# PL/SQL Individual assignments 1

## Hirwa Roy 24174

**Step 1: Problem Definition**

**Business Context:** SIMBA Supermarket with online and on site stores wants to improve sales and customer targeting.  
**Data Challenge:** Sales are recorded per transaction across regions and channels; management needs to identify top products by region, monitor monthly charges and separate customers by spending so marketing can target high value groups.  
**Expected Outcome:** Provide top products per region/ period, monthly running totals and growth, customer spending in percentage, and 3 month moving averages to support inventory and marketing decisions.

**Step 2: Success Criteria**

1. **Top 5 products per region per quarter** — use RANK() or DENSE\_RANK() to identify top 5 by revenue.
2. **Running monthly sales totals** — compute cumulative monthly revenue with SUM() OVER (ORDER BY month ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW).
3. **Month-over-month growth (%)** — use LAG() to compute prior month revenue and % growth.
4. **Customer quartiles** — divide customers into quartiles by total spend using NTILE(4) (or CUME\_DIST() for distribution insights).
5. **3-month moving averages** — compute 3-month moving average revenue with AVG() OVER (ORDER BY month ROWS BETWEEN 2 PRECEDING AND CURRENT ROW).

**Step 3: Database Schema**

Table customers {

  customer\_id int [pk]

  customer\_name varchar

  region varchar}

Table products {

  product\_id int [pk]

  name varchar

  category varchar

  price decimal(12,2)}

Table transactions {

  transaction\_id int [pk]

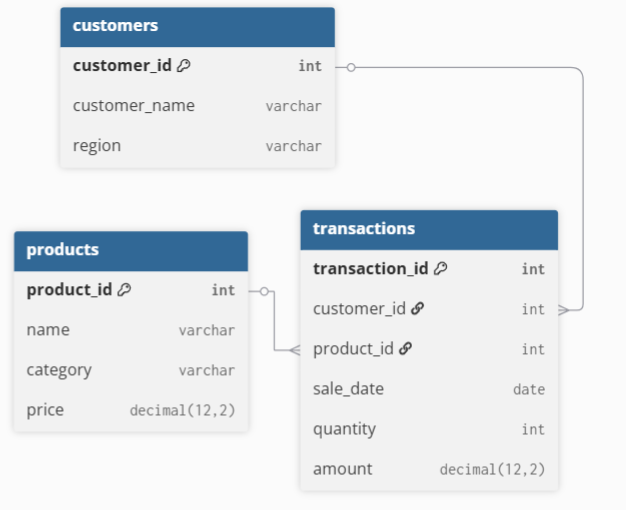
  customer\_id int [ref: > customers.customer\_id]

  product\_id int [ref: > products.product\_id]

  sale\_date date

  quantity int

  amount decimal(12,2)}



**Step 4: Window Functions Implementation**

Step 0: Sample Data Setup

--Drop tables if they exist

DROP TABLE IF EXISTS transactions;

DROP TABLE IF EXISTS products;

DROP TABLE IF EXISTS customers;

-- Create customers table

CREATE TABLE customers (

customer\_id INT PRIMARY KEY,

customer\_name VARCHAR(100),

region VARCHAR(50)

);

-- Create products table

CREATE TABLE products (

product\_id INT PRIMARY KEY,

name VARCHAR(100),

category VARCHAR(50),

price DECIMAL(12,2)

);

-- Create transactions table

CREATE TABLE transactions (

transaction\_id INT PRIMARY KEY,

customer\_id INT,

product\_id INT,

sale\_date DATE,

quantity INT,

amount DECIMAL(12,2),

FOREIGN KEY (customer\_id) REFERENCES customers(customer\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id)

);

-- Insert sample customers

INSERT INTO customers VALUES

(1001, 'John Doe', 'Kigali'),

(1002, 'Grace Uwimana', 'Butare'),

(1003, 'Eric Mukiza', 'Gisenyi'),

(1004, 'Alice Niyonsaba', 'Kigali'),

(1005, 'David Habimana', 'Butare');

-- Insert sample products

INSERT INTO products VALUES

(2001, 'Coffee Beans 1kg', 'Beverages', 25000),

(2002, 'Espresso Machine', 'Equipment', 250000),

(2003, 'Green Tea', 'Beverages', 15000);

-- Insert sample transactions

INSERT INTO transactions VALUES

(3001,1001,2001,'2024-01-15',1,25000),

(3002,1002,2001,'2024-02-05',2,50000),

(3003,1003,2002,'2024-01-20',1,250000),

(3004,1004,2003,'2024-03-10',3,45000),

(3005,1005,2001,'2024-03-15',1,25000),

(3006,1001,2002,'2024-02-18',1,250000),

(3007,1002,2003,'2024-03-20',2,30000);

**-- Step A: Ranking - Top 5 Customers by Revenue**

SELECT \*

FROM (

SELECT customer\_id,

customer\_name,

total\_revenue,

RANK() OVER (ORDER BY total\_revenue DESC) AS revenue\_rank

FROM (

SELECT c.customer\_id,

c.customer\_name,

SUM(t.amount) AS total\_revenue

FROM customers c

JOIN transactions t ON c.customer\_id = t.customer\_id

GROUP BY c.customer\_id, c.customer\_name

) AS sub

) AS ranked\_sub

WHERE revenue\_rank <= 5;

**-- Step B: Aggregate - Running Monthly Sales Totals**

WITH monthly AS (

SELECT DATE\_FORMAT(sale\_date,'%Y-%m-01') AS month\_start,

SUM(amount) AS month\_total

FROM transactions

GROUP BY DATE\_FORMAT(sale\_date,'%Y-%m-01')

)

SELECT month\_start,

month\_total,

SUM(month\_total) OVER (ORDER BY month\_start ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS running\_total

FROM monthly

ORDER BY month\_start;

**-- Step C: Navigation - Month-over-Month Growth**

WITH monthly AS (

SELECT DATE\_FORMAT(sale\_date,'%Y-%m-01') AS month\_start,

SUM(amount) AS month\_total

FROM transactions

GROUP BY DATE\_FORMAT(sale\_date,'%Y-%m-01')

)

SELECT month\_start,

month\_total,

LAG(month\_total) OVER (ORDER BY month\_start) AS prev\_month\_total,

ROUND(

(month\_total - LAG(month\_total) OVER (ORDER BY month\_start))

/ LAG(month\_total) OVER (ORDER BY month\_start) \* 100,2) AS pct\_growth

FROM monthly

ORDER BY month\_start;

**-- Step D: Distribution - Customer Quartiles**

SELECT customer\_id,

customer\_name,

total\_revenue,

NTILE(4) OVER (ORDER BY total\_revenue DESC) AS quartile

FROM (

SELECT c.customer\_id,

c.customer\_name,

SUM(t.amount) AS total\_revenue

FROM customers c

JOIN transactions t ON c.customer\_id = t.customer\_id

GROUP BY c.customer\_id, c.customer\_name

) AS sub

ORDER BY quartile, total\_revenue DESC;

**-- Step E: Moving Average (3 Month)**

WITH monthly AS (

SELECT DATE\_FORMAT(sale\_date,'%Y-%m-01') AS month\_start,

SUM(amount) AS month\_total

FROM transactions

GROUP BY DATE\_FORMAT(sale\_date,'%Y-%m-01')

)

SELECT month\_start,

month\_total,

AVG(month\_total) OVER (ORDER BY month\_start ROWS BETWEEN 2 PRECEDING AND CURRENT ROW) AS moving\_avg\_3m

FROM monthly

ORDER BY month\_start;

**-- Step F: Extra - PERCENT\_RANK & CUME\_DIST**

SELECT customer\_id,

customer\_name,

total\_revenue,

PERCENT\_RANK() OVER (ORDER BY total\_revenue DESC) AS pct\_rank,

CUME\_DIST() OVER (ORDER BY total\_revenue DESC) AS cume\_dist

FROM (

SELECT c.customer\_id,

c.customer\_name,

SUM(t.amount) AS total\_revenue

FROM customers c

JOIN transactions t ON c.customer\_id = t.customer\_id

GROUP BY c.customer\_id, c.customer\_name

) AS sub;