Coding Challenge-01
SQL

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

1. a). Execute OVER and PARTITION BY Clause in SQL Queries -

Creating the table Orders for executing OVER and PARTITION BY clause

```
Create Table Orders (
    orderid Int PRIMARY KEY,
    Orderdate DATE,
    CustomerName Varchar(100),
    Customercity Varchar(100),
    Orderamount Decimal(10,2)
);

-- inserting sample data in the orders table

insert into Orders (orderid,Orderdate,CustomerName,Customercity,Orderamount)
VALUES(1, '2024-01-01', 'Customer1', 'CityA', 100.00),
    (2, '2024-01-02', 'Customer2', 'CityB', 150.50),
    (3, '2024-01-03', 'Customer3', 'CityA', 200.25),
    (4, '2024-01-04', 'Customer4', 'CityC', 120.75),
    (5, '2024-01-05', 'Customer5', 'CityB', 180.00),
    (6, '2024-01-06', 'Customer6', 'CityA', 220.50),
    (7, '2024-01-07', 'Customer7', 'CityC', 130.00),
    (8, '2024-01-08', 'Customer8', 'CityB', 90.25),
```

100% 6:30

Action Output

	Time	Action	Response	Duration / Fetch Time
32	16:13:08	Create database CodingChallenge1	1 row(s) affected	0.010 sec
33	16:13:19	use CodingChallenge	0 row(s) affected	0.0030 sec
34	16:17:01	CREATE TABLE Orders (orderid INT PRIMARY KEY, Orderdate DATE, CustomerName VARCHAR(100), Customercity VARCHA...	0 row(s) affected	0.157 sec

Inserting 10 demo data in Orders table

```
insert into Orders (orderid,Orderdate,CustomerName,Customercity,Orderamount)
VALUES(1, '2024-01-01', 'Customer1', 'CityA', 100.00),
    (2, '2024-01-02', 'Customer2', 'CityB', 150.50),
    (3, '2024-01-03', 'Customer3', 'CityA', 200.25),
    (4, '2024-01-04', 'Customer4', 'CityC', 120.75),
    (5, '2024-01-05', 'Customer5', 'CityB', 180.00),
    (6, '2024-01-06', 'Customer6', 'CityA', 220.50),
    (7, '2024-01-07', 'Customer7', 'CityC', 130.00),
    (8, '2024-01-08', 'Customer8', 'CityB', 90.25),
    (9, '2024-01-09', 'Customer9', 'CityA', 110.75),
    (10, '2024-01-10', 'Customer10', 'CityC', 160.00);
```

100% 6:30

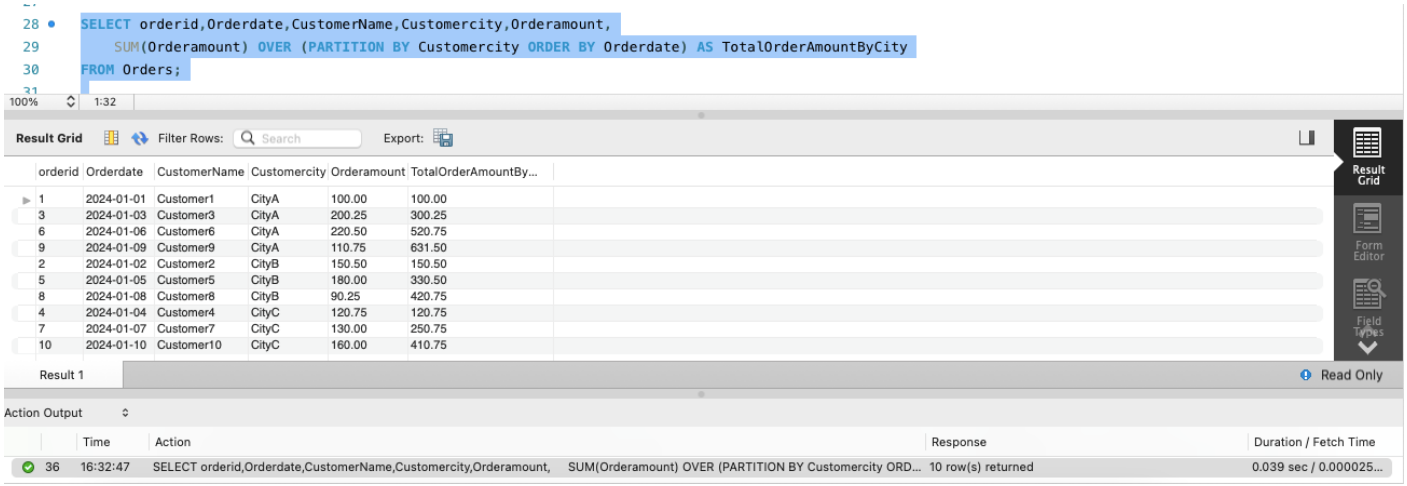
Action Output

	Time	Action	Response	Duration / Fetch Time
34	16:17:01	CREATE TABLE Orders (orderid INT PRIMARY KEY, Orderdate DATE, CustomerName VARCHAR(100), Customercity VARCHA...	0 row(s) affected	0.157 sec
35	16:24:03	insert into Orders (orderid,Orderdate,CustomerName,Customercity,Orderamount) VALUES(1, '2024-01-01', 'Customer1', 'CityA', 100.00...	10 row(s) affected Records: 10 Duplicates: 0 Warnin...	0.029 sec

Executing the query of OVER and PARTITON BY clause-

This query will calculate the sum of order amounts partitioned by city

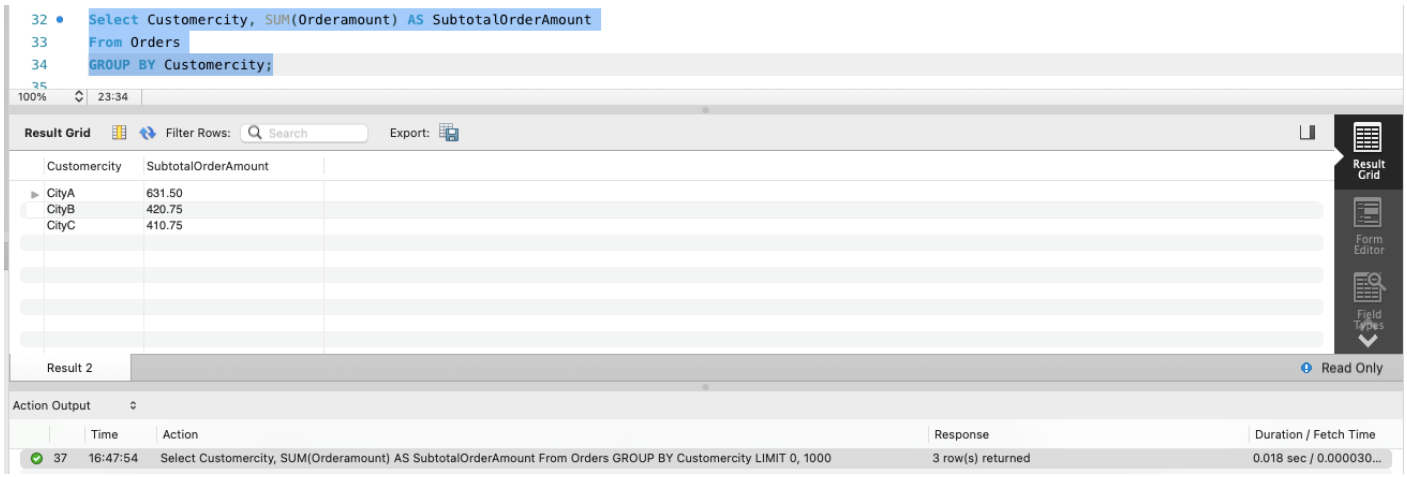
- SUM(Orderamount) calculates the cumulative sum of the Orderamount column.
- OVER is used to define the window frame for the window function.
- PARTITION BY Customercity indicates that the sum should be calculated separately for each unique value in the Customercity column.



1. b). Creating subtotals using SQL queries

On the above inserted table I have executed the subtotal query-

1. SELECT Customercity, SUM(Orderamount) AS SubtotalOrderAmount:
    - Customercity is one of the columns selected, representing the unique customer cities.
    - SUM(Orderamount) calculates the total order amount for each Customercity. The result of this calculation is given an alias SubtotalOrderAmount.
2. GROUP BY Customercity:
    - This clause is used to group the rows based on the values in the Customercity column. The SUM function then calculates the total order amount for each group i.e. each unique Customercity



1.c). Total Aggregations using SQL Queries-

1. SELECT Customercity, SUM(Orderamount) AS SubtotalOrderAmount:
    - Customercity is one of the columns selected.
    - SUM(Orderamount) calculates the subtotal of Orderamount for each Customercity.
2. GROUP BY Customercity WITH ROLLUP:
    - The GROUP BY clause is used to group the rows based on the values in the Customercity column.
    - The WITH ROLLUP modifier extends the grouping to include subtotals and a grand total.

```
36 •  SELECT Customercity, SUM(Orderamount) AS SubtotalOrderAmount
37    FROM Orders
38    GROUP BY Customercity
39    WITH ROLLUP;
```

100% 13:39

Result Grid Filter Rows: Search Export:

Customercity	SubtotalOrderAmount
CityA	631.50
CityB	420.75
CityC	410.75
NULL	1463.00

Result Grid Form Editor Field Types

Result 4 Read Only

Action Output

	Time	Action	Response	Duration / Fetch Time
39	16:56:55	SELECT Customercity, SUM(Orderamount) AS SubtotalOrderAmount FROM Orders GROUP BY Customercity WITH ROLLUP LIMIT 0, 1...	4 row(s) returned	0.021 sec / 0.000015...