# Coding Challenge-01 SQL

# Name- Ajay Chaudhary Batch- Data Engineering (Batch-1)

2. Execute all the join with examples.

Creating a table Customers and inserting demo values into it.

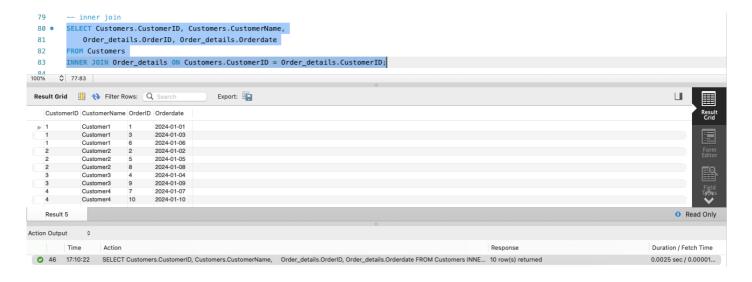
```
CREATE TABLE Customers (
             CustomerID INT PRIMARY KEY,
  48
             CustomerName VARCHAR(100),
  49
             Customercity VARCHAR(100)
  50
Action Output $
         Time
                                                                                                                                                            Duration / Fetch Time
 O 40 17:04:59 CREATE TABLE Customers ( CustomeriD INT PRIMARY KEY, CustomerName VARCHAR(100), Customercity VARCHAR(100))
                                                                                                                   0 row(s) affected
                                                                                                                                                            0.060 sec
         INSERT INTO Customers VALUES
 52 •
 53
            (1, 'Customer1', 'CityA'),
            (2, 'Customer2', 'CityB'),
  54
            (3, 'Customer3', 'CityA'),
  55
           (4, 'Customer4', 'CityC'),
 57
           (5, 'Customer5', 'CityB');
100% $ 1:58
Action Output $
         Time
41 17:05:32 INSERT INTO Customers VALUES (1, 'Customer1', 'CityA'), (2, 'Customer2', 'CityA'), (4, 'Customer4',... 5 row(s) affected Records: 5 Duplicates: 0 Warnings... 0.011 sec
```

Creating a table Order details and inserting values into it

```
59 • ⊝ CREATE TABLE Order_details (
  60
             OrderID INT PRIMARY KEY,
  61
             Orderdate DATE,
  62
             CustomerID INT,
             Orderamount DECIMAL(10,2),
  63
  64
             FOREIGN KEY (CustomerID) REFERENCES Customers(CustomerID)
  65
       $ 1:66
100%
Action Output
                                                                                                                                                                   Duration / Fetch Time
                                                                                                                         Response
O 44 17:07:59 CREATE TABLE Order_details ( OrderID INT PRIMARY KEY, Orderdate DATE, CustomerID INT, Orderamount DECIMAL(10,2),... 0 row(s) affected
                                                                                                                                                                   0.119 sec
          INSERT INTO Order_details VALUES
              (1, '2024-01-01', 1, 100.00),
              (2, '2024-01-02', 2, 150.50),
              (3, '2024-01-03', 1, 200.25),
              (4, '2024-01-04', 3, 120.75),
              (5, '2024-01-05', 2, 180.00),
              (6, '2024-01-06', 1, 220.50),
              (7, '2024-01-07', 4, 130.00),
  75
              (8, '2024-01-08', 2, 90.25),
              (9, '2024-01-09', 3, 110.75),
  76
  77
             (10, '2024-01-10', 4, 160.00);
Action Output
                                                                                                                                                                   Duration / Fetch Time
 45 17:08:28 INSERT INTO Order_details VALUES (1, '2024-01-01', 1, 100.00), (2, '2024-01-02', 2, 150.50), (3, '2024-01-03', 1, 200.25), (... 10 row(s) affected Records: 10 Duplicates: 0 Warnin...
```

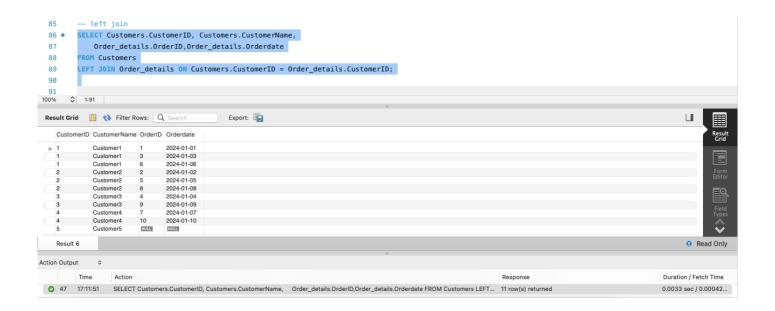
#### **Inner Join:**

- Returns only the rows where there is a match in both tables based on the specified condition.
- If there is no match, the row is excluded from the result set.



### Left (Outer) Join:

- Returns all rows from the left table and the matching rows from the right table.
- If there is no match in the right table, NULL values are returned for columns from the right table.



## Right (Outer) Join:

- Returns all rows from the right table and the matching rows from the left table.
- If there is no match in the left table, NULL values are returned for columns from the left table.



### Full (Outer) Join:

- Returns all rows when there is a match in either the left or right table.
- If there is no match, NULL values are returned for columns from the table without a match.



#### **Cross Join:**

A cross join returns the Cartesian product of two tables, combining every row from the first table with every row from the second table. In this case, it would create combinations of each customer with each order

