Advanced SQL Techniques for Sales Reporting

Introduction

Generating sales reports efficiently is crucial for businesses. In this article, we will explore different SQL techniques to compute total sales for Sales Support Agents, grouped by year and quarter. We will cover solutions ranging from beginner-friendly approaches to expert-level optimizations using Common Table Expressions (CTEs), window functions, and PIVOT tables.

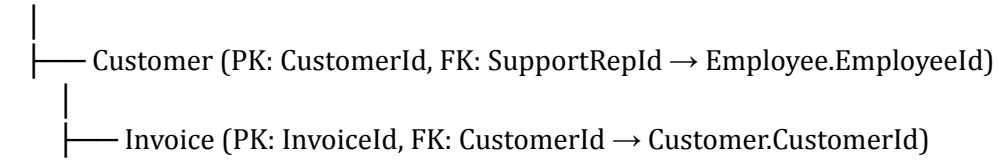
Problem Statement

We need to provide a report displaying the total sales for all Sales Support Agents, grouped by year and quarter, for the period between January 2010 and June 2012. The sales quarters are structured as follows:

- Q1: January - March ('First')
- Q2: April - June ('Second')
- Q3: July - September ('Third')
- Q4: October - December ('Fourth')

The data should be ordered by employee name, year, and quarter.

Entity-Relationship Diagram (ERD)



Solution 1: Beginner-Friendly Approach (Using GROUP BY and CASE)

```
SELECT
    e.FirstName + ' ' + e.LastName AS EmployeeName,
    YEAR(i.InvoiceDate) AS SalesYear,
    CASE
        WHEN DATEPART(QUARTER, i.InvoiceDate) = 1 THEN 'First'
        WHEN DATEPART(QUARTER, i.InvoiceDate) = 2 THEN 'Second'
        WHEN DATEPART(QUARTER, i.InvoiceDate) = 3 THEN 'Third'
        WHEN DATEPART(QUARTER, i.InvoiceDate) = 4 THEN 'Fourth'
    END AS SalesQuarter,
    SUM(i.Total) AS TotalSales
FROM Employee e
JOIN Customer c ON e.EmployeeId = c.SupportRepId
JOIN Invoice i ON c.CustomerId = i.CustomerId
WHERE e.Title = 'Sales Support Agent'
```

```
AND i.InvoiceDate BETWEEN '2010-01-01' AND '2012-06-30'
GROUP BY e.FirstName, e.LastName, YEAR(i.InvoiceDate), DATEPART(QUARTER,
i.InvoiceDate)
ORDER BY e.FirstName, e.LastName, YEAR(i.InvoiceDate), DATEPART(QUARTER,
i.InvoiceDate);
```

Solution 2: Intermediate Approach (Using CTE for Readability)

```
WITH SalesData AS (
    SELECT
        e.FirstName + ' ' + e.LastName AS EmployeeName,
        YEAR(i.InvoiceDate) AS SalesYear,
        DATEPART(QUARTER, i.InvoiceDate) AS SalesQuarterNumber,
        SUM(i.Total) AS TotalSales
    FROM Employee e
    JOIN Customer c ON e.EmployeeId = c.SupportRepId
    JOIN Invoice i ON c.CustomerId = i.CustomerId
    WHERE e.Title = 'Sales Support Agent'
    AND i.InvoiceDate BETWEEN '2010-01-01' AND '2012-06-30'
    GROUP BY e.FirstName, e.LastName, YEAR(i.InvoiceDate),
DATEPART(QUARTER, i.InvoiceDate)
)
SELECT
    EmployeeName,
    SalesYear,
    CASE
        WHEN SalesQuarterNumber = 1 THEN 'First'
        WHEN SalesQuarterNumber = 2 THEN 'Second'
        WHEN SalesQuarterNumber = 3 THEN 'Third'
        WHEN SalesQuarterNumber = 4 THEN 'Fourth'
    END AS SalesQuarter,
    TotalSales
FROM SalesData
ORDER BY EmployeeName, SalesYear, SalesQuarterNumber;
```

Solution 3: Expert-Level Approach (Using STRING_AGG for Compact Output)

```
WITH SalesData AS (
    SELECT
        e.FirstName + ' ' + e.LastName AS EmployeeName,
        YEAR(i.InvoiceDate) AS SalesYear,
        DATEPART(QUARTER, i.InvoiceDate) AS SalesQuarterNumber,
        SUM(i.Total) AS TotalSales
    FROM Employee e
    JOIN Customer c ON e.EmployeeId = c.SupportRepId
    JOIN Invoice i ON c.CustomerId = i.CustomerId
    WHERE e.Title = 'Sales Support Agent'
    AND i.InvoiceDate BETWEEN '2010-01-01' AND '2012-06-30'
    GROUP BY e.FirstName, e.LastName, YEAR(i.InvoiceDate),
```

```
DATEPART(QUARTER, i.InvoiceDate)
)
SELECT
    EmployeeName,
    SalesYear,
    STRING_AGG(
        CONCAT(
            CASE
                WHEN SalesQuarterNumber = 1 THEN 'First'
                WHEN SalesQuarterNumber = 2 THEN 'Second'
                WHEN SalesQuarterNumber = 3 THEN 'Third'
                WHEN SalesQuarterNumber = 4 THEN 'Fourth'
            END, ': $', FORMAT(TotalSales, 'N2')
        ), ', '
    ) AS QuarterlySalesSummary
FROM SalesData
GROUP BY EmployeeName, SalesYear
ORDER BY EmployeeName, SalesYear;
```

Solution 4: Ultimate Expert Approach (Using PIVOT for Table Format)

```
WITH SalesData AS (
    SELECT
        e.FirstName + ' ' + e.LastName AS EmployeeName,
        YEAR(i.InvoiceDate) AS SalesYear,
        DATEPART(QUARTER, i.InvoiceDate) AS SalesQuarterNumber,
        SUM(i.Total) AS TotalSales
    FROM Employee e
    JOIN Customer c ON e.EmployeeId = c.SupportRepId
    JOIN Invoice i ON c.CustomerId = i.CustomerId
    WHERE e.Title = 'Sales Support Agent'
    AND i.InvoiceDate BETWEEN '2010-01-01' AND '2012-06-30'
    GROUP BY e.FirstName, e.LastName, YEAR(i.InvoiceDate),
DATEPART(QUARTER, i.InvoiceDate)
)
SELECT * FROM (
    SELECT EmployeeName, SalesYear, SalesQuarterNumber, TotalSales
    FROM SalesData
) AS SourceTable
PIVOT (
    SUM(TotalSales) FOR SalesQuarterNumber IN ([1], [2], [3], [4])
) AS PivotTable
ORDER BY EmployeeName, SalesYear;
```

Conclusion

We explored multiple ways to generate a quarterly sales report, from basic SQL GROUP BY to advanced PIVOT solutions. Whether you're a beginner or an expert, these approaches ensure efficiency, clarity, and performance.