RETAIL SALES ANALYSIS USING SQL

By
Jagan Saravana

Project Overview

This project showcases essential SQL skills and techniques commonly used by data analysts to explore, clean, and analyze retail sales data. It involves setting up a retail sales database, conducting exploratory data analysis (EDA), and using SQL queries to answer key business questions. Ideal for beginners in data analysis, this project helps build a strong foundation in SQL.

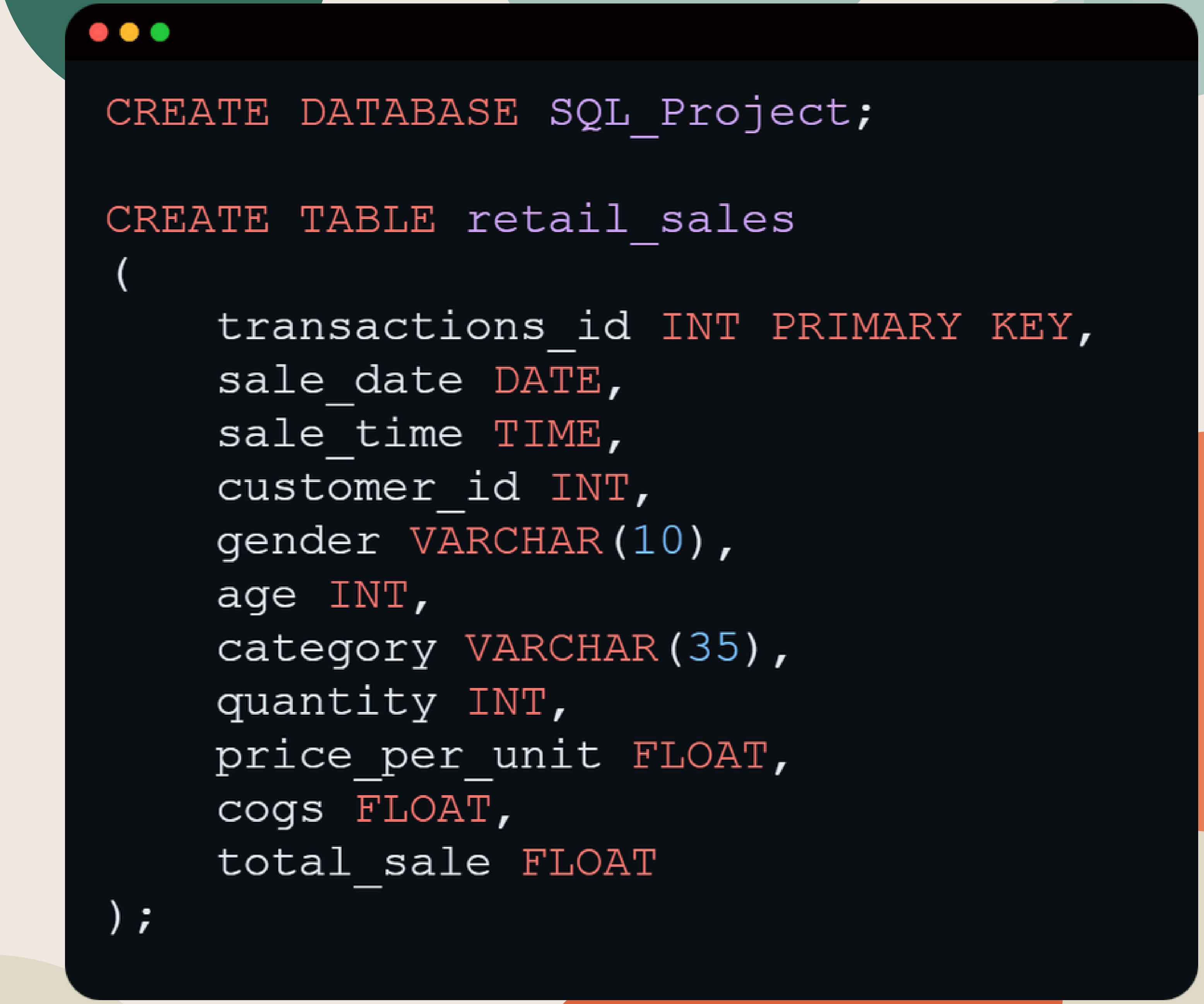
Objectives

1. **Set up a Retail Sales Database:** Create and populate a retail sales database using the provided sales data.
2. **Data Cleaning:** Detect and remove records containing missing or null values.
3. **Exploratory Data Analysis (EDA)**: Conduct basic EDA to gain a clear understanding of the dataset.
4. **Business Analysis:** Leverage SQL queries to answer key business questions and extract meaningful insights from the sales data.

Project Structure

1. Database Setup

- **Database Creation**: The project starts by creating a database named SQL_Project.
- **Table Creation:** A table named retail_sales is created to store the sales data. The table structure includes columns for transaction ID, sale date, sale time, customer ID, gender, age, product category, quantity sold, price per unit, cost of goods sold (COGS), and total sale amount.

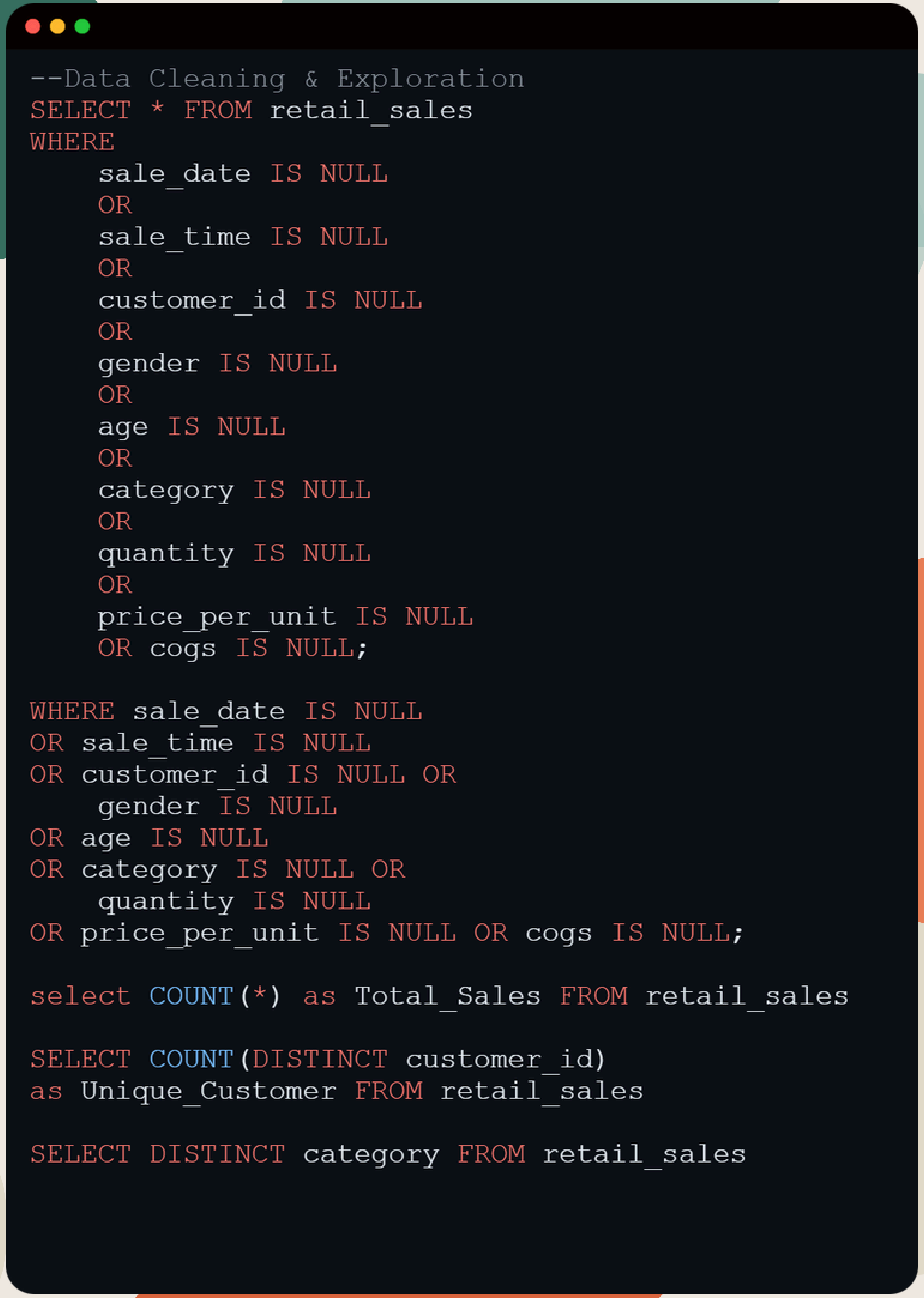


```
CREATE DATABASE SQL_Project;

CREATE TABLE retail_sales
(
    transactions_id INT PRIMARY KEY,
    sale_date DATE,
    sale_time TIME,
    customer_id INT,
    gender VARCHAR(10),
    age INT,
    category VARCHAR(35),
    quantity INT,
    price_per_unit FLOAT,
    cogs FLOAT,
    total_sale FLOAT
);
```

Project Structure

2. Data Exploration & Cleaning

- **Record Count:** Determine the total number of records in the dataset.
- **Customer Count**: Find out how many unique customers are in the dataset.
- **Category Count:** Identify all unique product categories in the dataset.
- **Null Value Check**: Check for any null values in the dataset and delete records with missing data.



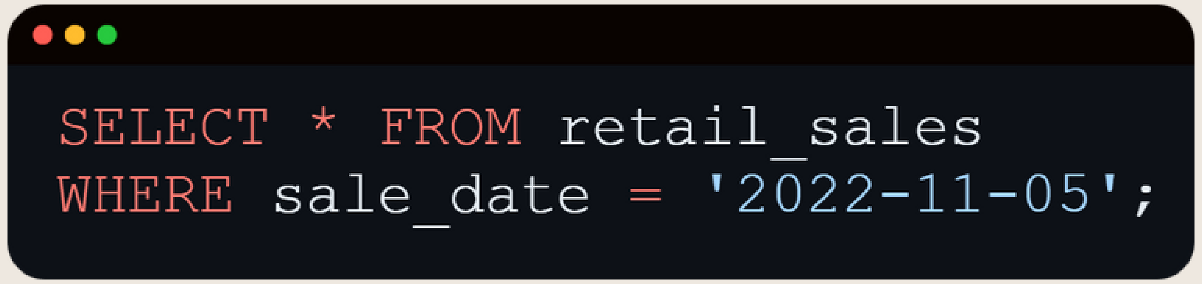
```sql
--Data Cleaning & Exploration
SELECT * FROM retail_sales
WHERE
    sale_date IS NULL
    OR
    sale_time IS NULL
    OR
    customer_id IS NULL
    OR
    gender IS NULL
    OR
    age IS NULL
    OR
    category IS NULL
    OR
    quantity IS NULL
    OR
    price_per_unit IS NULL
    OR cogs IS NULL;

WHERE sale_date IS NULL
OR sale_time IS NULL
OR customer_id IS NULL OR
    gender IS NULL
OR age IS NULL
OR category IS NULL OR
    quantity IS NULL
OR price_per_unit IS NULL OR cogs IS NULL;

select COUNT(*) as Total_Sales FROM retail_sales

SELECT COUNT(DISTINCT customer_id)
as Unique_Customer FROM retail_sales

SELECT DISTINCT category FROM retail_sales
```

Project Structure

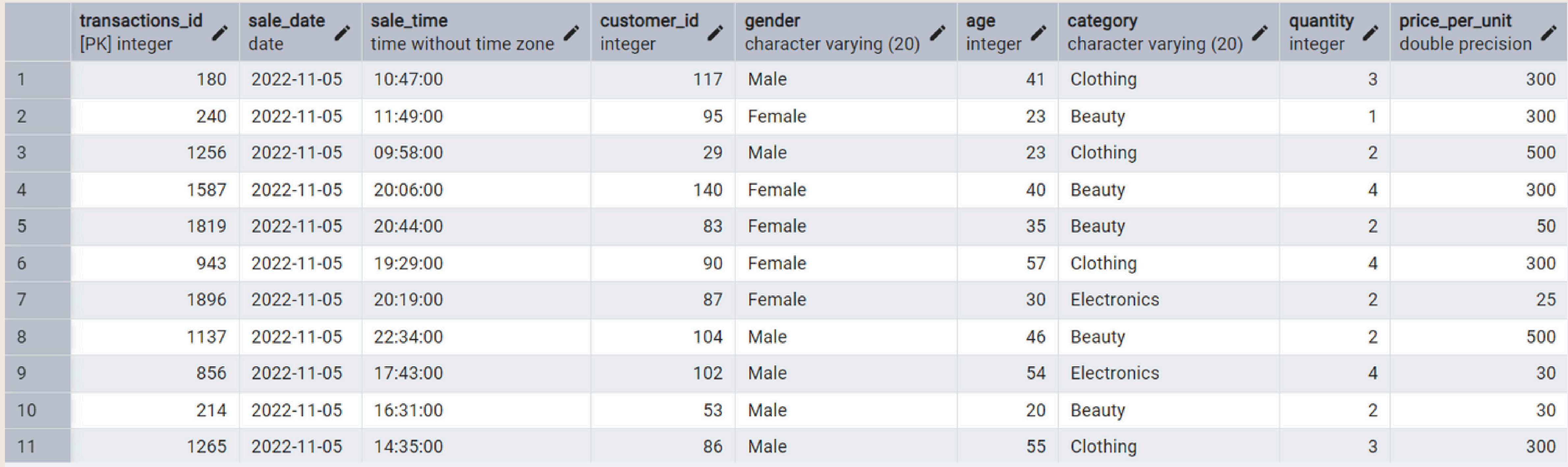
3. Data Analysis & Findings

The following SQL queries were developed to answer specific business questions:

1. Write a SQL query to retrieve all columns for sales made on '2022-11-05:

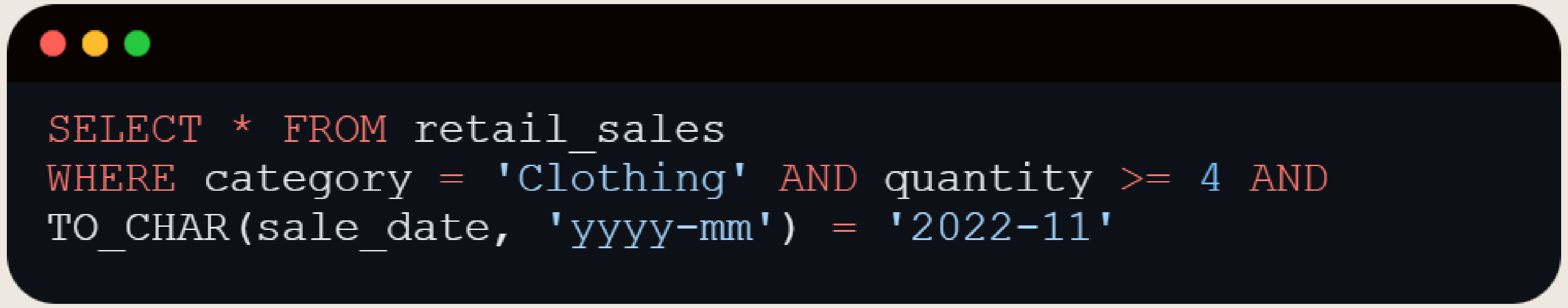


```sql
SELECT * FROM retail_sales
WHERE sale_date = '2022-11-05';
```

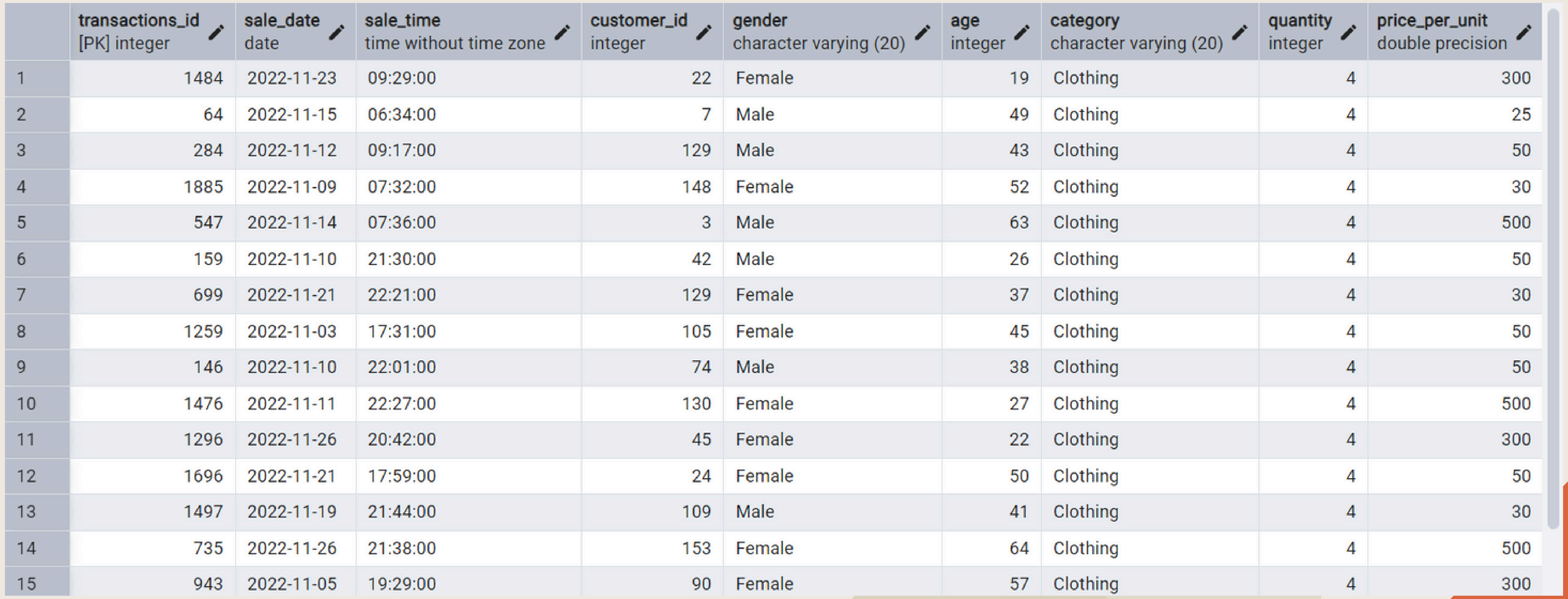


	transactions_id [PK] integer	sale_date date	sale_time time without time zone	customer_id integer	gender character varying (20)	age integer	category character varying (20)	quantity integer	price_per_unit double precision
1	180	2022-11-05	10:47:00	117	Male	41	Clothing	3	300
2	240	2022-11-05	11:49:00	95	Female	23	Beauty	1	300
3	1256	2022-11-05	09:58:00	29	Male	23	Clothing	2	500
4	1587	2022-11-05	20:06:00	140	Female	40	Beauty	4	300
5	1819	2022-11-05	20:44:00	83	Female	35	Beauty	2	50
6	943	2022-11-05	19:29:00	90	Female	57	Clothing	4	300
7	1896	2022-11-05	20:19:00	87	Female	30	Electronics	2	25
8	1137	2022-11-05	22:34:00	104	Male	46	Beauty	2	500
9	856	2022-11-05	17:43:00	102	Male	54	Electronics	4	30
10	214	2022-11-05	16:31:00	53	Male	20	Beauty	2	30
11	1265	2022-11-05	14:35:00	86	Male	55	Clothing	3	300

2. Write a SQL query to retrieve all transactions where the category is 'Clothing' and the quantity sold is more than 4 in the month of Nov-2022

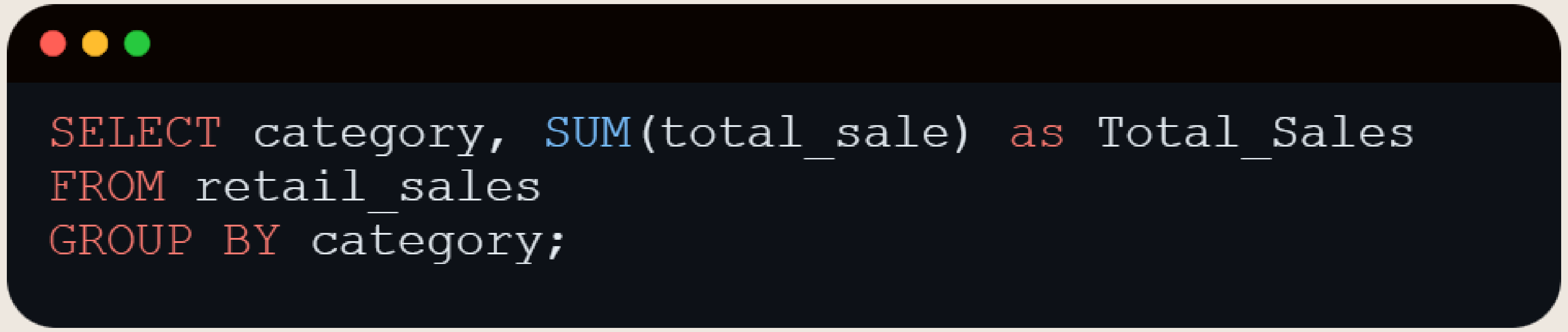


```
SELECT * FROM retail_sales
WHERE category = 'Clothing' AND quantity >= 4 AND
TO_CHAR(sale_date, 'yyyy-mm') = '2022-11'
```

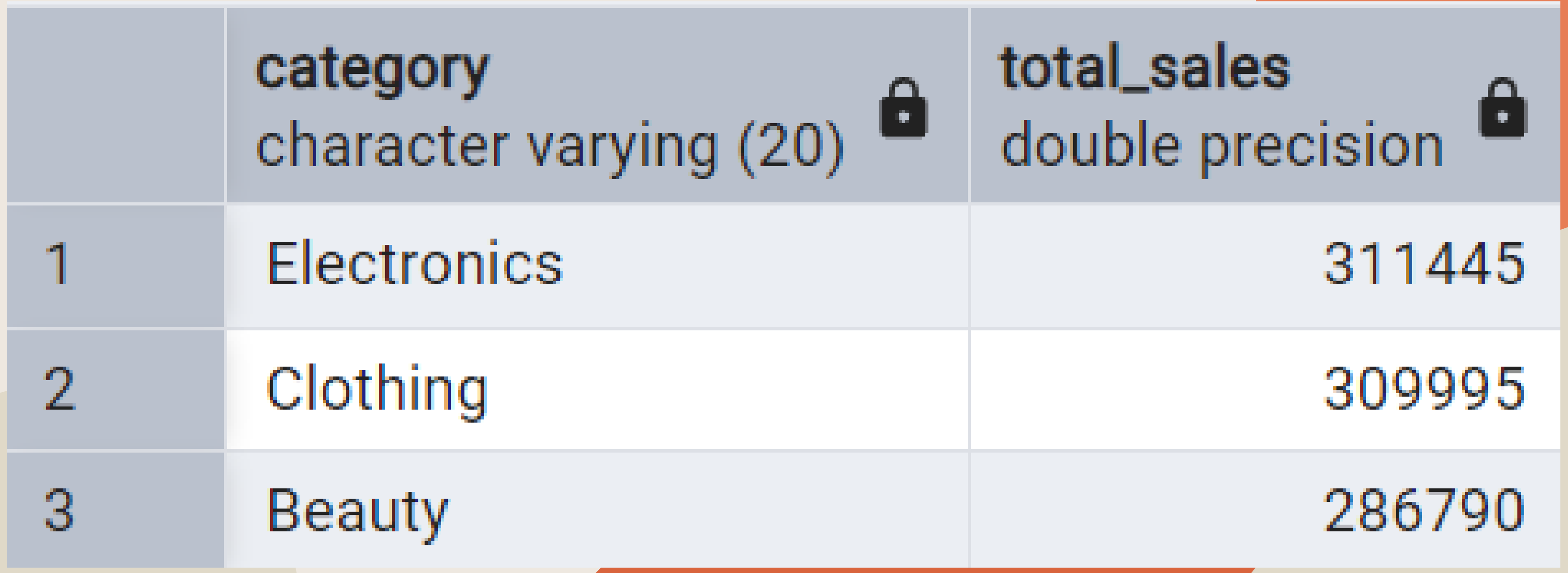


	transactions_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit
	[PK] integer	date	time without time zone	integer	character varying (20)	integer	character varying (20)	integer	double precision
1	1484	2022-11-23	09:29:00	22	Female	19	Clothing	4	300
2	64	2022-11-15	06:34:00	7	Male	49	Clothing	4	25
3	284	2022-11-12	09:17:00	129	Male	43	Clothing	4	50
4	1885	2022-11-09	07:32:00	148	Female	52	Clothing	4	30
5	547	2022-11-14	07:36:00	3	Male	63	Clothing	4	500
6	159	2022-11-10	21:30:00	42	Male	26	Clothing	4	50
7	699	2022-11-21	22:21:00	129	Female	37	Clothing	4	30
8	1259	2022-11-03	17:31:00	105	Female	45	Clothing	4	50
9	146	2022-11-10	22:01:00	74	Male	38	Clothing	4	50
10	1476	2022-11-11	22:27:00	130	Female	27	Clothing	4	500
11	1296	2022-11-26	20:42:00	45	Female	22	Clothing	4	300
12	1696	2022-11-21	17:59:00	24	Female	50	Clothing	4	50
13	1497	2022-11-19	21:44:00	109	Male	41	Clothing	4	30
14	735	2022-11-26	21:38:00	153	Female	64	Clothing	4	500
15	943	2022-11-05	19:29:00	90	Female	57	Clothing	4	300

3. Write a SQL query to calculate the total sales (total_sale) for each category.

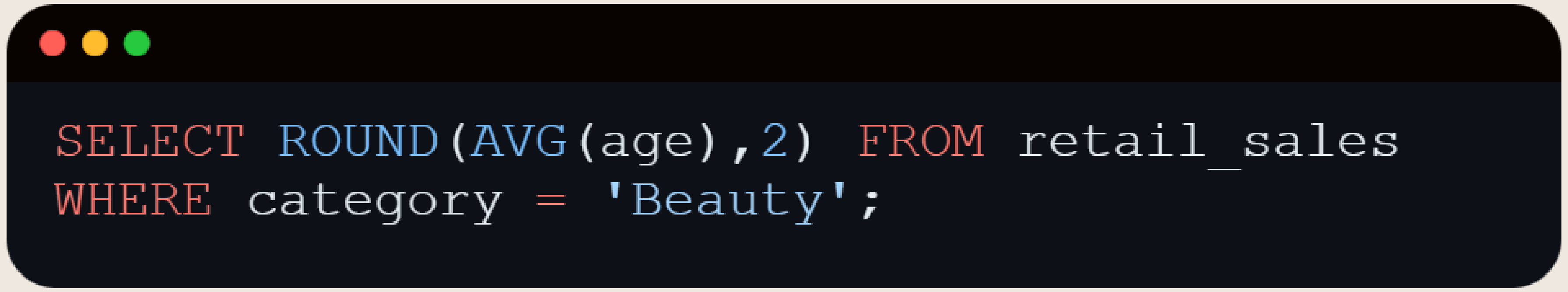


```
SELECT category, SUM(total_sale) as Total_Sales
FROM retail_sales
GROUP BY category;
```

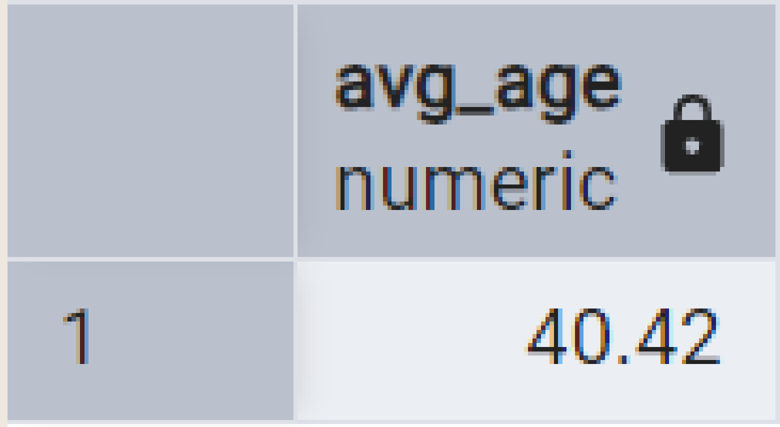


	category character varying (20)	total_sales double precision
1	Electronics	311445
2	Clothing	309995
3	Beauty	286790

4. Write a SQL query to find the average age of customers who purchased items from the 'Beauty' category.

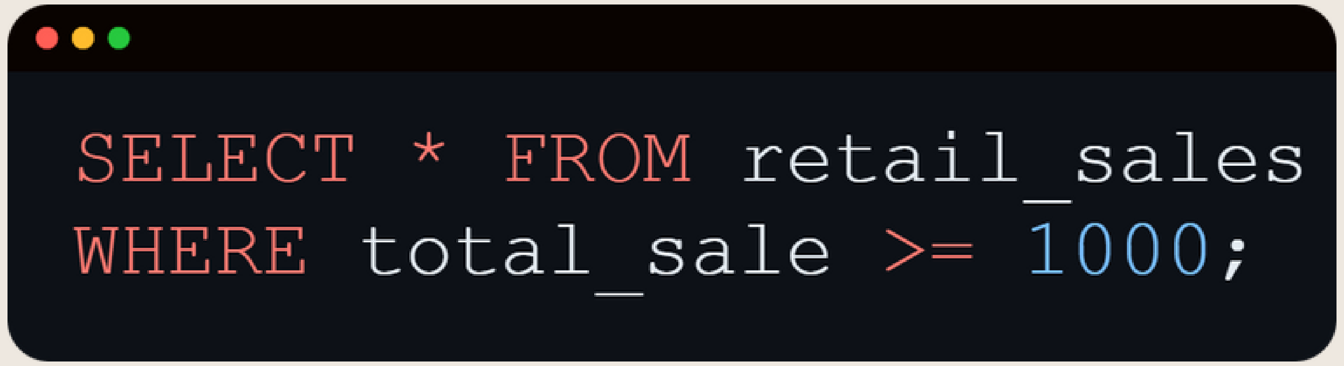


```
SELECT ROUND(AVG(age),2) FROM retail_sales
WHERE category = 'Beauty';
```

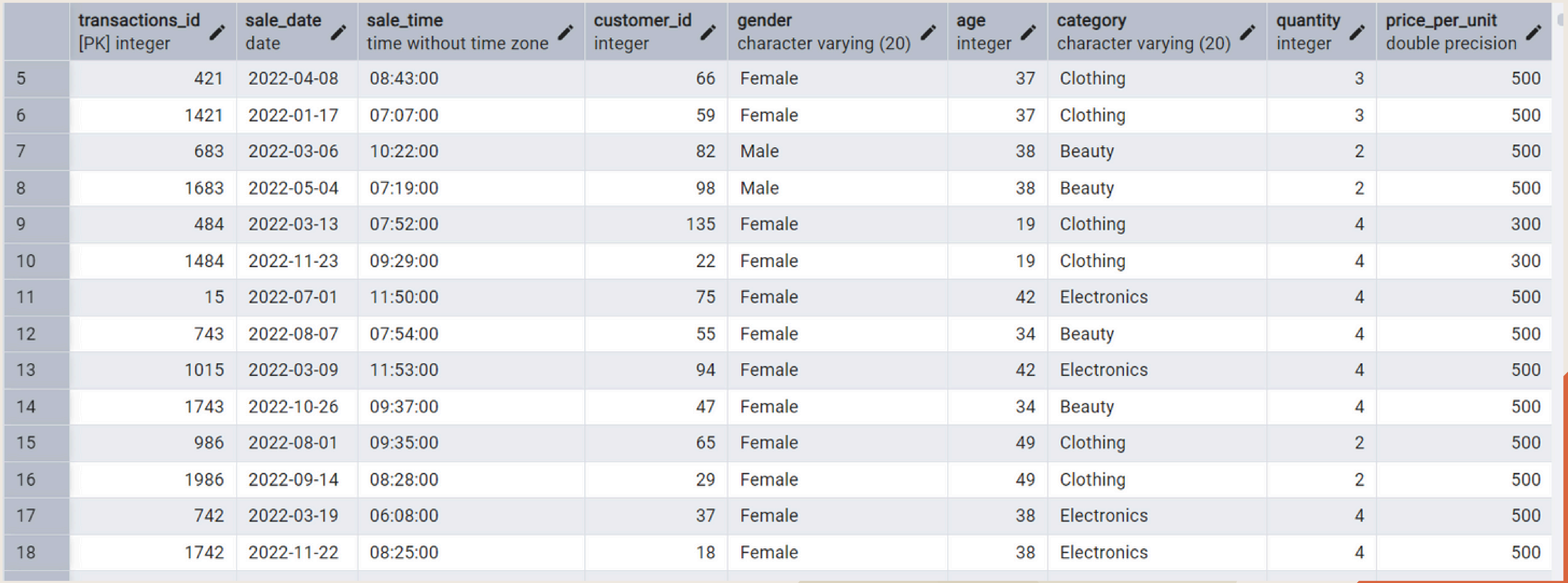


	avg_age numeric
1	40.42

5. Write a SQL query to find all transactions where the total_sale is greater than 1000.

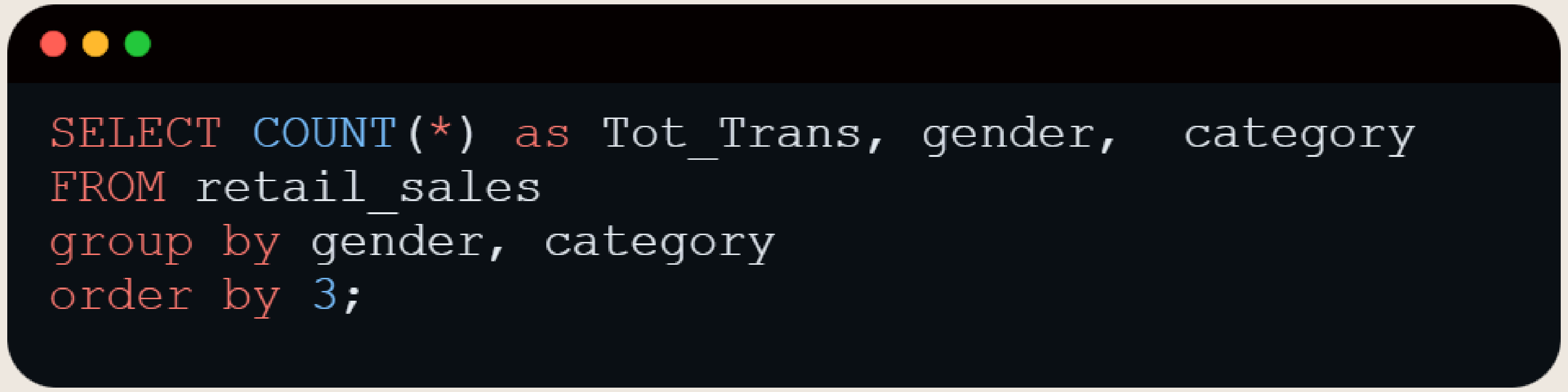


```
SELECT * FROM retail_sales
WHERE total_sale >= 1000;
```

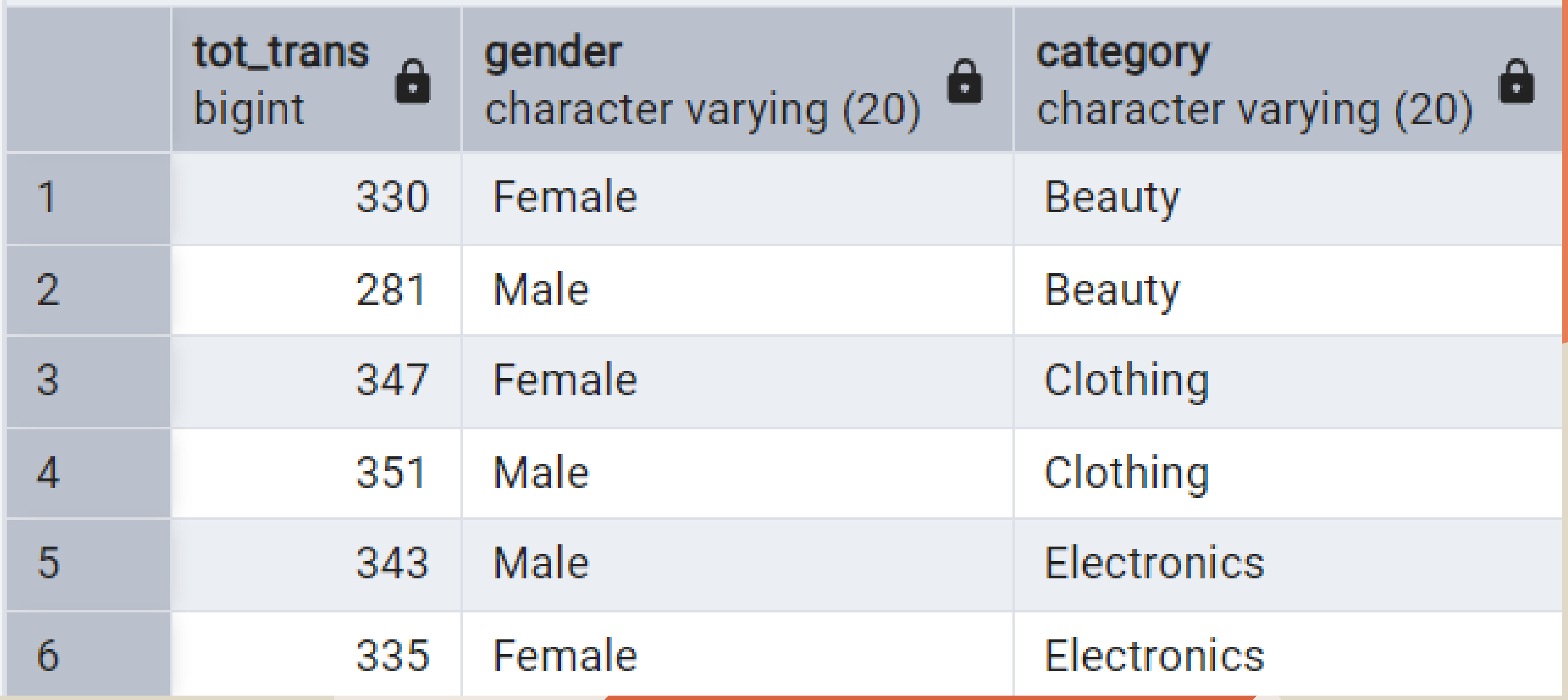


	transactions_id [PK] integer	sale_date date	sale_time time without time zone	customer_id integer	gender character varying (20)	age integer	category character varying (20)	quantity integer	price_per_unit double precision
5	421	2022-04-08	08:43:00	66	Female	37	Clothing	3	500
6	1421	2022-01-17	07:07:00	59	Female	37	Clothing	3	500
7	683	2022-03-06	10:22:00	82	Male	38	Beauty	2	500
8	1683	2022-05-04	07:19:00	98	Male	38	Beauty	2	500
9	484	2022-03-13	07:52:00	135	Female	19	Clothing	4	300
10	1484	2022-11-23	09:29:00	22	Female	19	Clothing	4	300
11	15	2022-07-01	11:50:00	75	Female	42	Electronics	4	500
12	743	2022-08-07	07:54:00	55	Female	34	Beauty	4	500
13	1015	2022-03-09	11:53:00	94	Female	42	Electronics	4	500
14	1743	2022-10-26	09:37:00	47	Female	34	Beauty	4	500
15	986	2022-08-01	09:35:00	65	Female	49	Clothing	2	500
16	1986	2022-09-14	08:28:00	29	Female	49	Clothing	2	500
17	742	2022-03-19	06:08:00	37	Female	38	Electronics	4	500
18	1742	2022-11-22	08:25:00	18	Female	38	Electronics	4	500

6. Write a SQL query to find the total number of transactions (transaction_id) made by each gender in each category.

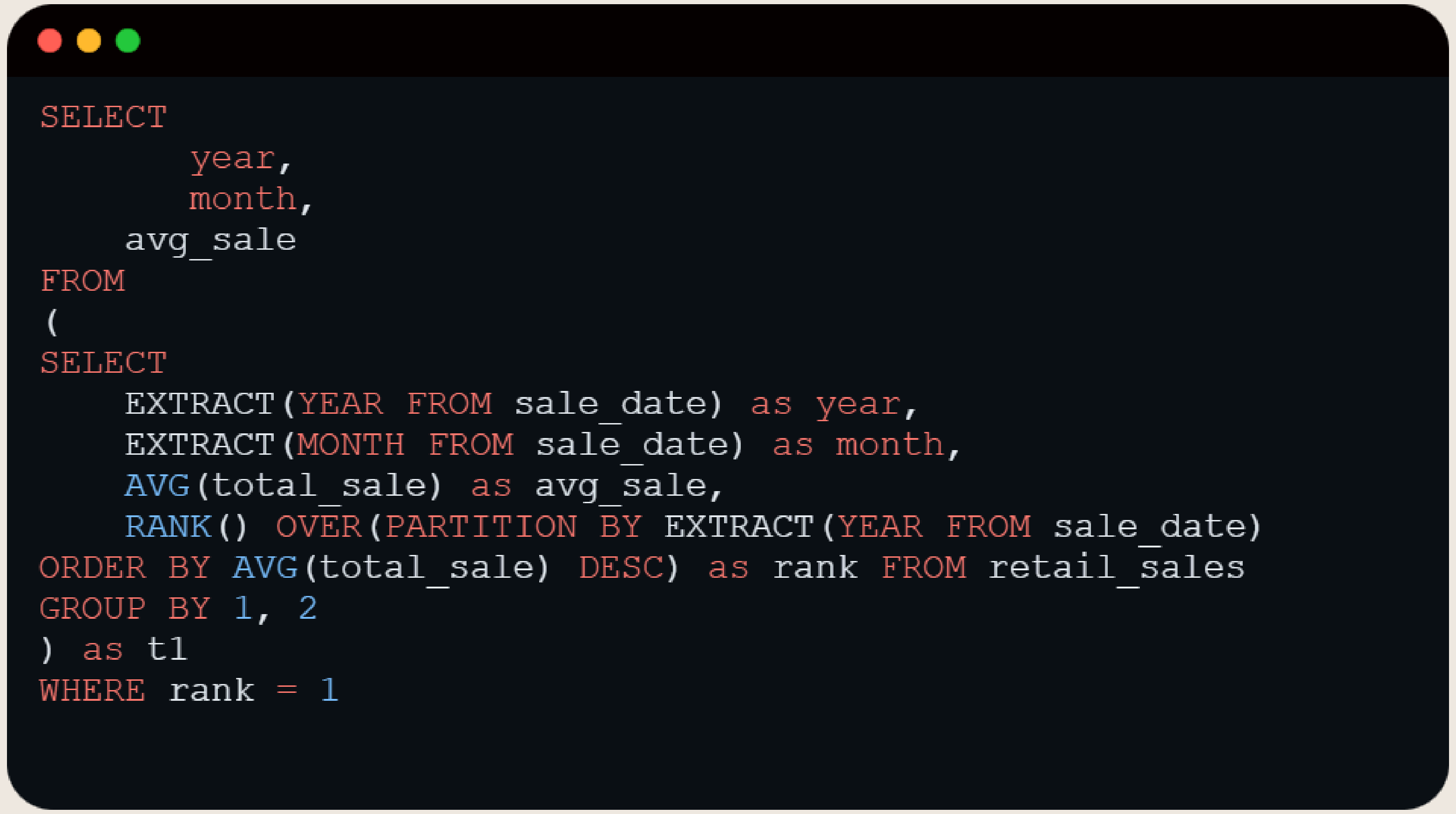


```
SELECT COUNT(*) as Tot_Trans, gender,  category
FROM retail_sales
group by gender, category
order by 3;
```

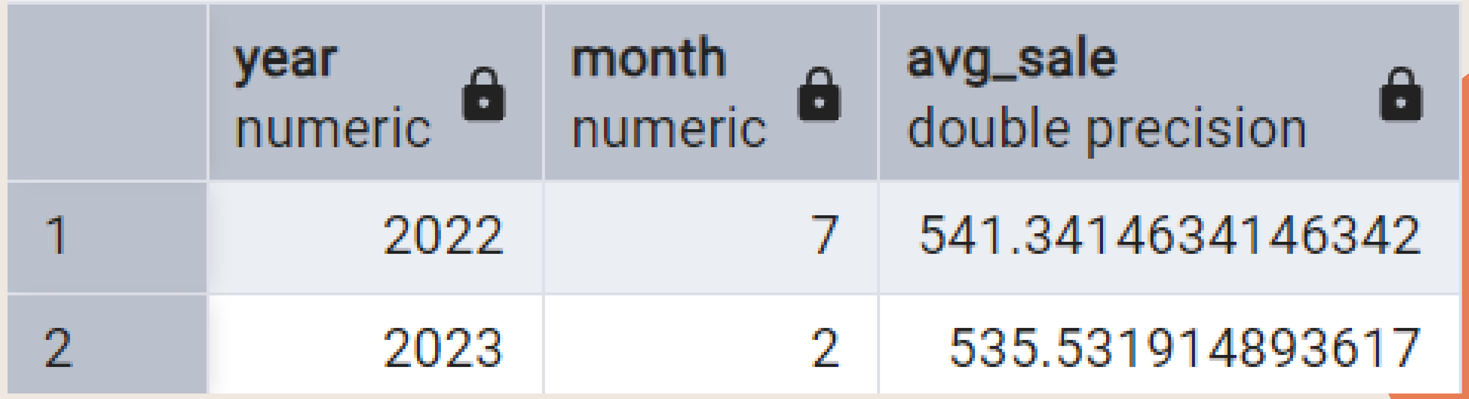


	tot_trans bigint	gender character varying (20)	category character varying (20)
1	330	Female	Beauty
2	281	Male	Beauty
3	347	Female	Clothing
4	351	Male	Clothing
5	343	Male	Electronics
6	335	Female	Electronics

7. Write a SQL query to calculate the average sale for each month. Find out best selling month in each year

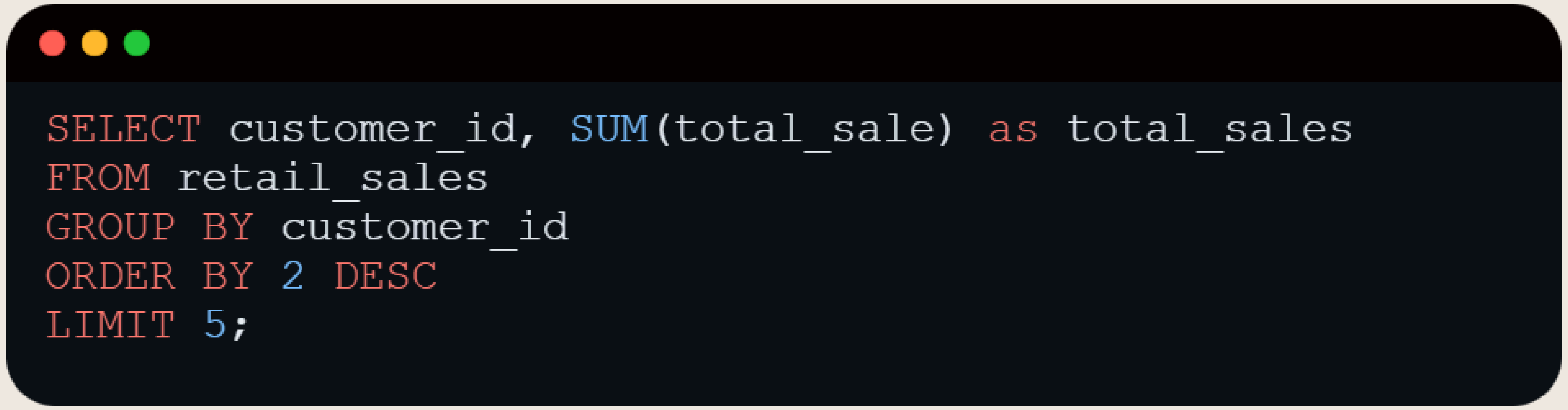


```
SELECT
       year,
       month,
    avg_sale
FROM
(
SELECT
    EXTRACT(YEAR FROM sale_date) as year,
    EXTRACT(MONTH FROM sale_date) as month,
    AVG(total_sale) as avg_sale,
    RANK() OVER(PARTITION BY EXTRACT(YEAR FROM sale_date)
ORDER BY AVG(total_sale) DESC) as rank FROM retail_sales
GROUP BY 1, 2
) as t1
WHERE rank = 1
```

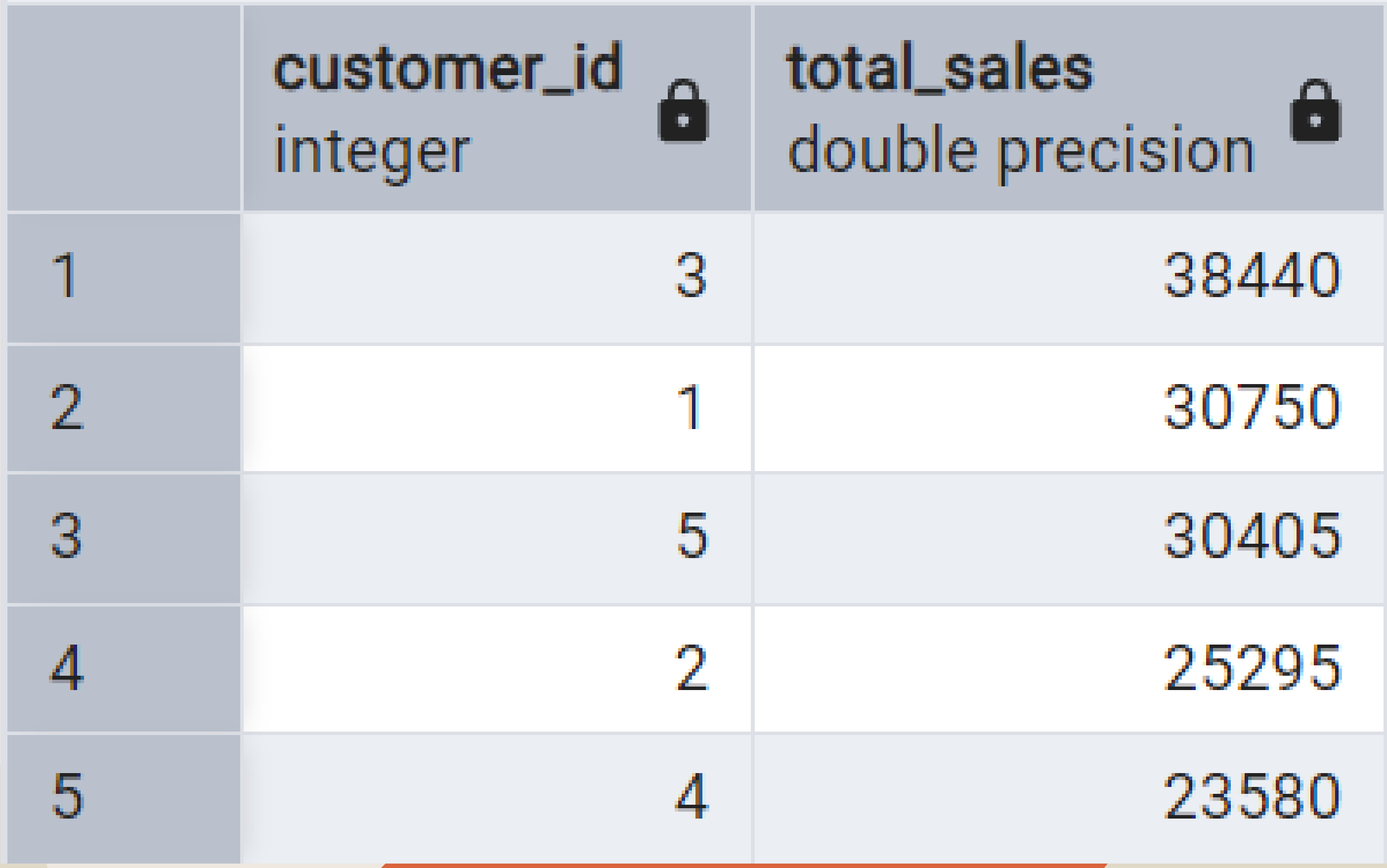


	year numeric	month numeric	avg_sale double precision
1	2022	7	541.341463414634
2	2023	2	535.531914893617

8. Write a SQL query to find the top 5 customers based on the highest total sales

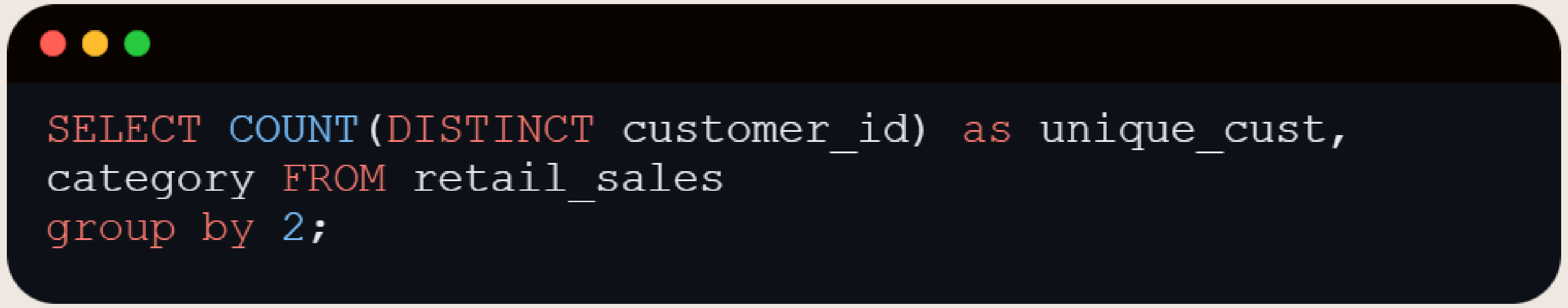


```sql
SELECT customer_id, SUM(total_sale) as total_sales
FROM retail_sales
GROUP BY customer_id
ORDER BY 2 DESC
LIMIT 5;
```

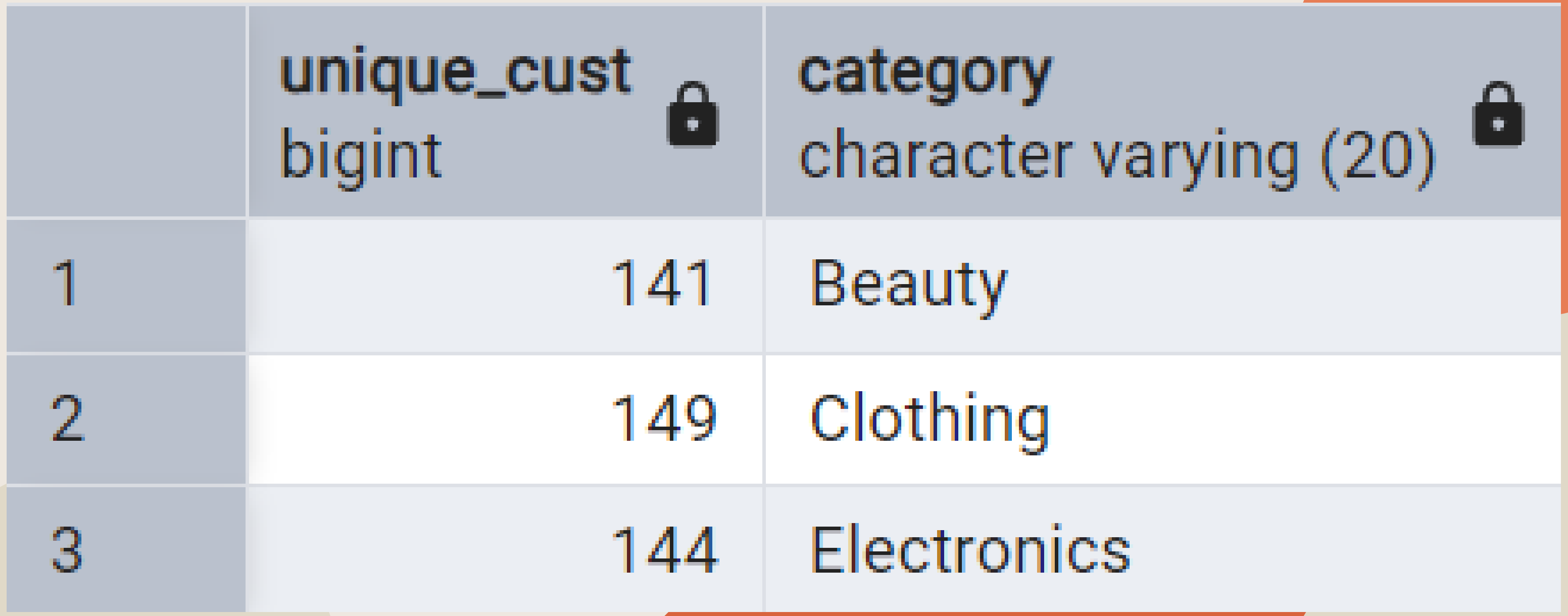


	customer_id integer	total_sales double precision
1	3	38440
2	1	30750
3	5	30405
4	2	25295
5	4	23580

9. Write a SQL query to find the number of unique customers who purchased items from each category.

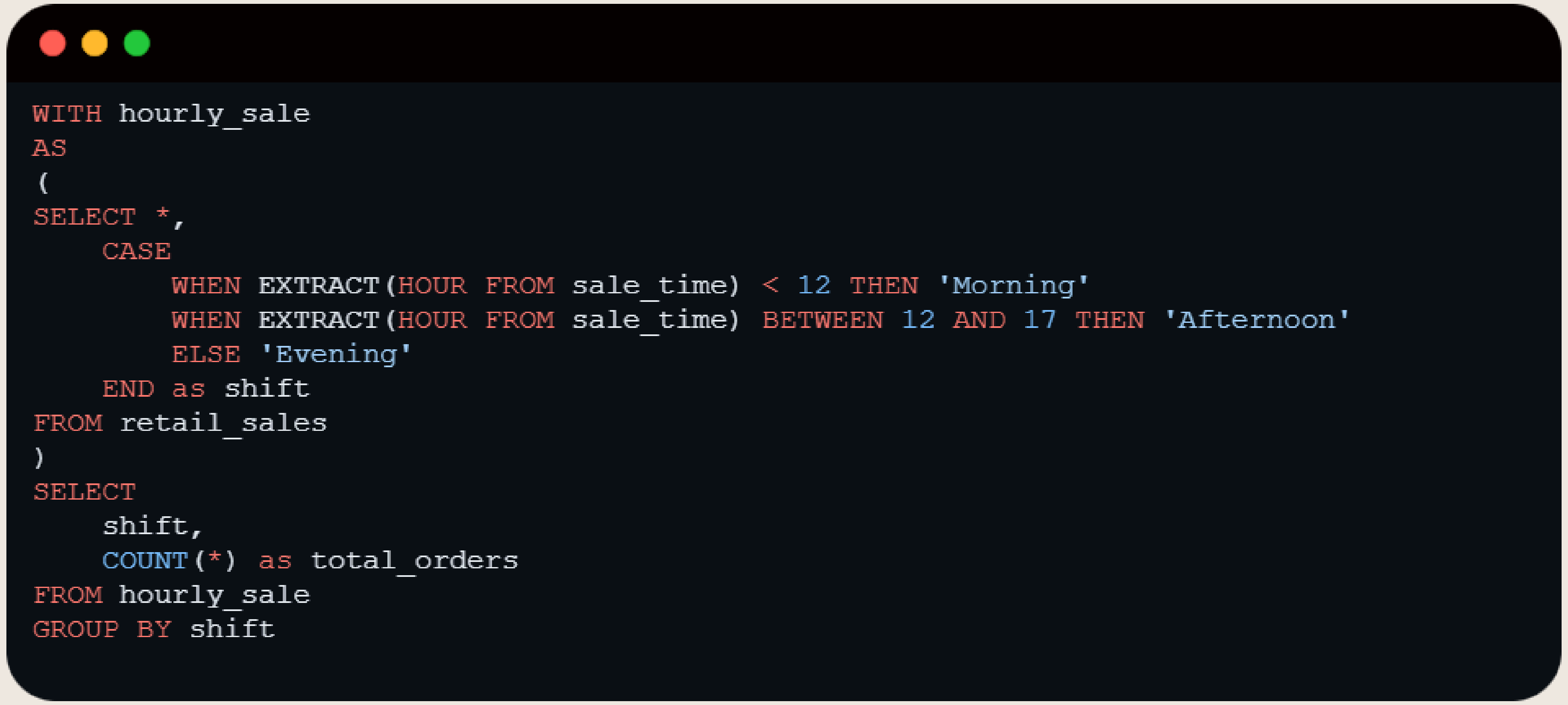


```sql
SELECT COUNT(DISTINCT customer_id) as unique_cust,
category FROM retail_sales
group by 2;
```

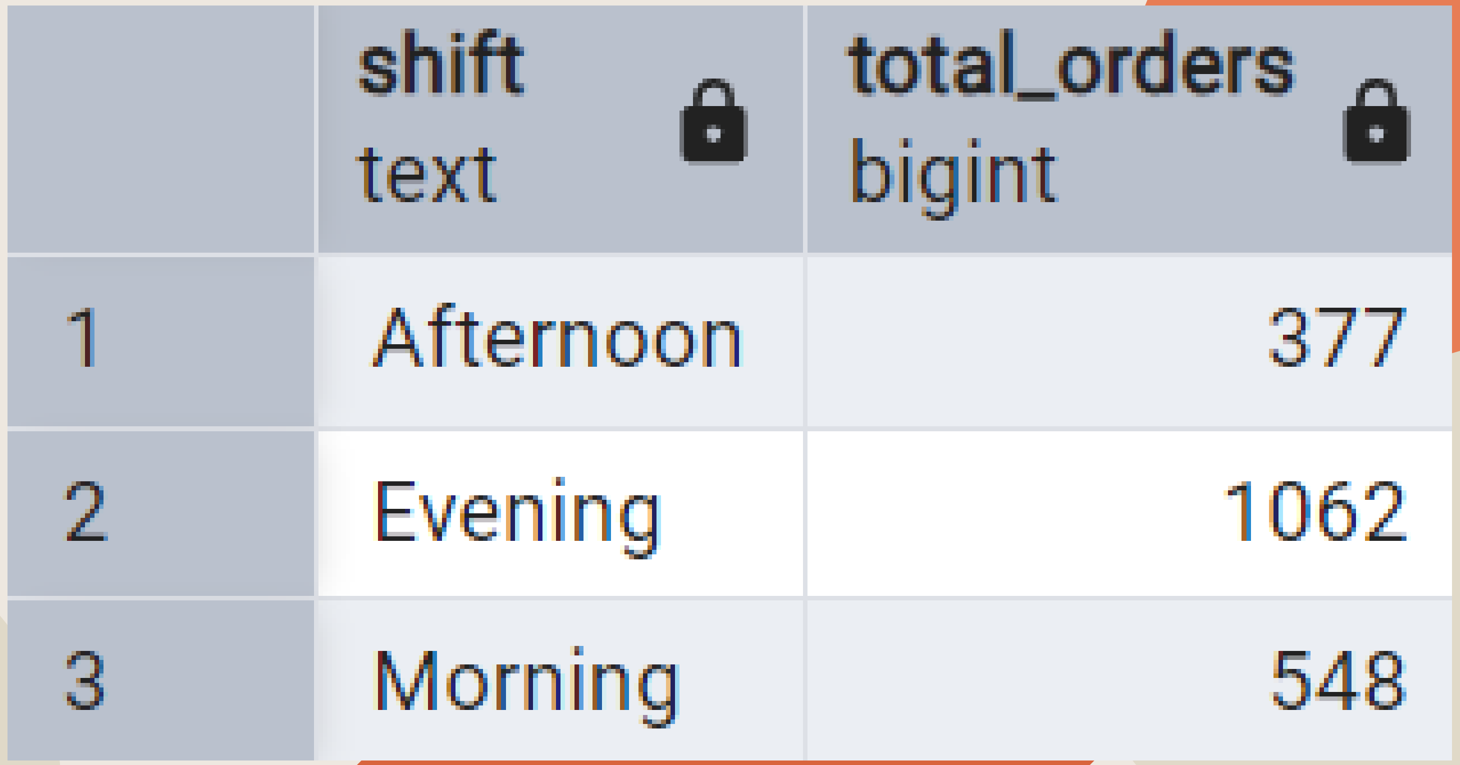


	unique_cust bigint	category character varying (20)
1	141	Beauty
2	149	Clothing
3	144	Electronics

10. Write a SQL query to create each shift and number of orders (Example Morning <12, Afternoon Between 12 & 17, Evening >17)



```
WITH hourly_sale
AS
(
SELECT *,
    CASE
        WHEN EXTRACT(HOUR FROM sale_time) < 12 THEN 'Morning'
        WHEN EXTRACT(HOUR FROM sale_time) BETWEEN 12 AND 17 THEN 'Afternoon'
        ELSE 'Evening'
    END as shift
FROM retail_sales
)
SELECT
    shift,
    COUNT(*) as total_orders
FROM hourly_sale
GROUP BY shift
```



	shift text	total_orders bigint
1	Afternoon	377
2	Evening	1062
3	Morning	548

Insights

- **Customer Demographics**: The dataset covers customers from various age groups, with sales spread across different categories like Clothing and Beauty.
- **High-Value Transactions**: Multiple transactions exceeded a total sale amount of 1000, indicating premium purchases.
- **Sales Trends**: Monthly sales analysis reveals fluctuations, helping identify peak seasons.
- **Customer Insights**: The analysis highlights top-spending customers and the most popular product categories.

Conclusion

This project effectively demonstrates how SQL can be used for data analysis in a retail sales context. By setting up a structured database, cleaning and analyzing the data, and deriving meaningful business insights, we have explored key aspects of SQL for data-driven decision-making. The findings provide valuable insights into customer demographics, high-value transactions, sales trends, and customer purchasing behavior.

The results from this analysis can help businesses optimize sales strategies, improve customer targeting, and identify peak sales periods. This project serves as a strong foundation for those looking to enhance their SQL skills for data analysis and business intelligence applications.

THANK YOU