**Question 1: Dynamic Column Aliasing**

You have a table named SalesData with the following columns:

* SalesID (INT, Primary Key)
* ProductName (VARCHAR(100))
* SalesAmount (DECIMAL(10,2))
* SalesDate (DATE)

Write an SQL query that dynamically renames the SalesAmount column based on the current month. For example, if the current month is March, the column should be aliased as Sales\_March.

Declare @MonthName nvarchar(20);

Set @MonthName=DATENAME(MONTH,GETDATE());

Declare @Query nvarchar(MAX);

Set @Query= ‘Select SalesId, ProductName SalesAmount as Sales\_’+@MonthName+’, SalesDate from SalesData;’

Exec Sp\_executesql @Query;

**Question 2: Pivoting Data for Custom Report**

Given a table EmployeeSalary:

* EmployeeID (INT, Primary Key)
* EmployeeName (VARCHAR(100))
* SalaryMonth (VARCHAR(10)) -- E.g., 'January', 'February'
* SalaryAmount (DECIMAL(10,2))

Write an SQL query to pivot the data so that the result displays one row per employee with separate columns for each month's salary (e.g., January\_Salary, February\_Salary, etc.).

Select \* from ( select EmployeeID, EmployeeName, SalaryMonth, SalaryAmount from EmployeeSalary) as Source pivot ( sum(SalaryAmount) for SalaryMonth in ([January], [February], [March], [April], [May], [June], [July], [August], [September], [October], [November], [December]) as PivotTable;

**Question 3: Ranking and Filtering Data**

Given a table CustomerTransactions:

* TransactionID (INT, Primary Key)
* CustomerID (INT, Foreign Key to Customers)
* TransactionAmount (DECIMAL(10,2))
* TransactionDate (DATE)

Write an SQL query to return the top 3 transactions per customer, ranked by TransactionAmount in descending order. If two transactions have the same amount, order them by TransactionDate in descending order.

WITH RankedTransactions AS (

SELECT

TransactionID,

CustomerID,

TransactionAmount,

TransactionDate,

ROW\_NUMBER() OVER (

PARTITION BY CustomerID

ORDER BY TransactionAmount DESC, TransactionDate DESC

) AS Rank

FROM CustomerTransactions

)

SELECT

TransactionID,

CustomerID,

TransactionAmount,

TransactionDate

FROM RankedTransactions

WHERE Rank <= 3

ORDER BY CustomerID, Rank;

**Question 4: Conditional Aggregation**

Given a table Orders:

* OrderID (INT, Primary Key)
* CustomerID (INT, Foreign Key to Customers)
* OrderAmount (DECIMAL(10,2))
* OrderDate (DATE)

Write an SQL query to return a summary of total OrderAmount per customer. Additionally, add separate columns for TotalOrdersThisYear and TotalOrdersLastYear.

SELECT

CustomerID,

SUM(OrderAmount) AS TotalOrderAmount,

COUNT(CASE WHEN YEAR(OrderDate) = YEAR(GETDATE()) THEN OrderID END) AS TotalOrdersThisYear,

COUNT(CASE WHEN YEAR(OrderDate) = YEAR(GETDATE()) - 1 THEN OrderID END) AS TotalOrdersLastYear

FROM Orders

GROUP BY CustomerID

ORDER BY CustomerID;

**Question 5: Generating a Custom Ranking Report**

Given a table StudentScores:

* StudentID (INT, Primary Key)
* StudentName (VARCHAR(100))
* Subject (VARCHAR(50))
* Score (INT)

Write an SQL query that assigns a rank to each student per subject based on their Score. If two students have the same score, they should receive the same rank, and the next rank should be adjusted accordingly (e.g., using DENSE\_RANK()).

SELECT

StudentID,

StudentName,

Subject,

Score,

DENSE\_RANK() OVER (

PARTITION BY Subject

ORDER BY Score DESC

) AS Rank

FROM StudentScores

ORDER BY Subject, Rank;