Basic Queries

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

```
select distinct(customer_city) from customers
order by customer_city;
```

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

```
select distinct(count(order_id)) from orders
where year(order_purchase_timestamp) = 2017;
```

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

```
select p.product_category, sum(o.price) as total_sales from order_items as o
join products as p
on o.product_id = p.product_id
group by p.product_category
order by total_sales desc;
```

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

```
select round( count(distinct case when payment_installments >1 then order_id end)*100
/count(distinct order_id) , 2) as Percentage_installments_orders from payments;
```

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

```
select customer_state, count(customer_id) as total_customers
from customers
group by customer_state;
```

Intermediate Queries

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

```
select extract(year from order_purchase_timestamp) as Year_extracted,
                    extract(month from order_purchase_timestamp) as month_extracted,
      count(customer_id) as Number_of_customers
from orders
where extract(year from order_purchase_timestamp) = 2018
group by Year_extracted, month_extracted
order by month_extracted;
```

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

```
with count_per_order as
(select o.order_id, o.customer_id, count(oi.order_id) as order_count
 from orders as o
 join order_items as oi
 on o.order_id= oi.order_id
 group by o.order_id, o.customer_id)

 select c.customer_city, round(avg(cp.order_count), 2) as avg_order
 from customers as c
 join count_per_order as cp
 on c.customer_id= cp.customer_id
 group by c.customer_city
 order by avg_order desc;
```

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

```
select p.product_category, round(sum(oi.price),2) as revenue,
round((sum(oi.price)/ (select sum(price) from order_items))*100, 2) as revenue_per_category
```

```
from products as p
join order_items as oi
on p.product_id= oi.product_id
group by p.product_category
order by revenue_per_category desc;
```

4. **Identify the correlation between product price and the number of times a product has been purchased.**

```
select p.product_category, count(oi.product_id) as count_of_products,
round(avg(oi.price), 2) as avg_price
from products as p
join order_items as oi
on p.product_id= oi.product_id
group by p.product_category;
```

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

```
select seller_id, total_sales, rank() over (order by total_sales) as ranks
from (
select s.seller_id, sum(oi.price) as total_sales
from sellers as s
join order_items as oi
on s.seller_id = oi.seller_id
group by s.seller_id) t
order by ranks;
```

Advanced Queries

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

```
with monthly_sales as
(select extract(year from o.order_purchase_timestamp) as sales_year,
                    extract(month from o.order_purchase_timestamp) as sales_month,
      round(sum(oi.price), 2) as Monthly_Sales
      from orders as o
      join order_items as oi
      on o.order_id = oi.order_id
      group by extract(year from o.order_purchase_timestamp), extract(month from
o.order_purchase_timestamp)
      order by extract(year from o.order_purchase_timestamp), extract(month from
o.order_purchase_timestamp))


      select sales_year, sales_month, Monthly_Sales,
      ROUND(SUM(monthly_sales) OVER (PARTITION BY sales_year ORDER BY sales_month
         ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW), 2) AS
cumulative_sales
FROM monthly_sales
ORDER BY sales_year, sales_month;
```

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

```
with year_sales as
(select extract(year from o.order_purchase_timestamp) as sales_year, round(sum(oi.price), 2) as
total_sales
from orders as o
      join order_items as oi
      on o.order_id = oi.order_id
      group by extract(year from o.order_purchase_timestamp)
      order by extract(year from o.order_purchase_timestamp))
          select sales_year as curr_year_sales, total_sales,
   LAG(total_sales) OVER (ORDER BY sales_year) AS previous_year_sales,
   round(
   (total_sales - LAG(total_sales) OVER (ORDER BY sales_year))
   /LAG(total_sales) OVER (ORDER BY sales_year)*100,
    2) as YoY_Sale_percentage
FROM year_sales
ORDER BY
    sales_year;
```

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

```
WITH customer_yearly_spend AS (
    SELECT
        YEAR(o.order_purchase_timestamp) AS sales_year,
        o.customer_id,
        ROUND(SUM(oi.price), 2) AS total_spent
    FROM orders o
    JOIN order_items oi
        ON o.order_id = oi.order_id
    GROUP BY
        YEAR(o.order_purchase_timestamp),
        o.customer_id
),
ranked_customers AS (
    SELECT sales_year, customer_id, total_spent,
        DENSE_RANK() OVER (
            PARTITION BY sales_year
            ORDER BY total_spent DESC
        ) AS spend_rank
    FROM customer_yearly_spend
)
SELECT
    sales_year, customer_id, total_spent
FROM ranked_customers
WHERE spend_rank <= 3
ORDER BY sales_year, spend_rank;
```