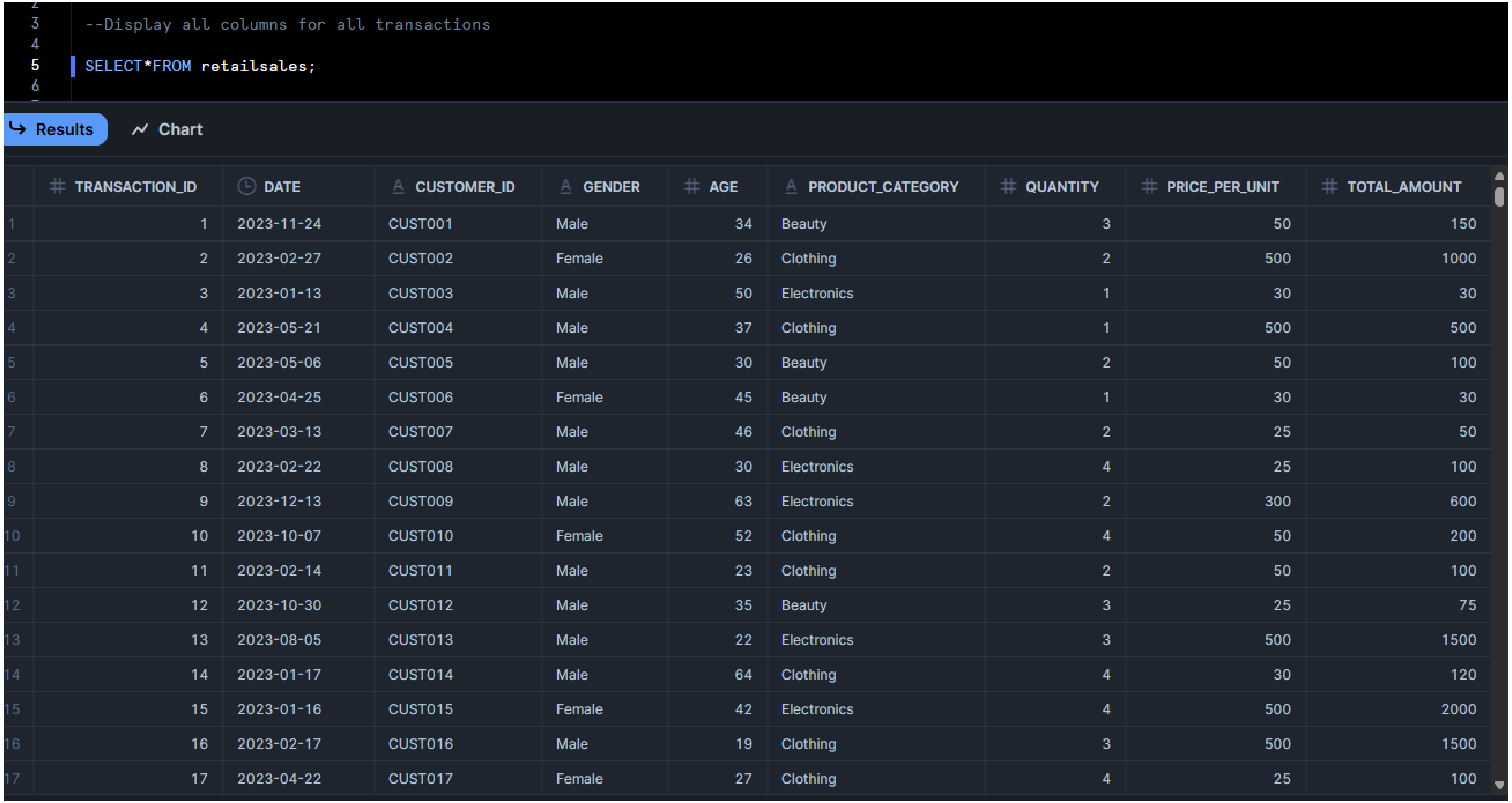
Practical Exercise 1

SQL on Snowflake

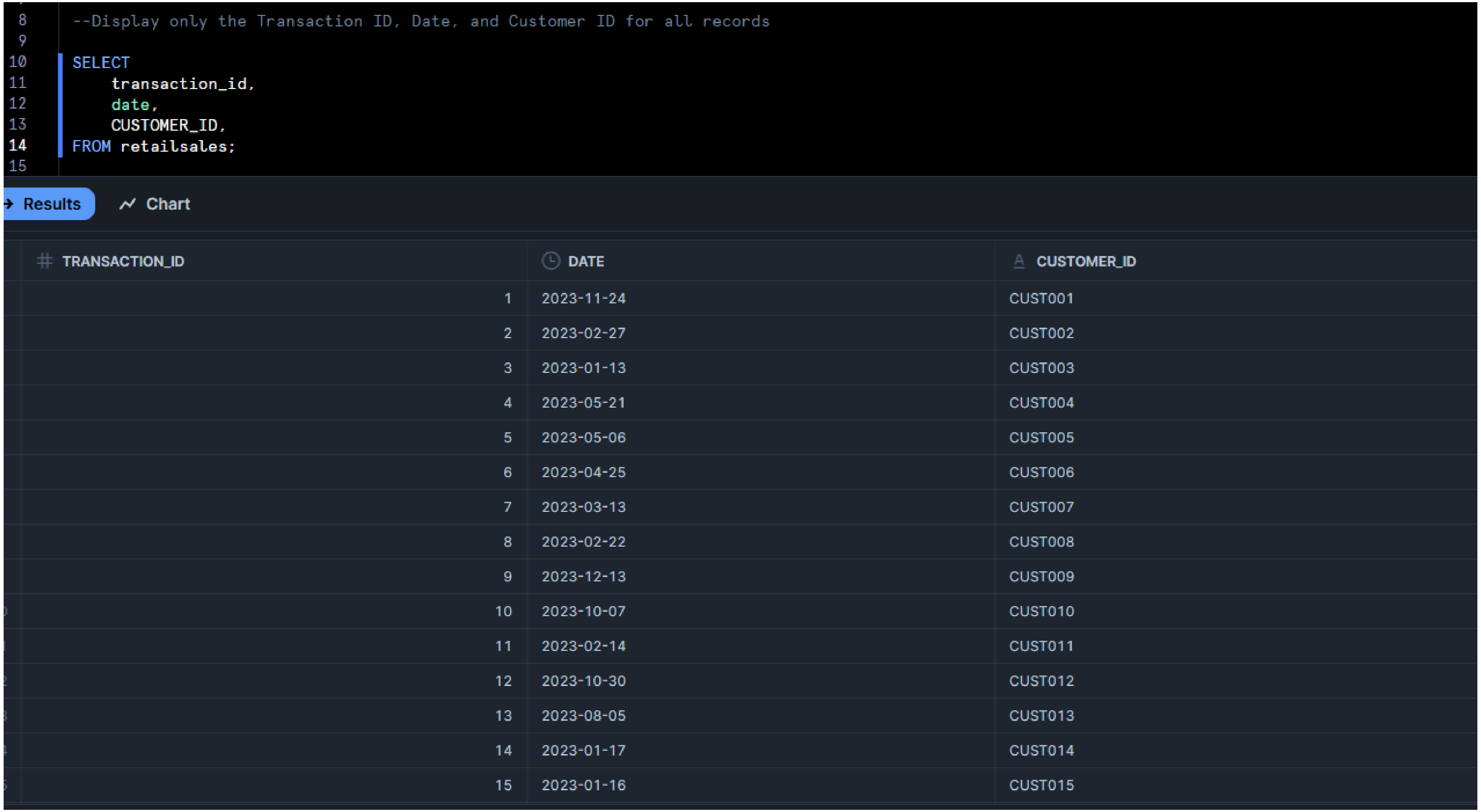
1. Display all columns for all transactions



```
--Display all columns for all transactions

SELECT*FROM retailsales;
```

	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
2	2	2023-02-27	CUST002	Female	26	Clothing	2	500	1000
3	3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
4	4	2023-05-21	CUST004	Male	37	Clothing	1	500	500
5	5	2023-05-06	CUST005	Male	30	Beauty	2	50	100
6	6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
7	7	2023-03-13	CUST007	Male	46	Clothing	2	25	50
8	8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
9	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
10	10	2023-10-07	CUST010	Female	52	Clothing	4	50	200
11	11	2023-02-14	CUST011	Male	23	Clothing	2	50	100
12	12	2023-10-30	CUST012	Male	35	Beauty	3	25	75
13	13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
14	14	2023-01-17	CUST014	Male	64	Clothing	4	30	120
15	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
16	16	2023-02-17	CUST016	Male	19	Clothing	3	500	1500
17	17	2023-04-22	CUST017	Female	27	Clothing	4	25	100

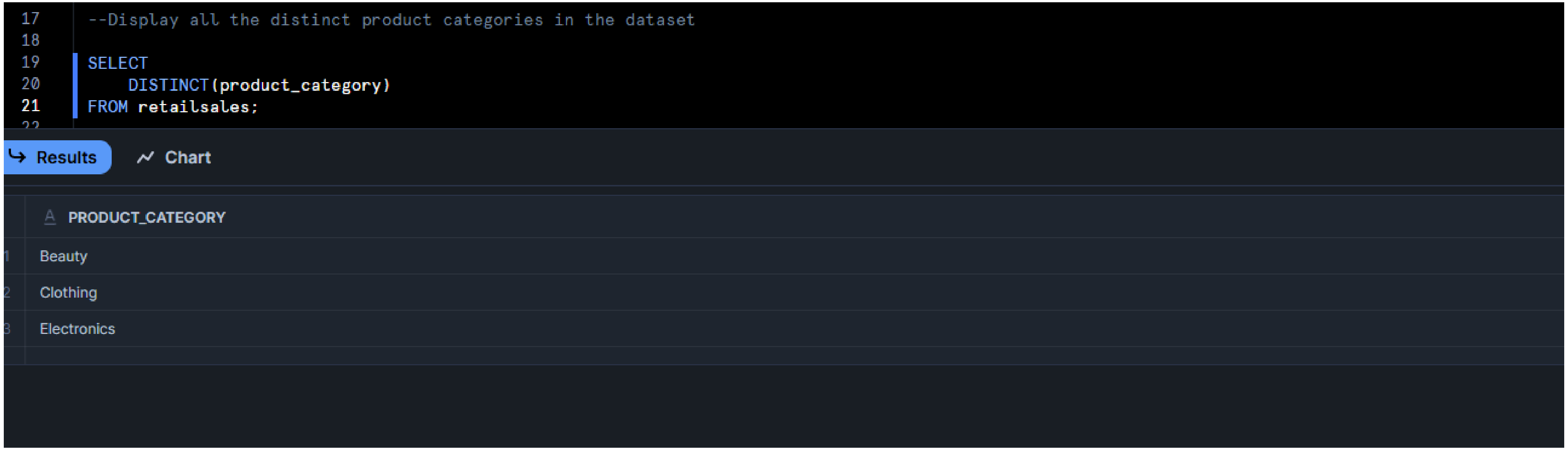
2. Display only the Transaction ID, Date, and Customer ID for all records



```
--Display only the Transaction ID, Date, and Customer ID for all records

SELECT
    transaction_id,
    date,
    CUSTOMER_ID,
FROM retailsales;
```

TRANSACTION_ID	DATE	CUSTOMER_ID
1	2023-11-24	CUST001
2	2023-02-27	CUST002
3	2023-01-13	CUST003
4	2023-05-21	CUST004
5	2023-05-06	CUST005
6	2023-04-25	CUST006
7	2023-03-13	CUST007
8	2023-02-22	CUST008
9	2023-12-13	CUST009
10	2023-10-07	CUST010
11	2023-02-14	CUST011
12	2023-10-30	CUST012
13	2023-08-05	CUST013
14	2023-01-17	CUST014
15	2023-01-16	CUST015

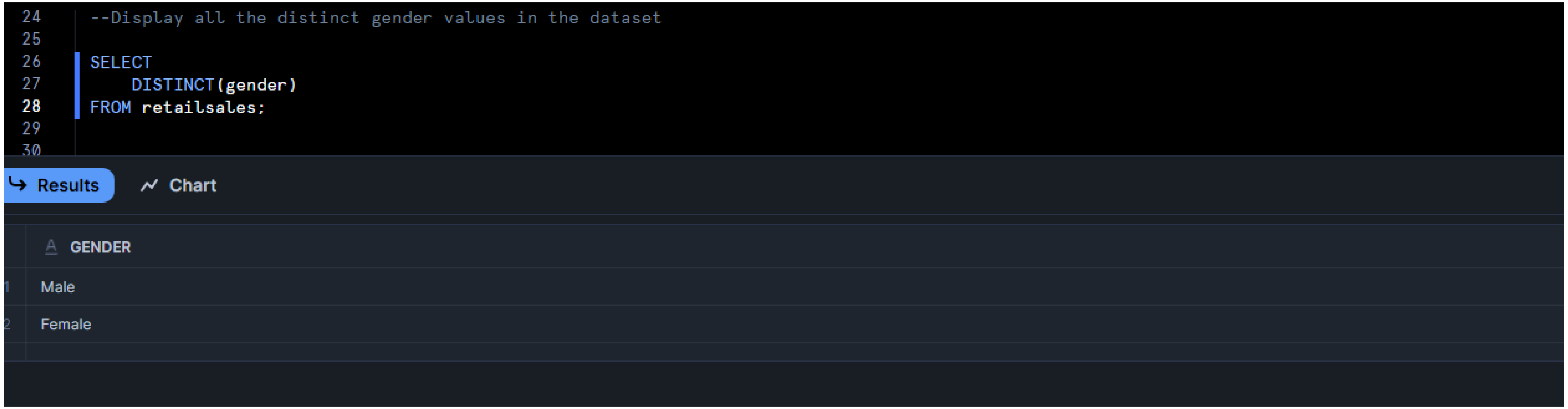
3. Display all the distinct product categories in the dataset.



```sql
--Display all the distinct product categories in the dataset

SELECT
    DISTINCT(product_category)
FROM retailsales;
```

	PRODUCT_CATEGORY
1	Beauty
2	Clothing
3	Electronics

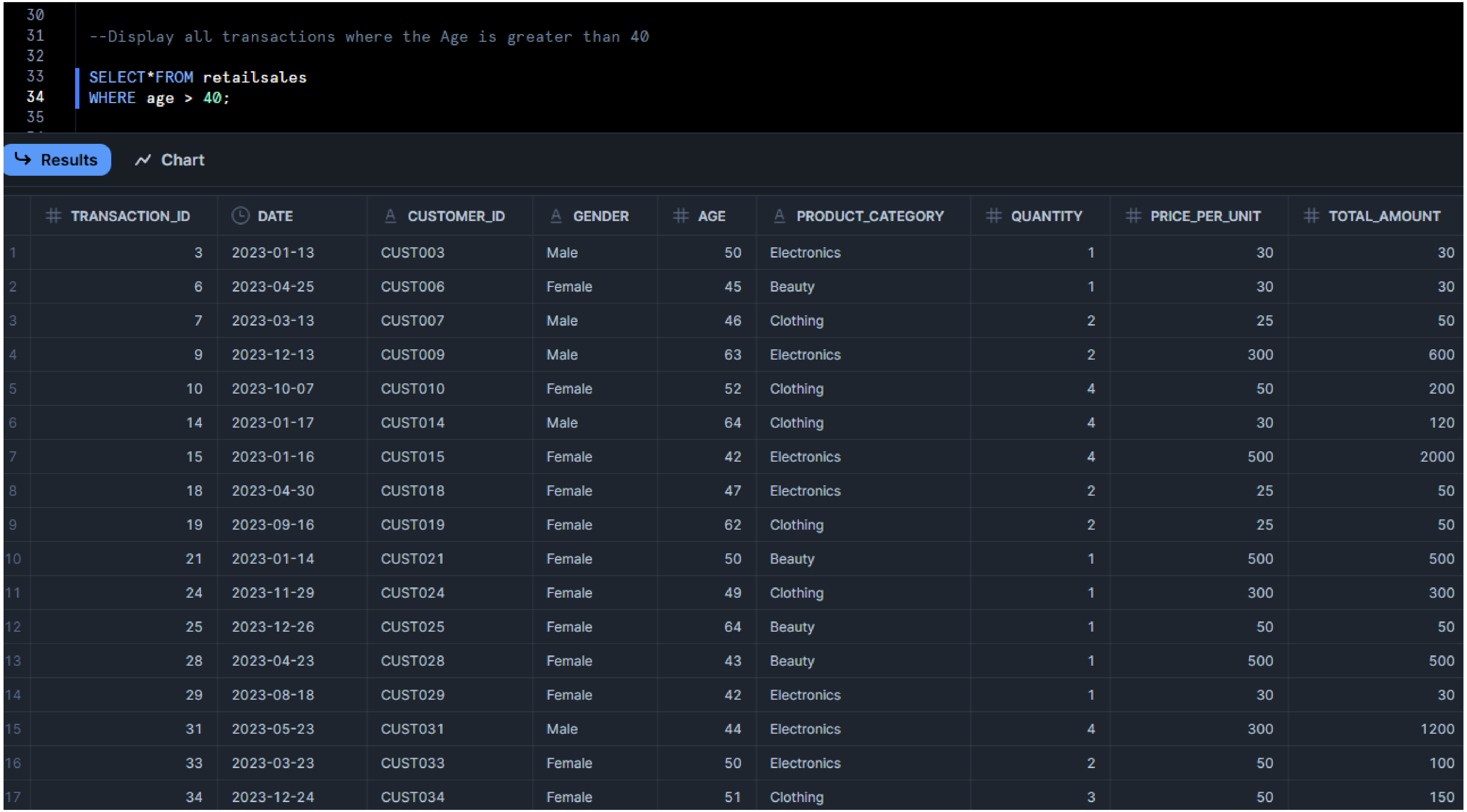
4. Display all the distinct gender values in the dataset



```
--Display all the distinct gender values in the dataset

SELECT
    DISTINCT(gender)
FROM retailsales;
```

	GENDER
1	Male
2	Female

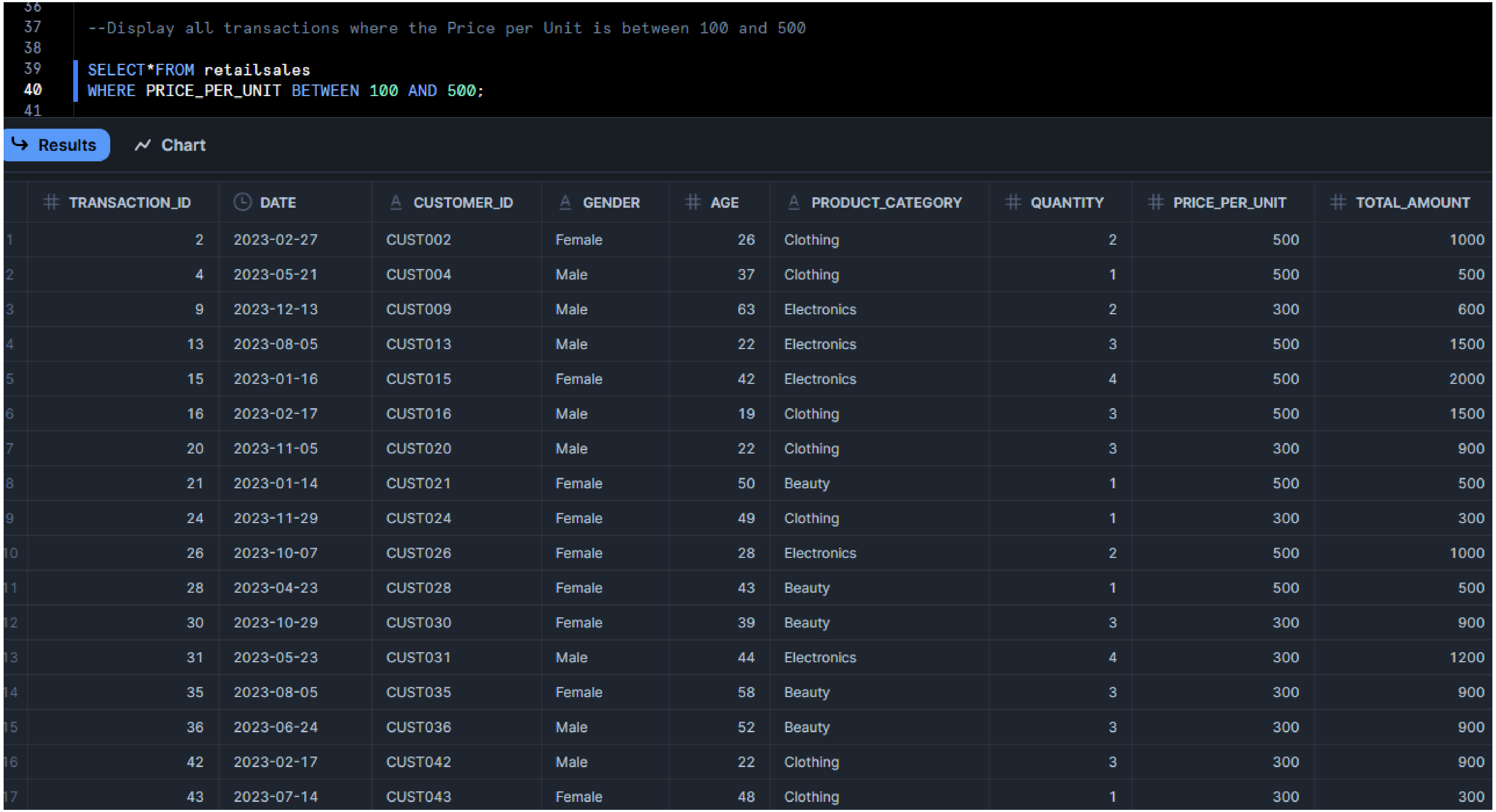
5. Display all transactions where the Age is greater than 40



```
--Display all transactions where the Age is greater than 40

SELECT*FROM retailsales
WHERE age > 40;
```

	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
2	6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
3	7	2023-03-13	CUST007	Male	46	Clothing	2	25	50
4	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
5	10	2023-10-07	CUST010	Female	52	Clothing	4	50	200
6	14	2023-01-17	CUST014	Male	64	Clothing	4	30	120
7	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
8	18	2023-04-30	CUST018	Female	47	Electronics	2	25	50
9	19	2023-09-16	CUST019	Female	62	Clothing	2	25	50
10	21	2023-01-14	CUST021	Female	50	Beauty	1	500	500
11	24	2023-11-29	CUST024	Female	49	Clothing	1	300	300
12	25	2023-12-26	CUST025	Female	64	Beauty	1	50	50
13	28	2023-04-23	CUST028	Female	43	Beauty	1	500	500
14	29	2023-08-18	CUST029	Female	42	Electronics	1	30	30
15	31	2023-05-23	CUST031	Male	44	Electronics	4	300	1200
16	33	2023-03-23	CUST033	Female	50	Electronics	2	50	100
17	34	2023-12-24	CUST034	Female	51	Clothing	3	50	150

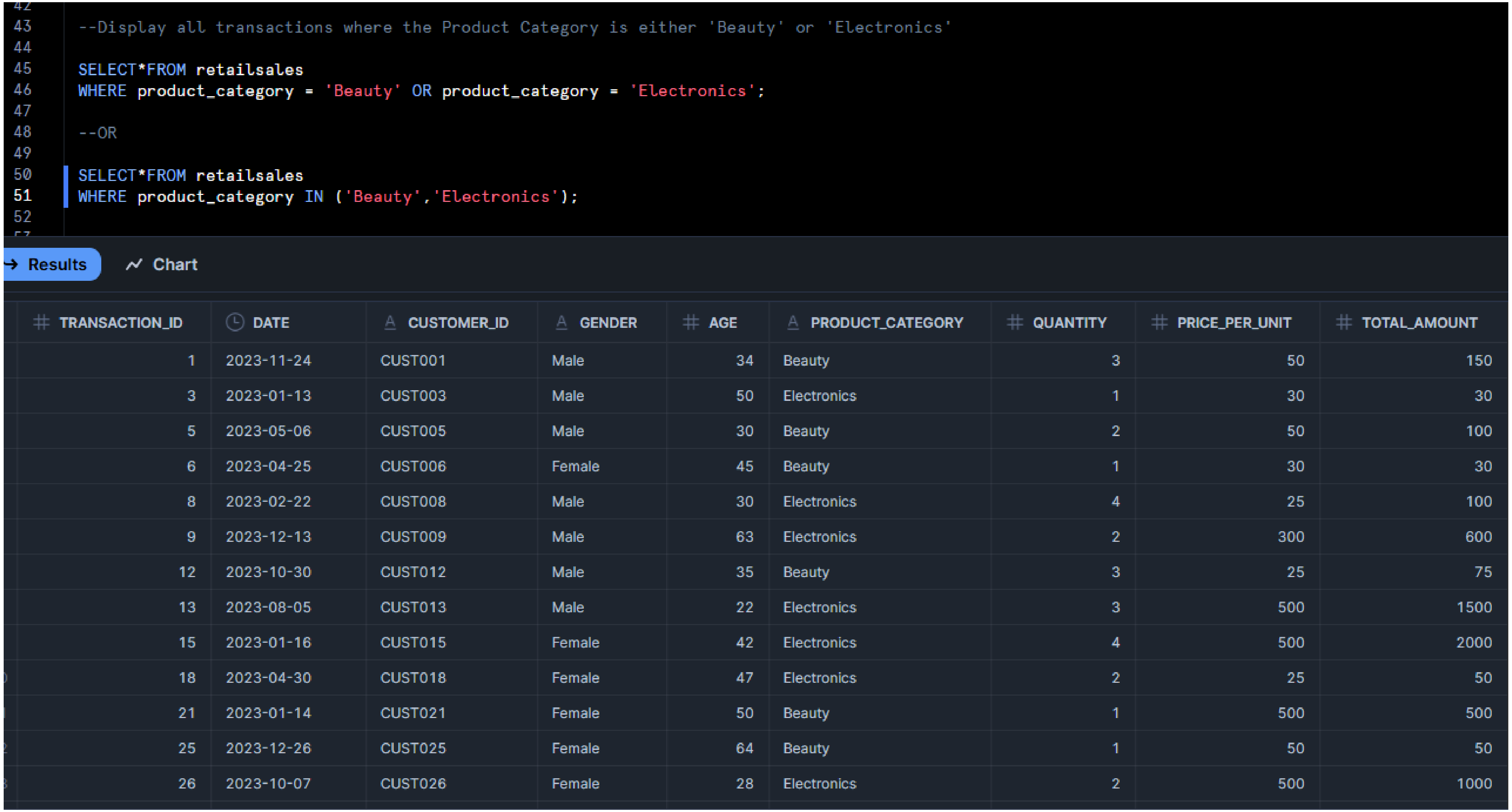
6. Display all transactions where the Price per Unit is between 100 and 500



```sql
--Display all transactions where the Price per Unit is between 100 and 500

SELECT*FROM retailsales
WHERE PRICE_PER_UNIT BETWEEN 100 AND 500;
```

	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	2	2023-02-27	CUST002	Female	26	Clothing	2	500	1000
2	4	2023-05-21	CUST004	Male	37	Clothing	1	500	500
3	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
4	13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
5	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
6	16	2023-02-17	CUST016	Male	19	Clothing	3	500	1500
7	20	2023-11-05	CUST020	Male	22	Clothing	3	300	900
8	21	2023-01-14	CUST021	Female	50	Beauty	1	500	500
9	24	2023-11-29	CUST024	Female	49	Clothing	1	300	300
10	26	2023-10-07	CUST026	Female	28	Electronics	2	500	1000
11	28	2023-04-23	CUST028	Female	43	Beauty	1	500	500
12	30	2023-10-29	CUST030	Female	39	Beauty	3	300	900
13	31	2023-05-23	CUST031	Male	44	Electronics	4	300	1200
14	35	2023-08-05	CUST035	Female	58	Beauty	3	300	900
15	36	2023-06-24	CUST036	Male	52	Beauty	3	300	900
16	42	2023-02-17	CUST042	Male	22	Clothing	3	300	900
17	43	2023-07-14	CUST043	Female	48	Clothing	1	300	300

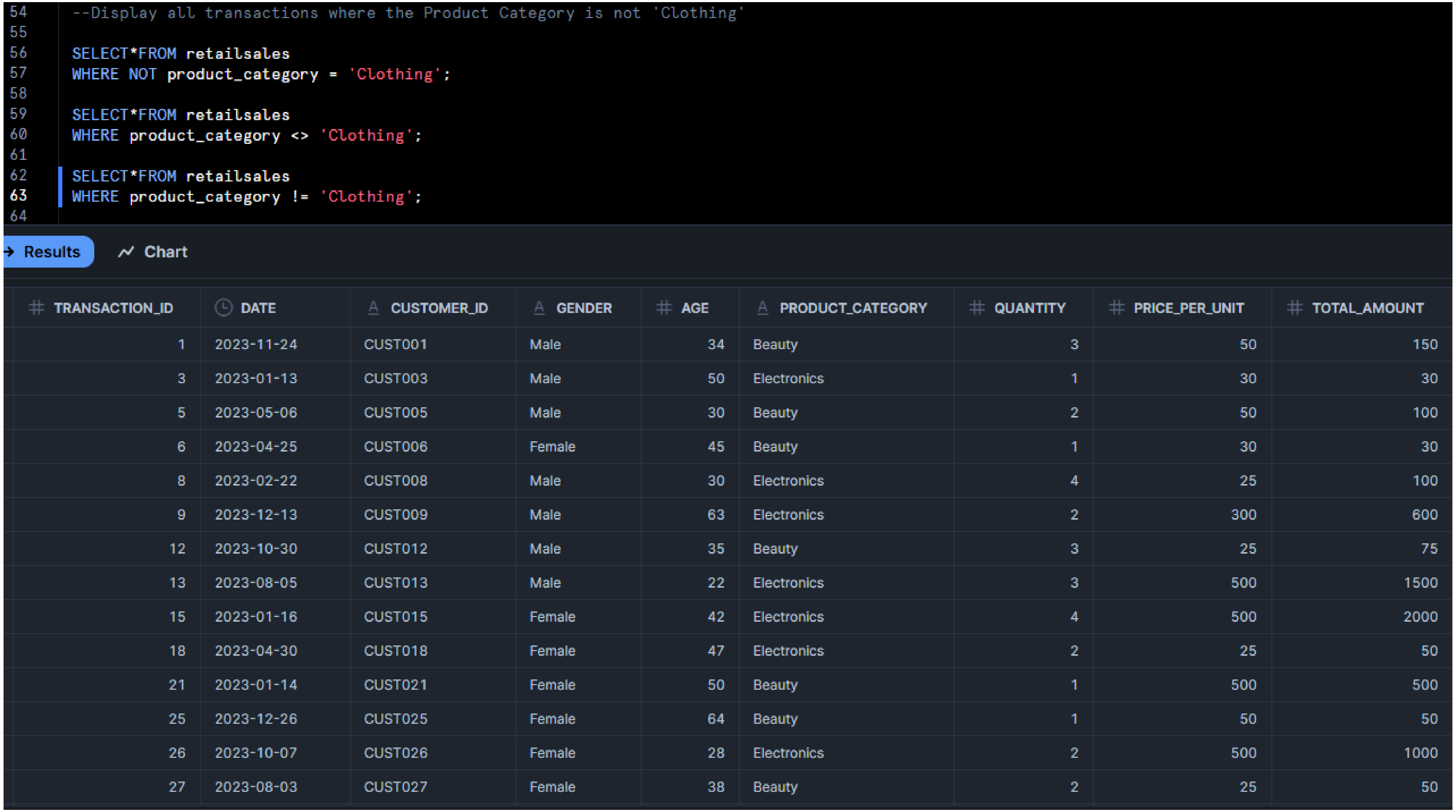
7. Display all transactions where the Product Category is either 'Beauty' or 'Electronics'



```
--Display all transactions where the Product Category is either 'Beauty' or 'Electronics'

SELECT*FROM retailsales
WHERE product_category = 'Beauty' OR product_category = 'Electronics';

--OR

SELECT*FROM retailsales
WHERE product_category IN ('Beauty','Electronics');
```

TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
5	2023-05-06	CUST005	Male	30	Beauty	2	50	100
6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
12	2023-10-30	CUST012	Male	35	Beauty	3	25	75
13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
18	2023-04-30	CUST018	Female	47	Electronics	2	25	50
21	2023-01-14	CUST021	Female	50	Beauty	1	500	500
25	2023-12-26	CUST025	Female	64	Beauty	1	50	50
26	2023-10-07	CUST026	Female	28	Electronics	2	500	1000

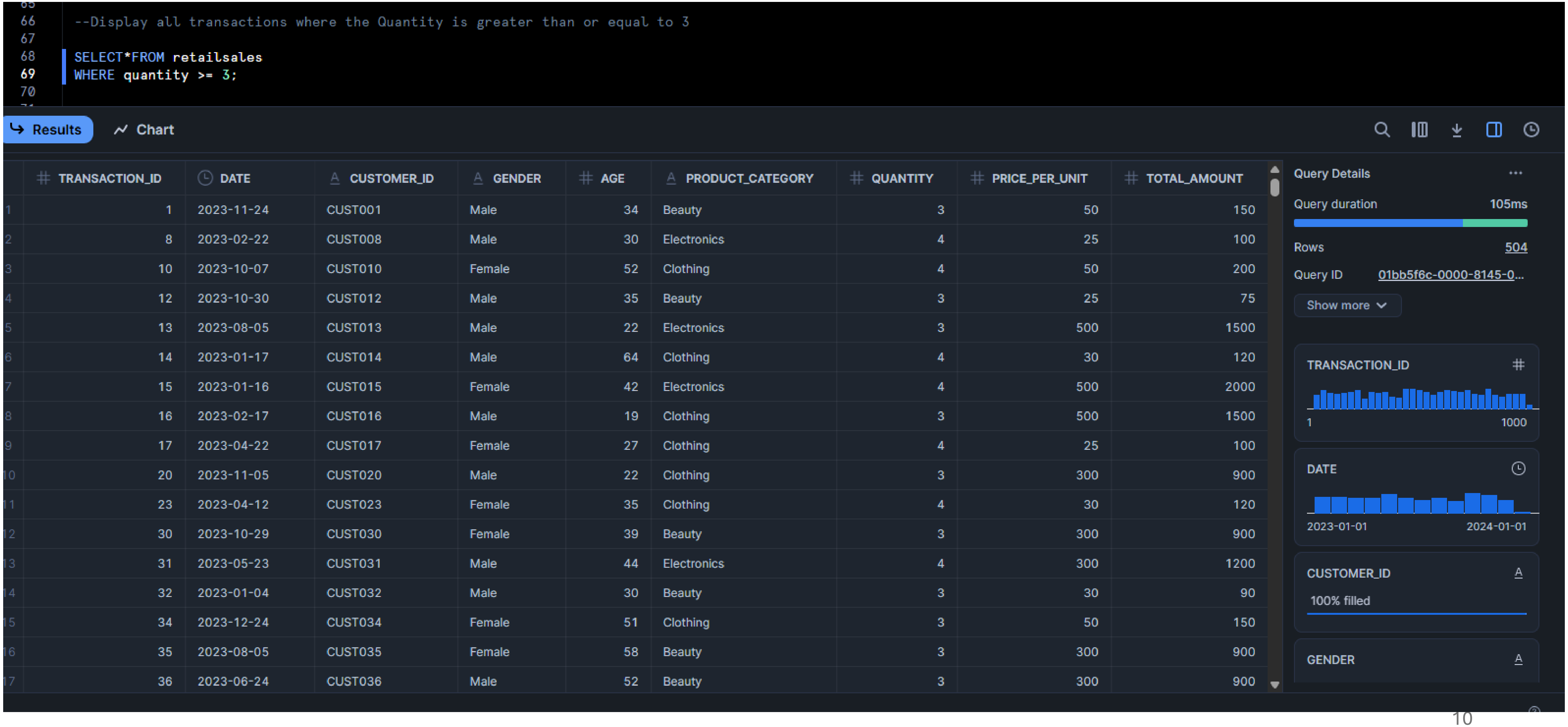
8. Display all transactions where the Product Category is not 'Clothing'.



```
--Display all transactions where the Product Category is not 'Clothing'

SELECT*FROM retailsales
WHERE NOT product_category = 'Clothing';

SELECT*FROM retailsales
WHERE product_category <> 'Clothing';

SELECT*FROM retailsales
WHERE product_category != 'Clothing';
```

TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
5	2023-05-06	CUST005	Male	30	Beauty	2	50	100
6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
12	2023-10-30	CUST012	Male	35	Beauty	3	25	75
13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
18	2023-04-30	CUST018	Female	47	Electronics	2	25	50
21	2023-01-14	CUST021	Female	50	Beauty	1	500	500
25	2023-12-26	CUST025	Female	64	Beauty	1	50	50
26	2023-10-07	CUST026	Female	28	Electronics	2	500	1000
27	2023-08-03	CUST027	Female	38	Beauty	2	25	50

9. Display all transactions where the Quantity is greater than or equal to 3



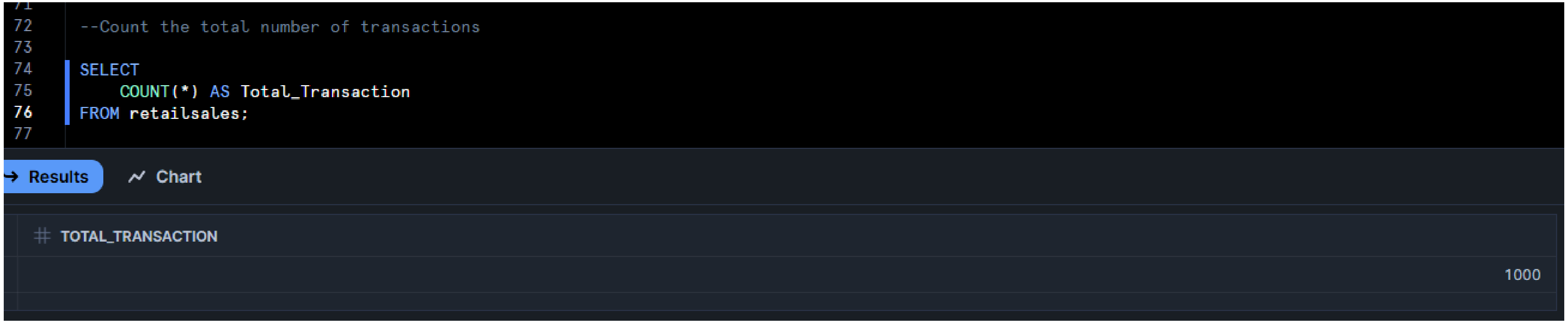
```
--Display all transactions where the Quantity is greater than or equal to 3

SELECT*FROM retailsales
WHERE quantity >= 3;
```

TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
10	2023-10-07	CUST010	Female	52	Clothing	4	50	200
12	2023-10-30	CUST012	Male	35	Beauty	3	25	75
13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
14	2023-01-17	CUST014	Male	64	Clothing	4	30	120
15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
16	2023-02-17	CUST016	Male	19	Clothing	3	500	1500
17	2023-04-22	CUST017	Female	27	Clothing	4	25	100
20	2023-11-05	CUST020	Male	22	Clothing	3	300	900
23	2023-04-12	CUST023	Female	35	Clothing	4	30	120
30	2023-10-29	CUST030	Female	39	Beauty	3	300	900
31	2023-05-23	CUST031	Male	44	Electronics	4	300	1200
32	2023-01-04	CUST032	Male	30	Beauty	3	30	90
34	2023-12-24	CUST034	Female	51	Clothing	3	50	150
35	2023-08-05	CUST035	Female	58	Beauty	3	300	900
36	2023-06-24	CUST036	Male	52	Beauty	3	300	900

Query Details

Query duration	105ms
Rows	504
Query ID	01bb5f6c-0000-8145-0...

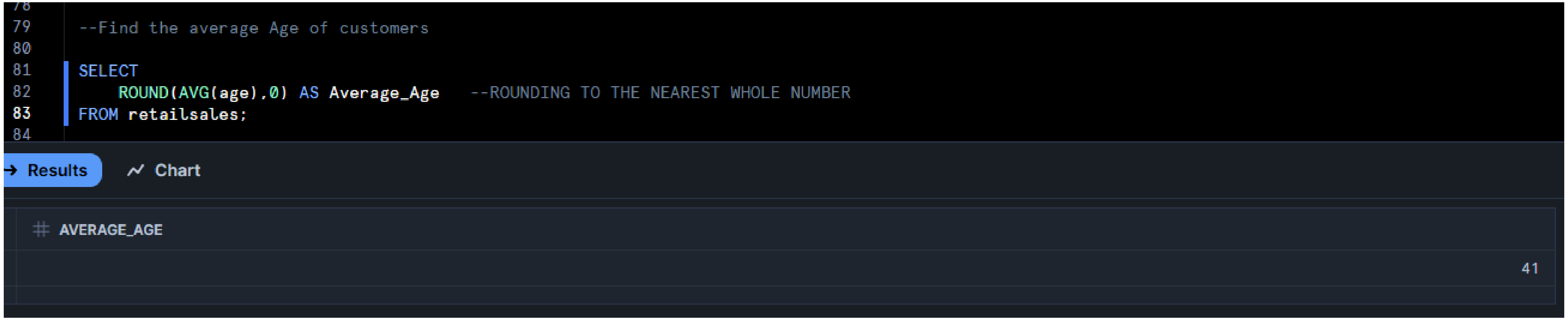
10. Count the total number of transactions



```
--Count the total number of transactions

SELECT
    COUNT(*) AS Total_Transaction
FROM retailsales;
```

```
TOTAL_TRANSACTION
                                                   1000
```

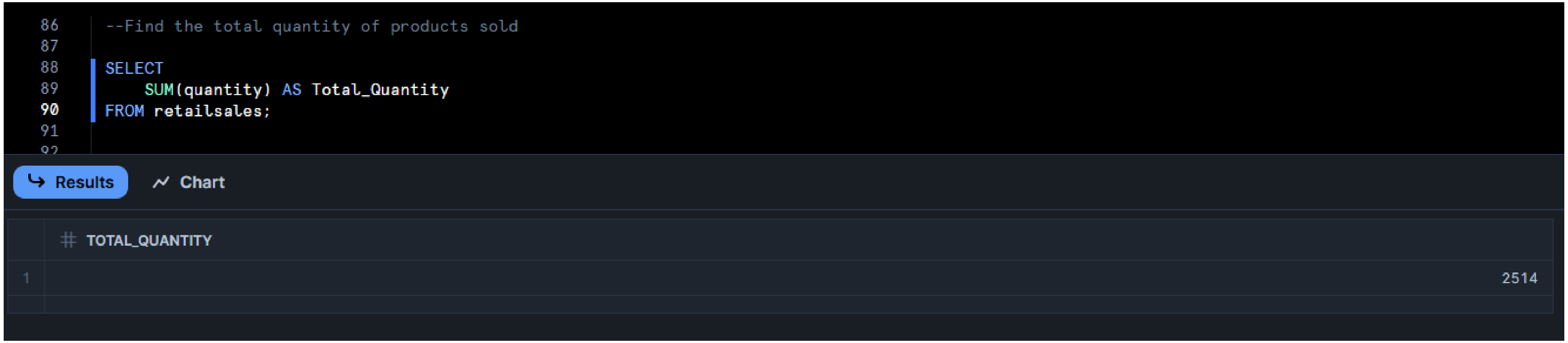
11. Find the average Age of customers



```sql
--Find the average Age of customers

SELECT
    ROUND(AVG(age),0) AS Average_Age   --ROUNDING TO THE NEAREST WHOLE NUMBER
FROM retailsales;

```

AVERAGE_AGE
41

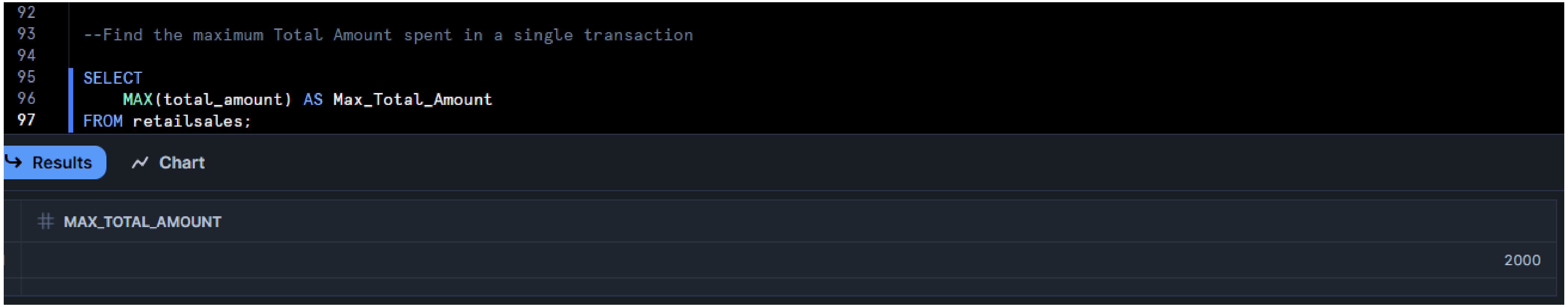
12. Find the total quantity of products sold



```
--Find the total quantity of products sold

SELECT
    SUM(quantity) AS Total_Quantity
FROM retailsales;
```

```
   TOTAL_QUANTITY
1            2514
```

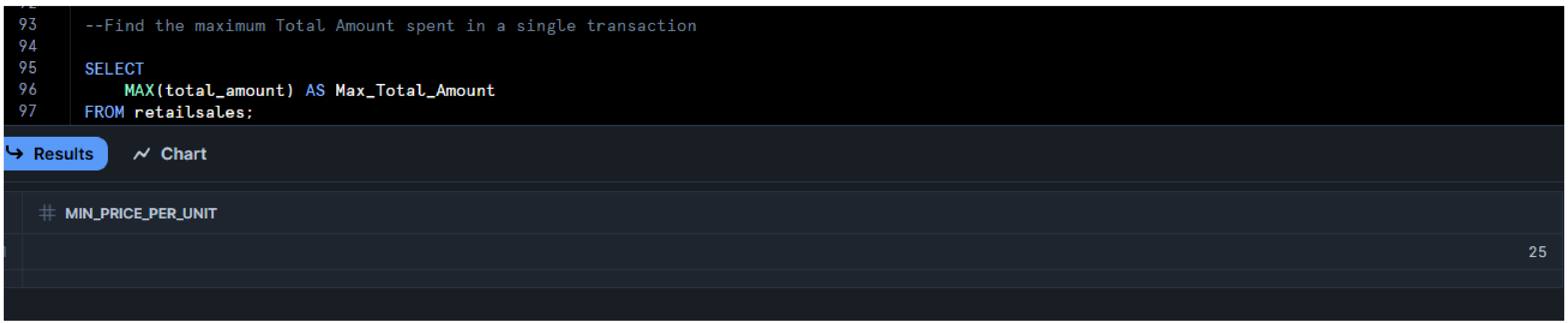
13. Find the maximum Total Amount spent in a single transaction



```
--Find the maximum Total Amount spent in a single transaction

SELECT
    MAX(total_amount) AS Max_Total_Amount
FROM retailsales;
```

MAX_TOTAL_AMOUNT
2000

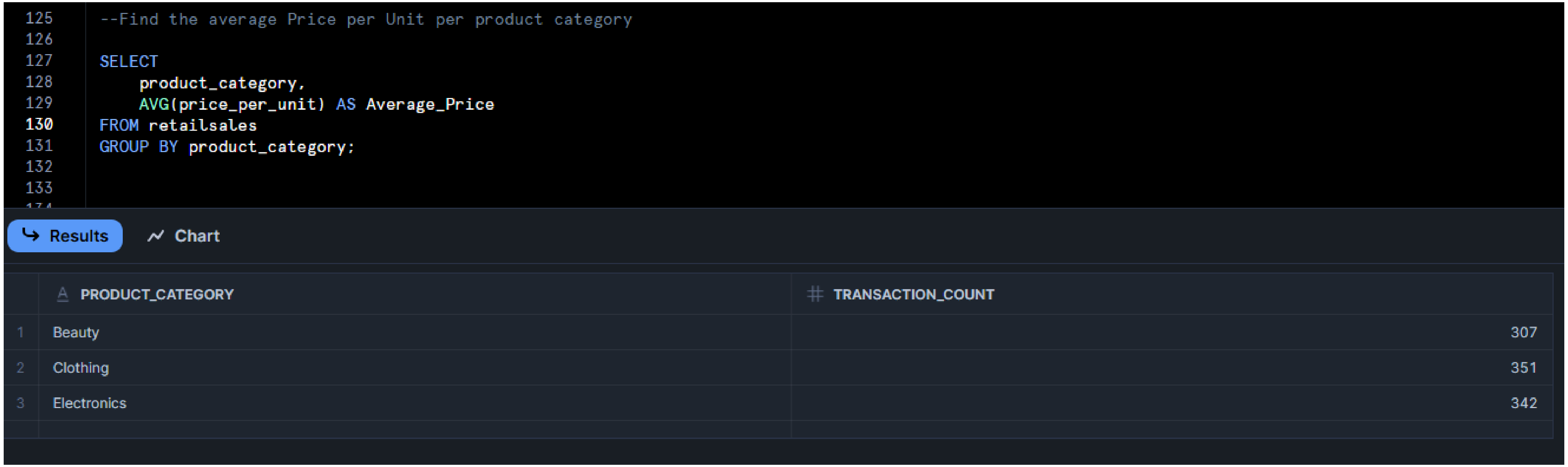
14. Find the minimum Price per Unit in the dataset



```sql
--Find the maximum Total Amount spent in a single transaction

SELECT
    MAX(total_amount) AS Max_Total_Amount
FROM retailsales;
```

MIN_PRICE_PER_UNIT
25

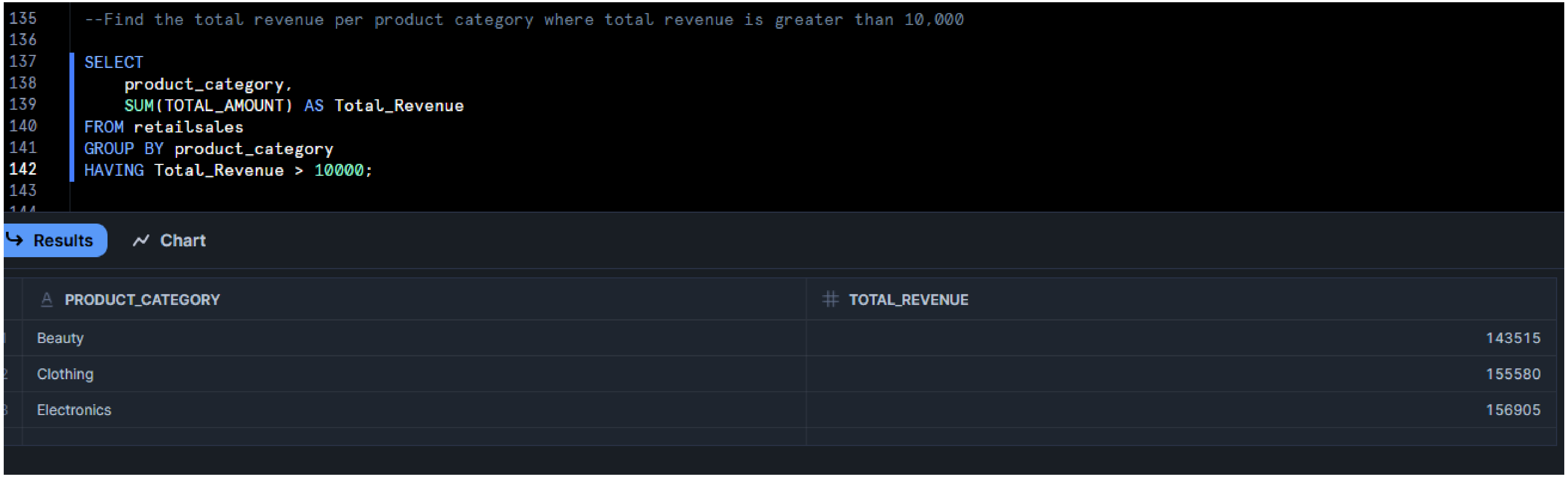
15. Find the number of transactions per Product Category.



```
--Find the average Price per Unit per product category

SELECT
    product_category,
    AVG(price_per_unit) AS Average_Price
FROM retailsales
GROUP BY product_category;
```

	PRODUCT_CATEGORY	TRANSACTION_COUNT
1	Beauty	307
2	Clothing	351
3	Electronics	342

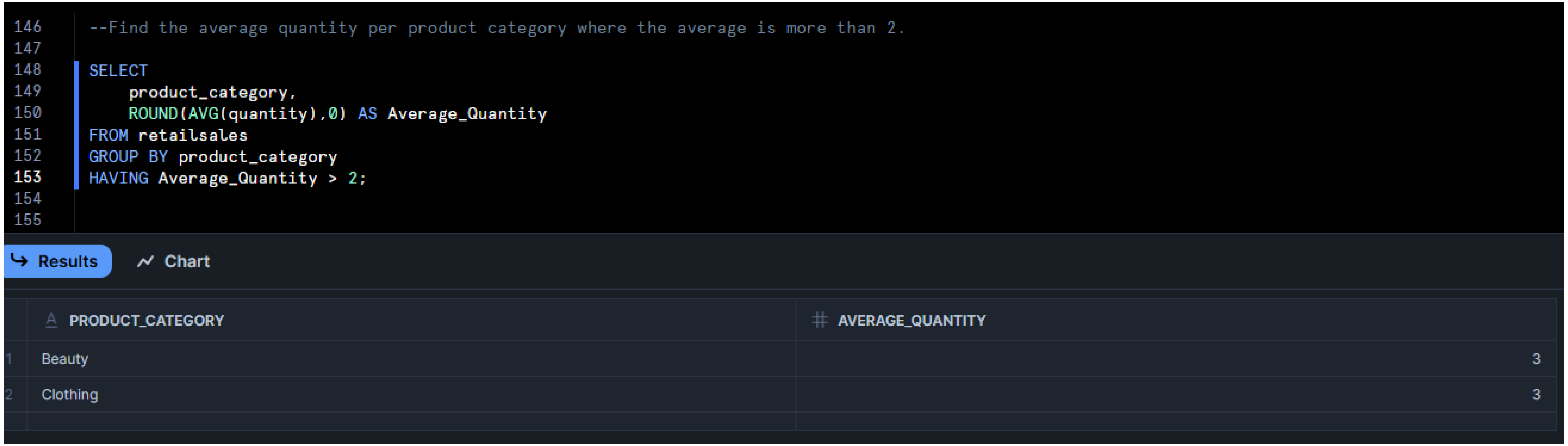
16. Find the total revenue per product category where total revenue is greater than 10,000



```sql
--Find the total revenue per product category where total revenue is greater than 10,000

SELECT
    product_category,
    SUM(TOTAL_AMOUNT) AS Total_Revenue
FROM retailsales
GROUP BY product_category
HAVING Total_Revenue > 10000;
```

	PRODUCT_CATEGORY	TOTAL_REVENUE
1	Beauty	143515
2	Clothing	155580
3	Electronics	156905

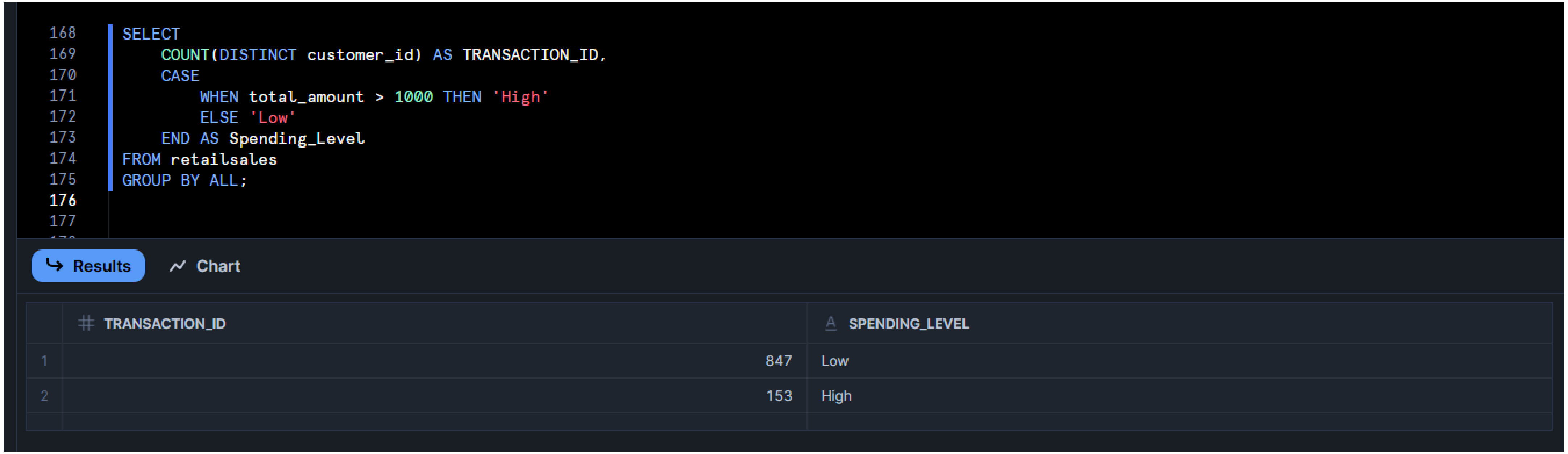
17. Find the average quantity per product category where the average is more than 2



```
--Find the average quantity per product category where the average is more than 2.

SELECT
    product_category,
    ROUND(AVG(quantity),0) AS Average_Quantity
FROM retailsales
GROUP BY product_category
HAVING Average_Quantity > 2;
```

	PRODUCT_CATEGORY	AVERAGE_QUANTITY
1	Beauty	3
2	Clothing	3

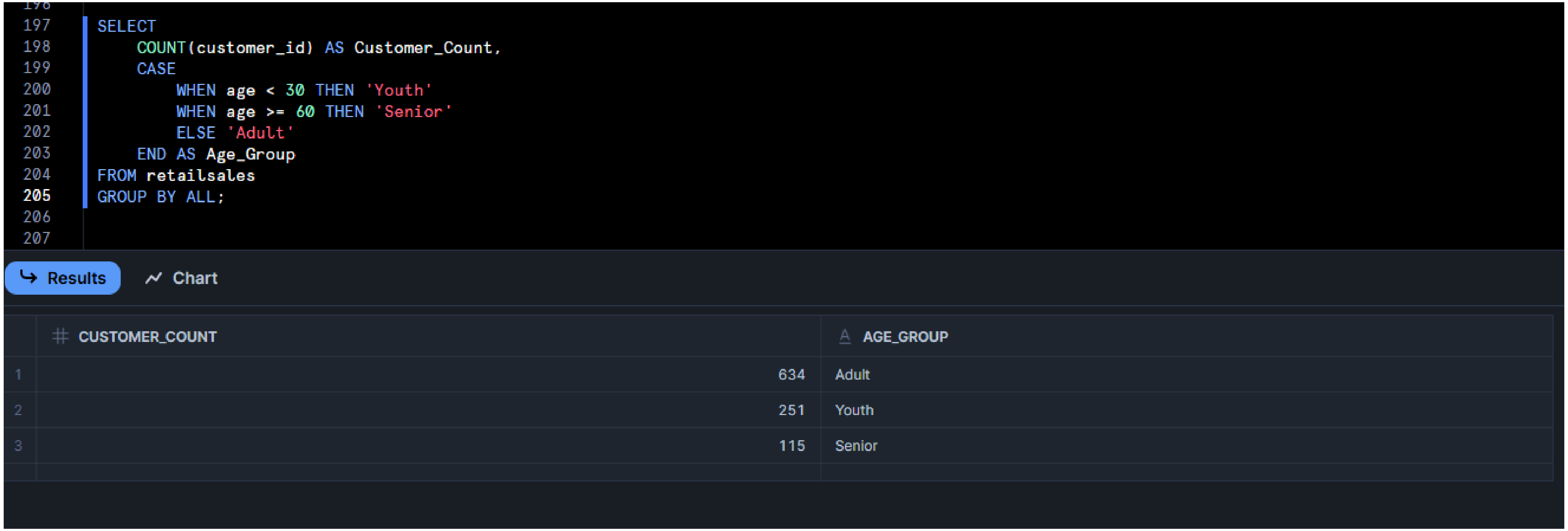
18. Display a column called Spending_Level that shows 'High' if Total Amount > 1000, otherwise 'Low'



```
SELECT
    COUNT(DISTINCT customer_id) AS TRANSACTION_ID,
    CASE
        WHEN total_amount > 1000 THEN 'High'
        ELSE 'Low'
    END AS Spending_Level
FROM retailsales
GROUP BY ALL;
```

	TRANSACTION_ID	SPENDING_LEVEL
1	847	Low
2	153	High

19. Display a new column called Age_Group that labels customers



```sql
SELECT
    COUNT(customer_id) AS Customer_Count,
    CASE
        WHEN age < 30 THEN 'Youth'
        WHEN age >= 60 THEN 'Senior'
        ELSE 'Adult'
    END AS Age_Group
FROM retailsales
GROUP BY ALL;
```

	CUSTOMER_COUNT	AGE_GROUP
1	634	Adult
2	251	Youth
3	115	Senior