Unicorn Project

Team

- Vishnu Padmini Lanka
- Bianca Niemann
- Meenakshi Subramaniam
- Julius Abongwa

Mentors

- Sonia Seguin
- Eftychia Kolasi

Data Exploration with SQL

```
import pandas as pd
import sqlalchemy as sa
pd.set_option('display.max_rows', 1000)

unicorn_url = 'postgresql://Test:bQNxVzJL4g6u@ep-noisy-flower-846766-pooler.us-ea

engine = sa.create_engine(unicorn_url)
connection = engine.connect().execution_options(isolation_level="AUTOCOMMIT")
```

Tasks:

```
# 1.How many customers do we have in the data?
query = """
SELECT COUNT(DISTINCT customer_id)
FROM customers
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 2.What was the city with the most profit for the company in 2015?
query = """
SELECT shipping_city, SUM(order_profits) AS city_profits
FROM order_details od
JOIN orders o
ON o.order_id = od.order_id
WHERE EXTRACT(YEAR FROM o.order_date) = 2015
GROUP BY shipping_city
ORDER BY city_profits DESC
LIMIT 1
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 3.In 2015, what was the most profitable city's profit?
query = """
WITH profit_table AS (
SELECT shipping_city, SUM(od.order_profits) AS total_profit
FROM orders o
LEFT JOIN order_details od USING(order_id)
WHERE EXTRACT('YEAR' FROM o.order_date) = 2015
GROUP BY o.shipping_city
ORDER BY total_profit DESC
LIMIT 1
)
SELECT total_profit
FROM profit_table;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 4.How many different cities do we have in the data? Please refer just to the ci
query = """
SELECT COUNT(DISTINCT shipping_city)
FROM orders;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 5.Show the total spent by customers from low to high.
query = """

SELECT cu.customer_id, SUM(od.order_sales) AS total_spent
FROM order_details od
```

```
JOIN orders ord USING(order_id)
JOIN customers cu USING(customer_id)
GROUP BY cu.customer_id
ORDER BY total_spent
LIMIT 100
;

"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 6.What is the most profitable city in the State of Tennessee?
query = """
SELECT o.shipping_city, SUM(od.order_profits) AS total_profit
FROM order_details od
LEFT JOIN orders o USING(order_id)
WHERE o.shipping_state = 'Tennessee'
GROUP BY o.shipping_city
ORDER BY total_profit DESC
LIMIT 1;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 7.What’s the average annual profit for that city across all years?
query = """
WITH sum_over_years AS (
SELECT EXTRACT(YEAR FROM o.order_date) AS year, SUM(od.order_profits) AS total_pr
FROM order_details od
LEFT JOIN orders o USING(order_id)
WHERE o.shipping_state = 'Tennessee' AND o.shipping_city = 'Lebanon'
GROUP BY o.shipping_city, year
ORDER BY total_profit DESC
)
SELECT AVG(total_profit)
FROM sum_over_years
;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
WITH top_city_t AS (
SELECT shipping_city,
            SUM(order_profits)
FROM orders
JOIN order_details USING(order_id)
WHERE shipping_state = 'Tennessee'
```

```
GROUP BY shipping_city
ORDER BY  SUM(order_profits) DESC
LIMIT 1
)

SELECT AVG (annual_profits)
FROM (
SELECT EXTRACT(YEAR FROM order_date) as year,
      SUM(order_profits) as annual_profits
FROM orders
JOIN order_details USING(order_id)
WHERE shipping_state = 'Tennessee'
AND shipping_city = (SELECT shipping_city FROM top_city_t)
GROUP BY year
) as a


#8.What is the distribution of customer types in the data?
query = """
SELECT COUNT(customer_id) AS total_customer_distribution,customer_segment
FROM customers
GROUP BY customer_segment;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 9.What's the most profitable product category on average in Iowa across all yea
query = """
  SELECT product_category,AVG(order_profits) AS order_profits--,shipping_date
  FROM order_details od
  JOIN product p
  USING (product_id)
  JOIN orders
  USING (order_id)
  WHERE shipping_state = 'Iowa'
  GROUP BY product_category--,shipping_date
  ORDER BY order_profits DESC
  LIMIT 1;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#10.What is the most popular product in that category across all states in 2016?
query = """
SELECT product_name, SUM(quantity) AS total_quantity
  FROM order_details od
  JOIN product p
  USING (product_id)
  JOIN orders
  USING (order_id)
```

```
  WHERE EXTRACT(YEAR FROM order_date) = 2016 AND product_category = 'Furniture'
  GROUP BY product_name
  ORDER BY total_quantity DESC
  LIMIT 1;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#11.Which customer got the most discount in the data? (in total amount)
query1 = """
WITH joined_tables AS
    (SELECT *
    FROM order_details
    LEFT JOIN orders
    USING (order_id)
    LEFT JOIN customers
    USING (customer_id)
    LEFT JOIN product
    USING (product_id)),

    discount_calculations AS
(SELECT order_details_id, order_sales, order_discount, order_sales/(1-order_disco
(order_sales/(1-order_discount)) - order_sales AS absolute_discount_1,
(order_sales/(1-order_discount)) * order_discount AS absolute_discount_2,
customer_id
    FROM joined_tables
)

SELECT SUM(absolute_discount_1) AS total_absolute_discount,customer_id
FROM discount_calculations
GROUP BY customer_id
ORDER BY total_absolute_discount DESC
LIMIT 1;
"""

display(pd.read_sql(sa.text(query1),connection))
```

Show hidden output

```
# 12.How widely did monthly profits vary in 2018?
query = """
WITH months AS
(
  SELECT EXTRACT(month FROM order_date) AS months, SUM(order_profits) AS total_pr
FROM order_details od
JOIN orders o
USING (order_id)
WHERE EXTRACT(YEAR FROM order_date) = 2018
GROUP BY months
)
SELECT *,LAG(total_profits) OVER(ORDER BY months) as prev_months_profits,
```

```
total_profits - LAG(total_profits) OVER(ORDER BY months) as diff_profits
FROM months;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 13.Which was the biggest order regarding sales in 2015?
query = """
SELECT order_id, MAX(order_sales) AS max_order_sales
FROM order_details od
JOIN orders o
USING(order_id)
WHERE EXTRACT(YEAR FROM order_date) = 2015
GROUP BY order_id
ORDER BY max_order_sales DESC
LIMIT 1;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#14.What was the rank of each city in the East region in 2015 in quantity?
query1 = """
WITH east_city AS
(
  SELECT shipping_city, SUM(quantity) AS sum_quantity
  FROM orders o
  JOIN order_details od
  USING (order_id)
  WHERE EXTRACT(YEAR FROM order_date) = 2015 AND
  shipping_region = 'East'
  GROUP BY shipping_city
  --ORDER BY sum_quantity DESC
 )
 SELECT *,
  DENSE_RANK() OVER(ORDER BY sum_quantity DESC)
 FROM east_city;
"""
#Solution 2
query2 = """
WITH east_city AS
(
  SELECT shipping_city, SUM(quantity) AS sum_quantity
  FROM orders o
  JOIN order_details od
  USING (order_id)
  WHERE EXTRACT(YEAR FROM order_date) = 2015 AND
  shipping_region = 'East'
  GROUP BY shipping_city
  --ORDER BY sum_quantity DESC
```

```
 )
 SELECT *,
  RANK() OVER(ORDER BY sum_quantity DESC)
 FROM east_city;
"""

display(pd.read_sql(sa.text(query1),connection))
#display(pd.read_sql(sa.text(query2),connection))
```

Show hidden output

```
# 15.Display customer names for customers who are in the segment 'Consumer' or 'C
query = """
SELECT COUNT(DISTINCT customer_name)
FROM customers
WHERE customer_segment IN('Consumer','Corporate');
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#16.Calculate the difference between the largest and smallest order quantities fo
query = """
SELECT MAX(quantity)-MIN(quantity) AS diff_quantity
FROM order_details
WHERE product_id = 100;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#17.Calculate the percent of products that are within the category 'Furniture.' (
query1 = """
WITH product_count AS
(
SELECT COUNT(*) AS total_prod, product_category
FROM product
GROUP BY product_category
),
sum_count AS
(
SELECT SUM(total_prod) AS total_sum
FROM product_count
  )
  SELECT *,total_prod *100/ total_sum AS perc
  FROM product_count
  CROSS JOIN sum_count;
"""

#17.Calculate the percent of products that are within the category 'Furniture.' (
query2 = """
```

```
WITH product_count AS
(
SELECT COUNT(*) AS total_prod--, product_category
FROM product
--GROUP BY COUNT(*)
),
sum_count AS
(
SELECT COUNT(*) AS total_sum_furn--SUM(total_prod) AS total_sum
FROM product
  WHERE product_category = 'Furniture'
)
  SELECT *,ROUND(total_sum_furn * 100.0 /total_prod,2) AS perc
  FROM product_count
  CROSS JOIN sum_count;

"""



display(pd.read_sql(sa.text(query1),connection))
#display(pd.read_sql(sa.text(query2),connection))
```

Show hidden output

```
#18.Display the number of product manufacturers with more than 1 product in the p
query = """
SELECT DISTINCT product_manufacturer,COUNT(product_id)
FROM product
GROUP BY product_manufacturer
HAVING  COUNT(product_id)>1;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
# 19.Find what product manufacturers has more than 2 products . EX: A product wit
query = """
WITH prod_man AS
(
  SELECT COUNT (*),product_manufacturer
  FROM product
  GROUP BY product_manufacturer
  HAVING  COUNT(*)>1
 )
 SELECT product_manufacturer,COUNT(*)
 FROM prod_man
 GROUP BY product_manufacturer;

"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#20.Show the product_subcategory and the total number of products in the subcateg
query = """
SELECT product_subcategory, COUNT(*) AS total_prod
FROM product
GROUP BY product_subcategory
ORDER BY total_prod DESC, product_subcategory ASC;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#21.Show the order for the most to least number of products.
query = """
SELECT COUNT(product_id) AS prod_count, order_id
FROM order_details od
JOIN product p
USING (product_id)
GROUP BY order_id
ORDER BY prod_count DESC;
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output

```
#22.Show the product_id(s), the sum of quantities, where for each sale of product
query = """
SELECT product_id, SUM(quantity) AS total_quantity
FROM order_details
WHERE quantity >=100
GROUP BY product_id
ORDER BY total_quantity DESC
"""

display(pd.read_sql(sa.text(query),connection))
```

Show hidden output