ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

MODERN DATA MANAGEMENT & BUSINESS INTELLIGENCE

ASSIGNMENT 2

By Giotopoulou Panagiota: p2822005 & Vlassi Georgia: p2822001
Professor: Chantziantoniou Damianos

MSc in
Business Analytics

MSc in Business Analytics | Part Time 2020

Department of Management Science & Technology

Athens, Greece
December 14th 2020

CONTENTS

INTRODUCTION

SCOPE

The scope of this report is to provide a detailed description of the data transformation techniques that we used in order to import, clean, process and visualize our data.

The tools that we used are the following:

- Microsoft SQL Server Management Studio 18
- Visual Studio 17
- Power BI
- Microsoft Excel
- VI Editor

We chose to emphasize on the size of the dataset in order to focus on the ETL procedures and not in the idea - as a business case - as it was difficult enough to find online a dataset that had the required size and an interesting content at the same time. The business case that we chose is 'Google Playstore Apps' and in the following chapters, there is going to be a brief overview of the way we fulfilled the requirements of the assignment.

BUSINESS GOALS

The main idea of this assignment is to analyze a data collection that includes all the apps of Google Playstore and provide a visualization of the produced results. This information can be used among the business stakeholders – probably the marketing department of a company - in order to evaluate and decide in which app they should place their products' advertisements. The dataset includes all the apps along with the category they belong to, the rating, the released date, the number of installs per app, the price, size and many other useful information that is going to be analyzed in the next steps.

The results that we are going to provide are the following:

- Most popular apps that support ad hosting based on category
- Top rated apps in terms of rating and rating count along with their price grouped by user categories

DATA SOURCES

The dataset we chose is the 'Google Play Store Apps' and it can be found at the following link: https://www.kaggle.com/gauthamp10/google-playstore-apps.

It contains one CSV file of 603048 rows and 23 columns:

- App Name
- App Id
- Category
- Rating
- Rating Count
- Installs
- Minimum Installs
- Maximum Installs
- Free
- Price
- Currency
- Size
- Minimum Android
- Developer Id
- Developer Website
- Developer Email
- Released
- Last Updated
- Content Rating
- Ad Supported
- In App Purchases
- Editor's Choice

A short description of the data can be found below:

App name refers to the name of each application and it cannot be of null value as it is mandatory for our results.

App id is the App indicator

Category refers to the category (e.g. Education) to which the App belongs

Rating is the average rating of the application on a range of 0 to 5 and the **Rating Count** is the number of ratings that the application has.

Installs & **Minimum Installs** are fields of a same value that refer to the minimum installs that an app must have in order to be considered as an app that can host advertisements.

Maximum Installs are the total installs of the App.

Free is an indicator that shows whether an app is free of charge or not.

Minimum Android refers to the minimum Android version that is required in users' devices

Released & **Last Updated** are fields that indicate the release date and the date of the last update of the app respectively

In App Purchases is an indicator that shows whether an app includes billing transactions

Ad Supported likewise indicates whether an app can support ad hosting

Price is a numeric value that indicates the price of the App, if there is one

Currency is the suggested Currency (e.g. USD) provided by the App Developer

Developer Id & **Developer Website** & **Developer Email** are the values and indicators that refer to the developer that created the App

Content Rating indicates the age limit of the App users

Editor's choice is an indicator that shows whether the App is promoted

DATA CLEANING, IMPORT & TRANSFORMATION

DATA CLEANING

We performed the data cleaning in two phases.

- **PHASE A'**

During the first phase, we opened the CSV file with **Microsoft Excel**, sorted all the results in ascending order based on the App Name which we consider as mandatory. Then, we deleted all the rows that did not include an App Name and produced a Staging.csv file which consists of 528111 rows.

- **PHASE B'**

For the second part of the data cleaning procedure, we used **Visual Studio**.
By checking the Staging.csv file that was produced during the previous step we noticed that it included many Unicode characters which we needed to replace. For this reason, we collected all these characters and stored it in a csv file in order to replace them with ' '.
We tried to automate the 'query generation procedure' with **Ubuntu** add on for Windows 10 and more specifically with VI Editor and visual block and replace actions.

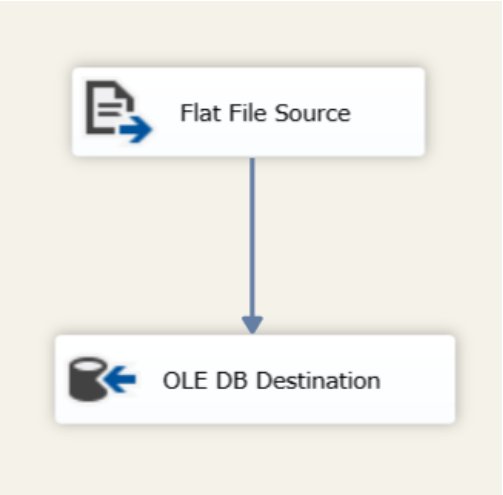
pegiot@DESKTOP-G1RRFT9: /mnt/c/Users/Penny/OneDrive/Desktop

```
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ƒ', '')  where [App Name] like '%ƒ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '"', '')  where [App Name] like '"%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ƒ', '')  where [App Name] like '%ƒ';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ƒ', '')  where [App Name] like 'ƒ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '-ƒ', '')  where [App Name] like '%-ƒ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ƒ', '')  where [App Name] like '%ƒ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '-', '')  where [App Name] like '-%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '-', '')  where [App Name] like '%-';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'Œ', '')  where [App Name] like 'Œ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'Œ', '')  where [App Name] like '%Œ';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'Œ', '')  where [App Name] like '%Œ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ä', '')  where [App Name] like'%ä%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'Ž', '')  where [App Name] like 'Ž%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'Ž', '')  where [App Name] like '%Ž';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'Ž', '')  where [App Name] like '%Ž%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'œ', '')  where [App Name] like '%œ%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ë', '')  where [App Name] like '%ë%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ì', '')  where [App Name] like '%ì%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '½', '')  where [App Name] like '%½%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '©', '')  where [App Name] like '%©%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '™', '')  where [App Name] like '%™%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '¡', '')  where [App Name] like '%¡%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'ž', '')  where [App Name] like '%ž%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '×', '')  where [App Name] like '%×%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '¢', '')  where [App Name] like '%¢%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '‰', '')  where [App Name] like '%‰%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], 'á', '')  where [App Name] like '%á%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '‹', '')  where [App Name] like '%‹%';
update Google_Apps.dbo.google_apps set [App Name] = REPLACE ([App Name], '·', '')  where [App Name] like '%·%';
```

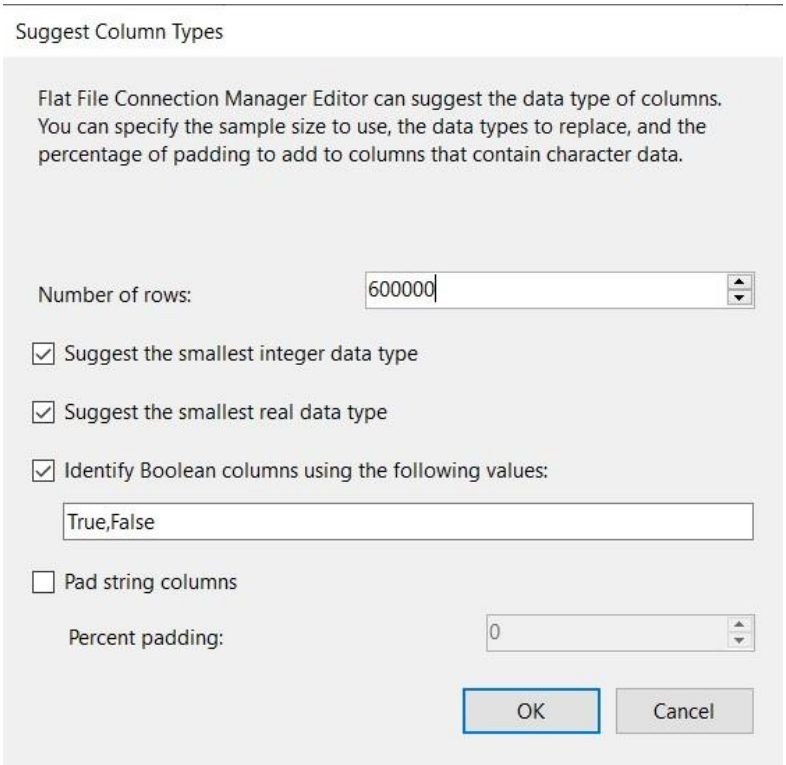
We stored these queries in .txt files and then we proceeded with **Visual Studio** where we performed the Data Cleaning & Import procedure.

DATA IMPORT

For the Data Import procedure, we created two agents; one for the file import and one for the data commission in MSSQL



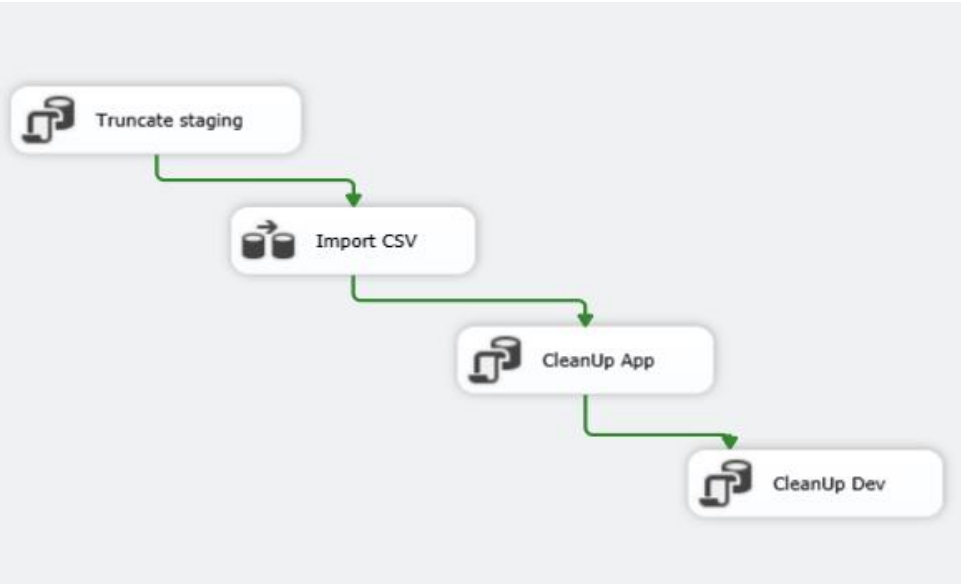
The Flat File that we used is the Staging.csv that was created at the first Step.
We used the number 600000 in the suggested Column Type Section for our 528111 row-number file.



The data import was successful as shown below:

[SSIS.Pipeline] Information: "OLE DB Destination" wrote 528111 rows.

After the first Import, we created another agent that truncates the staging table, in case there is one, and two other ones; The Cleanup App, which contains all the queries that are related to the data cleaning of App Name column and the Cleanup Dev that contains all the queries that update the Developer ID column. After the overall cleanup procedure, the table consists of 526480 rows.



DATA TRANSFORMATION

The next step of our project was to create the Fact Table.

Firstly, we defined which of the data would be the dimensions and which would be the metrics.

DIMENSIONS	METRICS
App Name	Rating
App ID	Rating Count
Category	Installs
Free	Minimum Installs
Currency	Maximum Installs
Size Unit **	Size
Minimum Android	Price
Developer Id	
Developer Email	
Developer Website	
Released	
Last Updated	
Content Rating	
Ad Supported	
In App Purchases	
Editor's Choice	

** We decided to split the Size column into two; the one would be a dimension that contains only the Unit (e.g. k, M) and the other one would be a metric that includes the size as a number.

All of the dimension tables were created in MSSQL and consisted of two columns

- ID, which is a primary key
- Label, which will be used as index for the fact table

The specifications used for the table creations can be found below:

The ID is a unique auto-increment value that starts from the value 1

Column Name	Data Type	Allow Nulls
id_lastupdated	int	☐
label_lastupdated	datetime2(7)	☑
		☐

Column Properties

Default Value or Binding	
Table Designer	
Collation	<database default>
Computed Column Specification	
Condensed Data Type	int
Description	
Deterministic	Yes
DTS-published	No
Full-text Specification	No
Has Non-SQL Server Subscriber	No
Identity Specification	Yes
(Is Identity)	Yes
Identity Increment	1
Identity Seed	1
Indexable	Yes
Is Columnset	No
Is Sparse	No
Merge-published	No
Not For Replication	No
Replicated	No
RowGuid	No
Size	4

Indexes/Keys

Selected Primary/Unique Key or Index:

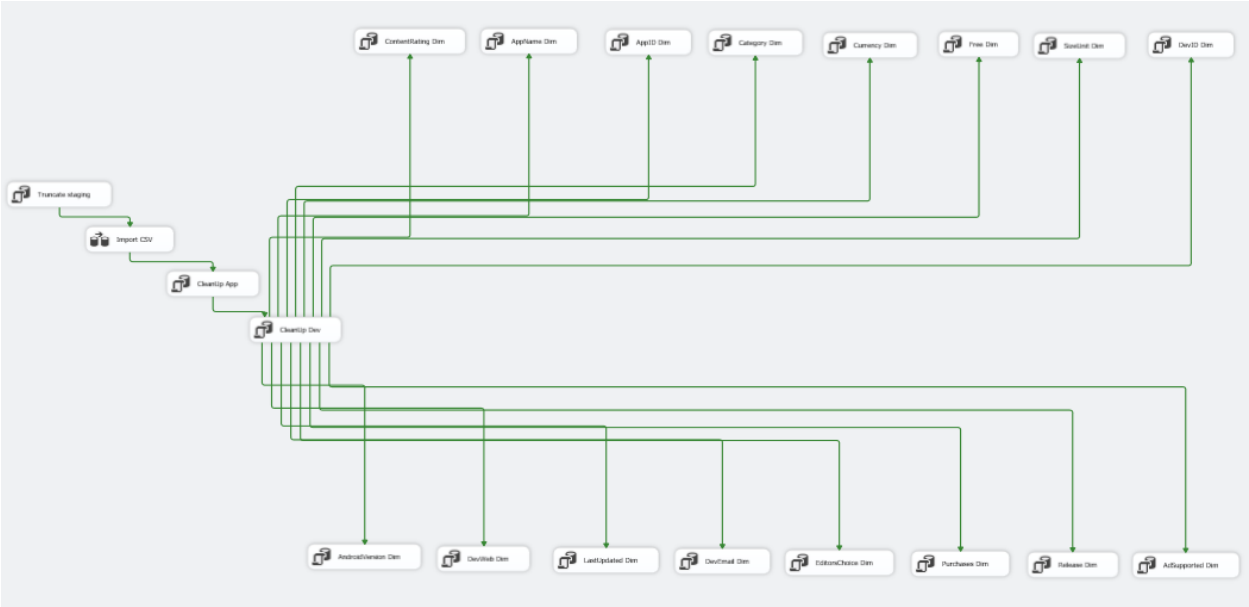
IX_adsupported_dim*
PK_adsupported_dim

Editing properties for new unique key or index.

(General)	
Columns	label_adsupported (ASC)
Is Unique	Yes
Type	Index
Identity	
(Name)	IX_adsupported_dim
Description	
Table Designer	
Create As Clustered	No
Data Space Specification	PRIMARY
Fill Specification	
Ignore Duplicate Keys	Yes

Add Delete Close

The dimension tables were filled with values via procedures defined in Visual Studio as shown below:



In order to fill the labels of each dimension we select the distinct values from the Staging table:

```
INSERT INTO lastupdated_dim (label_lastupdated) SELECT DISTINCT [Last Updated] FROM
[GoogleStore].[dbo].[staging];
```

All the Dimension tables have the following format

	id_category	label_category
1	42	Action
2	11	Adventure
3	35	Arcade
4	15	Art & Design
5	8	Auto & Vehicles
6	48	Beauty
7	9	Board
8	21	Books & Reference
9	10	Business
10	28	Card
11	26	Casino
12	38	Casual
13	44	Comics
14	1	Communication
15	36	Dating
16	37	Education
17	24	Educational
18	22	Entertainment
19	29	Events
20	46	Finance
21	45	Food & Drink
22	5	Health & Fitness
23	34	House & Home
24	47	Libraries & Demo
25	19	Lifestyle
26	4	Maps & Navigation
27	39	Medical
28	7	Music
29	27	Music & Audio
30	31	News & Magazines

FACT TABLE CREATION

FACT TABLE

For the Fact Table Creation, we used the following queries

#CREATE TABLE STATEMENTS

```
USE [GoogleStore]
GO
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE TABLE [dbo].[googlestore_ft](
[App Name] [int] NOT NULL,
[App Id] [int] NOT NULL,
[Category] [int] NOT NULL,
[Rating] [real] NOT NULL,
[Rating Count] [bigint] NULL,
[Installs] [decimal](28, 0) NULL,
[Minimum Installs] [bigint] NULL,
[Maximum Installs] [bigint] NULL,
[Free] [int] NOT NULL,
[Price] [real] NULL,
[Currency] [int] NOT NULL,
[Size] [varchar](18) NULL,
[Size Unit] [int] NOT NULL,
[Minimum Android] [int] NOT NULL,
[Developer Id] [int] NOT NULL,
[Developer Website] [int] NOT NULL,
[Developer Email] [int] NOT NULL,
[Released] [int] NOT NULL,
[Last Updated] [int] NOT NULL,
[Content Rating] [int] NOT NULL,
[Ad Supported] [int] NOT NULL,
[In App Purchases] [int] NOT NULL,
[Editors Choice] [int] NOT NULL
) ON [PRIMARY]
GO
```

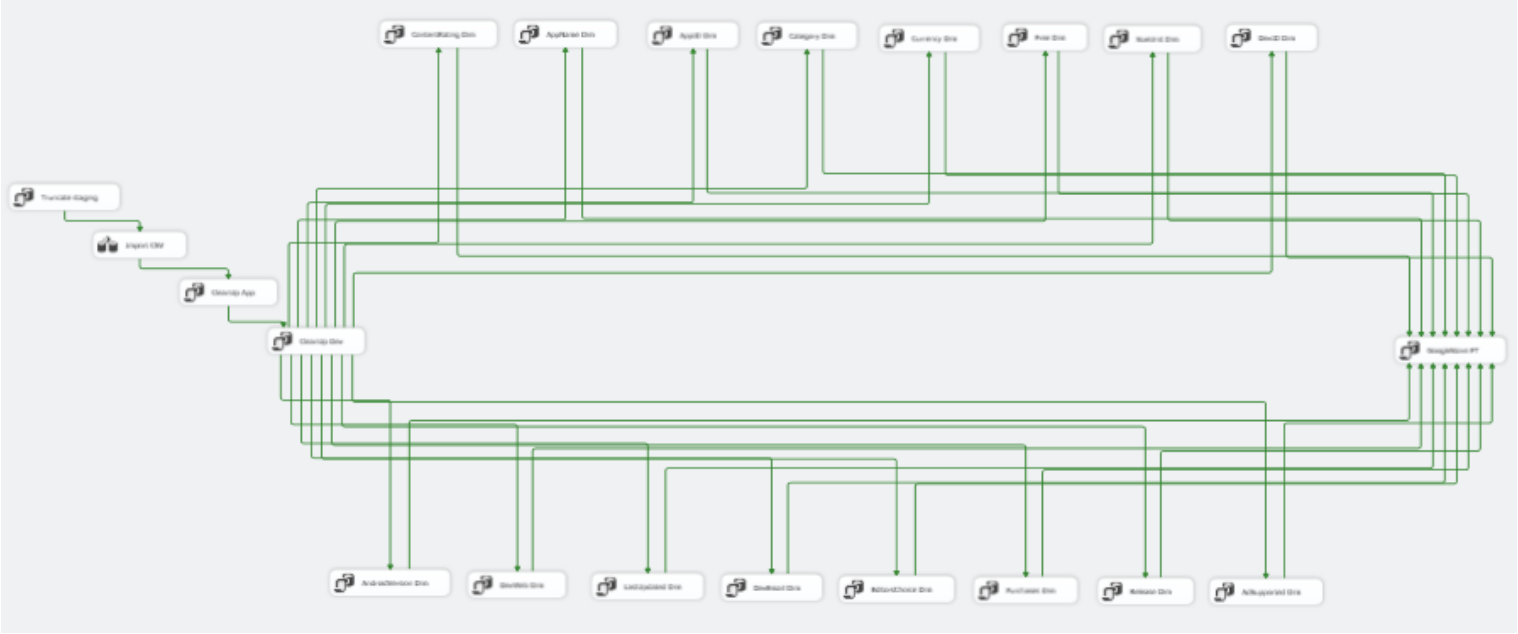
The fact table was filled with values from all dimensions via ETL procedures defined in Visual Studio. The query that was used to fill the table was the following:

#INSERT TABLE STATEMENTS

```
USE [GoogleStore];

INSERT INTO googlestore_ft
SELECT [appname_dim].[id_appname] AS [App Name],
[appid_dim].[id_appid] AS [App Id],
[category_dim].[id_category] AS [Category],
[staging].[Rating],
[staging].[Rating Count],
[staging].[Installs],
[staging].[Minimum Installs],
[staging].[Maximum Installs],
[free_dim].[id_free] AS [Free],
[staging].[Price],
[currency_dim].[id_currency] AS [Currency],
[staging].[Size],
[sizeunit_dim].[id_sizeunit] AS [Size Unit],
[androidversion_dim].[id_androidversion] AS [Minimum Android],
[devid_dim].[id_devid] AS [Developer Id],
[devweb_dim].[id_devweb] AS [Developer Website],
[devemail_dim].[id_devemail] AS [Developer Email],
[release_dim].[id_released] AS [Released],
[lastupdated_dim].[id_lastupdated] AS [Last Updated],
[contentrating_dim].[id_contentrating] AS [Content Rating],
[adsupported_dim].id_adsupported AS [Ad Supported],
[purchases_dim].[id_purchases] AS [In App Purchases],
[editorschoice_dim].[id_editorschoice] AS [Editors Choice]
FROM staging
INNER JOIN [appname_dim] ON [staging].[App Name] = [appname_dim].[label_appname]
INNER JOIN [appid_dim] ON [staging].[App Id] = [appid_dim].[label_appid]
INNER JOIN [category_dim] ON [staging].[Category] = [category_dim].[label_category]
INNER JOIN [free_dim] ON [staging].[Free] = [free_dim].[label_free]
INNER JOIN [currency_dim] ON [staging].[Currency] = [currency_dim].[label_currency]
INNER JOIN [sizeunit_dim] ON [staging].[Size Unit] = [sizeunit_dim].[label_sizeunit]
INNER JOIN [androidversion_dim] ON [staging].[Minimum Android] =
[androidversion_dim].[label_androidversion]
INNER JOIN [devid_dim] ON [staging].[Developer Id] = [devid_dim].[label_devid]
INNER JOIN [devweb_dim] ON [staging].[Developer Website] = [devweb_dim].[label_devweb]
INNER JOIN [devemail_dim] ON [staging].[Developer Email] = [devemail_dim].[label_devemail]
INNER JOIN [release_dim] ON [staging].[Released] = [release_dim].[label_released]
INNER JOIN [lastupdated_dim] ON [staging].[Last Updated] =
[lastupdated_dim].[label_lastupdated]
INNER JOIN [contentrating_dim] ON [staging].[Content Rating] =
[contentrating_dim].[label_contentrating]
INNER JOIN [adsupported_dim] ON [staging].[Ad Supported]
=[adsupported_dim].[label_adsupported]
INNER JOIN [purchases_dim] ON [staging].[In App Purchases] =
[purchases_dim].[label_purchases]
INNER JOIN [editorschoice_dim] ON [staging].[Editors Choice] =
[editorschoice_dim].[label_editorschoice];
```

The procedure was automated in GoogleStore Dim which included the insert statement mentioned above

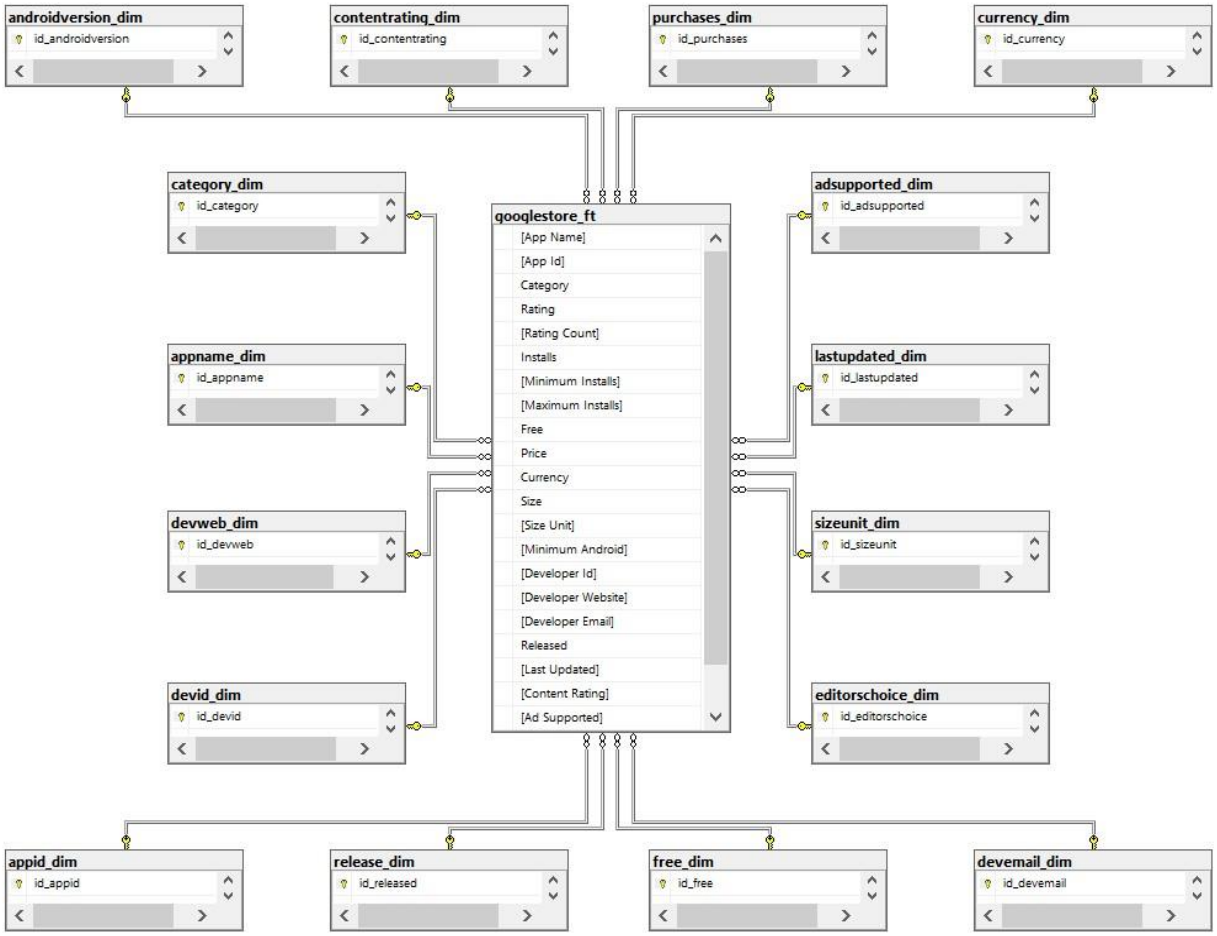


The Fact Table was successfully created and had the following format

	App Name	App Id	Category	Rating	Rating Count	Installs	Minimum Installs	Maximum Installs	Free	Price	Currency	Size	Size Unit	Minimum Android	Developer Id	Developer Website	Developer Email	Released	Last Updated	Content Rating	Ad Supported
1	519194	195905	46	4.8	266	1000	1000	4156	2	0	19	18	1	79	81869	91604	136102	730	211122	5	1
2	401817	259403	17	0	0	500	500	537	2	0	19		3	1	12831	100176	17912	415	149854	5	1
3	382245	10757	21	5	28	1000	1000	2134	2	0	19	8.6	1	51	7998	100710	81526	2354	96345	5	1
4	283275	398818	22	3.2	64	10000	10000	28848	2	0	19	92	1	95	82357	92432	138174	425	395957	5	1
5	106935	402416	13	0	0	10	10	40	2	0	19	3.6	1	94	23045	35174	136508	436	221891	5	1
6	308018	499603	5	0	0	100	100	181	1	1.99	19	51	1	88	6396	38729	66688	2114	135419	5	1
7	282850	406315	37	4.6	13	50	50	50	1	2.49	19	17	1	103	9030	31040	83003	433	379751	5	1
8	204348	219742	12	4	33	1000	1000	1453	2	0	19	27	1	40	60957	37549	6846	2449	79848	5	1
9	290825	24749	37	0	0	10	10	43	2	0	19	2.3	1	116	53835	125873	36273	846	75445	5	1
10	220066	29942	18	3.6	14	5000	5000	6286	2	0	19	245	2	109	40	38729	117264	2797	102973	5	1
11	305498	448871	37	4.7	38	1000	1000	2140	2	0	19	2.4	1	88	68271	102077	110888	1974	403953	5	1
12	210962	456403	13	4.2	2684	100000	100000	146765	2	0	19	43	1	94	46480	58184	11294	2404	200778	5	1
13	301525	497915	13	3.5	8	1000	1000	1425	2	0	19	8.8	1	40	27463	131550	20432	3562	284053	5	1
14	395105	29941	18	0	0	100	100	119	1	0.99	19	244	2	109	40	38729	117264	2371	122553	5	1
15	354202	315478	46	0	0	500	500	606	2	0	19	15	1	88	81143	87080	2627	2344	37643	5	1
16	56212	392458	1	0	0	10	10	25	2	0	19	9.1	1	103	9066	18211	112844	2344	141258	5	1
17	168975	319891	18	4.4	69	10000	10000	33858	2	0	19	6.1	1	116	3957	66133	71866	2391	83070	5	1
18	38243	115146	35	0	0	10	10	43	1	0.99	19	17	1	51	47132	38729	70567	503	279060	5	1
19	45370	311307	20	3.6	83	10000	10000	41978	2	0	19	6.5	1	41	28884	112528	11375	2075	113681	5	1
20	5001	454589	48	3.7	105	10000	10000	14693	2	0	19	71	1	116	16162	101793	138837	1785	357130	5	1
21	395573	284005	30	4.3	4281	1000...	1000000	1492970	2	0	19	58	1	108	61067	69160	100323	1625	208380	5	1
22	109196	391920	46	4	21	1000	1000	4504	2	0	19	37	1	103	71236	120413	7605	2033	149852	5	1
23	242811	403942	37	0	0	50	50	71	2	0	19	9.2	1	88	37252	38729	149219	755	340543	5	1
24	503770	176633	18	4.2	91	10000	10000	12088	2	0	19	31	1	116	77117	25146	49376	1566	157737	5	1
25	335939	137425	48	0	0	100	100	237	2	0	19	90	2	9	54361	14309	23587	3484	259384	5	1
26	460948	168613	10	3.9	206	1000	1000	4496	2	0	19	591	2	80	80878	38729	139243	879	99674	5	1
27	433907	183394	12	4.7	164	1000	1000	3100	1	6.49	19		3	1	33384	132504	37657	3160	232060	5	1

STAR

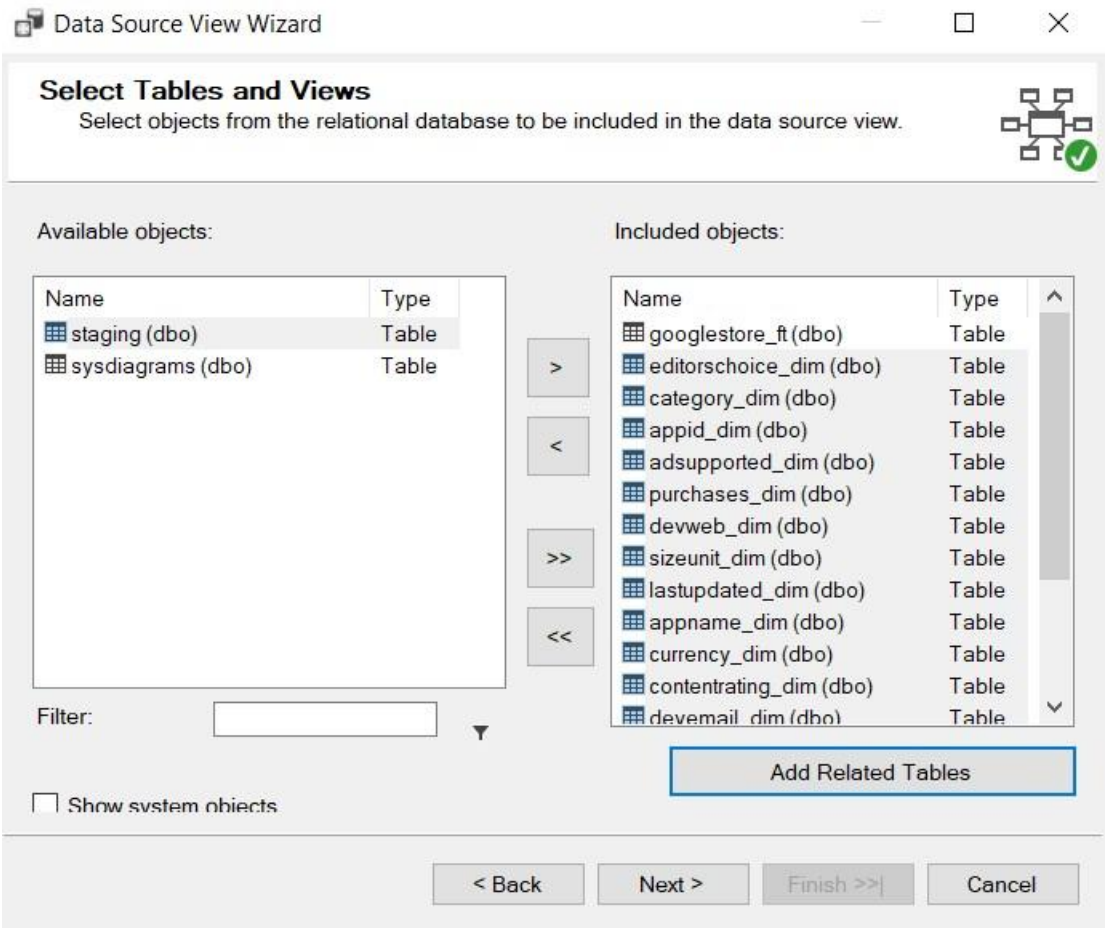
The star that was created is shown below



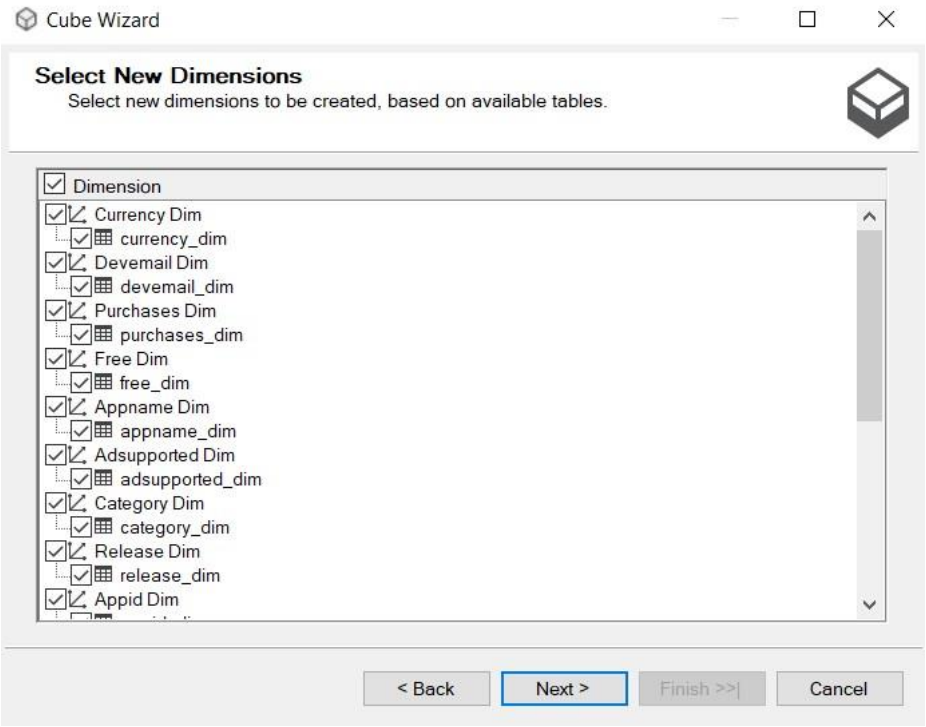
CUBE

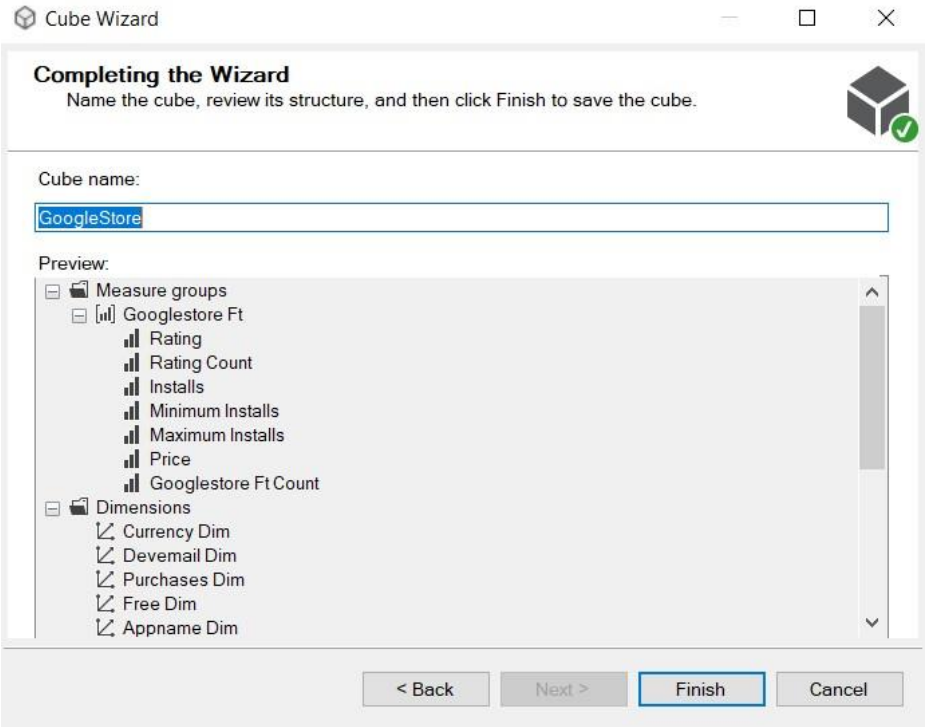
In order to create the cube, we created a new data source (GoogleStore.ds) and then followed the below steps:

- We selected the dimensions that are included in staging in order to create datasource views (GoogleStore.dsv)

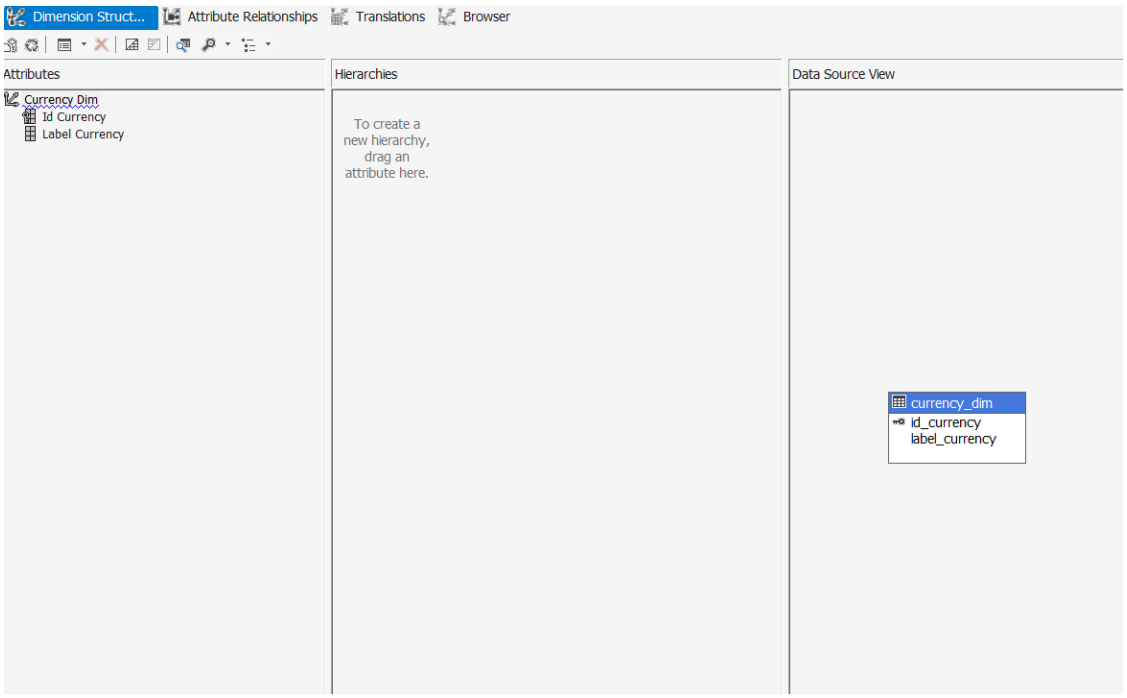


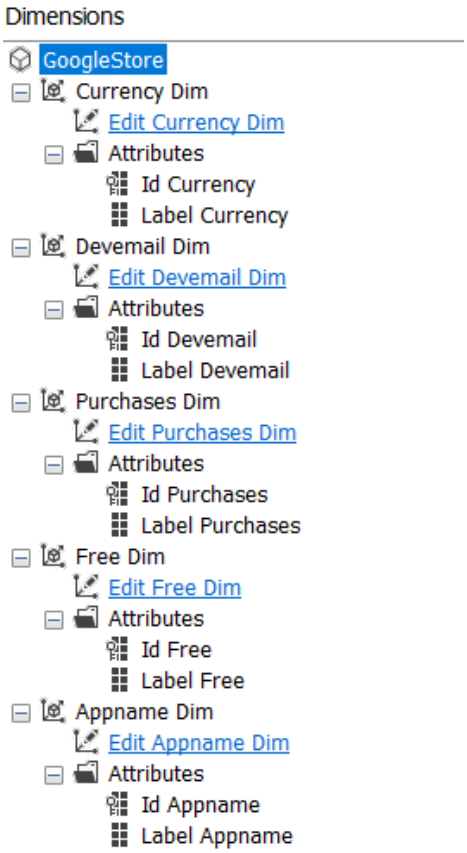
- We then created a new cube, including metrics and GoogleStore FT Count





- We added the Labels in each dimension as shown below:





- After the initial process, we added some calculations and a new measure regarding metrics

Name:
[Price Average]

Parent Properties
Parent hierarchy: Measures
Parent member: Change

Expression
```
[Measures].[Price]/[Measures].[Googlestore Ft Count]
```

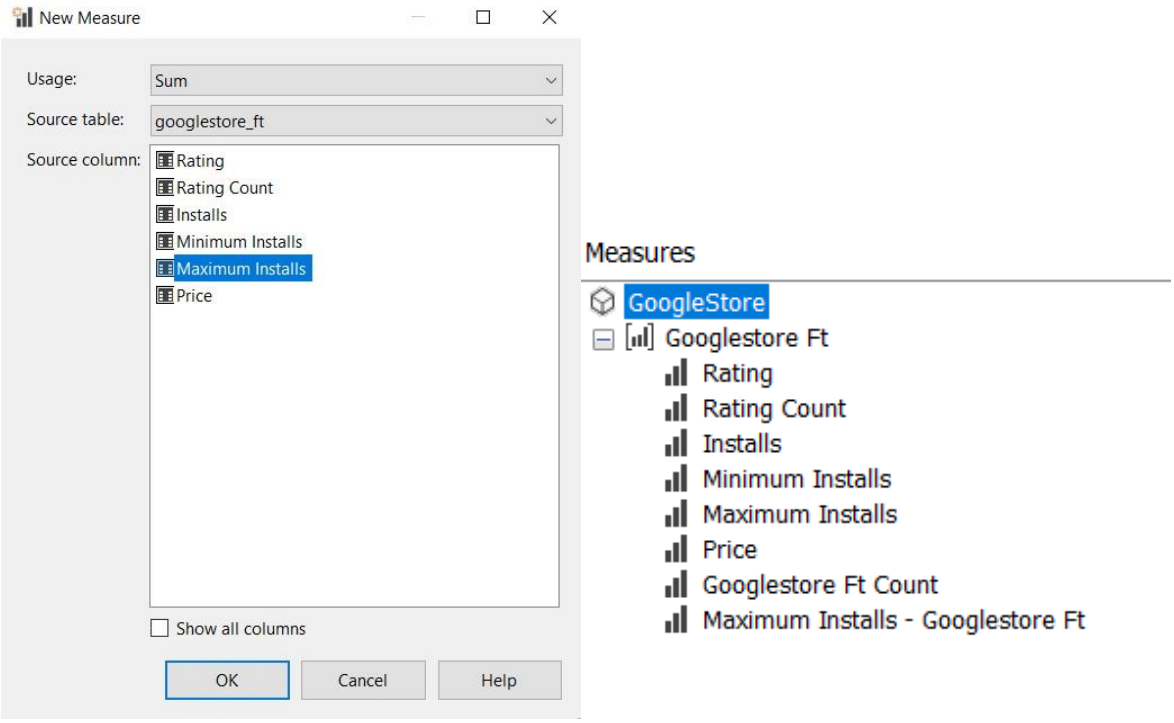
Additional Properties
Format string: "Currency"
Visible: True
Non-empty behavior:
Associated measure group: Googlestore Ft
Display folder:
Color Expressions
Font Expressions

Name:
[RatingCount Average]

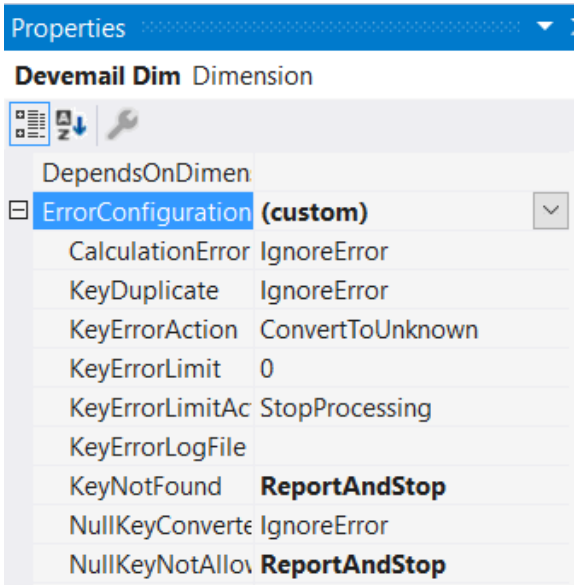
Parent Properties
Parent hierarchy: Measures
Parent member: Change

Expression
```
[Measures].[Rating Count]/[Measures].[Googlestore Ft Count]
```

Additional Properties
Format string: "Standard"
Visible: True
Non-empty behavior:
Associated measure group: Googlestore Ft
Display folder:
Color Expressions
Font Expressions



- We performed some changes in order to ignore an error that affected a non-mandatory – for our project-column during the process cube procedure

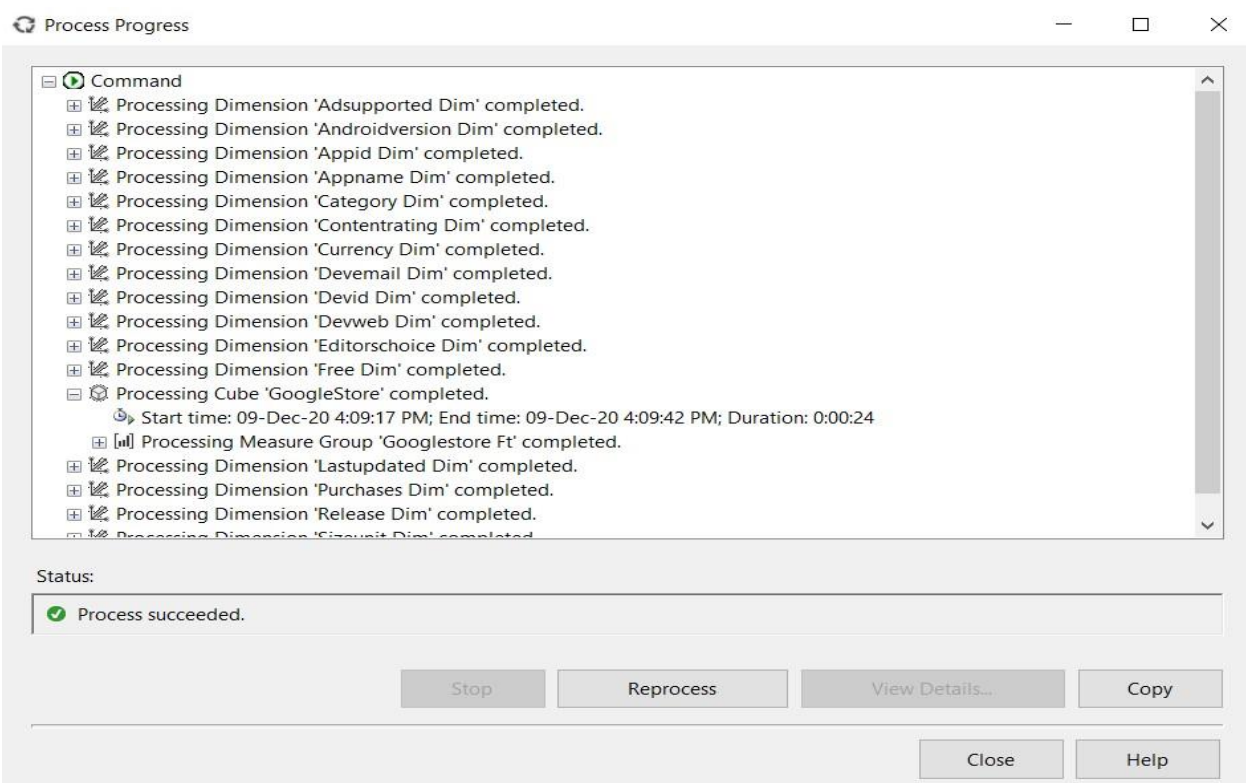


- Methodologies as Drill Down and Pivot could not be applied to our data analysis, as the produced columns had no dependencies. However, a first attempt of slicing is shown below. More filters and group by processes are performed during our Power BI visualizations:

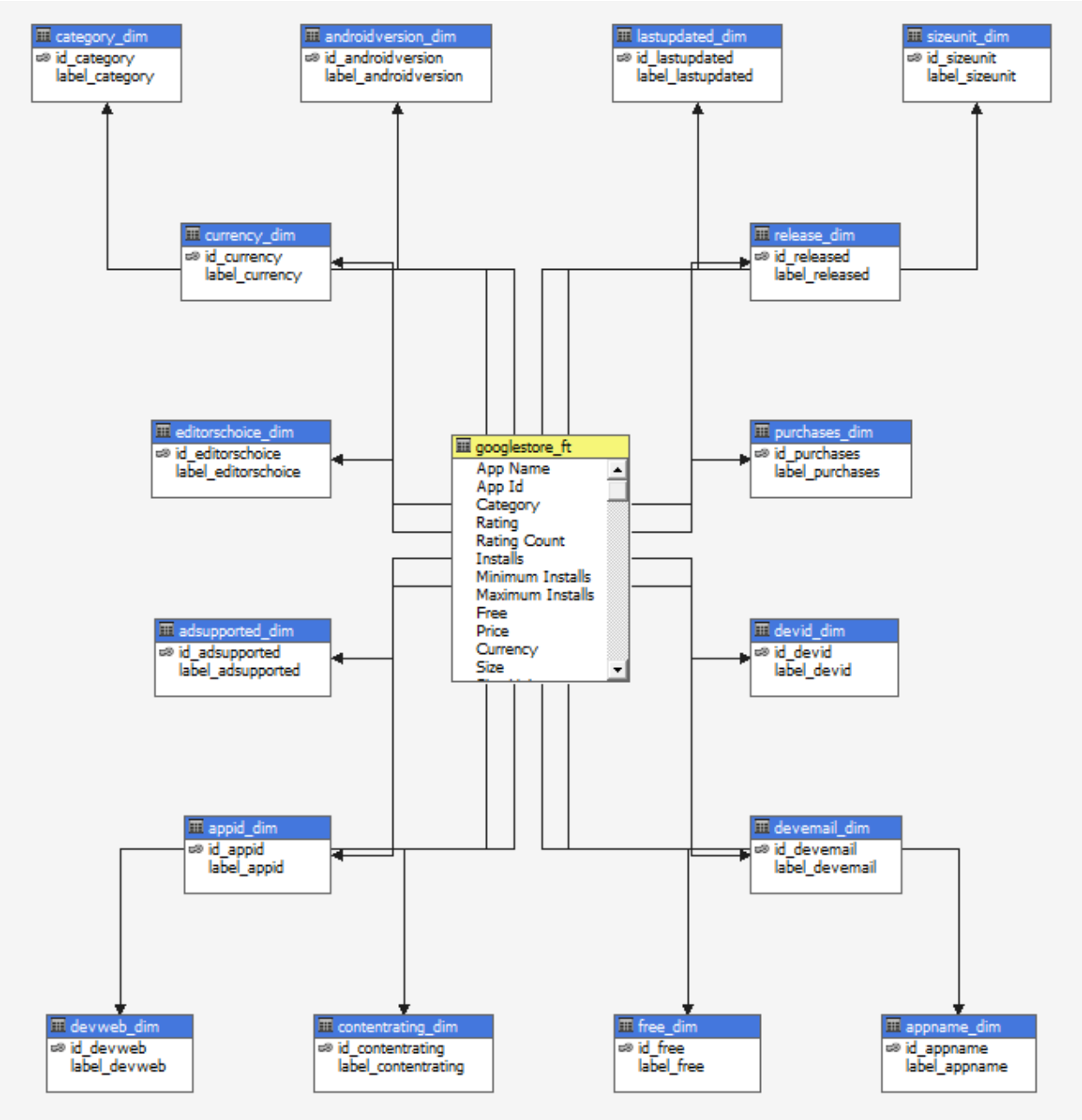
```
SELECT
NON EMPTY { [Measures].[Maximum Installs] } ON COLUMNS,
NON EMPTY { ([Appname Dim].[Label Appname].[Label Appname].ALLMEMBERS * [Adsupported Dim].[Label Adsupported].[Label Adsupported].ALLMEMBERS ) } DIMENSION
PROPERTIES MEMBER_CAPTION,
MEMBER_UNIQUE_NAME ON ROWS
FROM [GoogleStore] CELL PROPERTIES VALUE, BACK_COLOR, FORE_COLOR, FORMATTED_VALUE, FORMAT_STRING, FONT_NAME, FONT_SIZE, FONT_FLAGS
```

Label Appname	Label Adsupported	Maximum Installs
CD Carpe Die...	0	2025
CD El Nacional...	0	7819
CD ELGOIBAR	1	611
CD OLIVAR D...	0	130
CD Passos	0	466
CD SFA - New ...	0	15504
CD Upload Tool	0	161
CD UTIEL	1	628
CD Zing	0	3643
CDA	0	220
CDA Casino	0	4559
CDA Dubai	0	6552
CDA ICMS	0	29
CDA Masar	0	64
CDA Metepec	0	195
CDA Sanad Re...	0	63
CDA Tracking ...	0	1122
CDAC	0	1609
CDAC Class (c...	1	16

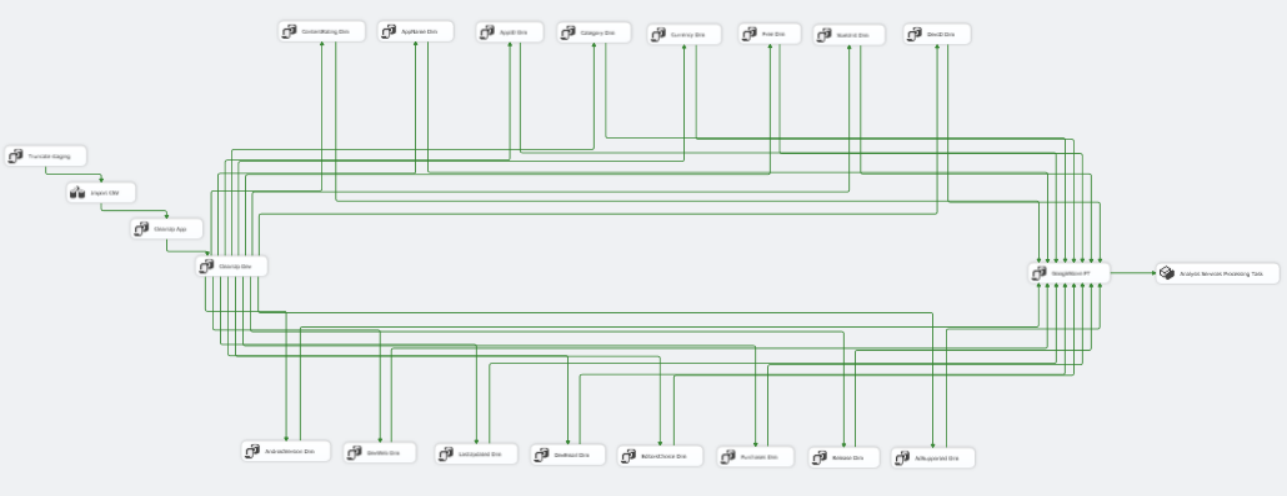
- Finally, we processed the cube



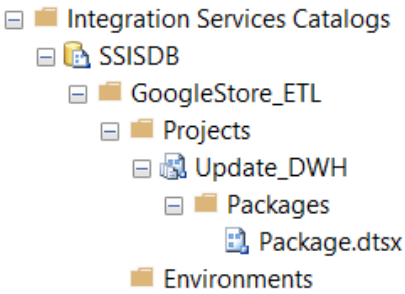
The final schema of cube:



The Analysis Services Processing Task (Cube) has been added to our ETL procedure as shown below:

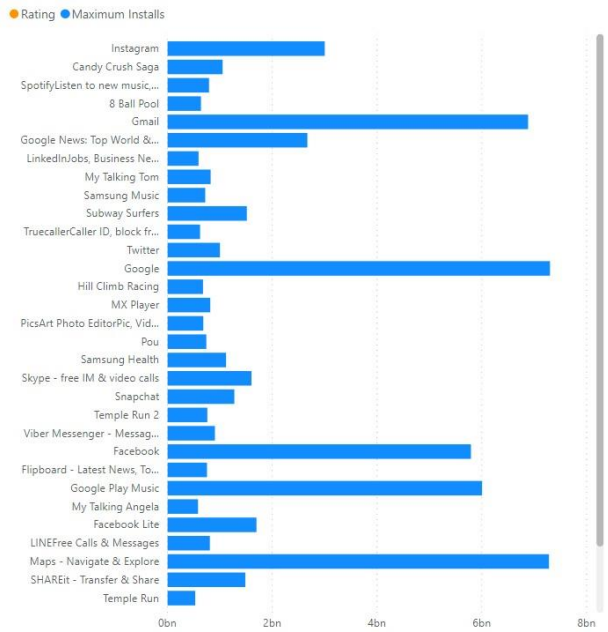


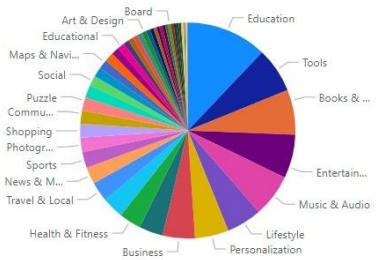
Finally, the ETL procedure has been deployed to SQL Server Management Studio. We could, also, add a scheduler to run the procedure once a day, to add new applications in our server and run the Cube calculations:

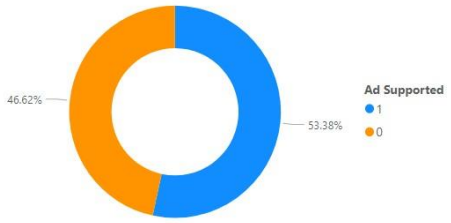


POWER BI VISUALIZATIONS

In order to achieve our business goal, which is to evaluate and decide in which app, stakeholders of the company, should place their products' advertisements, we drew some visualizations.



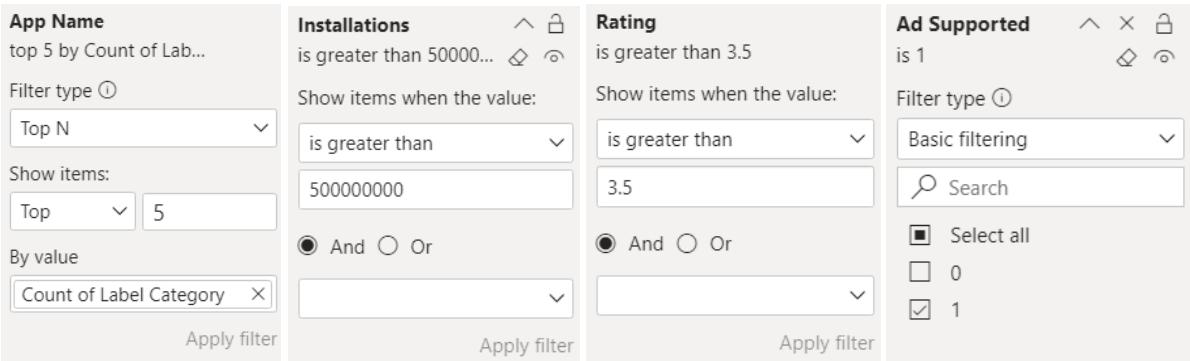




Initially, we create Pie chart, to present the ratio of Categories. We, also, draw a Donut chart to present the percentage of applications that support advertisements.

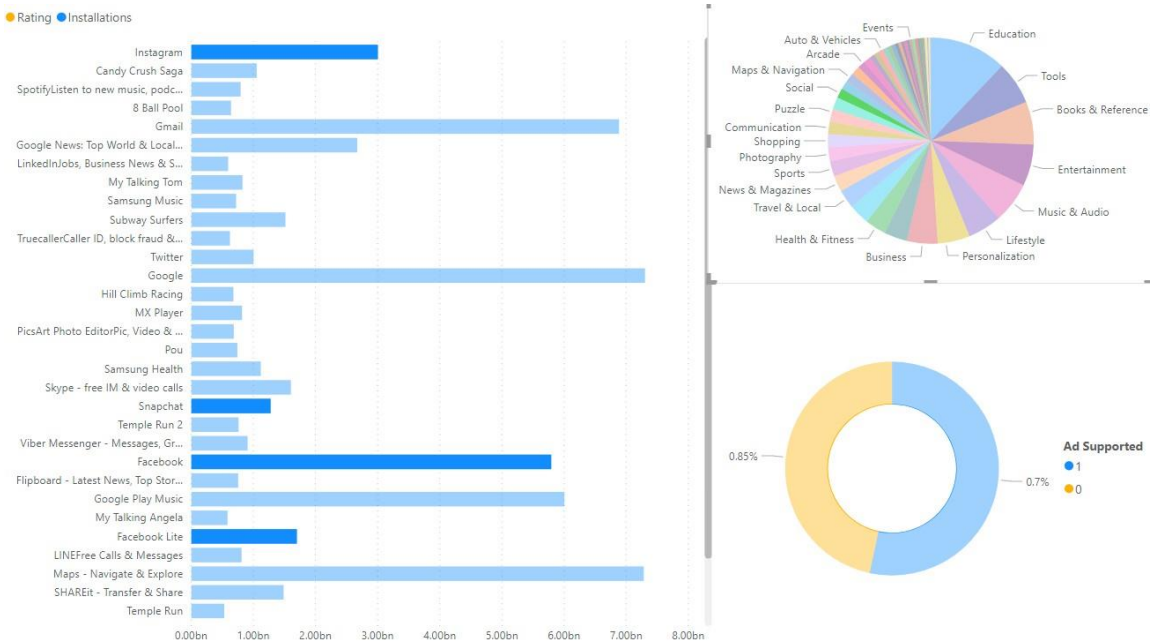
In the left, a Clustered bar chart is constructed, to present the top five applications of each category, that have Installations greater than 500.000.000 and Rating above 3.5.

We used the below to filter our dataset:



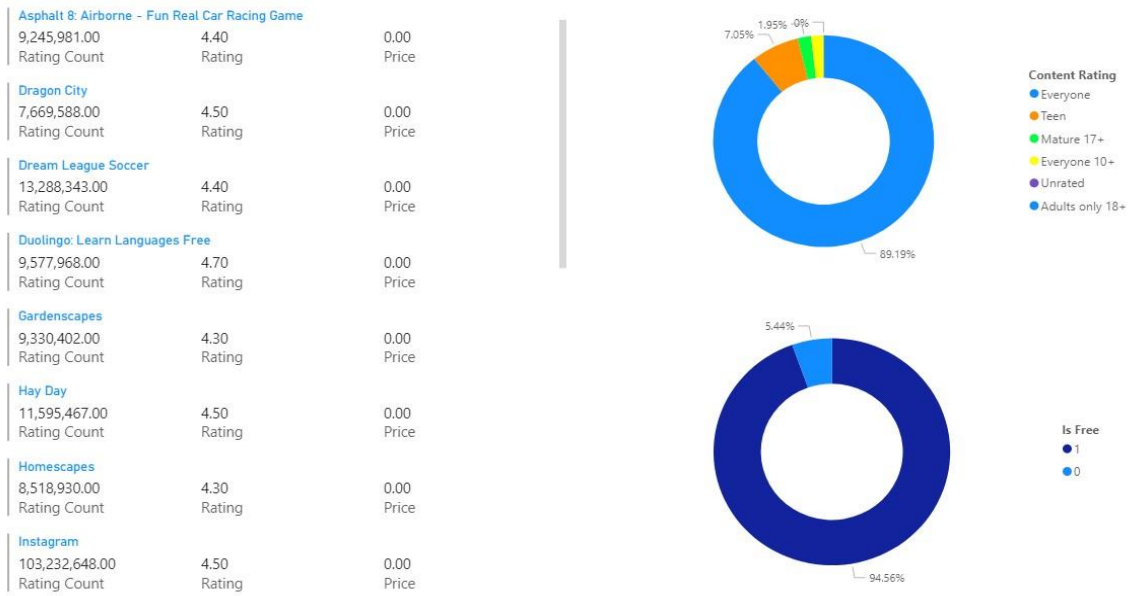
For example, if want to present the top social applications, we can select the 'social' slice from the Pie chart and blue part of the Donut chart, which indicates the applications that support advertisements.

The result would be the bold applications at Clustered bar chart, Instagram, Snapchat, Facebook and so on.

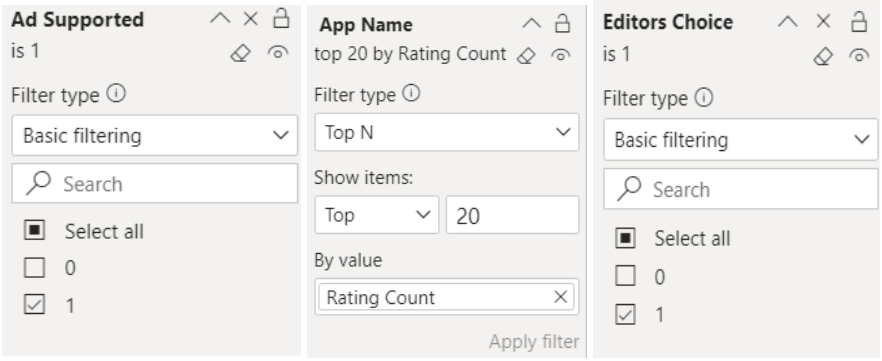


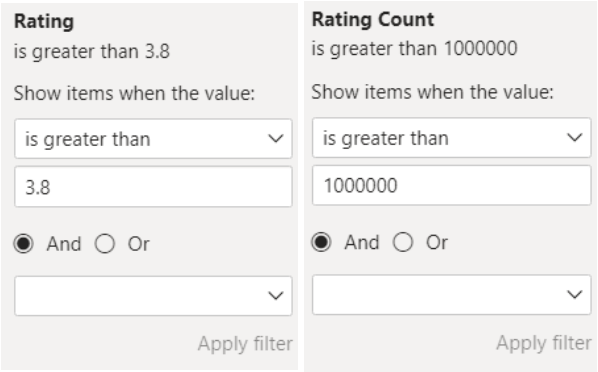
We created a Donut chart, to present the ratio of ratings regarding content. Another Donut chart is created to show the percentage of applications that are free.

In the left, a Multi-row card is constructed, to present the top five applications of each category, that have Installations greater than 500.000.000 and Rating above 3.5.



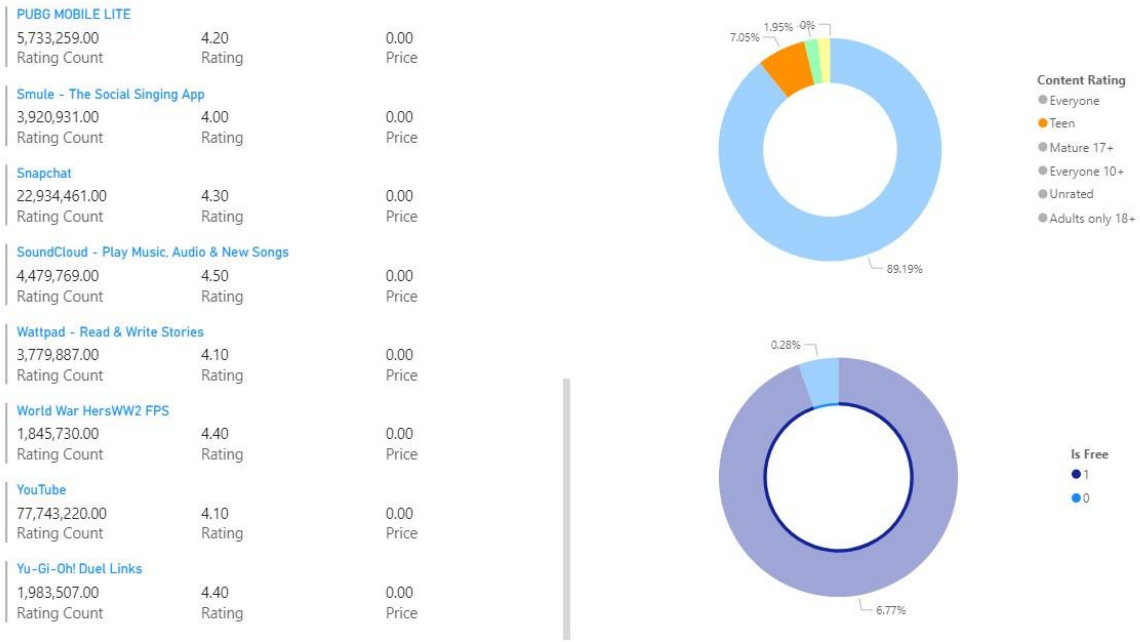
We used the below to filter our dataset:





For example, if we want to present the top Teen applications that are free, we can select the 'Teen' slice from the Pie chart and blue part of the Donut chart, which indicates the applications that are free.

The result would be the applications shown on the Multi-row card in the left, including the Rating count of these categories, their rating and their price.



CHALLENGES

In general, we did not face any significant challenges with regards to the assignment. We would say that our biggest difficulty was to pick the appropriate dataset in terms of size, complexity, content and potential business outcome. We focused on finding a dataset with hundreds of thousands of rows as described in assignments requirements.

However, a challenge that we faced is the data cleaning of the dataset. To be more specific, although we included a python script in our ETL process that replaced all the ASCI characters with null, many Unicode characters did not get deleted. We overcame this challenge quickly by generating queries that updated all this characters and symbols to Null.

Last, we faced some difficulties in installation and configuration of Microsoft SQL Server Management Studio and Visual Studio Data Tools, until found the appropriate and compatible versions.