**SQL queries and outputs of *e\_commerce data analysis project***

**Basic Queries**

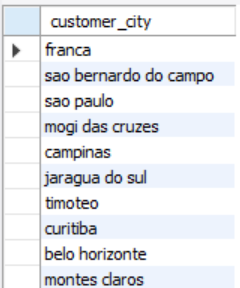
**1. List all unique cities where customers are located.**

**Code:**

*SELECT DISTINCT customer\_city*

*FROM customers;*

**Output:**

****

**2. Count the number of orders placed in 2017.**

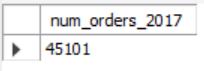
**Code:**

*SELECT COUNT(order\_id) AS num\_orders\_2017*

*FROM orders*

*WHERE YEAR(order\_purchase\_timestamp) = 2017;*

**Output:**

****

**3. Find the total sales per category.**

**Code:**

*SELECT p.product\_category AS product\_category, ROUND(SUM(oi.price + oi.freight\_value),2) AS category\_sales -- if we want to include shipping cost as well*

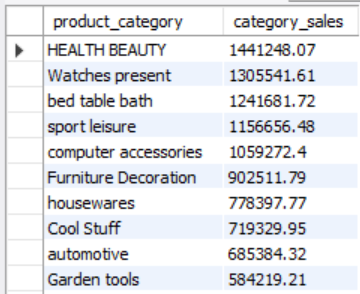
*FROM products p*

*JOIN order\_items oi ON oi.product\_id = p.product\_id*

*GROUP BY p.product\_category*

*ORDER BY category\_sales DESC;*

**Output:**

****

**Code:**

*SELECT p.product\_category AS product\_category, ROUND(SUM(oi.price),2) AS category\_sales -- if we don't want to include shipping cost as well*

*FROM products p*

*JOIN order\_items oi ON oi.product\_id = p.product\_id*

*GROUP BY p.product\_category*

*ORDER BY category\_sales DESC;*

**Output:**

****

**4. Calculate the percentage of orders that were paid in installments.**

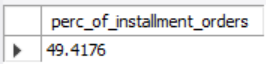
**Code:**

*SELECT*

*COUNT((CASE WHEN payment\_installments > 1 THEN order\_id END))\*100/COUNT(\*) AS perc\_of\_installment\_orders*

*FROM payments;*

**Output:**

****

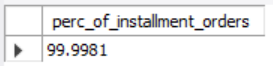
**Code:**

*SELECT -- Considering payment\_installments = 1 as installment as there are few rows with payment\_installments = 0*

*COUNT((CASE WHEN payment\_installments >= 1 THEN order\_id END))\*100/COUNT(\*) AS perc\_of\_installment\_orders*

*FROM payments;*

**Output:**

****

**5. Count the number of customers from each state.**

**Code:**

*SELECT*

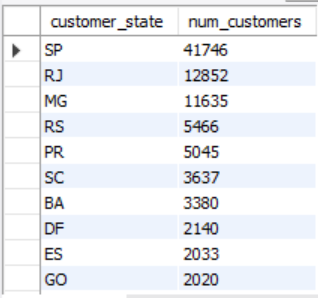
*customer\_state, COUNT(customer\_id) AS num\_customers*

*FROM customers*

*GROUP BY customer\_state*

*ORDER BY num\_customers DESC;*

**Output:**

****

**Intermediate Queries**

**1. Calculate the number of orders per month in 2018.**

**Code:**

*WITH num\_orders\_2018 AS(*

*SELECT*

*MONTHNAME(order\_purchase\_timestamp) AS month\_of\_2018,*

*MONTH(order\_purchase\_timestamp) AS month\_num,*

*COUNT(order\_id) AS num\_orders*

*FROM orders*

*WHERE YEAR(order\_purchase\_timestamp) = 2018*

*GROUP BY month\_of\_2018, month\_num*

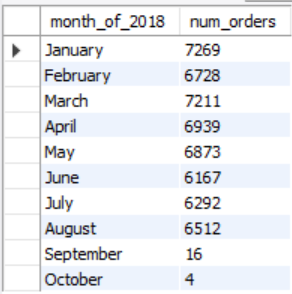
*ORDER BY month\_num*

*)*

*SELECT month\_of\_2018, num\_orders*

*FROM num\_orders\_2018;*

**Output:**

****

**2. Find the average number of products per order, grouped by customer state.**

**Code:**

*WITH state\_order\_items AS (*

*SELECT*

*c.customer\_state, oi.order\_id, COUNT(\*) AS num\_items*

*FROM order\_items oi*

*JOIN orders o ON o.order\_id = oi.order\_id*

*JOIN customers c ON c.customer\_id = o.customer\_id*

*GROUP BY c.customer\_state, oi.order\_id*

*)*

*SELECT*

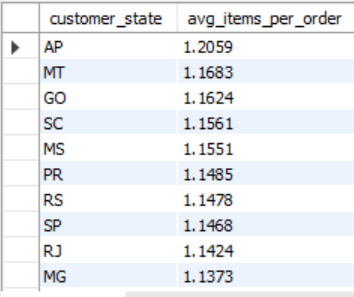
*customer\_state, AVG(num\_items) AS avg\_items\_per\_order*

*FROM state\_order\_items*

*GROUP BY customer\_state*

*ORDER BY avg\_items\_per\_order DESC;*

**Output:**

****

**3. Calculate the percentage of total revenue contributed by each product category.**

**Code:**

*SELECT*

*p.product\_category,*

*ROUND(SUM(oi.price)\*100/(SELECT SUM(price) FROM order\_items),2) AS perc\_contribution\_in\_sales*

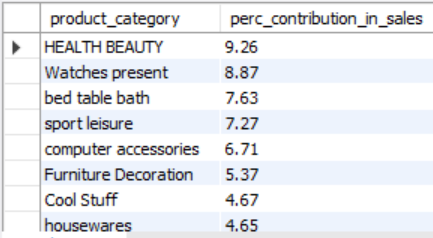
*FROM order\_items oi*

*JOIN products p ON p.product\_id = oi.product\_id*

*GROUP BY p.product\_category*

*ORDER BY perc\_contribution\_in\_sales DESC;*

**Output:**

****

**4. Identify the correlation between product price and the number of times a product has been purchased(total no. of products here will be from product\_category)**

**Code:**

*SELECT*

*p.product\_category,*

*COUNT(oi.product\_id) AS total\_items\_puchased,*

*ROUND(SUM(oi.price),2) AS sales\_amount,*

*ROUND(AVG(price),2) AS avg\_price\_per\_item*

*FROM order\_items oi*

*JOIN products p ON p.product\_id = oi.product\_id*

*GROUP BY p.product\_category;*

*/\**

*Correlation in python - code*

*query = """SELECT*

*p.product\_category,*

*COUNT(oi.product\_id) AS total\_items\_puchased,*

*ROUND(SUM(oi.price),2) AS sales\_amount,*

*ROUND(AVG(price),2) AS avg\_price\_per\_item*

*FROM order\_items oi*

*JOIN products p ON p.product\_id = oi.product\_id*

*GROUP BY p.product\_category;"""*

*cur.execute(query)*

*data = cur.fetchall()*

*df = pd.DataFrame(data,columns = ["product\_category", "total\_items\_puchased", "sales\_amount", "avg\_price\_per\_item"])*

*df.head()*

*import numpy as np*

*arr1 = df["total\_items\_puchased"]*

*arr2 = df["avg\_price\_per\_item"]*

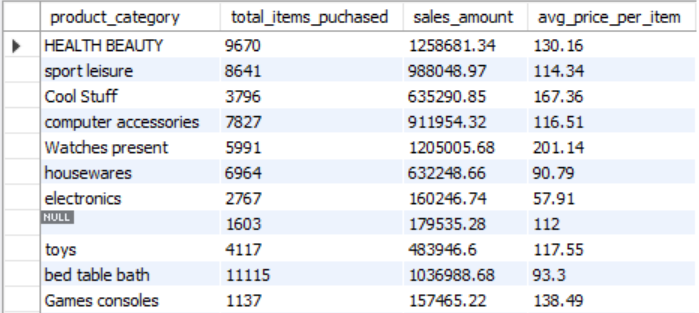
*a = np.corrcoef([arr1,arr2])*

*print(f"The correlation coefficient is \033[1m{round(a[0][-1],3)}\033[0m." )*

*\*/*

**Output:**

**SQL Table**

****

**Python**

****

**5. Calculate the total revenue generated by each seller, and rank them by revenue.**

**Code:**

*SELECT*

*oi.seller\_id, ROUND(SUM(payment\_value),2) AS seller\_revenue,*

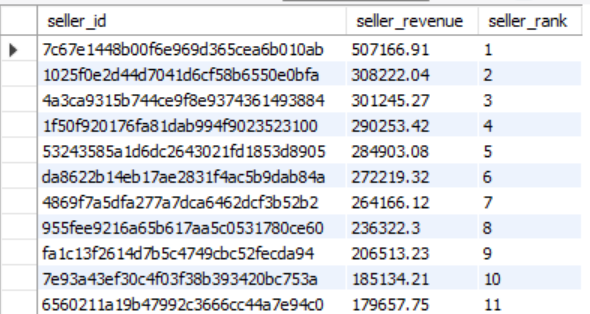
*DENSE\_RANK() OVER(ORDER BY SUM(payment\_value) DESC) AS seller\_rank*

*FROM order\_items oi*

*JOIN payments p ON p.order\_id = oi.order\_id*

*GROUP BY oi.seller\_id;*

**Output:**

****

**Advanced Queries**

**1. Calculate the moving average of order values for each customer over their order history.(say moving average of last 3 orders)**

**Code:**

*SELECT*

*t.customer\_id, t.order\_purchase\_timestamp, t.payment\_value, customer\_order\_count, moving\_avg\_3\_orders*

*FROM(SELECT*

*o.customer\_id, o.order\_purchase\_timestamp, p.payment\_value,*

*COUNT(\*) OVER (PARTITION BY o.customer\_id) AS customer\_order\_count,*

*ROUND(AVG(p.payment\_value)OVER(PARTITION BY o.customer\_id ORDER BY o.order\_purchase\_timestamp ROWS*

*BETWEEN 2 PRECEDING AND CURRENT ROW),2) AS moving\_avg\_3\_orders*

*FROM orders o*

*JOIN payments p ON p.order\_id = o.order\_id) AS t*

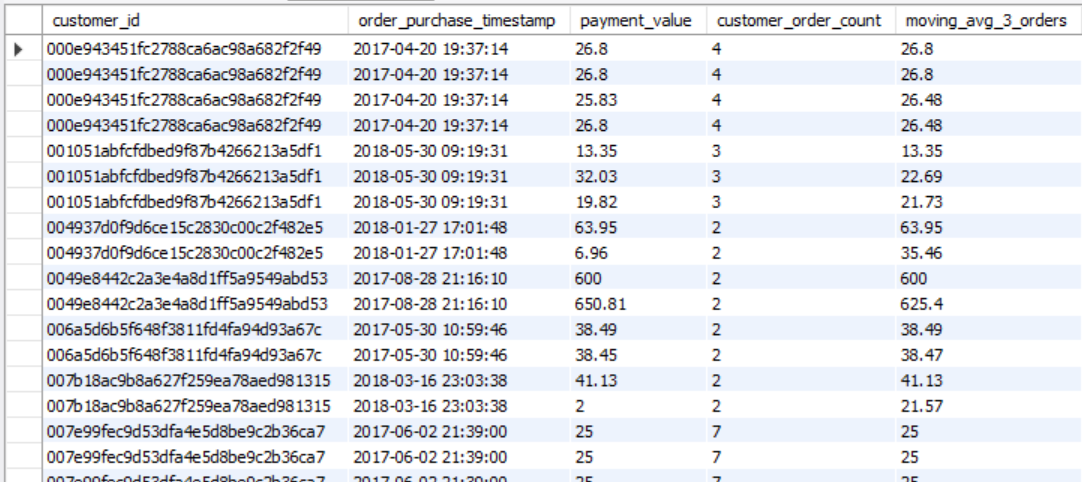
*WHERE customer\_order\_count > 1*

*-- customers with More than 1 order will be filtered in this data and*

*-- hence moving average will be calculated as per 2, 3, 4 or more orders*

*;*

**Output:**

****

**2. Calculate the cumulative sales per month for each year.**

**Code:**

*WITH monthly\_sales AS (*

*SELECT*

*YEAR(o.order\_purchase\_timestamp) AS year, MONTH(o.order\_purchase\_timestamp) AS month\_num, MONTHNAME(o.order\_purchase\_timestamp) AS month,*

*ROUND(SUM(p.payment\_value),2) AS monthly\_sales*

*FROM orders o*

*JOIN payments p ON p.order\_id = o.order\_id*

*GROUP BY year, month\_num, month*

*ORDER BY year, month\_num*

*)*

*SELECT*

*year, month, monthly\_sales,*

*ROUND(SUM(monthly\_sales) OVER(ORDER BY year, month\_num),2) AS monthly\_cumulative\_sales*

*FROM monthly\_sales;*

**Output:**

****

**3. Calculate the year-over-year growth rate of total sales.**

**Code:**

*SELECT*

*year, yearly\_sales, previous\_year\_sales,*

*ROUND((yearly\_sales - previous\_year\_sales)\*100/previous\_year\_sales,2) AS YoY\_growth\_percent*

*FROM (SELECT*

*YEAR(o.order\_purchase\_timestamp) AS year, ROUND(SUM(p.payment\_value),2) AS yearly\_sales,*

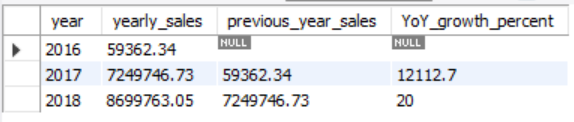
*LAG(ROUND(SUM(p.payment\_value),2),1) OVER(ORDER BY YEAR(o.order\_purchase\_timestamp)) AS previous\_year\_sales*

*FROM orders o*

*JOIN payments p ON p.order\_id = o.order\_id*

*GROUP BY year) AS year\_sale;*

**Output:**

****

**4. Calculate the month\_on\_month growth rate of total sales.**

**Code:**

*WITH monthly\_sales AS (*

*SELECT*

*YEAR(o.order\_purchase\_timestamp) AS year, MONTH(o.order\_purchase\_timestamp) AS month\_num, MONTHNAME(o.order\_purchase\_timestamp) AS month,*

*ROUND(SUM(p.payment\_value),2) AS monthly\_sales*

*FROM orders o*

*JOIN payments p ON p.order\_id = o.order\_id*

*GROUP BY year, month\_num, month*

*ORDER BY year, month\_num*

*)*

*SELECT*

*year, month, monthly\_sales,*

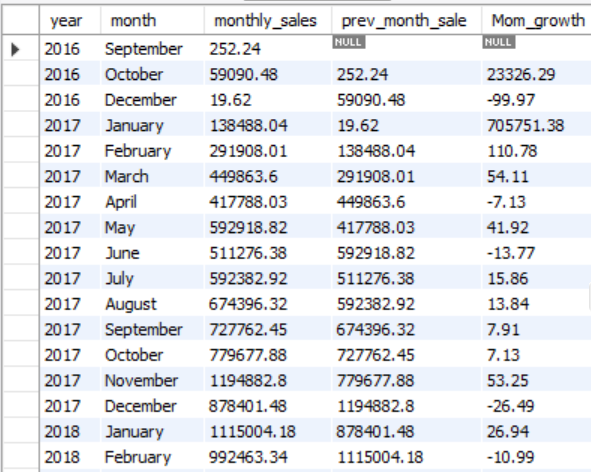
*LAG(monthly\_sales,1) OVER(ORDER BY year, month\_num) AS prev\_month\_sale,*

*ROUND((monthly\_sales - LAG(monthly\_sales,1) OVER(ORDER BY year, month\_num))\*100/LAG(monthly\_sales,1) OVER(ORDER BY year, month\_num),2) AS*

*Mom\_growth*

*FROM monthly\_sales;*

**Output:**

****

**5. Identify the top 3 customers who spent the most money in each year.**

**Code:**

*WITH customer\_spending AS (SELECT*

*YEAR(o.order\_purchase\_timestamp) AS year,*

*o.customer\_id, ROUND(SUM(p.payment\_value),2) AS spent,*

*-- Don't partition by customer\_id because then every customer will be assigned a new group and respective rank i.e. 1*

*DENSE\_RANK() OVER(PARTITION BY YEAR(o.order\_purchase\_timestamp) ORDER BY ROUND(SUM(p.payment\_value),2) DESC) AS customer\_rank\_in\_year*

*FROM orders o*

*JOIN payments p ON p.order\_id = o.order\_id*

*GROUP BY year, o.customer\_id*

*ORDER BY year)*

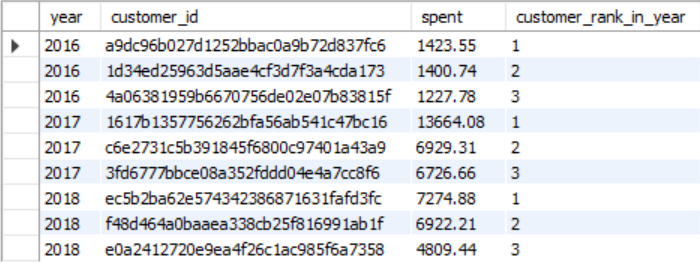
*SELECT*

*year, customer\_id, spent, customer\_rank\_in\_year*

*FROM customer\_spending*

*WHERE customer\_rank\_in\_year <= 3;*

**Output:**

****