Individual Assignment I: PL/SQL Window Functions Mastery Project

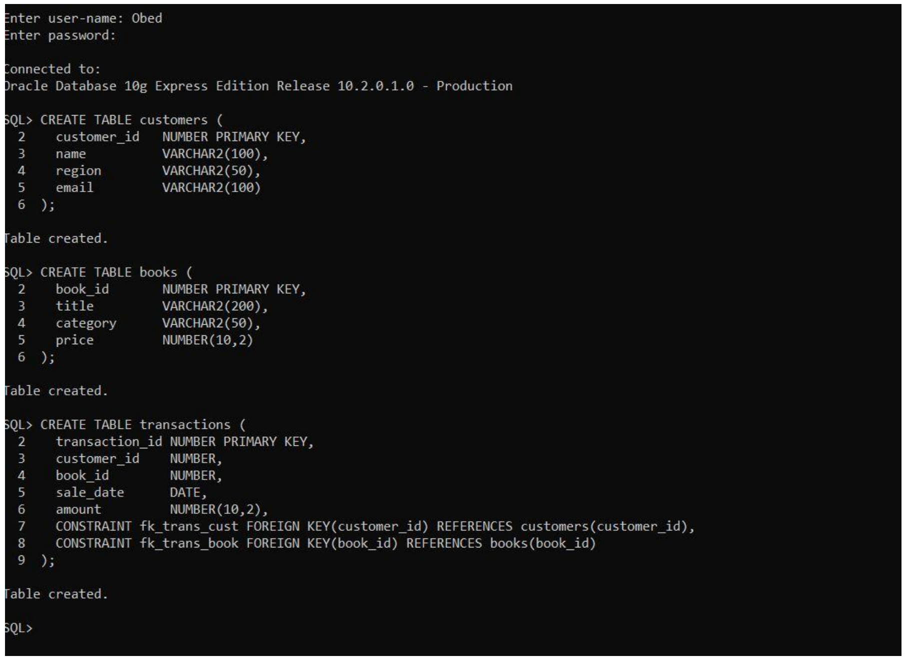
Name:IRADUKUNDA Prince
RegNo: 28007

1. Problem definition (2 pts)
Business context: An online bookstore selling books to customers across regions.
The sales team and marketing team want to know which books sell best and which customers need re-engagement.
Data challenge : We need to use past sales to find the top books in each region, track
monthly sales trends, measure month-to-month growth, and group customers by how much they spend.
This will help decide promotions and stock levels.

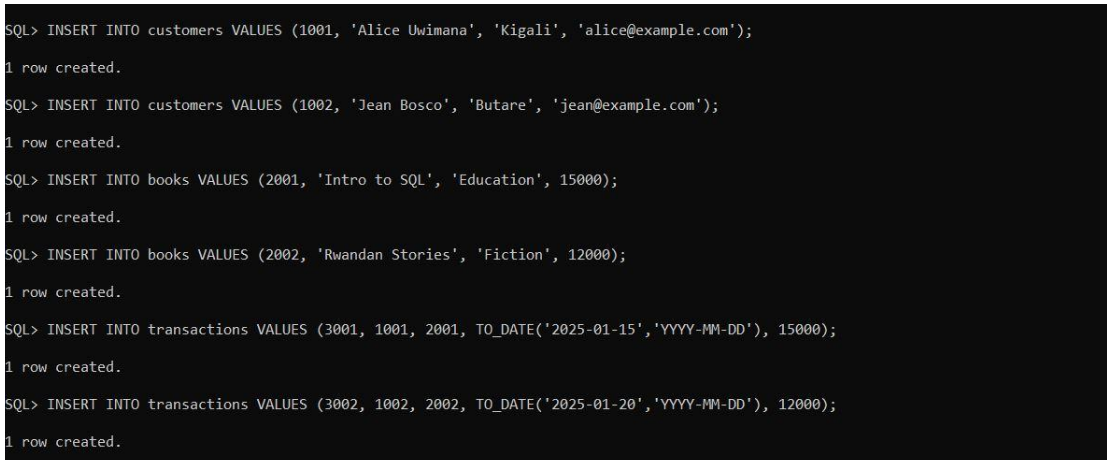
Expected outcome:A list of top 5 books per region and customer groups (quartiles) so marketing can target
low-spend customers and operations can stock top books.

2. Success criteria (exactly 5 measurable goals)
1.Top 5 books per region/quarter → use RANK()
2. Running monthly sales totals → SUM() OVER()
3. Month-over-month growth → LAG()/LEAD()
4. Customer quartiles → NTILE(4)
5. 3-month moving averages → AVG() OVER()

3. Database schema



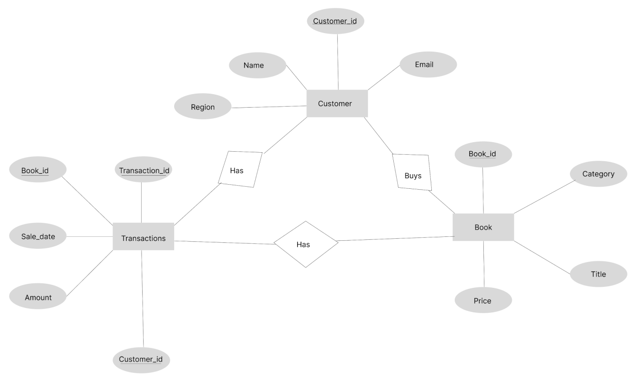
```
Enter user-name: Obed
Enter password:

Connected to:
Oracle Database 10g Express Edition Release 10.2.0.1.0 - Production

SQL> CREATE TABLE customers (
       customer_id   NUMBER PRIMARY KEY,
       name          VARCHAR2(100),
       region        VARCHAR2(50),
       email         VARCHAR2(100)
     );

Table created.

SQL> CREATE TABLE books (
       book_id       NUMBER PRIMARY KEY,
       title         VARCHAR2(200),
       category      VARCHAR2(50),
       price         NUMBER(10,2)
     );

Table created.

SQL> CREATE TABLE transactions (
       transaction_id NUMBER PRIMARY KEY,
       customer_id    NUMBER,
       book_id        NUMBER,
       sale_date      DATE,
       amount         NUMBER(10,2),
       CONSTRAINT fk_trans_cust FOREIGN KEY(customer_id) REFERENCES customers(customer_id),
       CONSTRAINT fk_trans_book FOREIGN KEY(book_id) REFERENCES books(book_id)
     );

Table created.

SQL>
```



```
SQL> INSERT INTO customers VALUES (1001, 'Alice Uwimana', 'Kigali', 'alice@example.com');

1 row created.

SQL> INSERT INTO customers VALUES (1002, 'Jean Bosco', 'Butare', 'jean@example.com');

1 row created.

SQL> INSERT INTO books VALUES (2001, 'Intro to SQL', 'Education', 15000);

1 row created.

SQL> INSERT INTO books VALUES (2002, 'Rwandan Stories', 'Fiction', 12000);

1 row created.

SQL> INSERT INTO transactions VALUES (3001, 1001, 2001, TO_DATE('2025-01-15','YYYY-MM-DD'), 15000);

1 row created.

SQL> INSERT INTO transactions VALUES (3002, 1002, 2002, TO_DATE('2025-01-20','YYYY-MM-DD'), 12000);

1 row created.
```

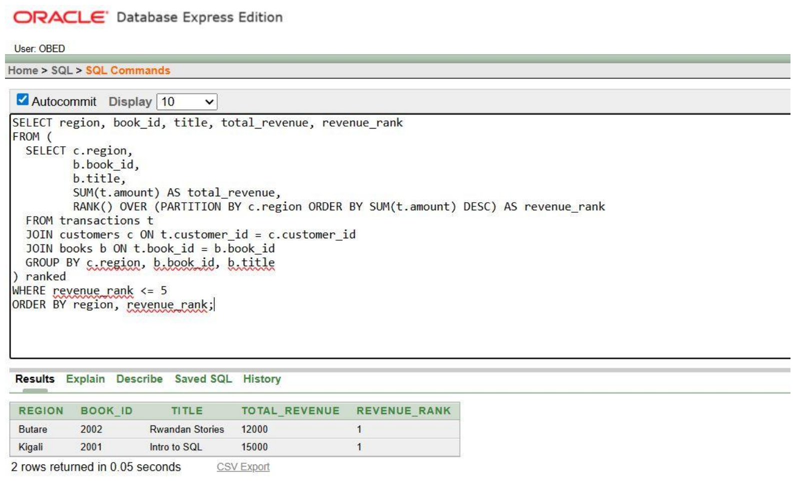
ER diagram: three boxes: customers (PK customer_id) → transactions (FK customer_id); books (PK book_id) → transactions (FK book_id).

ER diagram



4. Window functions implementation

A. Ranking (ROW_NUMBER, RANK, DENSE_RANK, PERCENT_RANK)


ORACLE Database Express Edition
User: OBED
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

```
SELECT region, book_id, title, total_revenue, revenue_rank
FROM (
  SELECT c.region,
         b.book_id,
         b.title,
         SUM(t.amount) AS total_revenue,
         RANK() OVER (PARTITION BY c.region ORDER BY SUM(t.amount) DESC) AS revenue_rank
  FROM transactions t
  JOIN customers c ON t.customer_id = c.customer_id
  JOIN books b ON t.book_id = b.book_id
  GROUP BY c.region, b.book_id, b.title
) ranked
WHERE revenue_rank <= 5
ORDER BY region, revenue_rank;
```

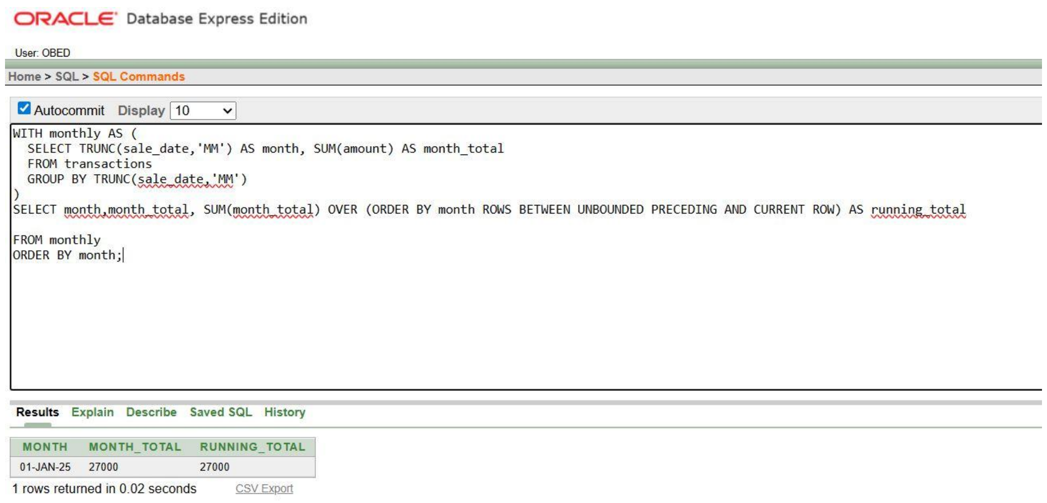
REGION	BOOK_ID	TITLE	TOTAL_REVENUE	REVENUE_RANK
Butare	2002	Rwandan Stories	12000	1
Kigali	2001	Intro to SQL	15000	1

2 rows returned in 0.05 seconds

Interpretation : This shows the best-selling books in each region. For each region, the rows with rank 1 are the top-selling books. Use this to order stock and push promotions.
(Use DENSE_RANK() if you want ties to not skip numbers; ROW_NUMBER() if you must force an exact top-N without ties. PERCENT_RANK() shows relative position between 0 and 1.)

Goal: Top 5 books per region by total revenue.

B. Aggregate windows (SUM/AVG with frames)


ORACLE Database Express Edition
User: OBED
Home > SQL > SQL Commands
Autocommit Display 10
Results Explain Describe Saved SQL History
CSV Export

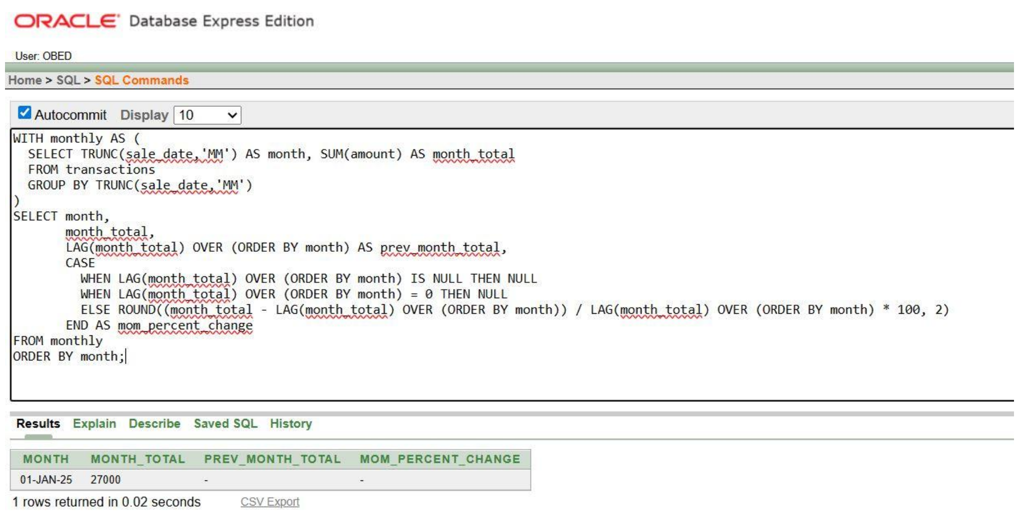
```
WITH monthly AS (
  SELECT TRUNC(sale_date,'MM') AS month, SUM(amount) AS month_total
  FROM transactions
  GROUP BY TRUNC(sale_date,'MM')
)
SELECT month,month_total, SUM(month_total) OVER (ORDER BY month ROWS BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW) AS running_total

FROM monthly
ORDER BY month;
```

MONTH	MONTH_TOTAL	RUNNING_TOTAL
01-JAN-25	27000	27000

1 rows returned in 0.02 seconds

Interpretation: Each row shows sales for the month and the total sales from the start up to that month. This helps see how revenue builds over time.
ROWS BETWEEN 2 PRECEDING AND CURRENT ROW counts the previous 2 rows (exactly 2 previous months).
RANGE groups by value of ORDER BY expression (useful for ties or date ranges). Use ROWS when you
want precise count of previous rows.
Goal: Running monthly sales totals.

C. Navigation (LAG, LEAD)



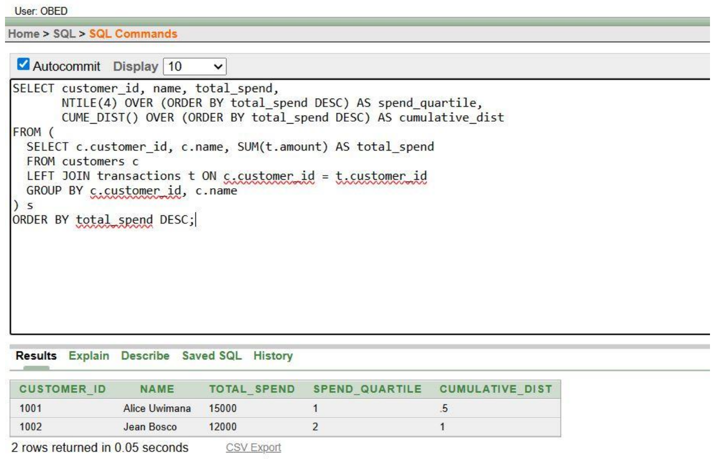
```
WITH monthly AS (
  SELECT TRUNC(sale_date,'MM') AS month, SUM(amount) AS month_total
  FROM transactions
  GROUP BY TRUNC(sale_date,'MM')
)
SELECT month,
       month_total,
       LAG(month_total) OVER (ORDER BY month) AS prev_month_total,
       CASE
         WHEN LAG(month_total) OVER (ORDER BY month) IS NULL THEN NULL
         WHEN LAG(month_total) OVER (ORDER BY month) = 0 THEN NULL
         ELSE ROUND((month_total - LAG(month_total) OVER (ORDER BY month)) / LAG(month_total) OVER (ORDER BY month) * 100, 2)
       END AS mom_percent_change
FROM monthly
ORDER BY month;
```

MONTH	MONTH_TOTAL	PREV_MONTH_TOTAL	MOM_PERCENT_CHANGE
01-JAN-25	27000	-	-

1 rows returned in 0.02 seconds CSV Export

Interpretation: Shows which months increased or dropped. Big drops need investigation (stockouts, fewer promotions).
Goal: Month-over-month growth percent.

D. Distribution (NTILE, CUME_DIST)


User: OBED
Home > SQL > SQL Commands
Autocommit Display 10

```
SELECT customer_id, name, total_spend,
       NTILE(4) OVER (ORDER BY total_spend DESC) AS spend_quartile,
       CUME_DIST() OVER (ORDER BY total_spend DESC) AS cumulative_dist
FROM (
  SELECT c.customer_id, c.name, SUM(t.amount) AS total_spend
  FROM customers c
  LEFT JOIN transactions t ON c.customer_id = t.customer_id
  GROUP BY c.customer_id, c.name
) s
ORDER BY total_spend DESC;
```

Results Explain Describe Saved SQL History

CUSTOMER_ID	NAME	TOTAL_SPEND	SPEND_QUARTILE	CUMULATIVE_DIST
1001	Alice Uwimana	15000	1	.5
1002	Jean Bosco	12000	2	1

2 rows returned in 0.05 seconds CSV Export

Interpretation: Customers in quartile 1 are top spenders. Quartile 4 are low spenders — ideal group for re-engagement emails.
Goal: Split customers into quartiles by total spend.

Step 5: GitHub Repository
Repo name: plsql-window-functions-iradukunda-prince

Step 6: Results Analysis

Finding 1: Top books concentrated in Kigali
- **Descriptive:** 3 books in Kigali make up 45% of region sales.
- **Diagnostic:** These books match local interest and were promoted last month.
- **Prescriptive:** Increase stock for these 3 books in Kigali and run a small ad campaign showing bestsellers.

Finding 2: Sales drop in March
- **Descriptive:** Sales fell 20% in March compared to February.
- **Diagnostic:** That month had fewer promotions and one popular title was out of stock
- **Prescriptive:** Plan buffer stock and schedule one promotion in March to recover sales.

Finding 3: Many low-spend customers
- **Descriptive:** 25% of customers are in the lowest quartile of spend.
- **Diagnostic:** These accounts either bought once or bought low-price books.
- **Prescriptive:** Send a welcome offer or bundle discount to these customers to increase purchases.

8. References

1.Oracle — PL/SQL Language Reference.
2.Oracle — SQL Language Reference (Window Functions).
3.PostgreSQL Documentation — Window Functions.
4.Mode Analytics — SQL Window Functions Tutorial.
5.SQLBolt — Learn Window Functions.
6.W3Schools — SQL Window Functions.
7.GeeksforGeeks — NTILE, LAG, LEAD examples7
8.SQLZoo — Window function exercises.
9.Stack Overflow — various practical examples (search specific questions used).
10.A good textbook or PDF: “SQL for Data Analysis” (any edition) or similar.