Assignment:

1. SQL
2. Python Pandas

QUESTIONS SHEET:

[EASY Questions]

1. First and Last Purchase Date for Each Customer
2. Select All Products in Our Catalog
3. Find Specific Products with a Sales Price Greater Than $50
4. Count of Orders for Each Customer
5. Total Number of Unique Customers That Have Purchased from Us
6. Total Number of Unique Customers by Product
7. Summarize Sales by Product
8. Summarize Sales by Month
9. Find Orders Within a Date Range
10. Customers with No Orders
11. Total Sales Per Customer
12. Average Sales Price of Products in Each Category
13. Find the Top 5 Products with the Highest Total Sales
14. Find the Total Number of Returns Made by Each Customer
15. Total Clicks Grouped by Month
16. Total Clicks by Device Type
17. Total Clicks from Campaigns
18. Conversion Rate by Device Type (Clicks and Orders)

[MEDIUM QUESTIONS]

1. Highest Total Sale for an Order & Total Orders at Highest Price
2. Customers Who Have Ordered Every Product
3. Total Lifetime Sales, Order Count, and Average Order Value for Every Customer
4. Highest Number of Orders Placed by a Customer & Who Was It
5. What Did This Customer Purchase? (Follow-up to Previous Question)
6. Highest Total Returns Made by a Customer
7. Top 5 Reasons for Returning a Package
8. Total Amount of Sales Price Returned from Orders

9. Breakout Amount of Returned Orders by Product (Including Product Name)
10. Percent of Total Orders Returned
11. Percent of Total Orders Returned by Product
12. Highest Returned Product by Percent
13. Conversion Rate Calculation (Clicks to Orders)
14. Campaign Analysis: Total Orders, Clicks, Sales, Units for Each Campaign
15. Campaign with the Highest Conversion Rate
16. Discount/Promo That Generated the Most Sales

[HARD QUESTIONS]

1. Most Sold Product in the Best Performing Campaign (By Unit and Sale Price)
2. State with the Highest Total Sales
3. Highest Selling Products by State (Ranking)
4. List Customers Who Have Never Used a Discount
5. Update the Query to Count Total Distinct Customers Who Never Used a Discount
6. Average Frequency of Customer Repurchasing (Bucketed into Timeframes)

ANSWER SHEET:

[EASY Questions]

ORDERS TABLE:

First and Last Purchase Date for Each Customer: For each customer, find the date of their first and last purchase.

```
SELECT customer_id,
       MIN(sales_date) AS first_purchase,
       MAX(sales_date) AS last_purchase
FROM Orders
GROUP BY customer_id;
```

Select all the products in our catalog:

```
Select * from product
```

Find Specific Products: How would you find all products with a sales_price greater than $50?

```
SELECT * FROM Product WHERE sales_price > 50;
```

Count of Orders for Each Customer: Can you write a query to count the number of orders placed by each customer?

```
Select customer_id, count(distinct order_id) from Orders group by 1
```

what is the total number of unique customers that have purchased from us?

```
select count(distinct customer_id) as total_customers from orders
```

what is the total number of unique customers by product?

```
select product_id, count(distinct customer_id) as total_customers from orders group by 1
```

Summarize Sales by Product: Write a query to calculate the total sales (total_sales) for each product.

```
SELECT product_id, SUM(total_sales) as total_sales FROM Orders GROUP BY product_id;
```

Now summarize sales by month:

```
SELECT date_trunc('month', sales_date) as month, product_id, SUM(total_sales) as total_sales
FROM Orders GROUP BY 1,2;
```

Find Orders Within a Date Range: How can you find all orders placed between two specific dates?

```
SELECT * FROM Orders WHERE sales_date BETWEEN '2022-01-01' AND '2022-12-31';
```

Customers with No Orders: Write a query to find customers who have not placed any orders.

```
SELECT * FROM Customer WHERE customer_id NOT IN (SELECT customer_id FROM
Orders);
```

Total Sales Per Customer: Write a query to find the total sales for each customer.

```
SELECT customer_id, SUM(total_sales) AS total_customer_sales
FROM Orders
GROUP BY customer_id;
```

Average Sales Price of Products in Each Category

```
SELECT category_id, AVG(sales_price) AS avg_sales_price
FROM Orders
GROUP BY category_id;
```

Find the top 5 product with the highest total sales.

```
SELECT product_id, SUM(total_sales) AS total_sales
FROM Orders
GROUP BY 1
ORDER BY total_sales DESC
LIMIT 5;
```

RETURNS TABLE:

Find the total number of returns made by each customer.

```
SELECT customer_id, COUNT(*) AS total_returns
FROM Returns
GROUP BY customer_id;
```

CLICKS TABLE:

```
select date_trunc('month', click_date) as month, count(distinct click_id) as total_clicks
from clicks
group by 1
order by total_clicks desc;
```

```
select device_type, count(distinct click_id) as total_clicks
from clicks
group by 1
```

```
order by total_clicks desc
```

```
select campaign_id, count(distinct clicks) as total_clicks
from
clicks
where campaign_id is not null
group by 1
order by 2 desc
```

What is the conversion rate by device type? What is the clicks and orders broken out by device type?

```
SELECT c.device_type,
       COUNT(DISTINCT c.click_id) AS total_clicks,
       COUNT(DISTINCT o.order_id) AS total_orders,
       (COUNT(DISTINCT o.order_id) * 100.0 / COUNT(DISTINCT c.click_id)) AS
conversion_rate
FROM Clicks c
LEFT JOIN Orders o ON c.order_id = o.order_id
GROUP BY c.device_type;
```

[MEDIUM Questions]

ORDERS TABLE:

What is the highest total sale for an order? & how many total orders have been placed at the highest price?

```
with orders as (
select order_id,
sum(total_sales) as total_sales
from orders
group by 1
order by total_sales desc
)

select
count(distinct order_id) as total_orders
from orders
where total_sales = (select max(total_sales) from orders)
```

Find Customers Who Have Ordered Every Product: Identify customers who have ordered every single product at least once.

```
select
customer_id,
count(distinct product_id) as total_products
from orders
group by 1
having
count(distinct product_id)  = (select count(distinct product_id) from product)
```

Get the total lifetime sales of every customer, total order count and average order value (sales / orders)

```
SELECT
c.customer_id,
c.name,
SUM(o.total_sales) AS lifetime_sales,
COUNT(o.order_id) AS order_count,
SUM(o.total_sales) / COUNT(o.order_id) AS avg_order_value
FROM Customer c
JOIN Orders o ON c.customer_id = o.customer_id
GROUP BY c.customer_id;
```

What was the highest number of orders a customer placed and who was the customer?

```
select customer_id, total_orders
from (
select customer_id, count(distinct order_id) as total_orders
FROM Orders
group by 1
having count(distinct order_id) > 1
)
order by total_orders desc
limit 1
```

What did this customer purchase?

What is the highest total returns a customer made?

```
select
customer_id,
total_returns
from (
SELECT customer_id, COUNT(*) AS total_returns
FROM Returns
GROUP BY customer_id
)
order by total_returns desc
limit 1;
```

What are the top 5 reasons someone returned a package?

```
SELECT concession_reason, count(distinct return_id) as total_returns
FROM RETURNS
group by 1
order by total_returns desc
Limit 5;
```

RETURNS TABLE:

What was the total amount of sales price returned from orders alone?

```
SELECT sum(total_sales) as total_returns
FROM RETURNS a
join orders b
on a.order_id = b.order_id
```

What was the break out amount of returned orders by product? please also include the product name

```
SELECT
a.product_id,
c.product_name,
sum(total_sales) as total_returns
FROM RETURNS a
join orders b
on a.order_id = b.order_id
join product c
on a.product_id = c.product_id
group by 1,2
order by total_returns desc;
```

What is the percent of total orders returned (round to two decimal places)?

```
select
total_returned_sales,
total_sales,
round(total_returned_sales/total_sales,2) as pct_sales_returned
from (
SELECT
sum(case when b.order_id is not null then total_sales else 0 end) as total_returned_sales,
sum(a.total_sales) as total_sales
FROM orders a
left join returns b
on a.order_id = b.order_id
)
```

What is the percent of total orders returned by product (round to two decimal places)?

```
select
product_id,
product_name,
total_returned_sales,
total_sales,
round(total_returned_sales/total_sales,2) as pct_sales_returned
from (
SELECT
c.product_id,
c.product_name,
sum(case when b.order_id is not null then total_sales else 0 end) as total_returned_sales,
sum(a.total_sales) as total_sales
FROM orders a
left join returns b
on a.order_id = b.order_id
join product c
on a.product_id = c.product_id
group by 1,2
)
```

Which product was the highest returned product? Round to nearest 3 decimal places

```
select
product_id,
product_name,
total_returned_sales,
total_sales,
round(total_returned_sales/total_sales,3) as pct_sales_returned
from (
SELECT
c.product_id,
c.product_name,
sum(case when b.order_id is not null then total_sales else 0 end) as total_returned_sales,
sum(a.total_sales) as total_sales
FROM orders a
left join returns b
on a.order_id = b.order_id
join product c
on a.product_id = c.product_id
group by 1,2
)
order by pct_sales_returned desc;
```

CLICKS TABLE:

```
select
total_clicks,
total_orders,
round((total_orders :: numeric (38,6) /total_clicks :: numeric (38,6)),3)  as conversion
from (
select
count(distinct clicks) as total_clicks,
count(distinct order_id) as total_orders
from clicks
) a
```

Out of all the campaigns pull the total orders, total clicks, total sales, total units -

```
with click_data as (
select
campaign_id,
count(distinct click_id) as total_clicks
from clicks
group by 1
),

order_data as (
select
campaign_id,
sum(total_sales) as total_sales,
sum(units) as total_units,
count(distinct order_id) as total_orders
from orders
group by 1
)


select
a.campaign_id,
total_orders,
total_clicks,
total_sales,
total_units
from click_data a
join order_data b
on a.campaign_id = b.campaign_id
```

Which campaign gave us the highest conversion? orders/ clicks

```
with click_data as (
select
campaign_id,
count(distinct click_id) as total_clicks
from clicks
group by 1
),

order_data as (
select
campaign_id,
sum(total_sales) as total_sales,
sum(units) as total_units,
count(distinct order_id) as total_orders
from orders
group by 1
),

final as (
select
a.campaign_id,
total_orders,
total_clicks,
total_sales,
total_units
from click_data a
join order_data b
on a.campaign_id = b.campaign_id
)

select
campaign_id,
total_orders,
total_clicks,
round(total_orders :: numeric (30,6) /total_clicks :: numeric (30,6) ,3) as conversion
from final
order by conversion desc
```

DISCOUNT TABLE:

Which discount /promo generated the most sales?

```
with discount as (select discount_id, discount_description, discount_type from discount)

select
b.discount_id,
discount_description,
count(distinct order_id) as total_orders,
sum(sales_price) as total_sales,
sum(units) as total_units
from orders a
join discount b
on a.discount_id = b.discount_id
group by 1,2
order by total_sales desc
```

[HARD QUESTIONS]:

From the best performing campaign, what was the most sold product include product name - by unit and by sale price?

```
with click_data as (
select
campaign_id,
count(distinct click_id) as total_clicks
from clicks
group by 1
),

order_data as (
select
campaign_id,
sum(total_sales) as total_sales,
sum(units) as total_units,
count(distinct order_id) as total_orders
from orders
group by 1
),

final as (
select
a.campaign_id,
total_orders,
total_clicks,
total_sales,
total_units
from click_data a
join order_data b
on a.campaign_id = b.campaign_id
),

campaign_info as (
select
campaign_id,
total_orders,
total_clicks,
round(total_orders :: numeric (30,6) /total_clicks :: numeric (30,6) ,3) as conversion
from final
order by conversion desc
limit 1
)

select
a.product_id,
b.product_name,
sum(total_sales) as total_sales,
sum(units) as total_units
from orders a
join product b
on a.product_id = b.product_id

where campaign_id = (select campaign_id from campaign_info)
group by 1,2
order by total_sales desc
```

Which state do we sell the most products in? Rank these states.

```
with customer_data as (
select state, customer_id from customer
)

select
cd.state,
sum(total_sales) as total_sales,
row_number() over (order by sum(total_sales) desc) as rn
from customer_data cd join orders o
on o.customer_id = cd.customer_id
group by 1
```

By state, please rank the highest selling products

```
with customer_data as (
select state, customer_id from customer
)

select
cd.state,
product_id,
sum(total_sales) as total_sales,
row_number() over (partition by cd.state order by sum(total_sales) desc) as rn
from customer_data cd join orders o
on o.customer_id = cd.customer_id
group by 1,2
```

List all customers who have never used a discount for their orders.

```
with customers_no_disc as (
select distinct customer_id
from orders
where discount_id <> 'NULL'
)

select customer_id from orders where customer_id not in (select customer_id from
customers_no_disc)
```

Now update the query to count the total distinct customer ID

```
with customers_no_disc as (
select distinct customer_id
from orders
where discount_id <> 'NULL'
)

select
count(distinct customer_id) as total_customers
from (

select customer_id from orders where customer_id not in (select customer_id from
customers_no_disc)
)
```

What is the average frequency of each customer repurchasing again
Bucket into 30-60-90-180-365 as frequency

```
WITH OrderedSales AS (
  SELECT
    customer_id,
    sales_date,
    LAG(sales_date) OVER (PARTITION BY customer_id ORDER BY sales_date) AS
prev_sale_date
  FROM
    Orders
),
PurchaseDifferences AS (
  SELECT
    customer_id,
    sales_date,
    prev_sale_date,
    sales_date - prev_sale_date AS days_between_orders
  FROM
    OrderedSales
  WHERE
    prev_sale_date IS NOT NULL
),
FrequencyBuckets AS (
  SELECT
    customer_id,
    CASE
      WHEN days_between_orders <= 30 THEN '1. 1-30 days'
      WHEN days_between_orders <= 60 THEN '2. 31-60 days'
      WHEN days_between_orders <= 90 THEN '3. 61-90 days'
      WHEN days_between_orders <= 180 THEN '4. 91-180 days'
      WHEN days_between_orders <= 365 THEN '5. 181-365 days'
      ELSE '6. More than 365 days'
    END AS frequency_bucket,
    days_between_orders
  FROM
    PurchaseDifferences
)
SELECT
  frequency_bucket,
  round(AVG(days_between_orders),2) AS avg_days_between_orders,
  COUNT(distinct customer_id) AS total_customers
FROM
  FrequencyBuckets
GROUP BY
  frequency_bucket
ORDER BY
frequency_bucket
```