SQL JOINs & Window Functions Assignment

Business Scenario Example

Step 1: Problem Definition

Business Context

A retail e-commerce company operating in multiple regions wants to analyze sales performance.
Department: Sales & Business Intelligence
Industry: Retail / E-commerce

Data Challenge

The company stores customer, product, and transaction data in separate tables. Management struggles to identify top-performing products, inactive customers, and sales trends over time using simple queries.

Expected Outcome

Use SQL JOINs and Window Functions to analyze customer behavior, product performance, and monthly sales trends to support data-driven marketing and inventory decisions.

Step 2: Success Criteria

1. Identify Top 5 products per region → RANK()
2. Compute running monthly sales totals → SUM() OVER()
3. Measure month-over-month sales growth → LAG()
4. Segment customers into quartiles based on spending → NTILE(4)
5. Calculate 3-month moving average of sales → AVG() OVER()

Step 3: Database Schema Design

Tables

customers

customer_id (PK)

customer_name

region

signup_date

products

product_id (PK)

product_name

category

price

sales

sale_id (PK)

customer_id (FK)

product_id (FK)

sale_date

quantity

total_amount

Relationships

- customers 1 —— * sales
- products 1 —— * sales

Step 4: Part A — SQL JOINs

1. INNER JOIN

```
-- Retrieve all valid sales with customer and product details
SELECT c.customer_name, p.product_name, s.total_amount
FROM sales s
INNER JOIN customers c ON s.customer_id = c.customer_id
INNER JOIN products p ON s.product_id = p.product_id;
```

Business Interpretation:
Shows only completed transactions with valid customers and products, ensuring reliable revenue analysis.

2. LEFT JOIN

```
-- Customers who never made a purchase
```

```sql
SELECT c.customer_name, s.sale_id

FROM customers c

LEFT JOIN sales s ON c.customer_id = s.customer_id

WHERE s.sale_id IS NULL;
```

Interpretation:
Identifies inactive customers for re-engagement campaigns.

3. RIGHT JOIN

```sql
-- Products with no sales

SELECT p.product_name, s.sale_id

FROM sales s

RIGHT JOIN products p ON s.product_id = p.product_id

WHERE s.sale_id IS NULL;
```

Interpretation:
Helps detect underperforming or obsolete product

4. FULL OUTER JOIN

```sql
-- Compare customers and products including unmatched records

SELECT c.customer_name, p.product_name

FROM customers c

FULL OUTER JOIN products p

ON c.region = p.category;
```

Interpretation:
Reveals mismatches and unused data across dimensions.

5. SELF JOIN

```sql
-- Customers from the same region

SELECT c1.customer_name, c2.customer_name, c1.region

FROM customers c1

JOIN customers c2
```

```
ON c1.region = c2.region

AND c1.customer_id <> c2.customer_id;
```

Interpretation:
Useful for regional segmentation and peer comparison.

Step 5: Part B — Window Functions

1. Ranking Functions

```
SELECT region, product_id,

    SUM(total_amount) AS revenue,

    RANK() OVER (PARTITION BY region ORDER BY SUM(total_amount) DESC) AS rank_in_region

FROM sales s

JOIN customers c ON s.customer_id = c.customer_id

GROUP BY region, product_id;
```

Interpretation:
Ranks products by revenue within each region.

2. Aggregate Window Functions

```
SELECT sale_date,

    SUM(total_amount) OVER (

      ORDER BY sale_date

      ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

    ) AS running_total

FROM sales;
```

Interpretation:
Shows cumulative sales growth over time.

3. Navigation Functions

```
SELECT sale_date,

    SUM(total_amount) AS monthly_sales,

    LAG(SUM(total_amount)) OVER (ORDER BY sale_date) AS previous_month
```

```
FROM sales

GROUP BY sale_date;
```

Interpretation:
Allows month-to-month sales comparison.

4. Distribution Functions

```
SELECT customer_id,

       SUM(total_amount) AS total_spent,

       NTILE(4) OVER (ORDER BY SUM(total_amount)) AS spending_quartile

FROM sales

GROUP BY customer_id;
```

Interpretation:
Segments customers into spending tiers for targeted marketing.

Step 7: Results Analysis

Descriptive

Sales increased steadily, with a few products dominating regional revenue.

Diagnostic

High-performing regions had frequent repeat customers and higher average order values.

Prescriptive

Focus marketing on top-quartile customers and discontinue consistently inactive products.

Step 8: References

- Oracle / PostgreSQL / MySQL Official Documentation
- W3Schools SQL Window Functions
- PostgreSQL Tutorial