SQL Code

```sql
CREATE DATABASE RETAILDB_2;
USE RETAILDB_2;
CREATE TABLE Customersss (
    customer_id INT,
    name VARCHAR(100),
    email VARCHAR(100),
    city VARCHAR(50),
    signup_date DATE
);

CREATE TABLE Supplierss (
    supplier_id INT,
    supplier_name VARCHAR(100),
    contact_email VARCHAR(100),
    city VARCHAR(50)
);

CREATE TABLE Productsss (
    product_id INT,
    product_name VARCHAR(100),
    category VARCHAR(50),
    price DECIMAL(10,2),
    stock_qty INT,
    supplier_id INT
);

CREATE TABLE Orderrss (
    order_id INT,
    customer_id INT,
    order_date DATE,
    total_amount DECIMAL(10,2),
    payment_mode VARCHAR(50)
);

CREATE TABLE Order_Itemss (
    order_item_id INT,
    order_id INT,
    product_id INT,
    quantity INT,
    price_each DECIMAL(10,2)
);
SELECT * FROM Customersss;
SELECT * FROM Supplierss;
SELECT * FROM Productsss;
SELECT * FROM Orderrss;
SELECT * FROM Order_Itemss;

-- ==============================================================================================================================
-- == QUERIES



-- Q1. Fetch all products along with their supplier name
SELECT p.product_name, s.supplier_name
FROM Productsss p
INNER JOIN Supplierss s
ON p.supplier_id = s.supplier_id;

-- Q2. Find all customers and their orders (even if no order)
SELECT c.customer_id, c.name, o.order_id, o.order_date
FROM Customersss c
LEFT JOIN Orderrss o
ON c.customer_id = o.customer_id;

-- Q3. Get all suppliers and the products they supply (even if no product)
SELECT s.supplier_id, s.supplier_name, p.product_name
FROM Productsss p
RIGHT JOIN Supplierss s
ON p.supplier_id = s.supplier_id;

-- Q4. Show all customers and all orders (FULL OUTER JOIN simulation)
SELECT c.customer_id, c.name, o.order_id
FROM Customersss c
LEFT JOIN Orderrss o
ON c.customer_id = o.customer_id
UNION
SELECT c.customer_id, c.name, o.order_id
FROM Customersss c
RIGHT JOIN Orderrss o
ON c.customer_id = o.customer_id;

-- Q5. Products priced between 5000 and 50000 supplied from Mumbai
SELECT p.product_name, p.price
FROM Productsss p
JOIN Supplierss s
ON p.supplier_id = s.supplier_id
WHERE p.price BETWEEN 5000 AND 50000
AND s.city = 'Mumbai';

-- Q6. Customers who placed more than 2 orders
SELECT c.customer_id, c.name, COUNT(o.order_id) AS total_orders
FROM Customersss c
JOIN Orderrss o
ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name
HAVING COUNT(o.order_id) > 2;

-- Q7. Each supplier’s total sales value
SELECT s.supplier_id, s.supplier_name,
       SUM(oi.quantity * oi.price_each) AS total_sales
FROM Supplierss s
JOIN Productsss p ON s.supplier_id = p.supplier_id
JOIN Order_Itemss oi ON p.product_id = oi.product_id
GROUP BY s.supplier_id, s.supplier_name;

-- Q8. Average, highest, and lowest price of products in each category
```

```sql
SELECT category,
       AVG(price) AS avg_price,
       MAX(price) AS max_price,
       MIN(price) AS min_price
FROM Productsss
GROUP BY category;

-- Q9. Top 5 customers by total spending
SELECT c.customer_id, c.name,
       SUM(o.total_amount) AS total_spent
FROM Customersss c
JOIN Orderrss o
ON c.customer_id = o.customer_id
GROUP BY c.customer_id, c.name
ORDER BY total_spent DESC
LIMIT 5;

-- Q10. Number of unique products ordered by each customer
SELECT c.customer_id, c.name,
       COUNT(DISTINCT oi.product_id) AS unique_products
FROM Customersss c
JOIN Orderrss o ON c.customer_id = o.customer_id
JOIN Order_Itemss oi ON o.order_id = oi.order_id
GROUP BY c.customer_id, c.name;

-- Q11. Customers who placed an order greater than average order amount
SELECT *
FROM Orderrss
WHERE total_amount >
      (SELECT AVG(total_amount) FROM Orderrss);

-- Q12. Products that have never been ordered
SELECT product_name
FROM Productsss
WHERE product_id NOT IN
      (SELECT product_id FROM Order_Itemss);

-- Q13. Customers who ordered at least one Electronics product
SELECT DISTINCT c.customer_id, c.name
FROM Customersss c
JOIN Orderrss o ON c.customer_id = o.customer_id
JOIN Order_Itemss oi ON o.order_id = oi.order_id
JOIN Productsss p ON oi.product_id = p.product_id
WHERE p.category = 'Electronics';

-- Q14. Suppliers whose products have been ordered more than 100 times
SELECT s.supplier_id, s.supplier_name,
       SUM(oi.quantity) AS total_quantity
FROM Supplierss s
JOIN Productsss p ON s.supplier_id = p.supplier_id
JOIN Order_Itemss oi ON p.product_id = oi.product_id
GROUP BY s.supplier_id, s.supplier_name
HAVING SUM(oi.quantity) > 100;

-- Q15. Most expensive product(s)
```

```sql
SELECT product_name, price
FROM Productsss
WHERE price = (SELECT MAX(price) FROM Productsss);

-- Q16. Orders placed by customers from Mumbai, Delhi, or Bengaluru
SELECT o.*
FROM Orderrss o
JOIN Customersss c
ON o.customer_id = c.customer_id
WHERE c.city IN ('Mumbai', 'Delhi', 'Bengaluru');

-- Q17. Orders where payment mode is NOT UPI or Credit Card
SELECT *
FROM Orderrss
WHERE payment_mode NOT IN ('UPI', 'Credit Card');

-- Q18. Customers who have no email address
SELECT *
FROM Customersss
WHERE email IS NULL;

-- Q19. Suppliers not from the same city as any customer
SELECT *
FROM Supplierss
WHERE city NOT IN
      (SELECT DISTINCT city FROM Customersss);

-- Q20. Latest 3 orders, skipping first 2
SELECT *
FROM Orderrss
ORDER BY order_date DESC
LIMIT 3 OFFSET 2;
```