# BI System Specifications Document

Date: 11.07.2024
Written By: Omer Barmoha

## Content

# 1. General

## 1.1. Project Objectives

The project's goal is to create a full-scale BI solution from the PriorityERP database for TipRanks, an Israeli company specializing in the financial sector that uses artificial intelligence to analyze vast financial datasets and provides advanced stock market research tools for retail investors. The BI solution will leverage data from PriorityERP to develop summary data tables on sales data, user insights, various investment insights, private services and products, platform, dates, and more. The dashboards and reports included in the BI solution will enable management, department managers, and analysts to gain insights into customer preferences, behaviors, and loyalty, including optimizing engaged users and increasing market share through targeted strategies.

### 1.2.1 Project Contents

In this project, we will build a Data Mart that will contain information crucial for analyzing retail investor behavior and optimizing TipRanks' service offerings.

1. **Data Cleaning and Preparation:** Before analysis, comprehensive data cleaning and preparation will be conducted to ensure data quality and consistency.
2. **Main Summary Tables to be Built:**

- **Fact_Orders:** The transaction table tells us information about all the users' orders by User number, Agent ID, products, different platforms and shows us all the movements in the account. The data loading process for this table will be incremental to keep it updated.
- **Dim_Products:** Provides information about products categorized by types and subcategories.
- **Dim_Users:** Each record indicates a specific user in the system, with details such as username, Country, States and City and Classifies users based on their subscription level (Basic, Premium, or Diamond) and Discount from Service price.
- **Dim_Investment_Options:** Provides Information about the different platforms TipRanks works with such as Stock, ETF, Crypto, Currency, Commodity and Options and the number of options available in each.
- **Dim_Agents:** Table that shows me the information about the company's agents and the percentage of commissions they receive from each sale.

- **History Management Table**
- **Transfer_table:** Tracks all updates made to the tables within the BI solution.
- **Dim_Products_History:** Stores historical records of changes made to product information, utilizing Slowly Changing Dimensions (SCD) Type 4 to track product changes over time effectively.

**1.2.2** The reports will include data visualizations that support the project's goal in the following ways:

**Sales Analysis:**

The sales report will include data on sales (revenue, number of orders, types of services taken, and number of service units per option) by date, country, service, platform (Crypto, Stock, ETF..), and the number of securities. This will help the department assess the performance of all necessary sales components for growth. The reports will help identify sales trends such as seasonality and service category trends, order and revenue analysis, identifying top performers, and analyzing behavioral differences between the various platforms. All these will support data-driven strategic decision-making, which can lead to increased sales and revenue.

**Customer Analysis:**

The customer analysis report will include data on TipRanks customers by date, country, subscription type, product, and category. This report aims to help TipRanks better understand its customers' behavior, such as what (services), where (platform), and when they make purchases. This is crucial for maintaining customer flow and reaching new customers.

**Executive Dashboard:**

The executive dashboard will include key visual data from both reports. The dashboard will provide a broader perspective on the data and combine metrics from both sales and customer analysis.

## 2. Gnatt

Gantt chart

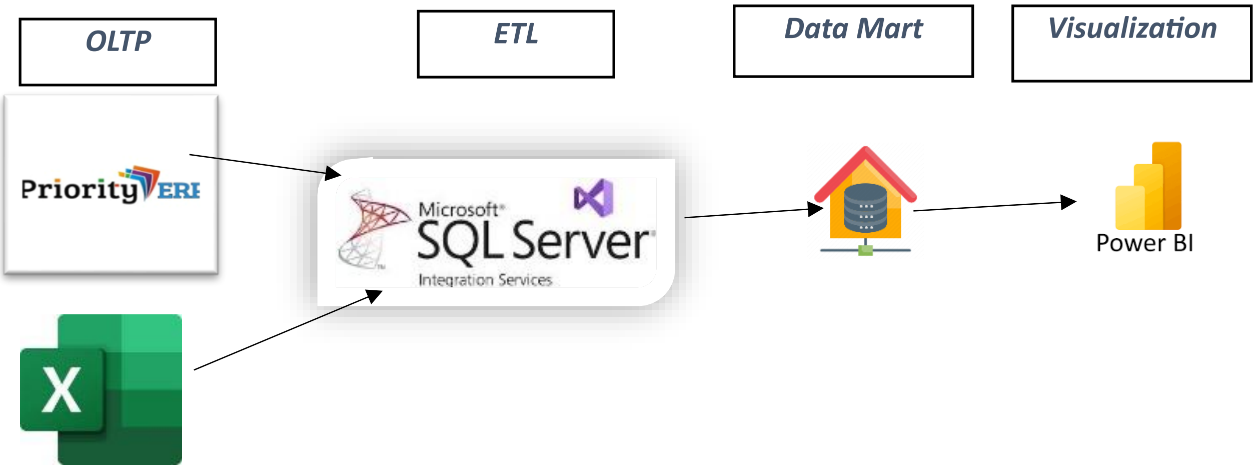
## 3. Technical Specification

### 3.1. Prerequisites

| | |
|---|---|
| **SQL Server** | ERP system in the operational DB (PriorityERP) - tables, data (SQL files) |
| **Excel** | Organizing data and adapting it to the requested reports |
| **SSIS** | ETL processes in Visual Studio |
| **Data refresh processes** | Definition of JOBS in SSMS |
| **Power BI** | Creating reports and dashboards |

### 3.2. Solution Architecture

3.2.1 HLD - High Level Design:



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

### 3.2.2 Power BI Reports:

#### 3.2.2.1 Sales Department Report:

- **Total Sales:** Total service sales for the period
- **YTD Sales:** Total sales from the beginning of the year to the current date
- **Total Orders:** Total number of orders
- **Average Order Price:** Total sales divided by the number of orders
- **Total Units:** Number of service units sold
- **Average Revenue per Customer:** Total sales divided by the number of customers
- **Number of Active Users:** How many users are active and paying users
- **Total Sales and Year-over-Year Growth:** Graph showing total sales with growth or decline rate compared to the previous year (can also show orders or units instead of sales)
- **Total Sales by Month and Day - Different Platforms:** Graph showing sales distribution by month and day, across different platforms and investment types (can also show orders or units instead of sales)
- **Top Platforms by Sales:** Graph showing the investment platform with the highest sales (can also show orders or units instead of sales)
- **Sales by Country:** Graph showing sales distribution by users' geographical location
- **Sales by Category and Subcategory:** Graph showing sales distribution by categories and subcategories of different services (can also show orders or units instead of sales)

#### 3.2.2.2 Customer Department Report:

- **Total Customers:** Total number of registered customers
- **Total Paying Customers:** Total number of customers with different service levels
- **New Customers:** Number of customers who made their first purchase during the period
- **Percentage of New Customers:** (Number of new customers divided by total number of customers) * 100
- **YTD Customers:** Number of new customers who joined from the beginning of the year to the current date
- **Average Revenue per Customer:** Total sales divided by the number of customers
- **Average Number of Orders per Customer:** Total number of orders divided by the number of customers
- **Total Customers and New Customers by Month and Day:** Graph showing the distribution of customers and new customers by month and day
- **Customers by Country:** Graph showing customer distribution by country

- **Customers by Number of Orders:** Graph showing customer distribution by the number of orders placed by each customer
- **Top Products by Customer:** Products purchased by the most customers
- **Average Revenue per Customer Compared to Previous Year by Month and Day:** Graph showing average revenue per customer in the current period compared to the same period in the previous year

**3.2.2.3 Executive Dashboard:**

- **Total Sales:** Total sales for the period
- **Average Monthly Revenue:** Average monthly sales
- **Total Orders:** Total number of orders
- **Total Customers:** Total number of customers
- **YTD Sales:** Total sales from the beginning of the year to the current date
- **MTD Sales**
- **Total Sales and Month-over-Month Growth by Quarter and Month**
- **Revenue from Different Platforms**
- **Top Selling Services by Revenue**
- **Total Users by Quarter and Month**
- **Revenue by Country**

# 4. Functional Specification

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

4.1.1. Source to Target
• Source To Target link

A total of 14 tables will be used from the operational database.

4.1.2. ERD model of the TipRanks database
• ERD link.

## 4.2. ETL processes

The ETL process is done in SSIS using 15 packages.

All packages include 2 recurring SQL execution tasks (Update StartTime and Insert Data to Data_Transfer), and row count changes that oversee the Data_Transfer update Table. When in every run a trigger is activated on the Data_Transfer table that calculates the run time from start to finish and returns us the result in seconds

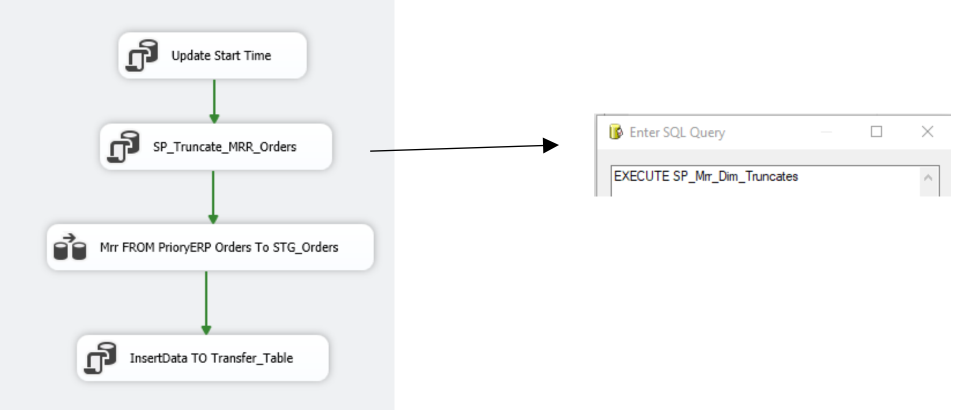
# ***The SSIS Package***

## 1. facts Orders:

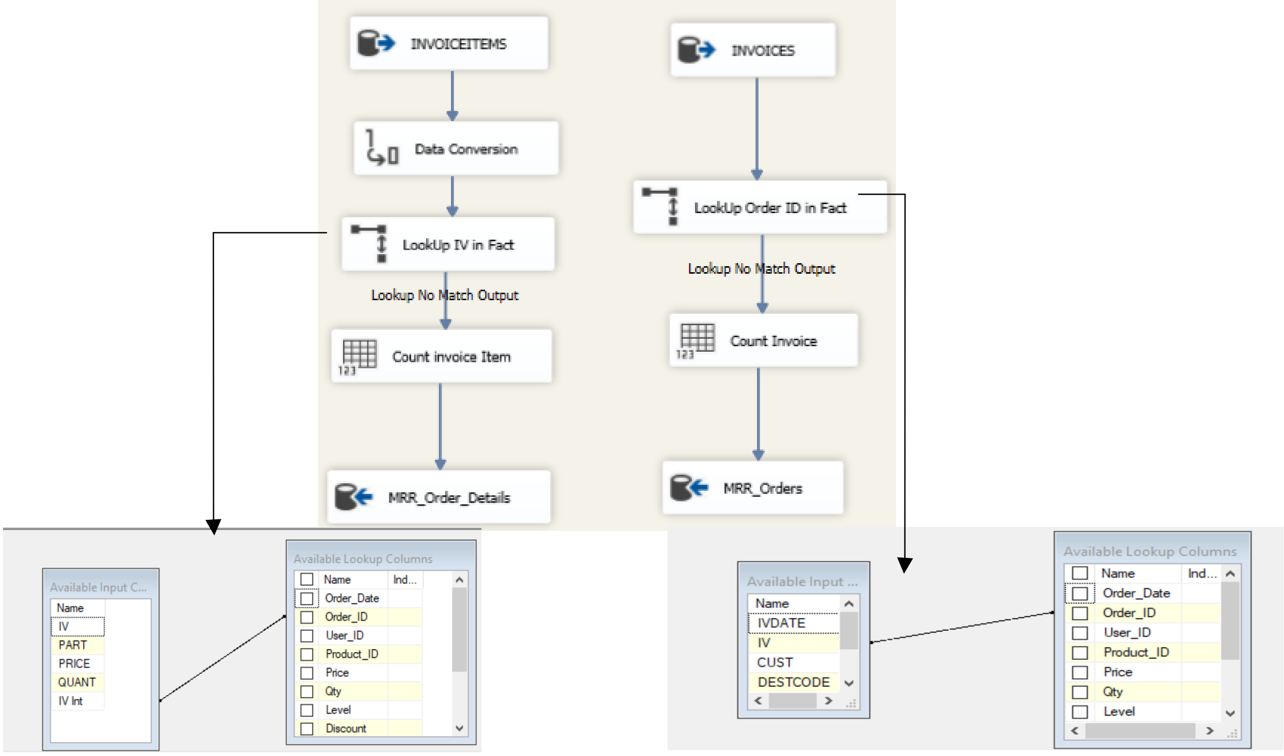
### 1.1 MRR_Orders package:

Abbreviated mirror tables (using a stored procedure) and data loaded from PriorityERP Database (INVOICES and INVOICEITMES) for TipRanksDM mirror tables.

In the data flow, data is incrementally loaded using lookup transformations, meaning only new transactions that cannot be found in the Fact_Orders table are loaded:

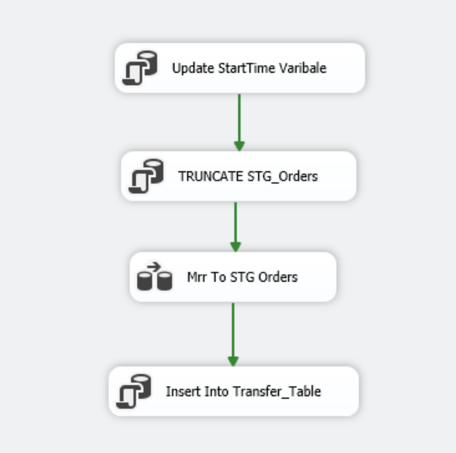

Update Start Time
SP_Truncate_MRR_Orders
Mrr FROM PrioryERP Orders To STG_Orders
InsertData TO Transfer_Table
Enter SQL Query

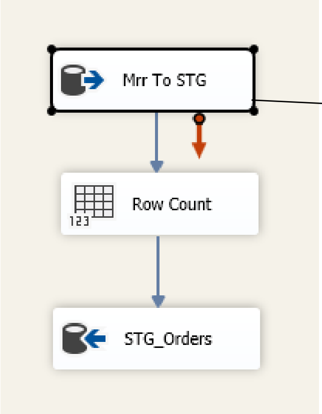
```
EXECUTE SP_Mrr_Dim_Truncates
```

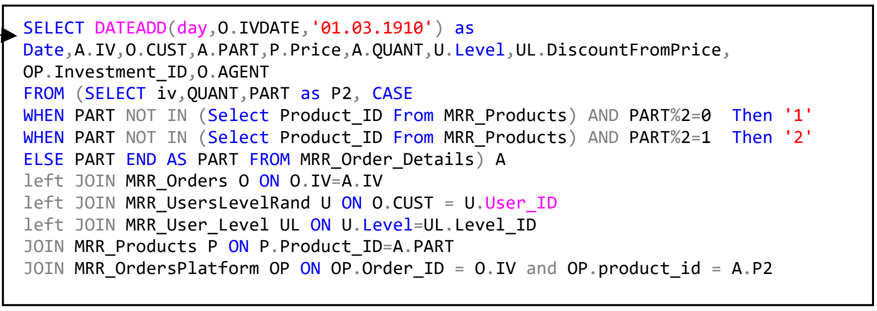


### 1.2 STG_Orders package:

STG_Orders table is truncated, and the mirror tables are joined and loaded using a data flow task.



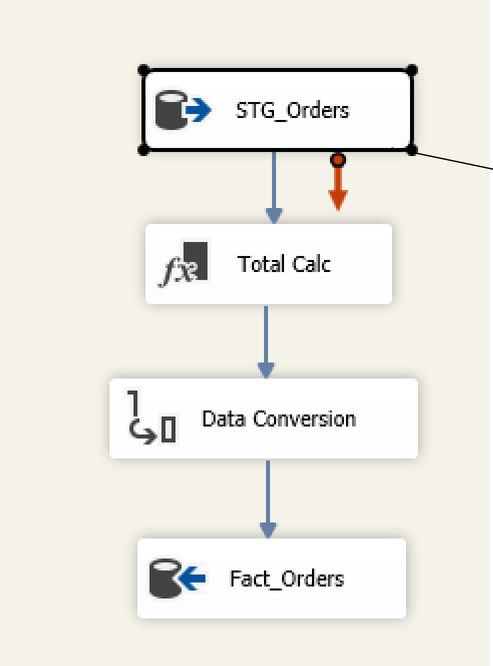


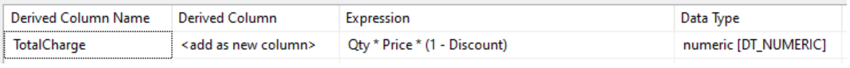


```
SELECT DATEADD(day,O.IVDATE,'01.03.1910') as
Date,A.IV,O.CUST,A.PART,P.Price,A.QUANT,U.Level,UL.DiscountFromPrice,
OP.Investment_ID,O.AGENT
FROM (SELECT iv,QUANT,PART as P2, CASE
WHEN PART NOT IN (Select Product_ID From MRR_Products) AND PART%2=0  Then '1'
WHEN PART NOT IN (Select Product_ID From MRR_Products) AND PART%2=1  Then '2'
ELSE PART END AS PART FROM MRR_Order_Details) A
left JOIN MRR_Orders O ON O.IV=A.IV
left JOIN MRR_UsersLevelRand U ON O.CUST = U.User_ID
left JOIN MRR_User_Level UL ON U.Level=UL.Level_ID
JOIN MRR_Products P ON P.Product_ID=A.PART
JOIN MRR_OrdersPlatform OP ON OP.Order_ID = O.IV and OP.product_id = A.P2
```

### 1.3 DM_Orders package Fact_Orders:

Data is loaded from STG_Orders to Fact_Orders, and a Total column is added.





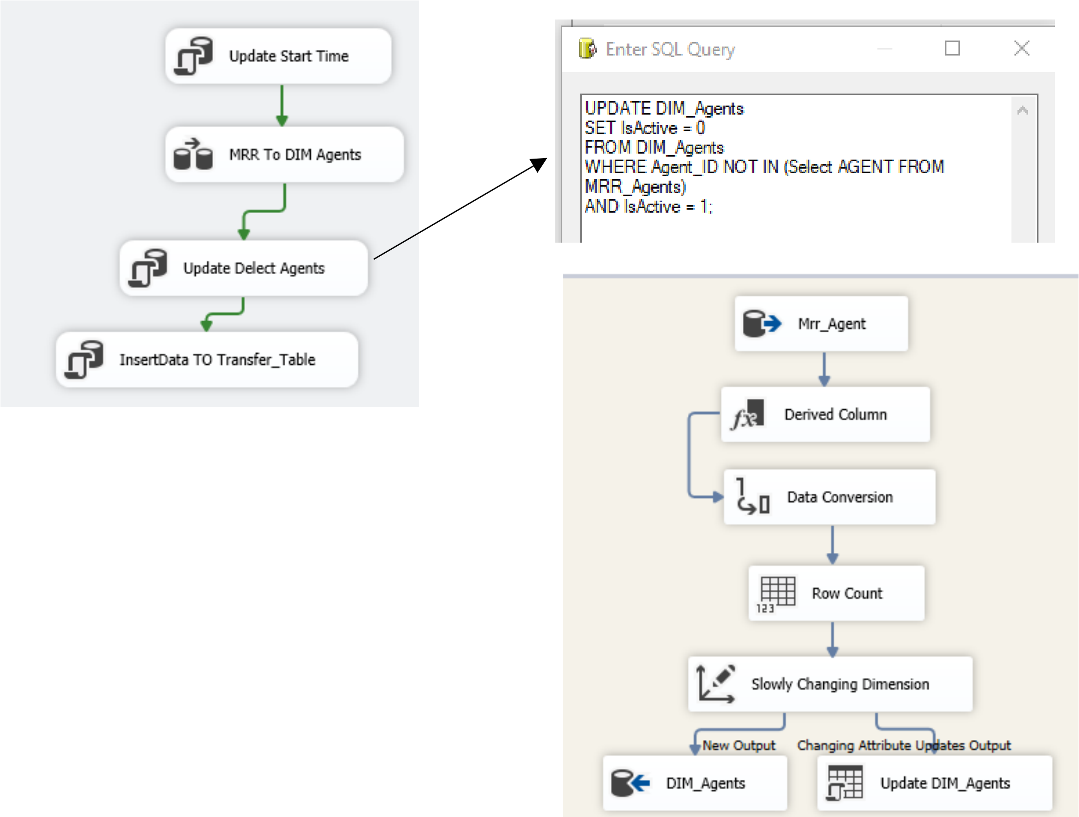
| Derived Column Name | Derived Column | Expression | Data Type |
|---|---|---|---|
| TotalCharge | <add as new column> | Qty * Price * (1 - Discount) | numeric [DT_NUMERIC] |

## 2. DM_Agents Table:

### 2.1 DIM_Agent package:

Because there is no need to make a connection with another table, the Agents appearance table is directed directly to the dimension table, by way of addition using Derived column IsActive column = 1 and in addition Data conversion – Convert the data types.

Therefore, the MRR table incrementally loads the dimension table using Slowly Changing Dimensions and looks for identical columns for updating or non-identical columns for DELETE or INSERT

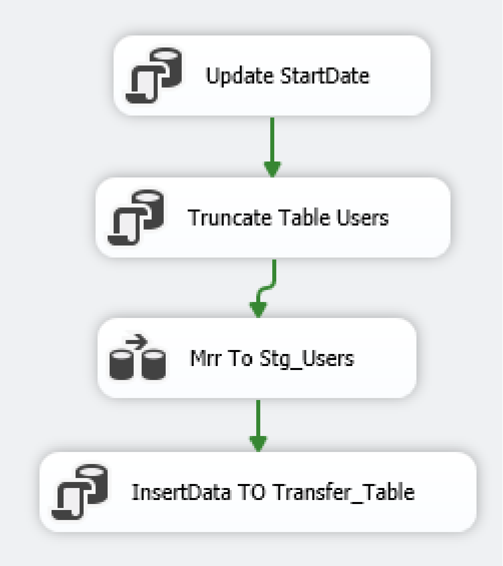


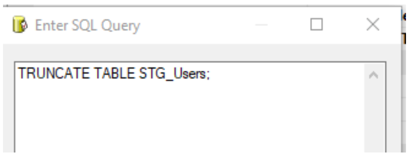
```
UPDATE DIM_Agents
SET IsActive = 0
FROM DIM_Agents
WHERE Agent_ID NOT IN (Select AGENT FROM
MRR_Agents)
AND IsActive = 1;
```

## 3. DM_Users Table:

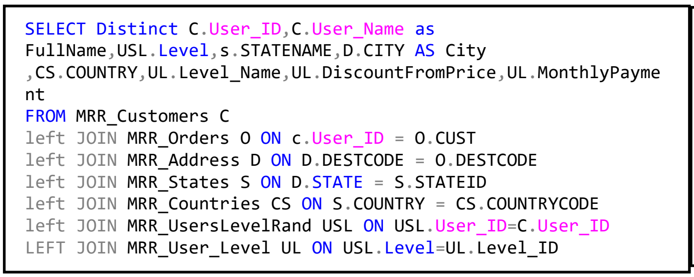
### 3.1 STG_Users package:

STG_Users table is truncated, and the mirror tables are joined and loaded using a data flow task.

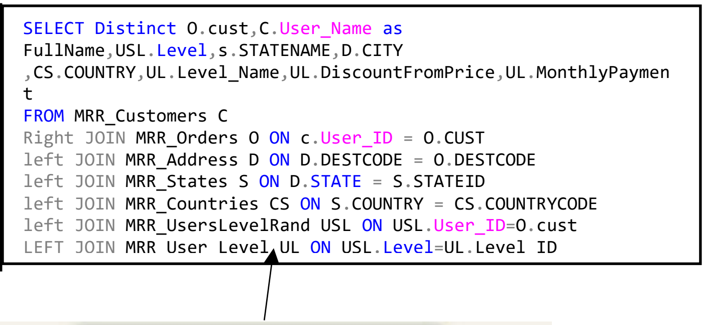




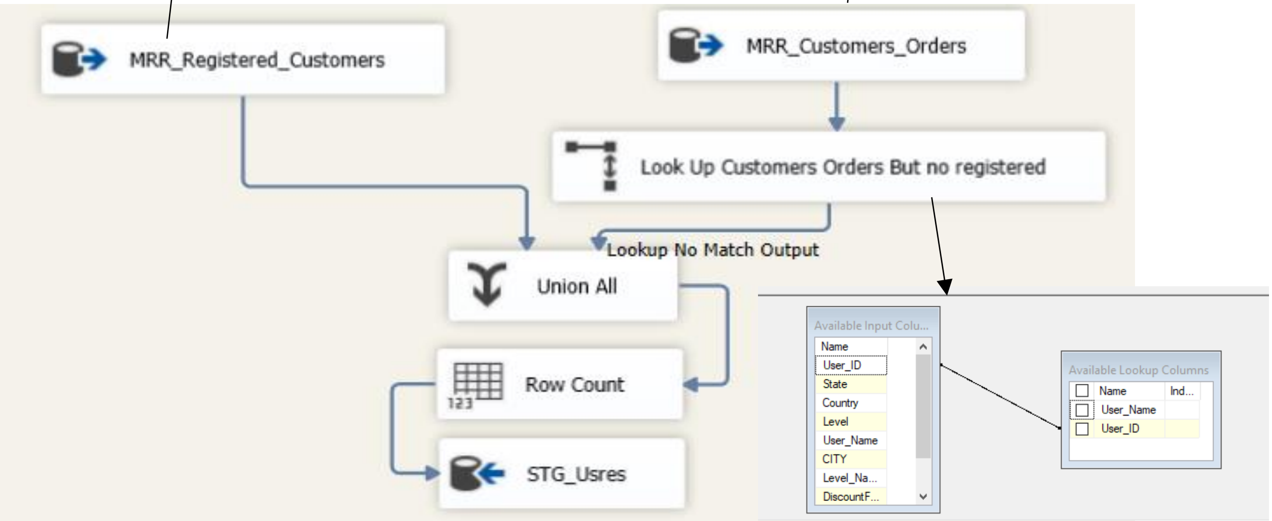
```
TRUNCATE TABLE STG_Users;
```



```
SELECT Distinct C.User_ID,C.User_Name as
FullName,USL.Level,s.STATENAME,D.CITY AS City
,CS.COUNTRY,UL.Level_Name,UL.DiscountFromPrice,UL.MonthlyPayme
nt
FROM MRR_Customers C
left JOIN MRR_Orders O ON c.User_ID = O.CUST
left JOIN MRR_Address D ON D.DESTCODE = O.DESTCODE
left JOIN MRR_States S ON D.STATE = S.STATEID
left JOIN MRR_Countries CS ON S.COUNTRY = CS.COUNTRYCODE
left JOIN MRR_UsersLevelRand USL ON USL.User_ID=C.User_ID
LEFT JOIN MRR_User_Level UL ON USL.Level=UL.Level_ID
```



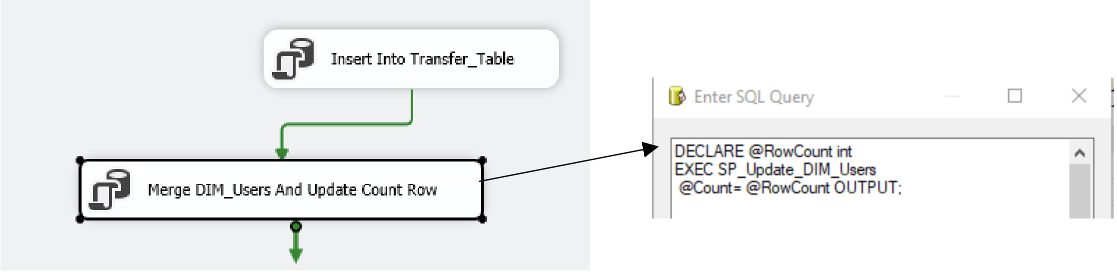
```
SELECT Distinct O.cust,C.User_Name as
FullName,USL.Level,s.STATENAME,D.CITY
,CS.COUNTRY,UL.Level_Name,UL.DiscountFromPrice,UL.MonthlyPaymen
t
FROM MRR_Customers C
Right JOIN MRR_Orders O ON c.User_ID = O.CUST
left JOIN MRR_Address D ON D.DESTCODE = O.DESTCODE
left JOIN MRR_States S ON D.STATE = S.STATEID
left JOIN MRR_Countries CS ON S.COUNTRY = CS.COUNTRYCODE
left JOIN MRR_UsersLevelRand USL ON USL.User_ID=O.cust
LEFT JOIN MRR_User_Level UL ON USL.Level=UL.Level_ID
```



In the data flow, the 7 mirror tables (MRR_Customers, MRR_Orders, MRR_Address, MRR_States, MRR_Countries, MRR_UsersLevelRand, MRR_User_Level) are joined, and the data is loaded to STG_Users table.

And all these mirror tables are performed together with a Union against the MRR_Orders table to make sure that indeed all the customers who make use are registered customers.

### 3.2 DM_Users package:


Insert Into Transfer_Table
Merge DIM_Users And Update Count Row
Enter SQL Query

```
DECLARE @RowCount int
EXEC SP_Update_DIM_Users
 @Count= @RowCount OUTPUT;
```

Data is incrementally loaded and updated in DIM_Users.

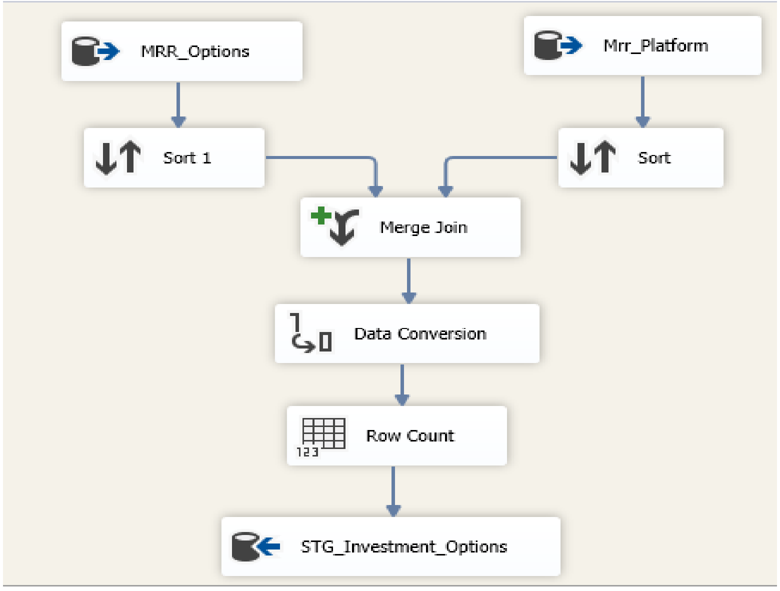
A merge stored procedure is executed in the Execute SQL Task, the merge statement works according to the following rational:
 1. New: User_ID **exists** in STG_Users (Source) but **not** in DIM_Users (Target)➔ Insert new.
2. Updated: User_ID **exists** in STG_Users **and** in DIM_Users **but one or more** of the other columns does not match ➔ Update the record in DIM_Users.
3. Deleted: User_ID **exist** in DIM_Users but **not** in STG_Users ➔ Updates the IsActive to = 0.

## 4. DM_Investment_Options

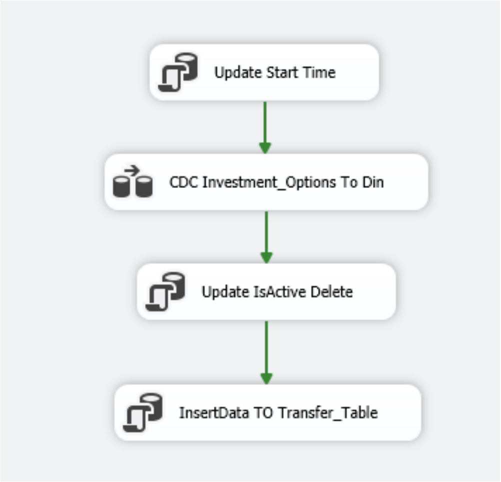
### 4.1 STG_Investment_Options package:

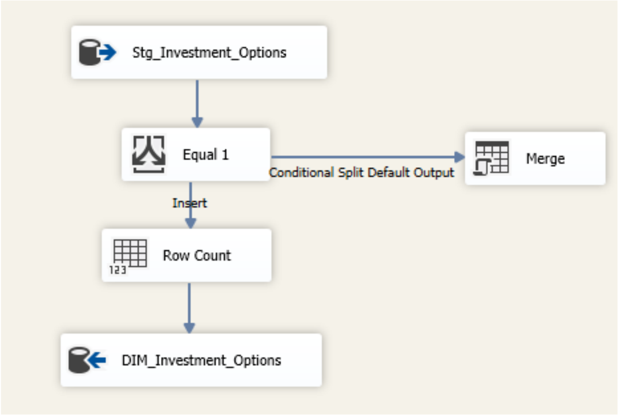
STG_Investment_Options table is truncated, and the mirror tables are joined and loaded using a data flow task

In the data flow, the 2 mirror tables (MRR_Options, MRR_Platforms) are Merge, and the data is loaded to STG_Investment_Options table.



### 4.2 DM_Investment_Options package:





Conditional Split – Check if the CDC Status Column is equal to 1 If equal insert the record to DIM_Investment_Options table.

If equal to 1, go to **Insert** **new** records to Dim_Investment_Options.

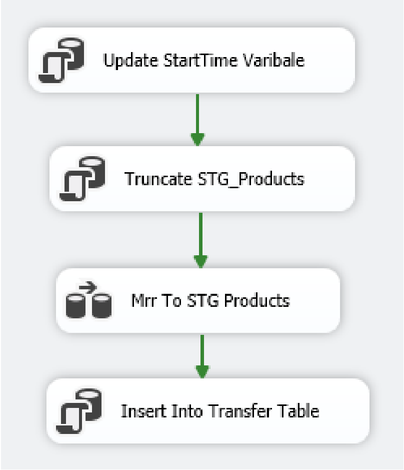
2. For Status 2, **Updated** Investment Options

3.when **Deleted**, Status 3 Investment Options  Update the IsActive column in Dim_Investment_Options = 0.

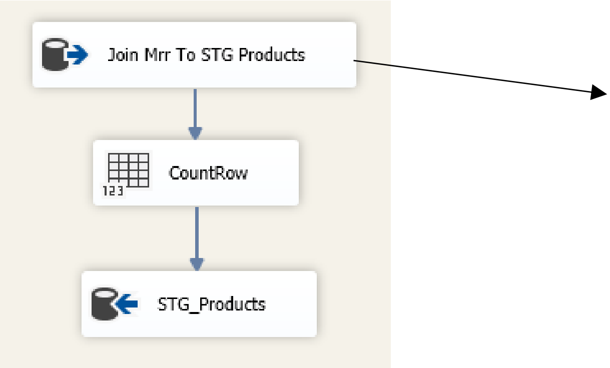
## 5. DIM_Products Table:

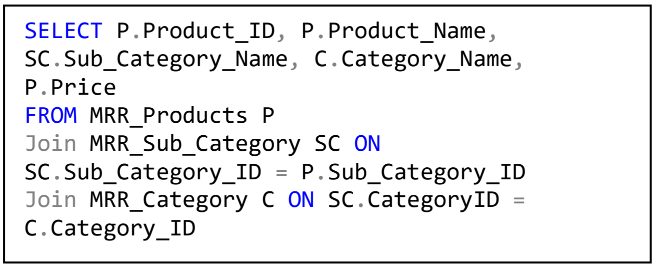
### 5.1 STG_Products package:

STG_Products table is truncated, and the mirror tables are joined and loaded using a data flow task. Product names, categories, and subcategories are updated using a stored procedure, executed in an Execute SQL task.



In the data flow, the 3 mirror tables (MRR_Products, MRR_Category, MRR_Sub_Category) are joined, and the data is loaded to STG_Products table.





```
SELECT P.Product_ID, P.Product_Name,
SC.Sub_Category_Name, C.Category_Name,
P.Price
FROM MRR_Products P
Join MRR_Sub_Category SC ON
SC.Sub_Category_ID = P.Sub_Category_ID
Join MRR_Category C ON SC.CategoryID =
C.Category_ID
```

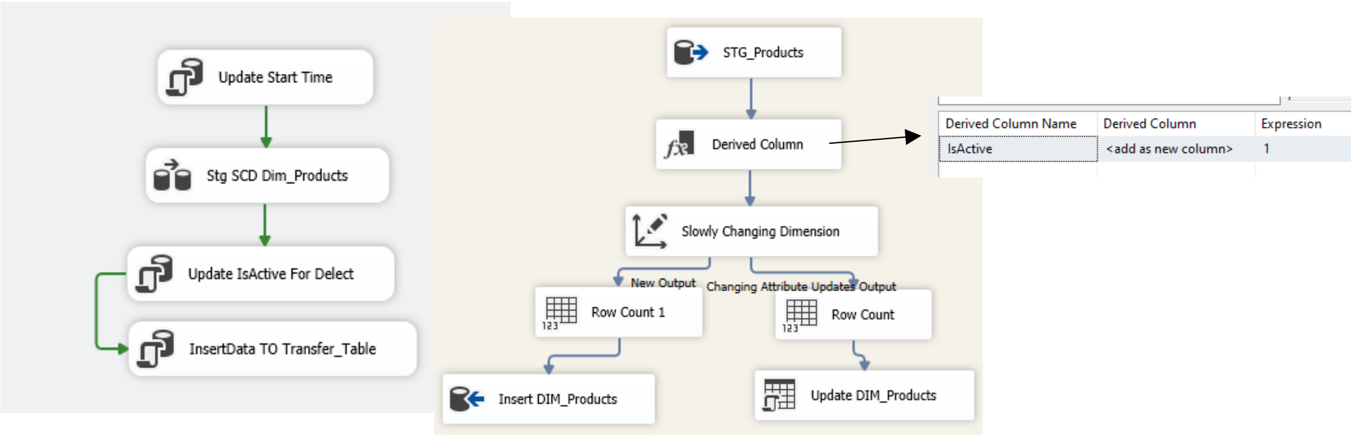
## 5.2 DM_Products package:

Control Flow:

Update IsActive – Contain stored procedure that update delete products isActive = 0 in the DIM_Products.

Update Start Time For the Transfer Table

Data Flow:



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

Derived column - Add isActive column with Default = 1

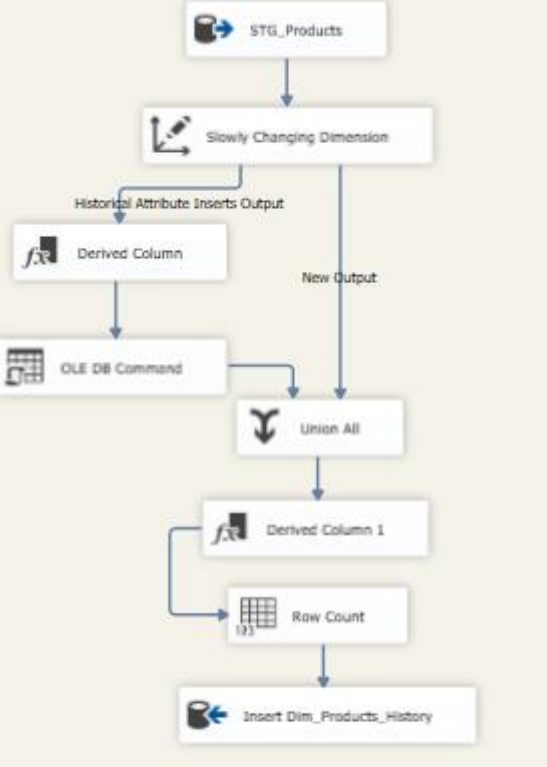
Data is incrementally loaded and updated in Dim_Products.
Deleted records are updated in Dim_Products using an Execute SQL task.

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

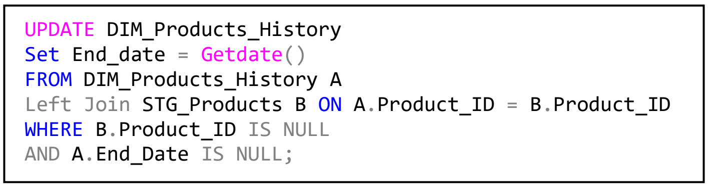
## 6. Dim_Products_History Table:

### 6.1 DM_Products_History package:

Data is incrementally loaded and updated in Dim_Products_History.
Deleted records are updated in Dim_Products_History using an Execute SQL task.


STG_Products
Slowly Changing Dimension
Historical Attribute Inserts Output
Derived Column
New Output
OLE DB Command
Union All
Derived Column 1
Row Count
Insert Dim_Products_History

Deleted records located using the following query:



```
UPDATE DIM_Products_History
Set End_date = Getdate()
FROM DIM_Products_History A
Left Join STG_Products B ON A.Product_ID = B.Product_ID
WHERE B.Product_ID IS NULL
AND A.End_Date IS NULL;
```

In the Data Flow, Incremental load to the Dim_Products_History table is done using the Slowly Changing Dimension transformation (change type: Historical Attribute).

Track changes in data, every row get StartDate with timestamp and if there is change, the SCD add new row to the Dim_Products_History and update the End_Date.

## 7. All Dim Tables – Mirror

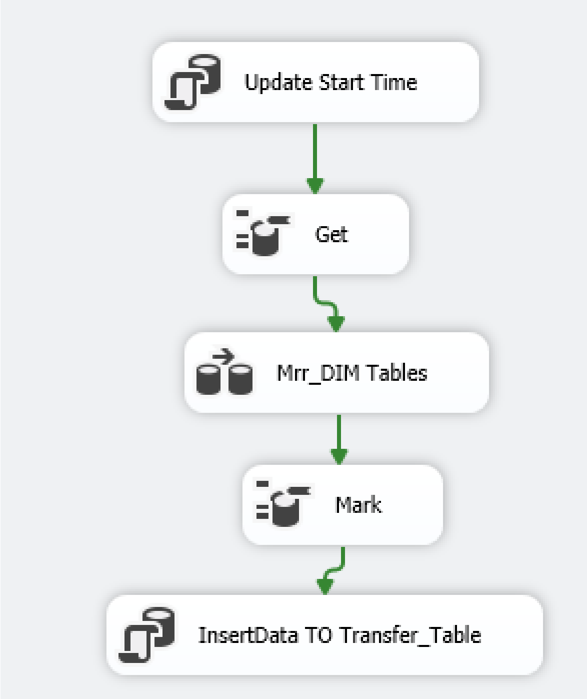
### 7.1 MRR_Dim package:

This package is responsible for loading data from PriorityERP tables to all mirror tables relevant for the dim tables (12 tables in total).

Also, in addition to Priority ERP, we also used EXCEL for the purpose of attracting system users to each customer and connecting the investment types to the orders (MRR_OrdersPlatform, MRR_UsersLevelRand)

All mirror tables (except MRR_Orders And MRR_Order_Details) are truncated using a stored procedure.

In the Control Flow:



Update Start Time ➔ insert new row into transfer table with the name of the package and the current time and date.

Insert Data to Transfer_Table ➔ Updating the inserted row in transfer table with the number of rows changed in the data flow and the end time.

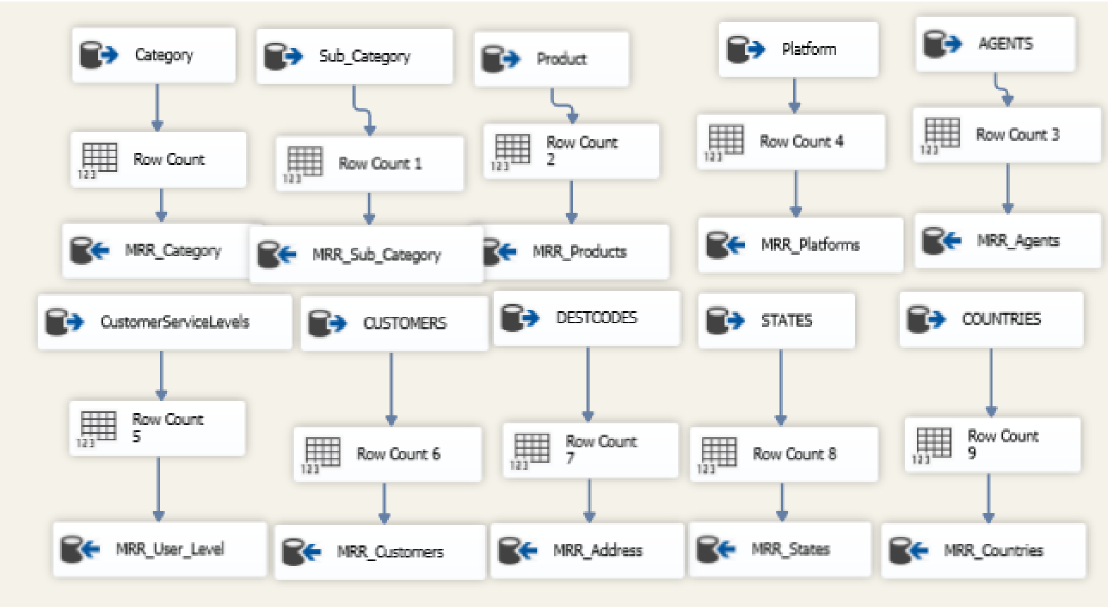
GET ➔ Read from table cdc_states the timestamp of the last time data was update in the Stores table.

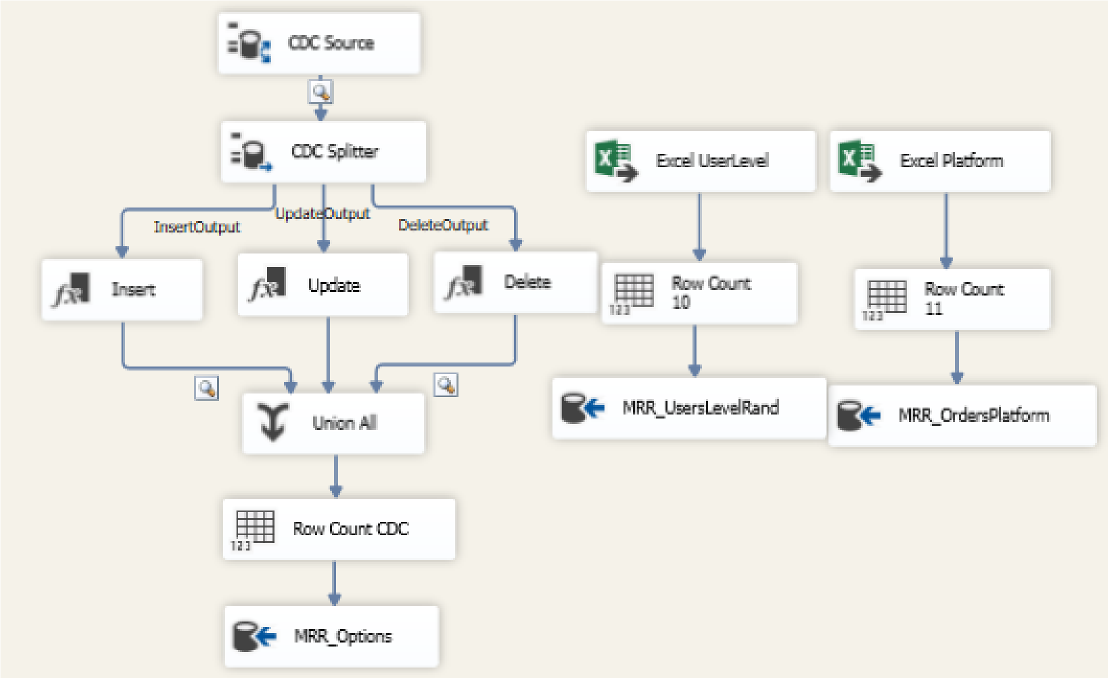
Mark ➔ Update the timestamp for the next CDC.

Data Flow Task ➔ Transfer OLTP table to MRR tables.

In The Data Flow:

Row Count - Count the rows and calculate for insert in transfer table, the number of the row then copied from OLTP tables to MRR tables.





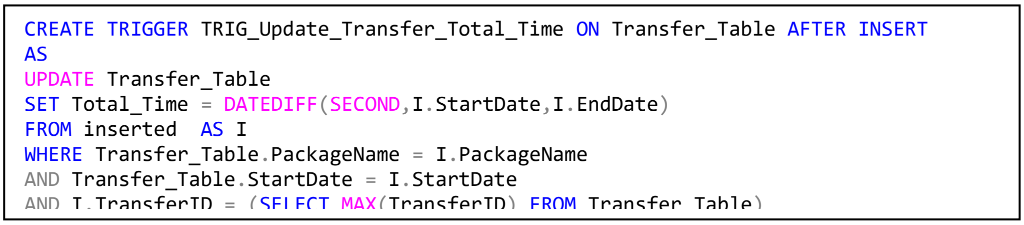
The data imported from Excel is the data on the Users numbers and the level of service they are a User when the data refers to all Users, registered and unregistered in the company's systems.

The options_table shows us the connection between the orders placed and the type of option selected and the platform it is on.

Because of the CDC Process on Options Table in the PriorityERP database, CDC Source read the added columns and the CDC splitter split it to derived columns
insert with Status = 1, update with Status = 2, delete with Status = 3.

## 8. Transfer_Table Table

**To monitor the ETL process**, a Transfer_Table table was created documenting each data insert, which table was updated and in which package, how many rows were inserted, start, end times and the time range that passed from the beginning of the time to the end of the time of the entire run (this operation works with the help of a trigger that works on every insert of data into the table).



```
CREATE TRIGGER TRIG_Update_Transfer_Total_Time ON Transfer_Table AFTER INSERT
AS
UPDATE Transfer_Table
SET Total_Time = DATEDIFF(SECOND,I.StartDate,I.EndDate)
FROM inserted  AS I
WHERE Transfer_Table.PackageName = I.PackageName
AND Transfer_Table.StartDate = I.StartDate
AND I.TransferID = (SELECT MAX(TransferID) FROM Transfer_Table)
```

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, and an insert statement is executed in the last task, inserting the values of the user variables: PackageName, TableName, StartTime, Count (which is updated in the data flow), with GETDATE () as EndTime.

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

## 9. Project_TipRanks ➔ Transfer to a production environment

## 10. Deploy To Production

TipRanksDM Deploy all packages to production

## 4.3. Visualization in Power BI:

4.3.1. The reports were created using Power BI Desktop and published to the Power BI Service.

The model in Power BI includes the fact table and the 4-dimension tables (no

including the product history table). A Dim_Date table was added to these tables, together

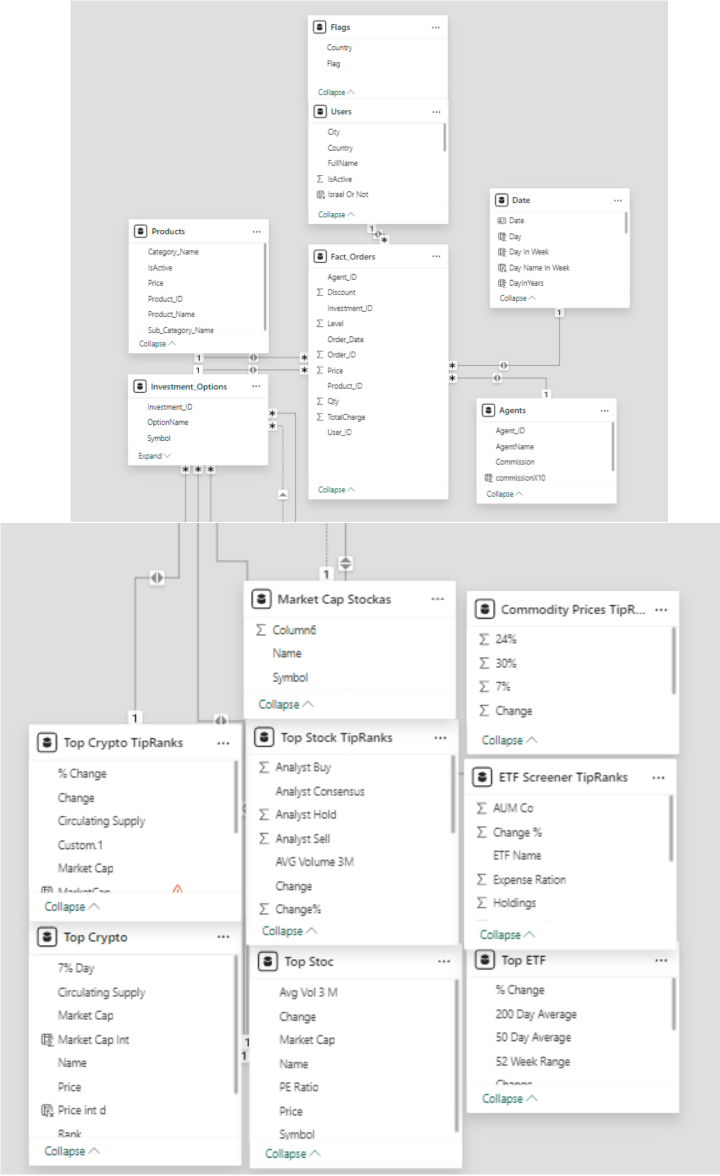
with 2 more tables:

• Table of flags – a table containing the countries and their flags.

was related to Dim_Users using the country name.

• Capital market data tables that include the data of the TipRanks website - a table detailing the prices of options, Market Cap

Buy and sell recommendations and a variety of other data used for maximum accessibility.



### 4.3.2. reports:

The project includes 3 reports: TipRanks OverView, Users Details, Sales Details.

#### 4.3.3.1. TipRanks OverView:

This report was created to provide a broader view of the company's position, it

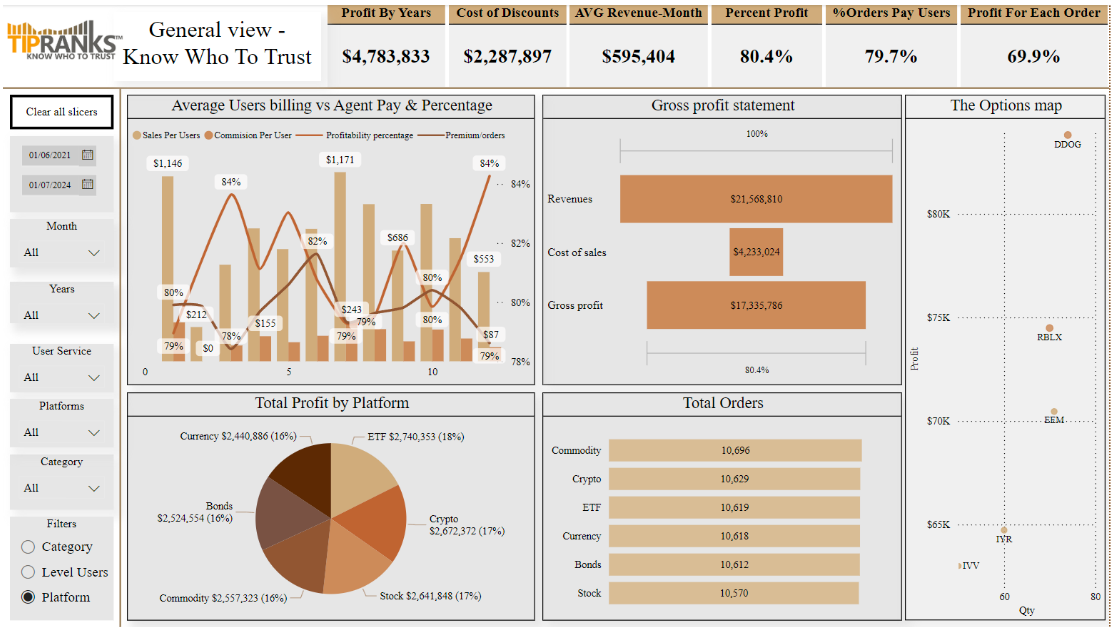
Includes key KPIs, sales performance and the best-selling products by quantity (X-axis) and their profit (Y-axis).
The OverView graphs show:

1. Average revenue per customer and average payments to agents and the percentages of profitability from each order, as well as the percentage of premium subscribers' orders from all the company's subscribers.

2. Presentation of the company's profit which includes income, expenses and net profit and the percentage remaining before additional payments

3. Total revenue by: (there is a slicer that can be selected according to what you want to check category/platform/Users Level)

4. Total orders according to the same slicer.



General view - Know Who To Trust

| Profit By Years | Cost of Discounts | AVG Revenue-Month | Percent Profit | %Orders Pay Users | Profit For Each Order |
|---|---|---|---|---|---|
| $4,783,833 | $2,287,897 | $595,404 | 80.4% | 79.7% | 69.9% |

Clear all slicers

01/06/2021
01/07/2024

Month: All

Years: All

User Service: All

Platforms: All

Category: All

Filters
○ Category
○ Level Users
● Platform

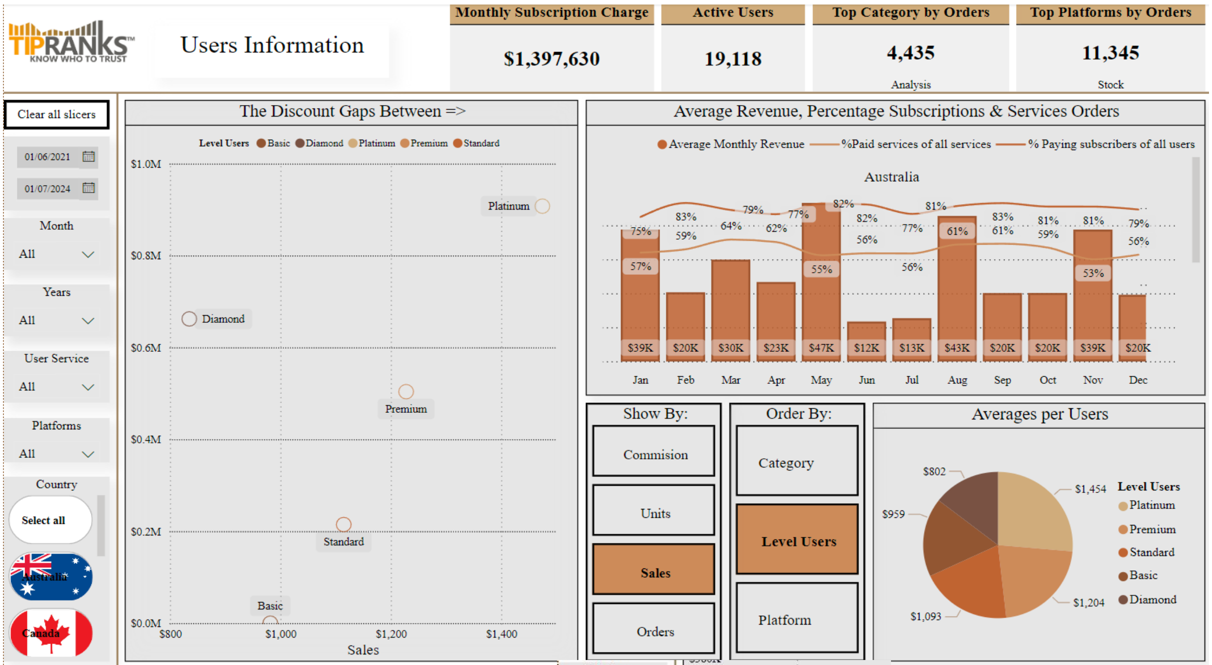
**KPI:** Revenues for the year, total cost of all discounts, average monthly revenue, percentage of profitability compared to the previous year (YOY), the percentage of orders of paying subscribers from the total of orders in relation to last year, percentage of profitability from each order.

### 4.3.3.2. Users Details:

This report is created for managers and service departments to better understand

Customer behavior by segments/countries/different service levels on platforms and the sales characteristics of TipRanks services and products.
At the top of the graphs you can see average revenue per customer, profit per customer, the percentage of orders from monthly paying subscribers to the total number of subscribers and the percentage of paying subscribers from the total number of subscribers segmented by date (months, years).

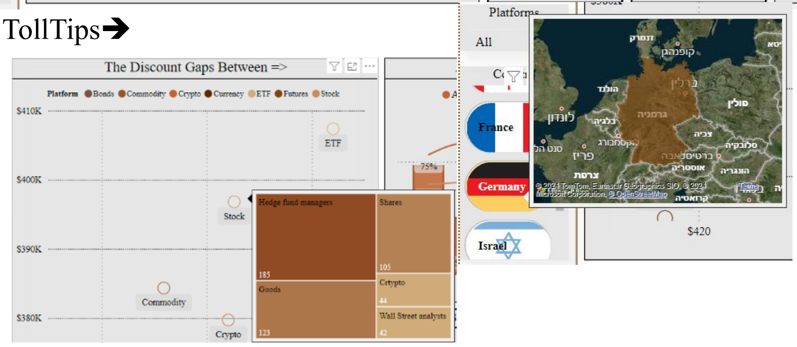
At the bottom of the report, you can see 2 graphs showing on the right side the average for each Users and on the left side the gap between the original price and the discounts given. We can switch between what we want to see (quantity/orders/commission/sales) and also according to which cut (category/level Users/platform) Using the buttons in the center, the user can control the data displayed in the graphs and change them.
On the left side of the report, you can filter by date, year, month, platform, service level, and country (when you stand on the country you see the world map and where it is located).



Users Information

| Monthly Subscription Charge | Active Users | Top Category by Orders | Top Platforms by Orders |
|---|---|---|---|
| $1,397,630 | 19,118 | 4,435<br>Analysis | 11,345<br>Stock |

TollTips➔



**KPI:** Total revenues for the company, subscription costs, total active customers in the company, the product category with the most orders, and the platform with the most orders

4.3.3.3. Sales Details:

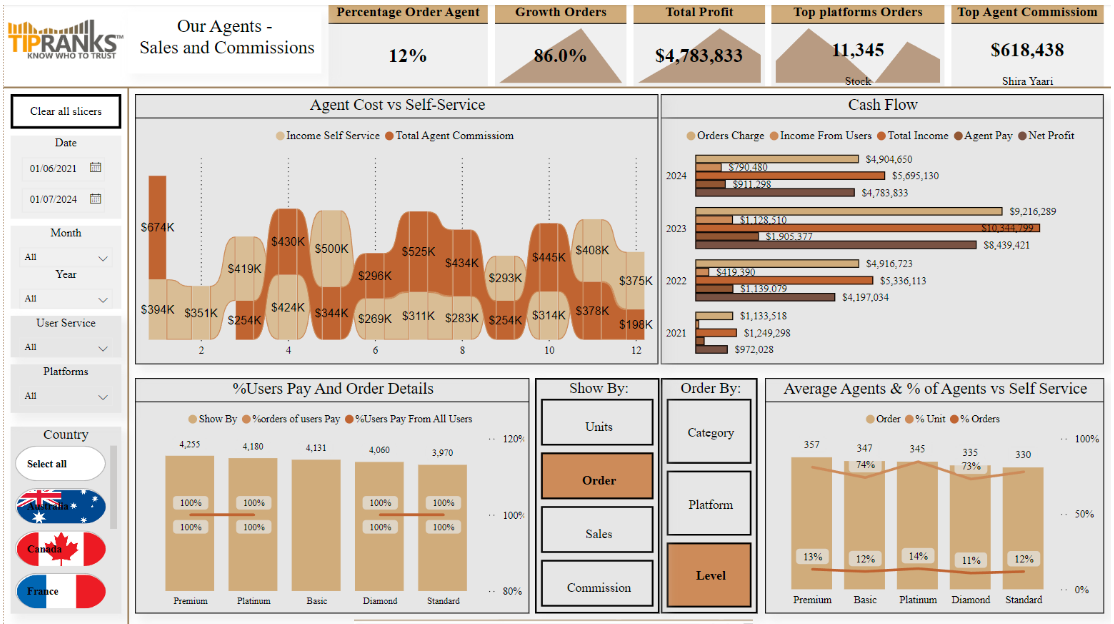
This report is created so that the sales department and managers can track the trends and understand

the company's performance

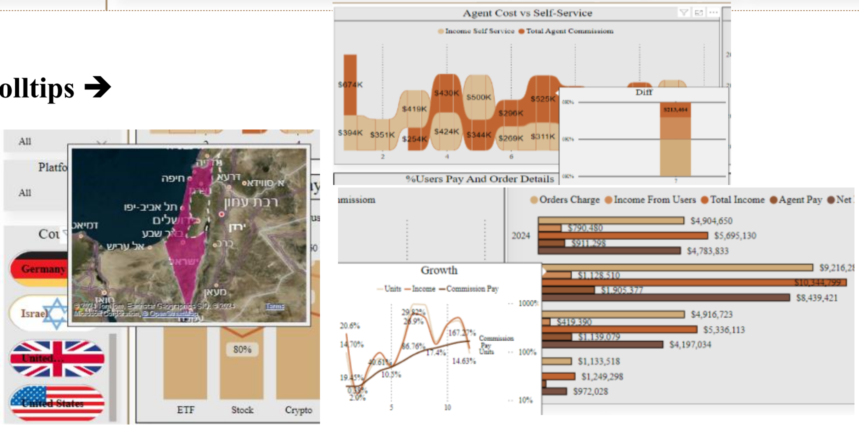
and the performance of sales agents in front of self-service.

In its initial state, the graphs show revenue data and Agent cost data against self-service revenue (when you can stand on the column and see the gap between them) and the flow of money that includes sales revenue, revenue from monthly payments and expenses to agents. Also, at the bottom of the report you can see 2 graphs showing on the right side the average for each agent and on the left side the total for the period, you can switch between what you want to see (units/orders/commission/sales) and also according to which cut (category/service level/platform) Using the buttons in the center, the user can control the data displayed in the graphs and change them.

On the left side of the report, you can filter by date, year, month, platform, service level, and country (when you stand on the country you see the world map and where it is located).



**Tolltips ➔**



**KPI:** Percentage of orders placed by agents, percentage of annual sales growth, total profits for the year, the platform with the most orders and the agent with the highest commissions.