

“CUSTOMER BEHAVIOR ANALYSIS AND INSIGHTS GENERATION USING SQL”

This project is composed of the following:

Introduction: Provides an overview of the project's objectives and scope.

Objective 01: I conducted a thorough analysis using seven queries to gain a deeper understanding of the “menu_items” table, including its structure, and data distribution.

Objective 02: I performed examination of the "order_details" table by executing a series of six queries to delve into the table's content and structure.

Objective 03: I delved into the world of customer behavior, examining the patterns and trends that shape their interactions with our business. To achieve this, I executed approximately five queries that provided valuable insights

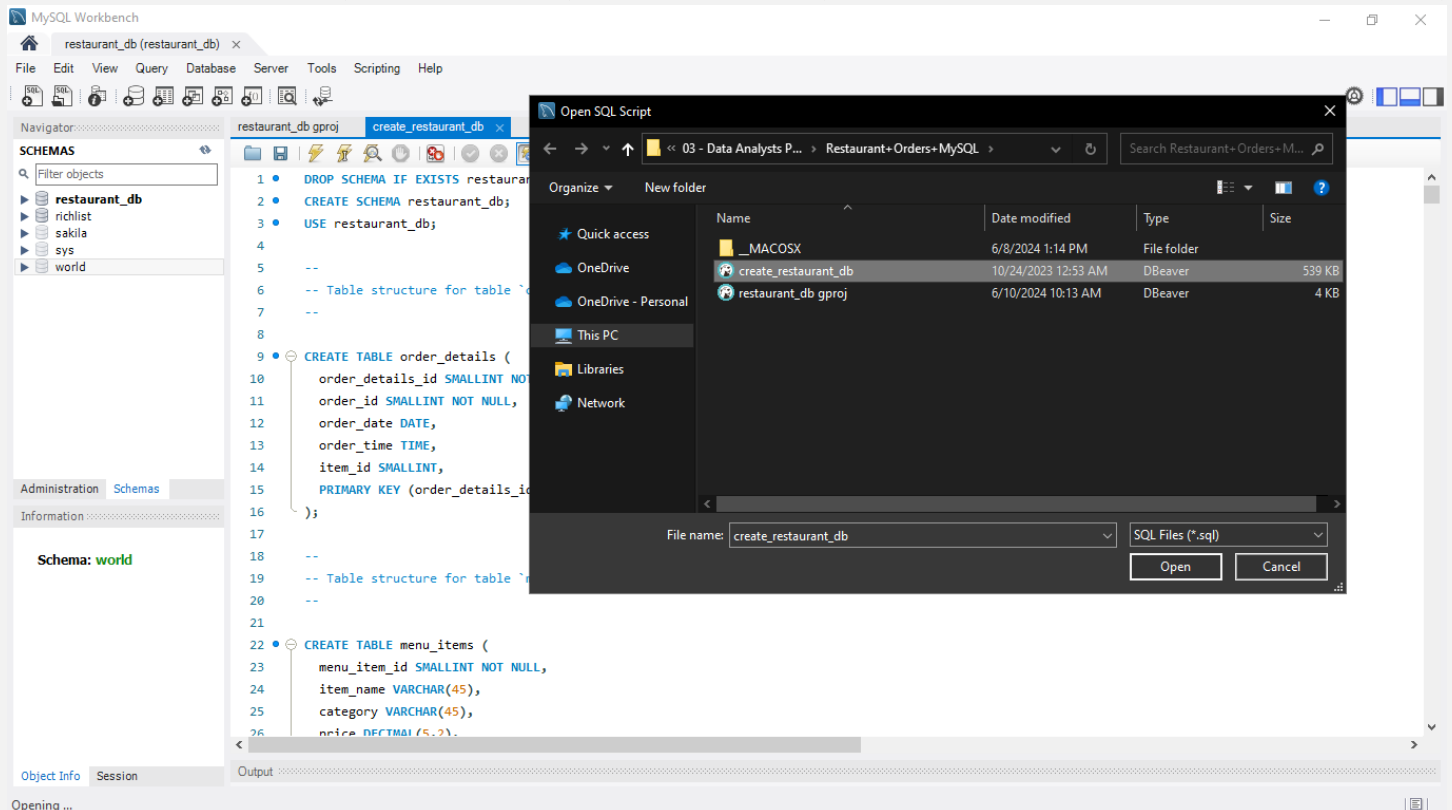
Summary of Findings: I organized a concise overview of the main results derived from SQL queries.

Key Insights: The critical observations or revelations derived from the summary of findings. These insights provide actionable information and guide decision-making.

INTRODUCTION

This project leverages SQL to extract valuable insights from a database by analyzing data stored in two tables: menu_items and order_details. To accomplish this, I utilized MySQL Workbench, a powerful software tool, to execute SQL queries and access the information stored in the database.

The menu_items table contains detailed information about the menu items, including their menu_item_id, item_name, category, and price. This table provides a comprehensive overview of the menu offerings, allowing for analysis of menu structure, pricing, and item diversity. The order_details table, on the other hand, contains records of order_details_id, order_id, order_date, order_time and item_id,



OBJECTIVE 01:

I viewed the content of the “menu_items” table to gain insights into the type of data I’ll be working with.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'restaurant_db' schema with tables 'menu_items' and 'order_details'. The main query editor contains the following SQL code:

```
1 • USE restaurant_db;
2
3 -- 1. View the menu_items table.
4 • select * from menu_items;
5
6 -- 2. Find the number of items on the menu.
7
8
9 -- 3. What are the least and most expensive items on the menu?
10
```

The 'Result Grid' at the bottom displays the contents of the 'menu_items' table:

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
113	Framame	Asian	5.00

I delved deeper by tallying the number of items on the menu.

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'restaurant_db' schema with tables 'menu_items' and 'order_details'. The main query editor contains the following SQL code:

```
1 • USE restaurant_db;
2
3
4
5 -- OBJECTIVE 1: EXPLORE THE ITEMS TABLE --
6
7 -- 1. View the menu_items table.
8 • select * from menu_items;
9
10 -- 2. Find the number of items on the menu.
11 • select count(*) from menu_items;
```

The 'Result Grid' at the bottom displays the result of the count query:

count(*)
32

I found that “Edamame” and the Asian menu items were the least expensive on the menu.

The screenshot shows MySQL Workbench with a query window containing four SQL queries. The first query counts the number of items on the menu. The second query orders items by price ascending. The third query orders items by price descending. The fourth query counts the number of Italian dishes. The result grid shows the output of the second query, listing items from least to most expensive.

```
6 -- 2. Find the number of items on the menu.
7 • select count(*) from menu_items;
8
9 -- 3. What are the least and most expensive items on the menu?
10 • select * from menu_items order by price asc;
11 • select * from menu_items order by price desc;
12
13 -- 4. How many Italian dishes are on the menu?
14
15
```

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	17.95

I determined that Shrimp Scampi from the Italian menu was the most expensive item on the menu.

The screenshot shows MySQL Workbench with the same four SQL queries as the previous image. The result grid shows the output of the third query, listing items from most to least expensive.

```
6 -- 2. Find the number of items on the menu.
7 • select count(*) from menu_items;
8
9 -- 3. What are the least and most expensive items on the menu?
10 • select * from menu_items order by price asc;
11 • select * from menu_items order by price desc;
12
13 -- 4. How many Italian dishes are on the menu?
14
15
```

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

I identified there are 9 Italian dishes out of the 32 total dishes.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure, including tables like 'menu_items' and 'order_details'. The 'Query' pane contains the following SQL code:

```
8
9 -- 3. What are the least and most expensive items on the menu?
10 • select * from menu_items order by price asc;
11 • select * from menu_items order by price desc;
12
13 -- 4. How many Italian dishes are on the menu?
14 • select count(*) from menu_items
15   where category = 'Italian';
16
17
```

The 'Result Grid' pane shows the result of the query, displaying a single row with the count of Italian dishes:

count(*)
9

The 'Table: menu_items' pane shows the table structure:

Columns:

- menu_item_id: smallint PK
- item_name: varchar(45)
- category: varchar(45)
- price: decimal(5,2)

I determined that Spaghetti tends to be the least expensive option, while Shrimp Scampi is typically the most expensive.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure, including tables like 'menu_items' and 'order_details'. The 'Query' pane contains the following SQL code:

```
13 -- 4. How many Italian dishes are on the menu?
14 • select count(*) from menu_items
15   where category = 'Italian';
16
17 -- 5. What are the least and most expensive Italian dishes on the menu?
18 • select * from menu_items
19   where category = 'Italian'
20   order by price asc;
21
22 -- 6. How many dishes are in each category?
```

The 'Result Grid' pane shows the result of the query, displaying a list of Italian dishes ordered 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

The 'Table: menu_items' pane shows the table structure:

Columns:

- menu_item_id: smallint PK
- item_name: varchar(45)
- category: varchar(45)
- price: decimal(5,2)

I looked for the number of dishes in each category.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure, including tables like 'menu_items' and 'order_details'. The 'Query' editor in the center contains the following SQL code:

```
20 order by price asc;
21
22 -- 6. How many dishes are in each category?
23 • select category, count(menu_item_id)
24 from menu_items
25 group by category
26
27
28 -- 7. What is the average dish price within each category?
29
```

The 'Result Grid' at the bottom displays the results of the query:

category	count(menu_item_id)
American	6
Asian	8
Mexican	9
Italian	9

The 'Table: menu_items' information is also visible on the left, showing columns: menu_item_id (smallint PK), item_name (varchar(45)), category (varchar(45)), and price (decimal(5,2)).

I wanted to look for the average dish price per category. I identified the least and most expensive average.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure, including tables like 'menu_items' and 'order_details'. The 'Query' editor in the center contains the following SQL code:

```
24 from menu_items
25 group by category;
26
27
28 -- 7. What is the average dish price within each category?
29 • select category, avg(price)
30 from menu_items
31 group by category;
32
33
34
```

The 'Result Grid' at the bottom displays the results of the query:

category	avg(price)
American	10.06667
Asian	13.475000
Mexican	11.800000
Italian	16.750000

The 'Table: menu_items' information is also visible on the left, showing columns: menu_item_id (smallint PK), item_name (varchar(45)), category (varchar(45)), and price (decimal(5,2)).

OBJECTIVE 02:

I viewed the content of the “order_details” table to gain insights into the type of data I’ll be working with.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'order_details' table is highlighted in the Schemas pane. The main editor displays a SQL query to view the contents of the 'order_details' table. The query is as follows:

```
-- OBJECTIVE 2: EXPLORE THE ORDERS TABLE --  
  
-- 1. View the order_details table.  
select * from order_details;  
  
-- 2. What is the date range of the table?  
  
-- 3. How many orders were made within this date range?  
  
-- 4. How many items were ordered within this date range?
```

The Result Grid shows the first 10 rows of the 'order_details' table:

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

The left pane shows the 'menu_items' table structure:

Table: menu_items

Columns:

- menu_item_id (smallint PK)
- item_name (varchar(45))
- category (varchar(45))
- price (decimal(5,2))

I searched for the specific date and time when each order was placed.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'order_details' table is highlighted in the Schemas pane. The main editor displays a SQL query to find the date range of the 'order_details' table. The query is as follows:

```
-- OBJECTIVE 2: EXPLORE THE ORDERS TABLE --  
  
-- 1. View the order_details table.  
select * from order_details;  
  
-- 2. What is the date range of the table?  
select min(order_date), max(order_date)  
from order_details;  
  
-- 3. How many orders were made within this date range?
```

The Result Grid shows the results of the query:

min(order_date)	max(order_date)
2023-01-01	2023-03-31

The left pane shows the 'menu_items' table structure:

Table: menu_items

Columns:

- menu_item_id (smallint PK)
- item_name (varchar(45))
- category (varchar(45))
- price (decimal(5,2))

A total of 5,370 orders were recorded within the specified date range.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The left sidebar displays the 'SCHEMAS' tree with 'restaurant_db' expanded, showing tables like 'menu_items' and 'order_details'. The main editor contains a SQL script with three queries. The third query is highlighted, showing the result of a count of distinct order IDs.

```
41 -- OBJECTIVE 2: EXPLORE THE ORDERS TABLE --
42
43 -- 1. View the order_details table.
44 • select * from order_details;
45
46 -- 2. What is the date range of the table?
47 • select min(order_date), max(order_date)
48   from order_details;
49
50 -- 3. How many orders were made within this date range?
51 • select count(distinct(order_id)) from order_details;
52
```

The 'Result Grid' shows the output of the third query:

count(distinct(order_id))
5370

The 'Information' panel on the left shows the structure of the 'menu_items' table:

Table: menu_items

Columns:

- menu_item_id: smallint PK
- item_name: varchar(45)
- category: varchar(45)
- price: decimal(5,2)

A total of 12,234 items were ordered within the specified date range.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The left sidebar displays the 'SCHEMAS' tree with 'restaurant_db' expanded, showing tables like 'menu_items' and 'order_details'. The main editor contains a SQL script with six queries. The fourth query is highlighted, showing the result of a count of all items ordered.

```
49
50 -- 3. How many orders were made within this date range?
51 • select count(distinct(order_id)) from order_details;
52
53 -- 4. How many items were ordered within this date range?
54 • select count(*) from order_details;
55
56 -- 5. Which orders had the most number of items?
57
58
59 -- 6. How many orders had more than 12 items?
```

The 'Result Grid' shows the output of the fourth query:

count(*)
12234

The 'Information' panel on the left shows the structure of the 'menu_items' table:

Table: menu_items

Columns:

- menu_item_id: smallint PK
- item_name: varchar(45)
- category: varchar(45)
- price: decimal(5,2)

Seven order IDs have the highest number of items, with each order containing 14 items.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure, including tables like 'menu_items' and 'order_details'. The 'Query' editor in the center contains the following SQL code:

```
-- 4. How many items were ordered within this date range?
54 • select count(*) from order_details;
55
-- 5. Which orders had the most number of items?
57 • select order_id, count(item_id) as num_items
58   from order_details
59  group by order_id
60  order by num_items desc;
61
-- 6. How many orders had more than 12 items?
```

The 'Result Grid' at the bottom displays the results of the query, showing the top 10 orders with the highest number of items. The columns are 'order_id' and 'num_items'.

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

By reversing the order of the results using the 'asc' keyword instead of 'desc', I discovered that there are orders with a quantity of zero items.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Query' editor in the center contains the following SQL code:

```
-- 4. How many items were ordered within this date range?
54 • select count(*) from order_details;
55
-- 5. Which orders had the most number of items?
57 • select order_id, count(item_id) as num_items
58   from order_details
59  group by order_id
60  -- order by num_items desc;
61  order by num_items asc;
62
-- 6. How many orders had more than 12 items?
```

The 'Result Grid' at the bottom displays the results of the query, showing the top 10 orders with the lowest number of items. The columns are 'order_id' and 'num_items'.

order_id	num_items
4410	0
50	0
4307	0
125	0
1044	0
3275	0
147	0
2614	0
200	0
4914	0

I sought to identify the number of orders that contained more than 12 items. Upon analysis, I found that there were 20 orders that exceeded this threshold.

MySQL Workbench

restaurant_db (restaurant_db) x

File Edit View Query Database Server Tools Scripting Help

Navigator: restaurant_db.gproj

SCHEMAS

Filter objects

restaurant_db

- Tables
 - menu_items
 - order_details
- Views
- Stored Procedures
- Functions
- richlist
- sakila
- sys
- world

Administration Schemas

Information

Table: menu_items

Columns:

- menu_item_id smallint PK
- item_name varchar(45)
- category varchar(45)
- price decimal(5,2)

```
61 -- order by num_items asc;
62
63 -- 6. How many orders had more than 12 items?
64 • select count(*) from
65 (
66   select order_id, count(item_id) as num_items
67   from order_details
68   group by order_id
69   having num_items>12
70 ) as num_orders;
71
```

Result Grid

count(*)
20

Filter Rows: Export: Wrap Cell Content: Read Only

Result 36 x

Output

Query Completed

OBJECTIVE 03:

I revisited the menu_items and order_details tables to identify the common attributes that connect them. By joining these tables based on their shared linking point, I was able to create a unified view of the data, allowing for more effective visualization and analysis.

MySQL Workbench interface showing a query editor with the following SQL code:

```
-- OBJECTIVE 3: ANALYZE CUSTOMER BEHAVIOR --  
  
-- 1. Combine the menu_items and order_details tables into a single table.  
select * from menu_items;  
select * from order_details;  
  
-- 2. What were the least and most ordered items? What categories were they in?  
  
-- 3. What were the top 5 orders that spent the most money?
```

The result grid displays 11 rows of data from the order_details table:

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

MySQL Workbench interface showing a query editor with the following SQL code:

```
-- OBJECTIVE 3: ANALYZE CUSTOMER BEHAVIOR --  
  
-- 1. Combine the menu_items and order_details tables into a single table.  
select * from menu_items;  
select * from order_details;  
  
select * from order_details od left join menu_items mi  
on od.item_id = mi.menu_item_id;  
  
-- 2. What were the least and most ordered items? What categories were they in?
```

The result grid displays 11 rows of data from the joined query:

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

I identified the Chicken Tacos, a popular Mexican dish, as the item with the lowest order frequency.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure. The 'Query' editor contains the following SQL code:

```
79 • select * from order_details;
80
81 • select * from order_details od left join menu_items mi
82   on od.item_id = mi.menu_item_id;
83
84 -- 2. What were the least and most ordered items? What categories were they in?
85 • select item_name, category, count(order_details_id) as num_purchases
86   from order_details od left join menu_items mi
87   on od.item_id = mi.menu_item_id
88  group by item_name, category
89  order by num_purchases asc;
```

The 'Result Grid' shows the results of the query, sorted by the number of purchases in ascending order. The first row is 'Chicken Tacos' with 123 purchases.

item_name	category	num_purchases
Chicken Tacos	Mexican	123
Hot Dog	American	137
Potstickers	Asian	205
Cheese Lasagna	Italian	207
Steak Tacos	Mexican	214
Cheese Quesadillas	Mexican	233
Chips & Guacamole	Mexican	237
Veggie Burger	American	238
Shrimp Scampi	Italian	239
Fettuccine Alfredo	Italian	249
Hot Dog	American	257

I discovered that the Hamburger, a classic American dish, is the most frequently ordered item.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The 'Schemas' pane on the left shows the database structure. The 'Query' editor contains the following SQL code:

```
79 • select * from order_details;
80
81 • select * from order_details od left join menu_items mi
82   on od.item_id = mi.menu_item_id;
83
84 -- 2. What were the least and most ordered items? What categories were they in?
85 • select item_name, category, count(order_details_id) as num_purchases
86   from order_details od left join menu_items mi
87   on od.item_id = mi.menu_item_id
88  group by item_name, category
89  order by num_purchases desc;
```

The 'Result Grid' shows the results of the query, sorted by the number of purchases in descending order. The first row is 'Hamburger' with 622 purchases.

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

I pinpointed the top 5 most popular orders in terms of total spend.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The SQL editor contains a query to find the top 5 orders by total spend. The results are displayed in a table below the editor.

```
91
92 -- 3. What were the top 5 orders that spent the most money?
93 • select order_id, sum(price) as total_spend
94 from order_details od left join menu_items mi
95 on od.item_id = mi.menu_item_id
96 group by order_id
97 order by total_spend desc
98 limit 5;
99
100 -- 4. View the details of the highest spend order. What insights can you gather from the
101
```

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

Result 55 x Read Only

Query Completed

I delved deeper into the details of order_id 440 and found that Italian dishes are the most frequently ordered category.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The query editor contains the following SQL code:

```
98 limit 5;  
99  
100 -- 4. View the details of the highest spend order. What insights can you gather from the results?  
101 • select *  
102 from order_details od left join menu_items mi  
103 on od.item_id = mi.menu_item_id  
104 where order_id = 440;
```

The result grid displays 16 rows of data for order_id 440. The columns are: order_details_id, order_id, order_date, order_time, item_id, menu_item_id, item_name, category, and price.

order_details_id	order_id	order_date	order_time	item_id	menu_item_id	item_name	category	price
1003	440	2023-01-08	12:16:34	116	116	Steak Tacos	Mexican	13.95
1004	440	2023-01-08	12:16:34	103	103	Hot Dog	American	9.00
1005	440	2023-01-08	12:16:34	124	124	Spaghetti	Italian	14.50
1006	440	2023-01-08	12:16:34	125	125	Spaghetti & Meatballs	Italian	17.95
1007	440	2023-01-08	12:16:34	125	125	Spaghetti & Meatballs	Italian	17.95
1008	440	2023-01-08	12:16:34	126	126	Fettuccine Alfredo	Italian	14.50
1009	440	2023-01-08	12:16:34	126	126	Fettuccine Alfredo	Italian	14.50
1010	440	2023-01-08	12:16:34	109	109	Korean Beef Bowl	Asian	17.95
1011	440	2023-01-08	12:16:34	127	127	Meat Lasagna	Italian	17.95
1012	440	2023-01-08	12:16:34	113	113	Edamame	Asian	5.00
1013	440	2023-01-08	12:16:34	122	122	Chips & Salsa	Mexican	7.00
1014	440	2023-01-08	12:16:34	131	131	Chicken Parmesan	Italian	17.95
1015	440	2023-01-08	12:16:34	106	106	French Fries	American	7.00
1016	440	2023-01-08	12:16:34	132	132	Eggplant Parmesan	Italian	16.95

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The query editor contains the following SQL code:

```
96 group by order_id  
97 order by total_spend desc  
98 limit 5;  
99  
100 -- 4. View the details of the highest spend order. What insights can you gather from the results?  
101 • select category, count(item_id) as num_items  
102 from order_details od left join menu_items mi  
103 on od.item_id = mi.menu_item_id  
104 where order_id = 440  
105 group by category;
```

The result grid displays 4 rows of data for order_id 440. The columns are: category and num_items.

category	num_items
Mexican	2
American	2
Italian	8
Asian	2

An analysis of the order data reveals that Italian food is the most popular choice among customers. Upon further examination of individual order IDs, it was found that four out of five order IDs consistently order Italian dishes, indicating a strong preference for this dish.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The SQL editor contains the following query:

```
109 where order_id = 440
110 group by category;
111
112 -- 5. View the details of the top 5 highest spend orders. What insights can you gather from the results?
113 • select category, count(item_id) as num_items
114 from order_details od left join menu_items mi
115 on od.item_id = mi.menu_item_id
116 where order_id IN (440,2075,1957,330,2675)
117 group by category;
118
```

The results are displayed in the 'Result Grid' tab, showing the following data:

category	num_items
Asian	17
American	10
Italian	26
Mexican	16

The interface also shows the 'Navigator' panel on the left with the 'restaurant_db' schema selected, and the 'Object Info' and 'Session' tabs at the bottom.

The screenshot shows the MySQL Workbench interface with the 'restaurant_db' database selected. The SQL editor contains the following query:

```
119 • select order_id, category, count(item_id) as num_items
120 from order_details od left join menu_items mi
121 on od.item_id = mi.menu_item_id
122 where order_id IN (440,2075,1957,330,2675)
123 group by order_id, category;
```

The results are displayed in the 'Result Grid' tab, showing the following data:

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
2075	Asian	3
2075	Mexican	3
2075	American	1
2075	Italian	6
2675	American	3
2675	Asian	3
2675	Italian	4
2675	Mexican	4

The interface also shows the 'Navigator' panel on the left with the 'restaurant_db' schema selected, and the 'Object Info' and 'Session' tabs at the bottom.

SQL script saved to 'C:\Users\helbert\Desktop\projects\03 - Data Analysts Portfolio Project - Restaurant Order Analysts in SQL\Restaurant-Orders+MySQL\restaurant_db gproj.sql'

SUMMARY OF FINDINGS:

1. Menu Insights (Objective 01):

- Edamame and Asian menu items are the least expensive.
- Shrimp Scampi (from the Italian menu) is the most expensive item.
- There are 9 Italian dishes out of a total of 32 dishes.
- Spaghetti tends to be the least expensive option, while Shrimp Scampi is typically the most expensive.

2. Order Details Insights (Objective 02):

- 5,370 orders were recorded within the specified date range.
- A total of 12,234 items were ordered.
- Seven order IDs contain 14 items each.
- Some orders have zero items.
- There were 20 orders with more than 12 items.

3. Common Attributes (Objective 03):

- Chicken Tacos (a popular Mexican dish) has the lowest order frequency.
- The Hamburger (a classic American dish) is the most frequently ordered item.
- The top 5 orders in terms of total spend were identified.

4. Italian Food Popularity:

- Order ID 440 consistently orders Italian dishes, indicating a strong preference for Italian food among customers.

These insights provide valuable information for optimizing menu offerings, understanding customer preferences, and managing inventory.

KEY INSIGHTS:

1. Menu Insights:

- **Pricing Strategy:** The fact that “Edamame” and Asian menu items are the least expensive suggests that these items could serve as attractive options for cost-conscious customers. Consider promoting them as value-for-money choices.
- **Upselling Opportunity:** On the other hand, “Shrimp Scampi” being the most expensive item presents an opportunity for upselling. Train staff to recommend this premium dish to customers who are willing to splurge.
- **Italian Cuisine Focus:** With 9 out of 32 dishes belonging to Italian cuisine, it’s clear that Italian food is popular. Consider expanding the Italian menu or featuring Italian specials prominently.

2. Order Details Insights:

- **Order Volume Trends:** The 5,370 recorded orders indicate a steady flow of business. Monitor order volume over time to identify peak hours and allocate resources accordingly.
- **Zero-Item Orders:** Investigate why some orders had zero items. Are these cancellations or system errors? Address any issues to improve customer experience.
- **Large Group Orders:** The 20 orders with more than 12 items likely represent group dining. Consider offering group discounts or special packages to attract such gatherings.

3. Common Attributes:

- **Chicken Tacos vs. Hamburger:** The contrast between low-frequency “Chicken Tacos” and high-frequency “Hamburger” orders highlights customer preferences. Consider promoting the Hamburger further or experimenting with Chicken Tacos variations.
- **Top 5 High-Spend Orders:** Analyze the top 5 orders based on total spend. Are they special occasions, corporate events, or regular customers? Tailor marketing efforts accordingly.

4. Italian Food Popularity:

- **Customer Segmentation:** Order ID 440 consistently preferring Italian dishes suggests a specific customer segment. Create targeted marketing campaigns for Italian food enthusiasts.
- **Menu Optimization:** Given the popularity of Italian cuisine, ensure that Italian dishes are well-represented on the menu and maintain their quality.