SQL PROJECT (Restaurant Sales Analysis)

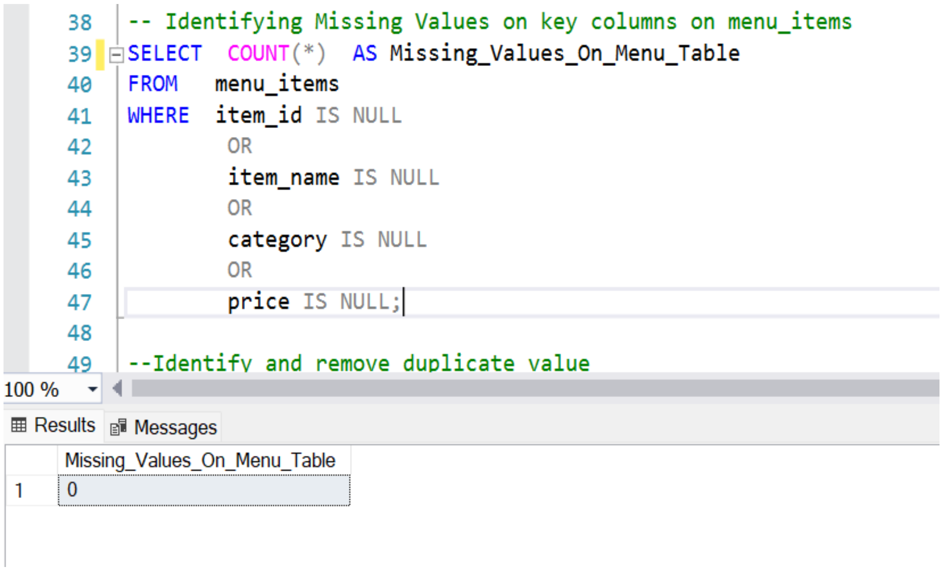
DATA CLEANING AND PROCESSING

Data cleaning is the crucial process of identifying and correcting errors, inconsistencies, and inaccuracies in datasets to enhance their quality for reliable analysis.

This entails dealing with problems such as duplicates, missing values, standardizing data types, etc. The cleaned dataset formed the solid foundation for subsequent analyses, it helps to ensure that data is accurate and well prepared for analysis.

Data Cleaning on Menu Table: The key column in the menu table was cleaned using the following data cleaning procedures.

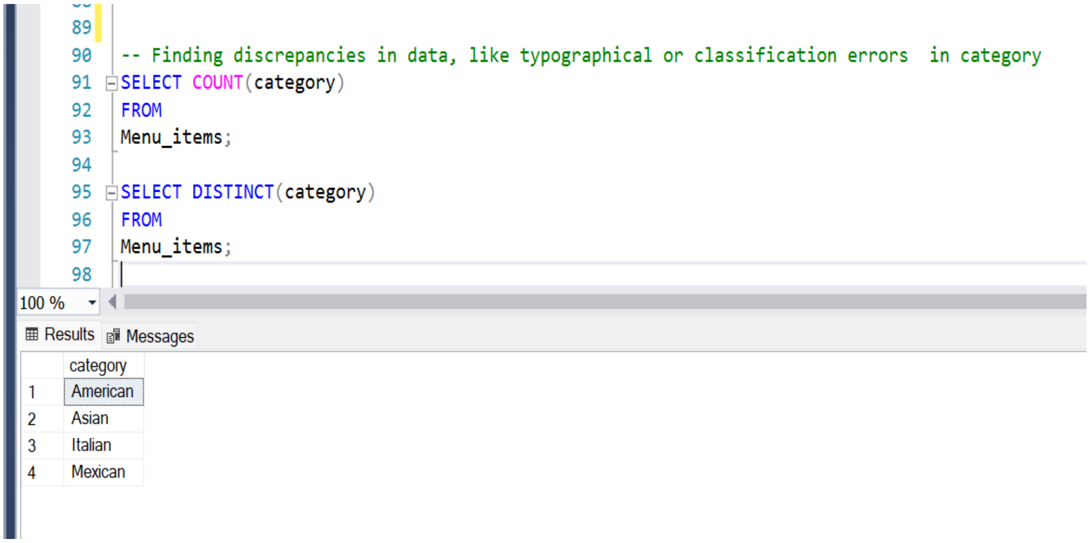
The generated result indicates that there are no missing values in the key columns in the menu table.



```sql
-- Identifying Missing Values on key columns on menu_items
SELECT  COUNT(*)  AS Missing_Values_On_Menu_Table
FROM    menu_items
WHERE   item_id IS NULL
         OR
         item_name IS NULL
         OR
         category IS NULL
         OR
         price IS NULL;

--Identify and remove duplicate value
```

	Missing_Values_On_Menu_Table
1	0

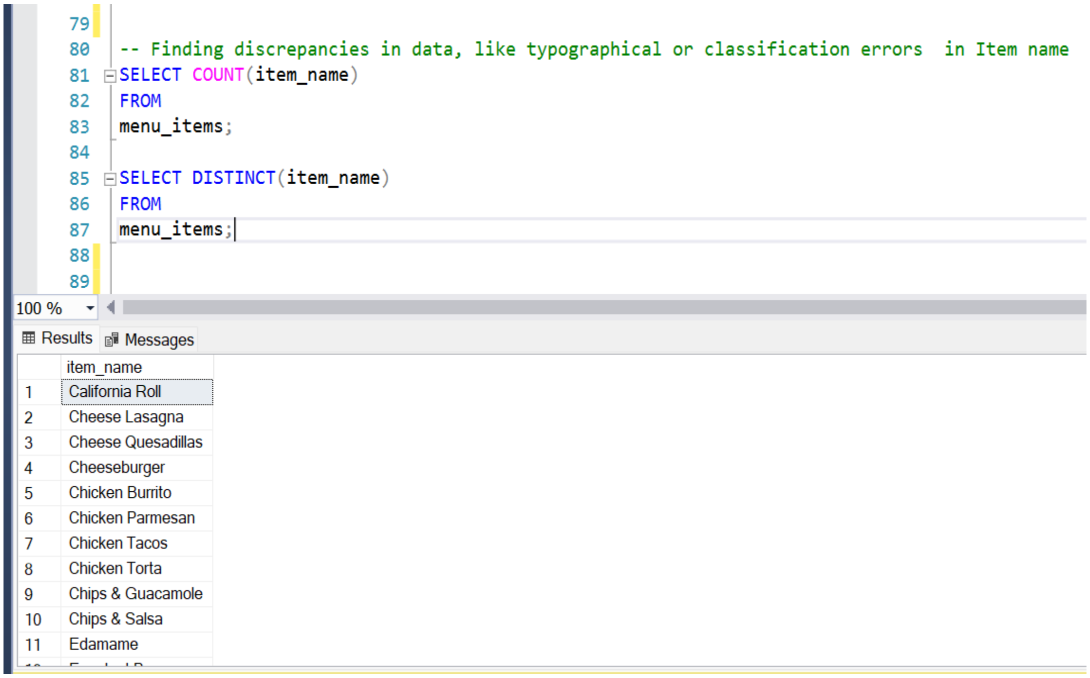
The generated result affirms that all values in the menu Category column are spelled correctly and are categorized appropriately.



```sql
-- Finding discrepancies in data, like typographical or classification errors  in category
SELECT COUNT(category)
FROM
Menu_items;

SELECT DISTINCT(category)
FROM
Menu_items;
```

	category
1	American
2	Asian
3	Italian
4	Mexican

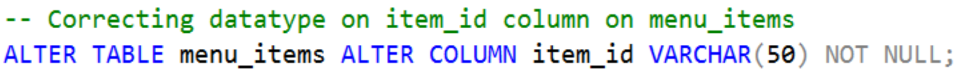
The generated result affirms that all values in the menu Category column are spelled correctly and are categorized appropriately.



```sql
-- Finding discrepancies in data, like typographical or classification errors  in Item name
SELECT COUNT(item_name)
FROM
menu_items;

SELECT DISTINCT(item_name)
FROM
menu_items;
```

	item_name
1	California Roll
2	Cheese Lasagna
3	Cheese Quesadillas
4	Cheeseburger
5	Chicken Burrito
6	Chicken Parmesan
7	Chicken Tacos
8	Chicken Torta
9	Chips & Guacamole
10	Chips & Salsa
11	Edamame

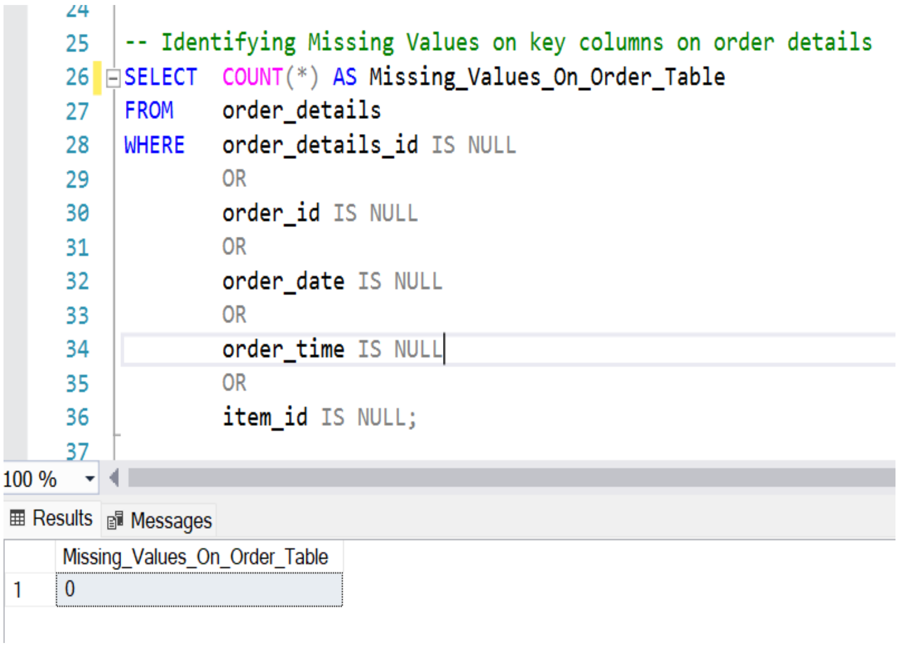
The generated result shows the datatype on the Primary key column (item id) on the menu table



```sql
-- Correcting datatype on item_id column on menu_items
ALTER TABLE menu_items ALTER COLUMN item_id VARCHAR(50) NOT NULL;
```

Data Cleaning on order Table: The key column in the ordered table was cleaned using the following data cleaning procedures.

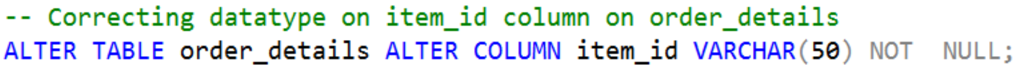
According to the generated result no values are missing from the order table's key columns.



```
-- Identifying Missing Values on key columns on order details
SELECT  COUNT(*) AS Missing_Values_On_Order_Table
FROM    order_details
WHERE   order_details_id IS NULL
        OR
        order_id IS NULL
        OR
        order_date IS NULL
        OR
        order_time IS NULL
        OR
        item_id IS NULL;
```

	Missing_Values_On_Order_Table
1	0

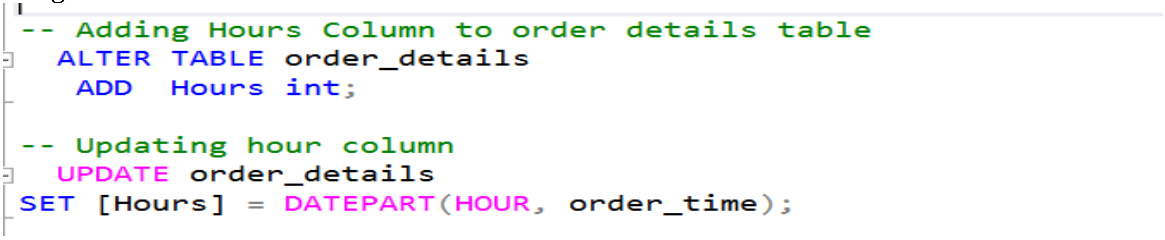
The generated result shows the datatype on the Primary key column (item id) on the order details table



```
-- Correcting datatype on item_id column on order_details
ALTER TABLE order_details ALTER COLUMN item_id VARCHAR(50) NOT  NULL;
```

Added a new column hour to the Order details table

A new column called "Hour" was added to the order table, calculating the time of the day that had more orders and less orders. This column is essential for determining the hour in a day that yield more sales and which item name is ordered most, as it offers insightful information about the most lucrative product categories.



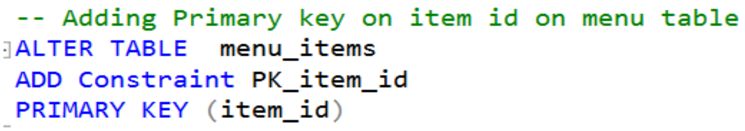
```
-- Adding Hours Column to order details table
  ALTER TABLE order_details
   ADD  Hours int;

-- Updating hour column
  UPDATE order_details
SET [Hours] = DATEPART(HOUR, order_time);
```

Established Relationships Between Tables.

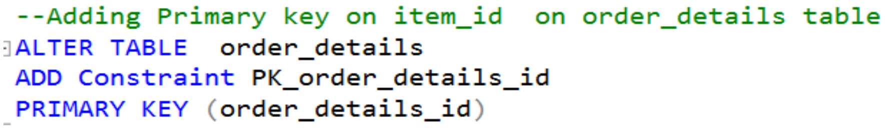
One of the most important aspects of database design in SQL is establishing relationships between tables, which is usually accomplished by using primary keys and foreign keys. A primary key functions as a unique identifier for each record in a table and is a fundamental concept in database design. Every record in the table has a unique value stored in a primary key. This uniqueness ensures that each record can be uniquely identified and distinguished from others in the same table.

A column, or group of columns, in a relational database table that creates a connection between data in two tables is called a foreign key. In addition to establishing relationships between tables, it serves to maintain referential integrity. One table's foreign key can be used to refer to another table's primary key. To guarantee that values in the foreign key column(s) match values already present in the primary key column of the referenced table, this relationship establishes a link between the data in the two tables.

The following procedure was used to establish relationships between the menu and order table.

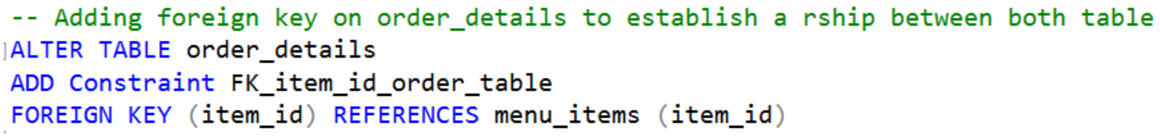


```
-- Adding Primary key on item id on menu table
ALTER TABLE  menu_items
ADD Constraint PK_item_id
PRIMARY KEY (item_id)
```



```
--Adding Primary key on item_id  on order_details table
ALTER TABLE  order_details
ADD Constraint PK_order_details_id
PRIMARY KEY (order_details_id)
```

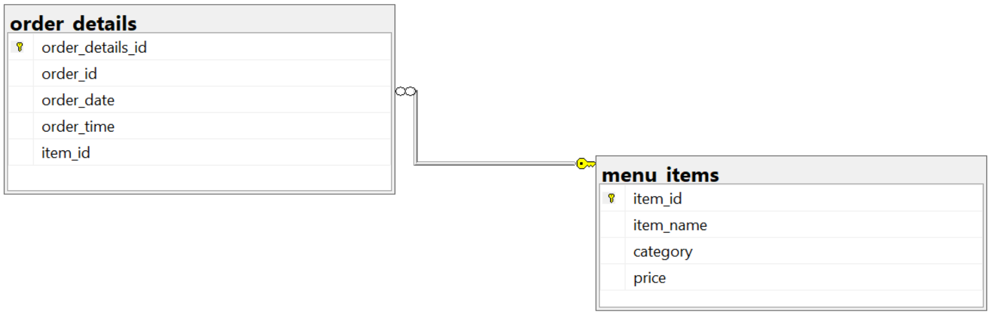
Adding Primary Key



```
-- Adding foreign key on order_details to establish a rship between both table
ALTER TABLE order_details
ADD Constraint FK_item_id_order_table
FOREIGN KEY (item_id) REFERENCES menu_items (item_id)
```

Adding Foreign Key

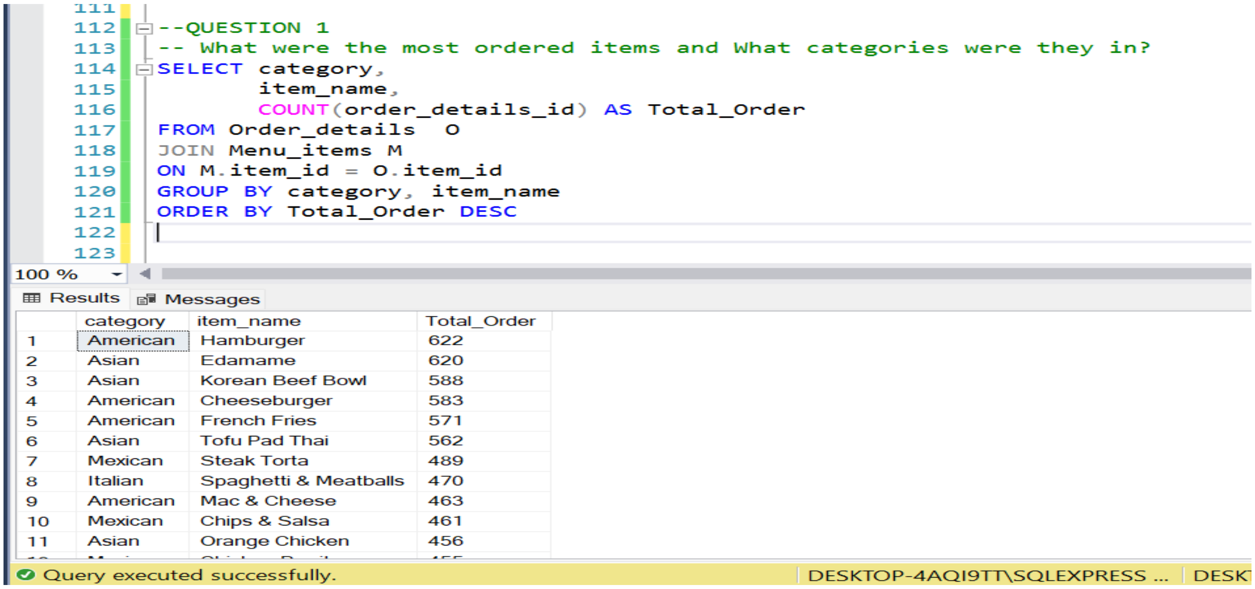
Establishing connections between the tables is essential to the analysis. In addition to maintaining data organization, it also speeds up query execution, prevents redundant data entry, and creates a flexible database structure. The outcome of the relationship that was established between the tables is shown below.



Recommended Analysis

QUESTION 1.

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

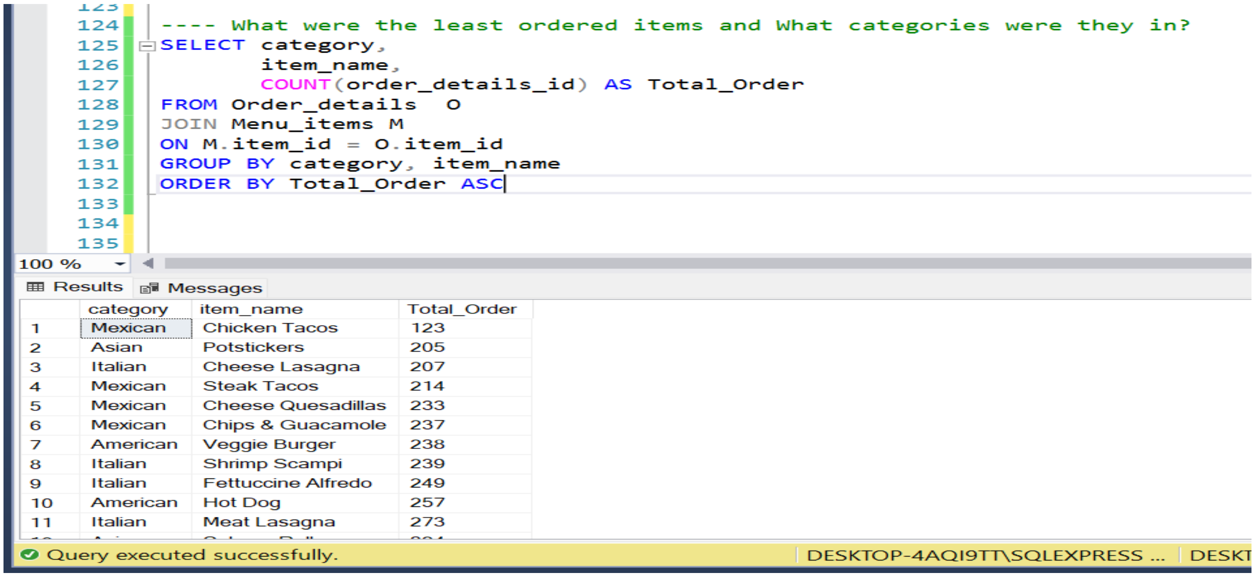


```
--QUESTION 1
-- What were the most ordered items and What categories were they in?
SELECT category,
       item_name,
       COUNT(order_details_id) AS Total_Order
FROM Order_details  O
JOIN Menu_items M
ON M.item_id = O.item_id
GROUP BY category, item_name
ORDER BY Total_Order DESC
```

	category	item_name	Total_Order
1	American	Hamburger	622
2	Asian	Edamame	620
3	Asian	Korean Beef Bowl	588
4	American	Cheeseburger	583
5	American	French Fries	571
6	Asian	Tofu Pad Thai	562
7	Mexican	Steak Torta	489
8	Italian	Spaghetti & Meatballs	470
9	American	Mac & Cheese	463
10	Mexican	Chips & Salsa	461
11	Asian	Orange Chicken	456

Query executed successfully. | DESKTOP-4AQI9TT\SQLEXPRESS ... | DESKT

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



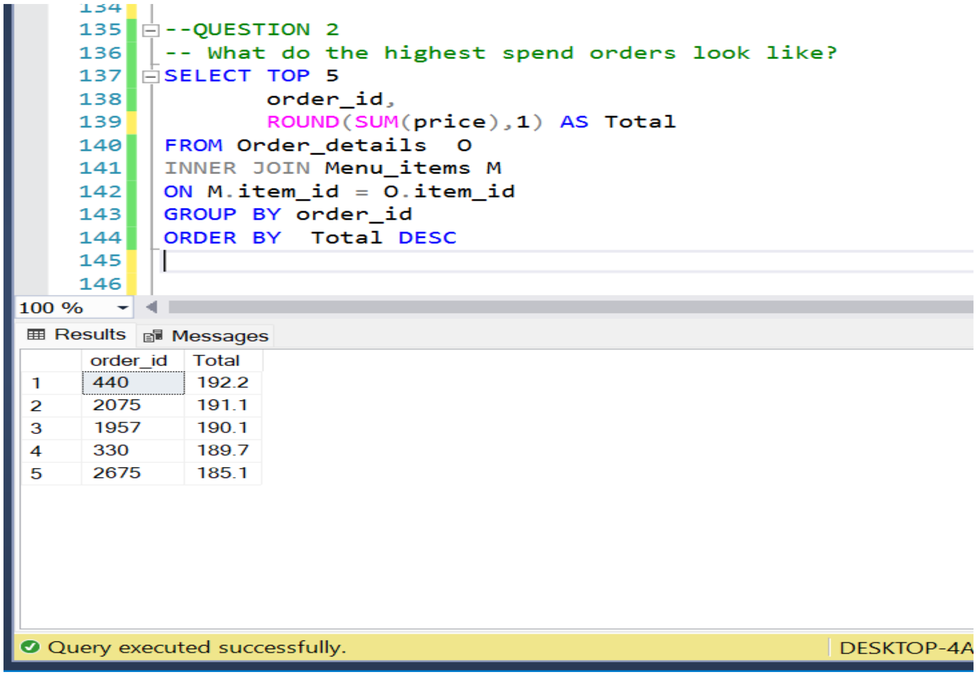
```
---- What were the least ordered items and What categories were they in?
SELECT category,
       item_name,
       COUNT(order_details_id) AS Total_Order
FROM Order_details  O
JOIN Menu_items M
ON M.item_id = O.item_id
GROUP BY category, item_name
ORDER BY Total_Order ASC
```

	category	item_name	Total_Order
1	Mexican	Chicken Tacos	123
2	Asian	Potstickers	205
3	Italian	Cheese Lasagna	207
4	Mexican	Steak Tacos	214
5	Mexican	Cheese Quesadillas	233
6	Mexican	Chips & Guacamole	237
7	American	Veggie Burger	238
8	Italian	Shrimp Scampi	239
9	Italian	Fettuccine Alfredo	249
10	American	Hot Dog	257
11	Italian	Meat Lasagna	273

Query executed successfully. | DESKTOP-4AQI9TT\SQLEXPRESS ... | DESKT

QUESTION 2.

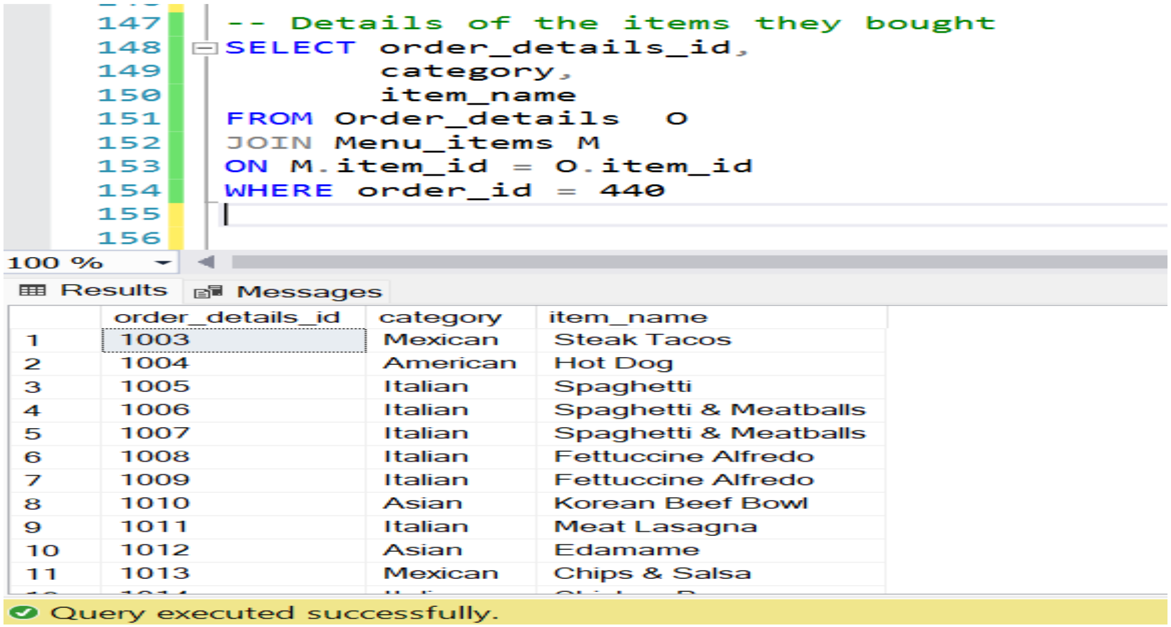
What do the highest spend orders look like? Which items did they buy and how much did they spend?



```sql
--QUESTION 2
-- What do the highest spend orders look like?
SELECT TOP 5
        order_id,
        ROUND(SUM(price),1) AS Total
FROM Order_details  O
INNER JOIN Menu_items M
ON M.item_id = O.item_id
GROUP BY order_id
ORDER BY  Total DESC
```

	order_id	Total
1	440	192.2
2	2075	191.1
3	1957	190.1
4	330	189.7
5	2675	185.1

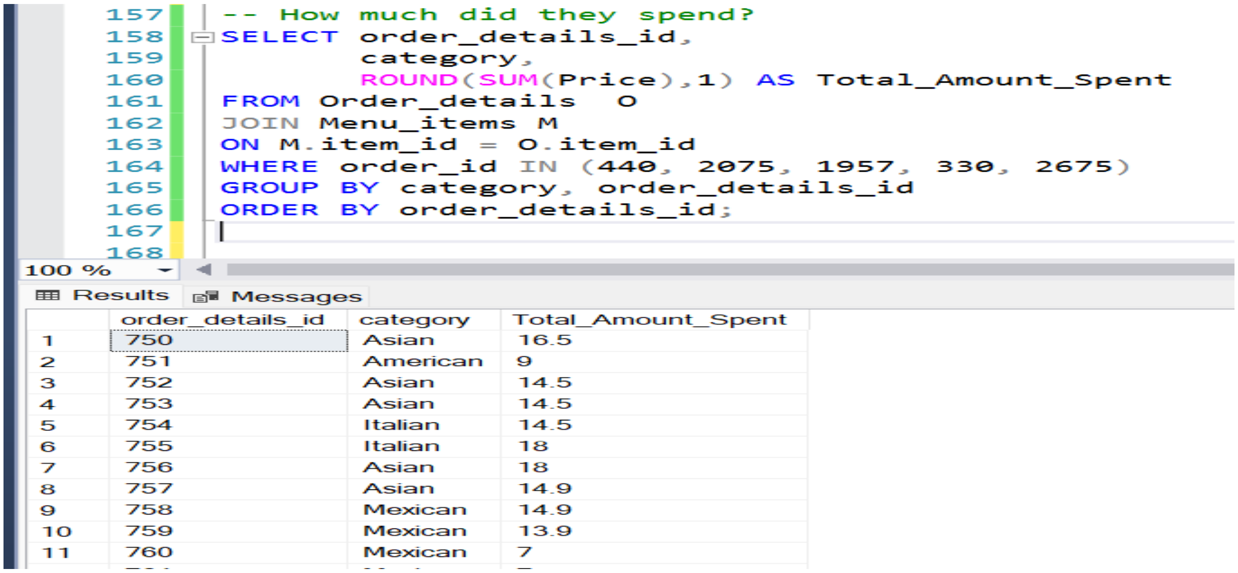
Query executed successfully.



```sql
-- Details of the items they bought
SELECT order_details_id,
       category,
       item_name
FROM Order_details  O
JOIN Menu_items M
ON M.item_id = O.item_id
WHERE order_id = 440
```

	order_details_id	category	item_name
1	1003	Mexican	Steak Tacos
2	1004	American	Hot Dog
3	1005	Italian	Spaghetti
4	1006	Italian	Spaghetti & Meatballs
5	1007	Italian	Spaghetti & Meatballs
6	1008	Italian	Fettuccine Alfredo
7	1009	Italian	Fettuccine Alfredo
8	1010	Asian	Korean Beef Bowl
9	1011	Italian	Meat Lasagna
10	1012	Asian	Edamame
11	1013	Mexican	Chips & Salsa

Query executed successfully.

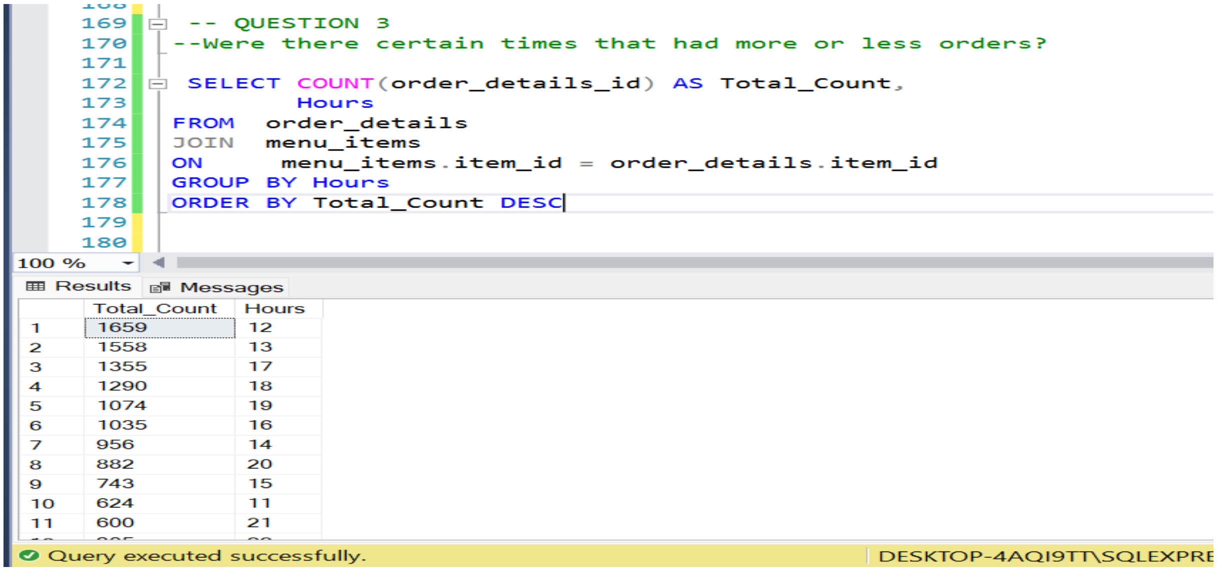


```
-- How much did they spend?
SELECT order_details_id,
       category,
       ROUND(SUM(Price),1) AS Total_Amount_Spent
FROM Order_details  O
JOIN Menu_items M
ON M.item_id = O.item_id
WHERE order_id IN (440, 2075, 1957, 330, 2675)
GROUP BY category, order_details_id
ORDER BY order_details_id;
```

	order_details_id	category	Total_Amount_Spent
1	750	Asian	16.5
2	751	American	9
3	752	Asian	14.5
4	753	Asian	14.5
5	754	Italian	14.5
6	755	Italian	18
7	756	Asian	18
8	757	Asian	14.9
9	758	Mexican	14.9
10	759	Mexican	13.9
11	760	Mexican	7

QUESTION 3.

Were there certain times that had more or less orders?



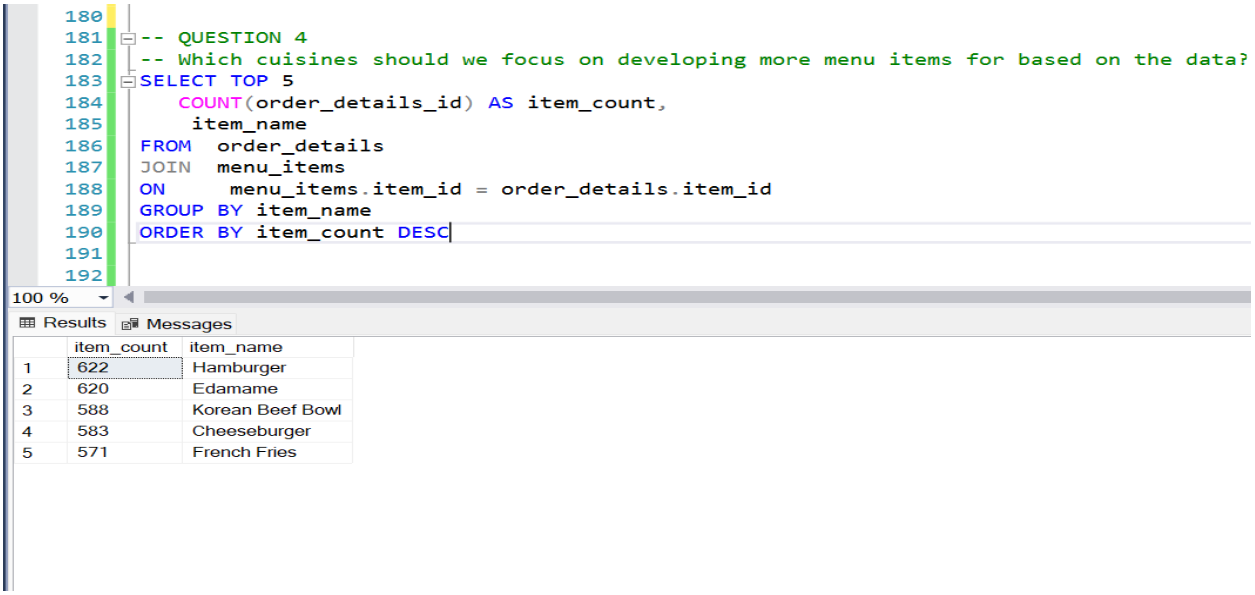
```
-- QUESTION 3
--Were there certain times that had more or less orders?

 SELECT COUNT(order_details_id) AS Total_Count,
        Hours
FROM   order_details
JOIN   menu_items
ON      menu_items.item_id = order_details.item_id
GROUP BY Hours
ORDER BY Total_Count DESC
```

	Total_Count	Hours
1	1659	12
2	1558	13
3	1355	17
4	1290	18
5	1074	19
6	1035	16
7	956	14
8	882	20
9	743	15
10	624	11
11	600	21

Query executed successfully.   DESKTOP-4AQI9TT\SQLEXPRE

QUESTION 4.

Which cuisines should we focus on developing more menu items for based on the data?



```sql
-- QUESTION 4
-- Which cuisines should we focus on developing more menu items for based on the data?
SELECT TOP 5
    COUNT(order_details_id) AS item_count,
     item_name
FROM   order_details
JOIN   menu_items
ON      menu_items.item_id = order_details.item_id
GROUP BY item_name
ORDER BY item_count DESC
```

	item_count	item_name
1	622	Hamburger
2	620	Edamame
3	588	Korean Beef Bowl
4	583	Cheeseburger
5	571	French Fries