Exploring Menu Trends Through SQL Data Analysis

Norman Shatto

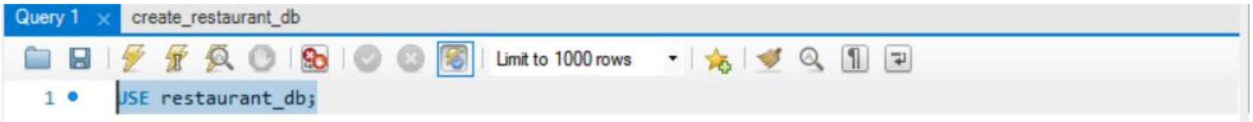
Table of Contents

Goal

In this project, I took on the role of a data analyst working with a restaurant to uncover actionable insights from its sales data. Using a MySQL database, I analyzed order patterns, transaction records, and customer behavior through a series of SQL queries. The analysis revealed key trends such as peak ordering times, the most popular menu items, and shifts in customer preferences. This provided the restaurant with valuable information to optimize operations and enhance customer satisfaction.

Objective 1: Explore the menu_items table.

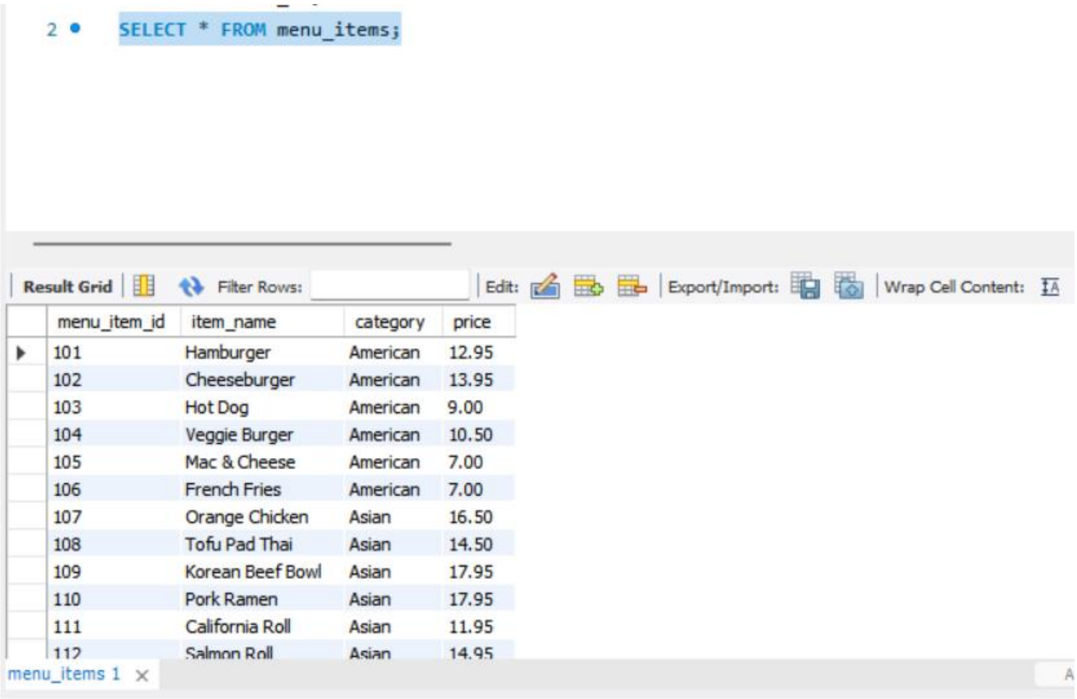
1. Opened a new query to use restaurant_db.



```
USE restaurant_db;
```

2. View the menu_items table.

"SELECT * FROM menu_items;" to show all of the menu items.

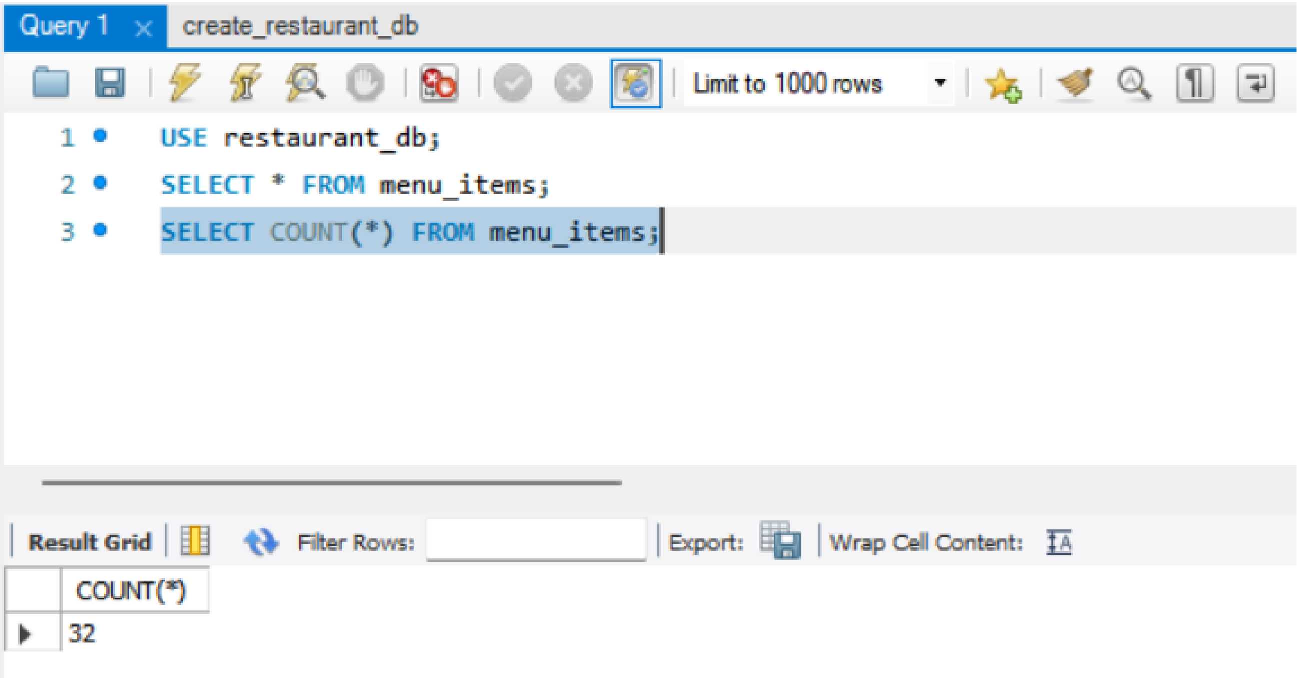


```
SELECT * FROM menu_items;
```

menu_item_id	item_name	category	price
101	Hamburger	American	12.95
102	Cheeseburger	American	13.95
103	Hot Dog	American	9.00
104	Veggie Burger	American	10.50
105	Mac & Cheese	American	7.00
106	French Fries	American	7.00
107	Orange Chicken	Asian	16.50
108	Tofu Pad Thai	Asian	14.50
109	Korean Beef Bowl	Asian	17.95
110	Pork Ramen	Asian	17.95
111	California Roll	Asian	11.95
112	Salmon Roll	Asian	14.95

3. Find the number of items on the menu.

   Using "***SELECT COUNT(*) FROM menu_items;***" to view the number of items on the menu.

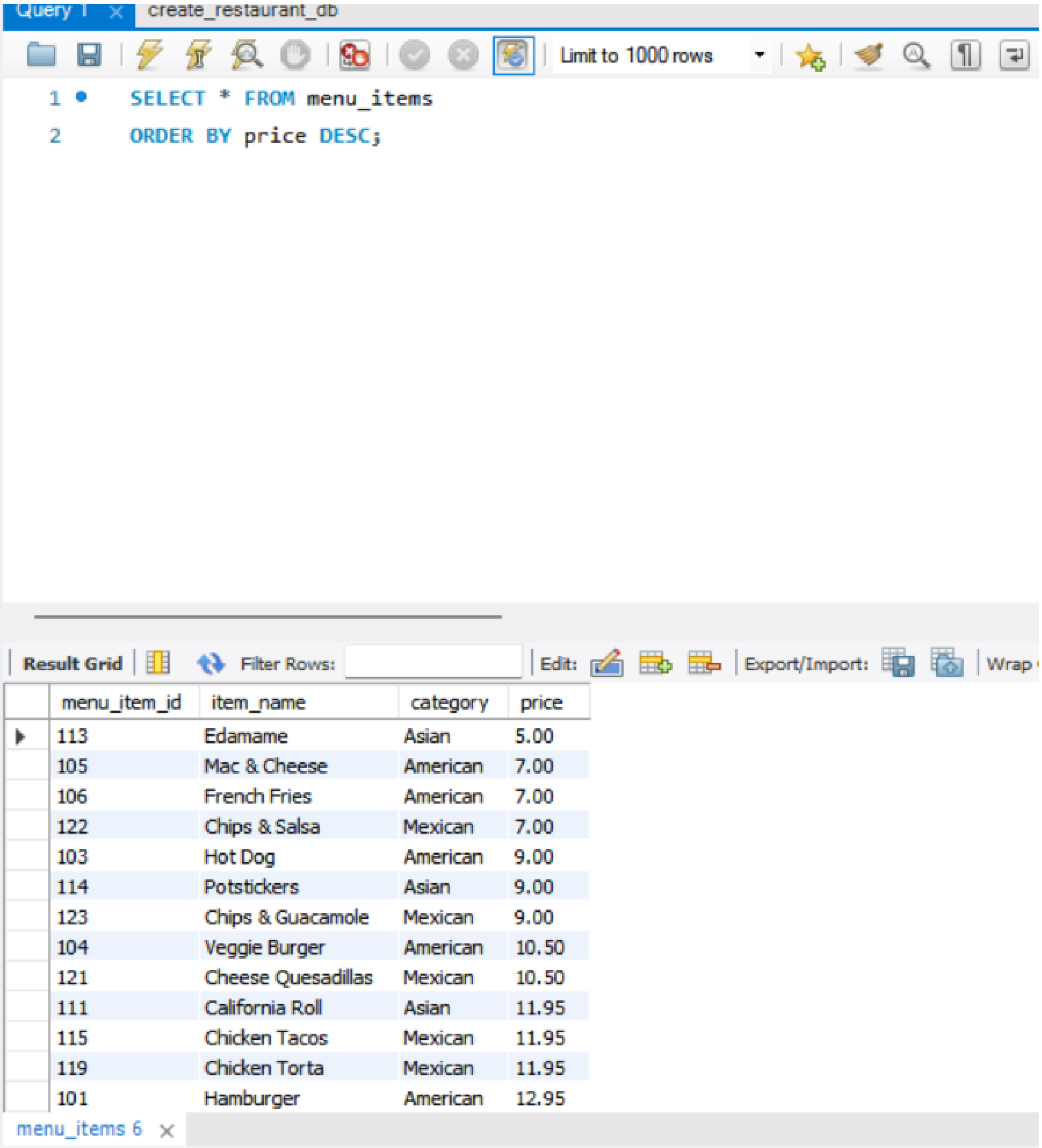


```
USE restaurant_db;
SELECT * FROM menu_items;
SELECT COUNT(*) FROM menu_items;
```

COUNT(*)
32

4. What are the least and most expensive items on the menu?

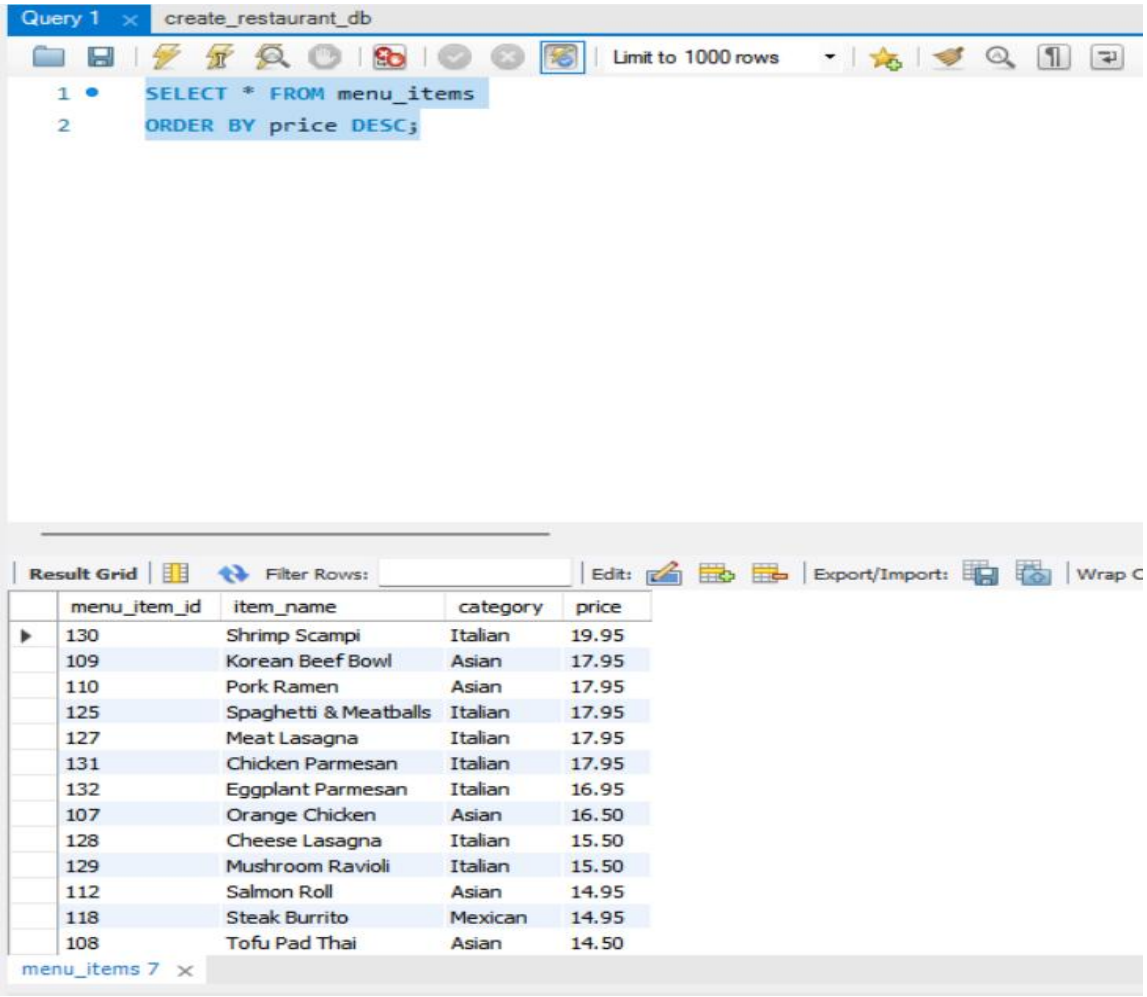
Using "***SELECT * FROM menu_items ORDER BY price;***" I sorted the table from least to most expensive. The results showed that edamame was the least expensive at $5.



```
SELECT * FROM menu_items
ORDER BY price DESC;
```

menu_item_id	item_name	category	price
113	Edamame	Asian	5.00
105	Mac & Cheese	American	7.00
106	French Fries	American	7.00
122	Chips & Salsa	Mexican	7.00
103	Hot Dog	American	9.00
114	Potstickers	Asian	9.00
123	Chips & Guacamole	Mexican	9.00
104	Veggie Burger	American	10.50
121	Cheese Quesadillas	Mexican	10.50
111	California Roll	Asian	11.95
115	Chicken Tacos	Mexican	11.95
119	Chicken Torta	Mexican	11.95
101	Hamburger	American	12.95

To find the most expensive I input “***SELECT * FROM menu_items ORDER BY price DESC;”*** and found that shrimp scampi is the most expensive at $19.95.

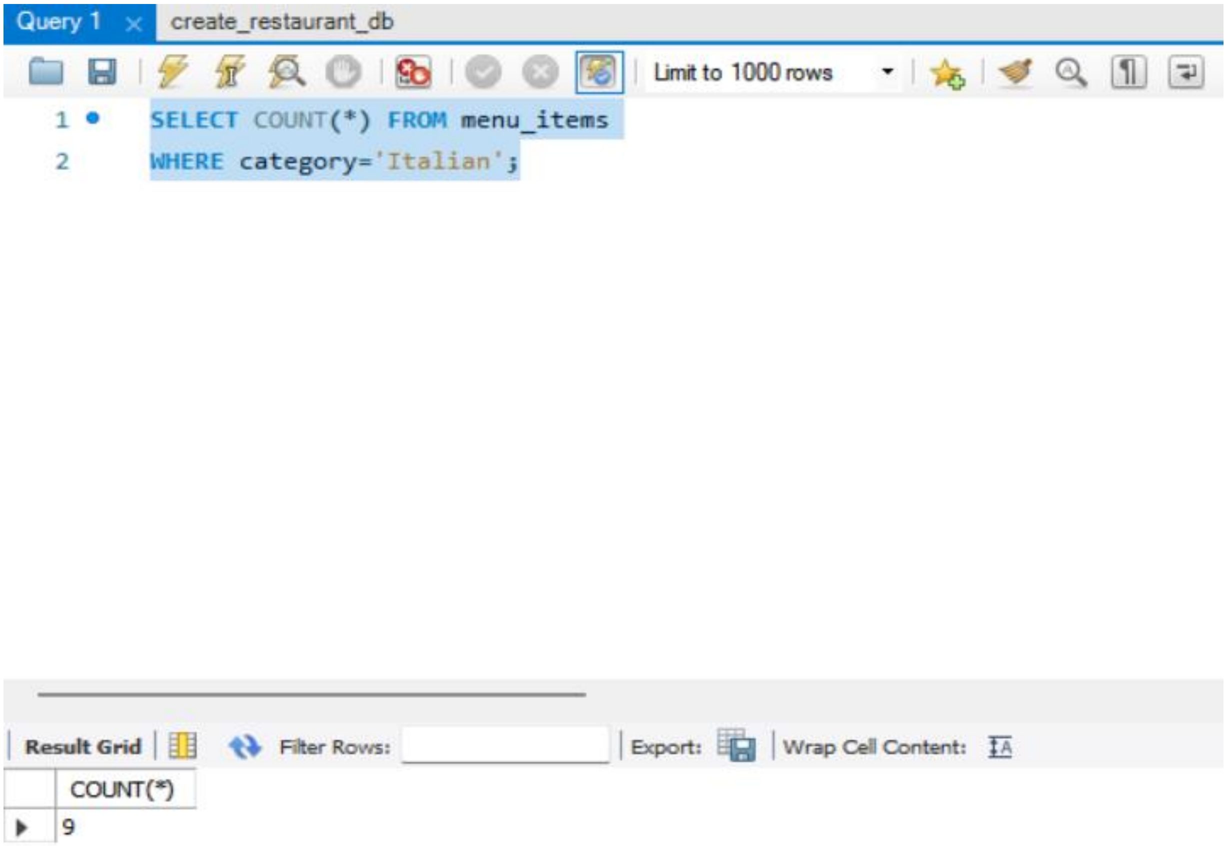


```
SELECT * FROM menu_items
ORDER BY price DESC;
```

menu_item_id	item_name	category	price
130	Shrimp Scampi	Italian	19.95
109	Korean Beef Bowl	Asian	17.95
110	Pork Ramen	Asian	17.95
125	Spaghetti & Meatballs	Italian	17.95
127	Meat Lasagna	Italian	17.95
131	Chicken Parmesan	Italian	17.95
132	Eggplant Parmesan	Italian	16.95
107	Orange Chicken	Asian	16.50
128	Cheese Lasagna	Italian	15.50
129	Mushroom Ravioli	Italian	15.50
112	Salmon Roll	Asian	14.95
118	Steak Burrito	Mexican	14.95
108	Tofu Pad Thai	Asian	14.50

5. How many Italian dishes are on the menu?

   Using ***“SELECT COUNT(*) FROM menu_items WHERE category='Italian';”*** it shows how many Italian dishes are on the menu.

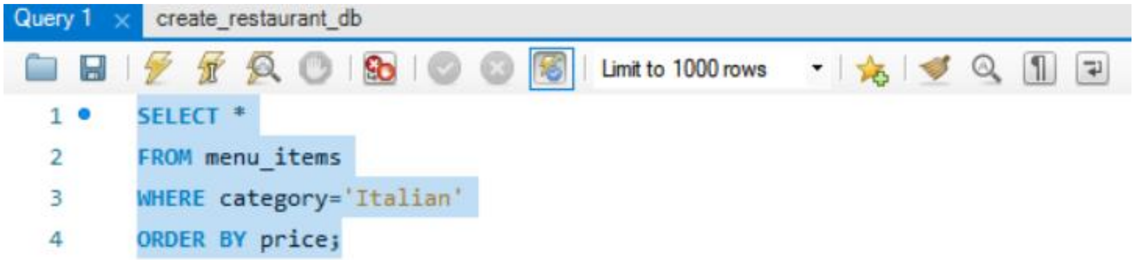


```
SELECT COUNT(*) FROM menu_items
WHERE category='Italian';
```

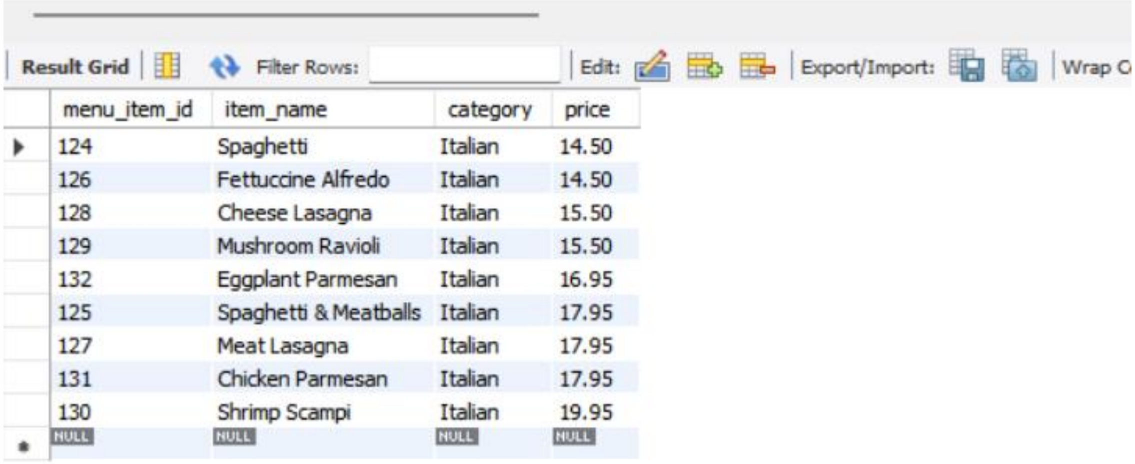
COUNT(*)
9

6. What are the most expensive Italian dishes on the menu?

   To find the most expensive Italian dish I input ***“SELECT * FROM menu_items WHERE category=’Italian’ ORDER BY price;”*** which showed spaghetti to be the most expensive at $14.50.



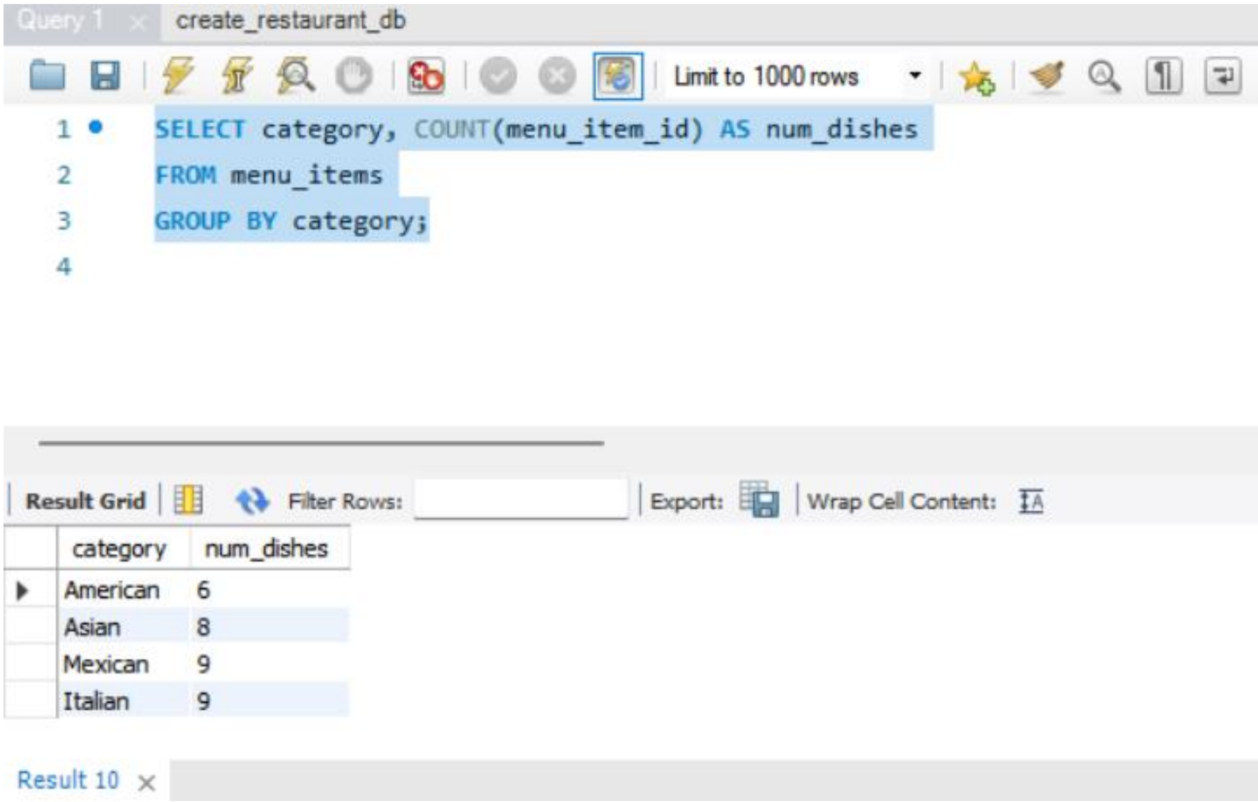
```
SELECT *
FROM menu_items
WHERE category='Italian'
ORDER BY price;
```



menu_item_id	item_name	category	price
124	Spaghetti	Italian	14.50
126	Fettuccine Alfredo	Italian	14.50
128	Cheese Lasagna	Italian	15.50
129	Mushroom Ravioli	Italian	15.50
132	Eggplant Parmesan	Italian	16.95
125	Spaghetti & Meatballs	Italian	17.95
127	Meat Lasagna	Italian	17.95
131	Chicken Parmesan	Italian	17.95
130	Shrimp Scampi	Italian	19.95
NULL	NULL	NULL	NULL

7. How many dishes are in each order?

   To find the number of dishes in each order I input ***"SELECT category, COUNT(menu_item_id) AS num_dishes FROM menu_items GROUP BY category;"***. This resulted in finding that there are 6 American, 8 Asian, 9 Mexican, and 9 Italian dishes in each order.

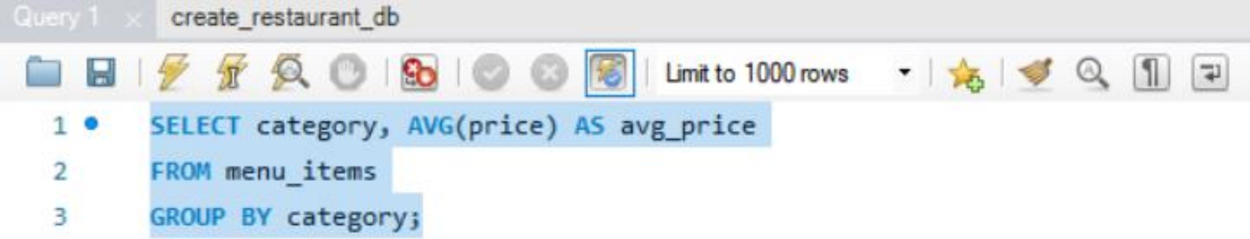


```
SELECT category, COUNT(menu_item_id) AS num_dishes
FROM menu_items
GROUP BY category;
```

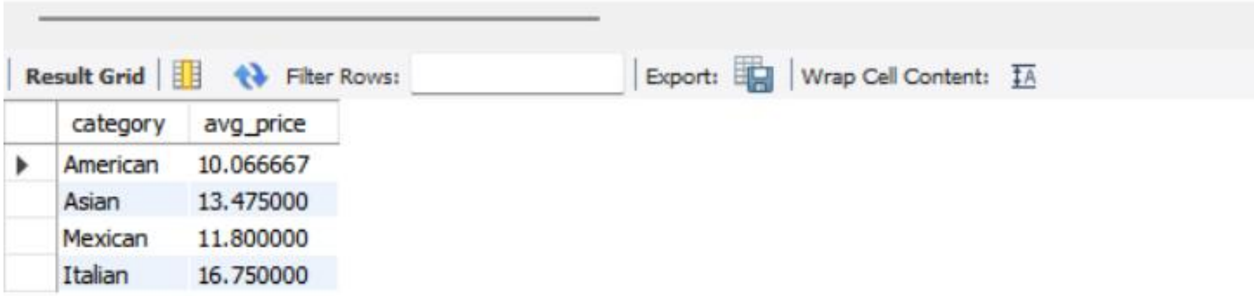
category	num_dishes
American	6
Asian	8
Mexican	9
Italian	9

8. What is the average dish price within each category?

   To find the average dish price within each category I input ***"SELECT category, AVG(price) AS avg_price FROM menu_items GROUP BY category;"*** This query resulted in finding that the average American dish is $10.06, Asian is $13.48, Mexican is $11.80, and Italian is $16.75.



```sql
SELECT category, AVG(price) AS avg_price
FROM menu_items
GROUP BY category;
```

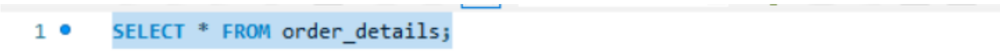


category	avg_price
American	10.066667
Asian	13.475000
Mexican	11.800000
Italian	16.750000

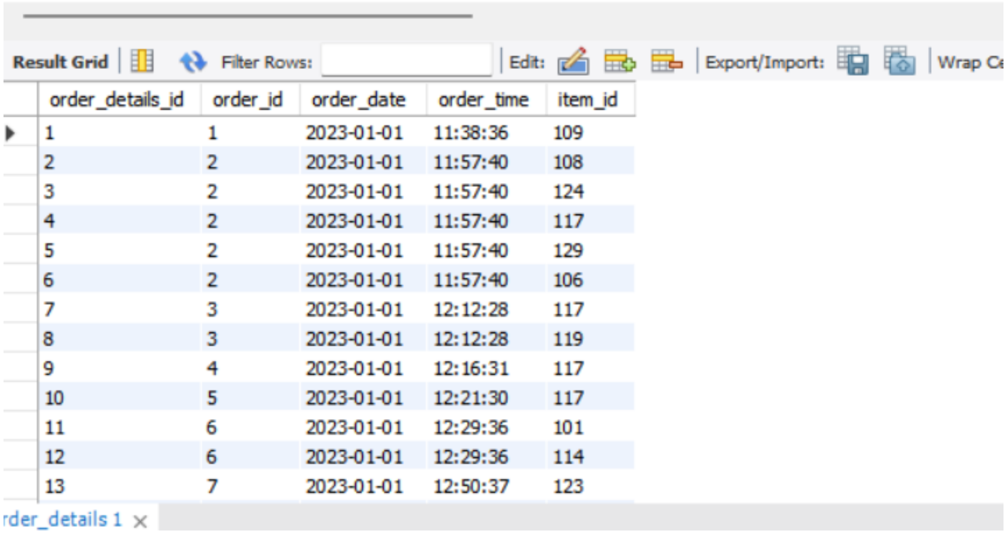
Objective 2: Explore the orders table

1. View the orders_details table.

   Query ***"SELECT * FROM order_details;"*** to show all of the order details.



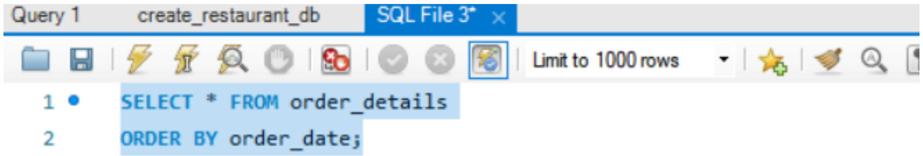
```sql
SELECT * FROM order_details;
```



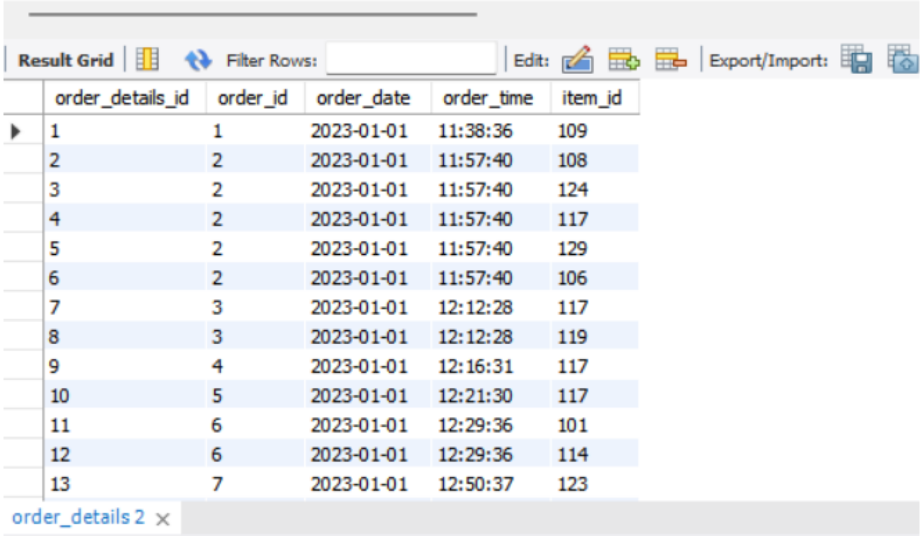
order_details_id	order_id	order_date	order_time	item_id
1	1	2023-01-01	11:38:36	109
2	2	2023-01-01	11:57:40	108
3	2	2023-01-01	11:57:40	124
4	2	2023-01-01	11:57:40	117
5	2	2023-01-01	11:57:40	129
6	2	2023-01-01	11:57:40	106
7	3	2023-01-01	12:12:28	117
8	3	2023-01-01	12:12:28	119
9	4	2023-01-01	12:16:31	117
10	5	2023-01-01	12:21:30	117
11	6	2023-01-01	12:29:36	101
12	6	2023-01-01	12:29:36	114
13	7	2023-01-01	12:50:37	123

2. What is the date range of the table?

To find the date range of the table, I input ***"SELECT * FROM order_details ORDERY BY order_dates;"***



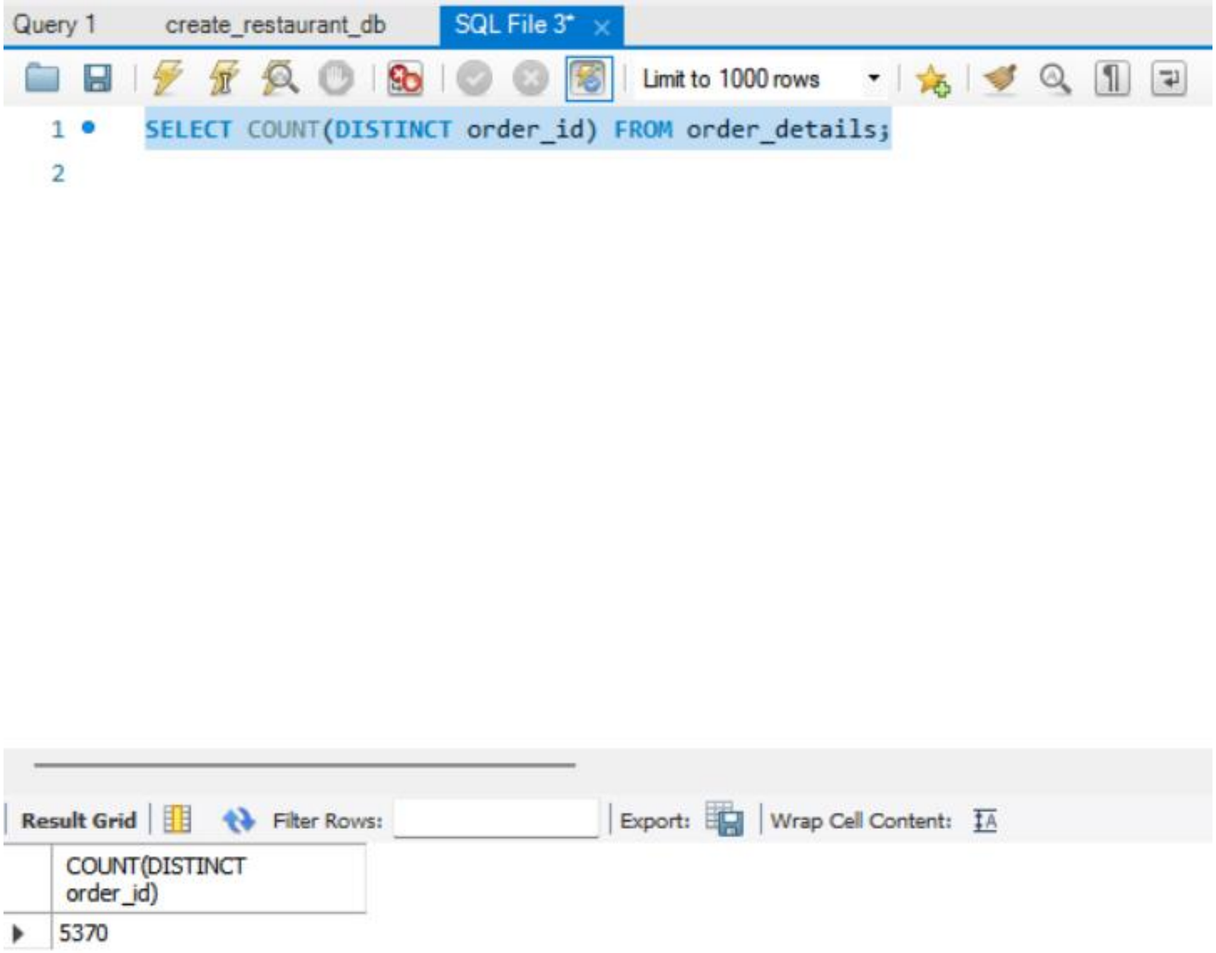
```
SELECT * FROM order_details
ORDER BY order_date;
```



order_details_id	order_id	order_date	order_time	item_id
1	1	2023-01-01	11:38:36	109
2	2	2023-01-01	11:57:40	108
3	2	2023-01-01	11:57:40	124
4	2	2023-01-01	11:57:40	117
5	2	2023-01-01	11:57:40	129
6	2	2023-01-01	11:57:40	106
7	3	2023-01-01	12:12:28	117
8	3	2023-01-01	12:12:28	119
9	4	2023-01-01	12:16:31	117
10	5	2023-01-01	12:21:30	117
11	6	2023-01-01	12:29:36	101
12	6	2023-01-01	12:29:36	114
13	7	2023-01-01	12:50:37	123

3. How many orders were made within this date range?

   To find how many orders were made I queried ***“SELECT COUNT(DISTINCT order_id) FROM order_details;”*** The query found that 5370 orders were made within this date range.

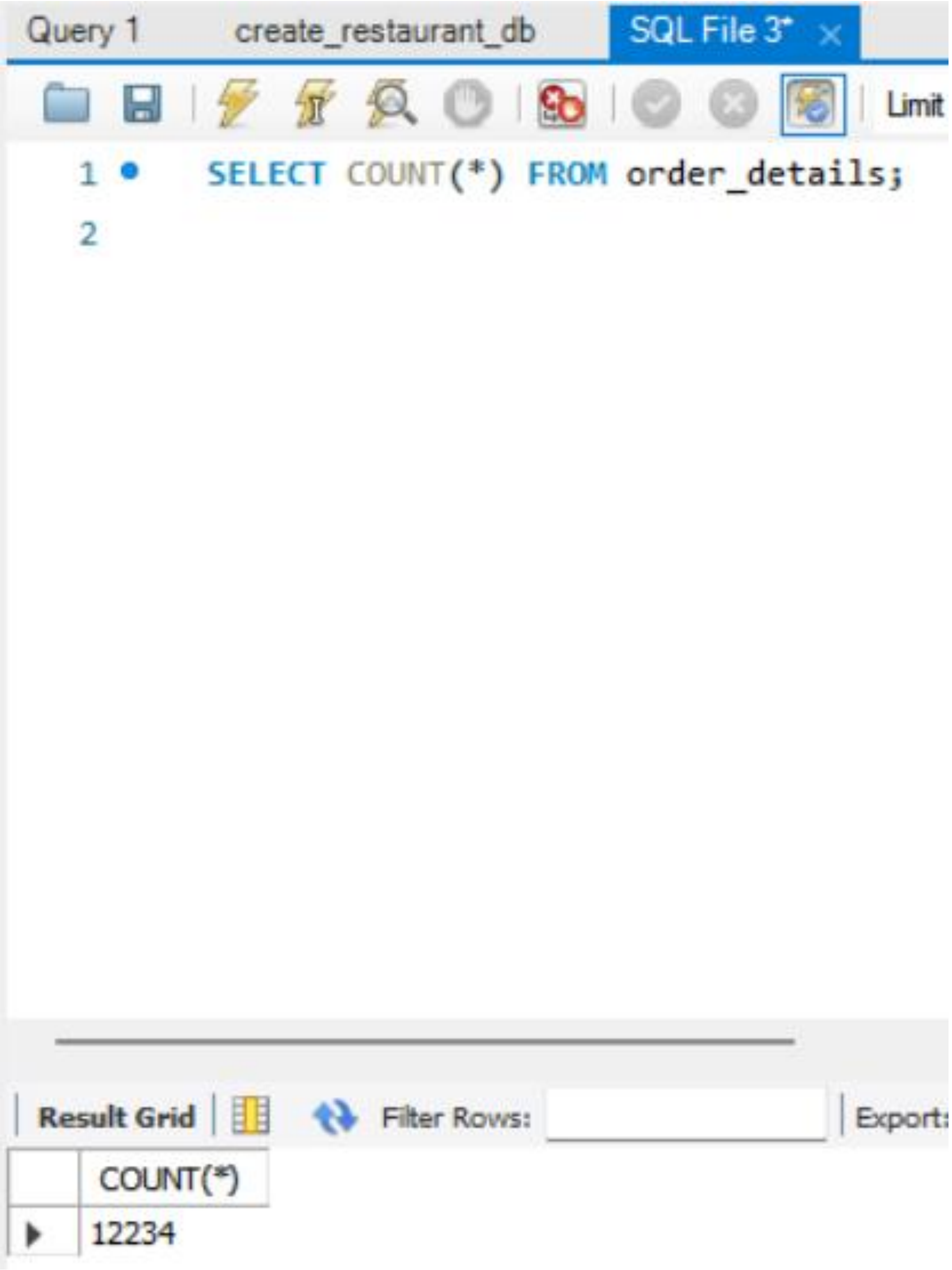


```sql
SELECT COUNT(DISTINCT order_id) FROM order_details;
```

```
COUNT(DISTINCT
order_id)
5370
```

4. How many orders were made within this date range?

   To find how many order I input ***"SELECT COUNT (*) FROM order_details;"*** which queried 12,234 ordered being made within the date range.

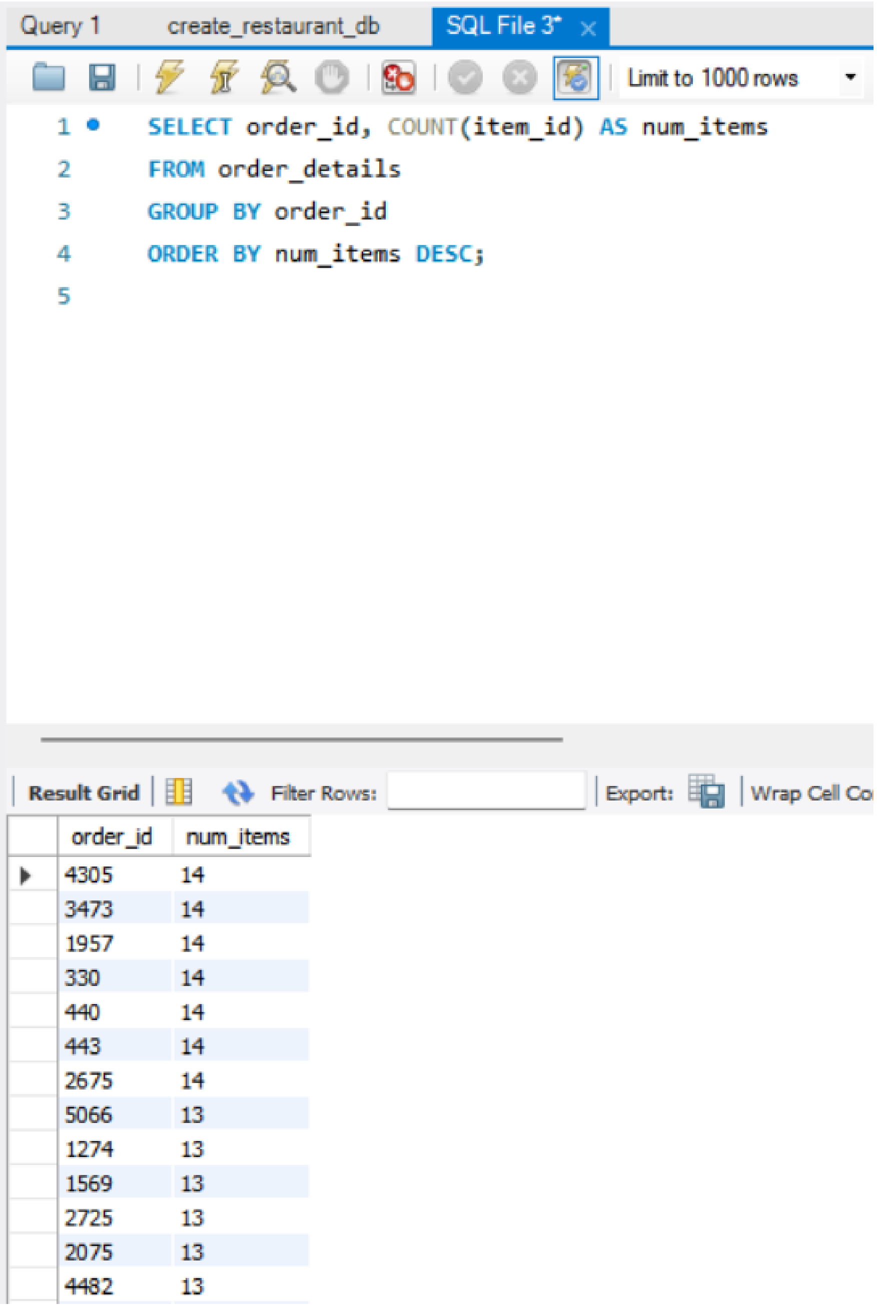


```
SELECT COUNT(*) FROM order_details;
```

```
COUNT(*)
12234
```

5. Which order had the most number of items?

   To find the most number of items I input ***"SELECT order_id, COUNT(item_id) AS num_items FROM order_details GROUP BY order_id ORDER BY num_items DESC;"*** This query resulted in finding that order_id 4305, 3473, 1957, 330, 440, and 443 had ordered 14 items.

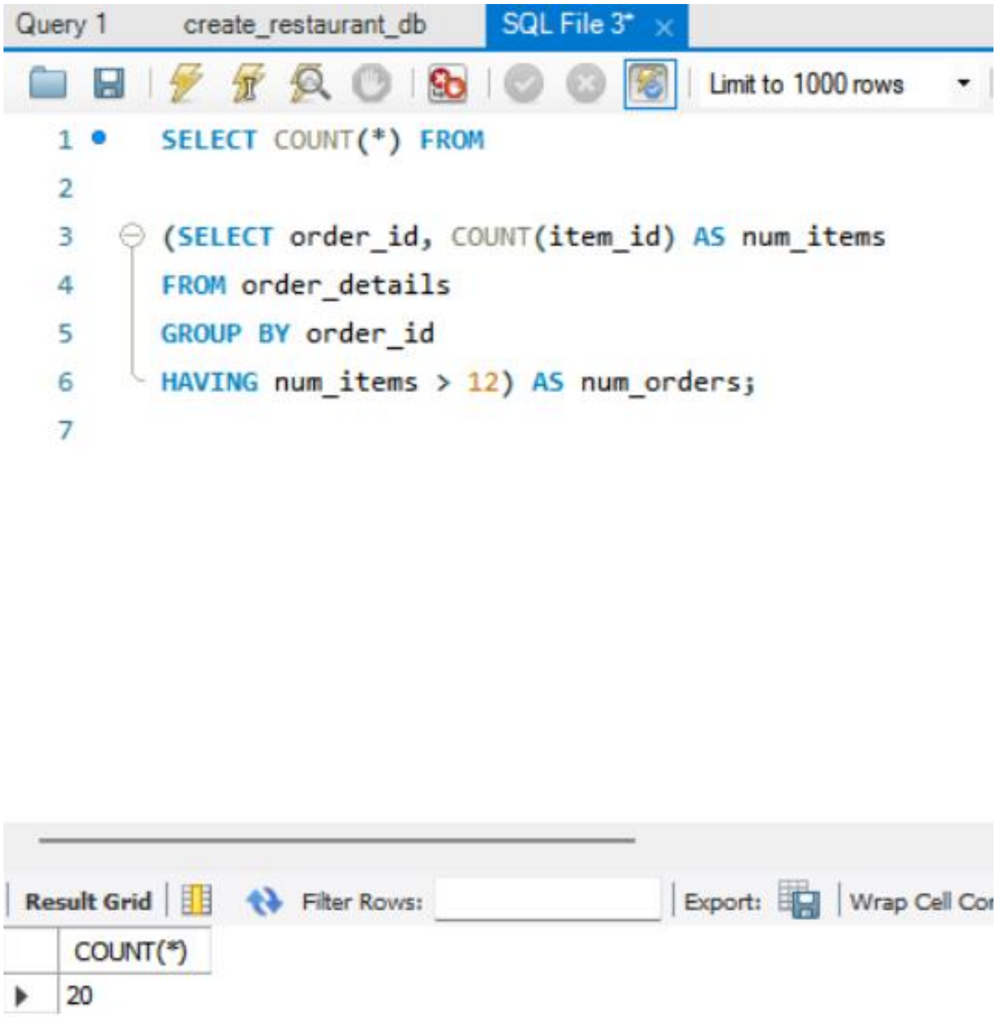


```
SELECT order_id, COUNT(item_id) AS num_items
FROM order_details
GROUP BY order_id
ORDER BY num_items DESC;
```

order_id	num_items
4305	14
3473	14
1957	14
330	14
440	14
443	14
2675	14
5066	13
1274	13
1569	13
2725	13
2075	13
4482	13

6. How many orders had more than twelve items?

   To start to find the how many order had more than twelve items I first input ***“(SELECT order_id, COUNT(item_id) AS num_items FROM order_details GROUP BY order_id HAVING num_items > 12) AS num_orders;”*** which gave all of the items that had more than twelve items. To find the number of orders that had more than twelve orders I put the first input into a subquery and above that subquery input ***“SELECT COUNT(*) FROM”*** which found that there were 20 orders with more than twelve orders.



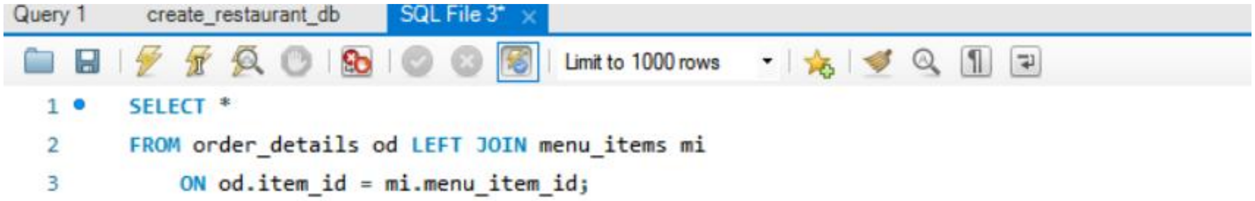
```
SELECT COUNT(*) FROM

(SELECT order_id, COUNT(item_id) AS num_items
FROM order_details
GROUP BY order_id
HAVING num_items > 12) AS num_orders;
```

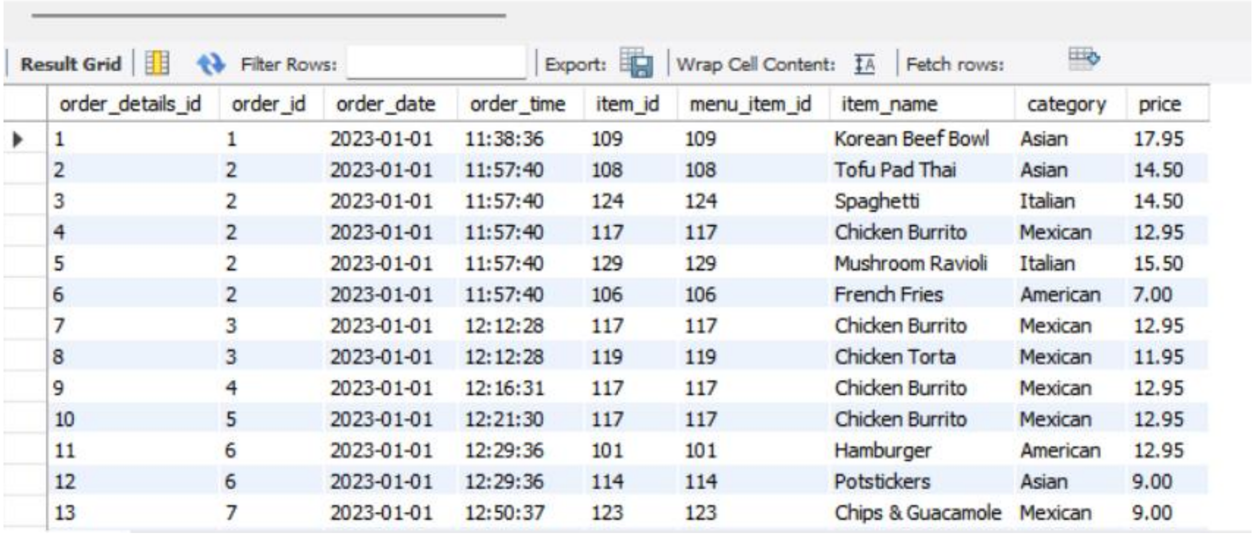
```
COUNT(*)
20
```

Objective 3: Analyze Customer Behavior

1. Combine the menu_items and order_details tables into a single table.
   To combine the menu_items and order_details tables I used a LEFT JOIN. This allowed me to combine and link each order to its corresponding menu item by using the item_id field. I input ***“SELECT * FROM order_details od LEFT JOIN menu_items mi ON od.item_id = mi.menu_item_id;”*** which combined both tables into one.



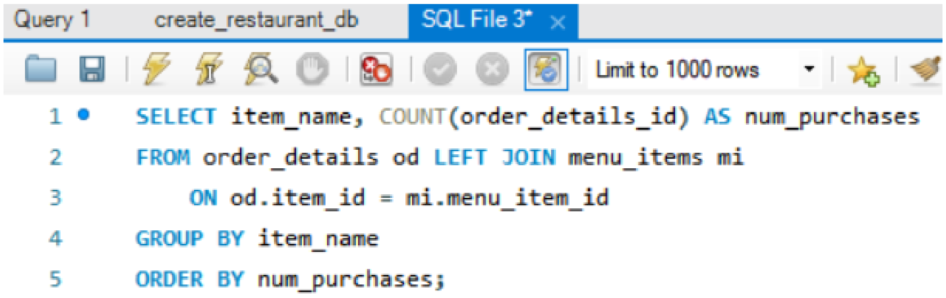
```
SELECT *
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id;
```



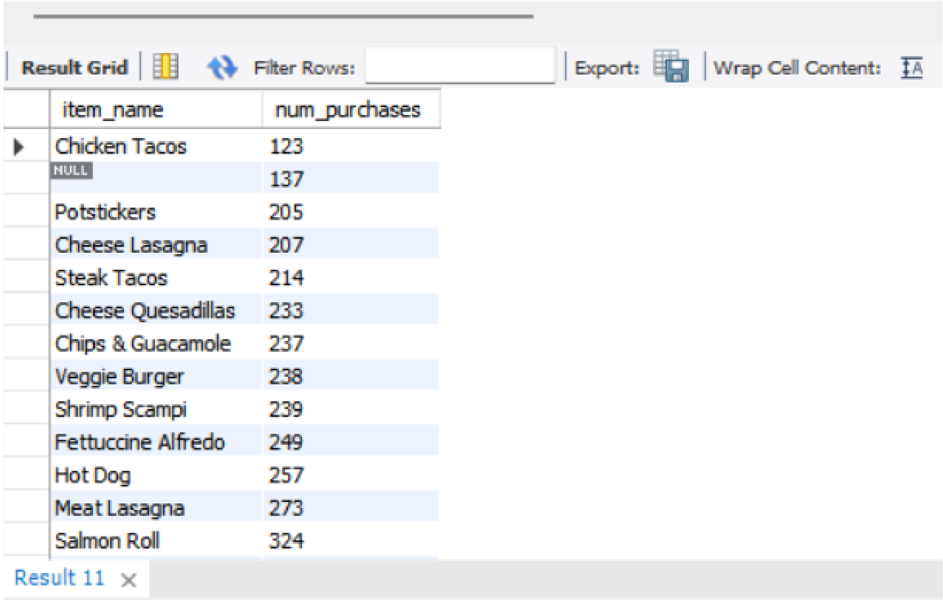
order_details_id	order_id	order_date	order_time	item_id	menu_item_id	item_name	category	price
1	1	2023-01-01	11:38:36	109	109	Korean Beef Bowl	Asian	17.95
2	2	2023-01-01	11:57:40	108	108	Tofu Pad Thai	Asian	14.50
3	2	2023-01-01	11:57:40	124	124	Spaghetti	Italian	14.50
4	2	2023-01-01	11:57:40	117	117	Chicken Burrito	Mexican	12.95
5	2	2023-01-01	11:57:40	129	129	Mushroom Ravioli	Italian	15.50
6	2	2023-01-01	11:57:40	106	106	French Fries	American	7.00
7	3	2023-01-01	12:12:28	117	117	Chicken Burrito	Mexican	12.95
8	3	2023-01-01	12:12:28	119	119	Chicken Torta	Mexican	11.95
9	4	2023-01-01	12:16:31	117	117	Chicken Burrito	Mexican	12.95
10	5	2023-01-01	12:21:30	117	117	Chicken Burrito	Mexican	12.95
11	6	2023-01-01	12:29:36	101	101	Hamburger	American	12.95
12	6	2023-01-01	12:29:36	114	114	Potstickers	Asian	9.00
13	7	2023-01-01	12:50:37	123	123	Chips & Guacamole	Mexican	9.00

2. What were the least and most ordered items? What categories were they in?

   To find the least ordered items I input ***"SELECT item_name, COUNT(order_details_id) AS num_purchases FROM order_details od LEFT JOIN menu_items mi ON od.item_id = mi.menu_item_id GROUP BY item_name ORDER BY num_purchases;"*** which showed chicken tacos were purchased 123 times.

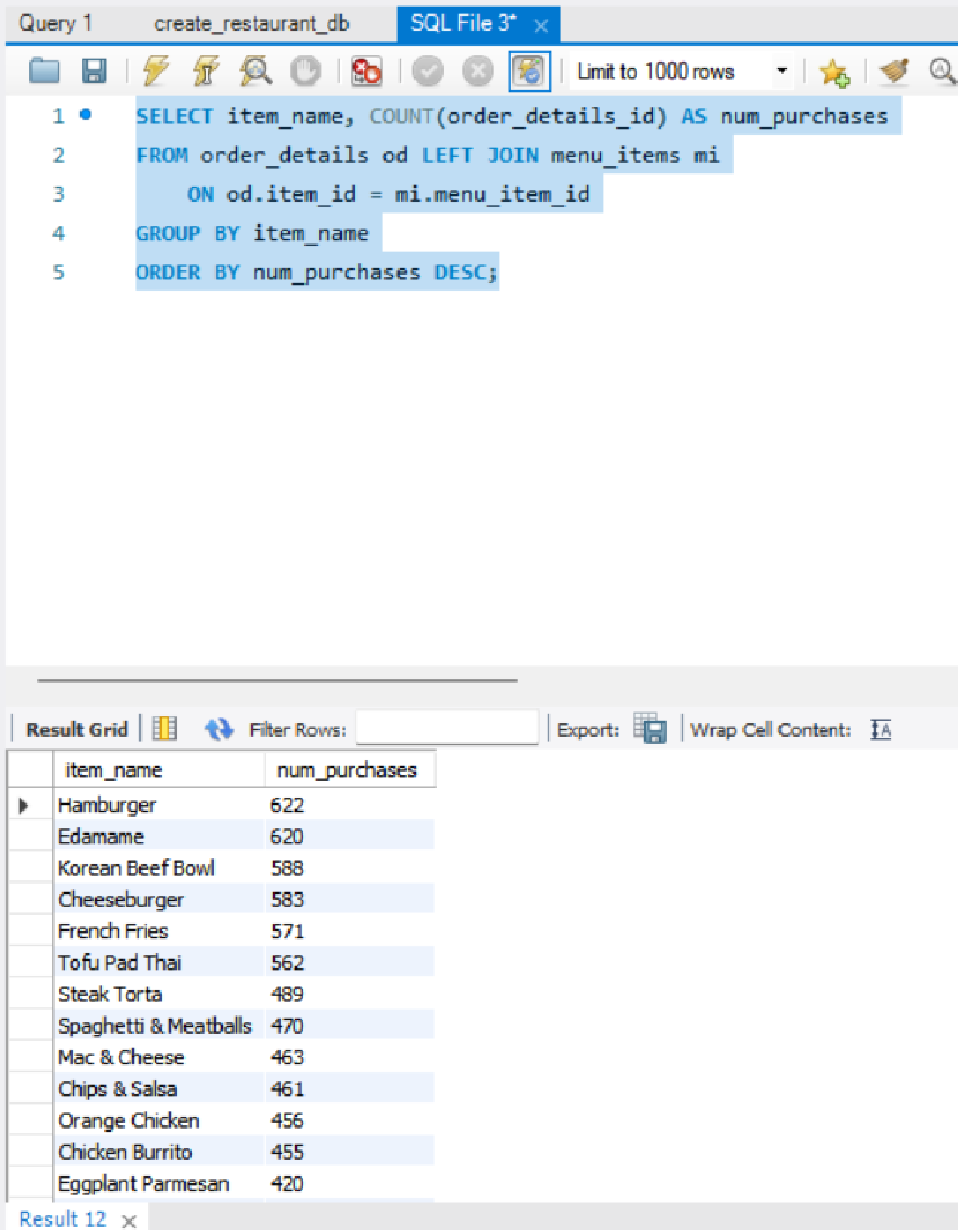


```
SELECT item_name, COUNT(order_details_id) AS num_purchases
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id
GROUP BY item_name
ORDER BY num_purchases;
```



item_name	num_purchases
Chicken Tacos	123
NULL	137
Potstickers	205
Cheese Lasagna	207
Steak Tacos	214
Cheese Quesadillas	233
Chips & Guacamole	237
Veggie Burger	238
Shrimp Scampi	239
Fettuccine Alfredo	249
Hot Dog	257
Meat Lasagna	273
Salmon Roll	324

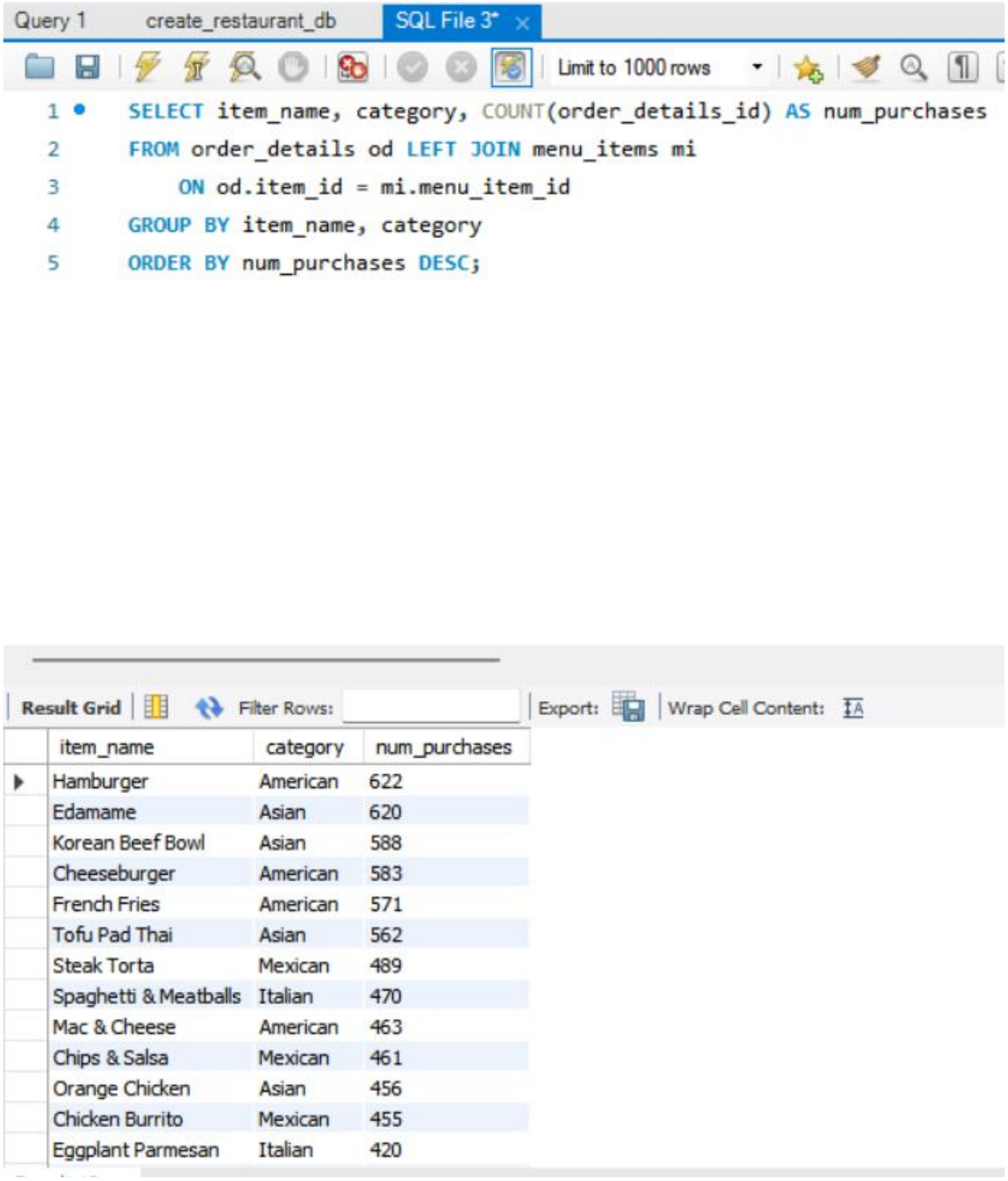
To find the most ordered items I used the same query but ordered it by DESC. This showed that hamburgers were the most purchased items with 622 purchases.



```sql
SELECT item_name, COUNT(order_details_id) AS num_purchases
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id
GROUP BY item_name
ORDER BY num_purchases DESC;
```

item_name	num_purchases
Hamburger	622
Edamame	620
Korean Beef Bowl	588
Cheeseburger	583
French Fries	571
Tofu Pad Thai	562
Steak Torta	489
Spaghetti & Meatballs	470
Mac & Cheese	463
Chips & Salsa	461
Orange Chicken	456
Chicken Burrito	455
Eggplant Parmesan	420

To find the category of food for chicken tacos and hamburgers I added "category" into the SELECT statement and in the GROUP BY statement. This showed that hamburgers were American and chicken tacos were Mexican dishes.

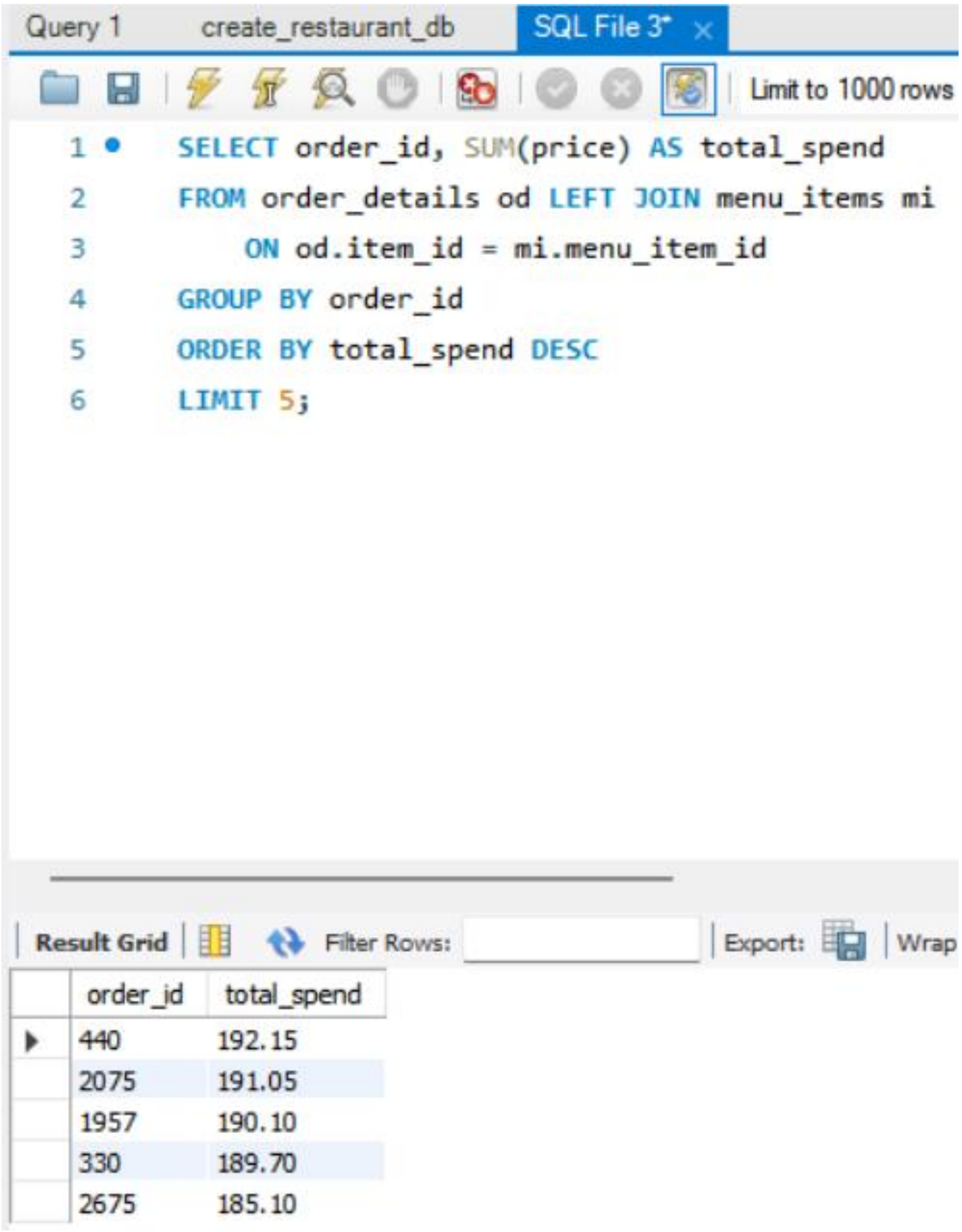


```sql
SELECT item_name, category, COUNT(order_details_id) AS num_purchases
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id
GROUP BY item_name, category
ORDER BY num_purchases DESC;
```

item_name	category	num_purchases
Hamburger	American	622
Edamame	Asian	620
Korean Beef Bowl	Asian	588
Cheeseburger	American	583
French Fries	American	571
Tofu Pad Thai	Asian	562
Steak Torta	Mexican	489
Spaghetti & Meatballs	Italian	470
Mac & Cheese	American	463
Chips & Salsa	Mexican	461
Orange Chicken	Asian	456
Chicken Burrito	Mexican	455
Eggplant Parmesan	Italian	420

3. What were the top 5 orders that spent the most money?

   To find the top 5 orders I input "***SELECT order_id, SUM(price) AS total_spend FROM order_details od LEFT JOIN menu_items mi ON od.item_id = mi.menu_item_id GROUP BY order_id ORDER BY total_spend DESC LIMIT 5;"*** which showed that order_id 440, 2075, 1957, 330, and 2675 were the top 5 orders.

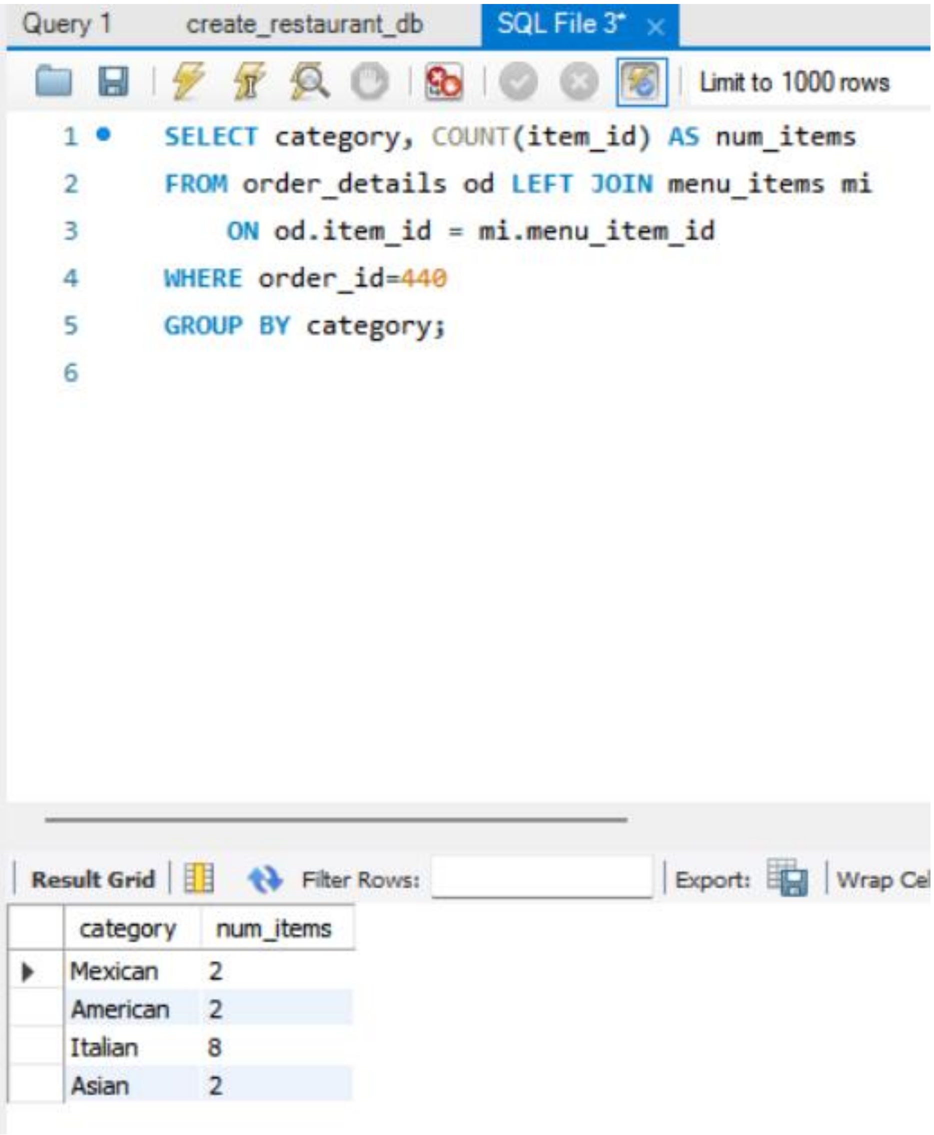


```sql
SELECT order_id, SUM(price) AS total_spend
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id
GROUP BY order_id
ORDER BY total_spend DESC
LIMIT 5;
```

order_id	total_spend
440	192.15
2075	191.05
1957	190.10
330	189.70
2675	185.10

4. View the details of the highest spent order. What insights can you gather from this?

   In the previous query I identified that order ID 440 spent the most money. This query showed that the customer ordered items from four different categories, Italian (8 items), Mexican (2 items), American (2 items), and Asian (2 items). This suggests that order 440 was a large, mixed order, possibly for a group rather than an individual.

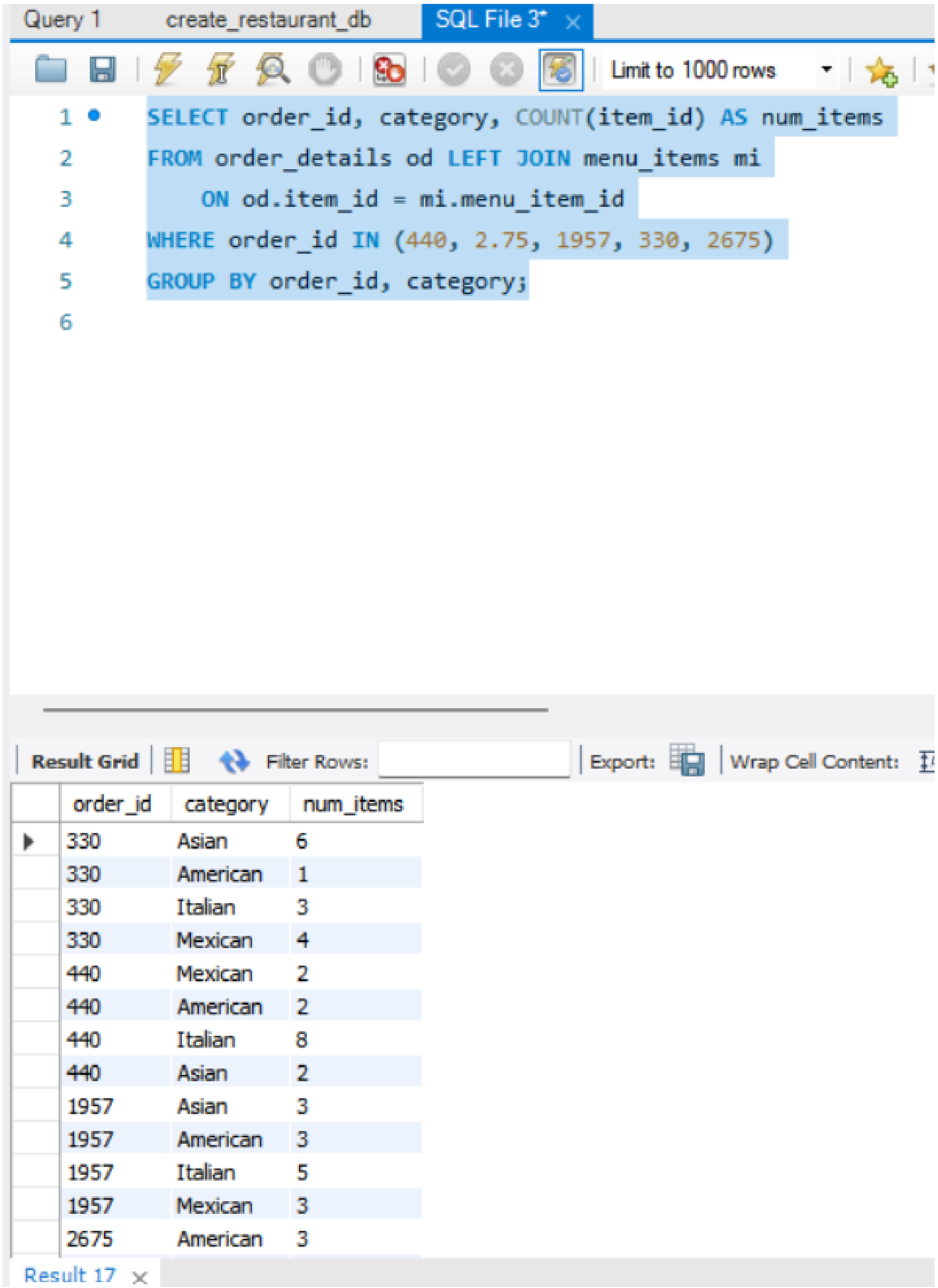


```
SELECT category, COUNT(item_id) AS num_items
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id
WHERE order_id=440
GROUP BY category;
```

category	num_items
Mexican	2
American	2
Italian	8
Asian	2

5. View the details of the top 5 highest paid orders. What insights can you gather from this?

   The data shows that the large orders included a mix of Asian, Italian, Mexican, and American but Italian being the most frequent. With these results I can suggest to keep the high priced Italian dishes on the menu since they are the most popular.
   To find the details of the top 5 highest paid orders,



```
SELECT order_id, category, COUNT(item_id) AS num_items
FROM order_details od LEFT JOIN menu_items mi
    ON od.item_id = mi.menu_item_id
WHERE order_id IN (440, 2.75, 1957, 330, 2675)
GROUP BY order_id, category;
```

order_id	category	num_items
330	Asian	6
330	American	1
330	Italian	3
330	Mexican	4
440	Mexican	2
440	American	2
440	Italian	8
440	Asian	2
1957	Asian	3
1957	American	3
1957	Italian	5
1957	Mexican	3
2675	American	3

Findings (Conclusion)

In summary, this project shows how the use of SQL to combine transactional (order) data with detailed information about menu items can provide the restaurant with valuable business insight. The project demonstrated how to join two tables: the order_details table and the menu_items table, apply different types of aggregation and filters, identify key orders by revenue, determine which cuisine categories represented the majority of these high-revenue orders, and illustrate trends in how customers make purchases. One of the most interesting findings was that all group or mixed-cuisine orders had a tendency to have the highest transactions. In many cases, Italian items were among the top-selling items on the menu. The findings also suggest high demand menu item categories should be prioritized and each menu item's performance should be monitored to maximize sales.