**SQL Assignment**

# Data sets used:

1. Products.csv
2. Stores.csv
3. Counties.csv
4. Sales.csv

# Database creation:

Environment: PostgreSQL

Database name: shaddida

# Table creation:

## Counties table

CREATE TABLE counties (

County VARCHAR(50) PRIMARY KEY,

Population VARCHAR(255)

);

## Products table

CREATE TABLE products (

Item\_no INT PRIMARY KEY,

Category\_name VARCHAR(100),

Item\_description VARCHAR(255),

Vendor INT,

Vendor\_name VARCHAR(100),

Bottle\_size INT,

Pack INT,

Inner\_pack INT,

Age VARCHAR(20),

Proof VARCHAR(50),

List\_date TIMESTAMP,

Upc VARCHAR(20),

Scc VARCHAR(20),

Bottle\_price NUMERIC(10,2),

Shelf\_price NUMERIC(10,2),

case\_cost NUMERIC(10,2)

);

## Stores table

CREATE TABLE stores (

Store INT PRIMARY KEY,

Name VARCHAR(100),

Store\_status VARCHAR(1),

Store\_address VARCHAR(100),

Address\_info VARCHAR(20)

);

## Sales table

CREATE TABLE sales (

Date TIMESTAMP,

Convenience\_store VARCHAR(100),

Store INT,

County\_number VARCHAR(50),

County VARCHAR(50),

Category VARCHAR(50),

Category\_name VARCHAR(50),

Vendor\_no VARCHAR(100),

Vendor VARCHAR(50),

Item INT,

Description VARCHAR(100),

Pack INT,

Liter\_size INT,

State\_btl\_cost NUMERIC(10,2),

Btl\_price NUMERIC(10,2),

Bottle\_qty INT,

Total NUMERIC(10,2)

);

# Data loading:

\copy counties FROM 'C:/Users/91991/Desktop/shaddiDa/counties.csv' DELIMITER '|' CSV

\copy products FROM 'C:/Users/91991/Desktop/shaddiDa/productsV1.csv' DELIMITER '|' CSV

\copy stores FROM 'C:/Users/91991/Desktop/shaddiDa/storesV1.csv' DELIMITER '|' CSV

\copy sales FROM 'C:/Users/91991/Desktop/shaddiDa/salesV1.csv' DELIMITER '|' CSV

# Data Cleaning:

Data cleaning is done using simple python script and are few operations performed by script.

1. Removed unnecessary "$" symbols from products and sales dataset
2. Removed unwanted "\s" and "\n" from stores dataset
3. Few rows in sales dataset have amount stored as string. i.e. 5800.00 is stored as "5,800.00", which throws error while loading data in table.

# SQL Questions:

1. **Which products have a case cost of more than $100?**

**Query:** SELECT item\_description as product\_name FROM products WHERE case\_cost > 100;

**Response:** 3730 rows

1. **Which tequilas have a case cost of more than $100?**

**Query:** SELECT item\_description FROM products WHERE category\_name = 'TEQUILA' AND case\_cost > 100;

**Response:** 422 rows

1. **Which tequilas or scotch whiskies have a case cost of more than $100?**

**Query:** SELECT item\_description,category\_name FROM products WHERE case\_cost >100 GROUP BY category\_name, item\_description HAVING category\_name LIKE '%TEQUILA%' OR category\_name LIKE '%SCOTCH%';

**Response:** 856 rows

1. **Which tequilas or scotch whiskies have a case cost between $100 and $120?**

**Query:** SELECT item\_description FROM products WHERE (case\_cost >= 100 AND case\_cost <= 120) AND (category\_name LIKE '%TEQUILA%' OR category\_name LIKE '%SCOTCH%');

**Response:** 120 rows

1. **Which whiskies of any kind cost more than $100?**

**Query:** SELECT item\_description FROM products WHERE bottle\_price > 100 AND (category\_name LIKE '%WHIS%');

**Response:** 40 rows

1. **Which whiskies of any kind cost between $100 and $150?**

**Query:** SELECT item\_description FROM products WHERE (bottle\_price BETWEEN 100 AND 150) AND (category\_name LIKE '%WHIS%');

**Response:** 26 rows

1. **Which products except tequilas cost between $100 and $120?**

**Query:** SELECT item\_description FROM products WHERE (bottle\_price BETWEEN 100 AND 120) AND (category\_name NOT LIKE '%TEQ%');

**Response:** 41 rows

1. **Which products are not from vendor ‘Jim Beam Brands’?**

**Query:** SELECT item\_description FROM products WHERE vendor\_name != 'Jim Beam Brands';

**Response:** 9052 rows

1. **Which products are over 90 proof?**

**Query:** SELECT item\_description FROM products WHERE CAST(proof AS INT) > 90;

**Response:** 765 rows

1. **Which products have a case cost of less than $60?**

**Query:** SELECT item\_description FROM products WHERE case\_cost < 60;

**Response:** 2826 rows

1. **Which products are either Single Malt Scotches or Canadian Whiskies (based on category name)?**

**Query:** SELECT item\_description FROM products WHERE category\_name IN ('SINGLE MALT SCOTCH','CANADIAN WHISKIES');

**Response:** 682 rows

1. **Which products have ‘Whiskies’ in the category name?**

**Query:** SELECT item\_description FROM products WHERE category\_name LIKE '%WHISK%';

**Response:** 1606 rows

1. **Which products have a shelf\_price between $4 and $10?**

**Query:** SELECT item\_description FROM products WHERE shelf\_price BETWEEN 4 AND 10;

**Response:** 2701 rows

1. **Which products have a bottle\_price between $4 and $10?**

**Query:** SELECT item\_description FROM products WHERE bottle\_price BETWEEN 4 AND 10;

**Response:** 3278 rows

1. **Which unique products come in packs larger than 12?**

**Query:** SELECT DISTINCT item\_description FROM products WHERE pack > 12;

**Response:** 1106 rows

1. **How many unique products have less than 12 in a pack?**

**Query:** SELECT COUNT(DISTINCT item\_description) FROM products WHERE pack < 12;

**Response:** 1 row (3595)

1. **Which unique products have a case cost of less than $70?**

**Query:** SELECT DISTINCT item\_description FROM products WHERE case\_cost > 70;

**Response:** 4718 rows

1. **Which unique products come in packs larger than 12 AND have a case\_cost of less than $70?**

**Query:** SELECT DISTINCT item\_description FROM products WHERE case\_cost < 70 AND pack > 12;

**Response:** 553 rows

1. **How many items in the product offering are being sold?**

**Query:** SELECT COUNT(DISTINCT description) FROM sales;

**Response:** 1 row (2958)

1. **Using bottle\_qty to size up item demand, query for the volume of bottles being sold by item description (as multiple item numbers exist for same products being sold by various vendors), calculate and rank the best selling items.**

**Query:** SELECT sum(Liter\_size) as "total\_volume", sum(bottle\_qty) as "total\_quantity\_sold", description, DENSE\_RANK() OVER (ORDER BY sum(bottle\_qty) DESC) Rank FROM sales GROUP BY description ORDER BY Rank;

**Response:** 2958 rows

1. **Narrow the focus of your analysis in item #20 above to include only the products with higher proof contents. Limit the focus to products with proof percent value greater than 80.**

**Query:** SELECT sum(s.Liter\_size) as "total\_volume", sum(s.bottle\_qty) as "total\_quantity\_sold", s.description, DENSE\_RANK() OVER (ORDER BY sum(s.bottle\_qty) DESC) Rank FROM sales s INNER JOIN products p ON s.description = p.item\_description WHERE CAST(p.proof as INT) > 80 GROUP BY description ORDER BY Rank LIMIT 10;

**Response:** 10 rows

1. **How many unique store locations have recorded sales?**

**Query:** SELECT COUNT(DISTINCT store) FROM sales;

**Response:** 1 row (1352)

1. **Create a list of all the transactions from the active stores locations in Polk county?**

**Query:** SELECT \* FROM sales s INNER JOIN stores st ON s.store = st.store WHERE st.store\_status = 'A' AND s.county = 'Polk';

**Response:** 561390 rows

1. **Create a list of all the store locations in Polk county, and show the sum total of their recorded sales along with the quantity of transactions?**

**Query:** SELECT COUNT(\*) as transaction\_count, store FROM sales WHERE county = 'Polk' GROUP BY store ORDER BY transaction\_count DESC;

**Response:** 197 rows

1. **Using Sales as the primary table, create links to Products and Stores tables.**

* **In the output, list the store number (from sales), category\_name (from sales) and two aggregated columns: average bottle\_price (from products) and average total price (from sales).**
* **Use a compounded WHERE clause to limit the calculations to the sales of Tequila (category\_name from Sales) from Active stores in Mason City, Iowa.**
* **Group and sort the data by the store number.**

**Query:** SELECT s.store, s.category\_name, AVG(p.bottle\_price\_$) as "Avg\_bottle\_price", AVG(s.total) as "total\_sale" FROM sales s INNER JOIN stores st ON s.store = st.store INNER JOIN products p ON s.category\_name = p.category\_name WHERE s.category\_name LIKE '%TEQ%' AND ((st.name LIKE '%Iowa%' OR st.name LIKE '%Mason City%') AND st.store\_status = 'A') GROUP BY s.category\_name, s.store ORDER BY s.store;

**Response:** 32 rows