Subquery Tasks

1. Find the **highest-spending customer** in 2024.

```
SQL> SELECT c.customer_id, c.name, SUM(o.total_amount) AS total_spent
  2  FROM Customer c
  3  JOIN Order_Details o ON c.customer_id = o.customer_id
  4  WHERE EXTRACT(YEAR FROM o.order_date) = 2024
  5  GROUP BY c.customer_id, c.name
  6  ORDER BY total_spent DESC
  7  FETCH FIRST 1 ROWS ONLY;

CUSTOMER_ID
-----------
NAME
--------------------------------------------------------------------------------
TOTAL_SPENT
-----------
          2
Bob Smith
       55.2
```

2. Retrieve the **most ordered product** based on quantity.

```
SQL> SELECT p.product_id, p.name, SUM(oi.quantity) AS total_quantity
  2  FROM Product p
  3  JOIN Order_Item oi ON p.product_id = oi.product_id
  4  GROUP BY p.product_id, p.name
  5  ORDER BY total_quantity DESC
  6  FETCH FIRST 1 ROWS ONLY;

PRODUCT_ID
----------
NAME
--------------------------------------------------------------------------------
TOTAL_QUANTITY
--------------
         5
Apples
             5
```

3. Find employees who **earn more than the lowest-paid manager**.

```
SQL> SELECT * FROM Employee WHERE role = 'Manager';

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------------------
ROLE                                                SALARY HIRE_DATE
-------------------------------------------------- ---------- ---------
         1
Michael Scott
Manager                                              75000 10-MAY-20
```

```
SQL> SELECT MIN(salary) FROM Employee WHERE role = 'Manager';

MIN(SALARY)
-----------
      75000
```

4. Retrieve customers who **placed orders only in 2023 but not in 2024**.

```
SQL>
SQL> SELECT c.customer_id, c.name
  2  FROM Customer c
  3  JOIN Order_Details o ON c.customer_id = o.customer_id
  4  WHERE EXTRACT(YEAR FROM o.order_date) = 2023
  5  AND c.customer_id NOT IN (
  6      SELECT customer_id FROM Order_Details WHERE EXTRACT(YEAR FROM order_date) = 2024
  7  );

CUSTOMER_ID
-----------
NAME
--------------------------------------------------------------------------------
         4
David Miller

         3
Charlie Brown
```

5. Find the **total revenue generated in February 2024**.

```
SQL> SELECT SUM(total_amount) AS total_revenue
  2  FROM Order_Details
  3  WHERE EXTRACT(YEAR FROM order_date) = 2024
  4  AND EXTRACT(MONTH FROM order_date) = 2;

TOTAL_REVENUE
-------------
         25.5
```

Joins Tasks

1. Find the **top 3 customers** with the **highest total spending**.

```
SQL> SELECT c.customer_id, c.name, SUM(o.total_amount) AS total_spent
  2  FROM Customer c
  3  JOIN Order_Details o ON c.customer_id = o.customer_id
  4  GROUP BY c.customer_id, c.name
  5  ORDER BY total_spent DESC
  6  FETCH FIRST 3 ROWS ONLY;

CUSTOMER_ID
-----------
NAME
--------------------------------------------------------------------------------
TOTAL_SPENT
-----------
          2
Bob Smith
       55.2

          3
Charlie Brown
       40.8

CUSTOMER_ID
-----------
NAME
--------------------------------------------------------------------------------
TOTAL_SPENT
-----------

          4
David Miller
         30
```

2. Retrieve **employee names** along with the **total revenue generated from their assigned orders**.

```
SQL> SELECT e.employee_id, e.name, COALESCE(SUM(o.total_amount), 0) AS total_revenue
  2  FROM Employee e
  3  LEFT JOIN Order_Details o ON e.employee_id = o.processed_by
  4  GROUP BY e.employee_id, e.name
  5  ORDER BY total_revenue DESC;

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------------------
TOTAL_REVENUE
-------------
          1
Michael Scott
            0

          2
Jim Halpert
            0

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------------------
TOTAL_REVENUE
-------------

          3
Pam Beesly
            0

          5
Kevin Malone

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------------------
TOTAL_REVENUE
-------------
            0

          4
Dwight Schrute
            0
```

3. Show the **most ordered product category** and its total quantity sold. 4. Retrieve employees who **earn more than their colleagues** using a **SELF JOIN**. 5. Find employees who **work under the same manager** using a **SELF JOIN**.

```
SQL> SELECT p.category, SUM(oi.quantity) AS total_quantity
  2  FROM Product p
  3  JOIN Order_Item oi ON p.product_id = oi.product_id
  4  GROUP BY p.category
  5  ORDER BY total_quantity DESC
  6  FETCH FIRST 1 ROWS ONLY;

CATEGORY                                           TOTAL_QUANTITY
-------------------------------------------------- --------------
Fruit                                                           5
```

```
SQL>
SQL> SELECT e1.employee_id, e1.name, e1.salary
  2  FROM Employee e1
  3  JOIN Employee e2 ON e1.salary > e2.salary
  4  GROUP BY e1.employee_id, e1.name, e1.salary
  5  ORDER BY e1.salary DESC;

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------
    SALARY
----------
         1
Michael Scott
     75000

         4
Dwight Schrute
     50000

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------
    SALARY
----------

         2
Jim Halpert
     30000

         5
Kevin Malone

EMPLOYEE_ID
-----------
NAME
--------------------------------------------------------------------
    SALARY
----------
     29000
```