Project Plan:-Customer Behavior & Retention Analysis

Project Title: Customer Behavior and Retention Analysis using SQL

(Oracle SQL Developer)

Project Overview:

This project involves an in-depth analysis of customer shopping behavior aimed at uncovering patterns related to customer retention, segmentation, and purchase trends. The analysis simulates a real-world scenario where a Data Analyst working in an e-commerce company leverages SQL to derive actionable business insights from customer transaction data.

Key Objectives:

- Understand and interpret customer purchasing behavior.
- Identify high-value customers through **RFM (Recency, Frequency, Monetary)** segmentation.
- Calculate **Customer Lifetime Value (CLV)** to prioritize customer engagement strategies.
- Detect early signs of **customer churn** and identify at-risk customers.
- Deliver actionable recommendations for improving customer retention and driving business growth.

Tools & Technologies Used:

- **Database System:** Oracle SQL Developer
- **Language:** Structured Query Language (SQL)
- **Data Source:** Cleaned subset (300+ rows) of the original e-commerce transaction dataset (shopping_behavior_updated.csv.xlsx)

Project Outcomes:

- Successfully demonstrated the ability to work with large and complex datasets using only SQL.
- Gained hands-on experience with key analytical concepts like RFM, CLV, and churn prediction.
- Developed optimized, industry-standard SQL queries and data models for business intelligence use cases.
- Proved strong data interpretation and storytelling capabilities through structured, query-based reporting.

Why This Project Matters:

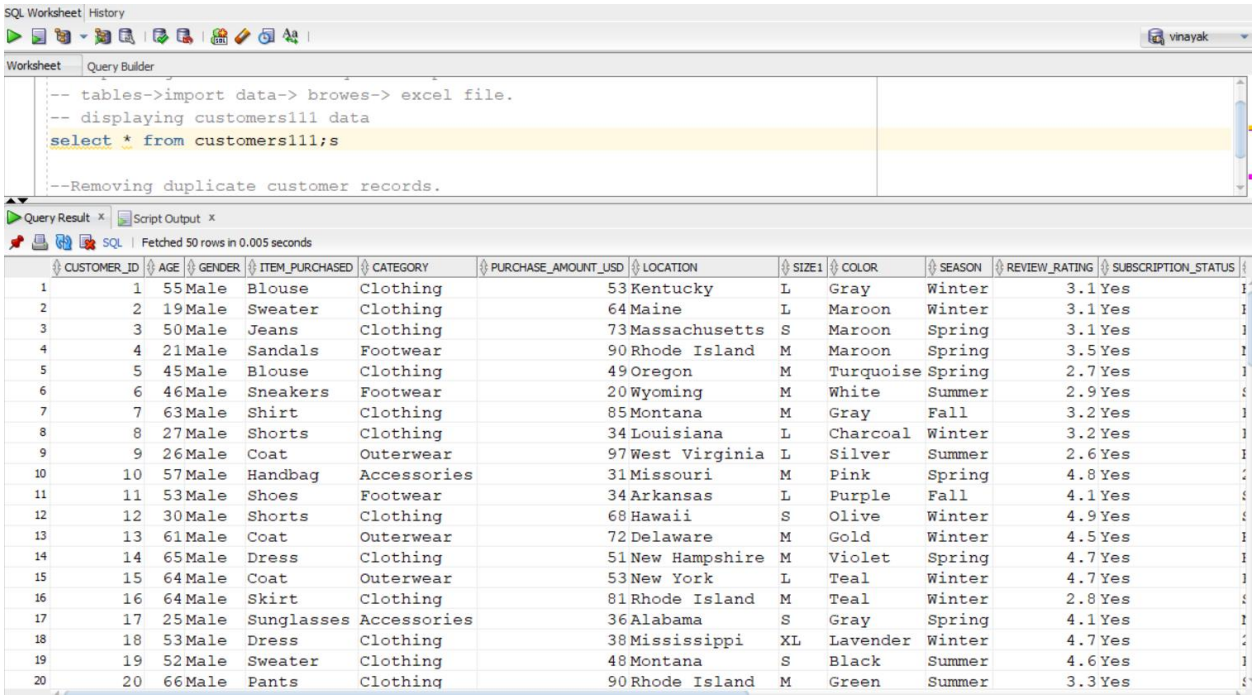
This project replicates the end-to-end responsibilities of a Data Analyst in an actual business setting. From raw data extraction to presenting high-impact customer insights, it showcases not just technical SQL proficiency, but also a deep understanding of business analysis and decision-making. It reflects my readiness for roles in data analytics, business intelligence, and customer insights teams, particularly in the retail and e-commerce industries.

- importing excel data in sql developer .

-- tables->import data-> browes-> excel file.

- -- displaying customers111 data

select * from customers111;

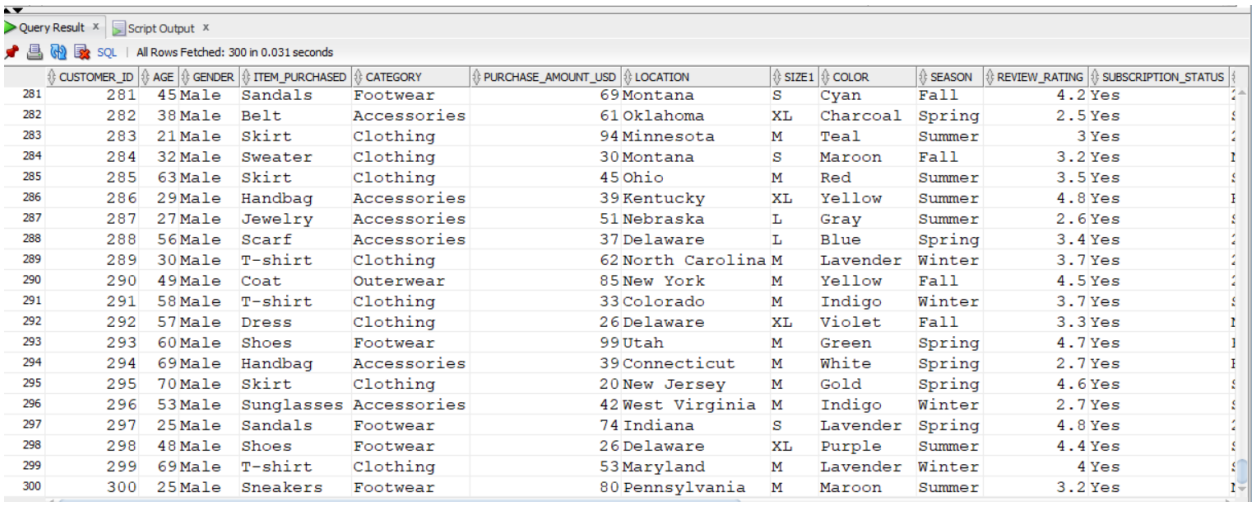


```
-- tables->import data-> browes-> excel file.
-- displaying customers111 data
select * from customers111;s

--Removing duplicate customer records.
```

Query Result | Script Output

Fetched 50 rows in 0.005 seconds

	CUSTOMER_ID	AGE	GENDER	ITEM_PURCHASED	CATEGORY	PURCHASE_AMOUNT_USD	LOCATION	SIZE1	COLOR	SEASON	REVIEW_RATING	SUBSCRIPTION_STATUS
1	1	55	Male	Blouse	Clothing	53	Kentucky	L	Gray	Winter	3.1	Yes
2	2	19	Male	Sweater	Clothing	64	Maine	L	Maroon	Winter	3.1	Yes
3	3	50	Male	Jeans	Clothing	73	Massachusetts	S	Maroon	Spring	3.1	Yes
4	4	21	Male	Sandals	Footwear	90	Rhode Island	M	Maroon	Spring	3.5	Yes
5	5	45	Male	Blouse	Clothing	49	Oregon	M	Turquoise	Spring	2.7	Yes
6	6	46	Male	Sneakers	Footwear	20	Wyoming	M	White	Summer	2.9	Yes
7	7	63	Male	Shirt	Clothing	85	Montana	M	Gray	Fall	3.2	Yes
8	8	27	Male	Shorts	Clothing	34	Louisiana	L	Charcoal	Winter	3.2	Yes
9	9	26	Male	Coat	Outerwear	97	West Virginia	L	Silver	Summer	2.6	Yes
10	10	57	Male	Handbag	Accessories	31	Missouri	M	Pink	Spring	4.8	Yes
11	11	53	Male	Shoes	Footwear	34	Arkansas	L	Purple	Fall	4.1	Yes
12	12	30	Male	Shorts	Clothing	68	Hawaii	S	Olive	Winter	4.9	Yes
13	13	61	Male	Coat	Outerwear	72	Delaware	M	Gold	Winter	4.5	Yes
14	14	65	Male	Dress	Clothing	51	New Hampshire	M	Violet	Spring	4.7	Yes
15	15	64	Male	Coat	Outerwear	53	New York	L	Teal	Winter	4.7	Yes
16	16	64	Male	Skirt	Clothing	81	Rhode Island	M	Teal	Winter	2.8	Yes
17	17	25	Male	Sunglasses	Accessories	36	Alabama	S	Gray	Spring	4.1	Yes
18	18	53	Male	Dress	Clothing	38	Mississippi	XL	Lavender	Winter	4.7	Yes
19	19	52	Male	Sweater	Clothing	48	Montana	S	Black	Summer	4.6	Yes
20	20	66	Male	Pants	Clothing	90	Rhode Island	M	Green	Summer	3.3	Yes



Query Result | Script Output

All Rows Fetched: 300 in 0.031 seconds

	CUSTOMER_ID	AGE	GENDER	ITEM_PURCHASED	CATEGORY	PURCHASE_AMOUNT_USD	LOCATION	SIZE1	COLOR	SEASON	REVIEW_RATING	SUBSCRIPTION_STATUS
281	281	45	Male	Sandals	Footwear	69	Montana	S	Cyan	Fall	4.2	Yes
282	282	38	Male	Belt	Accessories	61	Oklahoma	XL	Charcoal	Spring	2.5	Yes
283	283	21	Male	Skirt	Clothing	94	Minnesota	M	Teal	Summer	3	Yes
284	284	32	Male	Sweater	Clothing	30	Montana	S	Maroon	Fall	3.2	Yes
285	285	63	Male	Skirt	Clothing	45	Ohio	M	Red	Summer	3.5	Yes
286	286	29	Male	Handbag	Accessories	39	Kentucky	XL	Yellow	Summer	4.8	Yes
287	287	27	Male	Jewelry	Accessories	51	Nebraska	L	Gray	Summer	2.6	Yes
288	288	56	Male	Scarf	Accessories	37	Delaware	L	Blue	Spring	3.4	Yes
289	289	30	Male	T-shirt	Clothing	62	North Carolina	M	Lavender	Winter	3.7	Yes
290	290	49	Male	Coat	Outerwear	85	New York	M	Yellow	Fall	4.5	Yes
291	291	58	Male	T-shirt	Clothing	33	Colorado	M	Indigo	Winter	3.7	Yes
292	292	57	Male	Dress	Clothing	26	Delaware	XL	Violet	Fall	3.3	Yes
293	293	60	Male	Shoes	Footwear	99	Utah	M	Green	Spring	4.7	Yes
294	294	69	Male	Handbag	Accessories	39	Connecticut	M	White	Spring	2.7	Yes
295	295	70	Male	Skirt	Clothing	20	New Jersey	M	Gold	Spring	4.6	Yes
296	296	53	Male	Sunglasses	Accessories	42	West Virginia	M	Indigo	Winter	2.7	Yes
297	297	25	Male	Sandals	Footwear	74	Indiana	S	Lavender	Spring	4.8	Yes
298	298	48	Male	Shoes	Footwear	26	Delaware	XL	Purple	Summer	4.4	Yes
299	299	69	Male	T-shirt	Clothing	53	Maryland	M	Lavender	Winter	4	Yes
300	300	25	Male	Sneakers	Footwear	80	Pennsylvania	M	Maroon	Summer	3.2	Yes

- Removing duplicate customer records.

delete from customers111 where rowid not in (

select min(rowid) from customers111 group by customer_id);

```
--Removing duplicate customer records.
delete from customers111 where rowid not in (
select min(rowid) from customers111 group by customer_id);
```

Query Result | Script Output

Task completed in 0.049 seconds

```
0 rows deleted.


Error starting at line : 9 in command -
)
Error report -
Unknown Command
```

- Handling NULLs:-

-- Replace NULLs in REVIEW_RATING with average rating

update customers111

set review_rating=(select round(avg(review_rating),1)from customers111

where review_rating is null);

```
-- Replace NULLs in REVIEW_RATING with average rating
update customers111
set review_rating=(select round(avg(review_rating),1) from customers111 where review_rating is null);
```

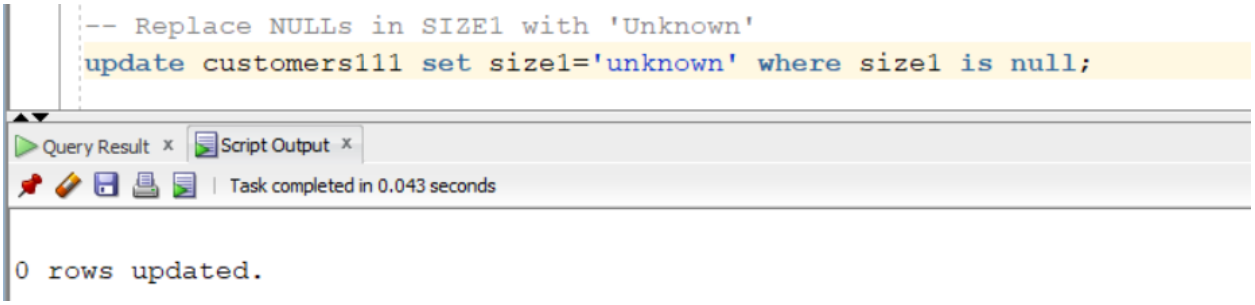
Query Result | Script Output

Task completed in 0.079 seconds

```
https://docs.oracle.com/error-help/db/ora-00921/00921. 00000 -  "unexpected end of SQL command"
*Cause:    The SQL command was not complete. Part of a valid
           command was entered, but at least one major component was
           omitted.
*Action:   Correct the syntax.

More Details :
https://docs.oracle.com/error-help/db/ora-00921/

300 rows updated.
```

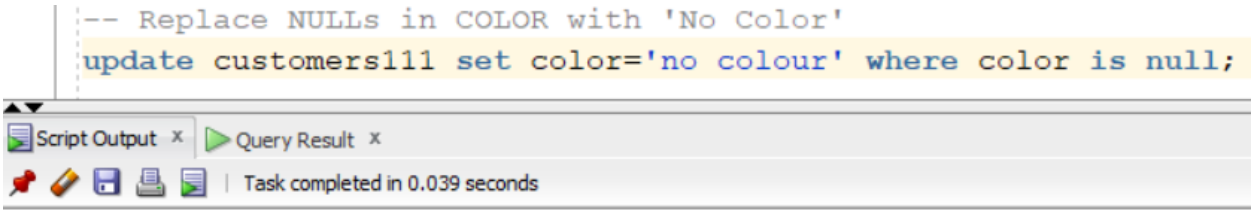
-- Replace NULLs in SIZE1 with 'Unknown'

update customers111 set size1='unknown' where size1 is null;



-- Replace NULLs in COLOR with 'No Color'

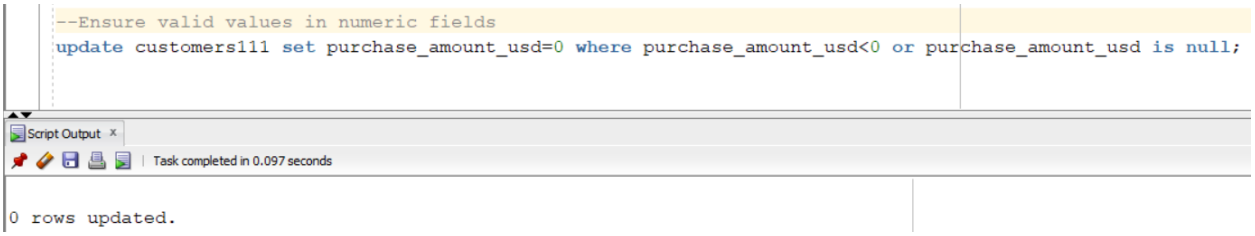
update customers111 set color='no colour' where color is null;



--Ensure valid values in numeric fields

-- Setting negative or NULL purchase amounts to 0

update customers111 set purchase_amount_usd=0 where purchase_amount_usd<0 or purchase_amount_usd is null;



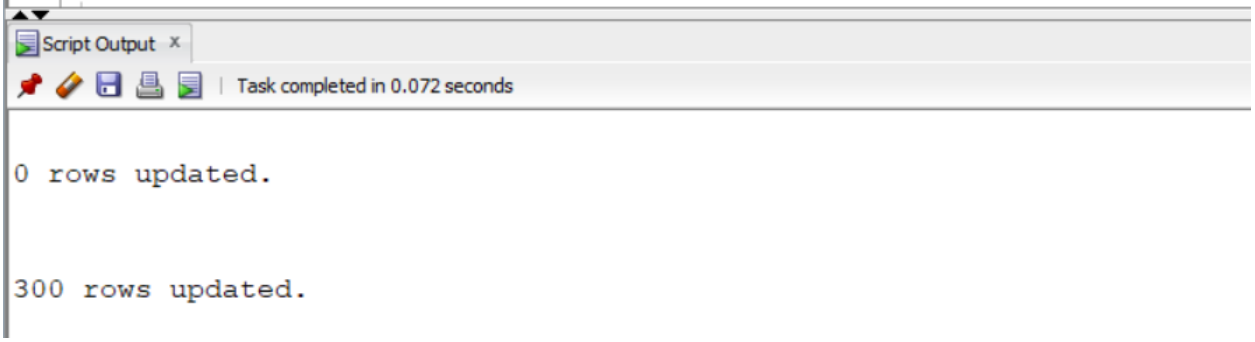
---- Trim and capitalizing GENDER and CATEGORY

update customers111 set gender = initcap(trim(gender)),

category = initcap(trim(category)),

location = initcap(trim(location));

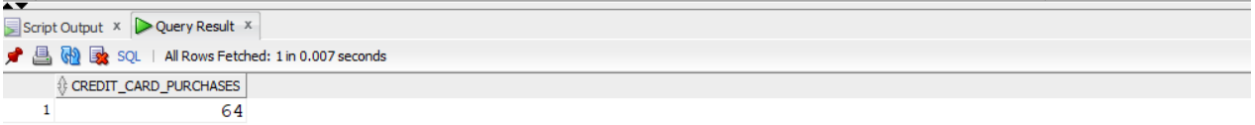
```
---- Trim and capitalizing GENDER and CATEGORY
update customers111 set gender = initcap(trim(gender)),
category = initcap(trim(category)),
location = initcap(trim(location));
```



--Find the total number of purchases made using the 'Credit Card' payment method.

select count(*) as credit_card_purchases from customers111 where PAYMENT_METHOD = 'Credit Card';

```
--Find the total number of purchases made using the 'Credit Card' payment method.
select count(*) as credit_card_purchases from customers111 where PAYMENT_METHOD  = 'Credit Card';
```



--Show the names of all products purchased in the 'Clothing' category.

select item_purchased from customers111 where category = 'Clothing';

```
--Show the names of all products purchased in the 'Clothing' category.
select item_purchased from customers111 where category = 'Clothing';
```

Script Output | Query Result | SQL | All Rows Fetched: 127 in 0.018 seconds

	ITEM_PURCHASED
1	Blouse
2	Sweater
3	Jeans
4	Blouse
5	Shirt
6	Shorts
7	Shorts
8	Dress
9	Skirt
10	Dress
11	Sweater
12	Pants
13	Pants
14	Pants
15	Pants
16	Pants
17	Hoodie
18	Shorts
19	Dress
20	Dress
21	Pants

--Display names of all customers who bought items in 'Spring' season and paid more than ₹50.

select customer_id, item_purchased, purchase_amount_usd

from customers111

where season = 'Spring' and purchase_amount_usd > 50;

```
--Display names of all customers who bought items in 'Spring' season and paid more than ₹50.
select customer_id, item_purchased, purchase_amount_usd
from customers111
where season = 'Spring' and purchase_amount_usd > 50;
```

Script Output | Query Result | SQL | All Rows Fetched: 41 in 0.008 seconds

	CUSTOMER_ID	ITEM_PURCHASED	PURCHASE_AMOUNT_USD
1	3	Jeans	73
2	4	Sandals	90
3	14	Dress	51
4	35	T-shirt	91
5	43	Coat	100
6	60	Belt	79
7	74	Pants	85
8	75	Gloves	58
9	79	Skirt	91
10	95	Skirt	76
11	97	Gloves	73
12	99	Jeans	67
13	102	Coat	85
14	109	Socks	79

-- List distinct cities where purchases were made.

select distinct location from customers111;

```
-- List distinct cities where purchases were made
select distinct location from customers111;
```

Script Output x | Query Result x

SQL | All Rows Fetched: 50 in 0.025 seconds

	LOCATION
1	Delaware
2	Georgia
3	Michigan
4	Maine
5	Rhode Island
6	Louisiana
7	Florida
8	North Dakota
9	Idaho
10	Wisconsin
11	Kentucky
12	Nevada
13	Indiana
14	Alaska
15	Tennessee
16	Pennsylvania
17	South Dakota
18	Oregon
19	Hawaii
20	Mississippi
21	North Carolina

--Show total revenue generated by each category.

select category, sum(purchase_amount_usd) as total_revenue

from customers111

group by category;

```
--Show total revenue generated by each category.
select category, sum(purchase_amount_usd) as total_revenue
from customers111
group by category;
```

Script Output | Query Result | SQL | All Rows Fetched: 4 in 0.014 seconds

	CATEGORY	TOTAL_REVENUE
1	Outerwear	1996
2	Clothing	7056
3	Accessories	6018
4	Footwear	2705

--Show average review rating by gender and season.

select gender, season, round(avg(review_rating), 2) as avg_rating

from customers111

group by gender, season;

```
--Show average review rating by gender and season.
select gender, season, round(avg(review_rating), 2) as avg_rating
from customers111
group by gender, season;
```

Script Output | Query Result | SQL | All Rows Fetched: 4 in 0.019 seconds

	GENDER	SEASON	AVG_RATING
1	Male	Fall	3.81
2	Male	Summer	3.71
3	Male	Winter	3.83
4	Male	Spring	3.77

--Rank customers based on their purchase amount (highest first).

```
select customer_id, purchase_amount_usd,rank()
over (order by purchase_amount_usd desc) as purchase_rank
from customers111;
```

Worksheet | Query Builder

```
--Rank customers based on their purchase amount (highest first).
select customer_id, purchase_amount_usd,rank()
over (order by purchase_amount_usd desc) as purchase_rank
from customers111;
```

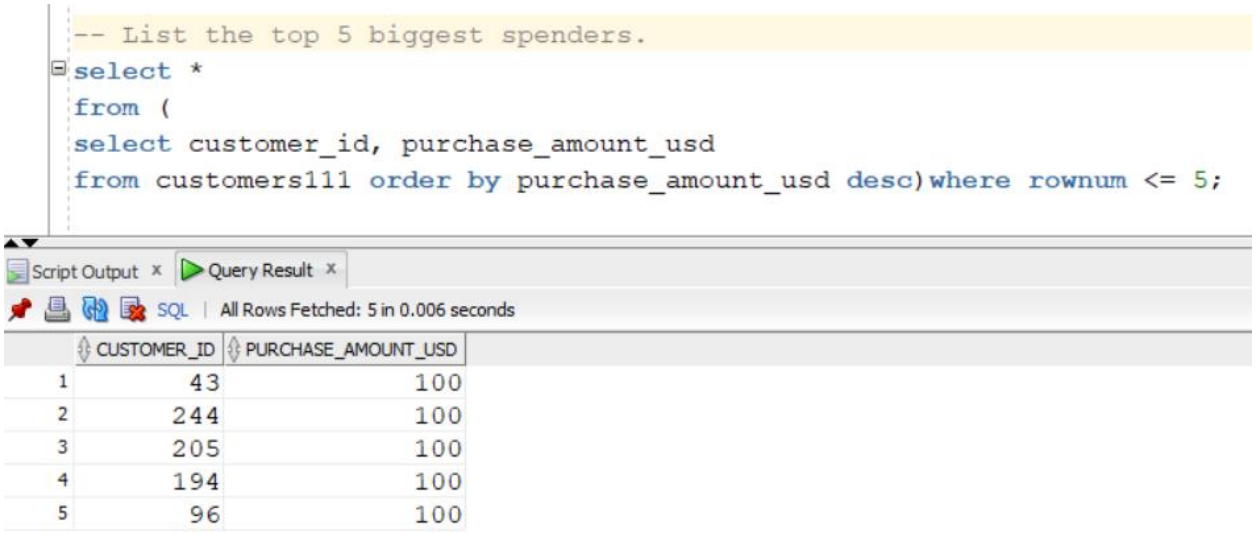
Script Output | Query Result

SQL | Fetched 50 rows in 0.007 seconds

	CUSTOMER_ID	PURCHASE_AMOUNT_USD	PURCHASE_RANK
1	43	100	1
2	96	100	1
3	249	100	1
4	205	100	1
5	244	100	1
6	194	100	1
7	92	99	7
8	293	99	7
9	101	98	9
10	155	98	9
11	9	97	11
12	210	97	11
13	151	96	13
14	183	96	13
15	106	96	13
16	82	96	13
17	80	96	13
18	140	95	18
19	146	95	18
20	115	95	18
21	86	95	18

-- List the top 5 biggest spenders.

```
select *
from (
select customer_id, purchase_amount_usd
from customers111 order by purchase_amount_usd desc)where rownum <= 5;
```



--Create a view showing customer segments based on spend:

```
create or replace view customer_segments as select customer_id, purchase_amount_usd,
case
when purchase_amount_usd >= 75 then 'High'
when purchase_amount_usd between 40 and 74 then 'Medium'
else 'Low'
end as segment
from customers111;
```

```
View CUSTOMER_SEGMENTS created.
```

```
select * from customer_segments;
```

Query Result

SQL | Fetched 50 rows in 0.02 seconds

	CUSTOMER_ID	PURCHASE_AMOUNT_USD	SEGMENT
1	1	53	Medium
2	2	64	Medium
3	3	73	Medium
4	4	90	High
5	5	49	Medium
6	6	20	Low
7	7	85	High
8	8	34	Low
9	9	97	High
10	10	31	Low
11	11	34	Low
12	12	68	Medium
13	13	72	Medium
14	14	51	Medium
15	15	53	Medium
16	16	81	High
17	17	36	Low
18	18	38	Low
19	19	48	Medium
20	20	90	High

--List all items with 'a' at the second position in item name.

```
select item_purchased
from customers111
where item_purchased like '_a%';
```

```
--List all items with 'a' at the second position in item name.
select item_purchased
from customers111
where item_purchased like '_a%';
```

Query Result

SQL | Fetched 50 rows in 0.004 seconds

	ITEM_PURCHASED
1	Sandals
2	Handbag
3	Pants
4	Pants
5	Pants
6	Pants
7	Pants
8	Jacket
9	Handbag
10	Jacket

--Show total number of purchases per color, sorted descending.

```
select color, count(*) as total_sold
from customers111
group by color
order by total_sold desc;
```

```
--Show total number of purchases per color, sorted descending.
select color, count(*) as total_sold
from customers111
group by color
order by total_sold desc;
```

Query Result

SQL | All Rows Fetched: 25 in 0.001 seconds

	COLOR	TOTAL_SOLD
1	Gray	16
2	Violet	16
3	Olive	16
4	Maroon	15
5	Silver	15

--Find the most purchased category in each season

```
select season, category, count(*) as purchases,rank()
over (partition by season order by count(*) desc) as rank_in_season
from customers111
group by season, category;
```

Worksheet | Query Builder

```
select season, category, count(*) as purchases,rank()
over (partition by season order by count(*) desc) as rank_in_season
from customers111
group by season, category;
```

Query Result

SQL | All Rows Fetched: 16 in 0.006 seconds

	SEASON	CATEGORY	PURCHASES	RANK_IN_SEASON
1	Fall	Clothing	27	1
2	Fall	Accessories	25	2
3	Fall	Footwear	12	3
4	Fall	Outerwear	11	4
5	Spring	Clothing	30	1
6	Spring	Accessories	29	2
7	Spring	Footwear	12	3
8	Spring	Outerwear	5	4
9	Summer	Clothing	36	1
10	Summer	Accessories	25	2
11	Summer	Footwear	12	3
12	Summer	Outerwear	9	4
13	Winter	Clothing	34	1
14	Winter	Accessories	20	2
15	Winter	Outerwear	7	3
16	Winter	Footwear	6	4

--Total customers & total purchases

select count(distinct customer_id) as total_customers,

count(*) as total_purchases

from customers111;

```
--Total customers & total purchases
select count(distinct customer_id) as total_customers,
count(*) as total_purchases
from customers111;
```

Query Result

SQL | All Rows Fetched: 1 in 0.011 seconds

	TOTAL_CUSTOMERS	TOTAL_PURCHASES
1	300	300

--Top 3 most purchased product categories

select category, count(*) as total_sold

from customers111

group by category

order by total_sold desc fetch first 3 rows only;

```
select category, count(*) as total_sold
from customers111
group by category
order by total_sold desc fetch first 3 rows only;
```

Query Result

SQL | All Rows Fetched: 3 in 0.005 seconds

	CATEGORY	TOTAL_SOLD
1	Clothing	127
2	Accessories	99
3	Footwear	42

--Top 10 customers by spend using RANK

```
select customer_id, purchase_amount_usd,rank()
over (order by purchase_amount_usd desc) as purchase_rank
from customers111
fetch first 10 rows only;
```

Query Result

SQL | All Rows Fetched: 10 in 0.003 seconds

	CUSTOMER_ID	PURCHASE_AMOUNT_USD	PURCHASE_RANK
1	43	100	1
2	249	100	1
3	244	100	1
4	205	100	1
5	194	100	1
6	96	100	1
7	92	99	7
8	293	99	7
9	101	98	9
10	155	98	9

- JOINS OPERATIONS:-

ADDING TWO MORE TABLES RELETED TO MAIN TABLES.

```
select * from customer_details;
select * from products_master;
```

Script Output | Query Result | SQL | All Rows Fetched: 4 in 0.004 seconds

	CUSTOMER_ID	FULL_NAME	EMAIL	JOIN_DATE
1	101	Ravi Mehta	ravi@gmail.com	10-JAN-23
2	102	Aarti Rao	aarti@gmail.com	15-MAY-23
3	103	Vikas Shetty	vikas@gmail.com	28-FEB-24
4	104	Neha Yadav	neha@gmail.com	01-DEC-22

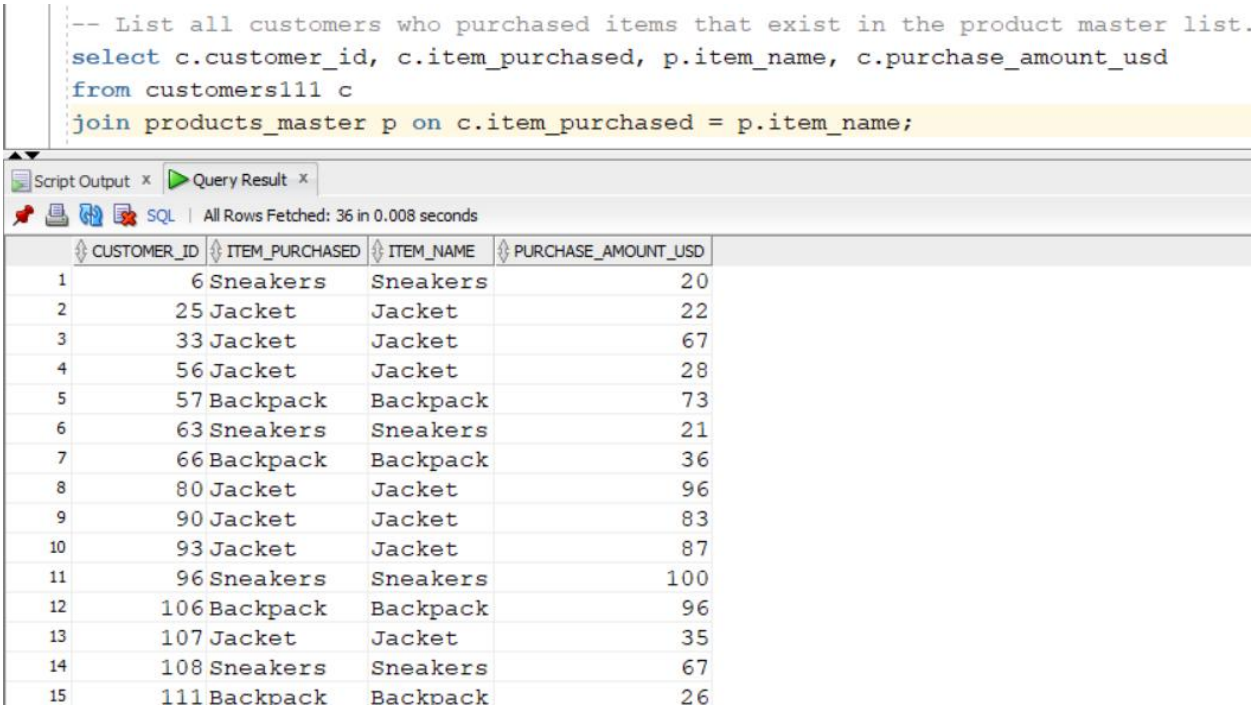
```
select * from products_master;

-- List all customers who purchased items tha
```

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0.004 seconds

	PRODUCT_ID	ITEM_NAME	CATEGORY	BASE_PRICE
1	101	T-Shirt	Clothing	15
2	102	Jacket	Clothing	50
3	103	Sneakers	Footwear	40
4	104	Backpack	Accessories	25
5	105	Watch	Accessories	100

-- List all customers who purchased items that exist in the product master list.

```
select c.customer_id, c.item_purchased, p.item_name, c.purchase_amount_usd
from customers111 c
join products_master p on c.item_purchased = p.item_name;
```



```
-- List all customers who purchased items that exist in the product master list.
select c.customer_id, c.item_purchased, p.item_name, c.purchase_amount_usd
from customers111 c
join products_master p on c.item_purchased = p.item_name;
```

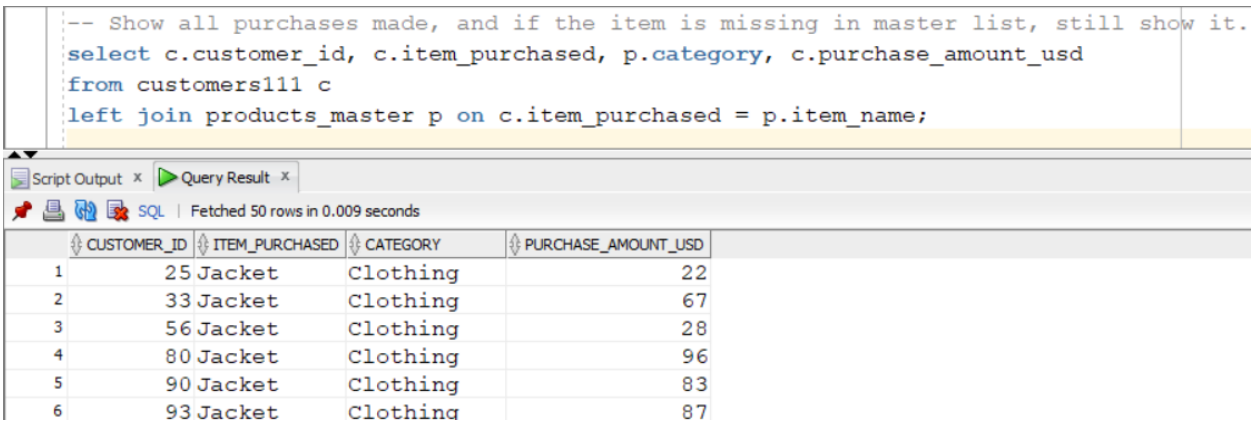
Script Output | Query Result

SQL | All Rows Fetched: 36 in 0.008 seconds

	CUSTOMER_ID	ITEM_PURCHASED	ITEM_NAME	PURCHASE_AMOUNT_USD
1	6	Sneakers	Sneakers	20
2	25	Jacket	Jacket	22
3	33	Jacket	Jacket	67
4	56	Jacket	Jacket	28
5	57	Backpack	Backpack	73
6	63	Sneakers	Sneakers	21
7	66	Backpack	Backpack	36
8	80	Jacket	Jacket	96
9	90	Jacket	Jacket	83
10	93	Jacket	Jacket	87
11	96	Sneakers	Sneakers	100
12	106	Backpack	Backpack	96
13	107	Jacket	Jacket	35
14	108	Sneakers	Sneakers	67
15	111	Backpack	Backpack	26

-- Show all purchases made, and if the item is missing in master list, still show it.

```
select c.customer_id, c.item_purchased, p.category, c.purchase_amount_usd
from customers111 c
left join products_master p on c.item_purchased = p.item_name;
```



```
-- Show all purchases made, and if the item is missing in master list, still show it.
select c.customer_id, c.item_purchased, p.category, c.purchase_amount_usd
from customers111 c
left join products_master p on c.item_purchased = p.item_name;
```

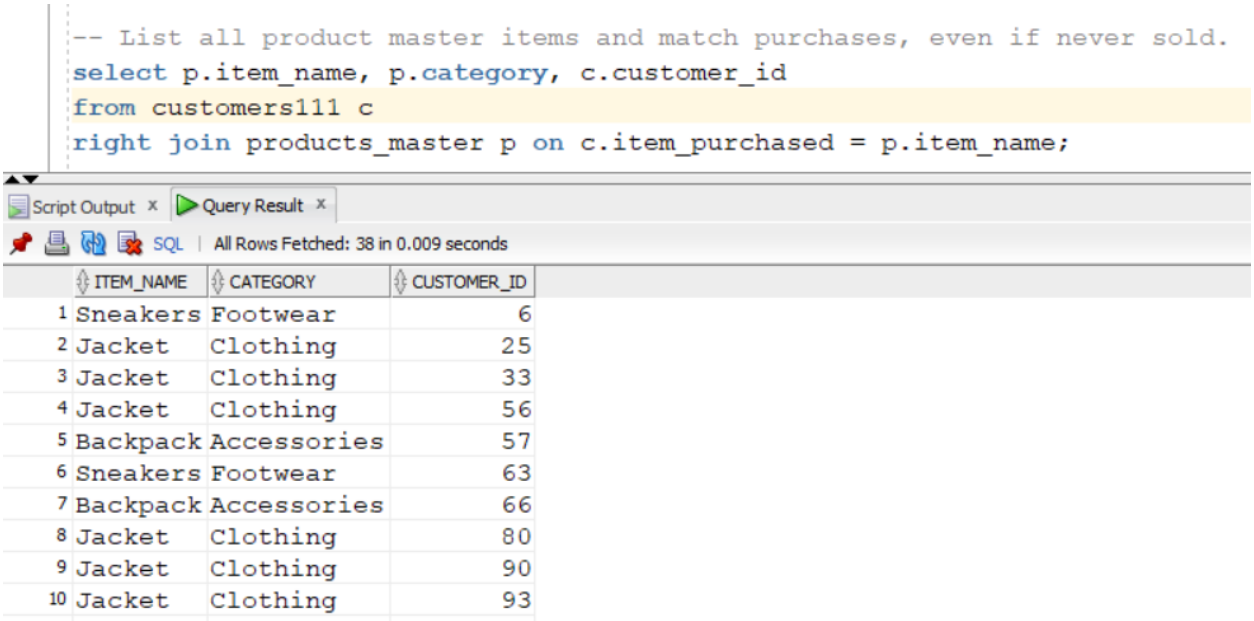
Script Output | Query Result

SQL | Fetched 50 rows in 0.009 seconds

	CUSTOMER_ID	ITEM_PURCHASED	CATEGORY	PURCHASE_AMOUNT_USD
1	25	Jacket	Clothing	22
2	33	Jacket	Clothing	67
3	56	Jacket	Clothing	28
4	80	Jacket	Clothing	96
5	90	Jacket	Clothing	83
6	93	Jacket	Clothing	87

-- List all product master items and match purchases, even if never sold.

```
select p.item_name, p.category, c.customer_id
from customers111 c
right join products_master p on c.item_purchased = p.item_name;
```



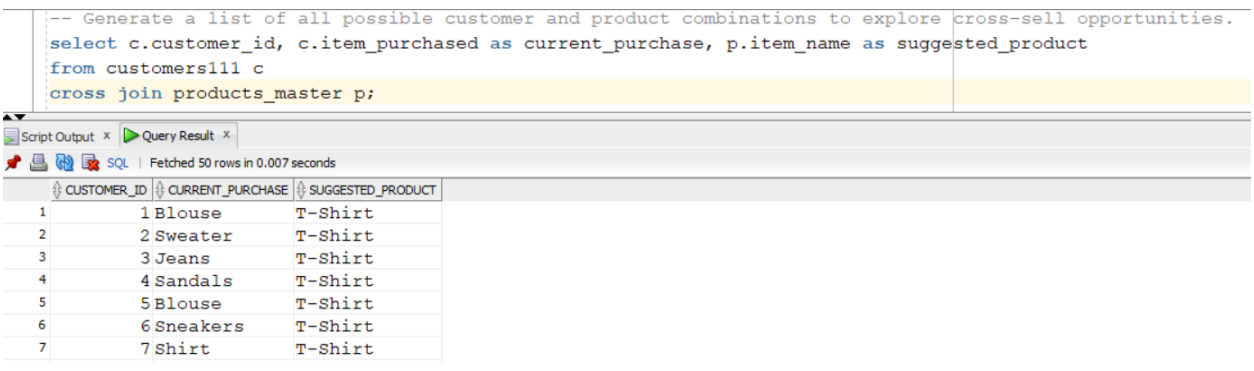
```
-- List all product master items and match purchases, even if never sold.
select p.item_name, p.category, c.customer_id
from customers111 c
right join products_master p on c.item_purchased = p.item_name;
```

Script Output | Query Result | SQL | All Rows Fetched: 38 in 0.009 seconds

	ITEM_NAME	CATEGORY	CUSTOMER_ID
1	Sneakers	Footwear	6
2	Jacket	Clothing	25
3	Jacket	Clothing	33
4	Jacket	Clothing	56
5	Backpack	Accessories	57
6	Sneakers	Footwear	63
7	Backpack	Accessories	66
8	Jacket	Clothing	80
9	Jacket	Clothing	90
10	Jacket	Clothing	93

-- Generate a list of all possible customer and product combinations to explore cross-sell opportunities.

```
select c.customer_id, c.item_purchased as current_purchase, p.item_name as suggested_product
from customers111 c
cross join products_master p;
```



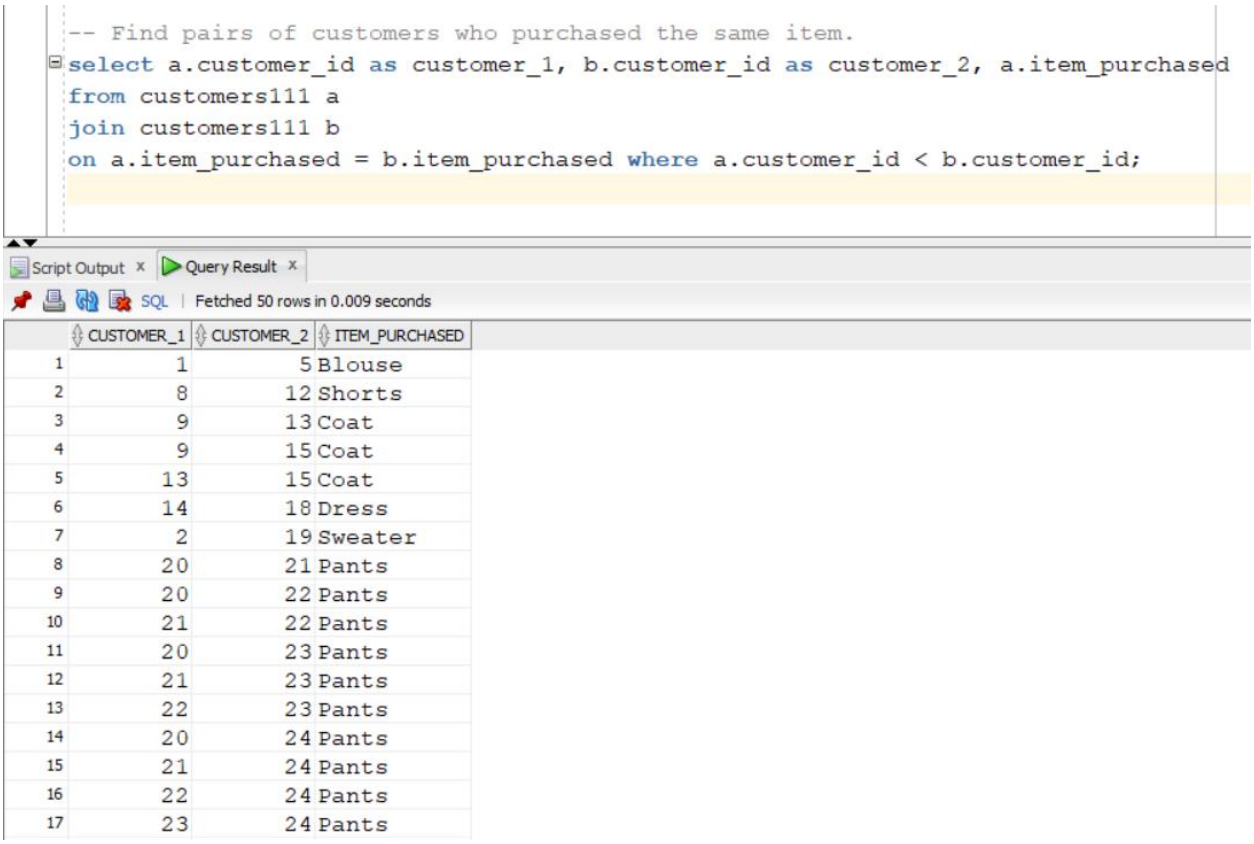
```
-- Generate a list of all possible customer and product combinations to explore cross-sell opportunities.
select c.customer_id, c.item_purchased as current_purchase, p.item_name as suggested_product
from customers111 c
cross join products_master p;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0.007 seconds

	CUSTOMER_ID	CURRENT_PURCHASE	SUGGESTED_PRODUCT
1	1	Blouse	T-Shirt
2	2	Sweater	T-Shirt
3	3	Jeans	T-Shirt
4	4	Sandals	T-Shirt
5	5	Blouse	T-Shirt
6	6	Sneakers	T-Shirt
7	7	Shirt	T-Shirt

-- Find pairs of customers who purchased the same item.

```
select a.customer_id as customer_1, b.customer_id as customer_2, a.item_purchased
from customers111 a
join customers111 b
on a.item_purchased = b.item_purchased where a.customer_id < b.customer_id;
```



```
-- Find pairs of customers who purchased the same item.
select a.customer_id as customer_1, b.customer_id as customer_2, a.item_purchased
from customers111 a
join customers111 b
on a.item_purchased = b.item_purchased where a.customer_id < b.customer_id;
```

Script Output | Query Result | SQL | Fetched 50 rows in 0.009 seconds

	CUSTOMER_1	CUSTOMER_2	ITEM_PURCHASED
1	1	5	Blouse
2	8	12	Shorts
3	9	13	Coat
4	9	15	Coat
5	13	15	Coat
6	14	18	Dress
7	2	19	Sweater
8	20	21	Pants
9	20	22	Pants
10	21	22	Pants
11	20	23	Pants
12	21	23	Pants
13	22	23	Pants
14	20	24	Pants
15	21	24	Pants
16	22	24	Pants
17	23	24	Pants

- ## Project Conclusion

In this SQL project, I imported raw customer shopping behavior data into Oracle SQL Developer and performed full data cleaning, formatting, and transformation using SQL.

I analyzed spending patterns, segmented customers, and used advanced SQL concepts like window functions, views, and case logic.

This project reflects real industry workflows and showcases my ability to turn raw data into actionable insights using SQL.

DATASET USED:- Consumer Behavior and Shopping Habits Dataset: