DATA CLEANING OF GOOGLE PLAYSTORE DATASET USING SQL

DATASET: Google PlayStore

Source: https://www.kaggle.com/datasets/lava18/google-play-store-apps?select=googleplaystore_user_reviews.csv

Columns : 13

Rows: 284

1. Create a schema named googlestore in MySQL

```
create schema googlestore;
```

2. Right click table to import table named googleplaystore

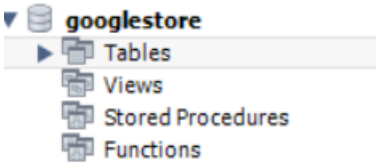
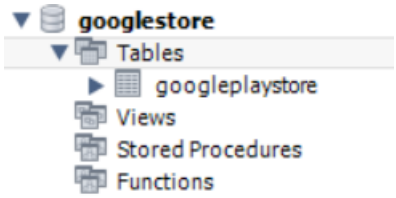


Table imported



Size column

Size
8.7M
25M
3.1M
21M
5.5M
17M
31M
4.2M
23M
25M
11M
4.2M
24M
11M
10M
24M
26M
Varies with device
5.7M
2.4M
3.4M
8.9M
2.9M
Varies with device
Varies with device
7.9M

1. Datatype of Size column mentioned as text,
   Size column containing ‘Varies with device’ to be modified to ‘NaN’

```
-- set the size column with 'varies with device' to NaN--
UPDATE googleplaystore
SET Size = 'NaN'
WHERE Size = 'Varies with device';
```

2. The “Size” column has values like “19M”, “201k”,etc. Replace ‘M’ with ‘000’.

```
-- replace ‘M’ with ‘000 in size column--
UPDATE googleplaystore
SET Size = replace(Size, 'M','000')
WHERE Size LIKE '%M';
```

3. There is an issue with the above step. E.g. 19M converted to 19000, 201K converted to 201 but 8.2M to 8.2000. This means the value for 8.2M is not converted to 8200.

```
-- If the value in the Size column is less than 10 then multiply that value by 1000.
UPDATE googleplaystore
SET Size = CASE
WHEN Size < 10 THEN Size*1000
ELSE Size
end;
```

4. Replace 'k' with an empty string ''.

```
--  replace 'k' with empty string--
UPDATE googleplaystore
SET Size = replace(Size, 'k','')
WHERE Size LIKE '%k';
```

5. Once it's done divide each value of the Size column by 1000 to convert it to Megabytes

```
-- to convert size column to megabytes--
UPDATE googleplaystore
SET Size = Size/1000;
```

6. Then convert the column to Float Datatype

```
-- datatype of size column changed from text to float--
ALTER TABLE googleplaystore
MODIFY COLUMN Size Float;
```

7. Indorder to clean ‘Install’ column replace “+”, “,” signs with an empty string

Installs
5,000,000+
50,000,000+
10,000+
100,000+
500,000+
10,000+
10,000,000+
500,000+
50,000+
500,000+
100,000+
10,000+
500,000+
10,000+
100,000+
10,000,000+
100,000+
1,000,000+
10,000+
100,000+
50,000+
100,000+
100,000+
1,000,000+
100,000+
100,000+

```
-- INSTALL Column--
UPDATE googleplaystore
SET Installs = replace(Installs, '+','')
WHERE Installs LIKE '%+';

-- INSTALL Column--
UPDATE googleplaystore
SET Installs = replace(Installs, ',','')
WHERE Installs LIKE '%,%';
```

Installs
50000
50000
1000000
1000000
10000

8. **Last updated column** is in text format. It needs to be changed to date format.

Last Updated
August 1, 2018
June 8, 2018
July 3, 2018
June 26, 2018
July 31, 2018
November 7, 2017
July 30, 2018
July 12, 2018
July 7, 2018
October 11, 2017
July 19, 2018
April 27, 2018
August 2, 2018
November 29, 2017
April 2, 2018
July 31, 2018
November 14, 2017
July 30, 2018

9. Add a new column **NewLastUpdated** in Datetime format

```
ALTER TABLE googleplaystore
ADD COLUMN NewLastUpdated DATETIME;
```

10. Update the '**NewLastUpdated**' column by converting the 'Last Updated' column from a text format to a DATETIME format using the STR_TO_DATE function. This can work if the format in the 'Last Updated' column matches the format string you provided, which is '%M %e, %Y'

```
UPDATE googleplaystore
SET NewLastUpdated = STR_TO_DATE(`Last Updated`, '%M %e, %Y');
```

11. Drop the **Last updated** column

```
ALTER TABLE googleplaystore
DROP COLUMN `Last Updated`;
```

12. Rename the '**NewLastUpdated**' column as '**LastUpdated**'

```
ALTER TABLE googleplaystore
CHANGE COLUMN NewLastUpdated LastUpdated DATETIME;
```

13. Display only date in '**LastUpdated**' column without timestamp

```
ALTER TABLE googleplaystore
CHANGE COLUMN LastUpdated LastUpdated DATE;
```

LastUpdated
2018-01-07
2018-01-15
2018-08-01
2018-06-08
2018-06-20