

BI Developer COURSE   
Final project

Technion 06-11/2020

**By: Matan Aviram, Limor Goodker, Rotem Raiter**

Table of Contents

[1 ABSTRACT 4](#_Toc55323453)

[1.1 Introduction 4](#_Toc55323454)

[1.2 Methodology 5](#_Toc55323455)

[1.3 Tools 6](#_Toc55323456)

[2 ETL 7](#_Toc55323457)

[2.1 “The Voice” operational data base 7](#_Toc55323458)

[“The Voice” data sources: 7](#_Toc55323459)

[2.2 Creating data warehouse 8](#_Toc55323460)

[The Voice MRR 9](#_Toc55323461)

[The Voice STG 12](#_Toc55323462)

[The Voice Dim tables 13](#_Toc55323463)

[The Voice DW 14](#_Toc55323464)

[3 SSRS 18](#_Toc55323465)

[3.1 General 18](#_Toc55323466)

[3.2 Reports 18](#_Toc55323467)

[Report 1: Customers Segments 19](#_Toc55323468)

[Report 2: Total calls and type by weekday 19](#_Toc55323469)

[Report 3: Revenue for Daypart and call type origin 20](#_Toc55323470)

[3.3 Report server security 21](#_Toc55323471)

[4 TABULAR MODEL 23](#_Toc55323472)

[4.1 Why and how 23](#_Toc55323473)

[4.2 Creating measures 23](#_Toc55323474)

[4.3 Setting hierarchies 24](#_Toc55323475)

[5 Power BI 25](#_Toc55323476)

[5.1 Dashboards 26](#_Toc55323477)

[5.2 Reports: 28](#_Toc55323478)

[6 Scripts 32](#_Toc55323479)

[6.1 SQL 32](#_Toc55323480)

[6.2 DAX 34](#_Toc55323481)

[7 Challenges and how we dealt with them 35](#_Toc55323482)

[7.1 Understanding the operational data base 35](#_Toc55323483)

[7.2 SSIS debugging and error handling 35](#_Toc55323484)

# ABSTRACT

## Introduction

This Project demonstrates a large cellular phone company named “*The Voice*”, which runs a worldwide operational system. Management would like to have a BI tool that will support them when making important decisions that will affect the future of the company. Such as:

* The optimal pricing system for each service
* Redirect resources from once service to another
* Tacking actions with customers that are likely to abandon the company
* Optimal resource sharing in order to cut expenses
* Track revenue over date and time

The Operational DB of The voice contained the following Tables: Customers, Countries, Countrypre, Package catalog, Customer invoice and Customer lines. In Addition, we had two CSV files contains call types and operators. Means, the company have more than one data source.

To do so, we need to make sure we have **one** relatable database source. It has to be easily queried, can support a large scale of changing data, free of NULLs and corrupt data (for example missing customer details), otherwise management decisions making maybe missed held.

Note: Along the file, we added screenshots as example for our process, if you would like an in depth details, scripts or even to view the original files please contact us.

## Methodology

The solution is based on the **star scheme** data warehouse by using the ETL process (Extract, transform, load), in which the data warehouse replicates the operational DB. The process for creating the DW contains three stages. Each of them owns none operational data base – Mrr, Stg and DW. The result is a DW Null and blanks free with no duplicate values.

Next step is to process all the changes from the DW to the tabular model using SSAS. The SSAS support multidimensional cube that is fast and efficient and can deliver self-service business intelligence (BI) to a variety of applications, including Excel, PowerPivot, Power View and SQL Server Reporting Services (SSRS).

Last step – Power BI. Here we can finally visualize our dynamic database on the measures and the DW data.

Illustrations:

Assumptions:

1. Each customer phone number that is 7 digits long
2. Each phone number has prefix in the length of 1 to 3 digits for its Country code
3. The next 1 to 3 digits related to specific operator

## Tools

* SQL server - Using SSMS in data base engine and analysis services for SQL and DAX Scripting, querying and jobs
* Visual Studio SSDT - Used for SSIS, SSRS and SSAS implementation
* SSIS – ETL process (Mrr - Stg - Dim and Fact)
* SSRS – create and deploy reports
* SSAS – Creating measures, rename table and columns for end users, build tables hierarchy and deploying to tabular server
* PBI Report server – All reports SSRS and PBI published there. Also useful for creating users, folders and adjust end user permissions.
* Power BI desktop – Data visualization, KPI’s and Dashboards for decision making and data analysts
* Power Bi Service - Publish Reports to Power BI Service

# ETL

In this stage, we have created 3 Data Bases as part of the ETL process. First, let us look at the operational data sources to understand how and why our ETL process was planned the way it is.

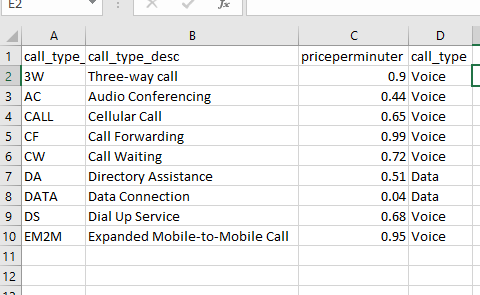
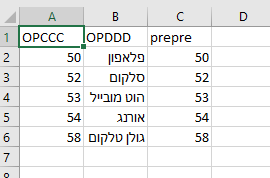
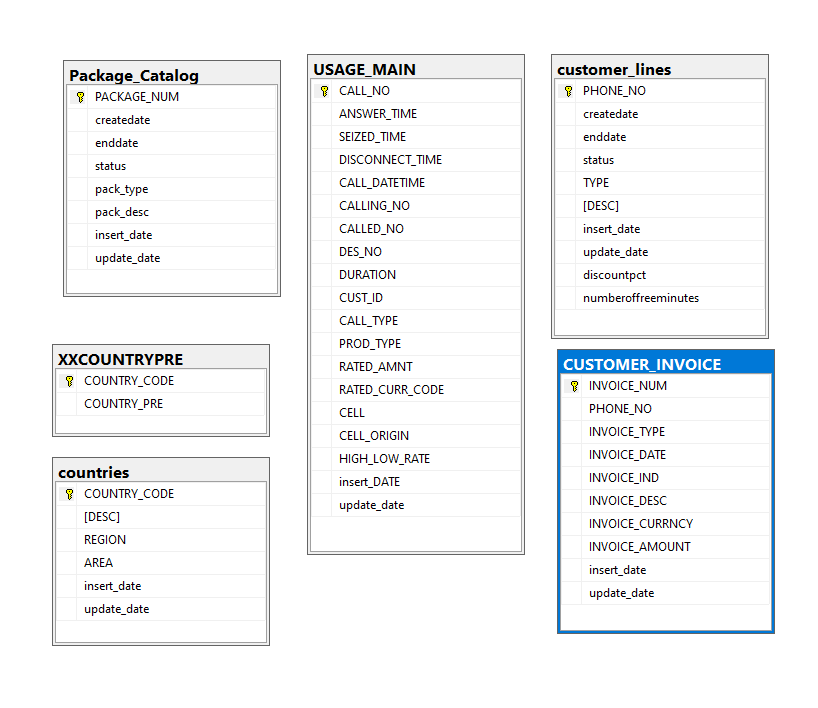
## “The Voice” operational data base

The operational data construct from three data sources. The first source is a rational DB, which contains the following tables: Customers, Countries, Countrypre, Package catalog, Customer invoice and Customer lines. The second and third data sources are CSV files for the call types and operators tables.

Example for data flaws:

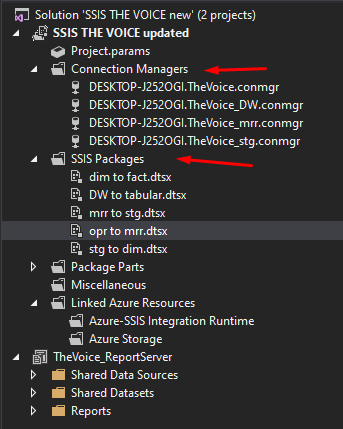
* Blank operators codes
* Null values in country name
* Duplicate country prefix code
* Null values in Customer name and address

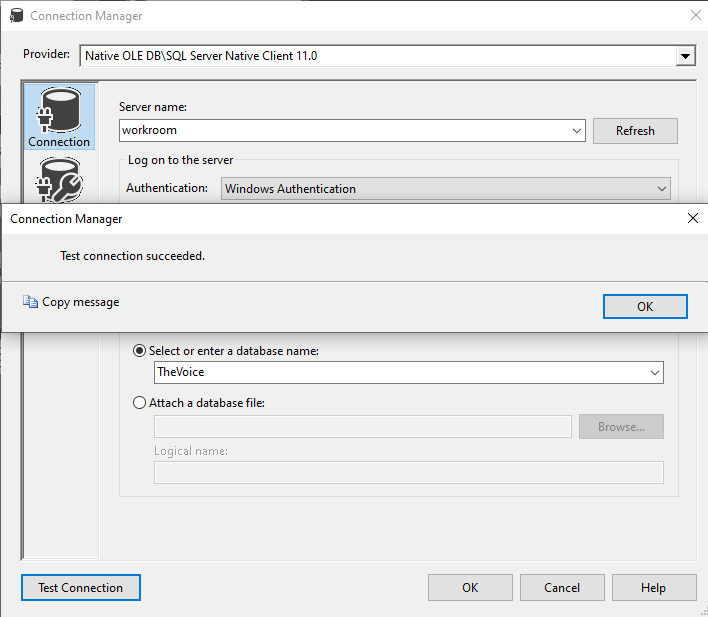
### “The Voice” data sources:

1. CSV files:
2. ERD:

## Creating data warehouse

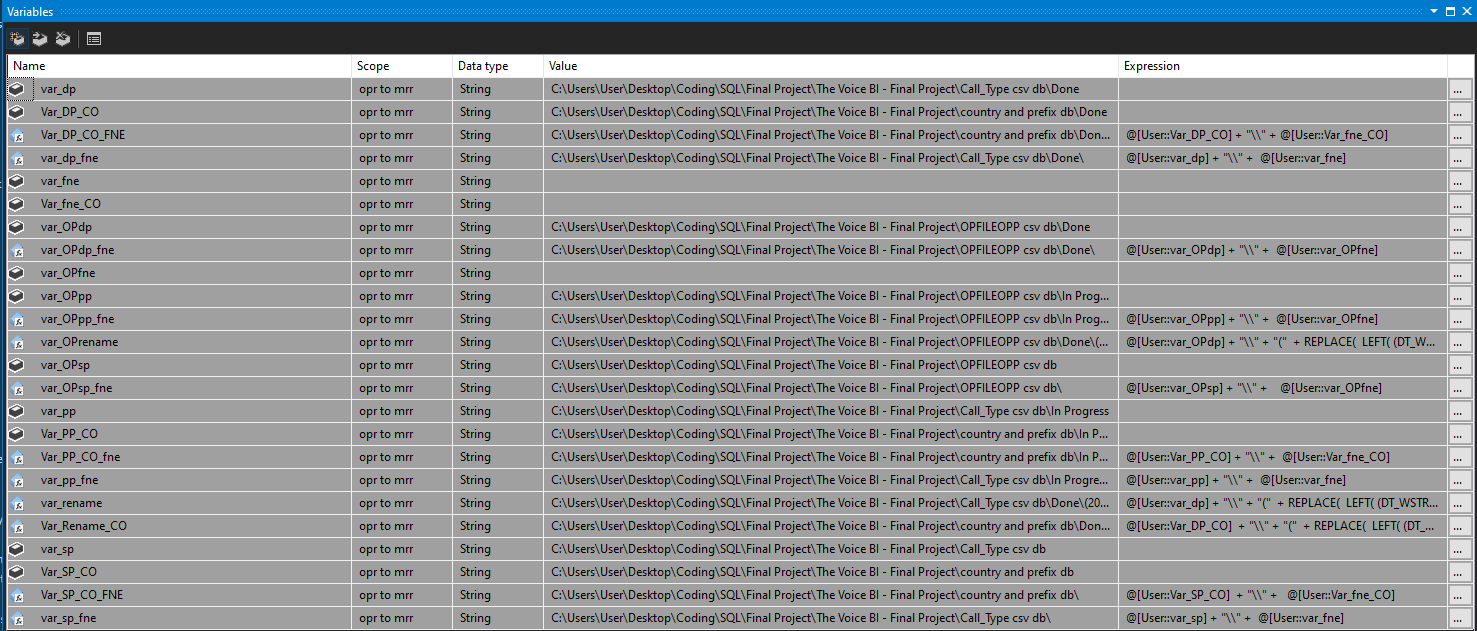
This part is the core of our project. The chart below demonstrate an overview of the ETL process, each square represent an ETL step for creating DW. For each database, we created a connection and a package in the SSIS, each package contain the relevant data flows for each of it tables.

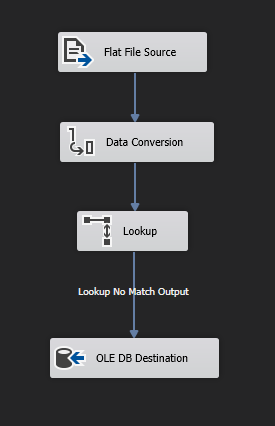
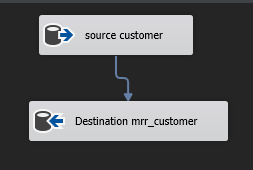


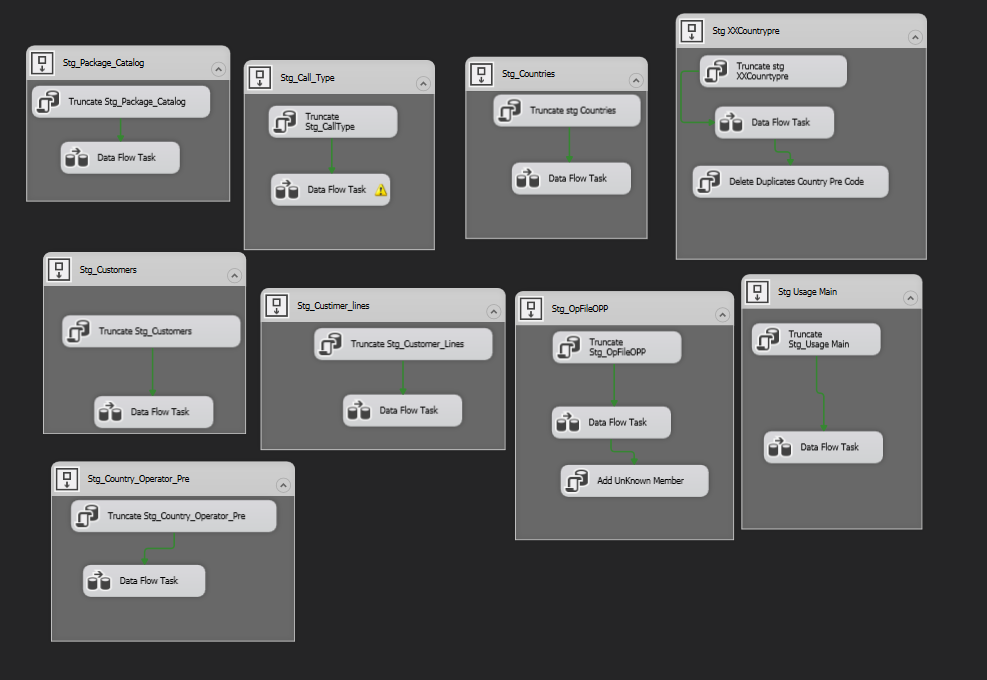
The Voice MRR – Using the SSIS, we mirrored the Data from The Voice operational data Base and the CSV files into the MRR. In addition, we created 3 new tables that contained the operators and deferent call types loaded dynamically from the CSV file provided by the customer. Note this is the first time we have single data source.

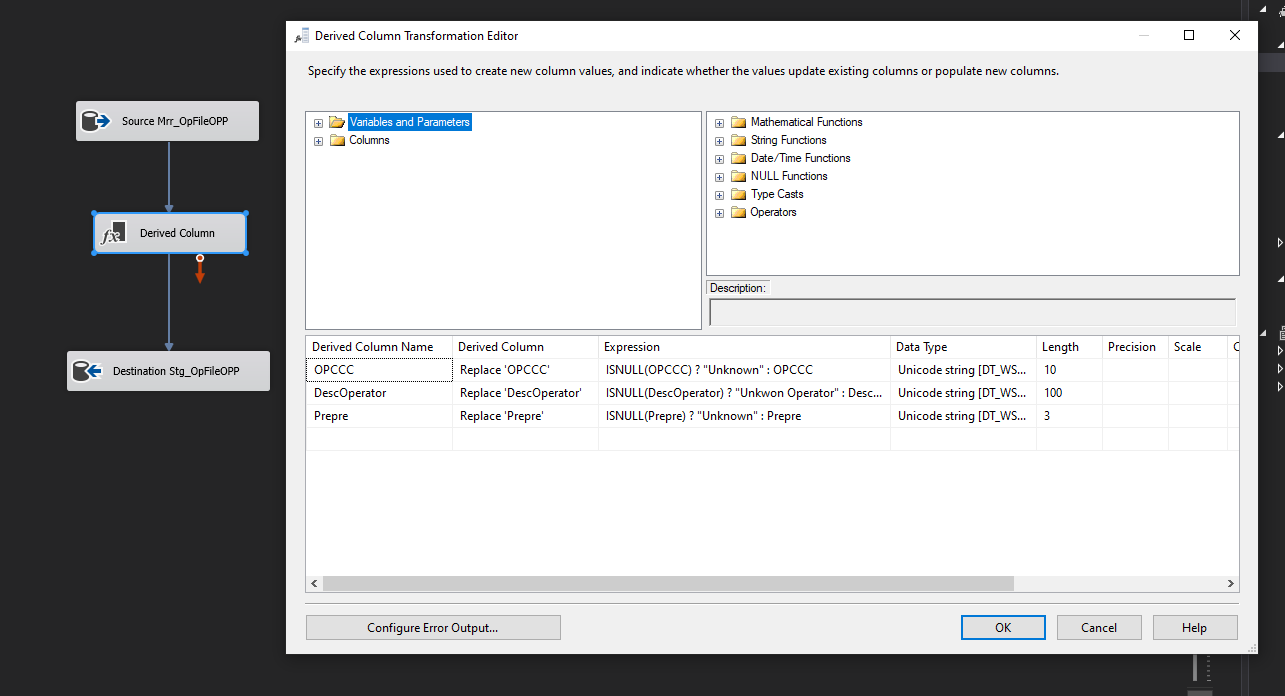
SSIS containers for MRR stage:

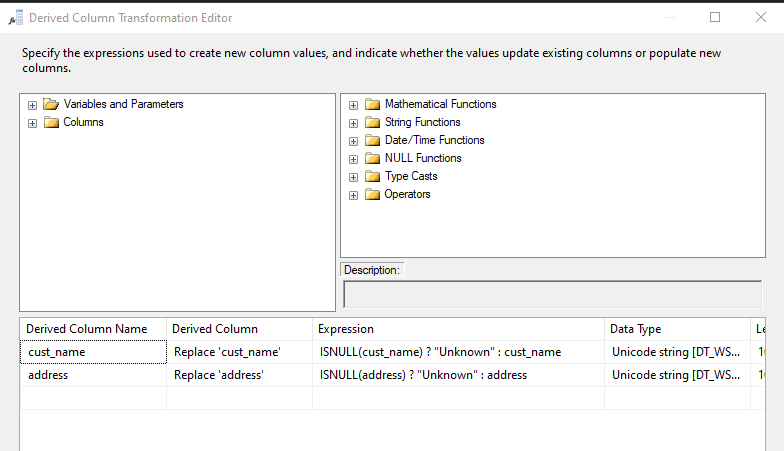
Variables for importing data from CSV files:



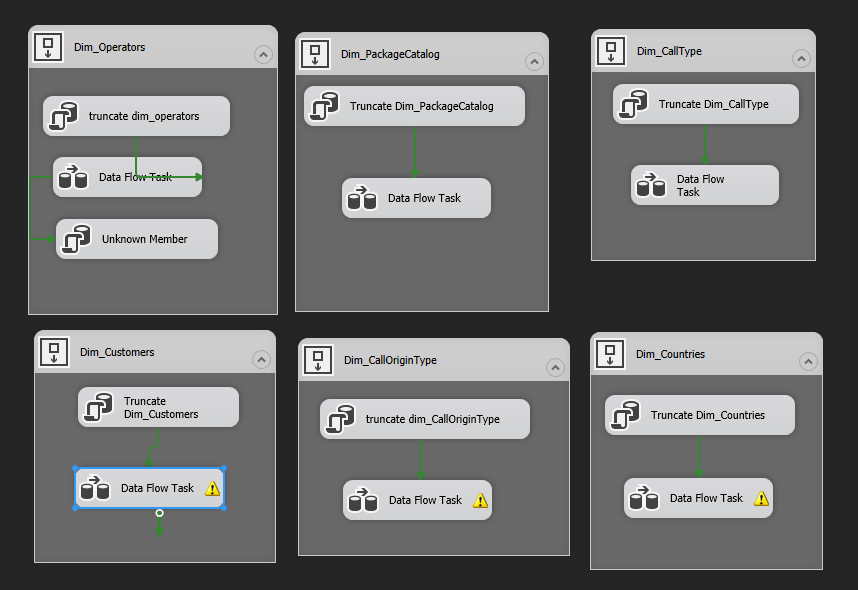
Data flows examples:

The Voice STG – After mirroring the data and combining it to a single data source, it is time to work on the data and data cleansing. By using the derived column tool, we dealt with Nulls, adding new columns, removing duplicate values, unknown member and converting data types.





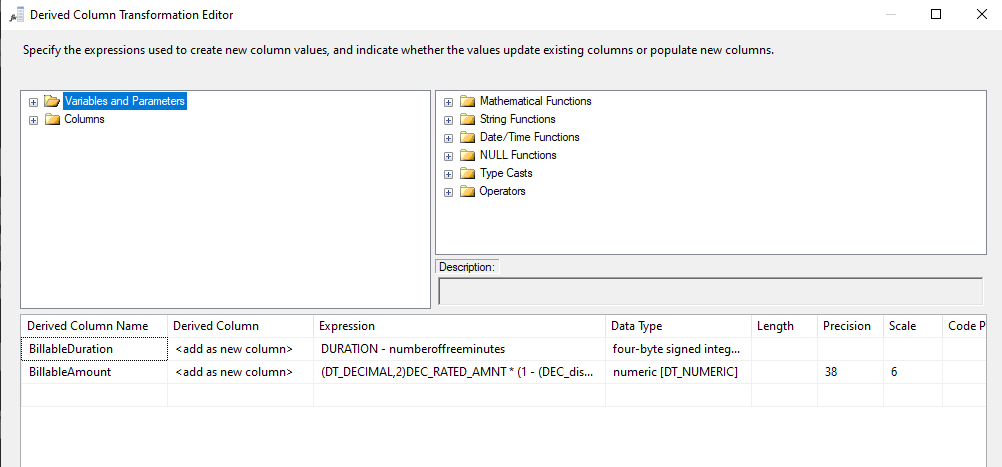
The Voice Dim tables – Updating data in dimensions tables for fact table. Dim tables describe the events in the fact table.

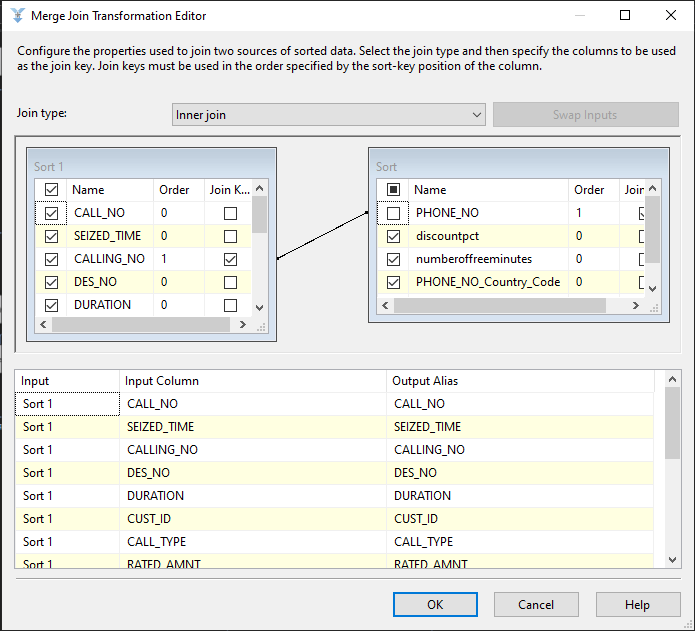


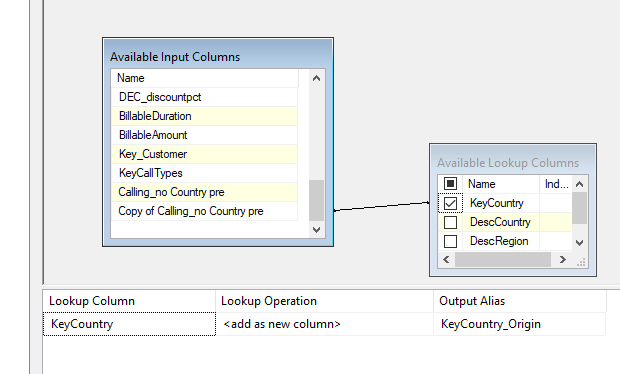
The Voice DW – In this stage, we have created a fact table with measures and surrogate keys connected to Dim tables using star scheme. We created unknown members in order to keep data integrity and avoiding data lost. Dim Date and Dim Time tables created as well on this stage. We have created a task that truncates all data from the DBs created and Load into all the tables the data from the operational The Voice DB including new transactions created.



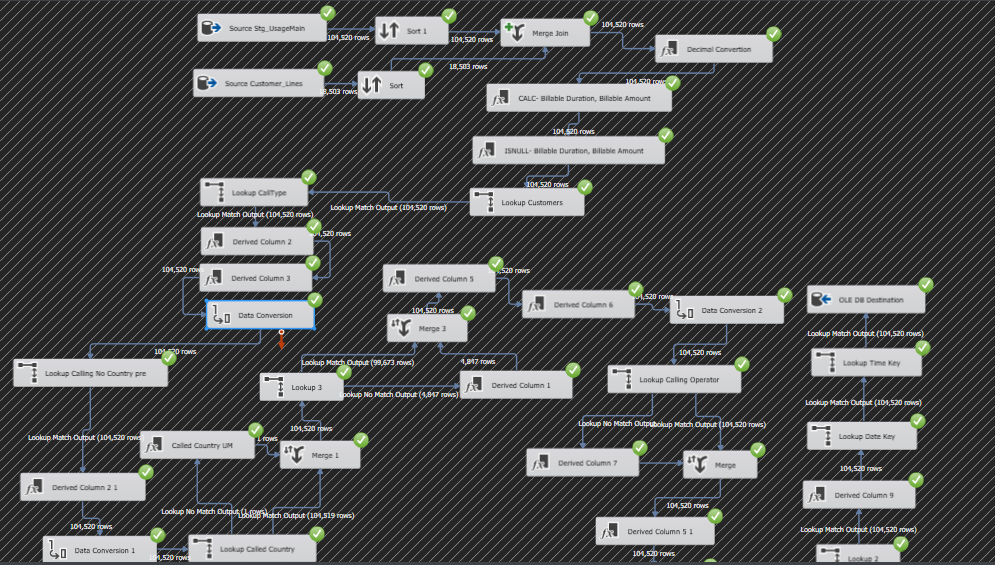
Casting data types from string to decimal:

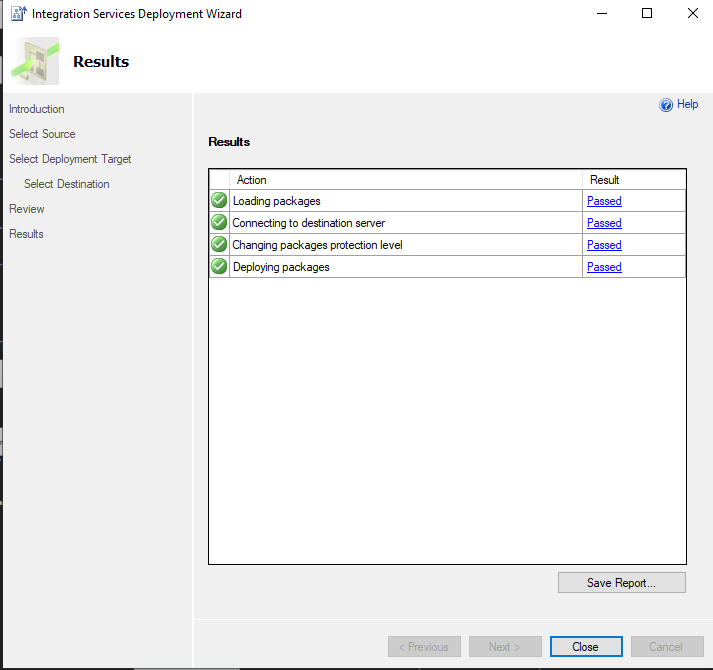


Merging data source

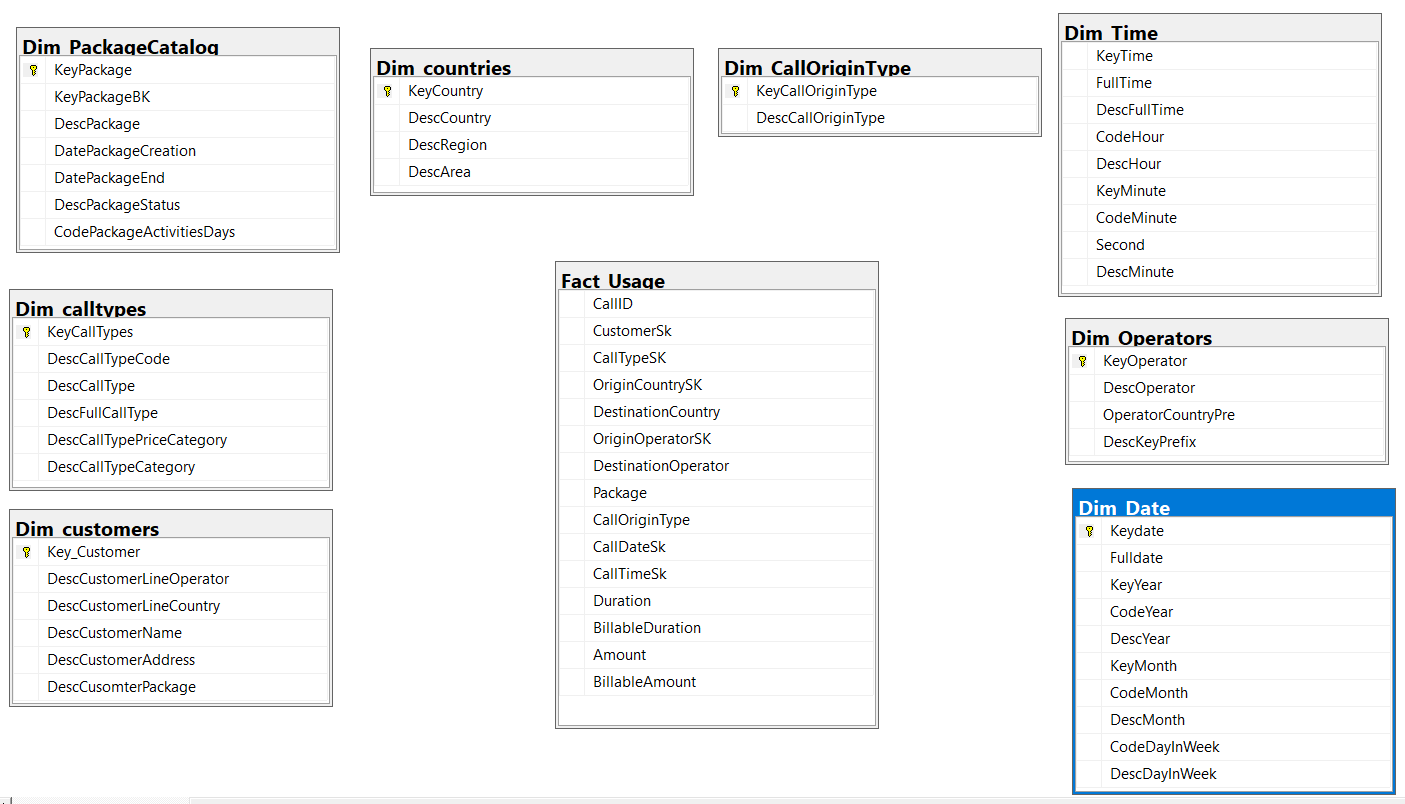
Lookup values:

Execute process:

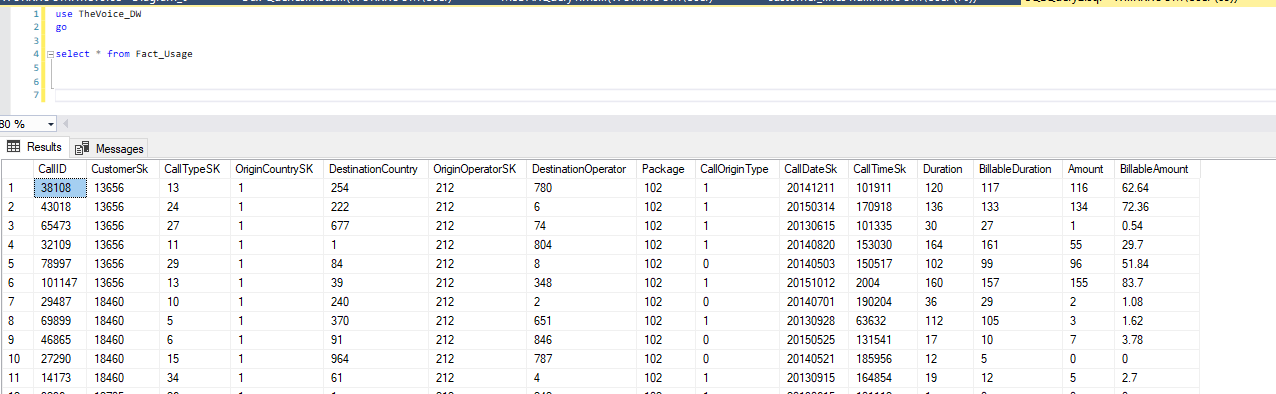


Deploy to SQL Server catalog:

The Voice DW ERD:



Data warehouse fact table result:



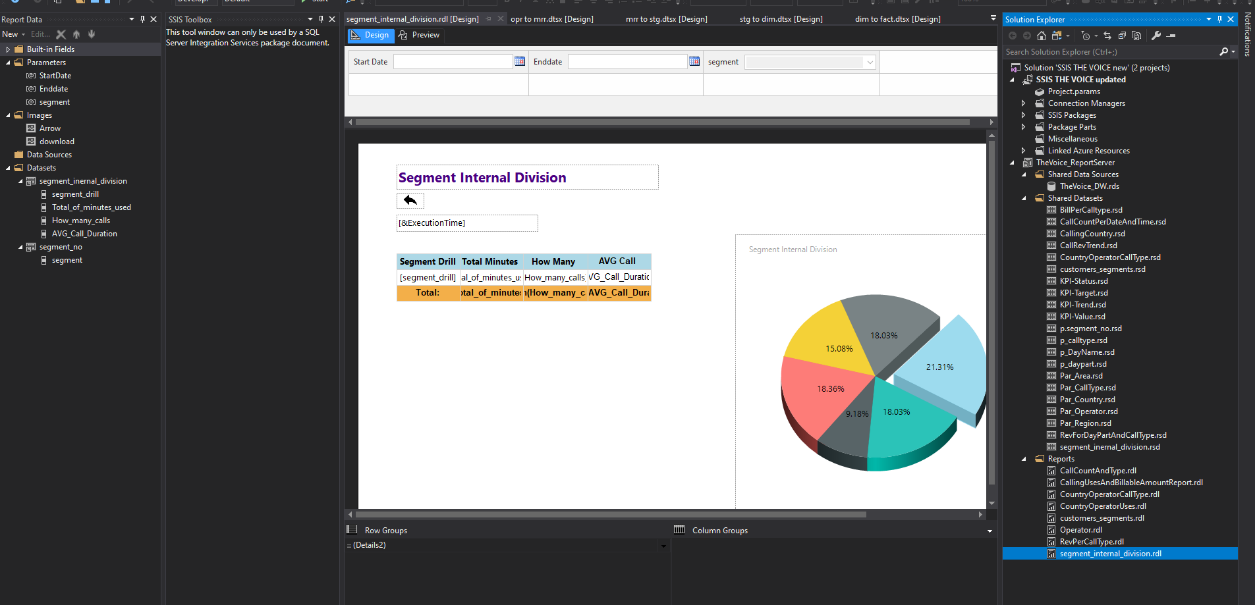
# SSRS

After completing the ETL process and data warehouse is up and functional, we are finally able to query our data and of course building reports. The SSRS give us the opportunity to build reports and to export to excel, which is more convenient for end users that are nor developers (The SSRS is built for developers)

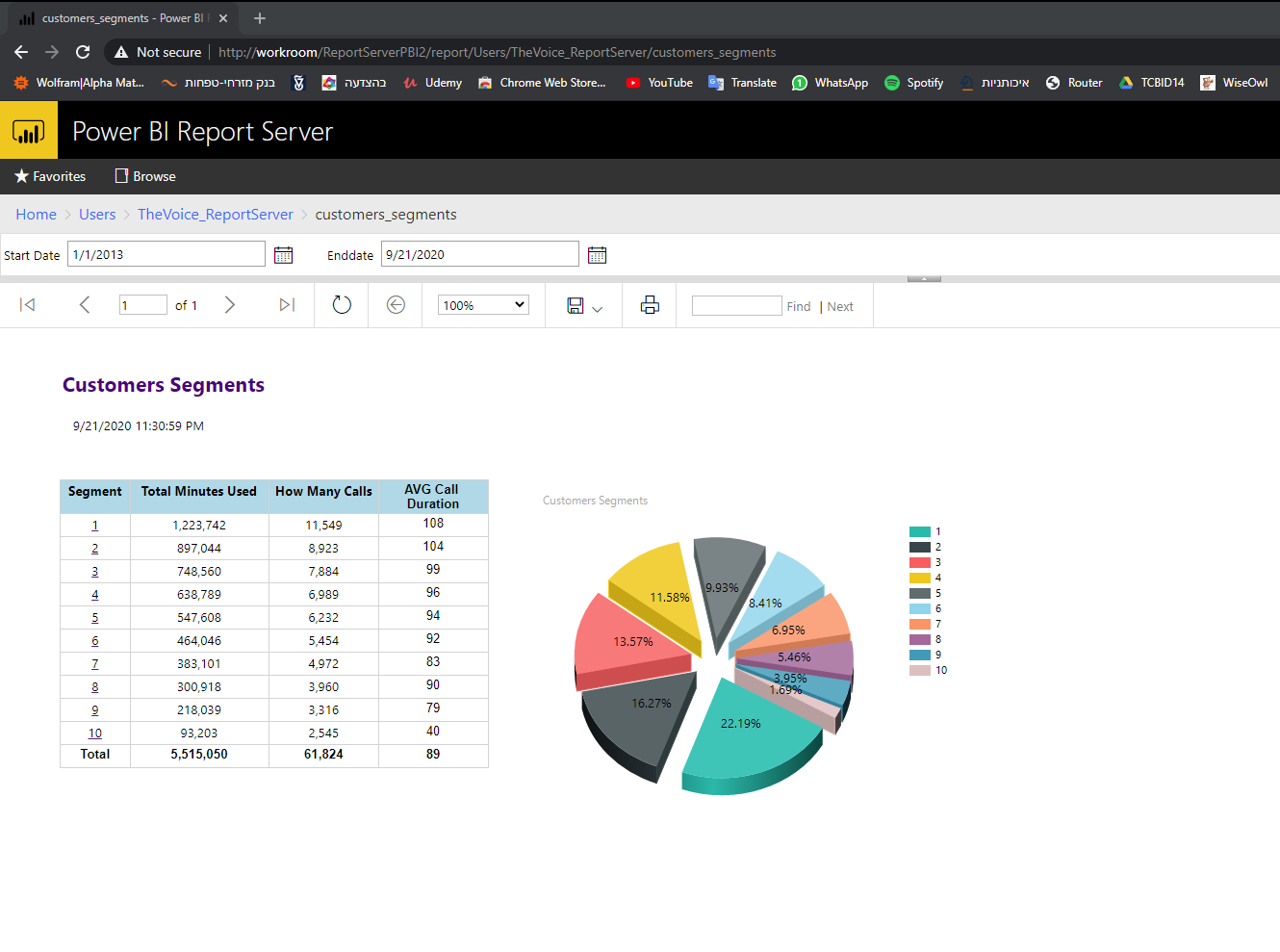
## General

As explained one of the key factors of having DW is the ability to query it directly. using SSRS Solution connected to The Voice DW, we created, designed and deployed reports to the report server. The SSRS can connect to as many as needed data source such as SQL Server and excel, also can export to excel. In addition deploying to the report server gives us the option to handle permissions, create users and arrange folders accordingly.

## Reports

We have created four reports using the SSRS, each of them relay on the DW as a single data source plus SQL queries for data sets. Handled under the same solution as the ETL process. Once done we deployed to on premise Power Bi report server. In the screenshot below, you can see the datasets, reports, and on the left the search bar parameters:

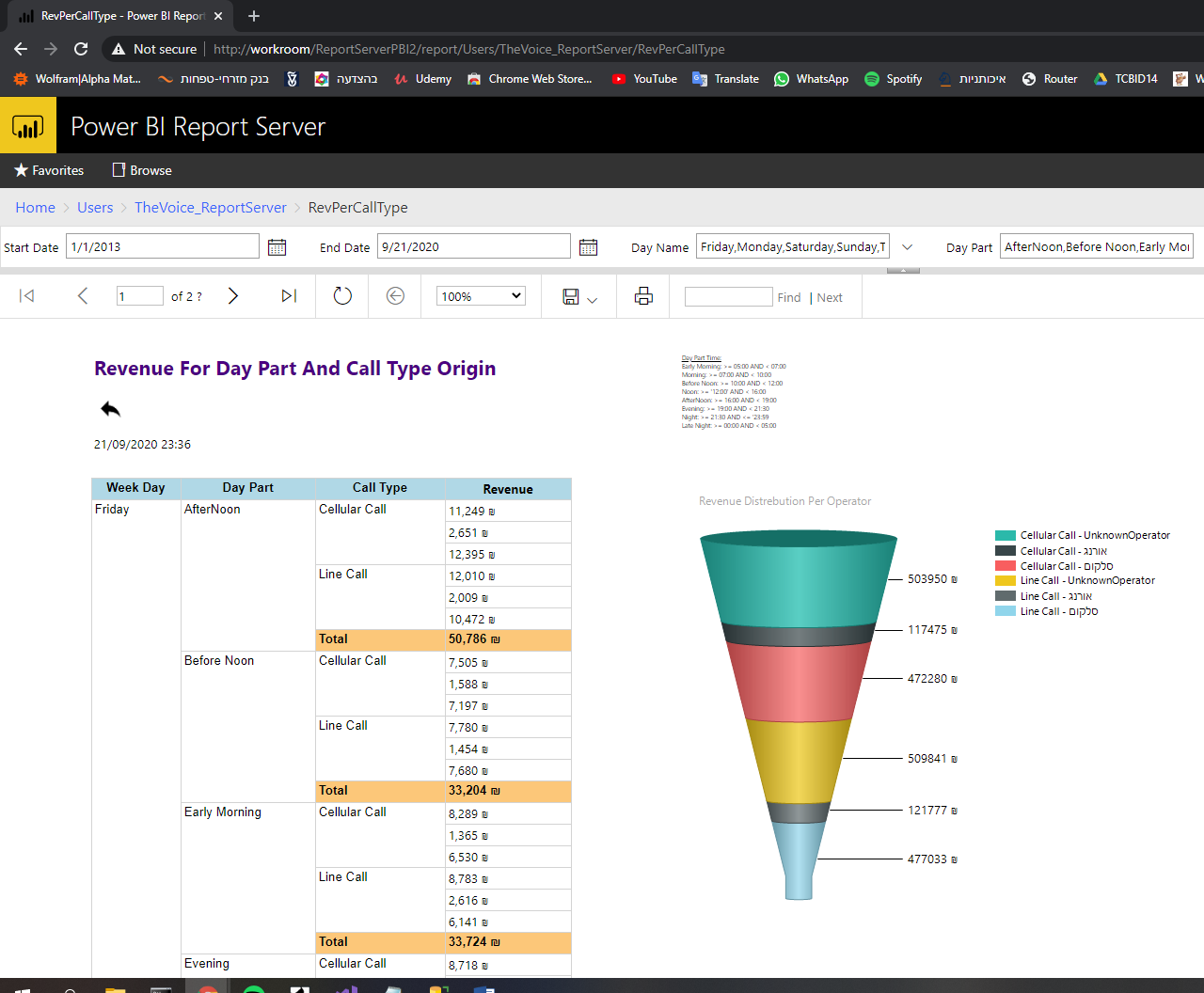
### Report 1: Customers Segments

Dividing all the customers to percentages based on their activities. From 10% of the customers with the highest usage to the 10% of the lowest usage. Clicking on a specific segment direct the user to another table that present the percentage inside the chosen segment. The table indicates the specific numeric characteristics of each segment for example 10% of the customers makes 22% of the calls or 30% of the customers makes more than 50% of the calls

### Report 2: Total calls and type by weekday

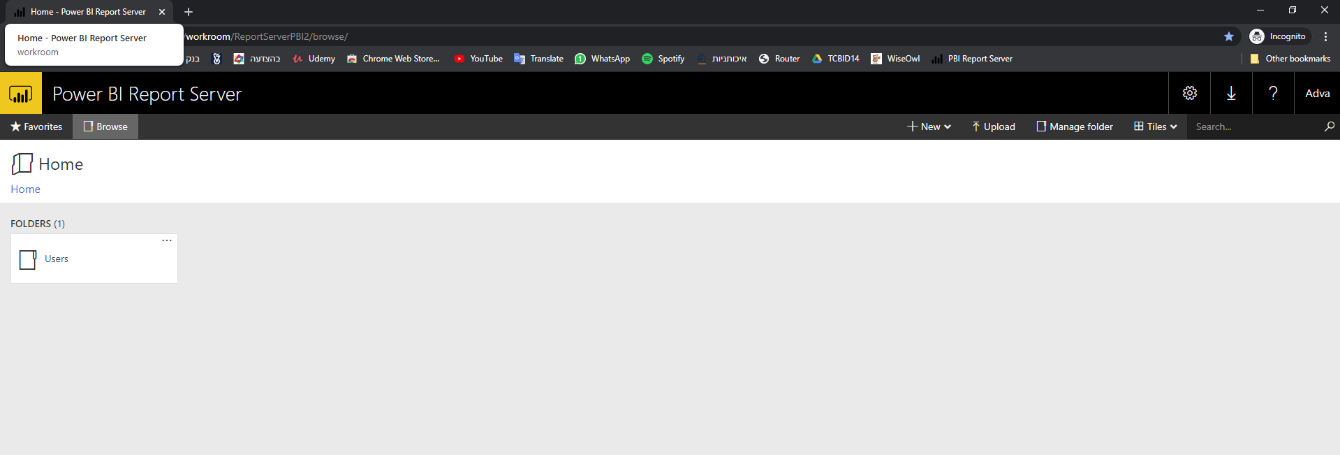
Summarize the number of calls per weekday. User can drill threw a specific day and see how many calls in each day part: Morning, before noon and evening and understand which day or day part indicating the weakest activity and which with the best activities. This information is important to allow the business to address deferent solutions in order to enlarge their activities in weekdays or part day for example.

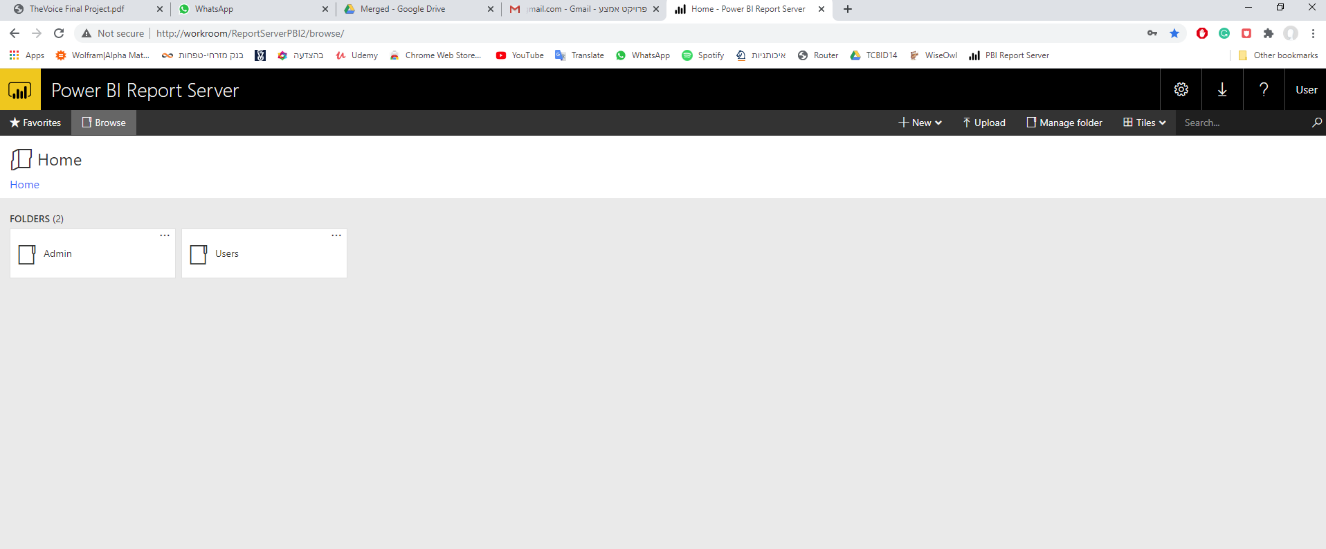
### Report 3: Revenue for Daypart and call type origin

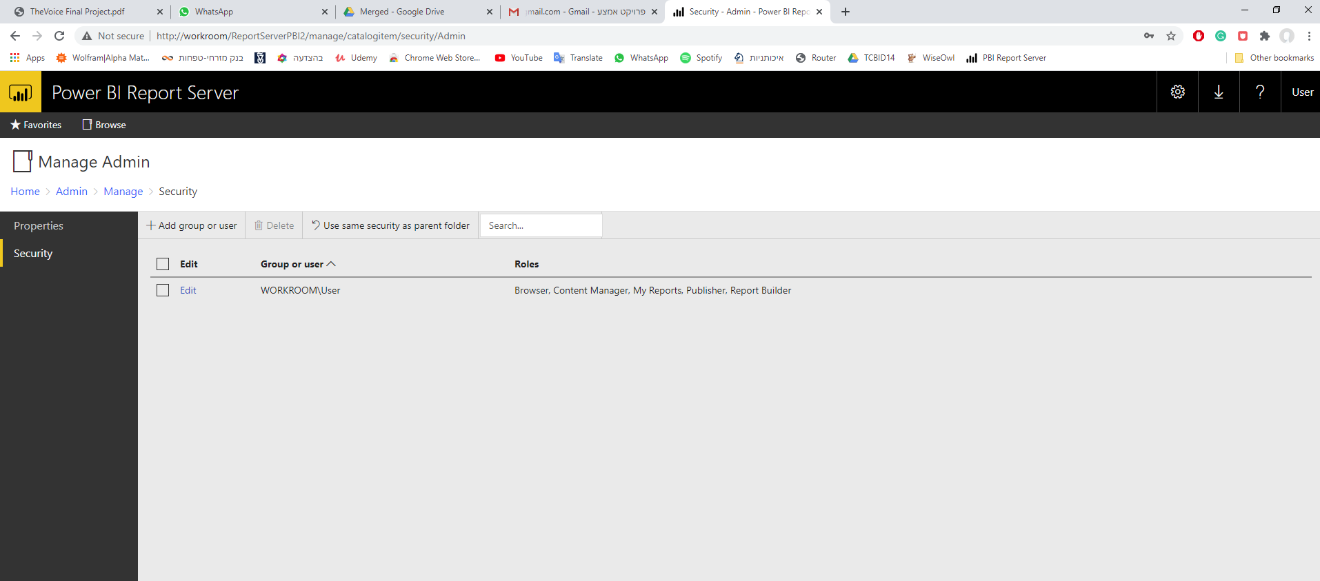
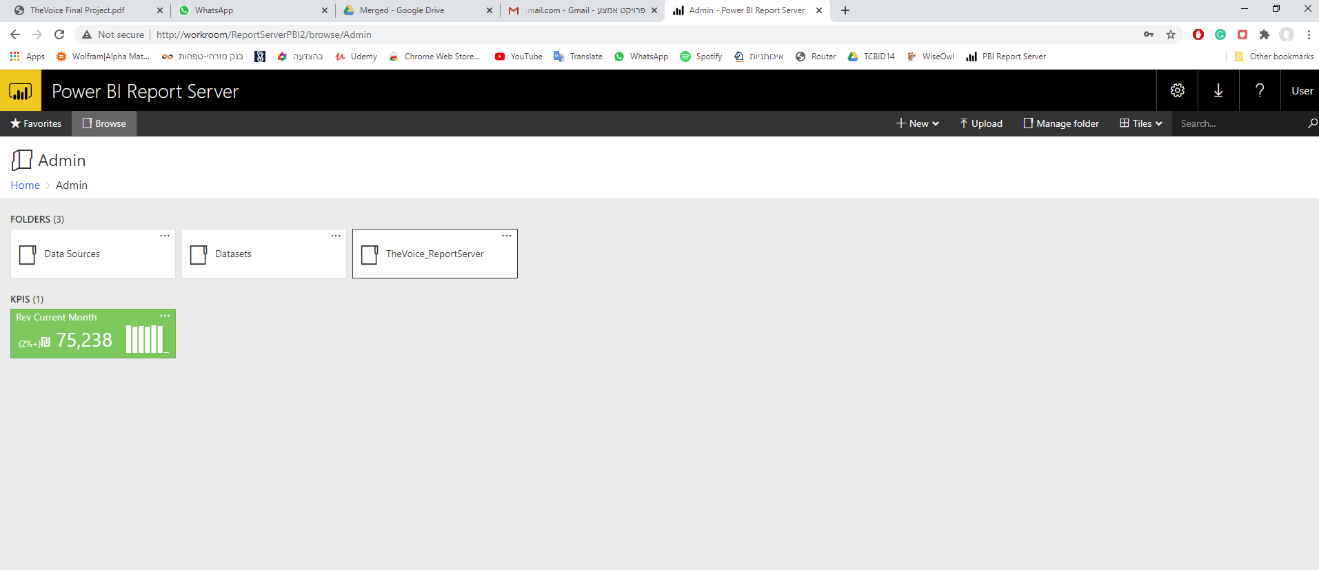
User can click a specific day in report 2 (Total calls and type by weekday) and get into this report that present the revenue each day, daypart and deferent destination call type (line or cellular).

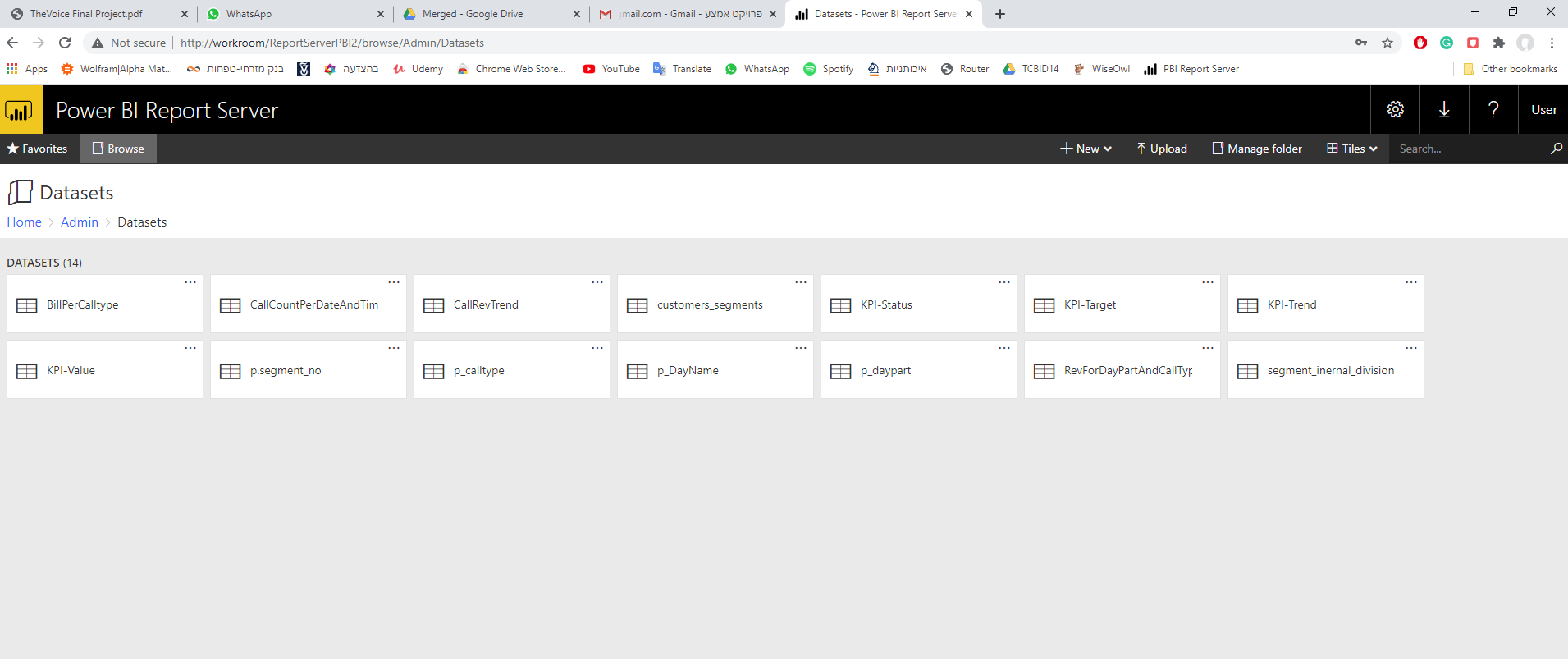
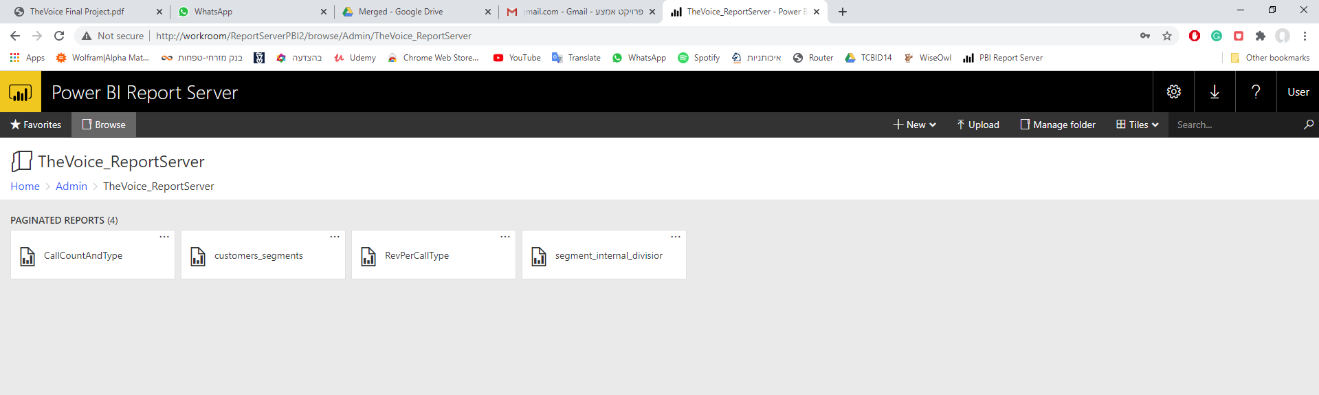
## Report server security

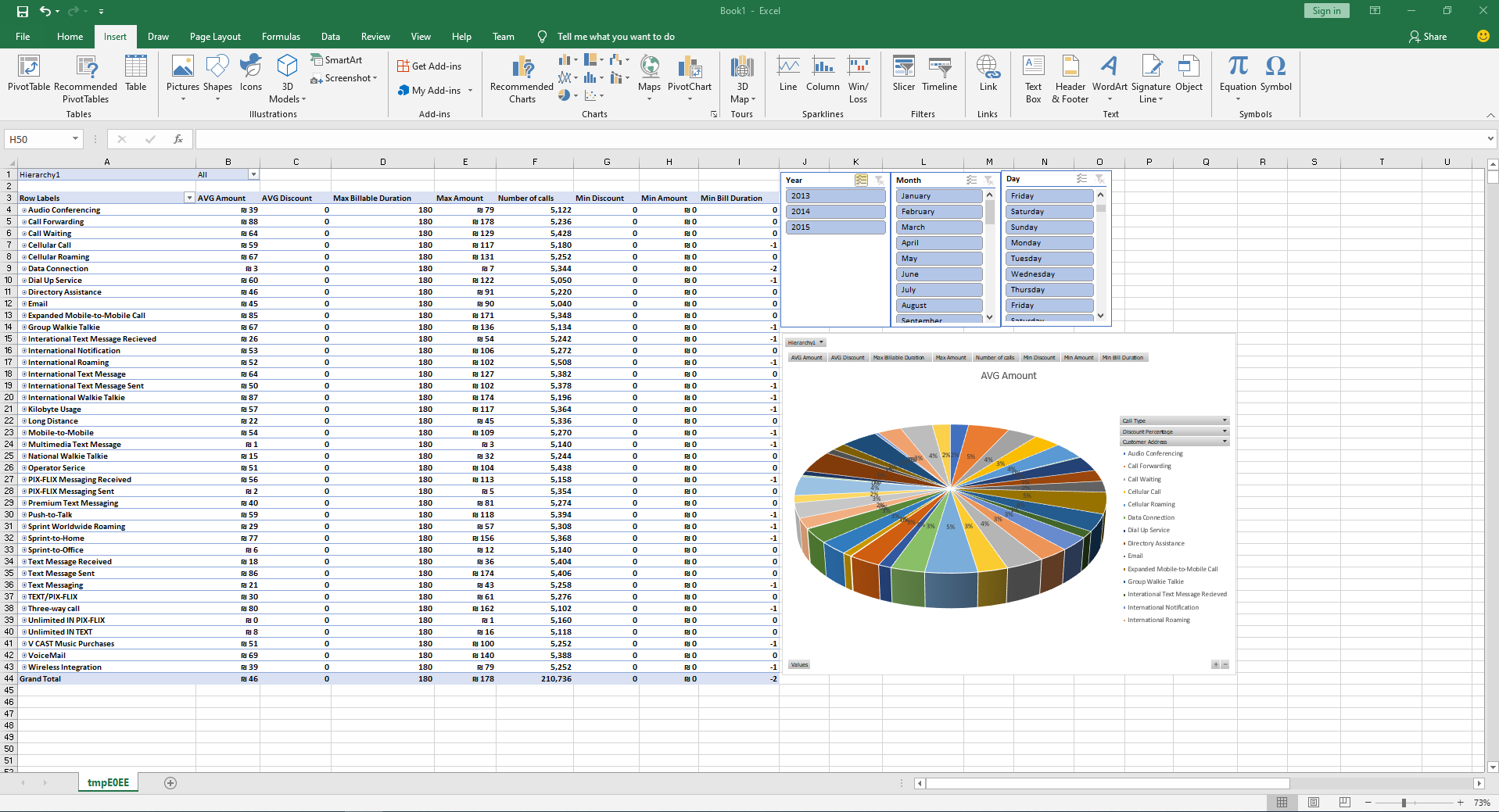
We defined 2 users – first was admin and the other as end user. The different between the folders is that the KPI’s are visible only under Admin folder, Therefor User Adva cannot see it.

* **User –** Windows admin user
* **Adva** – Second Windows user, cannot see Admin folder, only Users folder.



User name “user” (admin permissions) view:



KPI VISUAL (EXCEL):

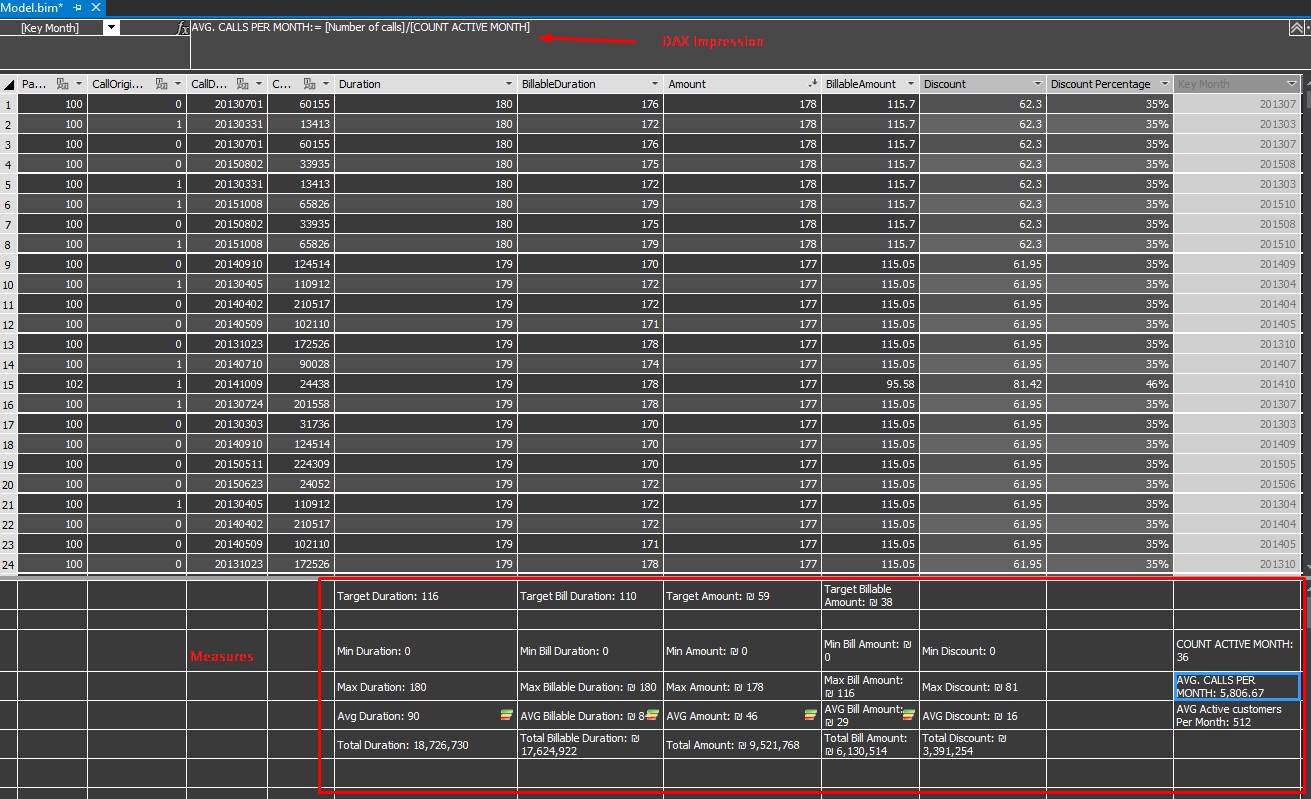
# TABULAR MODEL

## Why and how

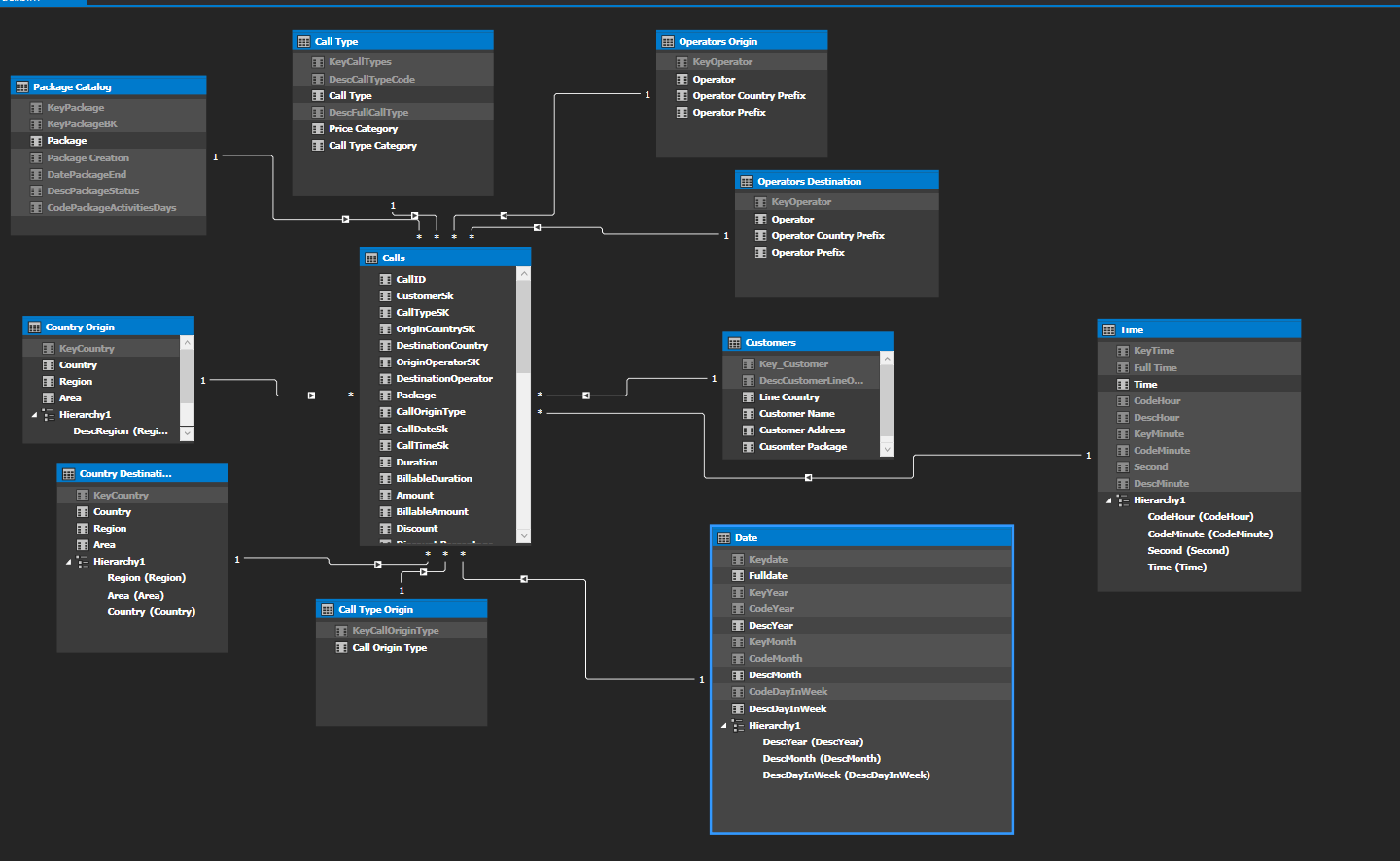
Tabular model uses in-memory instead of disk space, this advantage lead to better performance. The solution is fully implement by SSAS, we have created tabular model for end users to consume and analyze reports. Doing so by creating measures and dimensions available for slicing and KPI’s that enable a snap status on deferent business questions. For security control, we created perspectives for CEO, Analyst and Operations with deferent roles and permissions for each. In order to increase system efficiency we created 3 Partitions: all calls, 2015, not 2015.

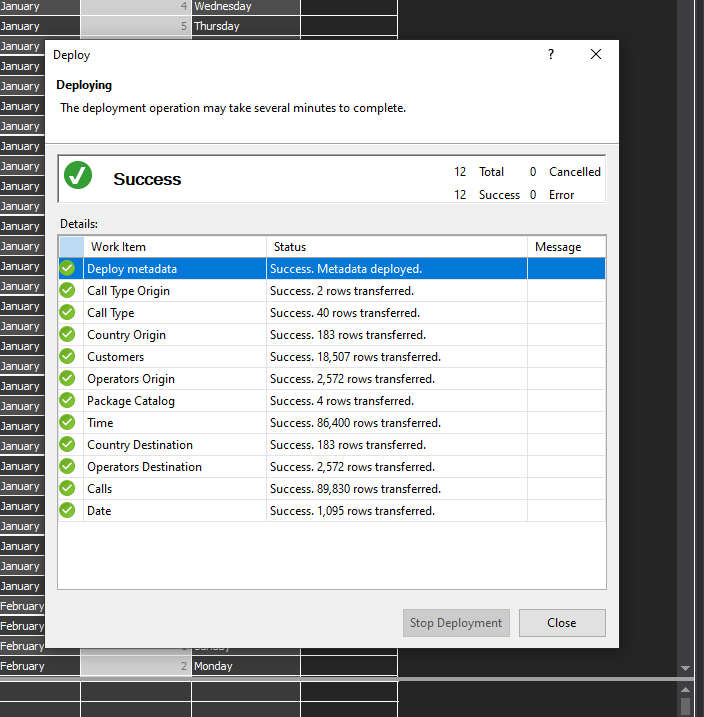
## Creating measures

Using DAX query we created various measures and KPI’s base on DW data like total revenue, average call duration, most called country. Another useful tool is to create new calculated column, we used it to calculate costumer discount percentage.

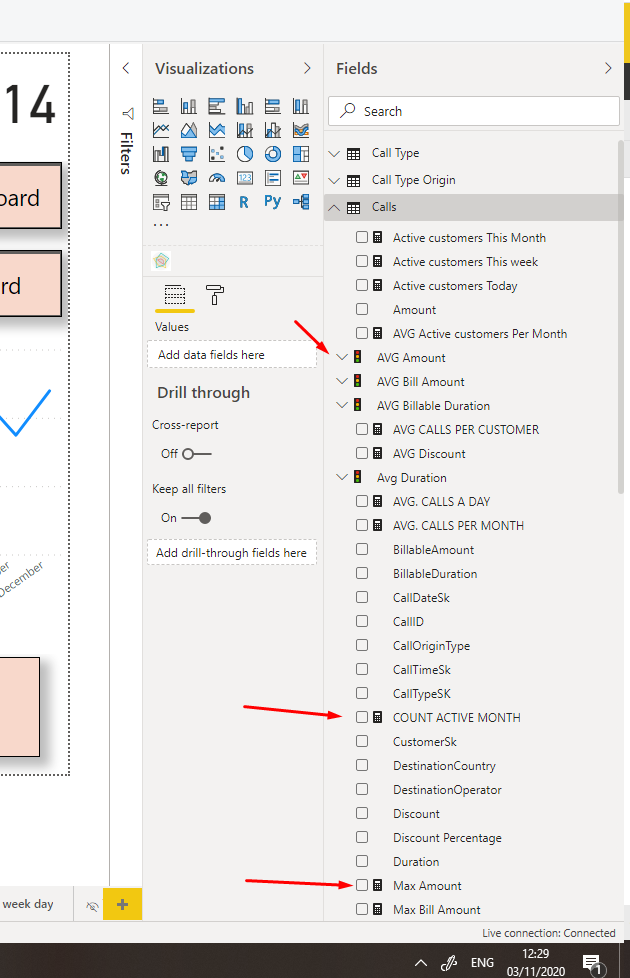


## Setting hierarchies

For each table we set Hierarchies and Relationships between the tables as seen below:

Deploy:

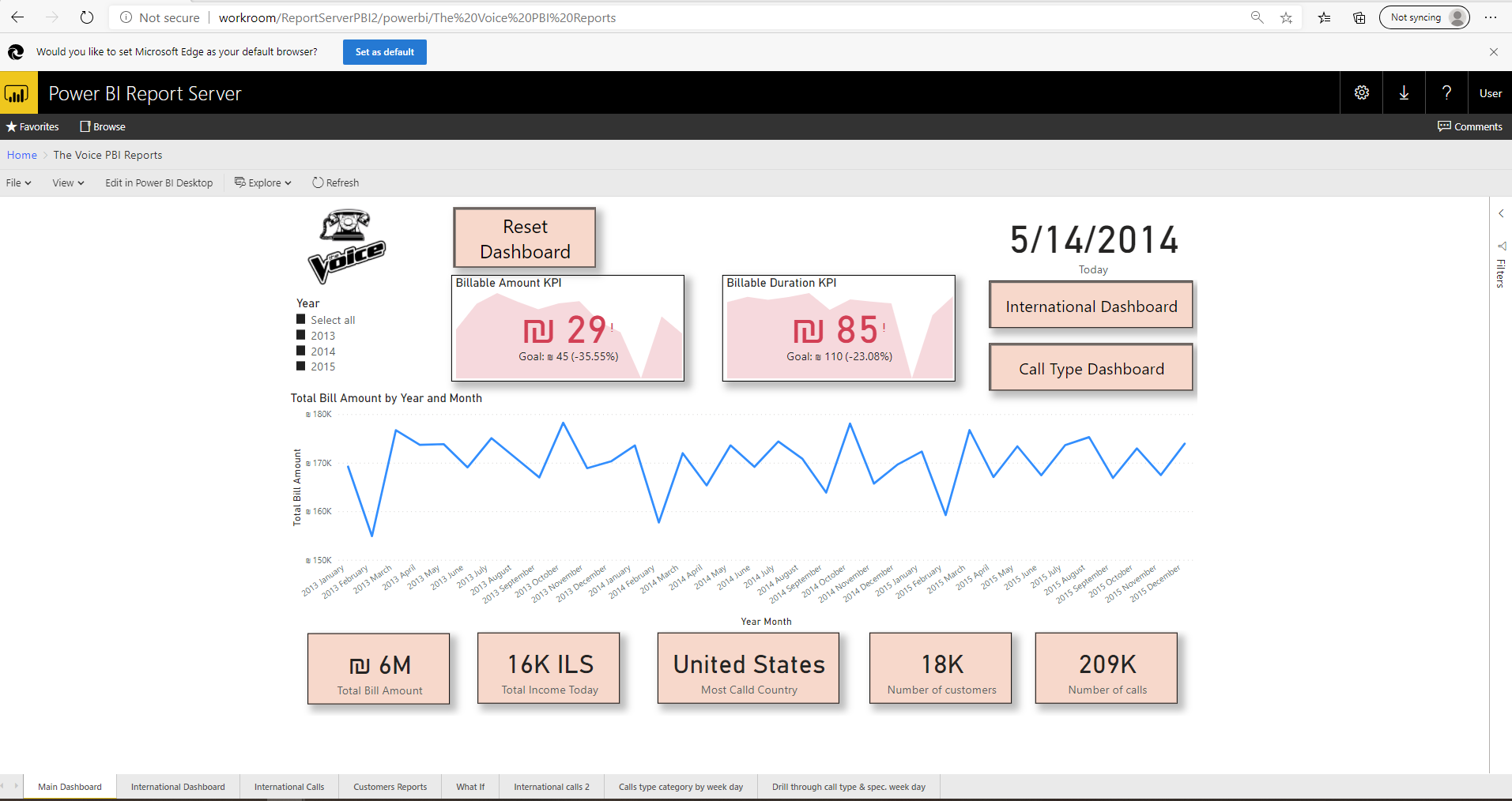
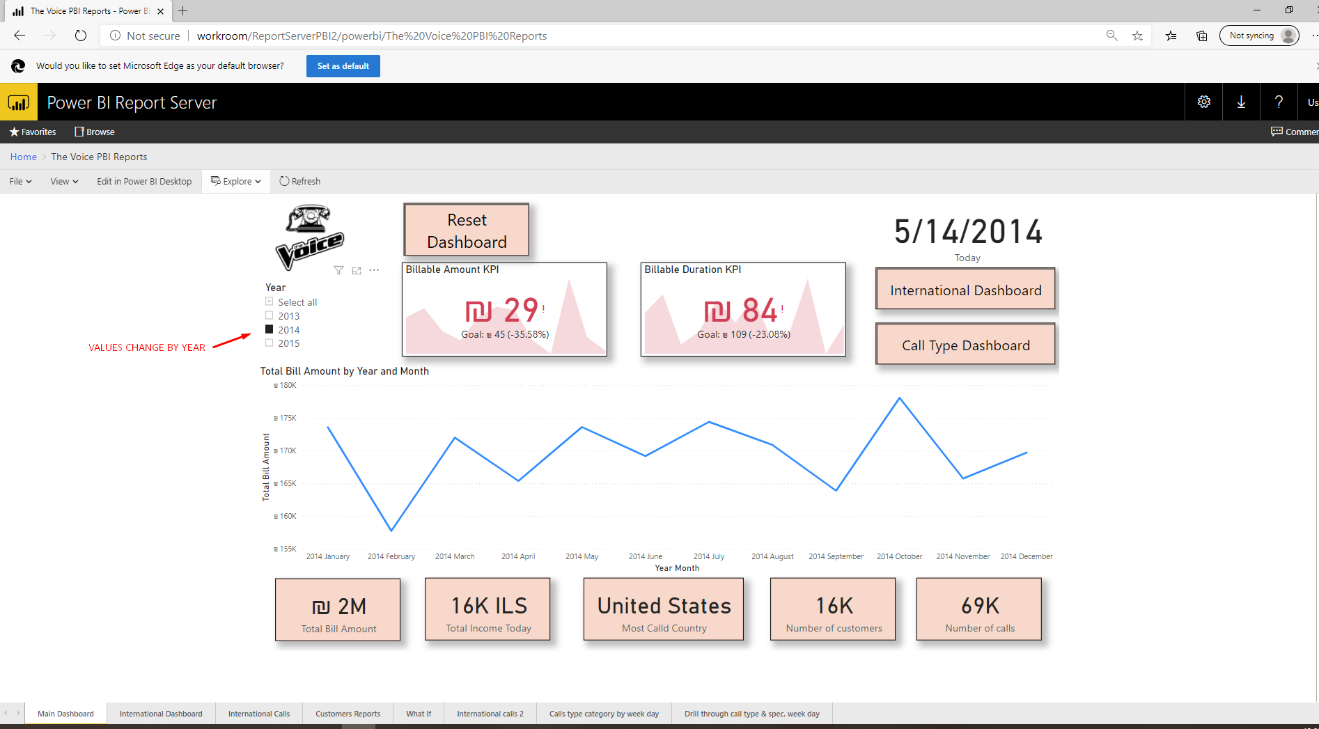
# Power BI

In the Power BI, we have set the tabular model as a data source for our reports (see below screenshot from power BI). It automatically pulled all the measures, KPI’s, table names we configure and created in the SSAS. We created 5 dynamic reports and 3 dashboard for the “The Voice” management and data analysts. Screenshot from Power BI desktop - Measures and table names as we set in the SSAS:

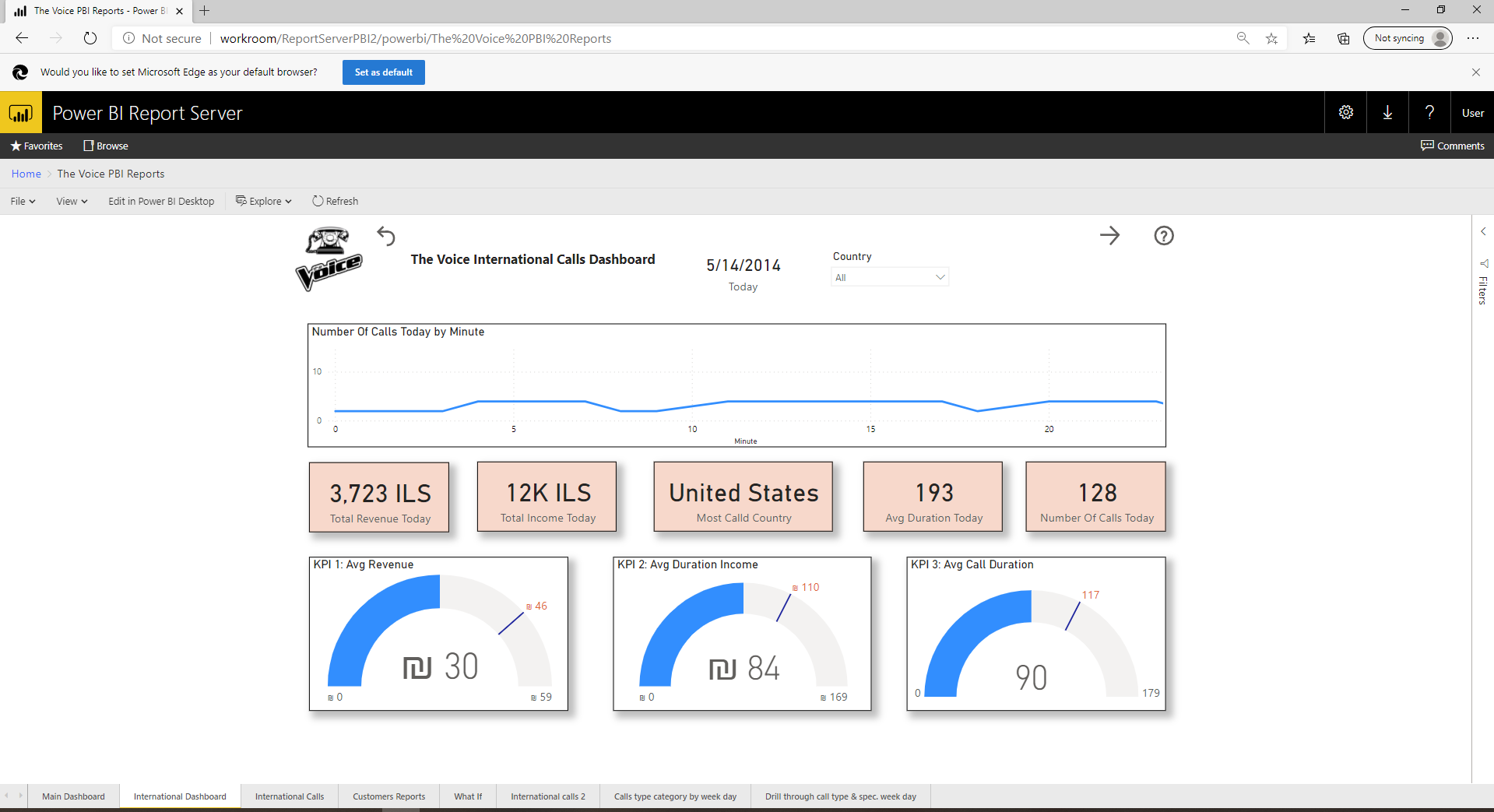
## Dashboards

There are three dashboard – main, international and call type. Each is a snapshot of today’s status. End user can consume KPI’s status and goals and to navigate to other reports as well. Mainly visualization are cards, KPI’s, gauge and buttons.

Main Dashboard - Today snapshot by cards, navigation buttons to international and call type dashboards.



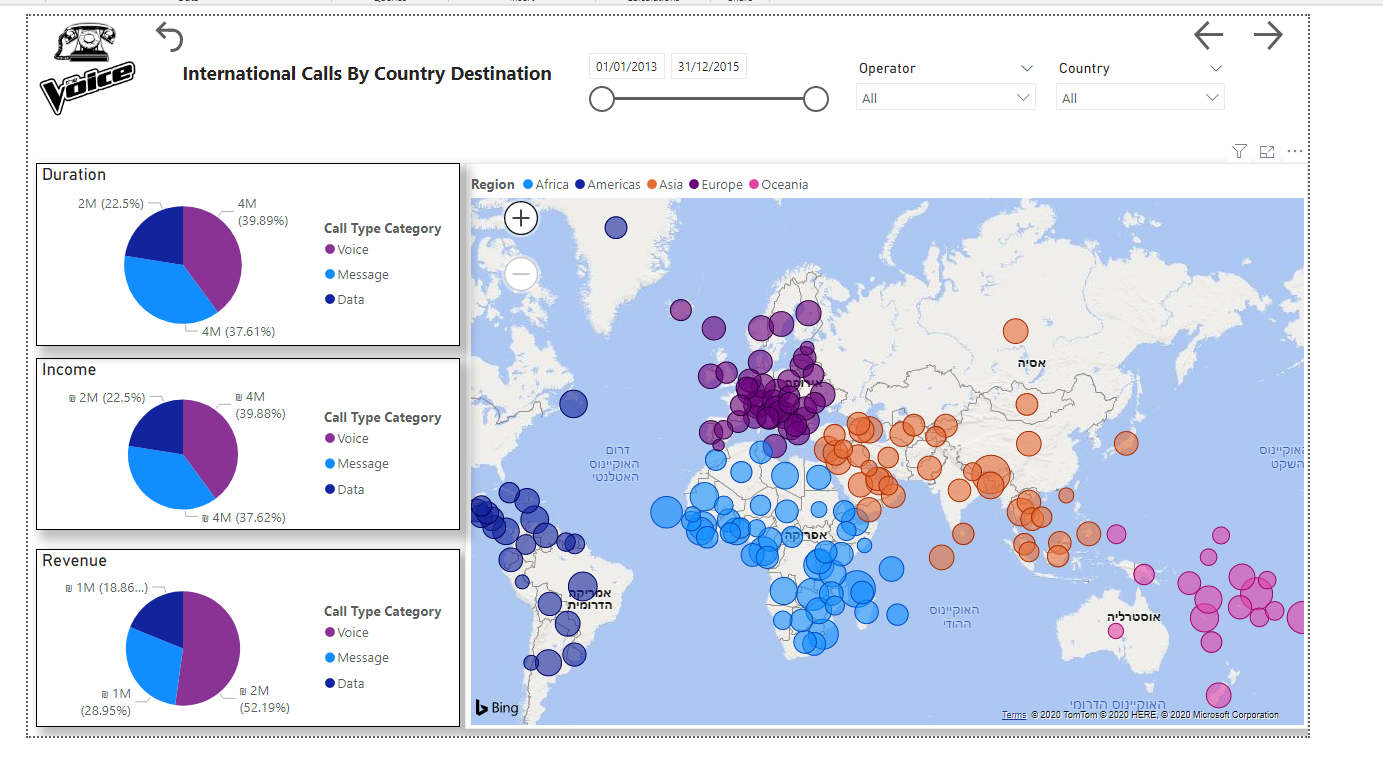
International Dashboard – Summarize all the activity done from Israel to other countries



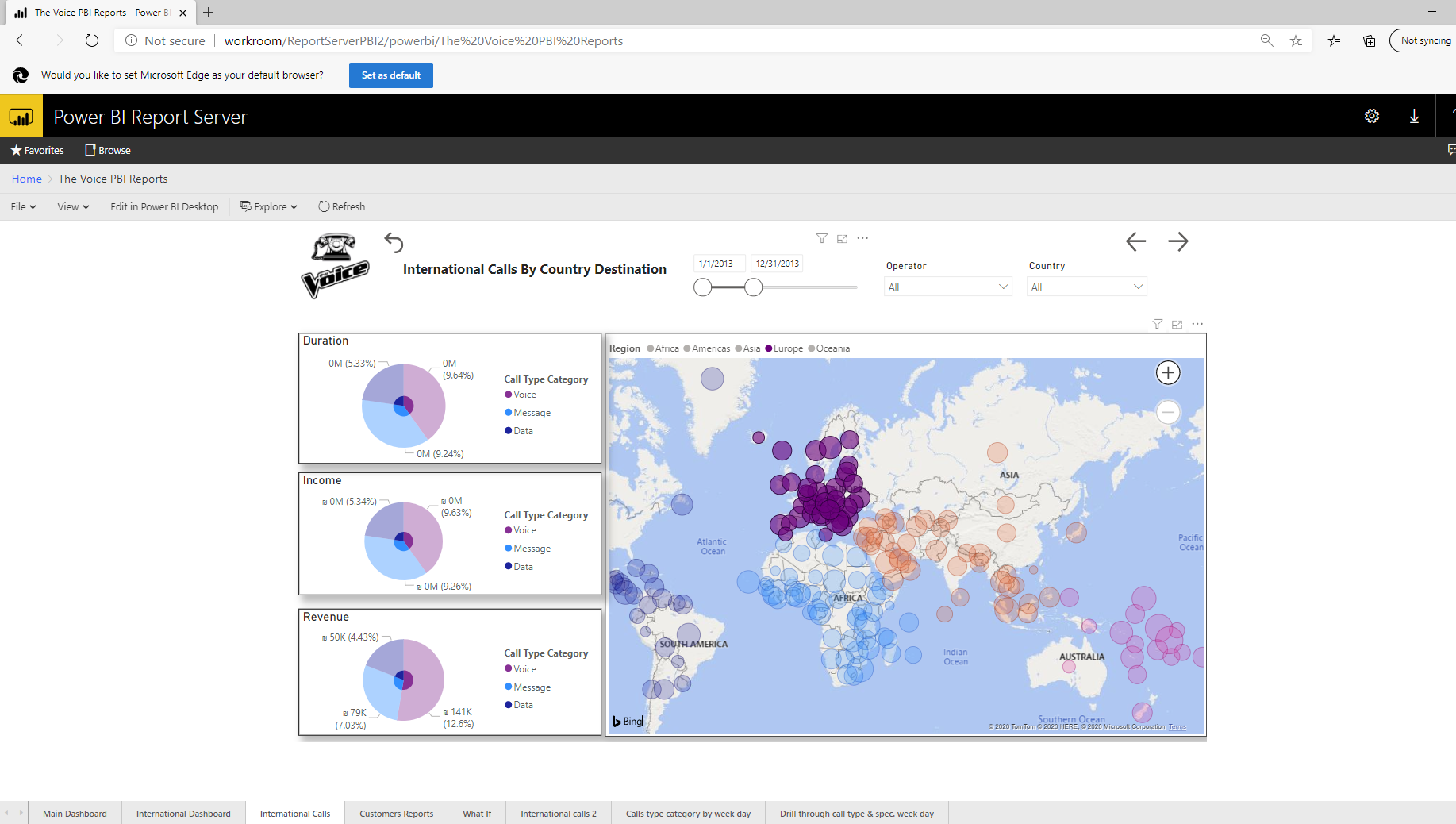
Call Type Dashboard - User can see the amount of call a day over the last 7 days compared to the AVG amount of calls per weekday. Two Dynamic Gages reflect the monthly amount of Calls and average amount, once user select a specific category. Also possible to drill threw to “Total calls this week by c all type” report.



## Reports:

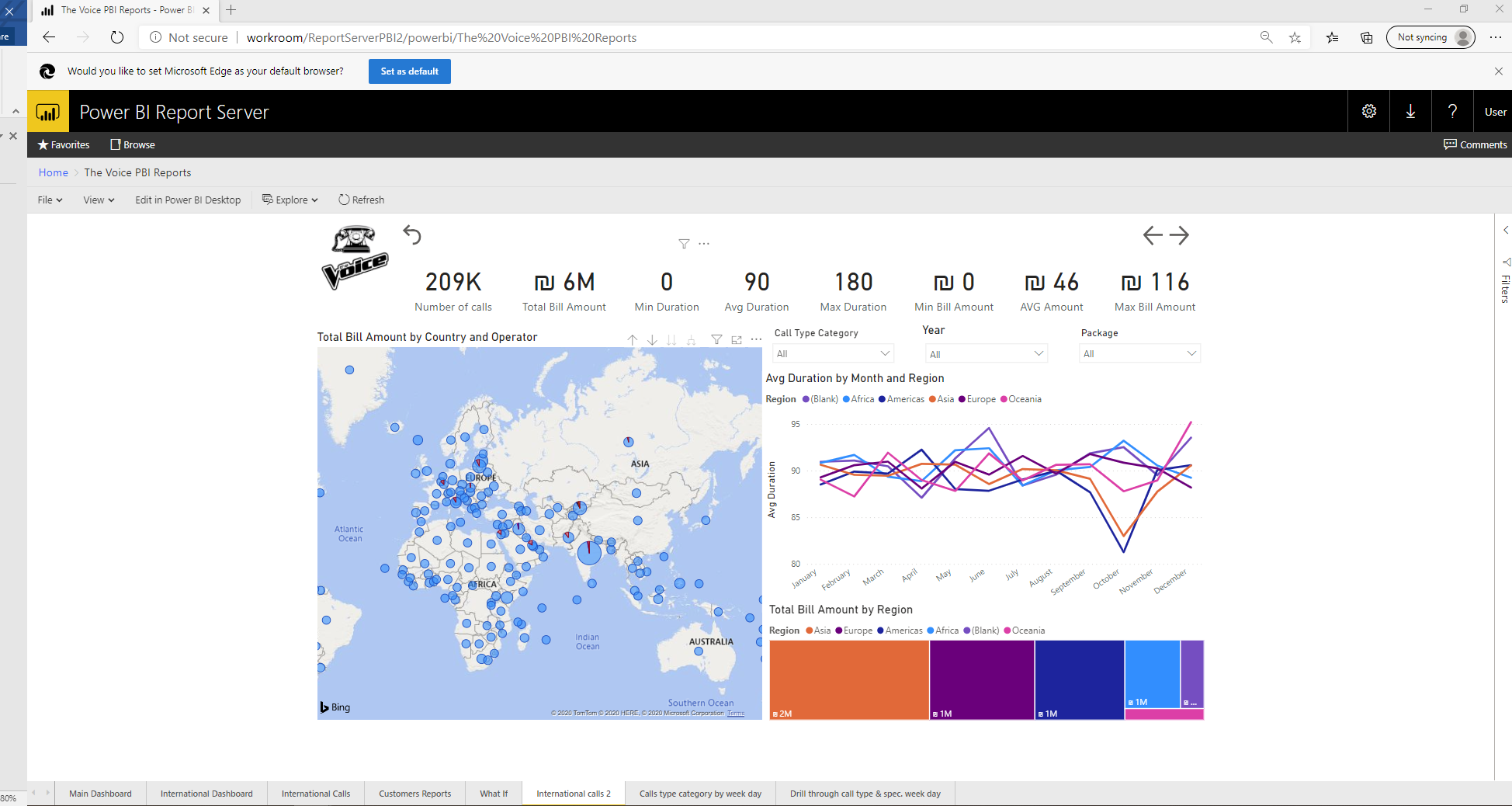
International Calls country destination - Can slice the world map by operator and call country destination and by region via the world map. In addition, on the left there are three pie charts based on measures for overview status.

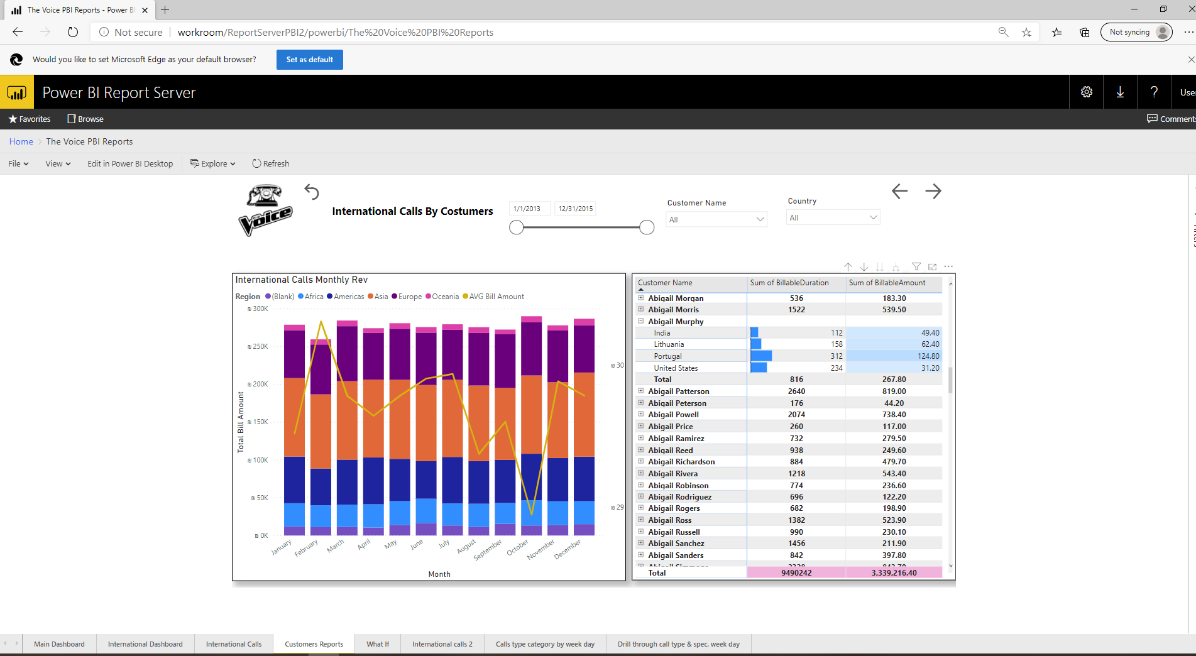
Filtering over Europe 2013:



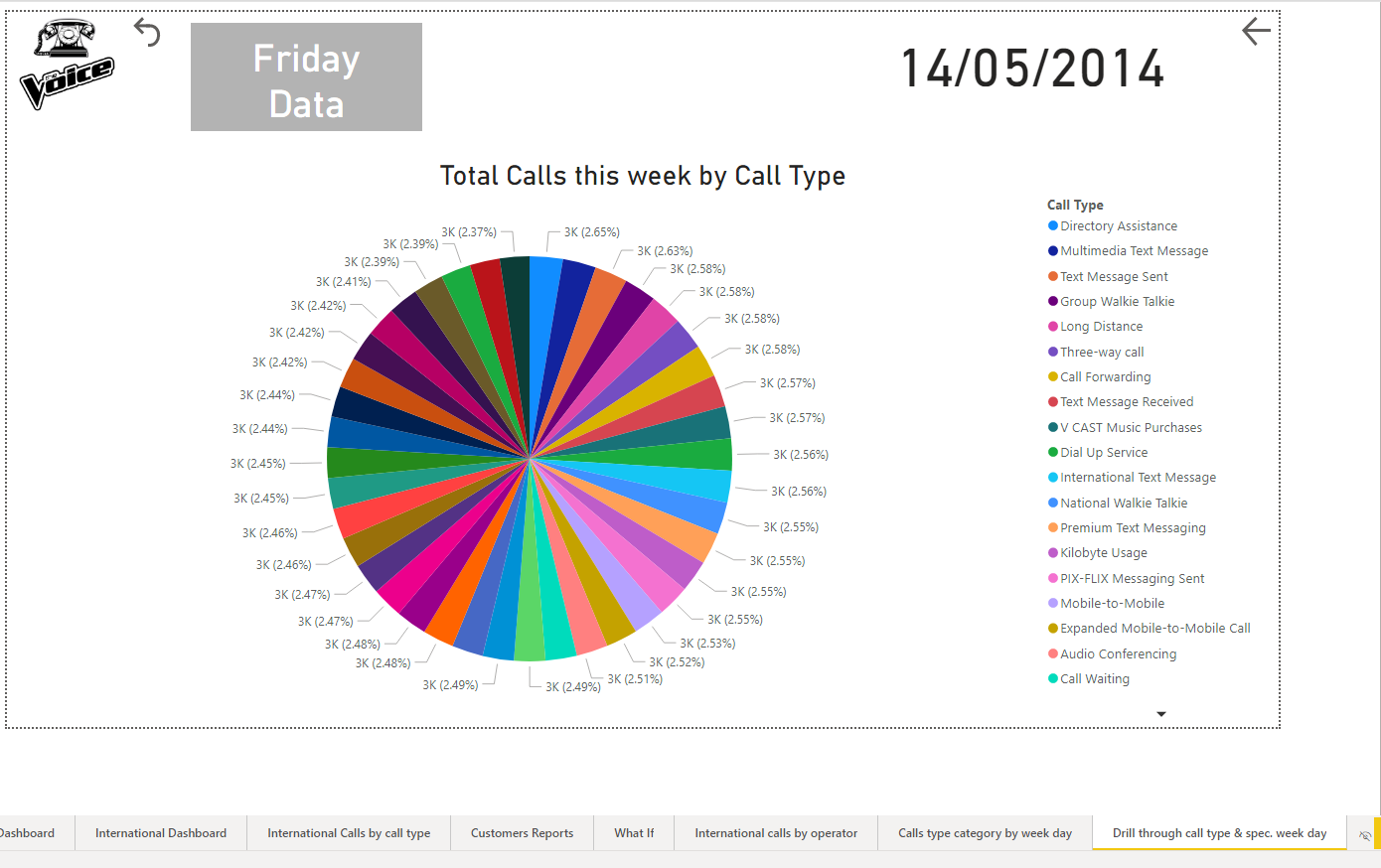
International Calls report 2 – The main different from the previous report is that here user can slice also by package. Report provide the revenue and duration of your destination calls. In the map visualization. Revenue available by region/area/country (drill down), the dots on the maps shows the different operators in the region/area/country. On the chart, you will see the average duration of the calls for each region during different month of the year.

At the top, you can see different information like number of calls, min and max duration and average billable amount and total revenue. You can filter the report by call type, year, and package and choose a region/area/country in the boxes under the chart (you can drill down to see area and country).

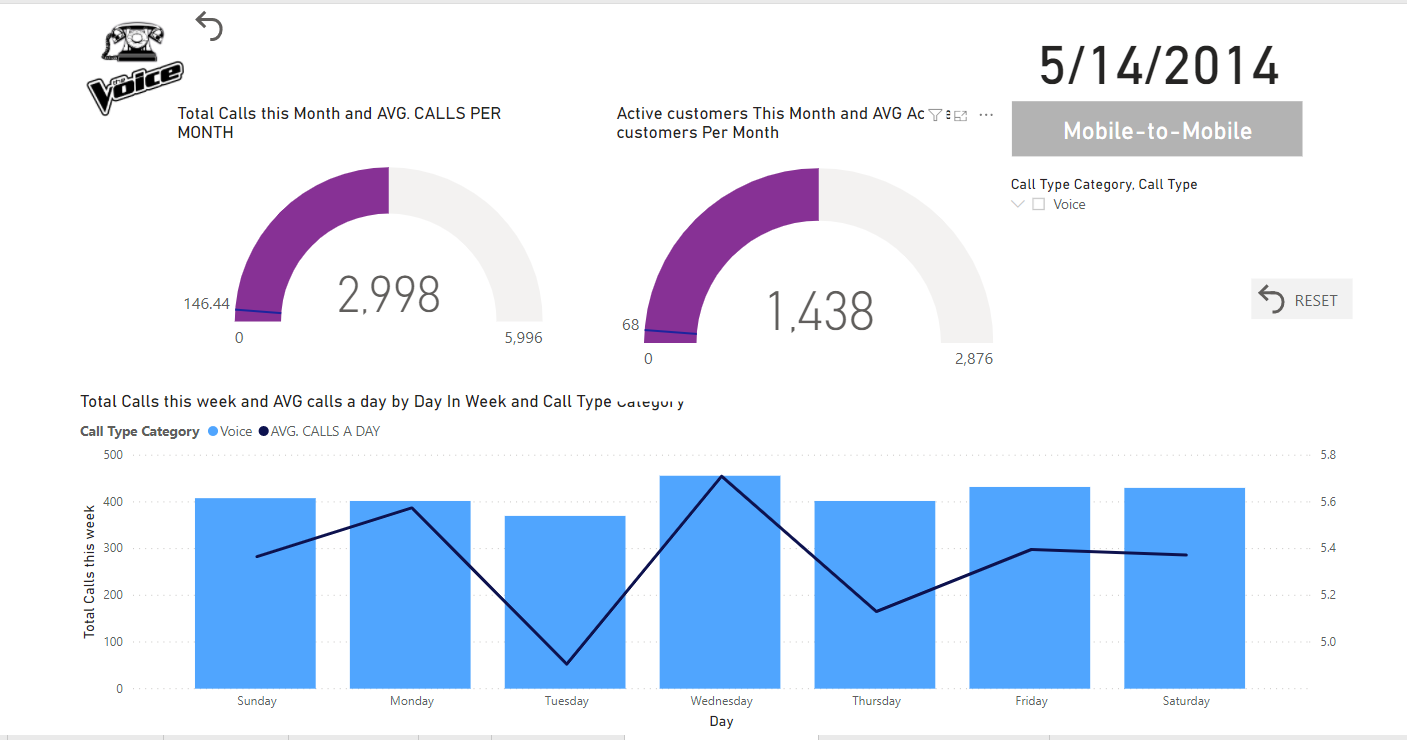


Costumer reports - Matrix for drill down each costumer called countries, duration and revenue. The left chart breaks down the total revenue per region plus the average changing revenue over time. Can slice the report by date, costumer name and called country. In the “sum of billable duration” column the blue data bar length, visualize clients ranking. The highest value will get full bar, and proportionally the other bars will set. Same idea apply to billable amount column just with range color.

Total calls this week by call type (Drill through call type dashboard) – Distribution of weekly call by call type. Also can drill down back to call type dashboard.



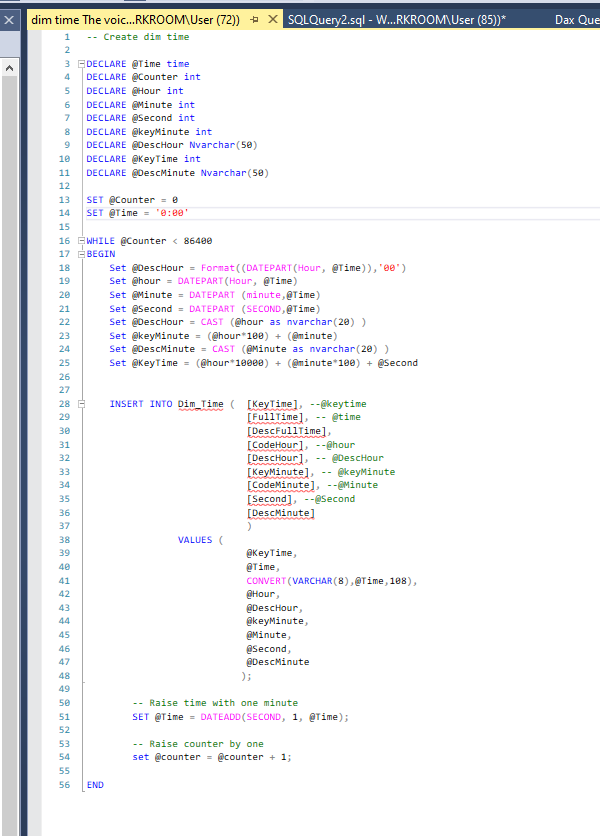
Back to dashboard via mobile to mobile drill threw :



# Scripts

In this section, you can review some of the scripts we use to fulfill the project solution. Of course, there many more, done in through all the parts listed above, but those can set a good example.

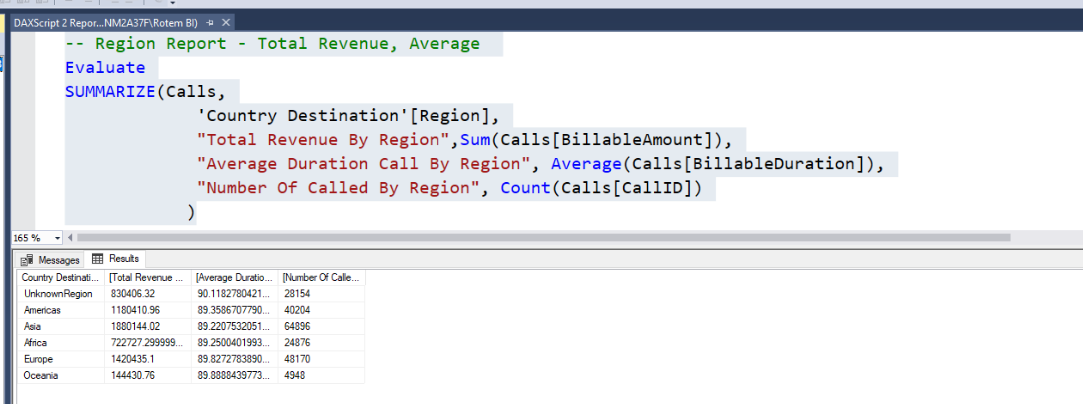
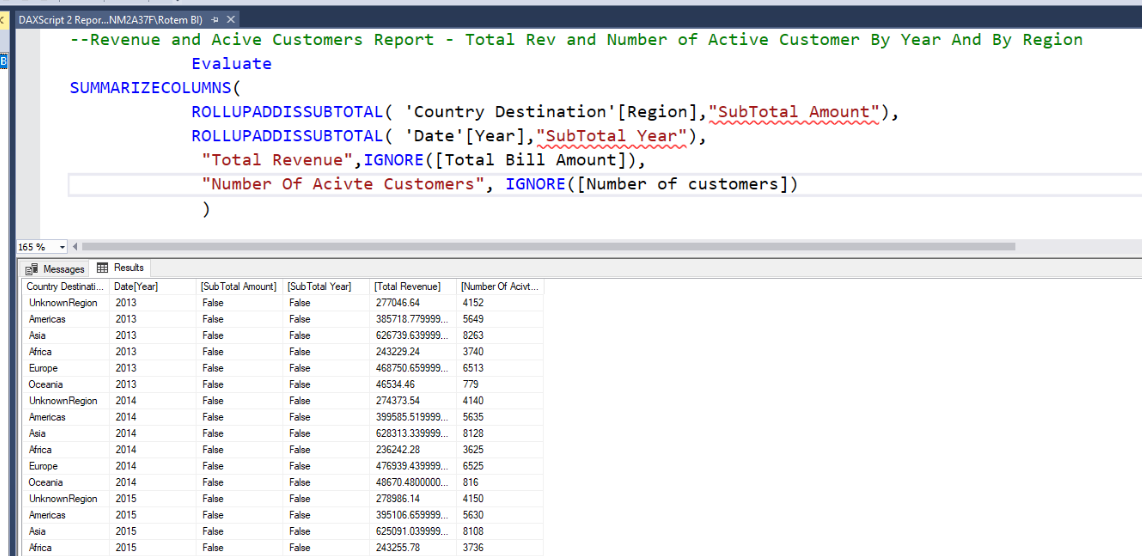
## SQL

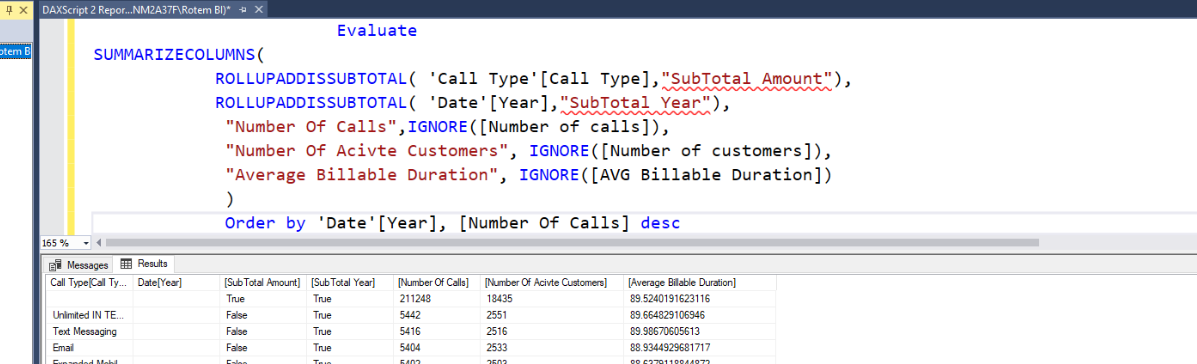
Dim time scripts – running right after creating dim time database and fill it with rows

Script goal is to add rows (=calls) to the operational database so we would have more data to analyze. Script is running over the dim date table and inserting random calls to usage\_main table accordingly.



## DAX

DAX query examples for creating reports in the analysis server: 



# Challenges and how we dealt with them

## Understanding the operational data base

The first struggle we encounter was to understand the operational database and the company business logic. Understanding each column data, the variables for calculating revenue and the connection between the tables are crucial for developing BI solution.

The key here was to massively query the database, searching for null values, duplicate values, blanks, corrupt data (such as negative call duration), cross join tables and even verifying value by calculating them manually.

## SSIS debugging and error handling

During the ETL process, from operational DB through MRR and to DW we had validation errors, bugs and missing data in the destination database.

We dealt with it mainly in 3 ways:

1. Developing by small steps – hard to debug and to develop end 2-end process in one shot, so we took it in small steps. Each time adding 1 table handling few derived columns and then QA. Once QA done and all fixes applied, we moved to the next table/column.
2. SSIS debug – we used the debug tool in the SSIS, defining exception steps. Also sometimes, we turned off the SSIS validation so we could query by SQL the DB; it helps understand the result.
3. QA – Done mainly by SQL to verify derived column result is as expected, but also by compare the number of rows in the source versus the destination. The expectation is to have the same number of rows and no data was lost in the way.