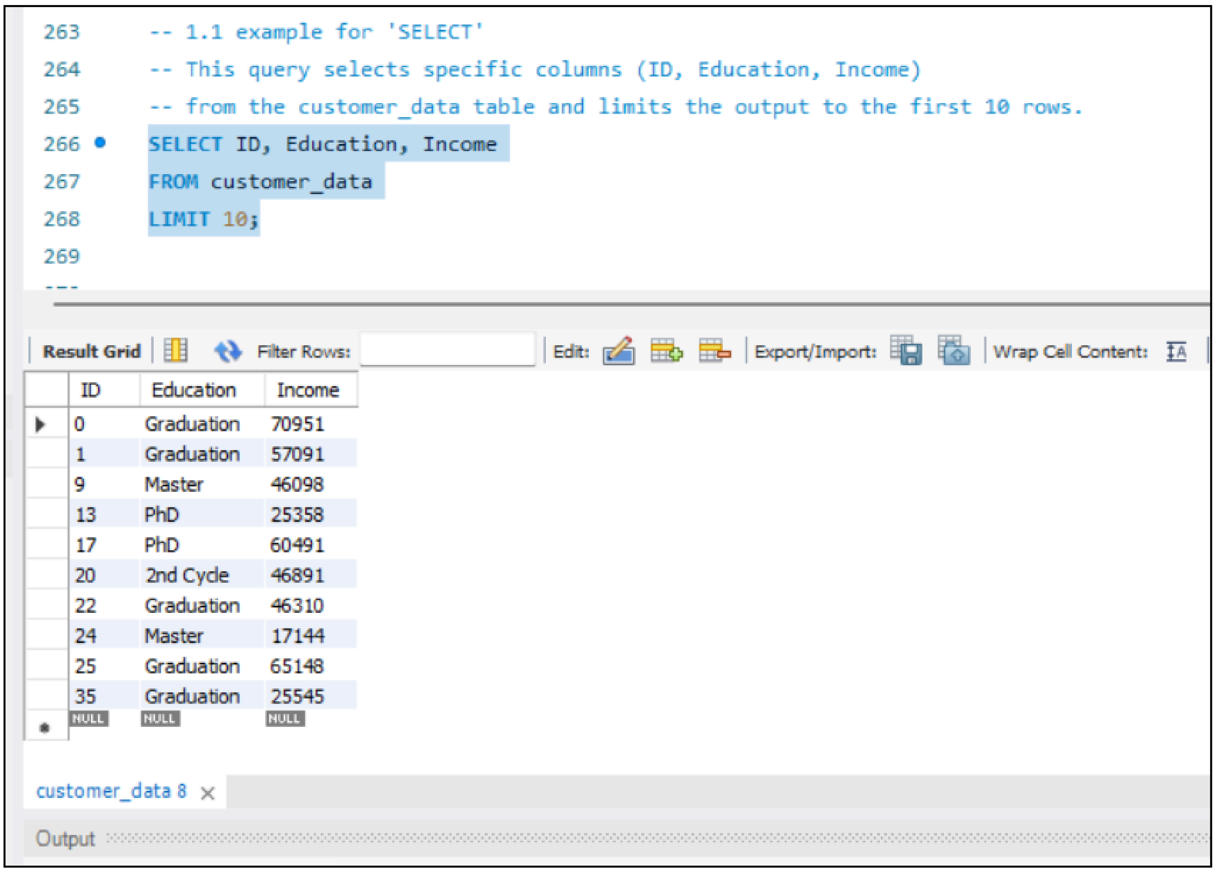
TASK 3_DATA ANALYST INTERNSHIP_ELIVATE LABS
- Anjali Gupta

a. Use SELECT, WHERE, ORDER BY, GROUP BY

1.1. Example for 'SELECT'
Output:

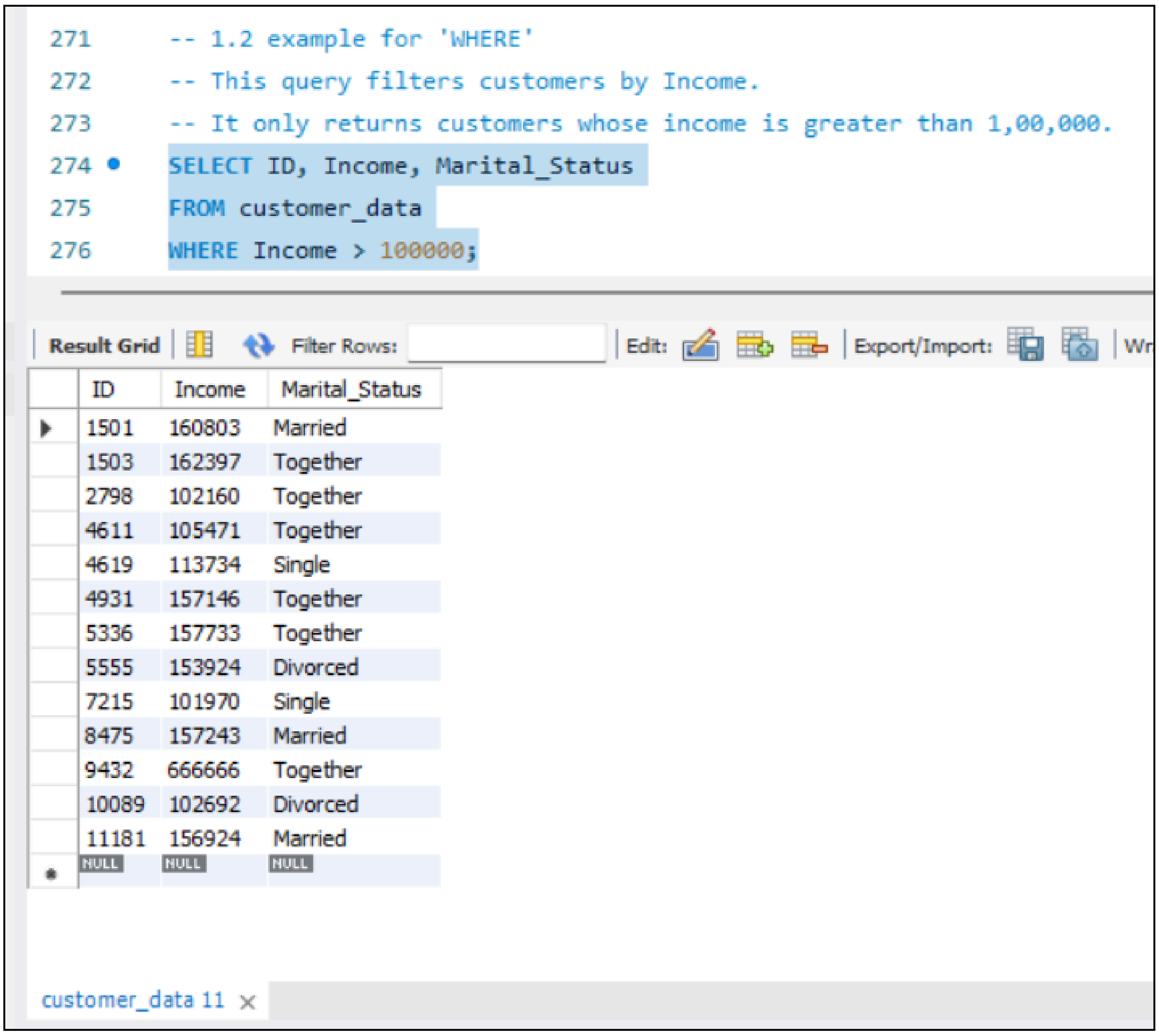


```
-- 1.1 example for 'SELECT'
-- This query selects specific columns (ID, Education, Income)
-- from the customer_data table and limits the output to the first 10 rows.
SELECT ID, Education, Income
FROM customer_data
LIMIT 10;
```

ID	Education	Income
0	Graduation	70951
1	Graduation	57091
9	Master	46098
13	PhD	25358
17	PhD	60491
20	2nd Cycle	46891
22	Graduation	46310
24	Master	17144
25	Graduation	65148
35	Graduation	25545
NULL	NULL	NULL

customer_data 8

1.2. Example for 'WHERE'
Output:

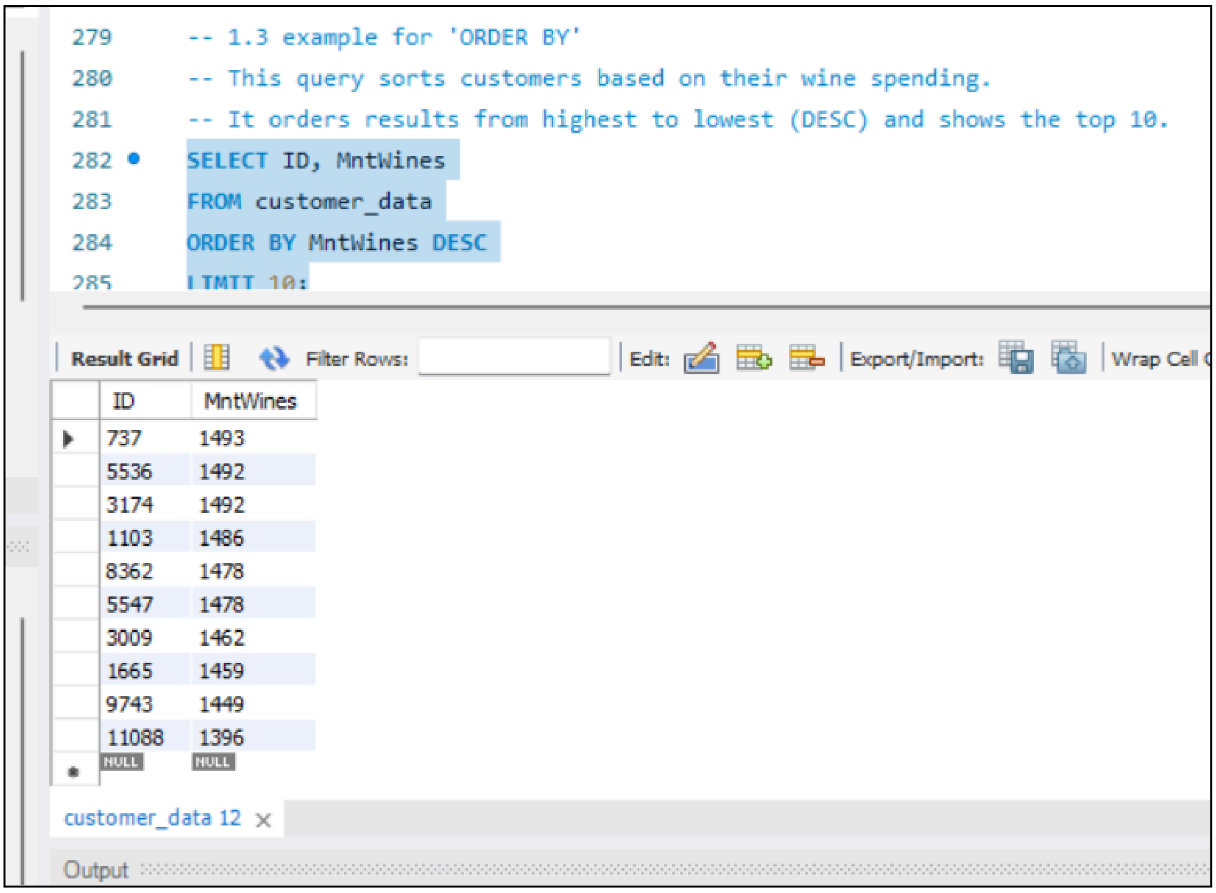


```sql
-- 1.2 example for 'WHERE'
-- This query filters customers by Income.
-- It only returns customers whose income is greater than 1,00,000.
SELECT ID, Income, Marital_Status
FROM customer_data
WHERE Income > 100000;
```

ID	Income	Marital_Status
1501	160803	Married
1503	162397	Together
2798	102160	Together
4611	105471	Together
4619	113734	Single
4931	157146	Together
5336	157733	Together
5555	153924	Divorced
7215	101970	Single
8475	157243	Married
9432	666666	Together
10089	102692	Divorced
11181	156924	Married
NULL	NULL	NULL

1.3. Example for 'ORDER BY'

Output:

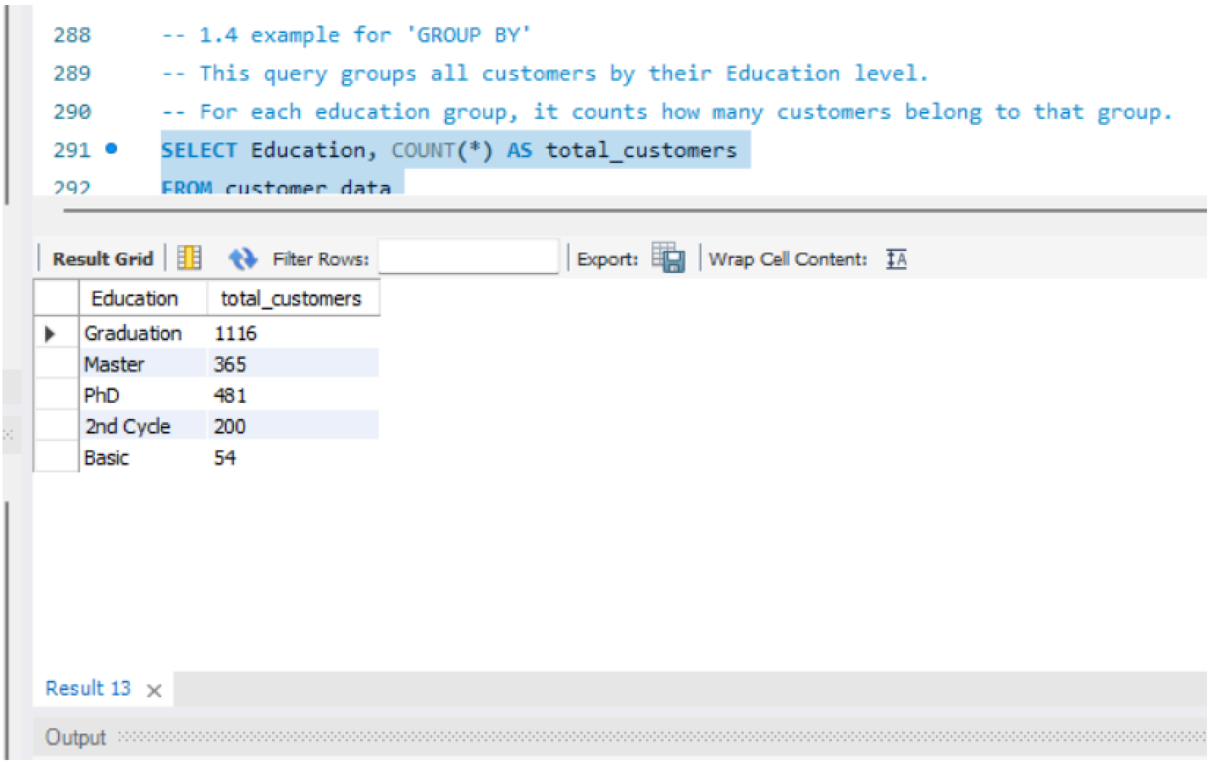


```
-- 1.3 example for 'ORDER BY'
-- This query sorts customers based on their wine spending.
-- It orders results from highest to lowest (DESC) and shows the top 10.
SELECT ID, MntWines
FROM customer_data
ORDER BY MntWines DESC
LIMIT 10;
```

ID	MntWines
737	1493
5536	1492
3174	1492
1103	1486
8362	1478
5547	1478
3009	1462
1665	1459
9743	1449
11088	1396
NULL	NULL

1.4. Example for 'GROUP BY'

Output:



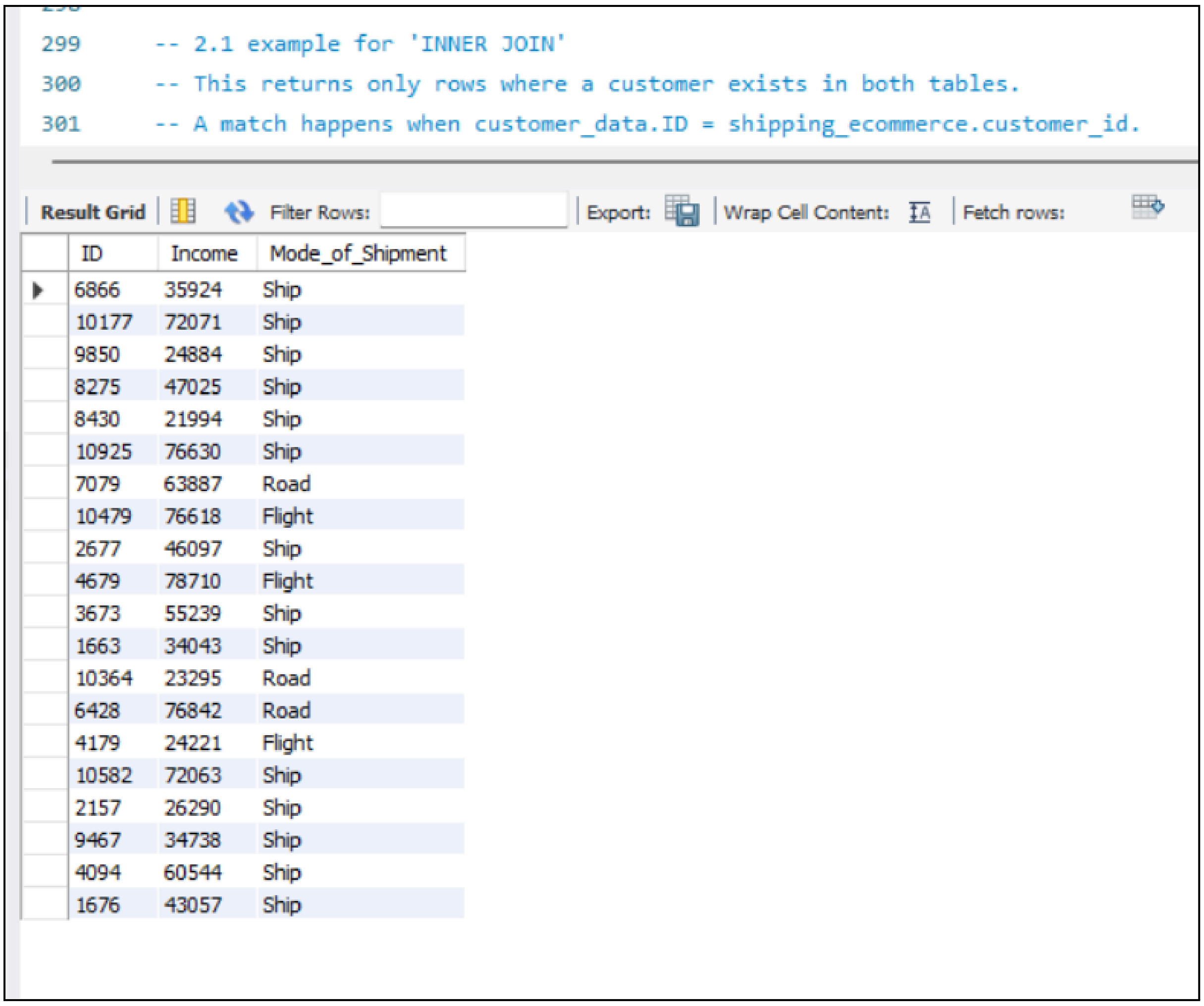
```
-- 1.4 example for 'GROUP BY'
-- This query groups all customers by their Education level.
-- For each education group, it counts how many customers belong to that group.
SELECT Education, COUNT(*) AS total_customers
FROM customer_data
```

Education	total_customers
Graduation	1116
Master	365
PhD	481
2nd Cycle	200
Basic	54

b. <u>Use JOINS (INNER, LEFT, RIGHT)</u>

2.1. Example for 'INNER JOIN'

Output:

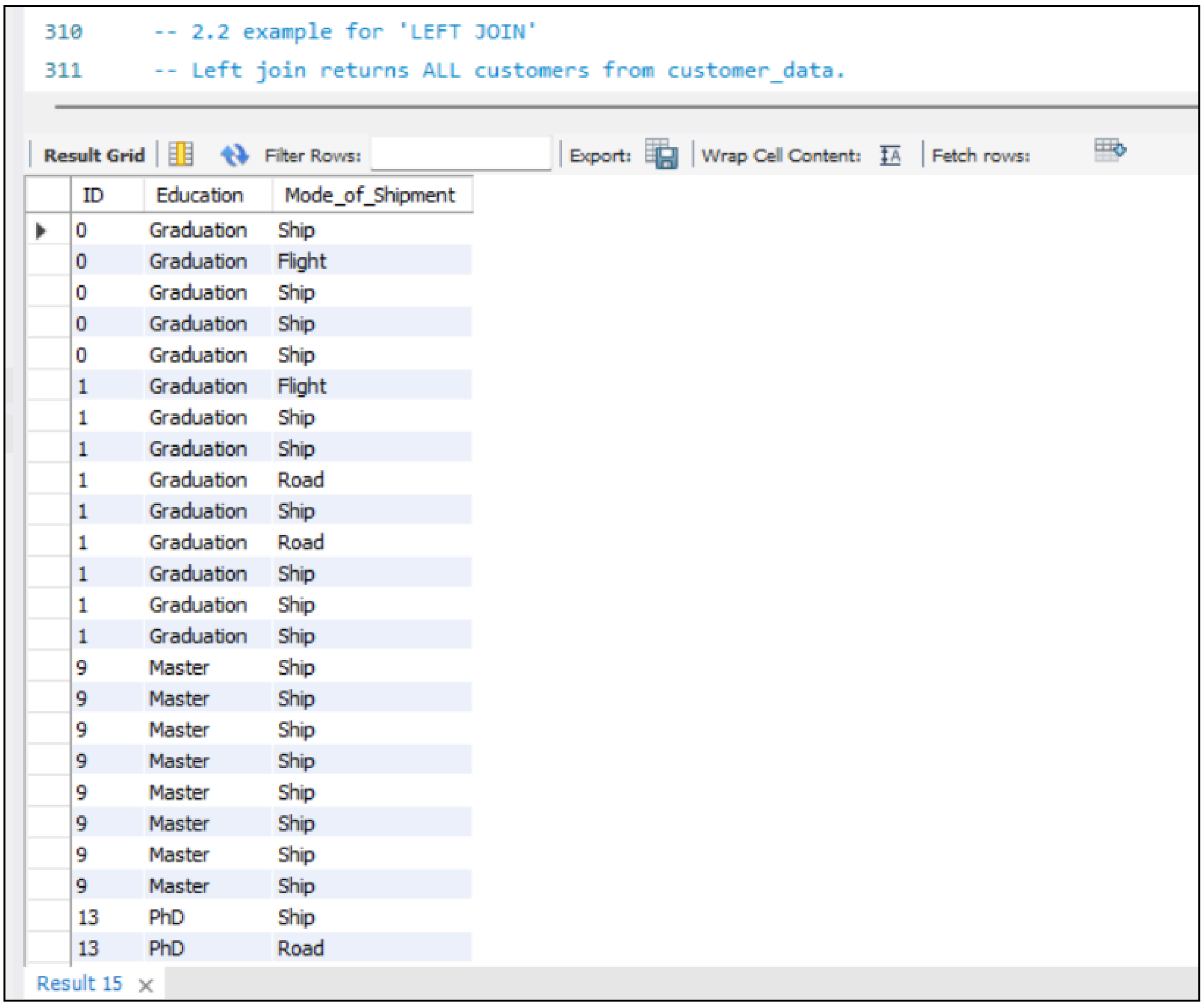


```
-- 2.1 example for 'INNER JOIN'
-- This returns only rows where a customer exists in both tables.
-- A match happens when customer_data.ID = shipping_ecommerce.customer_id.
```

ID	Income	Mode_of_Shipment
6866	35924	Ship
10177	72071	Ship
9850	24884	Ship
8275	47025	Ship
8430	21994	Ship
10925	76630	Ship
7079	63887	Road
10479	76618	Flight
2677	46097	Ship
4679	78710	Flight
3673	55239	Ship
1663	34043	Ship
10364	23295	Road
6428	76842	Road
4179	24221	Flight
10582	72063	Ship
2157	26290	Ship
9467	34738	Ship
4094	60544	Ship
1676	43057	Ship

2.2. Example for 'LEFT JOIN'

Output:



```
-- 2.2 example for 'LEFT JOIN'
-- Left join returns ALL customers from customer_data.
```

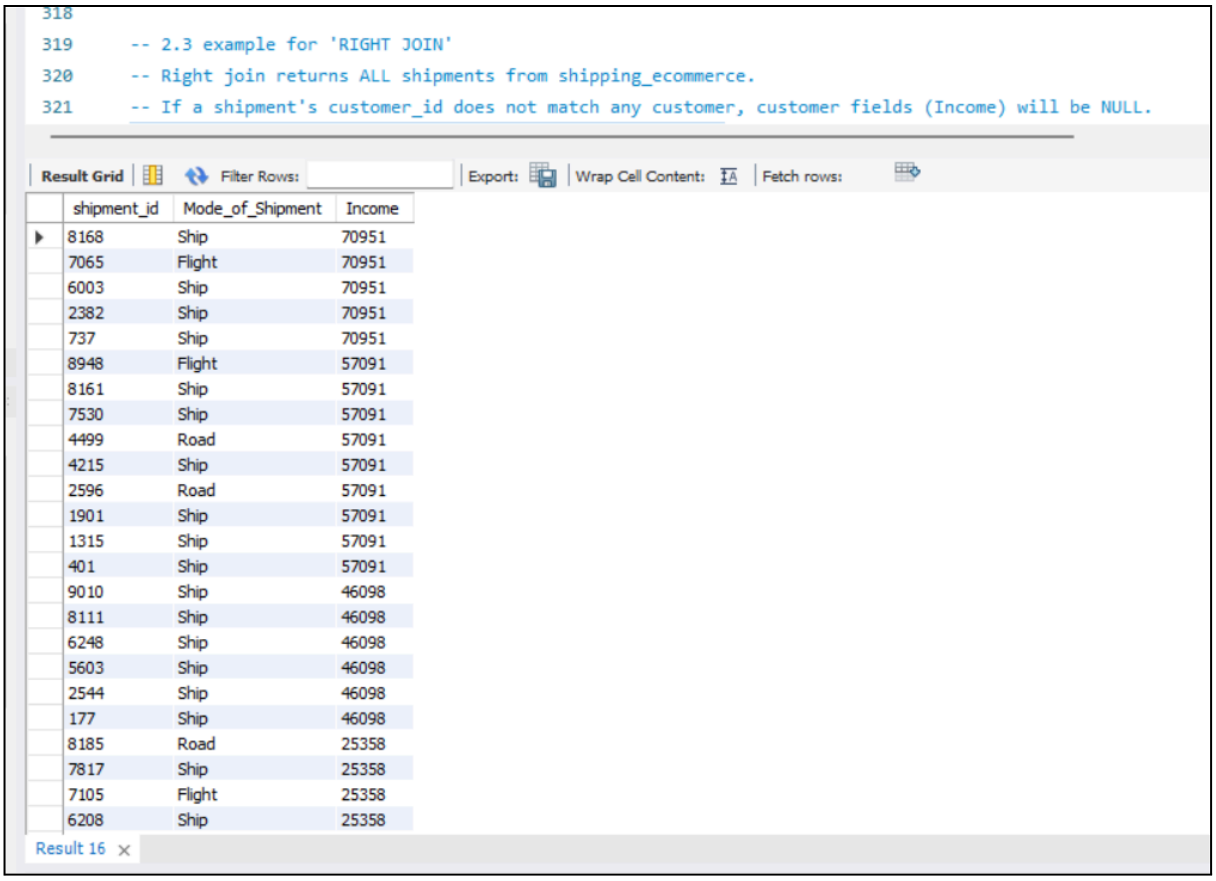
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

ID	Education	Mode_of_Shipment
0	Graduation	Ship
0	Graduation	Flight
0	Graduation	Ship
0	Graduation	Ship
0	Graduation	Ship
1	Graduation	Flight
1	Graduation	Ship
1	Graduation	Ship
1	Graduation	Road
1	Graduation	Ship
1	Graduation	Road
1	Graduation	Ship
1	Graduation	Ship
1	Graduation	Ship
9	Master	Ship
9	Master	Ship
9	Master	Ship
9	Master	Ship
9	Master	Ship
9	Master	Ship
9	Master	Ship
9	Master	Ship
13	PhD	Ship
13	PhD	Road

Result 15

2.3. Example for 'RIGHT JOIN'

Output:



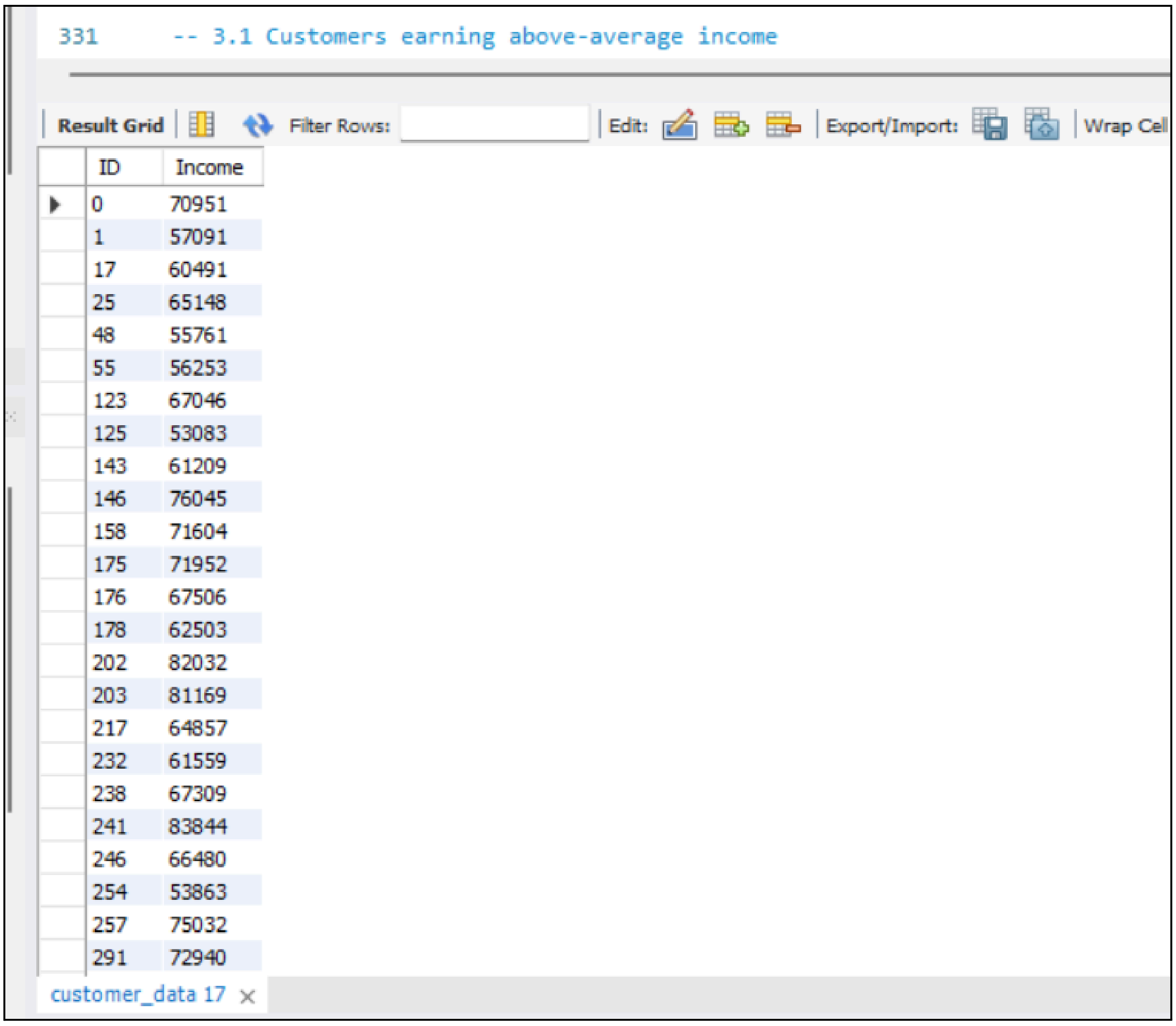
```
-- 2.3 example for 'RIGHT JOIN'
-- Right join returns ALL shipments from shipping_ecommerce.
-- If a shipment's customer_id does not match any customer, customer fields (Income) will be NULL.
```

shipment_id	Mode_of_Shipment	Income
8168	Ship	70951
7065	Flight	70951
6003	Ship	70951
2382	Ship	70951
737	Ship	70951
8948	Flight	57091
8161	Ship	57091
7530	Ship	57091
4499	Road	57091
4215	Ship	57091
2596	Road	57091
1901	Ship	57091
1315	Ship	57091
401	Ship	57091
9010	Ship	46098
8111	Ship	46098
6248	Ship	46098
5603	Ship	46098
2544	Ship	46098
177	Ship	46098
8185	Road	25358
7817	Ship	25358
7105	Flight	25358
6208	Ship	25358

c. <u>Write subqueries</u>

3.1. Customers earning above-average income

Output:



```
331        -- 3.1 Customers earning above-average income
```

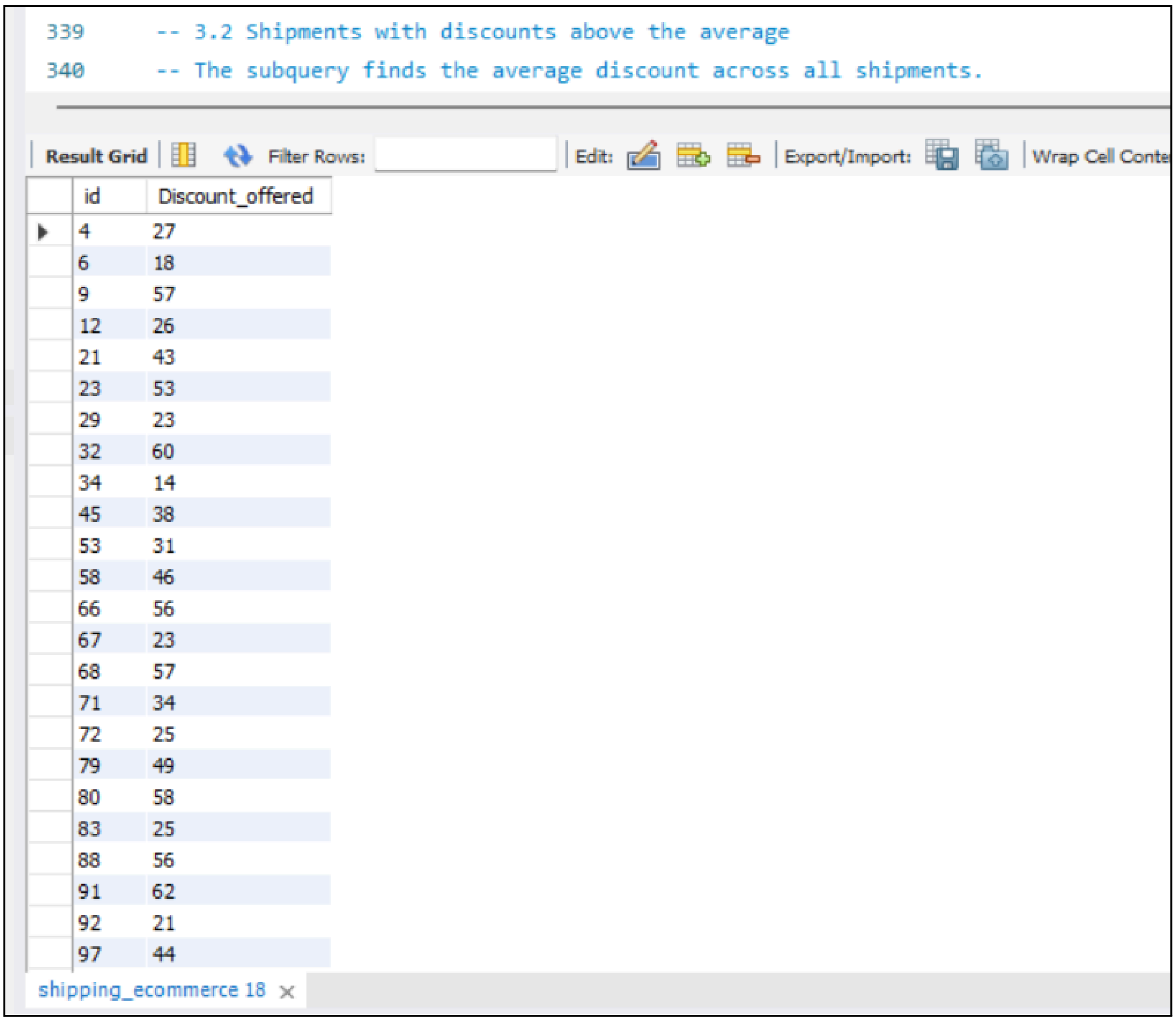
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell

ID	Income
0	70951
1	57091
17	60491
25	65148
48	55761
55	56253
123	67046
125	53083
143	61209
146	76045
158	71604
175	71952
176	67506
178	62503
202	82032
203	81169
217	64857
232	61559
238	67309
241	83844
246	66480
254	53863
257	75032
291	72940

customer_data 17

3.2. Shipments with discounts above the average

Output:

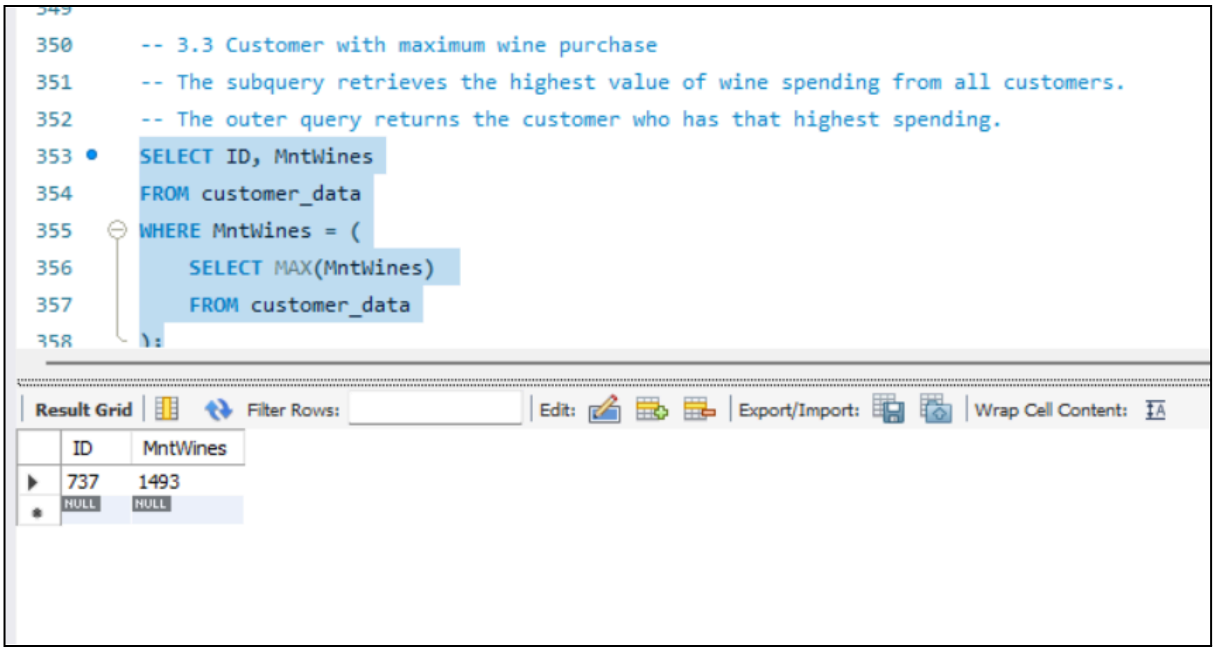


```
-- 3.2 Shipments with discounts above the average
-- The subquery finds the average discount across all shipments.
```

id	Discount_offered
4	27
6	18
9	57
12	26
21	43
23	53
29	23
32	60
34	14
45	38
53	31
58	46
66	56
67	23
68	57
71	34
72	25
79	49
80	58
83	25
88	56
91	62
92	21
97	44

3.3. Customer with maximum wine purchase

Output:



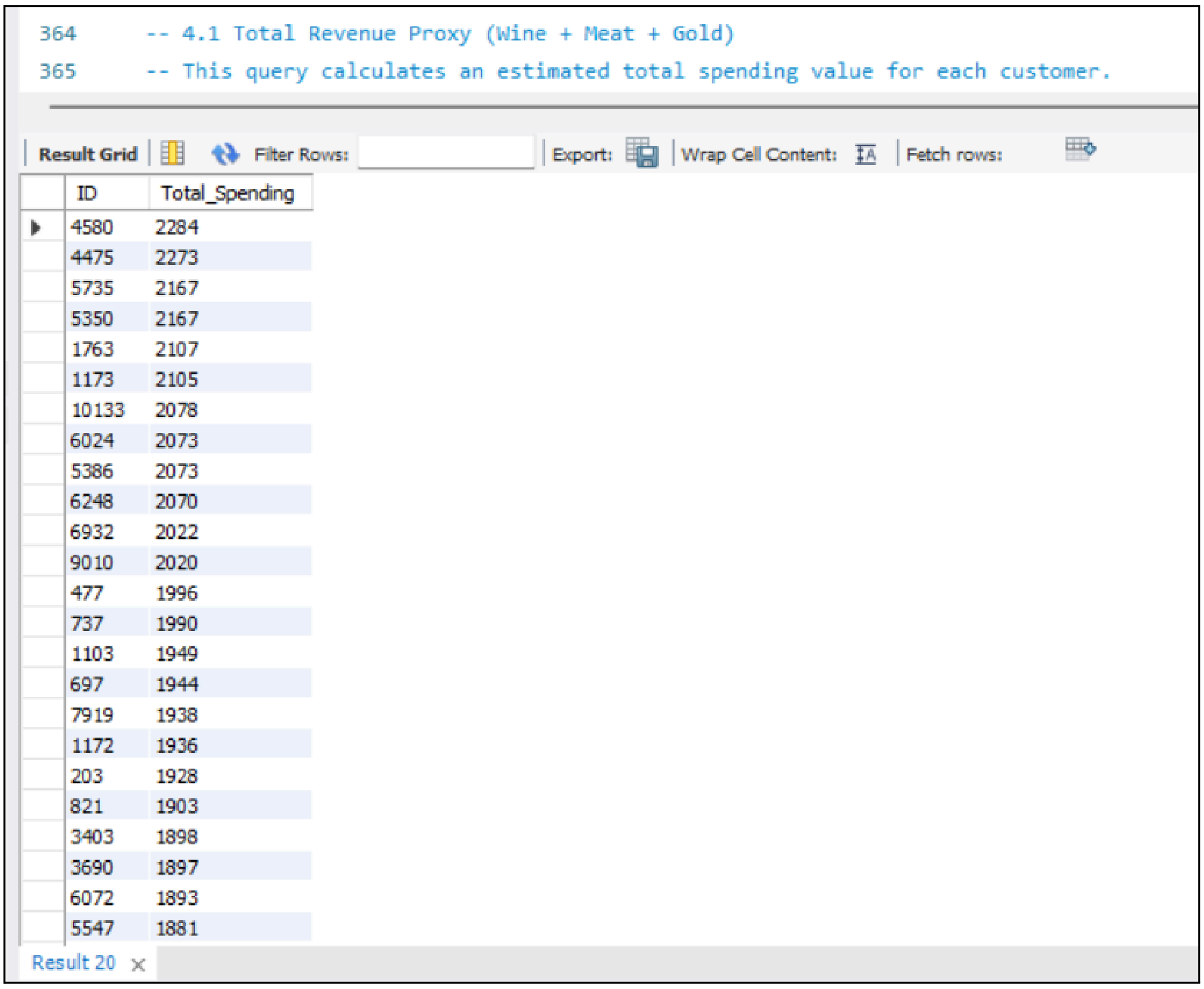
```
-- 3.3 Customer with maximum wine purchase
-- The subquery retrieves the highest value of wine spending from all customers.
-- The outer query returns the customer who has that highest spending.
SELECT ID, MntWines
FROM customer_data
WHERE MntWines = (
    SELECT MAX(MntWines)
    FROM customer_data
);
```

ID	MntWines
737	1493
NULL	NULL

d. Use aggregate functions (SUM, AVG)

4.1. Total Revenue Proxy (Wine + Meat + Gold)

Output:

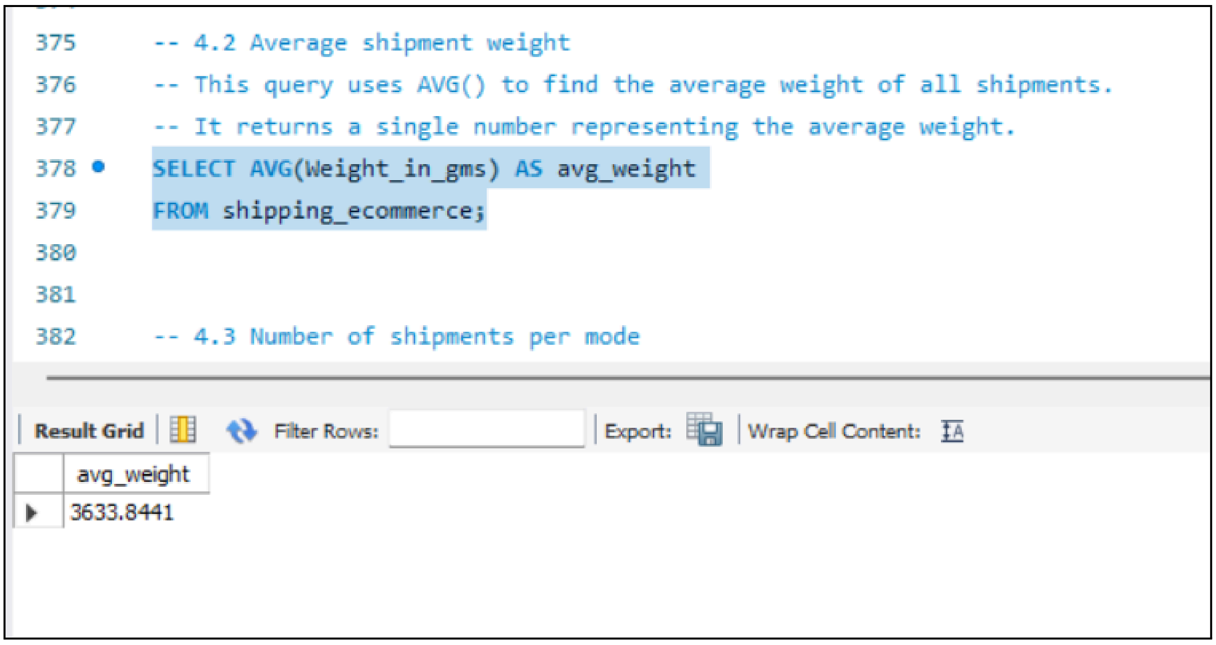


```
-- 4.1 Total Revenue Proxy (Wine + Meat + Gold)
-- This query calculates an estimated total spending value for each customer.
```

ID	Total_Spending
4580	2284
4475	2273
5735	2167
5350	2167
1763	2107
1173	2105
10133	2078
6024	2073
5386	2073
6248	2070
6932	2022
9010	2020
477	1996
737	1990
1103	1949
697	1944
7919	1938
1172	1936
203	1928
821	1903
3403	1898
3690	1897
6072	1893
5547	1881

4.2. Average shipment weight

Output:

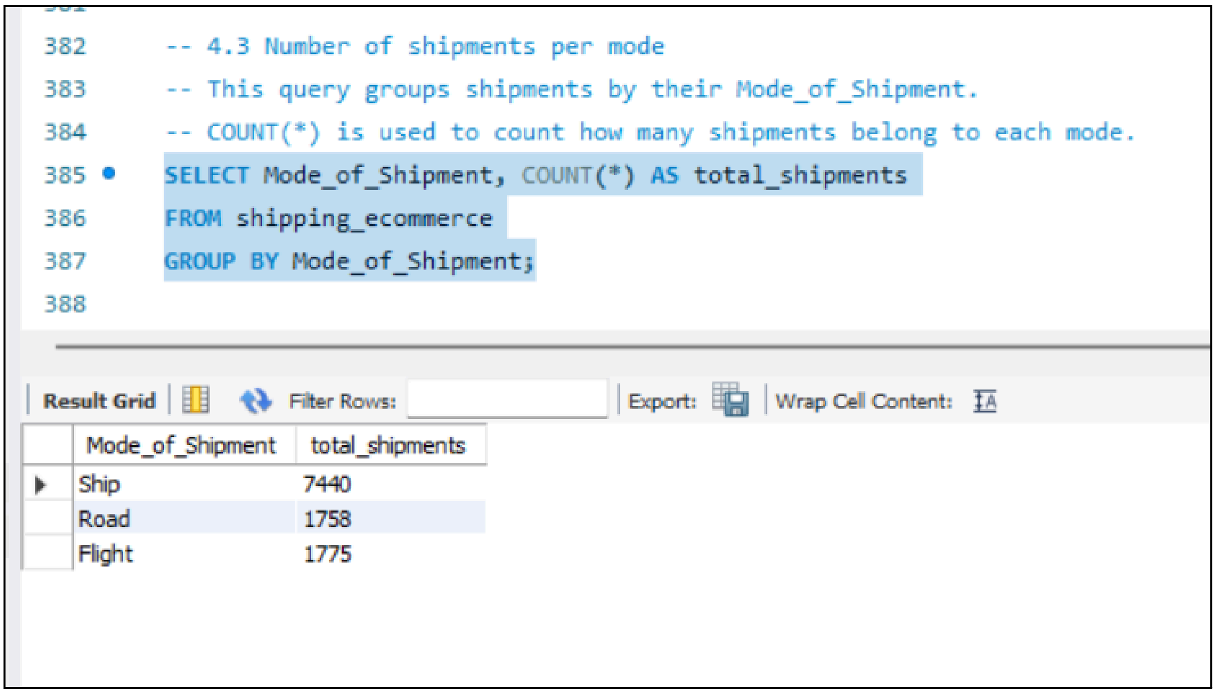

```sql
-- 4.2 Average shipment weight
-- This query uses AVG() to find the average weight of all shipments.
-- It returns a single number representing the average weight.
SELECT AVG(Weight_in_gms) AS avg_weight
FROM shipping_ecommerce;


-- 4.3 Number of shipments per mode
```

```
avg_weight
3633.8441
```

4.3. Number of shipments per mode

Output:

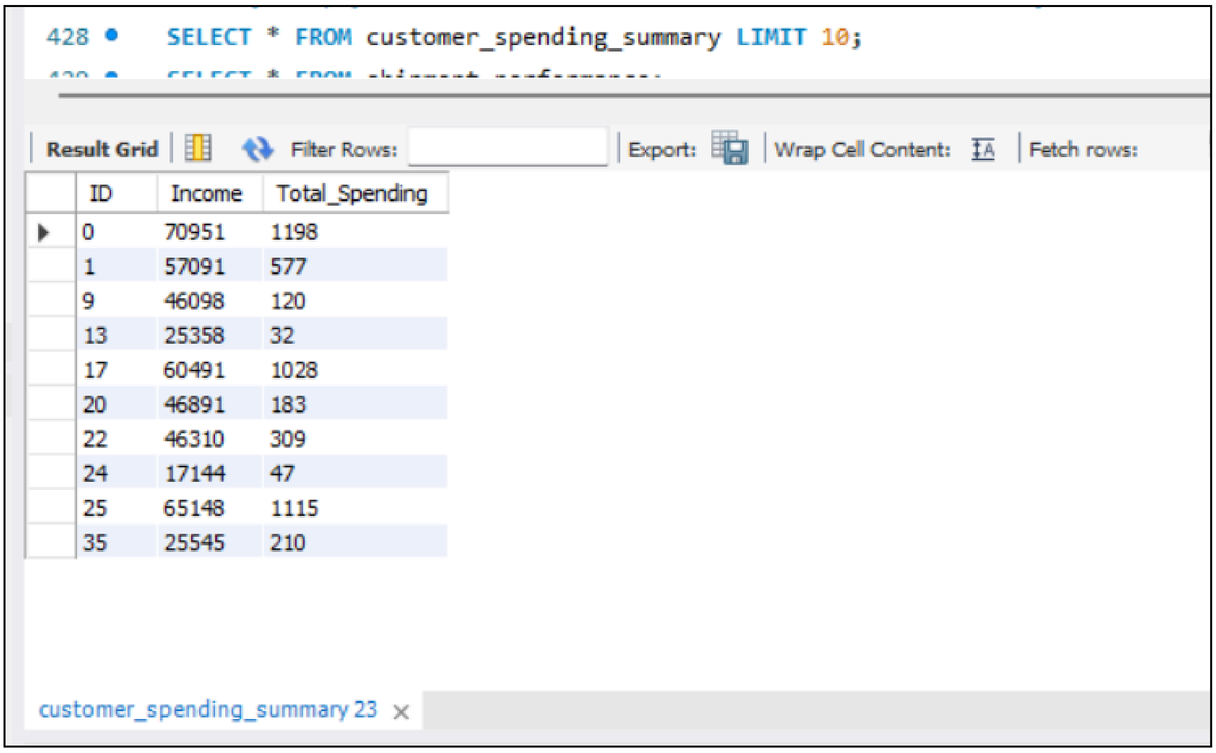

```sql
-- 4.3 Number of shipments per mode
-- This query groups shipments by their Mode_of_Shipment.
-- COUNT(*) is used to count how many shipments belong to each mode.
SELECT Mode_of_Shipment, COUNT(*) AS total_shipments
FROM shipping_ecommerce
GROUP BY Mode_of_Shipment;
```

Mode_of_Shipment	total_shipments
Ship	7440
Road	1758
Flight	1775

e.Create views for analysis

5.1. Customer Spending Summary View

Output:



```
SELECT * FROM customer_spending_summary LIMIT 10;
```

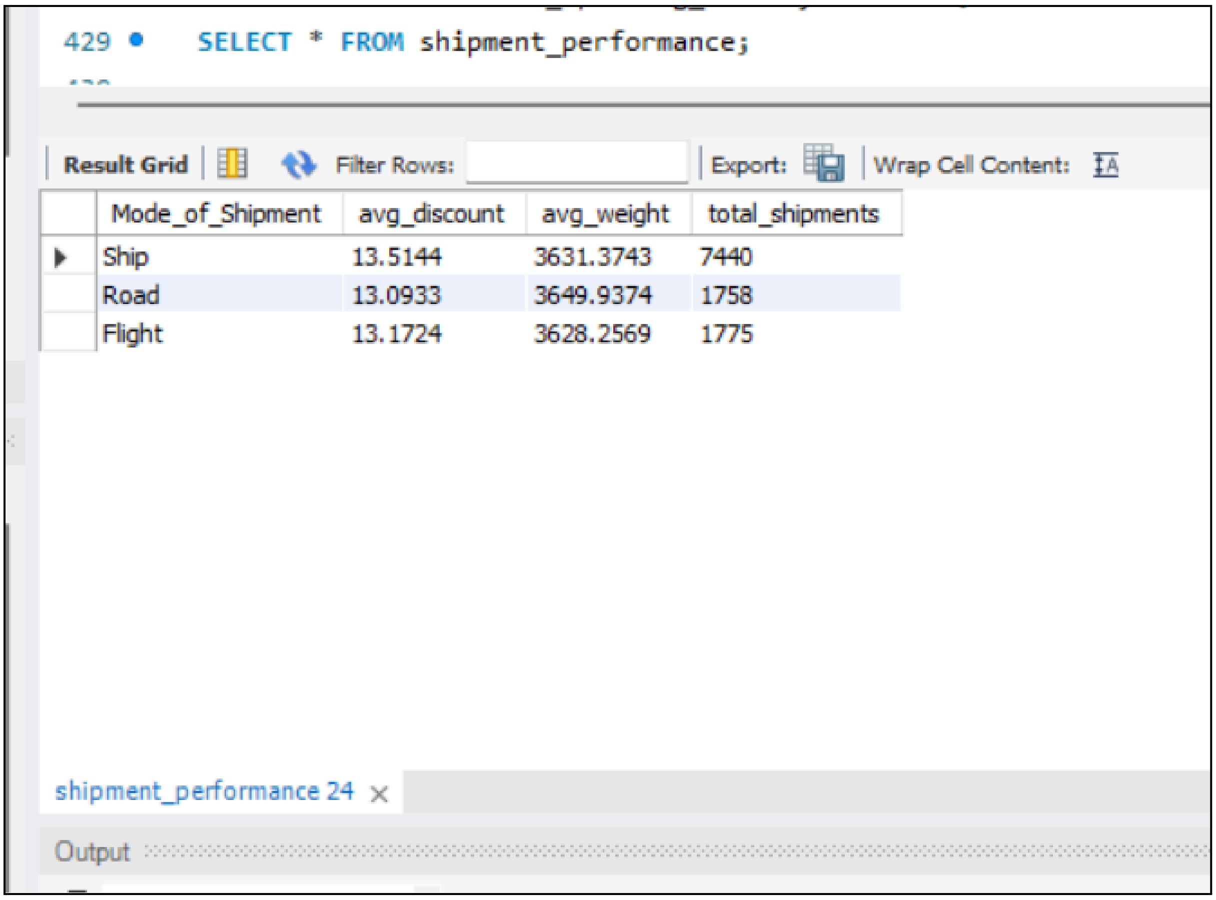
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

ID	Income	Total_Spending
0	70951	1198
1	57091	577
9	46098	120
13	25358	32
17	60491	1028
20	46891	183
22	46310	309
24	17144	47
25	65148	1115
35	25545	210

customer_spending_summary 23

5.2. Shipment Performance View

Output:



```
SELECT * FROM shipment_performance;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Mode_of_Shipment	avg_discount	avg_weight	total_shipments
Ship	13.5144	3631.3743	7440
Road	13.0933	3649.9374	1758
Flight	13.1724	3628.2569	1775

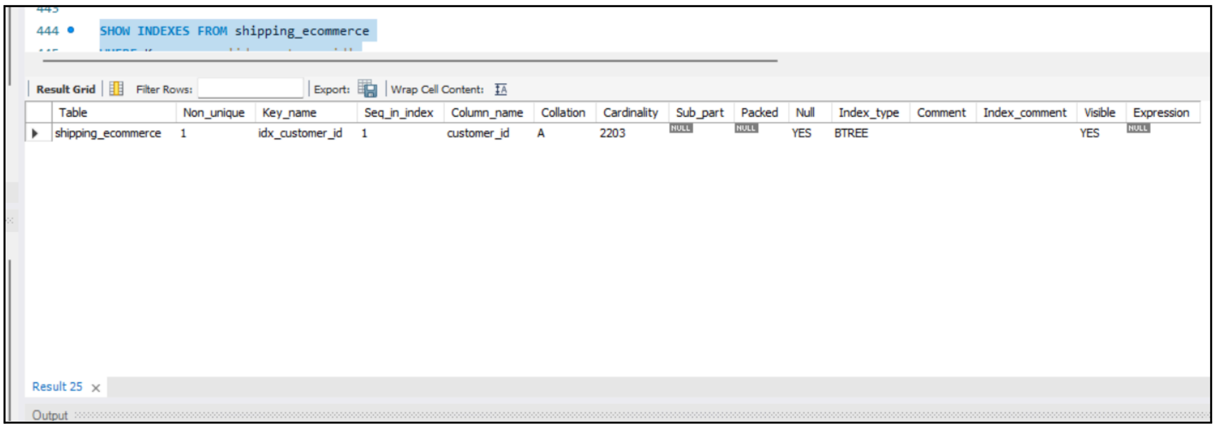
shipment_performance 24

Output

f. Optimize queries with indexes

6.1. Index for joining the two tables

Output:



```
SHOW INDEXES FROM shipping_ecommerce
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

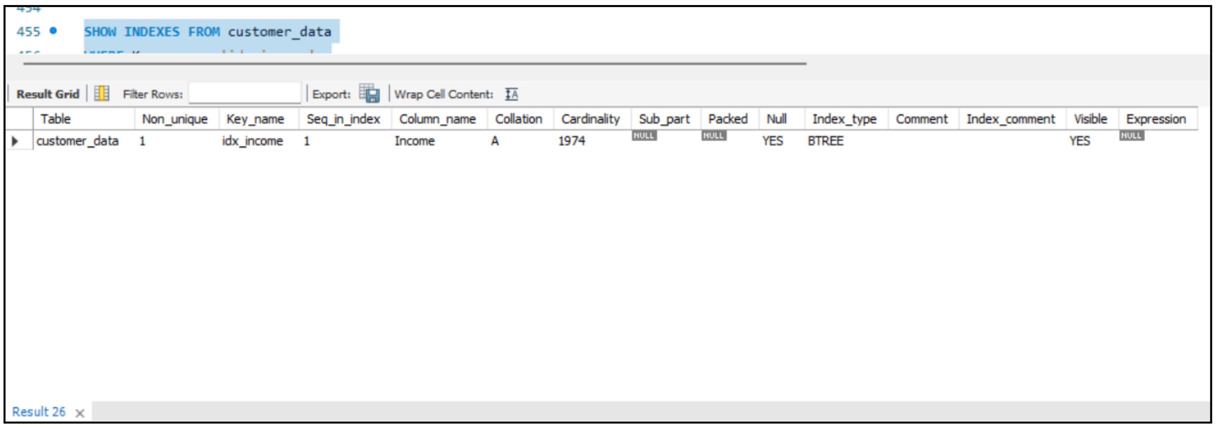
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
shipping_ecommerce	1	idx_customer_id	1	customer_id	A	2203	NULL	NULL	YES	BTREE			YES	NULL

Result 25 ×

Output

6.2. Index for commonly filtered columns

Output:

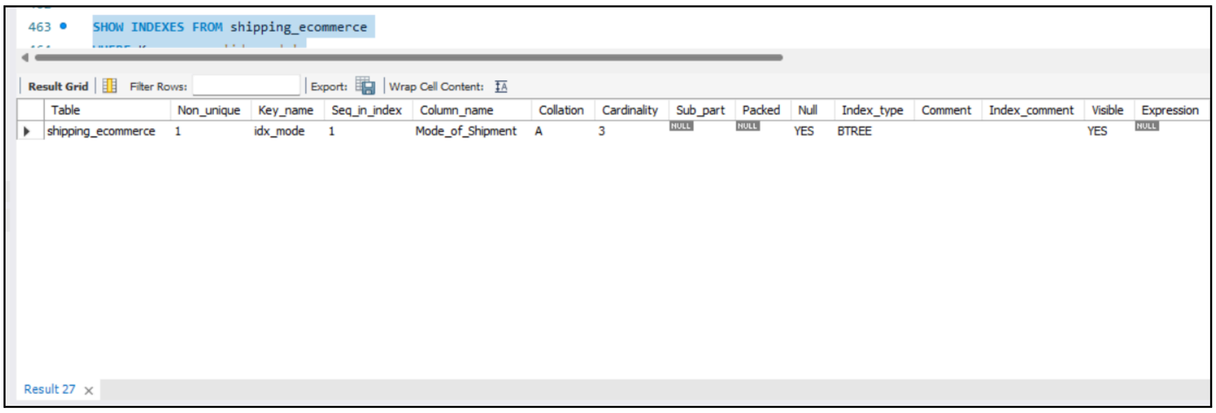


```
SHOW INDEXES FROM customer_data
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
customer_data	1	idx_income	1	Income	A	1974	NULL	NULL	YES	BTREE			YES	NULL

Result 26 ×

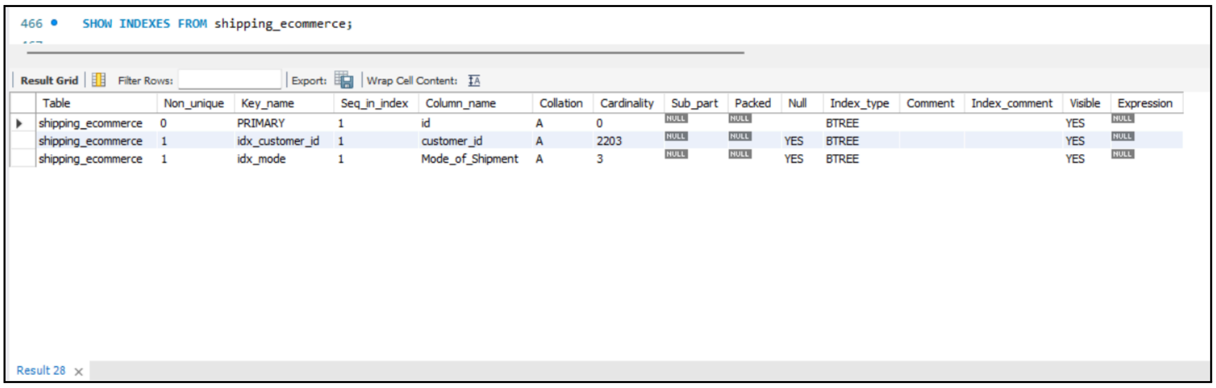


```
SHOW INDEXES FROM shipping_ecommerce
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

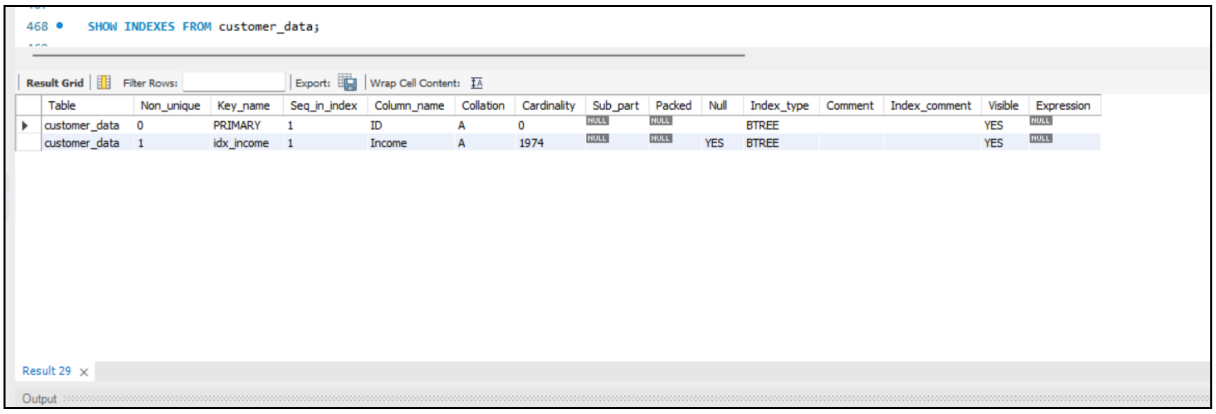
Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
shipping_ecommerce	1	idx_mode	1	Mode_of_Shipment	A	3	NULL	NULL	YES	BTREE			YES	NULL

Result 27 ×



```sql
SHOW INDEXES FROM shipping_ecommerce;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
shipping_ecommerce	0	PRIMARY	1	id	A	0	NULL	NULL		BTREE			YES	NULL
shipping_ecommerce	1	idx_customer_id	1	customer_id	A	2203	NULL	NULL	YES	BTREE			YES	NULL
shipping_ecommerce	1	idx_mode	1	Mode_of_Shipment	A	3	NULL	NULL	YES	BTREE			YES	NULL



```sql
SHOW INDEXES FROM customer_data;
```

Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type	Comment	Index_comment	Visible	Expression
customer_data	0	PRIMARY	1	ID	A	0	NULL	NULL		BTREE			YES	NULL
customer_data	1	idx_income	1	Income	A	1974	NULL	NULL	YES	BTREE			YES	NULL