29/09/25

NAME: TCHACHOU KEMDOUNG KELLY-HANS

ID: 27265

COURSE: Database Development with PL/SQL

INSTRUCTOR: ERIC MANIRAGUHA

GROUP: B

Individual Assignment I: PL/SQL Window Functions Mastery Project

STEP 1: Problem Definition

Business Context

Joie du Peuple, a company in the beverage distribution industry, specializes in selling drinks such as beer and juice. The sales and marketing department aims to gain a deeper understanding of product performance across various regions and time periods.

Data Challenge

Although the company collects transaction data regularly (including customer information, products, sales dates, and amounts), it is challenging to identify the best-selling drinks by region and track monthly sales growth. In addition, the company struggles to segment customers effectively for targeted marketing campaigns

Expected Outcome

By applying PL/SQL window functions, the company expects to identify top-performing drinks, monitor sales growth trends, calculate running totals, and segment customers into categories that will support smarter marketing and sales strategies.

STEP 2: Success criteria

Define exactly 5 measurable goals:

1-Top 5 products per region/quarter: RANK ()

-Rank the sales of beer and juice across regions and quarters to identify the most popular products

2-Running monthly sales totals: SUM () OVER ()

-Calculate cumulative monthly sales to track how revenue grows over time

3-Month-over-month growth: LAG () / LEAD ()

-Compare sales between consecutive months to measure growth or decline in drink sales

4-Customer quartiles: NTLE (4)

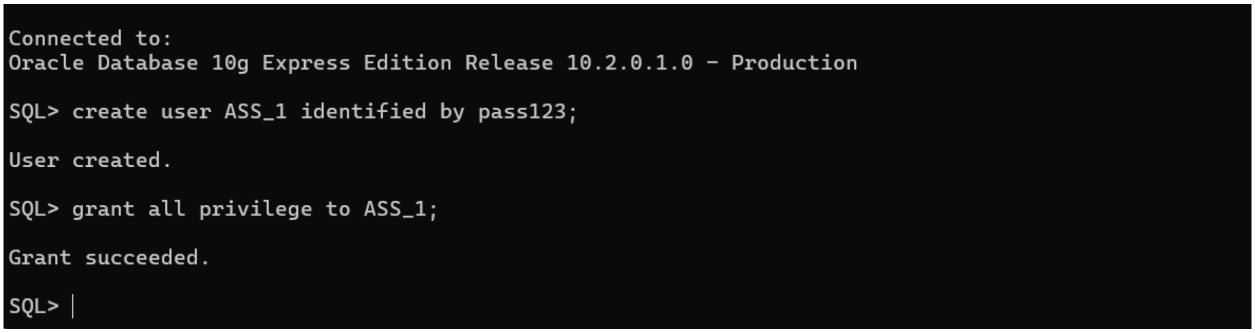
-Divide customers into four groups based on their total spending

5-3-month moving averages: AVG () OVER ()

-Calculate rolling averages of sales over 3 months to smooth out short-term fluctuations and reveal long-term trends

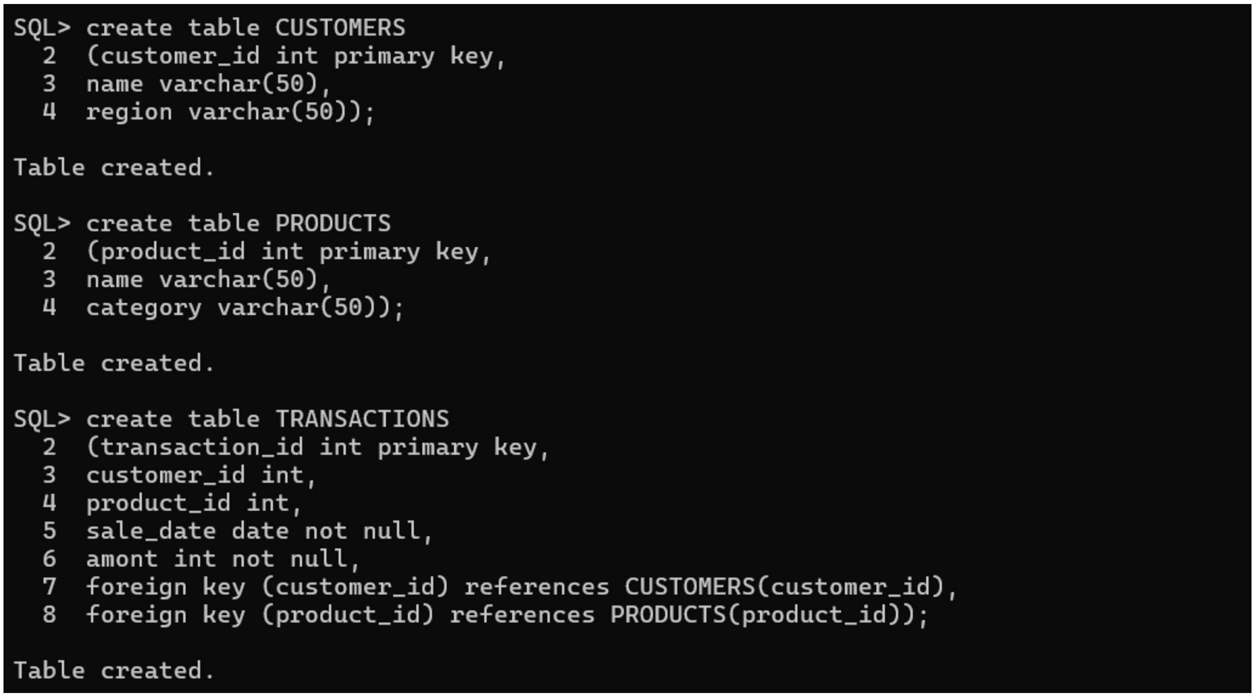
STEP 3: Database Schema

Create a user:



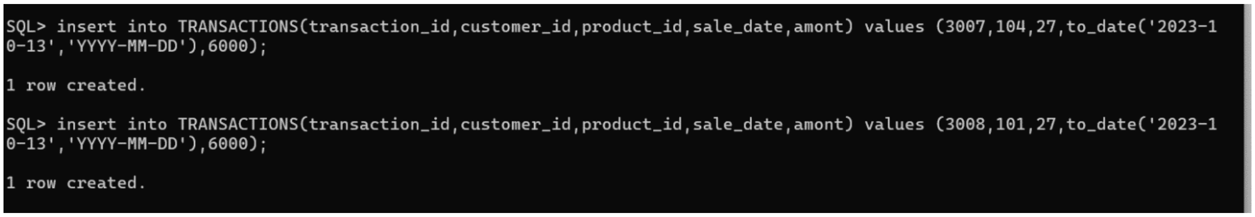
```
Connected to:
Oracle Database 10g Express Edition Release 10.2.0.1.0 - Production

SQL> create user ASS_1 identified by pass123;

User created.

SQL> grant all privilege to ASS_1;

Grant succeeded.

SQL> |
```

Create tables:



```
SQL> create table CUSTOMERS
     (customer_id int primary key,
     name varchar(50),
     region varchar(50));

Table created.

SQL> create table PRODUCTS
     (product_id int primary key,
     name varchar(50),
     category varchar(50));

Table created.

SQL> create table TRANSACTIONS
     (transaction_id int primary key,
     customer_id int,
     product_id int,
     sale_date date not null,
     amont int not null,
     foreign key (customer_id) references CUSTOMERS(customer_id),
     foreign key (product_id) references PRODUCTS(product_id));

Table created.
```

Insert into tables:



```
SQL> insert into TRANSACTIONS(transaction_id,customer_id,product_id,sale_date,amont) values (3007,104,27,to_date('2023-1
0-13','YYYY-MM-DD'),6000);

1 row created.

SQL> insert into TRANSACTIONS(transaction_id,customer_id,product_id,sale_date,amont) values (3008,101,27,to_date('2023-1
0-13','YYYY-MM-DD'),6000);

1 row created.
```

ER diagram:

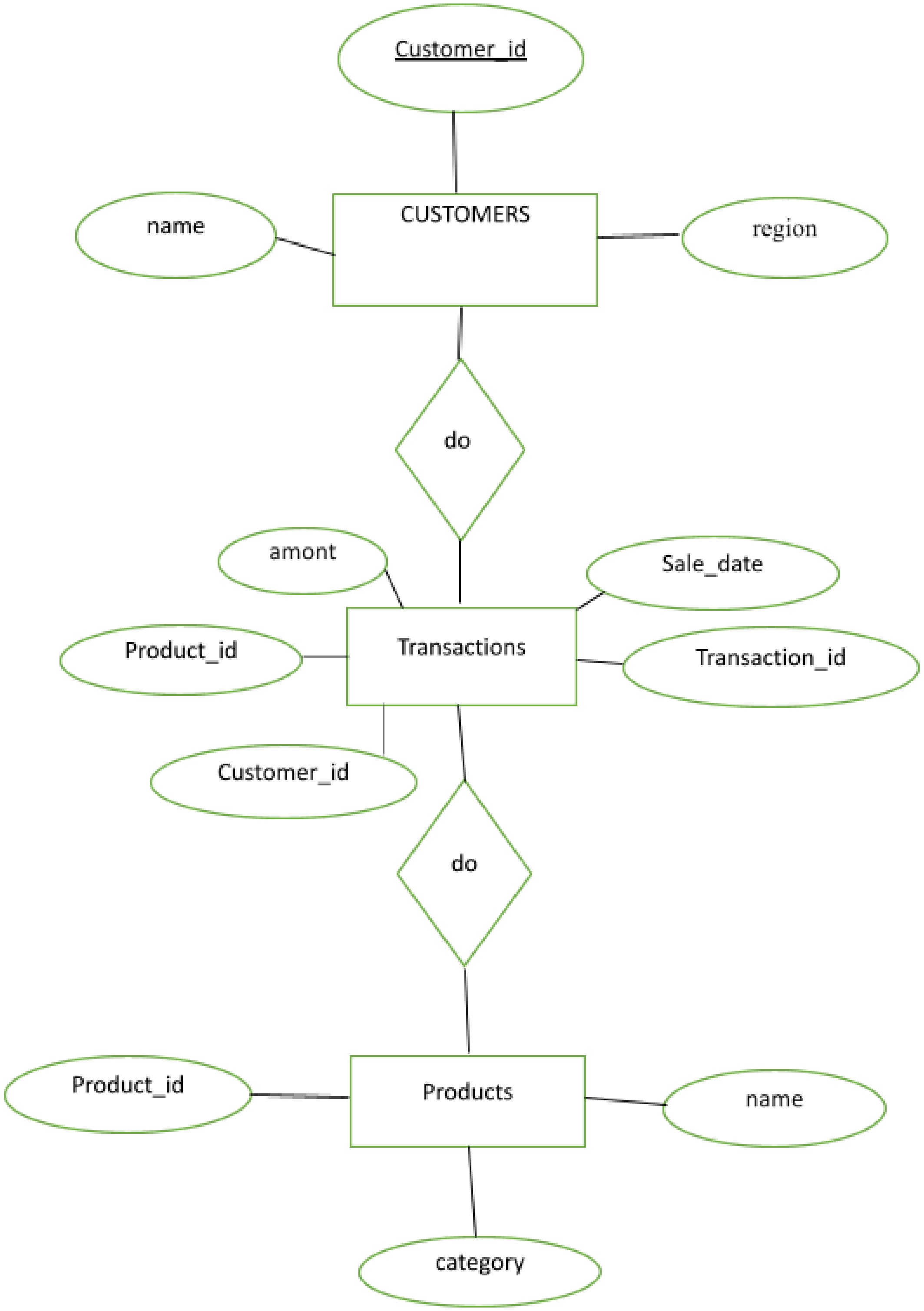
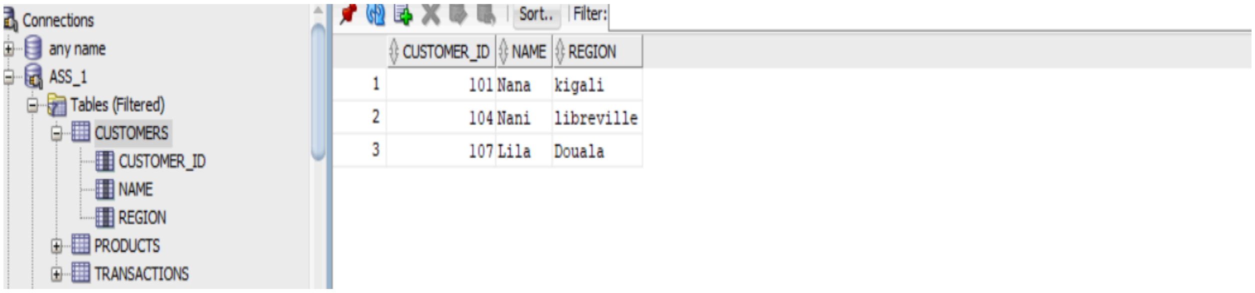
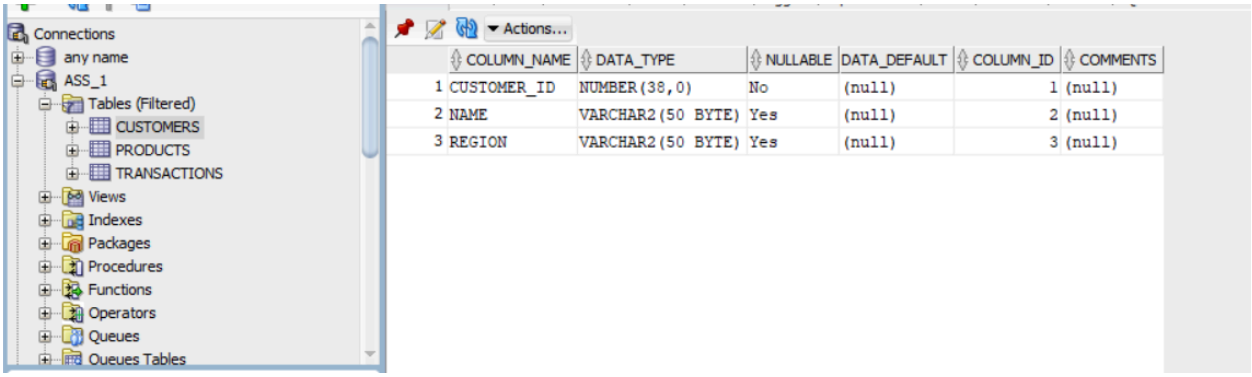

Customer_id
name
CUSTOMERS
region
do
amont
Sale_date
Product_id
Transactions
Transaction_id
Customer_id
do
Product_id
Products
name
category

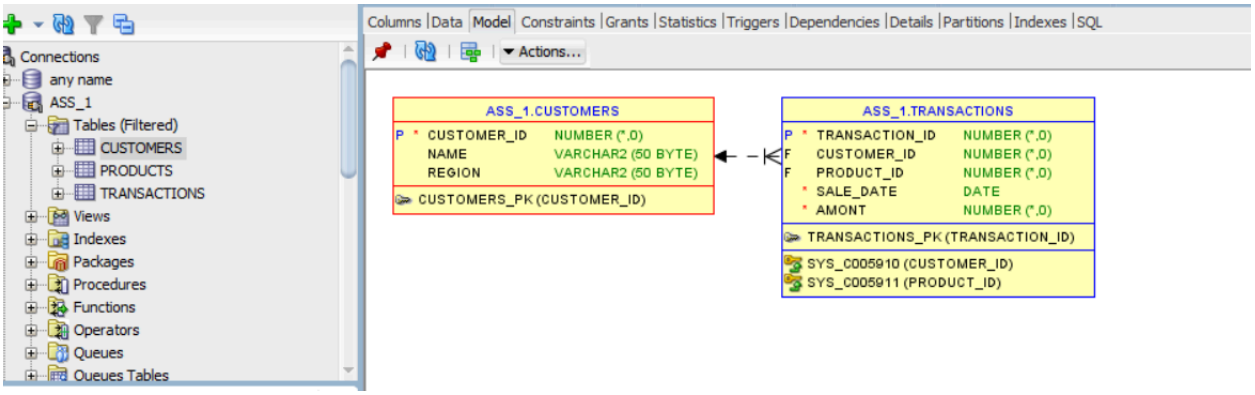
Table Customers:



	CUSTOMER_ID	NAME	REGION
1	101	Nana	kigali
2	104	Nani	libreville
3	107	Lila	Douala



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	CUSTOMER_ID	NUMBER(38,0)	No	(null)	1	(null)
2	NAME	VARCHAR2(50 BYTE)	Yes	(null)	2	(null)
3	REGION	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)



ASS_1.CUSTOMERS

P	*	CUSTOMER_ID	NUMBER (*,0)
		NAME	VARCHAR2 (50 BYTE)
		REGION	VARCHAR2 (50 BYTE)

CUSTOMERS_PK (CUSTOMER_ID)

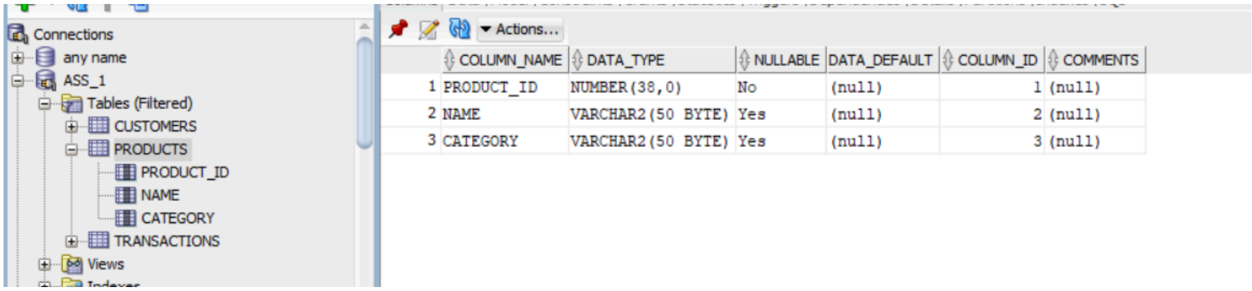
ASS_1.TRANSACTIONS

P	*	TRANSACTION_ID	NUMBER (*,0)
F		CUSTOMER_ID	NUMBER (*,0)
F		PRODUCT_ID	NUMBER (*,0)
	*	SALE_DATE	DATE
	*	AMONT	NUMBER (*,0)

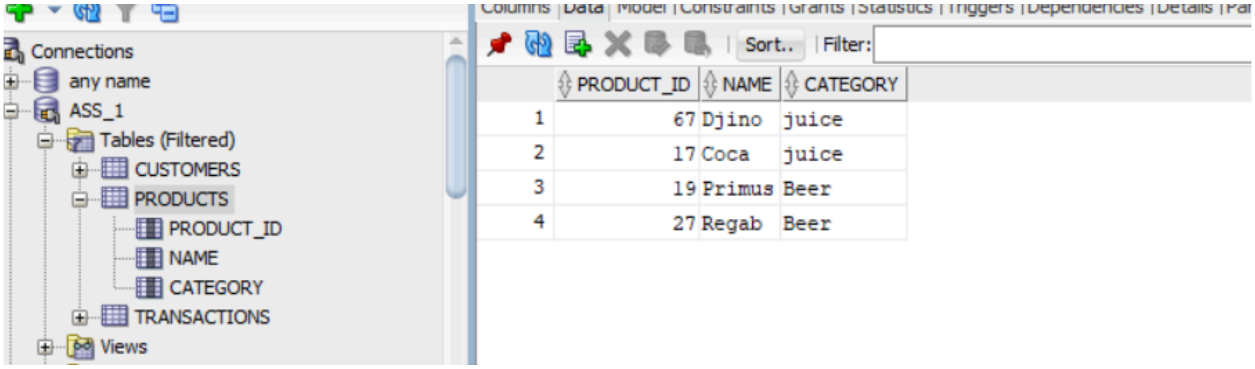
TRANSACTIONS_PK (TRANSACTION_ID)

SYS_C005910 (CUSTOMER_ID)
SYS_C005911 (PRODUCT_ID)

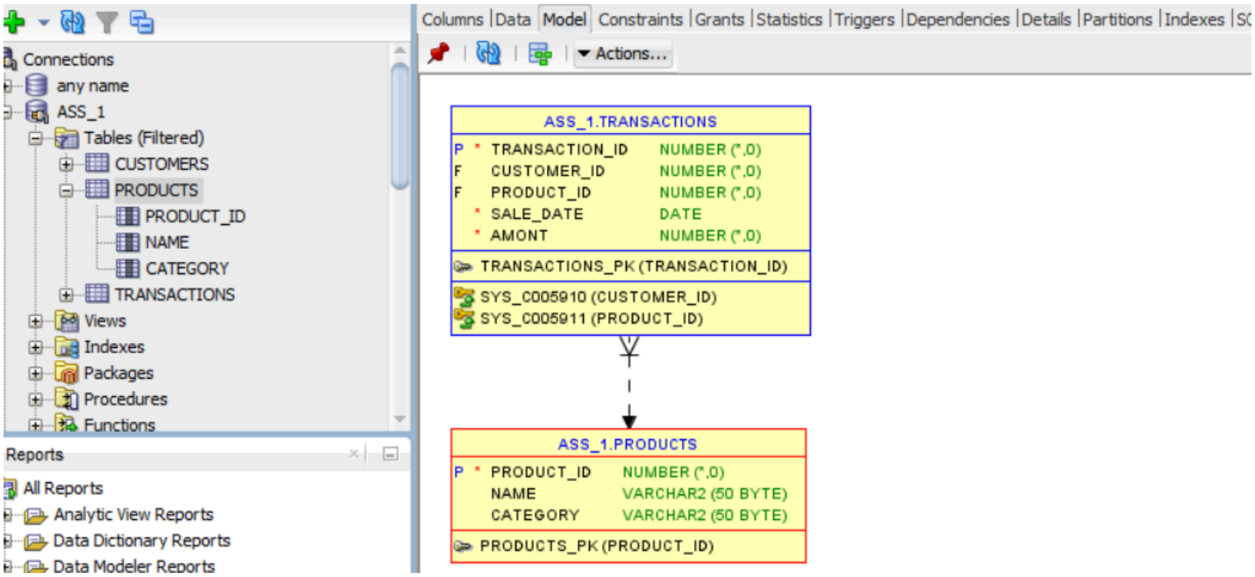
Table Products:



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	PRODUCT_ID	NUMBER(38,0)	No	(null)	1	(null)
2	NAME	VARCHAR2(50 BYTE)	Yes	(null)	2	(null)
3	CATEGORY	VARCHAR2(50 BYTE)	Yes	(null)	3	(null)



	PRODUCT_ID	NAME	CATEGORY
1	67	Djino	juice
2	17	Coca	juice
3	19	Primus	Beer
4	27	Regab	Beer



ASS_1.TRANSACTIONS

P	*	TRANSACTION_ID	NUMBER (*,0)
F		CUSTOMER_ID	NUMBER (*,0)
F		PRODUCT_ID	NUMBER (*,0)
	*	SALE_DATE	DATE
	*	AMONT	NUMBER (*,0)

TRANSACTIONS_PK (TRANSACTION_ID)

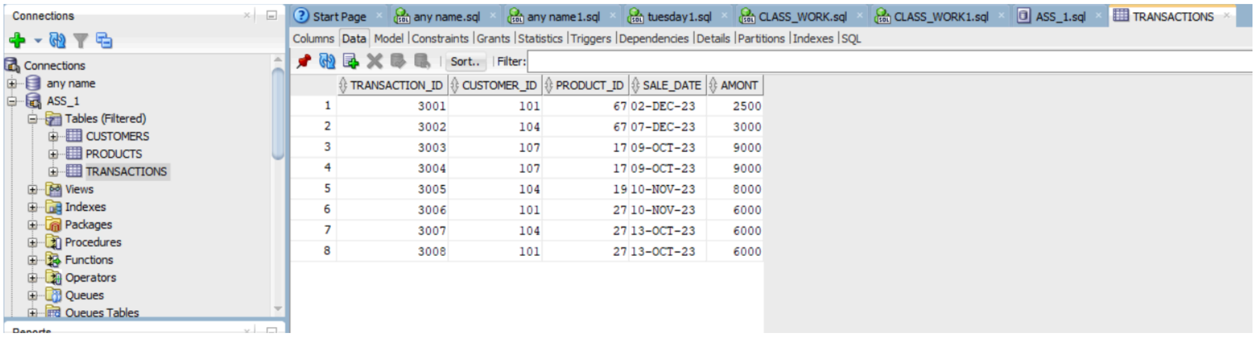
SYS_C005910 (CUSTOMER_ID)
SYS_C005911 (PRODUCT_ID)

ASS_1.PRODUCTS

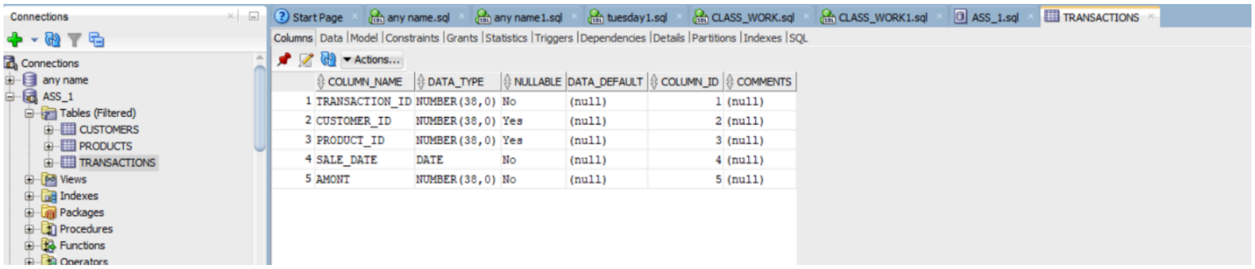
P	*	PRODUCT_ID	NUMBER (*,0)
		NAME	VARCHAR2 (50 BYTE)
		CATEGORY	VARCHAR2 (50 BYTE)

PRODUCTS_PK (PRODUCT_ID)

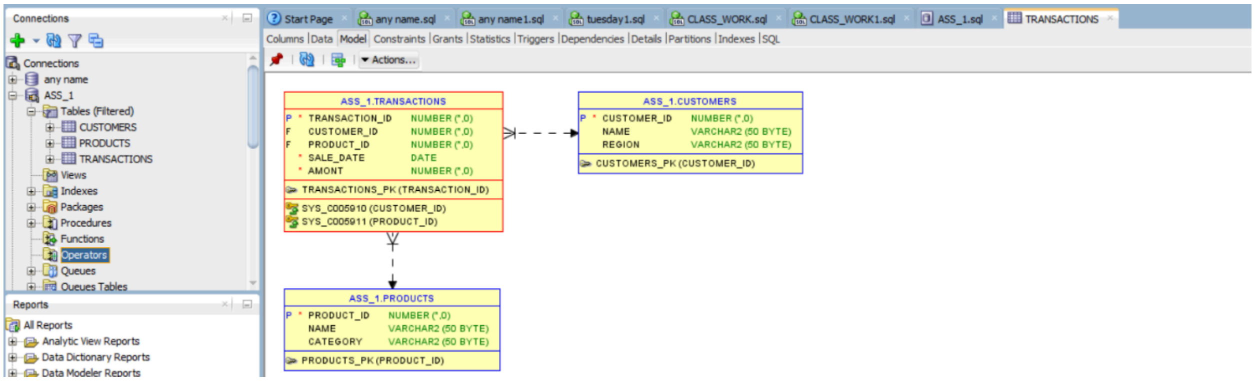
Table Transactions:



	TRANSACTION_ID	CUSTOMER_ID	PRODUCT_ID	SALE_DATE	AMONT
1	3001	101	67	02-DEC-23	2500
2	3002	104	67	07-DEC-23	3000
3	3003	107	17	09-OCT-23	9000
4	3004	107	17	09-OCT-23	9000
5	3005	104	19	10-NOV-23	8000
6	3006	101	27	10-NOV-23	6000
7	3007	104	27	13-OCT-23	6000
8	3008	101	27	13-OCT-23	6000



	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	TRANSACTION_ID	NUMBER(38,0)	No	(null)	1	(null)
2	CUSTOMER_ID	NUMBER(38,0)	Yes	(null)	2	(null)
3	PRODUCT_ID	NUMBER(38,0)	Yes	(null)	3	(null)
4	SALE_DATE	DATE	No	(null)	4	(null)
5	AMONT	NUMBER(38,0)	No	(null)	5	(null)



ASS_1.TRANSACTIONS

```
P * TRANSACTION_ID   NUMBER (*,0)
F   CUSTOMER_ID      NUMBER (*,0)
F   PRODUCT_ID       NUMBER (*,0)
  * SALE_DATE        DATE
  * AMONT            NUMBER (*,0)
TRANSACTIONS_PK (TRANSACTION_ID)
SYS_C005910 (CUSTOMER_ID)
SYS_C005911 (PRODUCT_ID)
```

ASS_1.CUSTOMERS

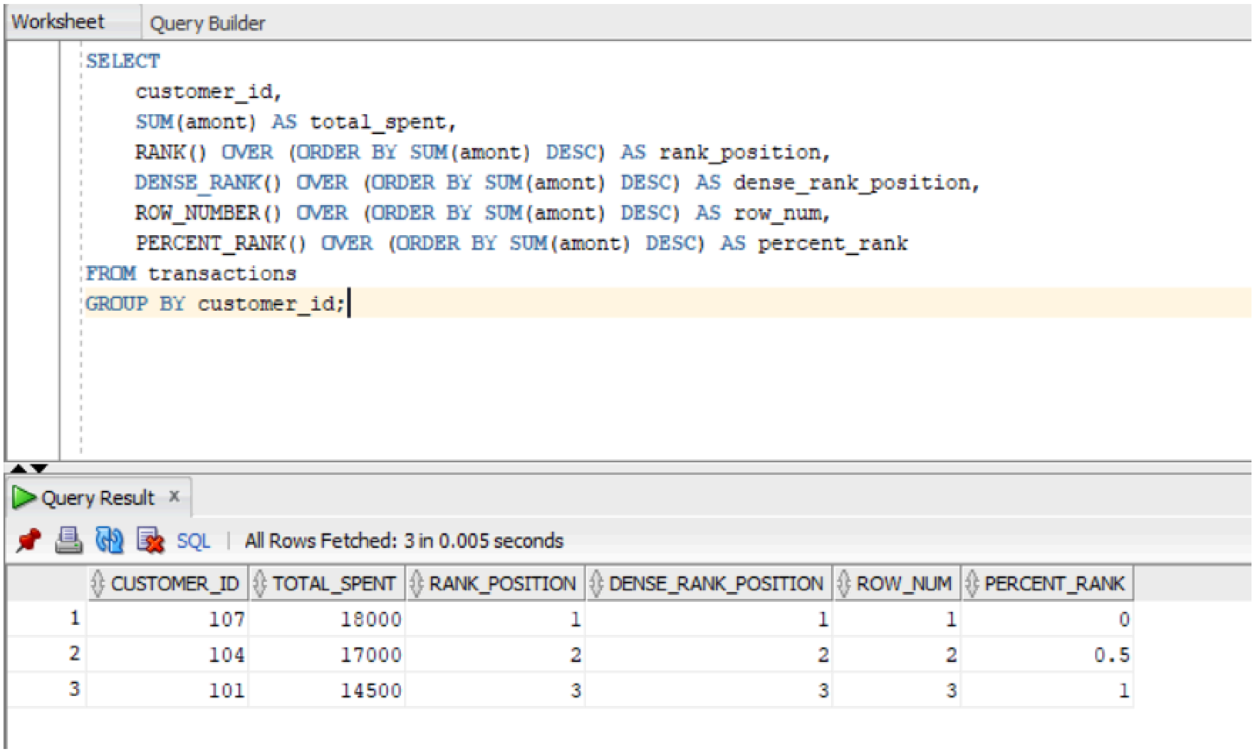
```
P * CUSTOMER_ID   NUMBER (*,0)
    NAME          VARCHAR2 (50 BYTE)
    REGION        VARCHAR2 (50 BYTE)
CUSTOMERS_PK (CUSTOMER_ID)
```

ASS_1.PRODUCTS

```
P * PRODUCT_ID   NUMBER (*,0)
    NAME         VARCHAR2 (50 BYTE)
    CATEGORY     VARCHAR2 (50 BYTE)
PRODUCTS_PK (PRODUCT_ID)
```

STEP 4: Window Functions Implementation

1-Ranking: ROW_NUMBER(), RANK(), DENSE_RANK(), PERCENT_RANK()

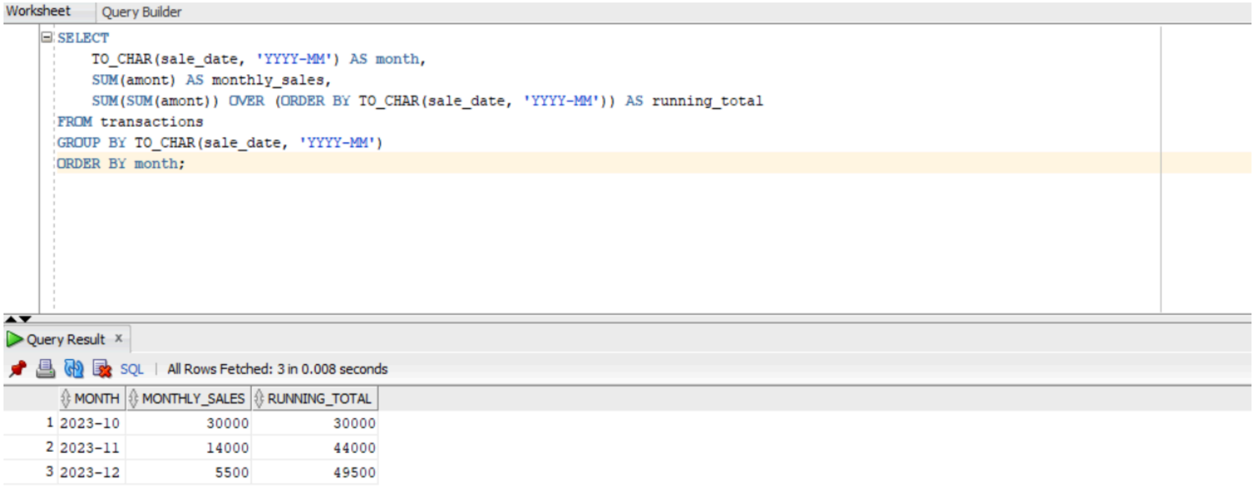


```
SELECT
    customer_id,
    SUM(amont) AS total_spent,
    RANK() OVER (ORDER BY SUM(amont) DESC) AS rank_position,
    DENSE_RANK() OVER (ORDER BY SUM(amont) DESC) AS dense_rank_position,
    ROW_NUMBER() OVER (ORDER BY SUM(amont) DESC) AS row_num,
    PERCENT_RANK() OVER (ORDER BY SUM(amont) DESC) AS percent_rank
FROM transactions
GROUP BY customer_id;
```

All Rows Fetched: 3 in 0.005 seconds

	CUSTOMER_ID	TOTAL_SPENT	RANK_POSITION	DENSE_RANK_POSITION	ROW_NUM	PERCENT_RANK
1	107	18000	1	1	1	0
2	104	17000	2	2	2	0.5
3	101	14500	3	3	3	1

2-Aggregate: SUM(), AVG, MIN(), MAX(), with frame comparisons (ROWS vs RANGE) use case: Running totals & trends

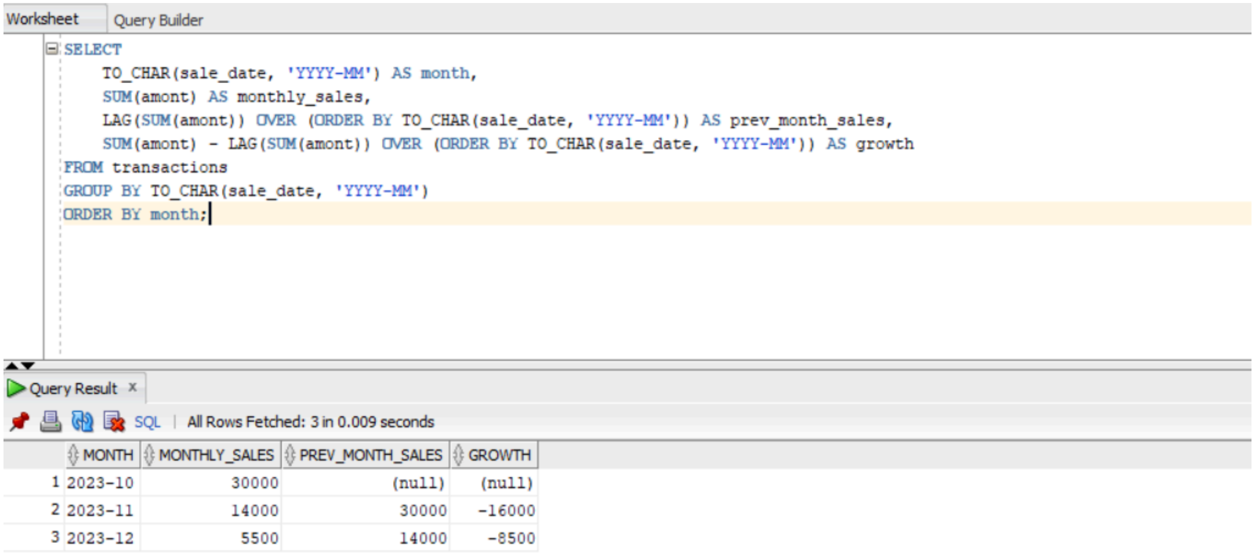


```sql
SELECT
    TO_CHAR(sale_date, 'YYYY-MM') AS month,
    SUM(amont) AS monthly_sales,
    SUM(SUM(amont)) OVER (ORDER BY TO_CHAR(sale_date, 'YYYY-MM')) AS running_total
FROM transactions
GROUP BY TO_CHAR(sale_date, 'YYYY-MM')
ORDER BY month;
```

All Rows Fetched: 3 in 0.008 seconds

	MONTH	MONTHLY_SALES	RUNNING_TOTAL
1	2023-10	30000	30000
2	2023-11	14000	44000
3	2023-12	5500	49500

3-Navigation: LAG(), LEAD(), growth % calculation use case: Period-to-period analysis

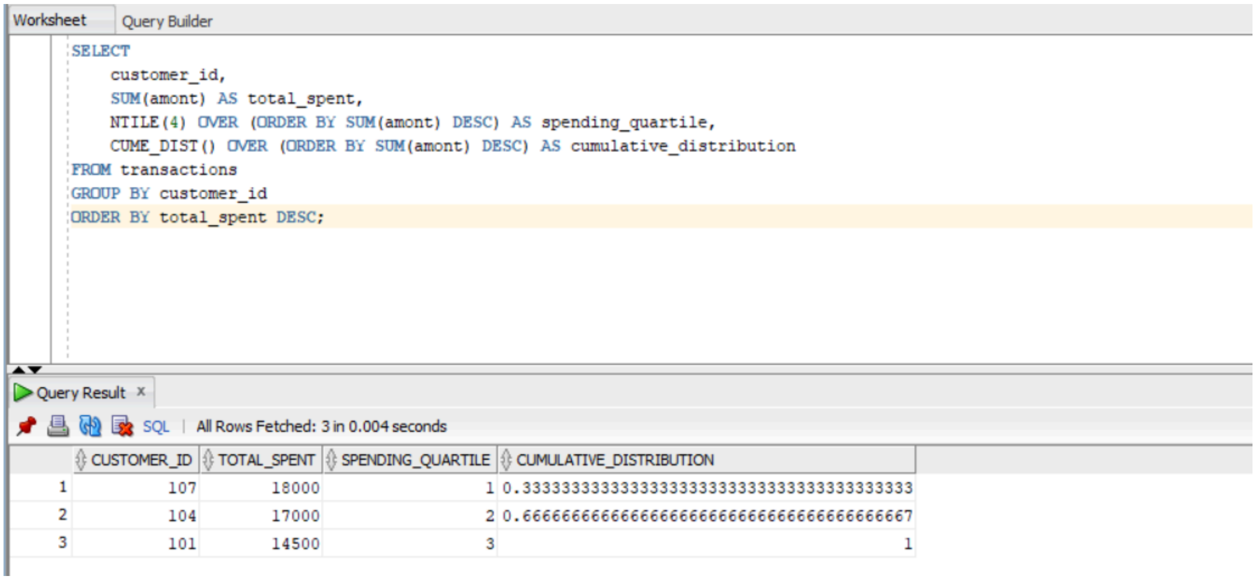


```sql
SELECT
    TO_CHAR(sale_date, 'YYYY-MM') AS month,
    SUM(amont) AS monthly_sales,
    LAG(SUM(amont)) OVER (ORDER BY TO_CHAR(sale_date, 'YYYY-MM')) AS prev_month_sales,
    SUM(amont) - LAG(SUM(amont)) OVER (ORDER BY TO_CHAR(sale_date, 'YYYY-MM')) AS growth
FROM transactions
GROUP BY TO_CHAR(sale_date, 'YYYY-MM')
ORDER BY month;
```

All Rows Fetched: 3 in 0.009 seconds

	MONTH	MONTHLY_SALES	PREV_MONTH_SALES	GROWTH
1	2023-10	30000	(null)	(null)
2	2023-11	14000	30000	-16000
3	2023-12	5500	14000	-8500

4-Distribution: NTILE(4), CUME_DIST() Use case: Customer segmentation



```
SELECT
    customer_id,
    SUM(amont) AS total_spent,
    NTILE(4) OVER (ORDER BY SUM(amont) DESC) AS spending_quartile,
    CUME_DIST() OVER (ORDER BY SUM(amont) DESC) AS cumulative_distribution
FROM transactions
GROUP BY customer_id
ORDER BY total_spent DESC;
```

All Rows Fetched: 3 in 0.004 seconds

	CUSTOMER_ID	TOTAL_SPENT	SPENDING_QUARTILE	CUMULATIVE_DISTRIBUTION
1	107	18000	1	0.3333333333333333333333333333333333333333
2	104	17000	2	0.6666666666666666666666666666666666666667
3	101	14500	3	1

STEP 5: GitHub Repository

Repo name: plsql-window-functions-TVHACHOUKEMDOUNG-Kelly-Hans

STEP 6: Results Analysis

1-Descriptive:

Customer 107 is the top spender with 18000, followed by 104 with 17000 and 101 with 14500. Total sales reached 49500 between October and December 2023. Sales started high in October with 30000 but dropped in November with 14000 and further in December with 5500. Month-over-month growth was -16000 in November and -8500 in December, showing two consecutive declines.

2-Diagnostic:

Sales performance was heavily dependent on a few high-spending customers, with 107 and 104 together representing 67% of total spending. The decline in sales after October suggests that seasonal factors or a lack of promotion in November and December may be at play. The quartile distribution reveals a narrow customer base, with most revenue generated from three clients, making the company vulnerable if even one of them reduces their purchases.

3-Presciptive

Joie du Peuple should create loyalty programs for top-spending customers to maintain their commitment. Targeted promotions for medium-spending customers can encourage them to buy more frequently. Seasonal campaigns during low-performing months like November and December will help stabilize sales. Expanding the customer base is necessary to reduce reliance on only a few heavy spenders. Finally, diversifying the product line, especially with new juice flavors, will attract more customers and balance seasonal fluctuations.

STEP 7: References

-Class notes, particularly Database Management Systems (DBMS), AUCA (2024-2025).

-YouTube, programming with SQL and PL/SQL tutorials.

-YouTube, Oracle Window Function Explained.

-Personal Experience, Depot Management(Family Business)