Group No. 154
Siddharth - 2021424
Sunny Dhaka - 2021429

ONLINE RETAIL STORE

- Embedded Queries

Query 1:

```
SELECT S.shipper_name, S.delivery_speed FROM shipper S WHERE
S.Delivery_speed >= 2;
```

Query 2:

```
Select * From billing_details, order_table Where
billing_details.billing_id = order_table.billing_id AND
billing_details.billing_id = 1;
```

- OLAP queries

Query 1:

Query to calculate the total sales and profit of each brand and category combination, including subtotals for each brand and category and a grand total:

```
SELECT
    b.brand_name,
    c.category_name,
    SUM(p.product_cost * i.quantity) AS total_sales
FROM
    brand b
    JOIN product p ON b.brand_id = p.brand_id
    JOIN has h ON p.product_id = h.product_id
    JOIN category c ON h.category_id = c.category_id
    JOIN inventory i ON p.product_id = i.product_id
GROUP BY b.brand_name, c.category_name WITH ROLLUP
ORDER BY b.brand_name, c.category_name;
```

Query 2: used to retrieve sales data from the database by brand and month to generate subtotals and grand totals.

```
SELECT b.brand_name, MONTH(o.Date_Time) AS month,
SUM(p.product_cost * o.Quantity) AS total_sales
FROM brand b
JOIN product p ON b.brand_id = p.brand_id
JOIN order_table o ON p.product_id = o.Product_ID
GROUP BY b.brand_name, MONTH(o.Date_Time) WITH ROLLUP
ORDER BY b.brand_name, month;
```

Query 3:
This query retrieves the average product cost for each brand over the years with the help of the GROUP BY clause and ROLLUP.

```
SELECT
    b.brand_name,
    YEAR(o.date_time) AS year,
    AVG(p.product_cost) AS avg_product_cost
FROM
    brand b
    JOIN product p ON b.brand_id = p.brand_id
    JOIN order_table o ON p.product_id = o.product_id
GROUP BY b.brand_name, YEAR(o.date_time) with ROLLUP
ORDER BY
    b.brand_name, YEAR(o.date_time);
```

Query 4:

This query is selecting sales and profit data for the year 2022 broken down by month. It is joining the order_table, cart, product, and inventory tables to calculate the total sales and total profit for each product in each order.
It is filtering the results to only include orders placed between January 1, 2022, and December 31, 2022. It is grouping the results by year and month, and including a ROLLUP clause to add subtotals for each year and a grand total at the end.

```
SELECT
  YEAR(o.date_time) AS year,
  MONTHNAME(o.date_time) AS month,
  SUM(p.product_cost * oi.quantity) AS total_sales,
  SUM((p.product_cost - p.product_cost * 0.2) * oi.quantity)
AS total_profit
FROM
  order_table o
  JOIN cart ct ON o.unique_id = ct.unique_id
  JOIN product p ON ct.product_id = p.product_id
  JOIN inventory oi ON ct.product_id = oi.product_id
WHERE
  o.date_time BETWEEN '2022-01-01' AND '2022-12-31'
GROUP BY
  YEAR(o.date_time),
  MONTHNAME(o.date_time) WITH ROLLUP
ORDER BY
  YEAR(o.date_time),
  MONTHNAME(o.date_time)
LIMIT 0, 1000;
```

- Triggers

Trigger 1: To check before deleting a customer record whether they have any order or not

```
delimiter //
CREATE TRIGGER prevent_customer_deletion
BEFORE DELETE ON customer
FOR EACH ROW
BEGIN
  IF (SELECT COUNT(*) FROM order_table WHERE Unique_id = OLD.customer_id)
> 0 THEN
  -- unique_id in order_table references Customer_id in Customer table
    SIGNAL SQLSTATE '45000'
      SET MESSAGE_TEXT = 'Cannot delete customer because orders exist for this
customer.';
  END IF;
```

```
END;//
-- Query to check the working of this trigger

DELETE FROM customer WHERE customer_id = 98;
```

Trigger 2: To check if each entry of Customer has a Unique Phone number only

```
delimiter //
CREATE TRIGGER check_phone_number_unique
BEFORE INSERT ON customer
FOR EACH ROW
BEGIN
   IF (SELECT COUNT(*) FROM customer WHERE PhoneNumber =
NEW.PhoneNumber) > 0 THEN
      SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Phone number must be
unique.';
   END IF;
END;//
```

- A Query example to check whether the trigger is working or not

```
-- Insert a new customer with a unique phone number

INSERT INTO customer (customer_Id, Address, Name, EmailID, Password,
PhoneNumber)
VALUES (104, '123 Main St', 'John Doe', 'johndoe@example.com', 'password123',
'555-1234');

-- This should not cause an error

-- Insert a new customer with the same phone number as the first one

INSERT INTO customer (customer_Id, Address, Name, EmailID, Password,
PhoneNumber)
VALUES (105, '456 Maple St', 'Jane Smith', 'janesmith@example.com',
'password456', '555-1234');
```

```
-- This should raise an error due to the duplicate phone number
```