Project Deadline 4

Ten SQL Queries

Queries:

1. **TO FIND THE PRODUCT WITH THEIR CATEGORY HAVING AVG RATING 9**

```
SELECT
    P.productID,
    P.name AS product_name,
    PC.categoryID,
    C.category_name,
    AVG(R.review_rating) AS average_rating
FROM
    Product P
JOIN
    Product_Review PR ON P.productID = PR.productID
JOIN
    Review R ON PR.reviewID = R.reviewID
JOIN
    Product_And_Category PC ON P.productID = PC.productID
JOIN
    Category C ON PC.categoryID = C.categoryID
GROUP BY
    P.productID, P.name, PC.categoryID, C.category_name
HAVING
    AVG(R.review_rating) >= 9;
```

2. **QUERY TO DISPLAY THE ENTIRE RECORD OF EACH CUSTOMER PROPERLY**

```
SELECT
    userID AS customerID,
    CONCAT(first_name, ' ', COALESCE(middle_name, ''), ' ', last_name) AS name,
    CONCAT(house_number, ', ', street_name, ', ', city, ', ', state, ' - ', pincode) AS
customer_address,
    age,
    MAX(ph.phone_number) AS primary_phone_number, -- Assuming you want to display
just one phone number
    email_ID AS email
```

```
FROM
   Customer
JOIN
   phone_number_customers ph ON Customer.phoneID = ph.phoneID
GROUP BY
   customerID;
```

3. QUERY TO SHOW THE SHIPPED ORDERS OF DELIVERY AGENTS WITH THEIR NAMES

```
SELECT
   os.orderID,
   os.orderstatus,
   CONCAT(da.first_name, ' ', COALESCE(da.middle_name, ''), ' ', da.last_name) AS
delivery_agent_name
FROM
   OrderShipment os
JOIN
   Delivery_Agent da ON os.deliveryagentID = da.uniqueID
WHERE
   os.orderstatus = 'Shipped';
```

4. QUERY TO SHOW THE TOP 3 CUSTOMERS WITH THE HIGHEST TOTAL TRANSACTIONS AMOUNTS

```
SELECT
   c.userID,
   CONCAT(c.first_name, ' ', COALESCE(c.middle_name, ''), ' ', c.last_name) AS
customer_name, SUM(t.amount) AS total_transaction_amount
FROM Customer c
JOIN CustomersTransactions ct ON c.userID = ct.userID
JOIN Transactions t ON ct.transactionID = t.transactionID
GROUP BY c.userID
ORDER BY total_transaction_amount DESC LIMIT 3;
```

5. QUERY TO PRINT THE TRANSACTION HISTORY OF ALL CUSTOMER

```
SELECT
   c.userID,
   CONCAT(c.first_name, ' ', COALESCE(c.middle_name, ''), ' ', c.last_name) AS
customer_name,
   t.transactionID,
   t.date_of_transaction,
   t.amount,
   t.transactions_status,
   t.payment_name
```

```
FROM
   Customer c
JOIN
   CustomersTransactions ct ON c.userID = ct.userID
JOIN
   Transactions t ON ct.transactionID = t.transactionID
ORDER BY
   c.userID, t.date_of_transaction;
```

6. Query to update a customer's address with userID 1 UPDATE Customer

```
SET
   house_number = '372',
   street_name = 'Naya Ghr',
   city = 'Grand theft auto vice city',
   pincode = 110000,
   state = 'IDK'
WHERE
   userID = 1;

-- To Verify: SELECT * from customer where userID =1;
```

7. Query to delete a product from the customer's cart

```
DELETE FROM Cart_has_Products
WHERE cartID = 1 AND productID = 2;
SELECT * from cart where userID=1;
-- Query to update the total amount in the customer's cart
UPDATE Cart
SET total_amount = (
   SELECT SUM(Product.price * Cart_has_Products.product_quantity)
   FROM Cart_has_Products
   JOIN Product ON Cart_has_Products.productID = Product.productID
   WHERE Cart_has_Products.cartID = Cart.cartID
)
WHERE userID = 1;
```

8. QUERY TO ADD A NEW PHONE NUMBER FOR A CUSTOMER

```
INSERT INTO phone_number_customers (phoneID, phone_number)
VALUES (
   (SELECT phoneID FROM customer WHERE userID = 5),
   '1234567890'
);
-- TO Verfiy: SELECT * from phone_number_customers where phoneID=5;
```

9. QUERY TO FETCH CUSTOMERS WHO HAVE NOT PURCHASED ANY PRODUCT

```
SELECT *
FROM customer
WHERE userID NOT IN (
   SELECT DISTINCT userID
   FROM customerstransactions
);
```

10. QUERY TO LIST CUSTOMERS WHO HAVE PURCHASED ALL THE PRODUCTS OF CATEGORY: GAMING

Consoles

```
SELECT O.userID
FROM Orders as O
EXCEPT
SELECT NotAnswer.userID
FROM
(SELECT userID , categoryProduct.productID
FROM Customer CROSS JOIN
(SELECT productID
FROM Product_And_Category PC , Category C
WHERE PC.categoryID = C.categoryID AND c.category_name = 'Gaming Consoles')
categoryProduct
EXCEPT
SELECT O.userID , OP.productID
FROM Orders as O JOIN Order_has_Products OP ON O.orderID = OP.orderID) AS
NotAnswer;
```