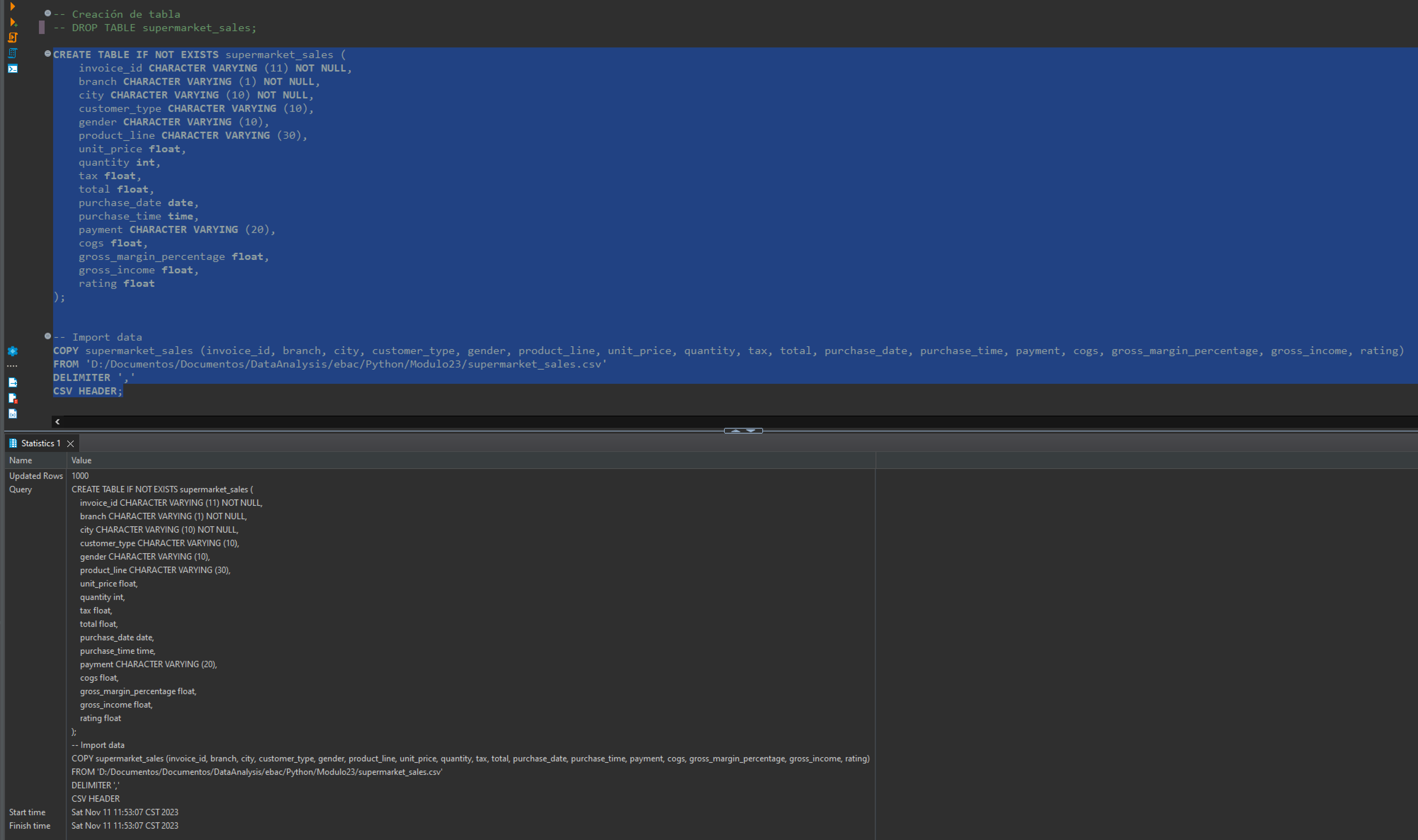


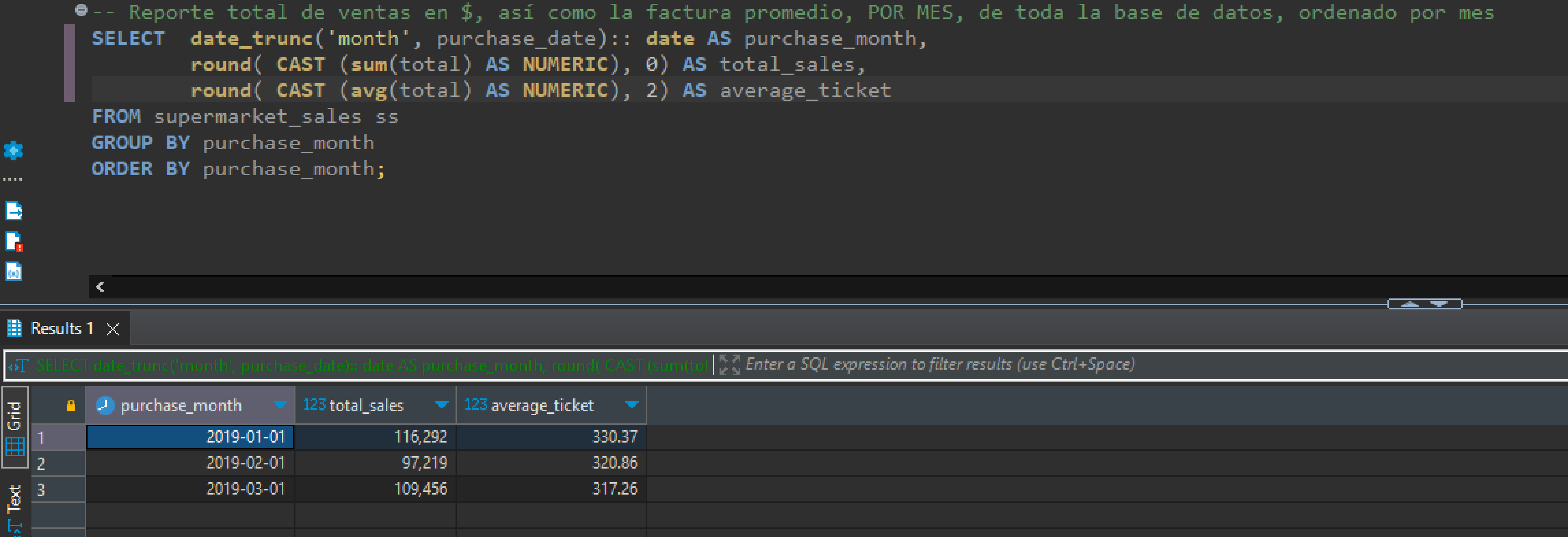
```sql
-- Creación de tabla
-- DROP TABLE supermarket_sales;

CREATE TABLE IF NOT EXISTS supermarket_sales (
    invoice_id CHARACTER VARYING (11) NOT NULL,
    branch CHARACTER VARYING (1) NOT NULL,
    city CHARACTER VARYING (10) NOT NULL,
    customer_type CHARACTER VARYING (10),
    gender CHARACTER VARYING (10),
    product_line CHARACTER VARYING (30),
    unit_price float,
    quantity int,
    tax float,
    total float,
    purchase_date date,
    purchase_time time,
    payment CHARACTER VARYING (20),
    cogs float,
    gross_margin_percentage float,
    gross_income float,
    rating float
);


-- Import data
COPY supermarket_sales (invoice_id, branch, city, customer_type, gender, product_line, unit_price, quantity, tax, total, purchase_date, purchase_time, payment, cogs, gross_margin_percentage, gross_income, rating)
FROM 'D:/Documentos/Documentos/DataAnalysis/ebac/Python/Modulo23/supermarket_sales.csv'
DELIMITER ','
CSV HEADER;
```

Statistics 1

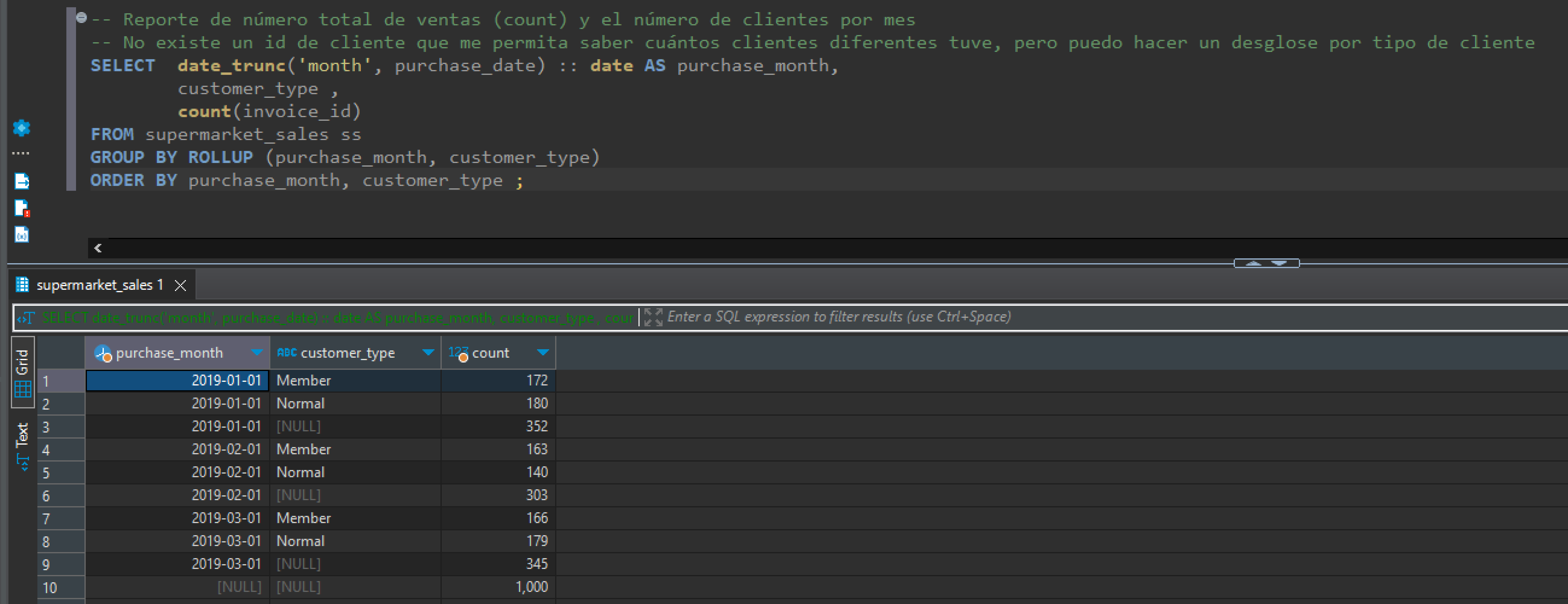
```
Name          Value
Updated Rows  1000
Query         CREATE TABLE IF NOT EXISTS supermarket_sales (
                  invoice_id CHARACTER VARYING (11) NOT NULL,
                  branch CHARACTER VARYING (1) NOT NULL,
                  city CHARACTER VARYING (10) NOT NULL,
                  customer_type CHARACTER VARYING (10),
                  gender CHARACTER VARYING (10),
                  product_line CHARACTER VARYING (30),
                  unit_price float,
                  quantity int,
                  tax float,
                  total float,
                  purchase_date date,
                  purchase_time time,
                  payment CHARACTER VARYING (20),
                  cogs float,
                  gross_margin_percentage float,
                  gross_income float,
                  rating float
              );
              -- Import data
              COPY supermarket_sales (invoice_id, branch, city, customer_type, gender, product_line, unit_price, quantity, tax, total, purchase_date, purchase_time, payment, cogs, gross_margin_percentage, gross_income, rating)
              FROM 'D:/Documentos/Documentos/DataAnalysis/ebac/Python/Modulo23/supermarket_sales.csv'
              DELIMITER ','
              CSV HEADER
Start time    Sat Nov 11 11:53:07 CST 2023
Finish time   Sat Nov 11 11:53:07 CST 2023
```



```sql
-- Reporte total de ventas en $, así como la factura promedio, POR MES, de toda la base de datos, ordenado por mes
SELECT  date_trunc('month', purchase_date):: date AS purchase_month,
        round( CAST (sum(total) AS NUMERIC), 0) AS total_sales,
        round( CAST (avg(total) AS NUMERIC), 2) AS average_ticket
FROM supermarket_sales ss
GROUP BY purchase_month
ORDER BY purchase_month;
```

Results 1
SELECT date_trunc('month', purchase_date):: date AS purchase_month, round(CAST (sum(tot
Enter a SQL expression to filter results (use Ctrl+Space)
Grid
Text

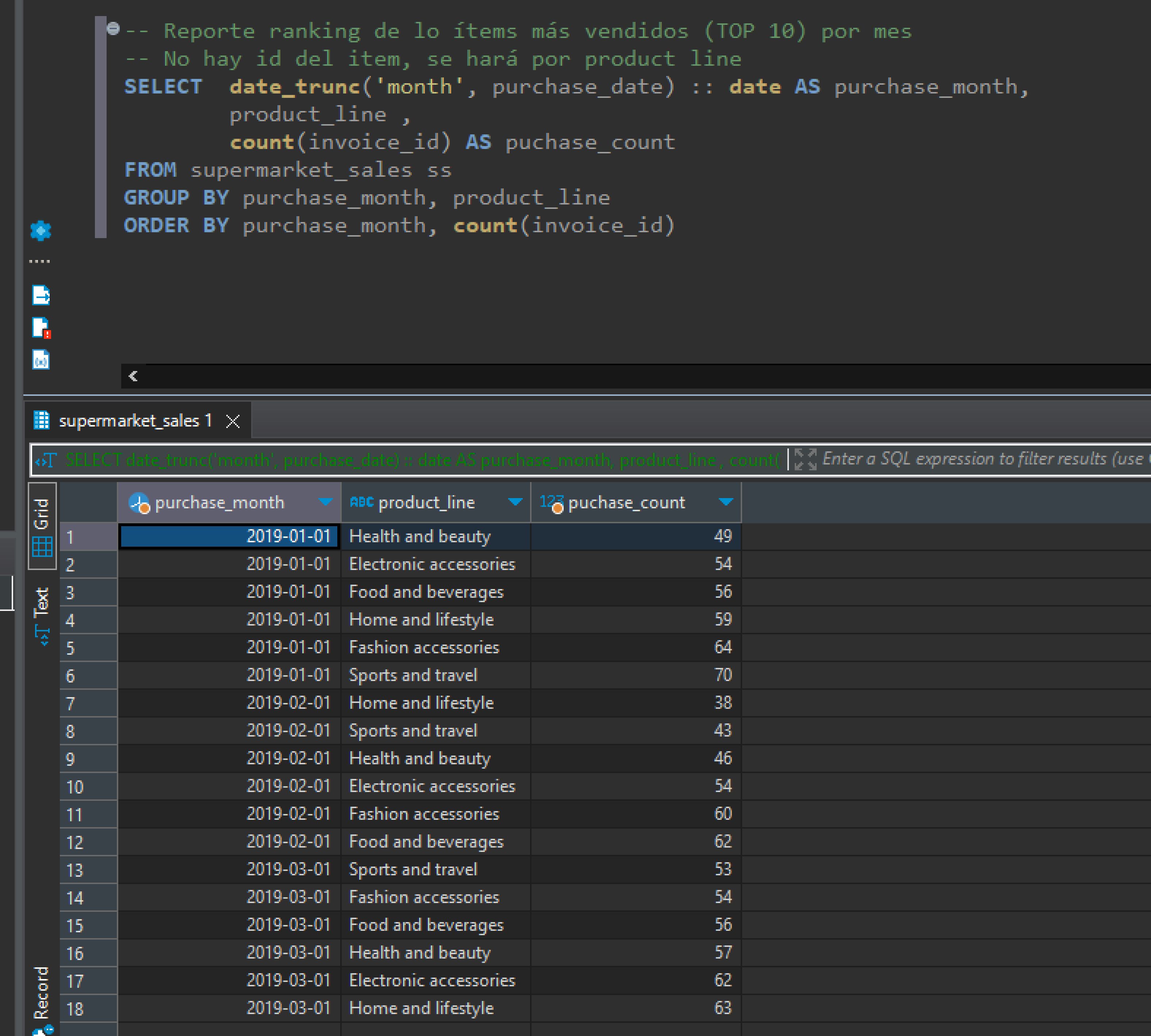
	purchase_month	total_sales	average_ticket
1	2019-01-01	116,292	330.37
2	2019-02-01	97,219	320.86
3	2019-03-01	109,456	317.26



```sql
-- Reporte de número total de ventas (count) y el número de clientes por mes
-- No existe un id de cliente que me permita saber cuántos clientes diferentes tuve, pero puedo hacer un desglose por tipo de cliente
SELECT  date_trunc('month', purchase_date) :: date AS purchase_month,
        customer_type ,
        count(invoice_id)
FROM supermarket_sales ss
GROUP BY ROLLUP (purchase_month, customer_type)
ORDER BY purchase_month, customer_type ;
```

supermarket_sales 1
SELECT date_trunc('month', purchase_date) :: date AS purchase_month, customer_type , cour
Enter a SQL expression to filter results (use Ctrl+Space)
Grid
Text

	purchase_month	customer_type	count
1	2019-01-01	Member	172
2	2019-01-01	Normal	180
3	2019-01-01	[NULL]	352
4	2019-02-01	Member	163
5	2019-02-01	Normal	140
6	2019-02-01	[NULL]	303
7	2019-03-01	Member	166
8	2019-03-01	Normal	179
9	2019-03-01	[NULL]	345
10	[NULL]	[NULL]	1,000

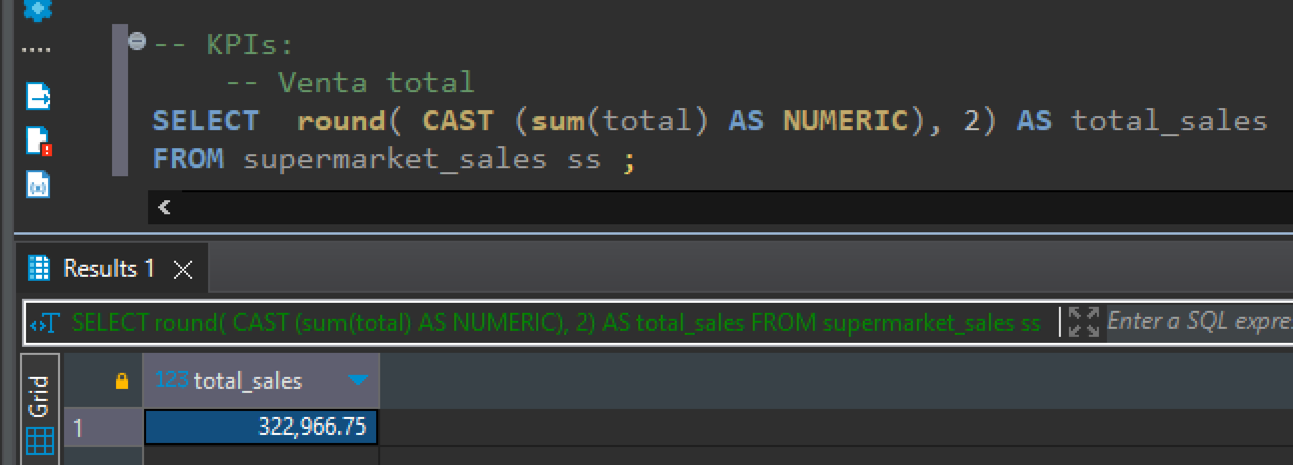


```sql
-- Reporte ranking de lo ítems más vendidos (TOP 10) por mes
-- No hay id del item, se hará por product line
SELECT  date_trunc('month', purchase_date) :: date AS purchase_month,
        product_line ,
        count(invoice_id) AS puchase_count
FROM supermarket_sales ss
GROUP BY purchase_month, product_line
ORDER BY purchase_month, count(invoice_id)
```

SELECT date_trunc('month', purchase_date) :: date AS purchase_month, product_line , count(

Enter a SQL expression to filter results (use

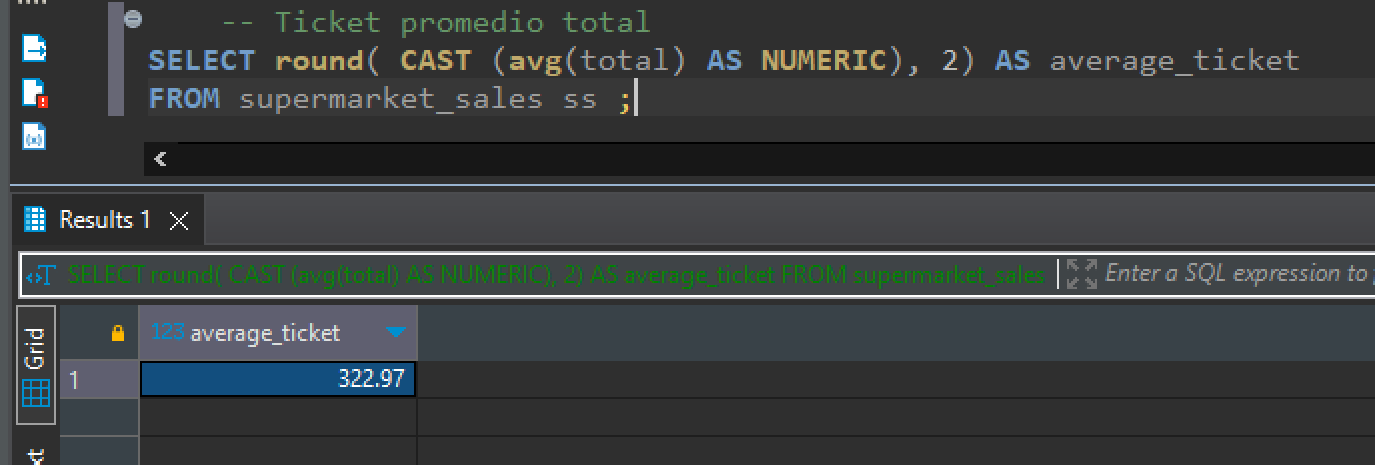
	purchase_month	product_line	puchase_count
1	2019-01-01	Health and beauty	49
2	2019-01-01	Electronic accessories	54
3	2019-01-01	Food and beverages	56
4	2019-01-01	Home and lifestyle	59
5	2019-01-01	Fashion accessories	64
6	2019-01-01	Sports and travel	70
7	2019-02-01	Home and lifestyle	38
8	2019-02-01	Sports and travel	43
9	2019-02-01	Health and beauty	46
10	2019-02-01	Electronic accessories	54
11	2019-02-01	Fashion accessories	60
12	2019-02-01	Food and beverages	62
13	2019-03-01	Sports and travel	53
14	2019-03-01	Fashion accessories	54
15	2019-03-01	Food and beverages	56
16	2019-03-01	Health and beauty	57
17	2019-03-01	Electronic accessories	62
18	2019-03-01	Home and lifestyle	63



```sql
-- KPIs:
    -- Venta total
SELECT  round( CAST (sum(total) AS NUMERIC), 2) AS total_sales
FROM supermarket_sales ss ;
```

```
SELECT round( CAST (sum(total) AS NUMERIC), 2) AS total_sales FROM supermarket_sales ss
```

```
     123 total_sales
1         322,966.75
```

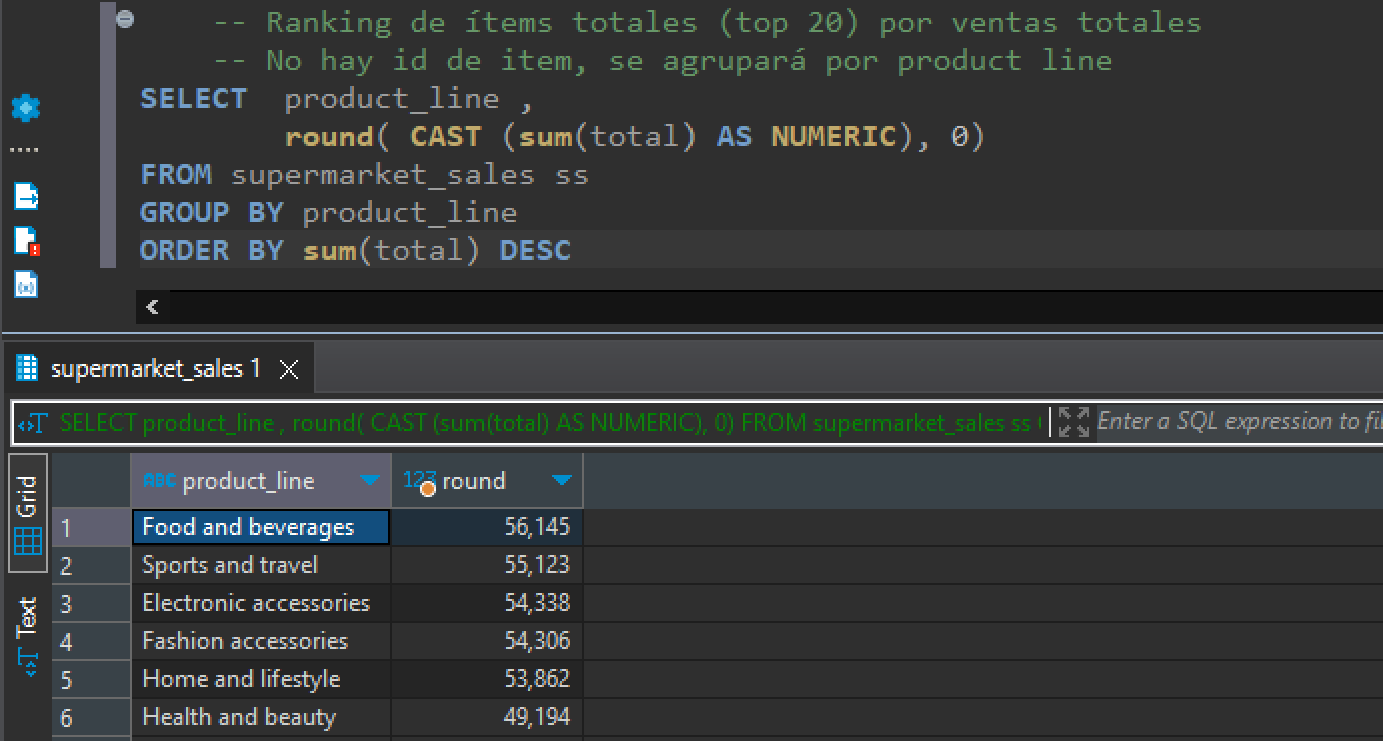


```
    -- Ticket promedio total
SELECT round( CAST (avg(total) AS NUMERIC), 2) AS average_ticket
FROM supermarket_sales ss ;
```

SELECT round(CAST (avg(total) AS NUMERIC), 2) AS average_ticket FROM supermarket_sales

Enter a SQL expression to

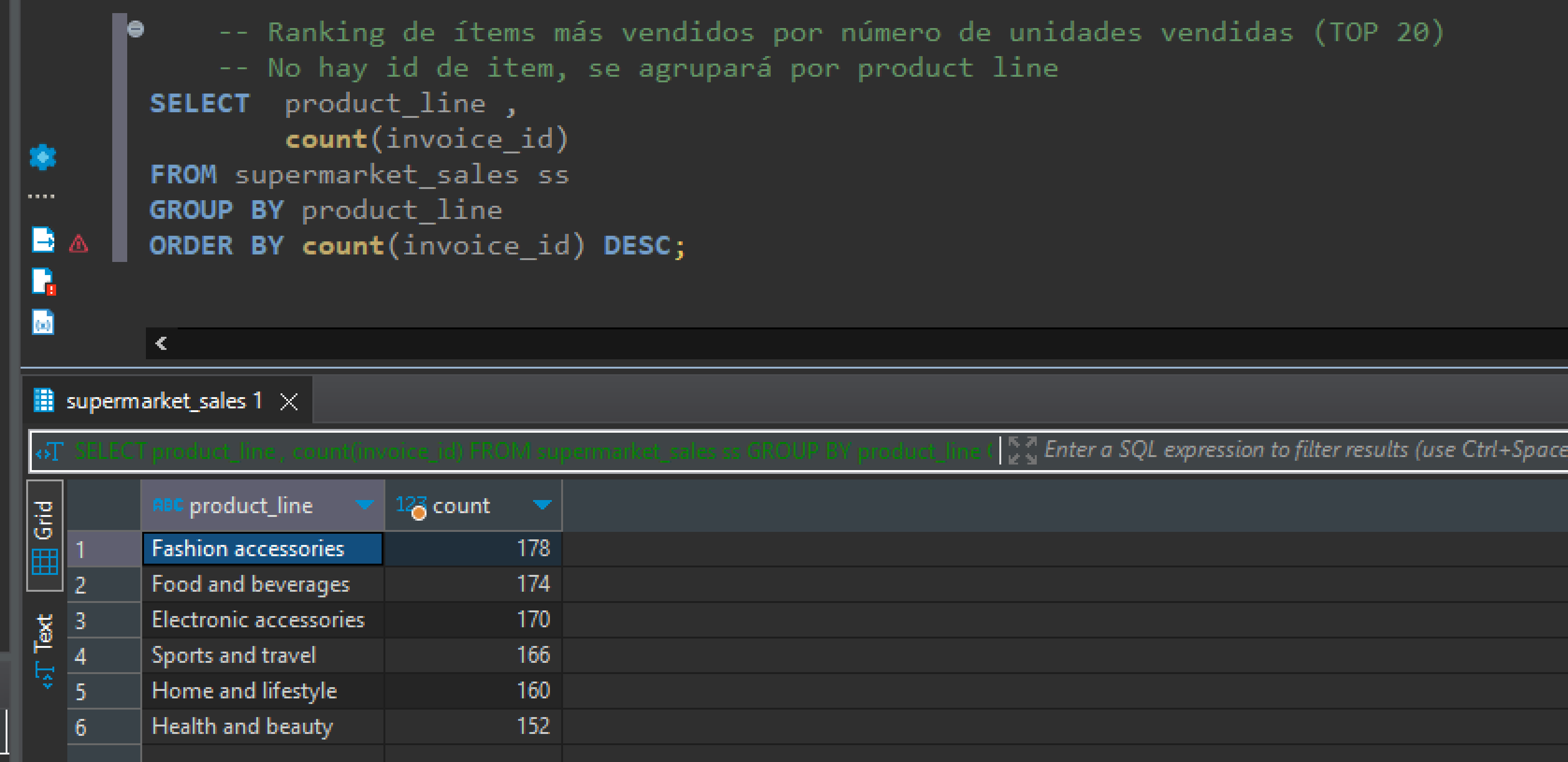
	123 average_ticket
1	322.97



```
    -- Ranking de ítems totales (top 20) por ventas totales
    -- No hay id de item, se agrupará por product line
SELECT  product_line ,
        round( CAST (sum(total) AS NUMERIC), 0)
FROM supermarket_sales ss
GROUP BY product_line
ORDER BY sum(total) DESC
```

```
SELECT product_line , round( CAST (sum(total) AS NUMERIC), 0) FROM supermarket_sales ss
```

	product_line	round
1	Food and beverages	56,145
2	Sports and travel	55,123
3	Electronic accessories	54,338
4	Fashion accessories	54,306
5	Home and lifestyle	53,862
6	Health and beauty	49,194



```sql
    -- Ranking de ítems más vendidos por número de unidades vendidas (TOP 20)
    -- No hay id de item, se agrupará por product line
SELECT  product_line ,
        count(invoice_id)
FROM supermarket_sales ss
GROUP BY product_line
ORDER BY count(invoice_id) DESC;
```

SELECT product_line , count(invoice_id) FROM supermarket_sales ss GROUP BY product_line (

Enter a SQL expression to filter results (use Ctrl+Space

	product_line	count
1	Fashion accessories	178
2	Food and beverages	174
3	Electronic accessories	170
4	Sports and travel	166
5	Home and lifestyle	160
6	Health and beauty	152