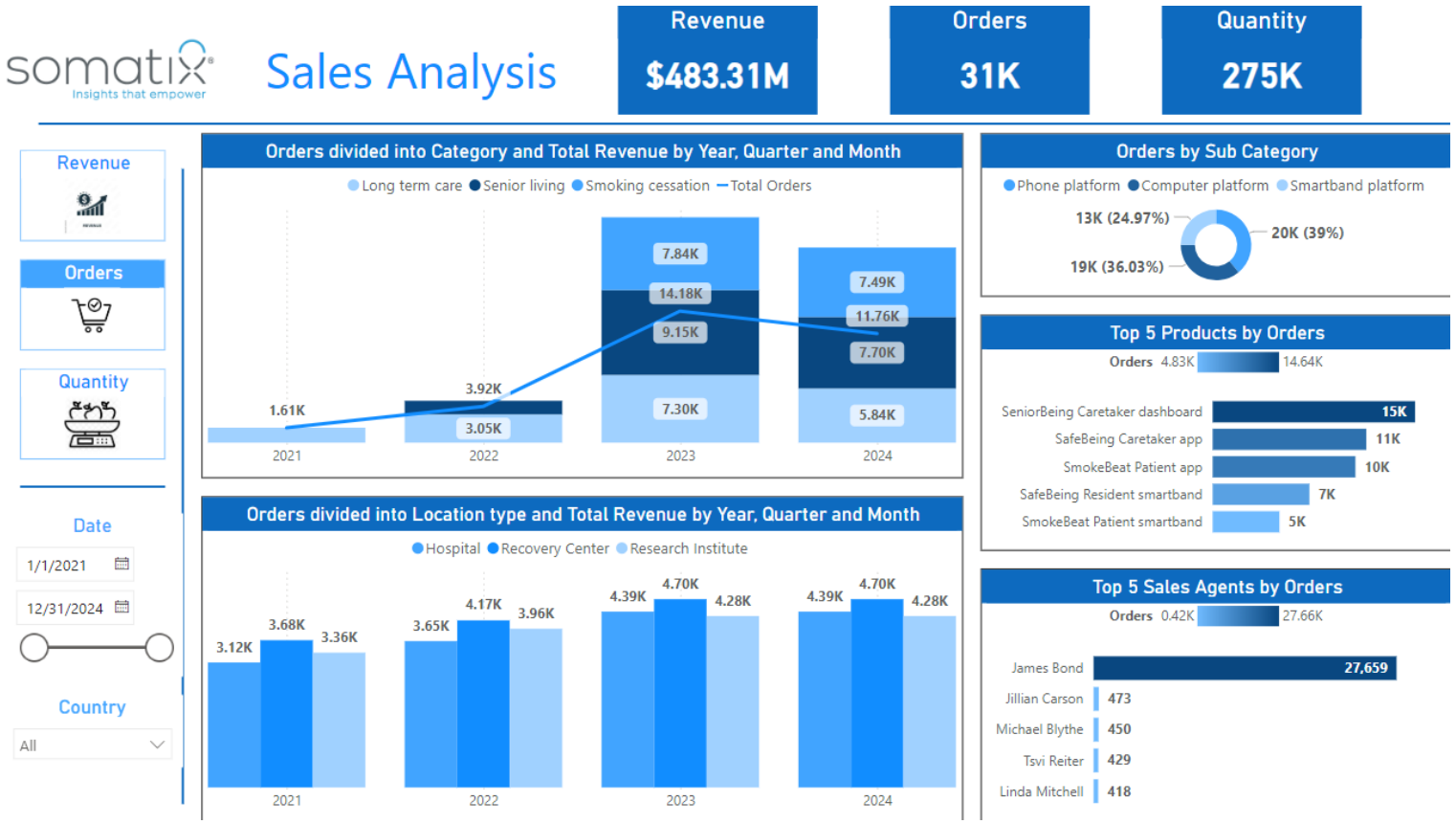
- Date: From 1/1/2021 to 12/31/2024
- Country

Revenue State

Visuals

- Revenue divided into Category and Total Revenue by Year, Quarter and Month
- Revenue by Sub Category
- Top 5 Products by Revenue
- Revenue divided into Location type and Total Revenue by Year, Quarter and Month
- Top 5 Sales Agents by Revenue

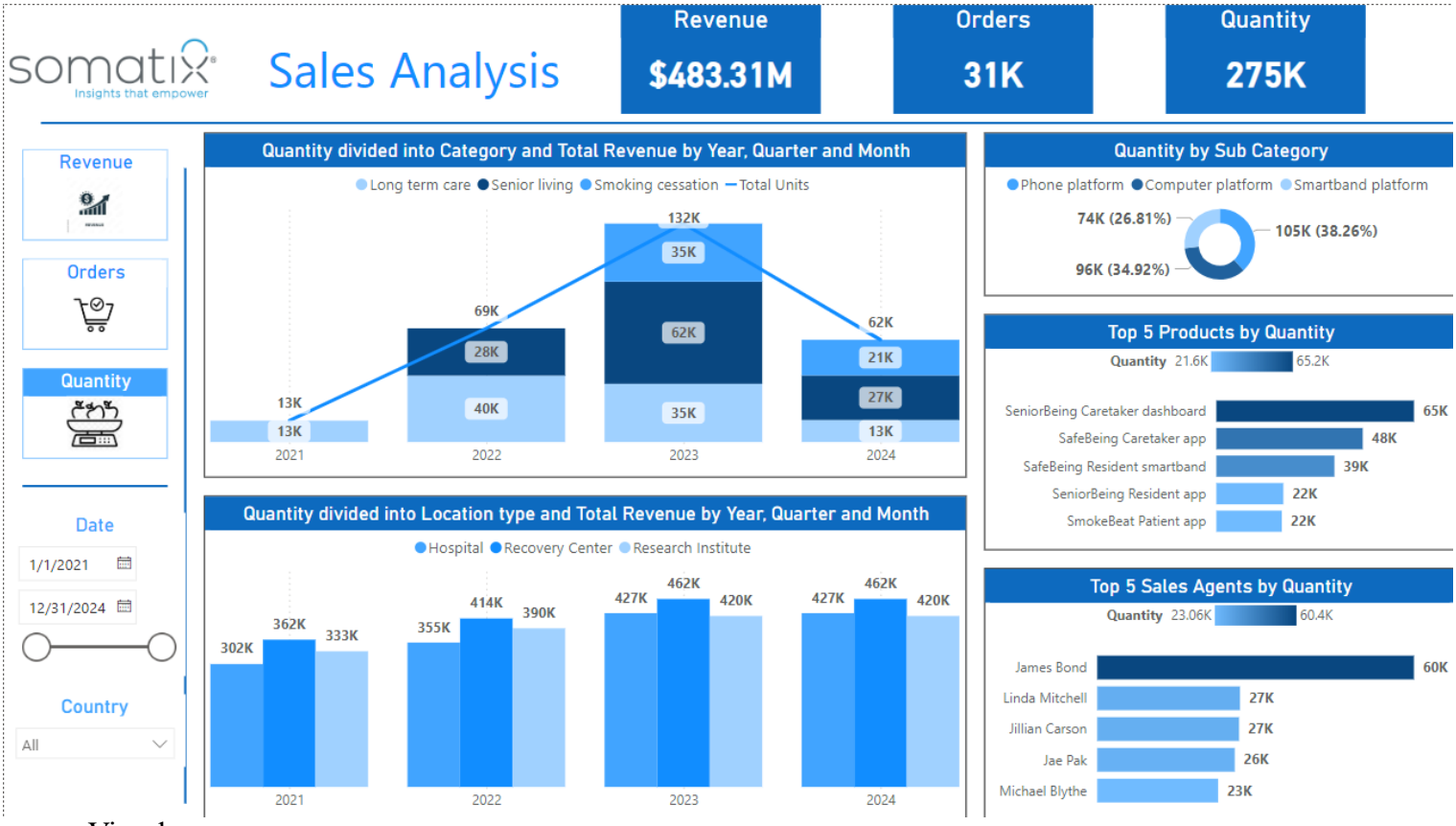
Orders State



Visuals

- Orders divided into Category and Total Revenue by Year, Quarter and Month
- Orders by Sub Category
- Top 5 Products by Orders
- Orders divided into Location type and Total Revenue by Year, Quarter and Month
- Top 5 Sales Agents by Orders

Quantity State

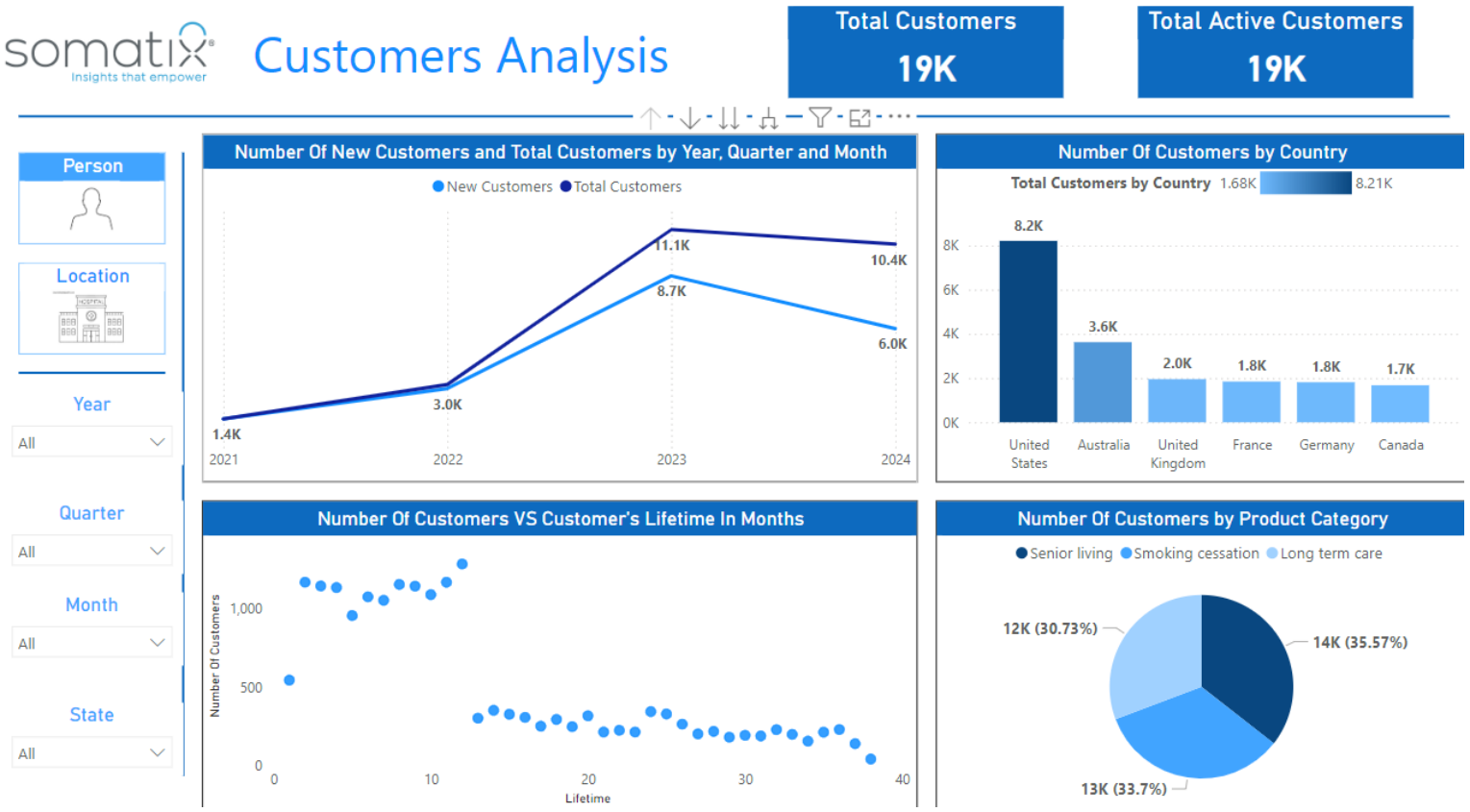


Visuals

- Quantity divided into Category and Total Revenue by Year, Quarter and Month
- Quantity by Sub Category
- Top 5 Products by Quantity
- Quantity divided into Location type and Total Revenue by Year, Quarter and Month
- Top 5 Sales Agents by Quantity

Customers Analysis:

This dashboard provides insights into the customer base, including the number of new and total customers over time, customer distribution by country, and product category preferences. It also includes an analysis of customer lifetime.



Same for al States(Person State, Location State)

Filters

- Year
- Quarter
- Month
- State

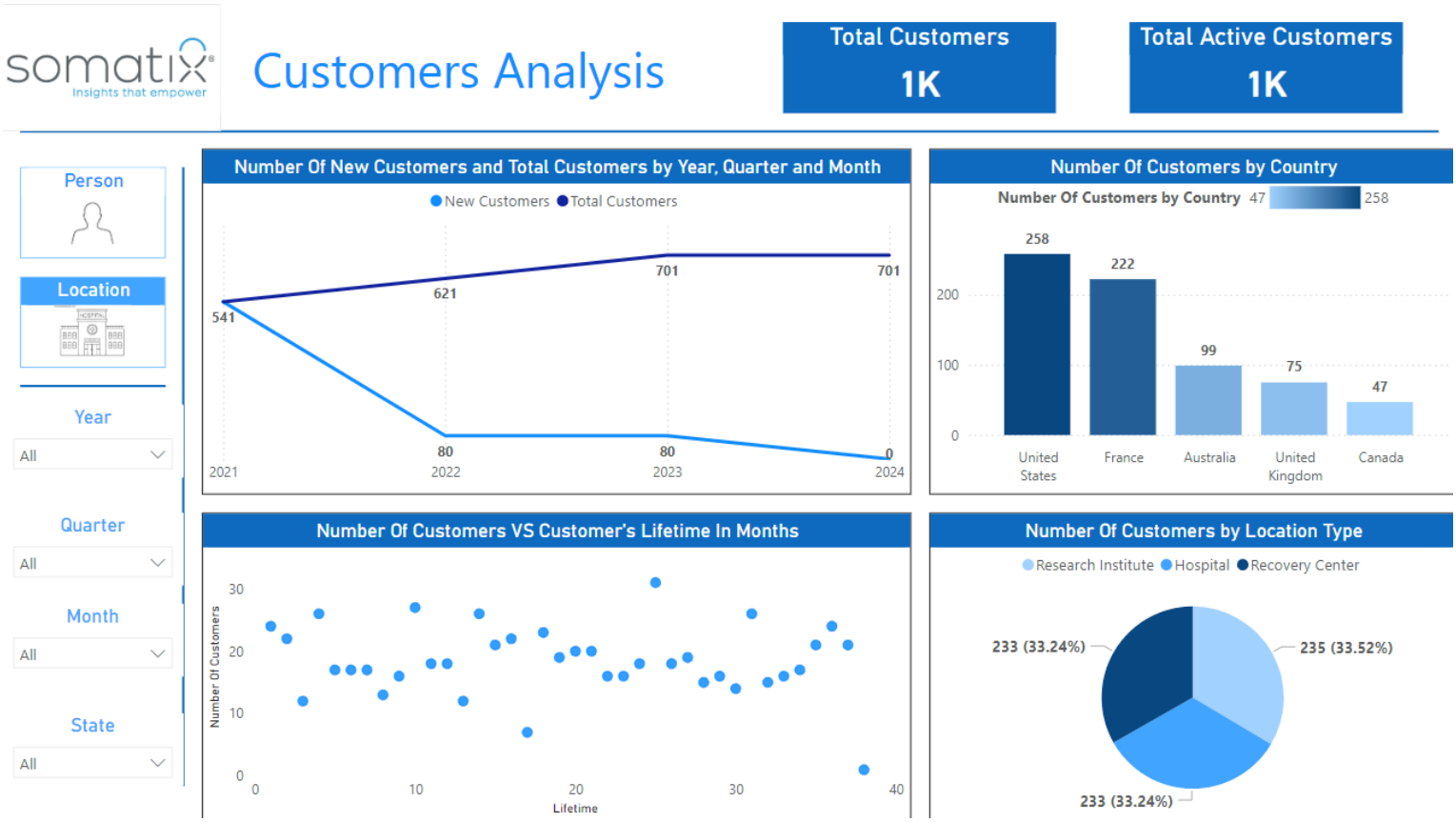
Person State

Key Metrics

- Total Customers
- Total Active Customers

Visuals

- Number of New Customers and Total Customers by Year, Quarter, and Month
- Number of Customers by Country
- Number of Customers vs. Customer's Lifetime in Months
- Number of Customers by Product Category



Location State

Key Metrics

- Total Customers
- Total Active Customers

Visuals

- Number of New Customers and Total Customers by Year, Quarter, and Month
- Number of Customers by Country
- Number of Customers vs. Customer's Lifetime in Months
- Number of Customers by Location Type