### App Trader

Your team has been hired by a new company called App Trader to help them explore and gain insights from apps that are made available through the Apple App Store and Android Play Store. App Trader is a broker that purchases the rights to apps from developers in order to market the apps and offer in-app purchase.

Unfortunately, the data for Apple App Store apps and Android Play Store Apps is located in separate tables with no referential integrity.

#### 1. Loading the data

a. Launch PgAdmin and create a new database called app\_trader.

b. Right-click on the app\_trader database and choose `Restore...`

c. Use the default values under the `Restore Options` tab.

d. In the `Filename` section, browse to the backup file `app\_store\_backup.backup` in the data folder of this repository.

e. Click `Restore` to load the database.

f. Verify that you have two tables:

- `app\_store\_apps` with 7197 rows

- `play\_store\_apps` with 10840 rows

#### 2. Assumptions

Based on research completed prior to launching App Trader as a company, you can assume the following:

a. App Trader will purchase apps for 10,000 times the price of the app. For apps that are priced from free up to $1.00, the purchase price is $10,000.

- For example, an app that costs $2.00 will be purchased for $20,000.

- The cost of an app is not affected by how many app stores it is on. A $1.00 app on the Apple app store will cost the same as a $1.00 app on both stores.

- If an app is on both stores, it's purchase price will be calculated based off of the highest app price between the two stores.

b. Apps earn $5000 per month, per app store it is on, from in-app advertising and in-app purchases, regardless of the price of the app.

- An app that costs $200,000 will make the same per month as an app that costs $1.00.

- An app that is on both app stores will make $10,000 per month.

c. App Trader will spend an average of $1000 per month to market an app regardless of the price of the app. If App Trader owns rights to the app in both stores, it can market the app for both stores for a single cost of $1000 per month.

- An app that costs $200,000 and an app that costs $1.00 will both cost $1000 a month for marketing, regardless of the number of stores it is in.

d. For every half point that an app gains in rating, its projected lifespan increases by one year. In other words, an app with a rating of 0 can be expected to be in use for 1 year, an app with a rating of 1.0 can be expected to last 3 years, and an app with a rating of 4.0 can be expected to last 9 years .

- App store ratings should be calculated by taking the average of the scores from both app stores and rounding to the nearest 0.5.

e. App Trader would prefer to work with apps that are available in both the App Store and the Play Store since they can market both for the same $1000 per month.

#### 3. Deliverables

a. Develop some general recommendations as to the price range, genre, content rating, or anything else for apps that the company should target.

b. Develop a Top 10 List of the apps that App Trader should buy.

Find high risk apps that they could buy

c. Submit a report based on your findings. All analysis work must be done using PostgreSQL, however you may export query results to create charts in Excel for your report.

Columns in both tables to look at:

1)name of app in both tables

2)highest review count + highest rating(round to nearest.5) (gives best average)

3)highest count of genre

4)content rating. Need to convert number rating in app\_store\_apps to match text rating in play\_store\_apps

5)Price range

6)only return apps that are free or cost 1$ as they will be the cheapest purchased app

7)Find profitability of each app. If app costs 10000$ to purchase and it is in both stores and earns 10000$ a month subtract 1000$ for marketing

8) how much app will cost and how much will make

8)incorporate this somehowe: -For every half point that an app gains in rating, its projected lifespan increases by one year. In other words, an app with a rating of 0 can be expected to be in use for 1 year, an app with a rating of 1.0 can be expected to last 3 years, and an app with a

rating of 4.0 can be expected to last 9 years.

Crazy answers

--5) create a table showing count of each category and percentage of overall genre

SELECT

a.primary\_genre

, COUNT(\*) AS genre\_count

, ROUND((COUNT(\*) \* 100.0) / SUM(COUNT(\*)) OVER (), 2) AS genre\_percentage

FROM

app\_store\_apps AS a

GROUP BY

a.primary\_genre

ORDER BY

genre\_count DESC

------------------------------------------------------------------------------

--6) combine the last two

SELECT

a.name

, a.primary\_genre

, ROUND(AVG((CAST(a.review\_count AS INTEGER) + CAST(p.review\_count AS INTEGER))/2.0),2) AS avg\_count

, ROUND(AVG((a.rating + p.rating)/2),0) AS avg\_rating

, a.primary\_genre

, COUNT(\*) AS genre\_count

, ROUND((COUNT(\*) \* 100.0) / SUM(COUNT(\*)) OVER (), 2) AS genre\_percentage

FROM app\_store\_apps AS a

INNER JOIN play\_store\_apps AS p

ON a.name=p.name

GROUP BY a.name, a.primary\_genre

ORDER by avg\_rating DESC, avg\_count DESC, genre\_count DESC

------------------------------------------------------------------------------

--7)add the price of 0 OR 1 using case

SELECT

ROUND(AVG((CAST(REPLACE(a.price::TEXT, '$', '') AS NUMERIC) + CAST(REPLACE(p.price::TEXT, '$', '') AS NUMERIC))/2),2)AS avg\_price

, a.name

, a.primary\_genre

, ROUND(AVG((CAST(a.review\_count AS INTEGER) + CAST(p.review\_count AS INTEGER))/2.0),2) AS avg\_count

, ROUND(AVG((a.rating + p.rating)/2),0) AS avg\_rating

, a.primary\_genre

, COUNT(\*) AS genre\_count

, ROUND((COUNT(\*) \* 100.0) / SUM(COUNT(\*)) OVER (), 2) AS genre\_percentage

, (CASE WHEN ROUND(AVG((CAST(REPLACE(a.price::TEXT, '$', '') AS NUMERIC) + CAST(REPLACE(p.price::TEXT, '$', '') AS NUMERIC))/2),2)<1 THEN 10000

WHEN ROUND(AVG((CAST(REPLACE(a.price::TEXT, '$', '') AS NUMERIC) + CAST(REPLACE(p.price::TEXT, '$', '') AS NUMERIC))/2),2)<2 THEN 10000

ELSE NULL END) AS total\_price

FROM app\_store\_apps AS a

INNER JOIN play\_store\_apps AS p

ON a.name=p.name

GROUP BY a.name, a.primary\_genre

ORDER by avg\_rating DESC, avg\_count DESC, genre\_count DESC