Second: Write queries that directly answer predetermined questions from a business stakeholder

Launch MySQL server, I have used XAMPP localhost.

Step1: Create database “fetch_reards”:

Step2: Creation of Tables:

Users:

Run SQL query/queries on database fetch_rewards:

```sql
CREATE TABLE Users (
    user_id VARCHAR(255) PRIMARY KEY,
    active BOOLEAN,
    created_Date DATETIME,
    last_Login DATETIME,
    role VARCHAR(255),
    sign_Up_Source VARCHAR(255),
    state VARCHAR(255)
);
```

CPGs table:

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0004 seconds.)

```sql
CREATE TABLE CPGs ( cpg_id VARCHAR(255) PRIMARY KEY, -- MongoDB ObjectID converted to string name VARCHAR(255) );
```

[Edit inline] [Edit] [Create PHP code]

Brands table:

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0004 seconds.)

```sql
CREATE TABLE Brands ( brand_id VARCHAR(255) PRIMARY KEY, -- MongoDB ObjectID converted to string bar_Code VARCHAR(255), brand_Code VARCHAR(255), category VARCHAR(255),
category_Code VARCHAR(255), name VARCHAR(255), top_Brand BOOLEAN, cpg_id VARCHAR(255), -- Foreign key to CPGs table FOREIGN KEY (cpg_id) REFERENCES CPGs(cpg_id) );
```

[Edit inline] [Edit] [Create PHP code]

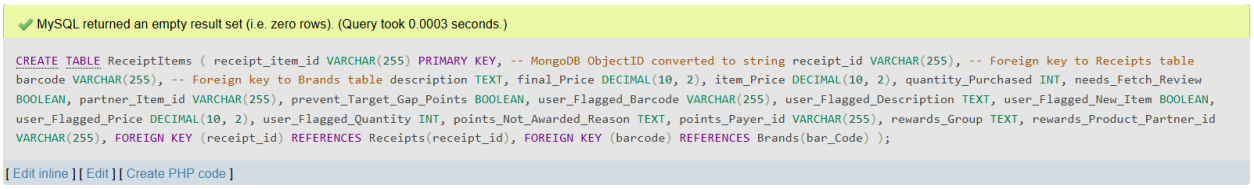
Receipts table:

MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```sql
CREATE TABLE Receipts ( receipt_id VARCHAR(255) PRIMARY KEY, -- MongoDB ObjectID converted to string bonus_Points_Earned INT, bonusPoints_Earned_Reason TEXT, create_Date
DATETIME, date_Scanned DATETIME, finished_Date DATETIME, modify_Date DATETIME, points_Awarded_Date DATETIME, points_Earned DECIMAL(10, 2), purchase_Date DATETIME,
purchased_Item_Count INT, rewards_Receipt_Status VARCHAR(255), total_Spent DECIMAL(10, 2), user_Id VARCHAR(255), -- Foreign key to Users table FOREIGN KEY (user_Id)
REFERENCES Users(user_id) );
```

[Edit inline] [Edit] [Create PHP code]

ReceiptItems table:



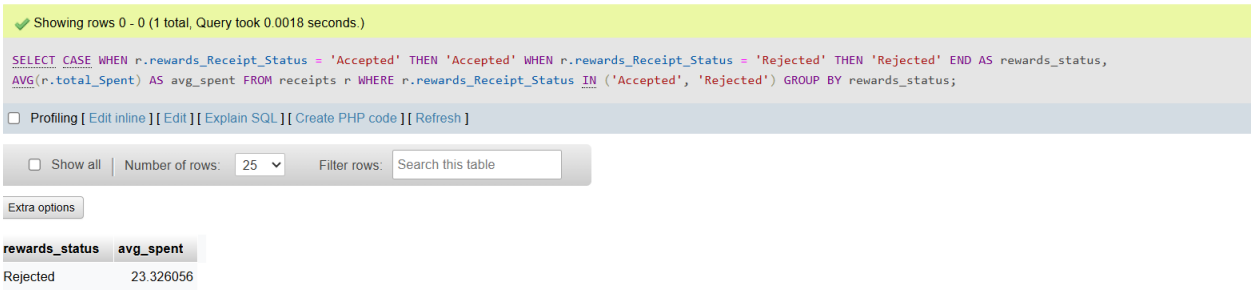
Step3: Import data from CSV files (user_normalized.csv, brands_normalized.csv, cpgs_normalized.csv, receipts_normalized.csv, receipt_items_normalized.csv) using “Import” feature of MySQL.

Step4: Write SQL queries against your new structured relational data model

Note: For queries involving data related to a specific month, such as the most recent month or the past six months, the current date has been considered as ‘March 1, 2021’. This is because, after converting the MongoDB milliseconds timestamp to a datetime format from the given JSON files, the latest recorded data falls within March 2021.

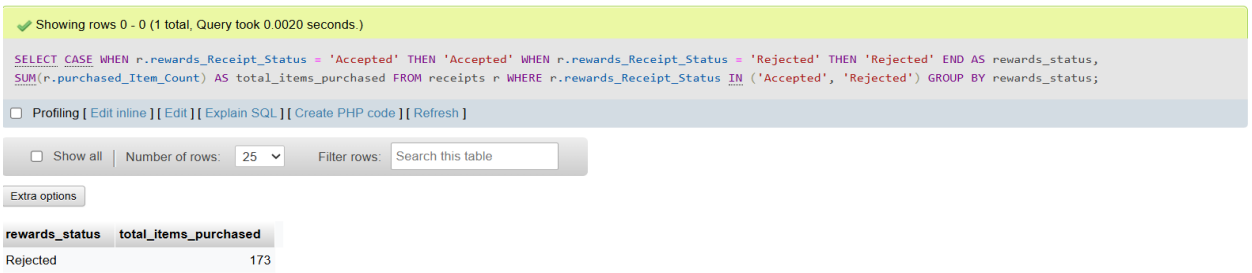
- When considering *average spend* from receipts with 'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?

Query:SELECT CASE WHEN r.rewards_Receipt_Status = 'Accepted' THEN 'Accepted' WHEN r.rewards_Receipt_Status = 'Rejected' THEN 'Rejected' END AS rewards_status, AVG(r.total_Spent) AS avg_spent FROM receipts r WHERE r.rewards_Receipt_Status IN ('Accepted', 'Rejected') GROUP BY rewards_status;



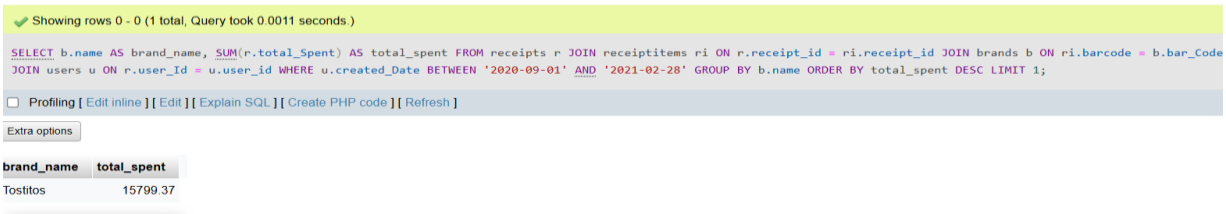
- When considering *total number of items purchased* from receipts with 'rewardsReceiptStatus’ of ‘Accepted’ or ‘Rejected’, which is greater?

Query: SELECT CASE WHEN r.rewards_Receipt_Status = 'Accepted' THEN 'Accepted' WHEN r.rewards_Receipt_Status = 'Rejected' THEN 'Rejected' END AS rewards_status, SUM(r.purchased_Item_Count) AS total_items_purchased FROM receipts r WHERE r.rewards_Receipt_Status IN ('Accepted', 'Rejected') GROUP BY rewards_status;



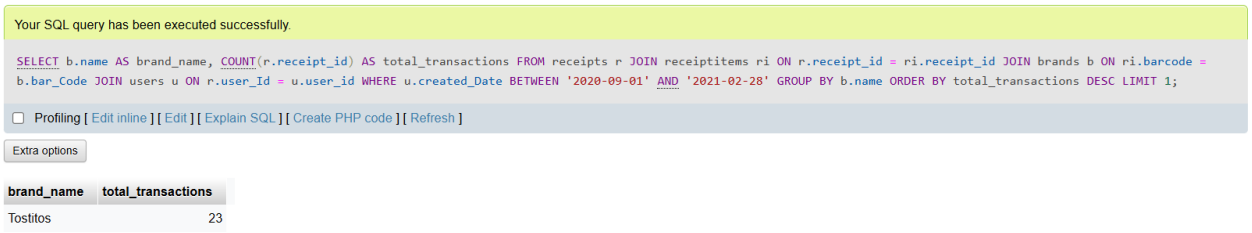
- Which brand has the most *spend* among users who were created within the past 6 months?

Query:SELECT b.name AS brand_name, SUM(r.total_Spent) AS total_spent FROM receipts r JOIN rece iptitems ri ON r.receipt_id = ri.receipt_id JOIN brands b ON ri.barcode = b.bar_Code JOIN users u ON r.user_Id = u.user_id WHERE u.created_Date BETWEEN '2020-09-01' AND '2021-02-28' GROUP BY b.name ORDER BY total_spent DESC LIMIT 1;



- Which brand has the most *transactions* among users who were created within the past 6 months?

Query:SELECT b.name AS brand_name, COUNT(r.receipt_id) AS total_transactions FROM receipts r J OIN receiptitems ri ON r.receipt_id = ri.receipt_id JOIN brands b ON ri.barcode = b.bar_Code JOIN us ers u ON r.user_Id = u.user_id WHERE u.created_Date BETWEEN '2020-09-01' AND '2021-02-28' GROUP BY b.name ORDER BY total_transactions DESC LIMIT 1;



- What are the top 5 brands by receipts scanned for most recent month?

Query:SELECT b.name AS brand_name, COUNT(ri_distinct.receipt_id) AS receipts_scanned FROM br ands b JOIN (SELECT DISTINCT r.receipt_id, ri.barcode FROM receipts r JOIN receiptitems ri ON r.rece ipt_id = ri.receipt_id WHERE r.date_Scanned BETWEEN '2021-02-01 00:00:00' AND '2021-03-01 23:59:59') AS ri_distinct ON ri_distinct.barcode = b.bar_Code GROUP BY b.name ORDER BY receipts_ scanned DESC LIMIT 5;

- How does the ranking of the top 5 brands by receipts scanned for the recent month compare to the ranking for the previous month?

Query: WITH recent_month AS (SELECT b.name AS brand_name, COUNT(ri_distinct.receipt_id) AS receipts_scanned FROM brands b JOIN (SELECT DISTINCT r.receipt_id, ri.barcode FROM receipts r JOIN receiptitems ri ON r.receipt_id = ri.receipt_id WHERE r.date_Scanned BETWEEN '2021-02-01 00:00:00' AND '2021-02-28 23:59:59') AS ri_distinct ON ri_distinct.barcode = b.bar_Code GROUP BY b.name ORDER BY receipts_scanned DESC LIMIT 5), previous_month AS (SELECT b.name AS brand_name, COUNT(ri_distinct.receipt_id) AS receipts_scanned FROM brands b JOIN (SELECT DISTINCT r.receipt_id, ri.barcode FROM receipts r JOIN receiptitems ri ON r.receipt_id = ri.receipt_id WHERE r.date_Scanned BETWEEN '2021-01-01 00:00:00' AND '2021-01-31 23:59:59') AS ri_distinct ON ri_distinct.barcode = b.bar_Code GROUP BY b.name ORDER BY receipts_scanned DESC LIMIT 5)SELECT recent_month.brand_name,recent_month.receipts_scanned AS recent_month_scanned,previous_month.receipts_scanned AS previous_month_scanned FROM recent_month LEFT JOIN previous_month ON recent_month.brand_name = previous_month.brand_name ORDER BY recent_month_scanned DESC;