

IMPORTANT SQL INTERVIEW QUESTIONS

Order of execution in an SQL query:-

- 1. FROM /Join**
- 2. WHERE**
- 3. GROUP BY**
- 4. HAVING**
- 5. SELECT**
- 6. DISTINCT**
- 7. ORDER BY**
- 8. LIMIT / OFFSET**

Question 1: Write a SQL query to calculate the cumulative sum of sales for each employee. The query should return the EmployeeID, SalesDate, and CumulativeSales columns, with the final output ordered by EmployeeID.

EmployeeID	SalesDate	SalesAmount
101	2024-08-01	1000
102	2024-08-01	1500
101	2024-08-02	2000
103	2024-08-02	2500
101	2024-08-03	3000

Solution -

SELECT

EmployeeID,

SalesDate,

SUM(SalesAmount) OVER (PARTITION BY EmployeeID

ORDER BY SalesDate) AS CumulativeSales

FROM Employee
ORDER BY
EmployeeID;

Solution Explanation by Execution steps:-

Step 1 -> **FROM - Employee Table**

EmployeeID	SalesDate	SalesAmount
101	2024-08-01	1000
102	2024-08-01	1500
101	2024-08-02	2000
103	2024-08-02	2500
101	2024-08-03	3000

Step 2 -> **PARTITION BY EmployeeID and ORDER BY SalesDate**

- Partition for EmployeeID = 101 (Ordered by SalesDate):

EmployeeID	SalesDate	SalesAmount
101	2024-08-01	1000
101	2024-08-02	2000
101	2024-08-03	3000

- Partition for EmployeeID = 102 (Ordered by SalesDate):

EmployeeID	SalesDate	SalesAmount
102	2024-08-01	1500

- Partition for EmployeeID = 103 (Ordered by SalesDate):

EmployeeID	SalesDate	SalesAmount
103	2024-08-02	2500

Step 3: Cumulative Sum Calculation (SUM() OVER) As New Column (CumulativeSales)-

- Partition for EmployeeID = 101:

EmployeeID	SalesDate	SalesAmount	CumulativeSales
101	2024-08-01	1000	1000
101	2024-08-02	2000	3000
101	2024-08-03	3000	6000

- Partition for EmployeeID = 102:

EmployeeID	SalesDate	SalesAmount	CumulativeSales
102	2024-08-01	1500	1500

- Partition for EmployeeID = 103:

EmployeeID	SalesDate	SalesAmount	CumulativeSales
103	2024-08-02	2500	2500

Final Output Ordered by EmployeeID and By Only Selecting Required 3 columns (EmployeeID, SalesDate, CumulativeSales):-

EmployeeID	SalesDate	CumulativeSales
101	2024-08-01	1000
101	2024-08-02	3000
101	2024-08-03	6000
102	2024-08-01	1500
103	2024-08-02	2500

Question 2: Write a SQL query to find employees who have a salary greater than their manager's salary from the **Employee** table.

EmployeeID	EmployeeName	Salary	ManagerID
1	Arjun	70000	5
2	Bharat	60000	5
3	Chetan	90000	4
4	Dinesh	80000	NULL
5	Esha	75000	4

Solution -

SELECT

e1.EmployeeID AS e1_EmployeeID,
e1.EmployeeName AS e1_EmployeeName,
e1.Salary AS e1_Salary

FROM

Employee e1

JOIN

Employee e2

ON

e1.ManagerID = e2.EmployeeID

WHERE

e1.Salary > e2.Salary;

Solution Explanation by Execution steps:-

Step 1 -> FROM - Employee Table

EmployeeID	EmployeeName	Salary	ManagerID
1	Arjun	70000	5
2	Bharat	60000	5
3	Chetan	90000	4
4	Dinesh	80000	NULL
5	Esha	75000	4

Step 2 -> JOIN Clause - e1.ManagerID = e2.EmployeeID

The **JOIN** clause links the **Employee** table (as **e1**) with itself (as **e2**) based on the **ManagerID**. This means for each employee in **e1**, we find the corresponding manager in **e2**.

e1_EmployeeID	e1_EmployeeName	e1_Salary	e1_ManagerID	e2_EmployeeID	e2_EmployeeName	e2_Salary	e2_ManagerID
5	Esha	75000	4	4	Dinesh	80000	NULL
3	Chetan	90000	4	4	Dinesh	80000	NULL
2	Bharat	60000	5	5	Esha	75000	4
1	Arjun	70000	5	5	Esha	75000	4

Step 3: WHERE Clause - e1.Salary > e2.Salary

The **WHERE** clause filters the rows where the employee's salary (**e1.Salary**) is greater than their manager's salary (**e2.Salary**).

e1_EmployeeID	e1_EmployeeName	e1_Salary	e1_ManagerID	e2_EmployeeID	e2_EmployeeName	e2_Salary	e2_ManagerID
3	Chetan	90000	4	4	Dinesh	80000	NULL

Step 4: SELECT Clause - SELECT e1.EmployeeID AS e1_EmployeeID, e1.EmployeeName AS e1_EmployeeName, e1.Salary AS e1_Salary

The **SELECT** clause retrieves the **EmployeeID**, **EmployeeName**, and **Salary** columns for employees who meet the condition.

Final Output:-

e1_EmployeeID	e1_EmployeeName	e1_Salary
3	Chetan	90000

Question 3: Given a table **Employees**, write a query to find the third highest salary.

EmployeeID	Name	Salary
1	Rahul	6000
2	Priya	7000
3	Ankit	8000
4	Sneha	9000
5	Ajay	9000
6	Riya	5000

Solution -

```
WITH SalaryRank AS (  
    SELECT Salary,  
           DENSE_RANK() OVER (ORDER BY Salary DESC) AS SalaryRank  
    FROM Employees  
)  
SELECT Salary  
FROM SalaryRank  
WHERE SalaryRank = 3;
```

Question 4: Given a table **Purchases**, write a query to find employees who bought a product for at least 3 consecutive days.

EmployeeID	PurchaseDate
1	2024-08-01
1	2024-08-02
1	2024-08-03
2	2024-08-01
2	2024-08-03
3	2024-08-02
3	2024-08-03
3	2024-08-04
4	2024-08-02

Solution -

```
SELECT DISTINCT p1.EmployeeID
FROM Purchases p1
JOIN Purchases p2 ON p1.EmployeeID = p2.EmployeeID
  AND DATE_ADD(p1.PurchaseDate, INTERVAL 1 DAY) = p2.PurchaseDate
JOIN Purchases p3 ON p2.EmployeeID = p3.EmployeeID
  AND DATE_ADD(p2.PurchaseDate, INTERVAL 1 DAY) = p3.PurchaseDate;
```