1. Display all customers whose email **is null** or **contains keyword** ‘fiu’

SELECT \* FROM Customers WHERE email IS NULL OR email LIKE '%fiu%';

2. Display the **number** of orders placed by all customers whose name **starts with** ‘J’.

SELECT Customers.name, COUNT(Orders.id) AS NumberOfOrders

FROM Customers

JOIN Orders ON Customers.id = Orders.customer\_id

WHERE Customers.name LIKE 'J%'

GROUP BY Customers.name;

3. Insert a new record (name = ‘John’, email = ‘abc@fiu.edu’) into the **Customers** table. Assume Customers.id will be automatically assigned by the database.

INSERT INTO Customers (name, email) VALUES ('John', 'abc@fiu.edu');

4. Create the Order\_Product table as follows:

* **id** is the primary key and its data type is **serial**
* **quantity** is an integer which **doesn’t allow null** and has a **default value of 0**
* **order\_id** and **product\_id** are foreign keys and will be **set to null on delete**.

CREATE TABLE Order\_Product (

id SERIAL PRIMARY KEY,

quantity INT NOT NULL DEFAULT 0,

order\_id INT,

product\_id INT,

FOREIGN KEY (order\_id) REFERENCES Orders(id) ON DELETE SET NULL,

FOREIGN KEY (product\_id) REFERENCES Products(id) ON DELETE SET NULL

);

5. Display the average price of all the products. For example, there are only three products in the database with a price of 7, 8 and 9. Then the average price would be 8.

SELECT AVG(price) AS average\_price FROM Products;

6. Display the calculated total amount for Order 103 (Orders.id = 103). How to calculate the total amount: for example, if an order includes 1 iPhone and 3 iPads, then the total for that order would be 1 \* iPhone\_price + 3 \* iPad\_price.

SELECT o.id AS order\_id, SUM(p.price \* op.quantity) AS total\_amount

FROM Orders o

JOIN Order\_Product op ON o.id = op.order\_id

JOIN Products p ON op.product\_id = p.id

WHERE o.id = 103

GROUP BY o.id;

7. Display all customers with the number of orders they placed. Note that customers with no purchase also need to be listed. An example output:

SELECT c.name AS customer\_name, COALESCE(COUNT(o.id), 0) AS number\_of\_orders

FROM Customers c

LEFT JOIN Orders o ON c.id = o.customer\_id

GROUP BY c.name;

8. Display all orders with their calculated total amount. For example, an order purchased 1 iPhone and 3 iPads, then the total for that order would be 1 \* iPhone\_price + 3 \* iPad\_price. An example output:

SELECT o.id AS order\_id, COALESCE(SUM(p.price \* op.quantity), 0) AS total\_amount

FROM Orders o

LEFT JOIN Order\_Product op ON o.id = op.order\_id

LEFT JOIN Products p ON op.product\_id = p.id

GROUP BY o.id

ORDER BY o.id;

9. Display the product with the most **total sales amount** (sale amount = price\*quantity, **total sales amount** = sum of sale amount in all the orders). For example, if iPhone12 (price = 10) was sold in two orders for amount of 3 and 1 respectively, then the total sales amount = 10 \* 3 + 10 \* 1 = 40. If there is a tie, display all of them. An example output (there is a tie in this example):

WITH ProductSales AS (

SELECT

p.id AS product\_id,

p.name AS product\_name,

SUM(p.price \* op.quantity) AS total\_sales\_amount

FROM Products p

JOIN Order\_Product op ON p.id = op.product\_id

GROUP BY p.id, p.name

)

SELECT

product\_name,

total\_sales\_amount

FROM ProductSales

WHERE total\_sales\_amount = (

SELECT MAX(total\_sales\_amount) FROM ProductSales

);

10. Display all categories and the most expensive product in them. If multiple products in the same category tie as the most expensive product in that category, display all of them (see iPhone 12 and iPhone 13 in the example below). Note that categories with no products also need to be listed, with null product columns. An example output:

WITH MaxPrices AS (

SELECT

c.id AS category\_id,

c.name AS category\_name,

MAX(p.price) AS max\_price

FROM Categories c

LEFT JOIN Products p ON c.id = p.category\_id

GROUP BY c.id, c.name

)

SELECT

mp.category\_name,

p.name AS most\_expensive\_product\_name,

mp.max\_price AS price

FROM MaxPrices mp

LEFT JOIN Products p ON mp.max\_price = p.price AND mp.category\_id = p.category\_id;