Assignment 1: Write a SELECT query to retrieve all columns from a 'customers' table, and modify it to return only the customer name and email address for customers in a specific city.

Retrieve All Columns:

SELECT * FROM customers;

Return Name and Email for Customers in a Specific City:

SELECT Name, Email

FROM customers

WHERE City = 'New York';

Assignment 2: Create a query using an INNER JOIN to combine 'orders' and 'customers' tables for customers in a specified region, and a LEFT JOIN to display all customers including those without orders.

SELECT c.customer_id, c.customer_name, o.order_id, o.order_date

FROM customers c

INNER JOIN orders o ON c.customer_id = o.customer_id

WHERE c.region = 'specified_region';

SELECT c.customer_id, c.customer_name, o.order_id, o.order_date

FROM customers c

LEFT JOIN orders o ON c.customer_id = o.customer_id;

Assignment 3: Utilize a subquery to find customers who have placed orders above the average order value, and write a UNION query to combine two SELECT statements with the same number of columns.

Utilize a Subquery to Find Customers with Orders Above Average Order Value:

```
SELECT customer_id, customer_name
FROM customers
WHERE customer_id IN (
   SELECT customer_id
   FROM orders
   GROUP BY customer_id
   HAVING AVG(order_value) > (
      SELECT AVG(order_value) FROM order ));
```

Write a UNION Query to Combine Two SELECT Statements:

```
SELECT column1, column2 FROM table1 UNION
SELECT column1, column2 FROM table2;
```

Assignment 4: Compose SQL statements to BEGIN a transaction, INSERT a new record into the 'orders' table, COMMIT the transaction, then UPDATE the 'products' table, and ROLLBACK the transaction.

```
START TRANSACTION;
INSERT INTO orders (order_id, customer_id, order_date, total_amount)
VALUES (1, 123, '2024-05-21', 100.00);
COMMIT;
UPDATE products SET stock_quantity = stock_quantity - 1
WHERE product_id = 1;

ROLLBACK;
```

Assignment 5: Begin a transaction, perform a series of INSERTs into 'orders', setting a SAVEPOINT after each, rollback to the second SAVEPOINT, and COMMIT the overall transaction.

```
START TRANSACTION;

INSERT INTO orders (order_id, customer_id, order_date, total_amount)
VALUES (1, 123, '2024-05-21', 100.00);
SAVEPOINT sp1;

INSERT INTO orders (order_id, customer_id, order_date, total_amount)
VALUES (2, 456, '2024-05-22', 150.00);
SAVEPOINT sp2;
INSERT INTO orders (order_id, customer_id, order_date, total_amount)
VALUES (3, 789, '2024-05-23', 200.00);
SAVEPOINT sp3;
ROLLBACK TO sp2;
COMMIT;
```

Assignment 6: Draft a brief report on the use of transaction logs for data recovery and create a hypothetical scenario where a transaction log is instrumental in data recovery after an unexpected shutdown.

Transaction logs are essential components of database management systems that record all changes made to a database during transactions. They serve as a vital tool for data recovery, ensuring that databases can be restored to a

consistent state in the event of unexpected shutdowns, system failures, or data corruption.

Purpose of Transaction Logs:

Transaction logs capture a chronological record of database modifications, including INSERTs, UPDATEs, and DELETEs, along with metadata such as timestamps and transaction IDs. These logs provide a detailed audit trail of database activity, enabling administrators to reconstruct transactions and restore data to a specific point in time.

Data Recovery Process:

In the event of an unexpected shutdown or system failure, transaction logs play a crucial role in data recovery. The recovery process typically involves replaying transaction logs to reapply committed transactions and roll back incomplete or uncommitted transactions. This process restores the database to a consistent state, minimizing data loss and ensuring data integrity.

Scenario:

Consider a hypothetical scenario where a retail company operates an online store with a customer database. During a routine maintenance operation, the database server experiences a sudden power outage, leading to an unexpected shutdown. As a result, the database becomes corrupted, and critical customer data is lost. In this scenario, transaction logs become instrumental in data recovery. The database administrators leverage transaction logs to restore the database to its state before the unexpected shutdown. They analyze the transaction logs to identify the last committed transactions and roll back any incomplete or uncommitted transactions.