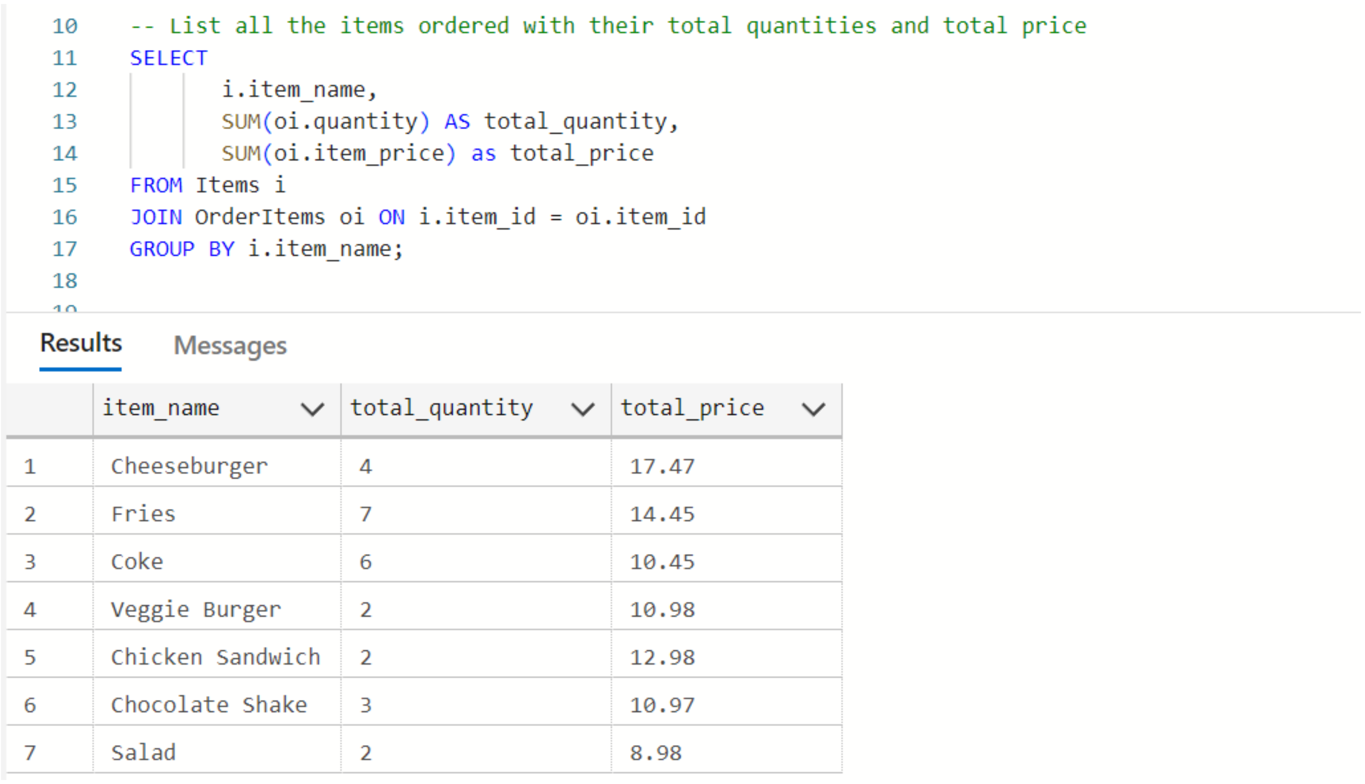
Data Analysis

The following section presents 10 SQL queries along with screenshots of their results, executed on various tables within the Restaurant_DB database. These queries demonstrate a range of SQL functionalities, including:

- **Joins**: Combining data from multiple tables.
- **Where Conditions**: Filtering data based on specified criteria.
- **Group By**: Aggregating data based on specified columns.
- **Having**: Filtering groups based on aggregate functions.
- **Aggregate Functions**: Such as **SUM**, **AVG**, and **MAX**, for summarizing data.
- **Limit**: Restricting the number of results returned.
- **Order By**: Sorting results based on specified columns.
- **Date Intervals**: Analyzing data over specific time periods.

These queries demonstrate various analytical techniques to extract and analyze data effectively from the database.

1. List all the items ordered with their total quantities and total price

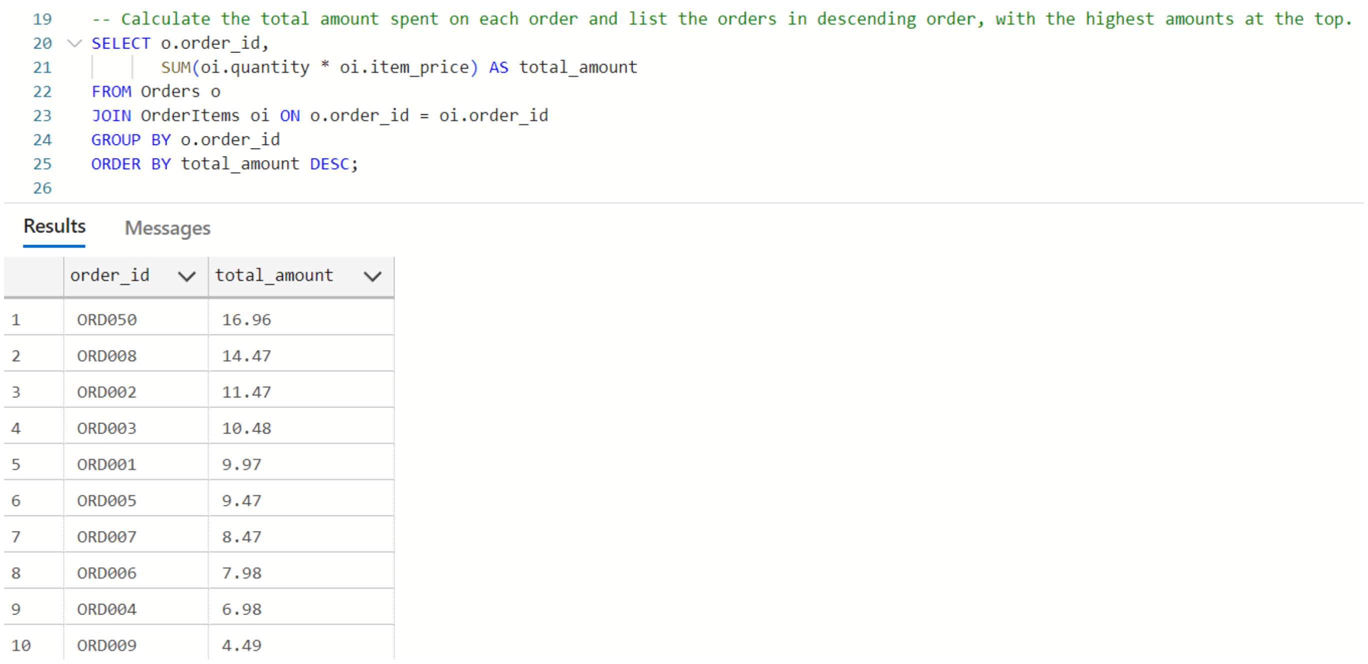


```sql
-- List all the items ordered with their total quantities and total price
SELECT
        i.item_name,
        SUM(oi.quantity) AS total_quantity,
        SUM(oi.item_price) as total_price
FROM Items i
JOIN OrderItems oi ON i.item_id = oi.item_id
GROUP BY i.item_name;
```

Results   Messages

	item_name	total_quantity	total_price
1	Cheeseburger	4	17.47
2	Fries	7	14.45
3	Coke	6	10.45
4	Veggie Burger	2	10.98
5	Chicken Sandwich	2	12.98
6	Chocolate Shake	3	10.97
7	Salad	2	8.98

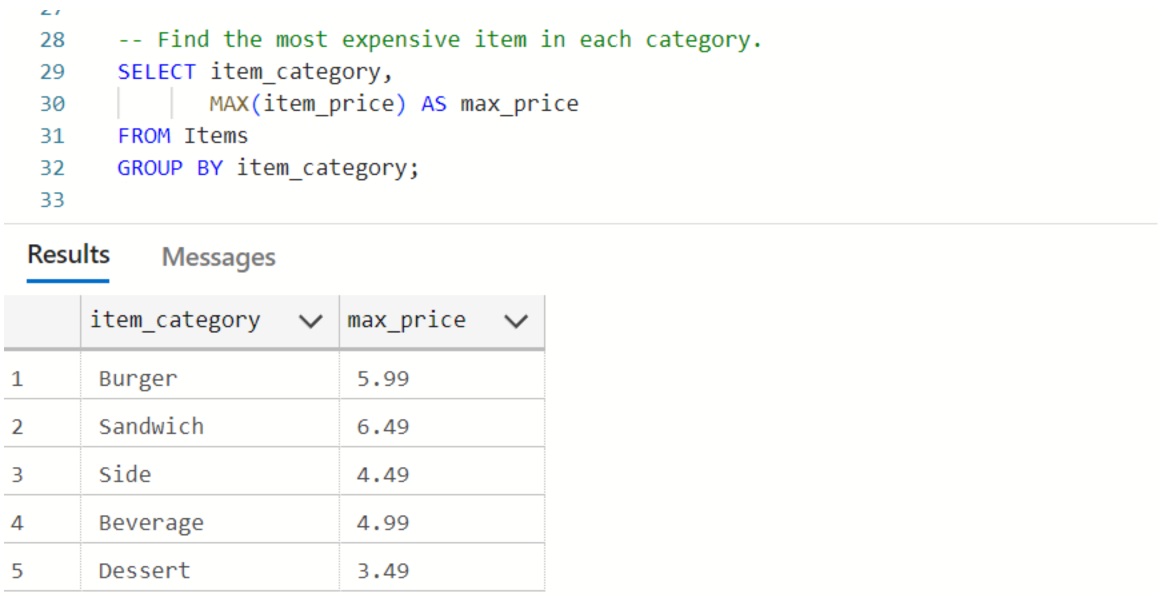
2. Calculate the total amount spent on each order and list the orders in descending order, with the highest amounts at the top.



```sql
-- Calculate the total amount spent on each order and list the orders in descending order, with the highest amounts at the top.
SELECT o.order_id,
       SUM(oi.quantity * oi.item_price) AS total_amount
FROM Orders o
JOIN OrderItems oi ON o.order_id = oi.order_id
GROUP BY o.order_id
ORDER BY total_amount DESC;
```

	order_id	total_amount
1	ORD050	16.96
2	ORD008	14.47
3	ORD002	11.47
4	ORD003	10.48
5	ORD001	9.97
6	ORD005	9.47
7	ORD007	8.47
8	ORD006	7.98
9	ORD004	6.98
10	ORD009	4.49

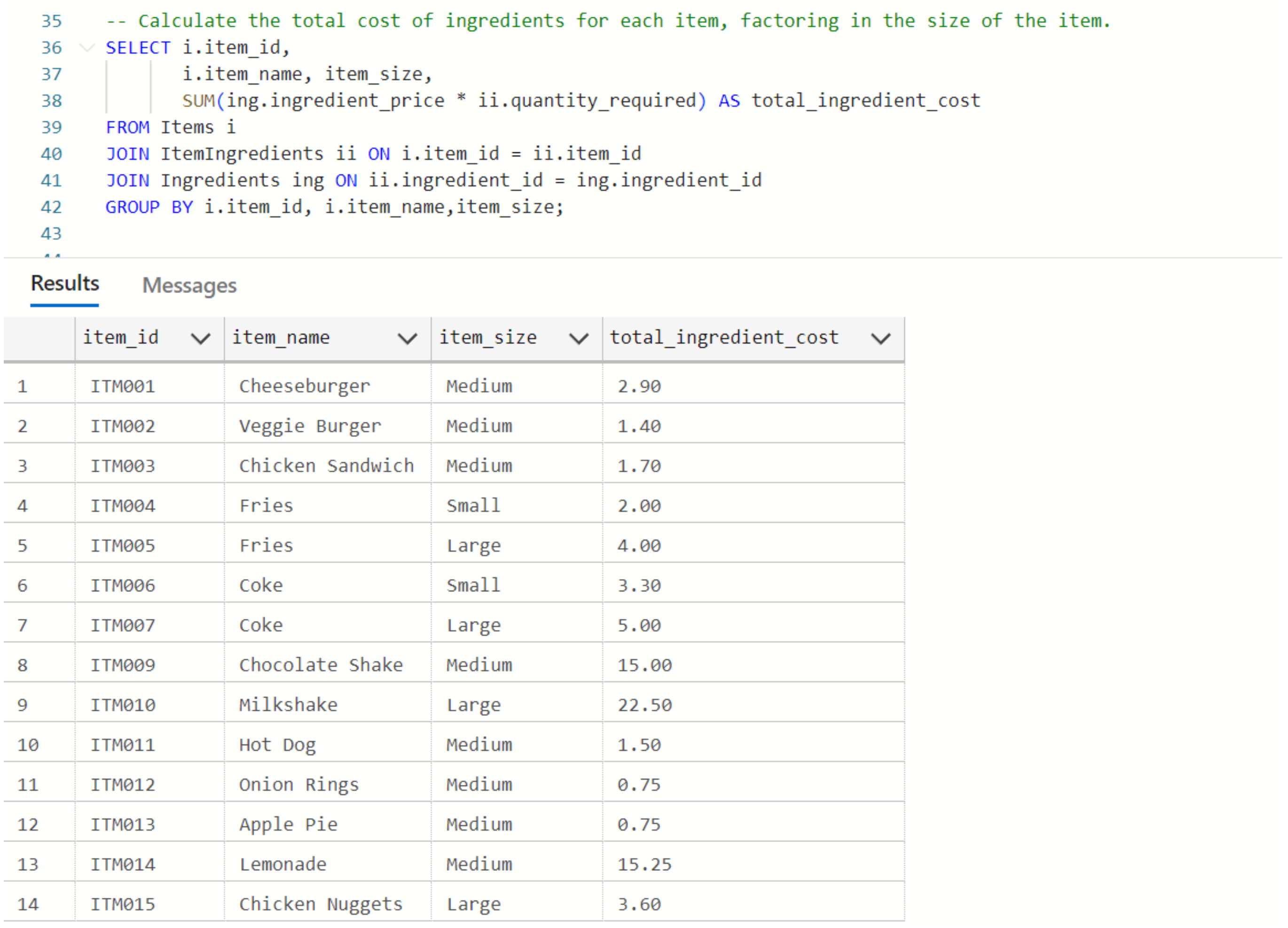
3. List the most expensive item in each category.



```sql
-- Find the most expensive item in each category.
SELECT item_category,
       MAX(item_price) AS max_price
FROM Items
GROUP BY item_category;
```

	item_category	max_price
1	Burger	5.99
2	Sandwich	6.49
3	Side	4.49
4	Beverage	4.99
5	Dessert	3.49

4. Calculate the total cost of ingredients for each item, factoring in the size of the item.

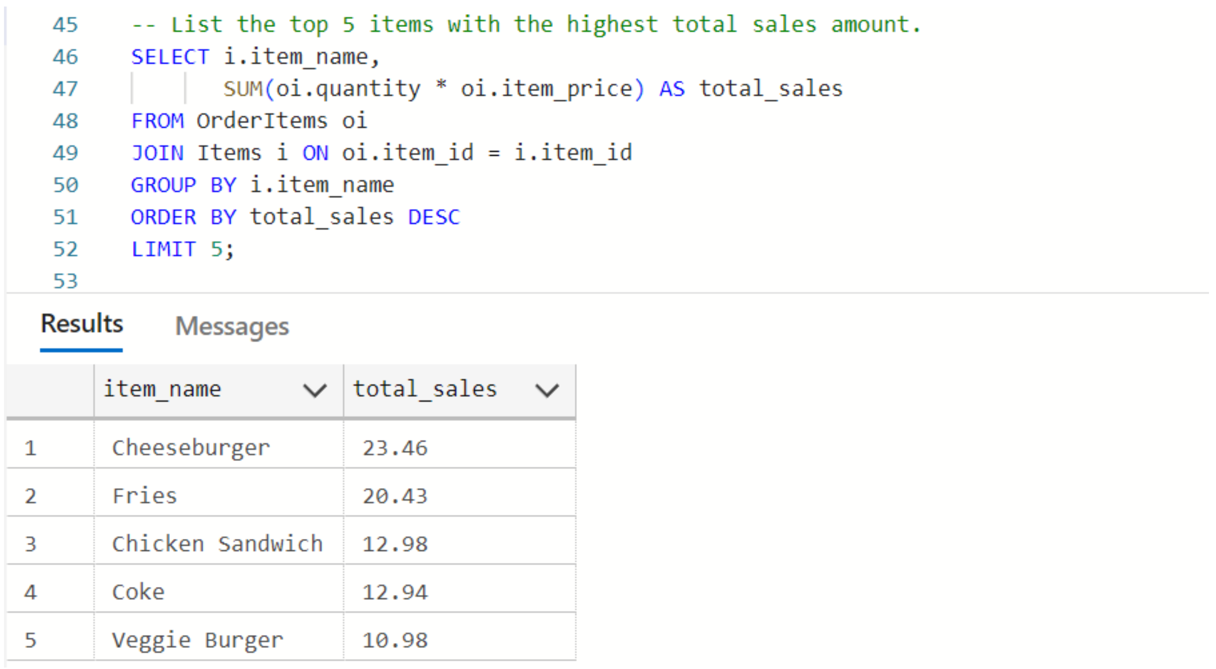


```sql
-- Calculate the total cost of ingredients for each item, factoring in the size of the item.
SELECT i.item_id,
       i.item_name, item_size,
       SUM(ing.ingredient_price * ii.quantity_required) AS total_ingredient_cost
FROM Items i
JOIN ItemIngredients ii ON i.item_id = ii.item_id
JOIN Ingredients ing ON ii.ingredient_id = ing.ingredient_id
GROUP BY i.item_id, i.item_name,item_size;
```

Results   Messages

	item_id	item_name	item_size	total_ingredient_cost
1	ITM001	Cheeseburger	Medium	2.90
2	ITM002	Veggie Burger	Medium	1.40
3	ITM003	Chicken Sandwich	Medium	1.70
4	ITM004	Fries	Small	2.00
5	ITM005	Fries	Large	4.00
6	ITM006	Coke	Small	3.30
7	ITM007	Coke	Large	5.00
8	ITM009	Chocolate Shake	Medium	15.00
9	ITM010	Milkshake	Large	22.50
10	ITM011	Hot Dog	Medium	1.50
11	ITM012	Onion Rings	Medium	0.75
12	ITM013	Apple Pie	Medium	0.75
13	ITM014	Lemonade	Medium	15.25
14	ITM015	Chicken Nuggets	Large	3.60

5. List the top 5 items with the highest total sales amount.

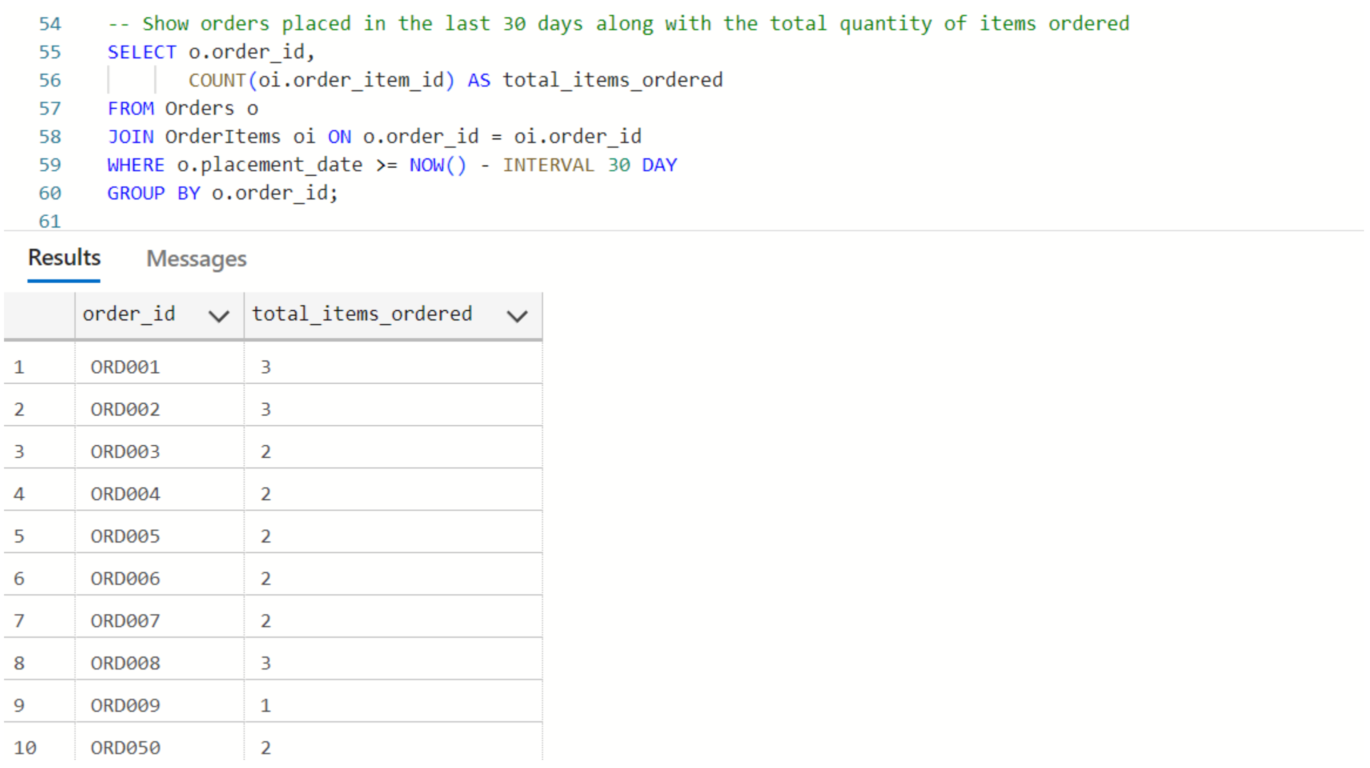


```sql
-- List the top 5 items with the highest total sales amount.
SELECT i.item_name,
       SUM(oi.quantity * oi.item_price) AS total_sales
FROM OrderItems oi
JOIN Items i ON oi.item_id = i.item_id
GROUP BY i.item_name
ORDER BY total_sales DESC
LIMIT 5;
```

Results   Messages

	item_name	total_sales
1	Cheeseburger	23.46
2	Fries	20.43
3	Chicken Sandwich	12.98
4	Coke	12.94
5	Veggie Burger	10.98

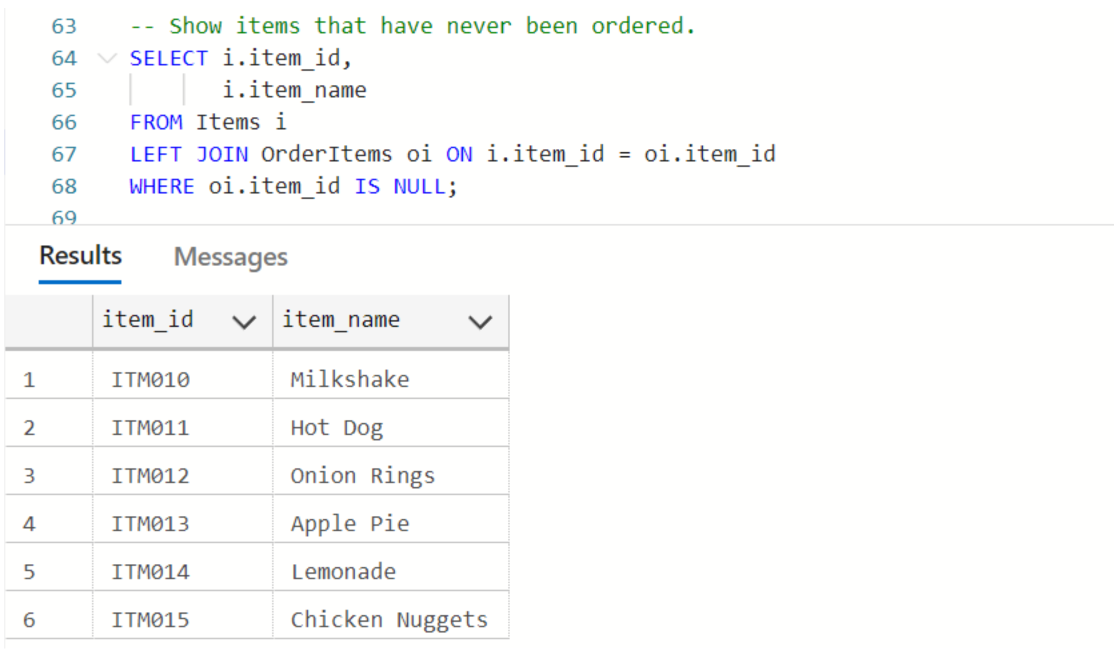
6. Show orders placed in the last 30 days along with the total quantity of items ordered



```sql
-- Show orders placed in the last 30 days along with the total quantity of items ordered
SELECT o.order_id,
       COUNT(oi.order_item_id) AS total_items_ordered
FROM Orders o
JOIN OrderItems oi ON o.order_id = oi.order_id
WHERE o.placement_date >= NOW() - INTERVAL 30 DAY
GROUP BY o.order_id;
```

	order_id	total_items_ordered
1	ORD001	3
2	ORD002	3
3	ORD003	2
4	ORD004	2
5	ORD005	2
6	ORD006	2
7	ORD007	2
8	ORD008	3
9	ORD009	1
10	ORD050	2

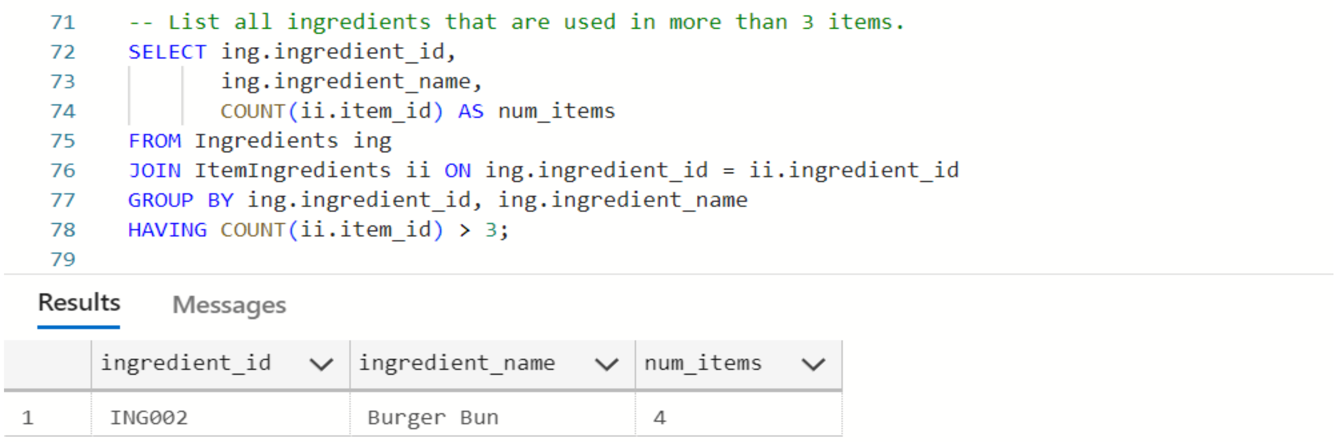
7. Show items that have never been ordered.



```sql
-- Show items that have never been ordered.
SELECT i.item_id,
       i.item_name
FROM Items i
LEFT JOIN OrderItems oi ON i.item_id = oi.item_id
WHERE oi.item_id IS NULL;
```

	item_id	item_name
1	ITM010	Milkshake
2	ITM011	Hot Dog
3	ITM012	Onion Rings
4	ITM013	Apple Pie
5	ITM014	Lemonade
6	ITM015	Chicken Nuggets

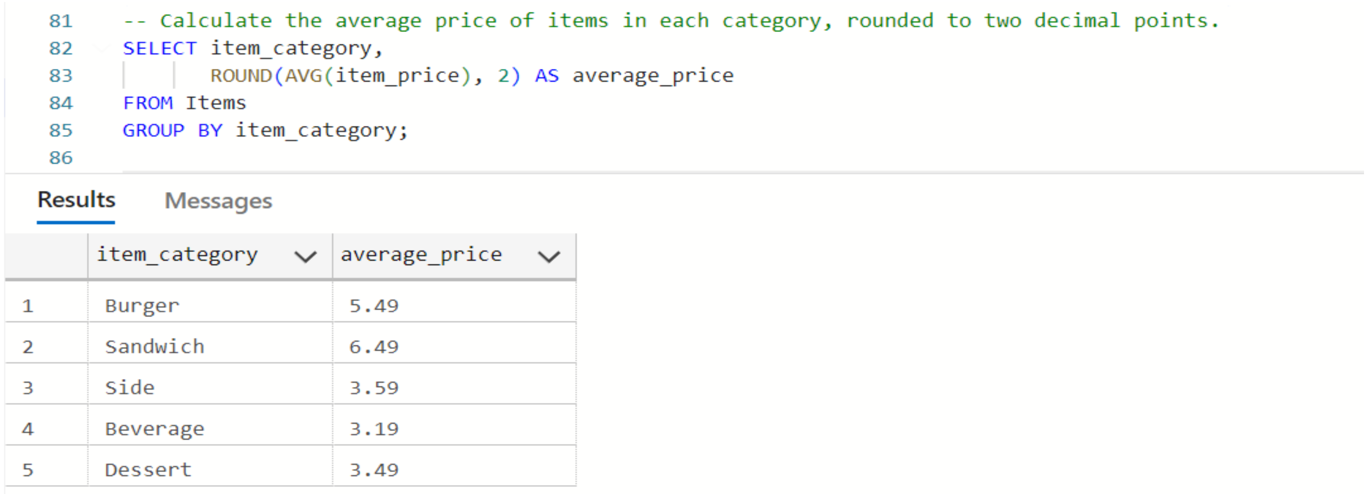
8. List all ingredients that are used in more than 3 items.



```sql
-- List all ingredients that are used in more than 3 items.
SELECT ing.ingredient_id,
       ing.ingredient_name,
       COUNT(ii.item_id) AS num_items
FROM Ingredients ing
JOIN ItemIngredients ii ON ing.ingredient_id = ii.ingredient_id
GROUP BY ing.ingredient_id, ing.ingredient_name
HAVING COUNT(ii.item_id) > 3;
```

	ingredient_id	ingredient_name	num_items
1	ING002	Burger Bun	4

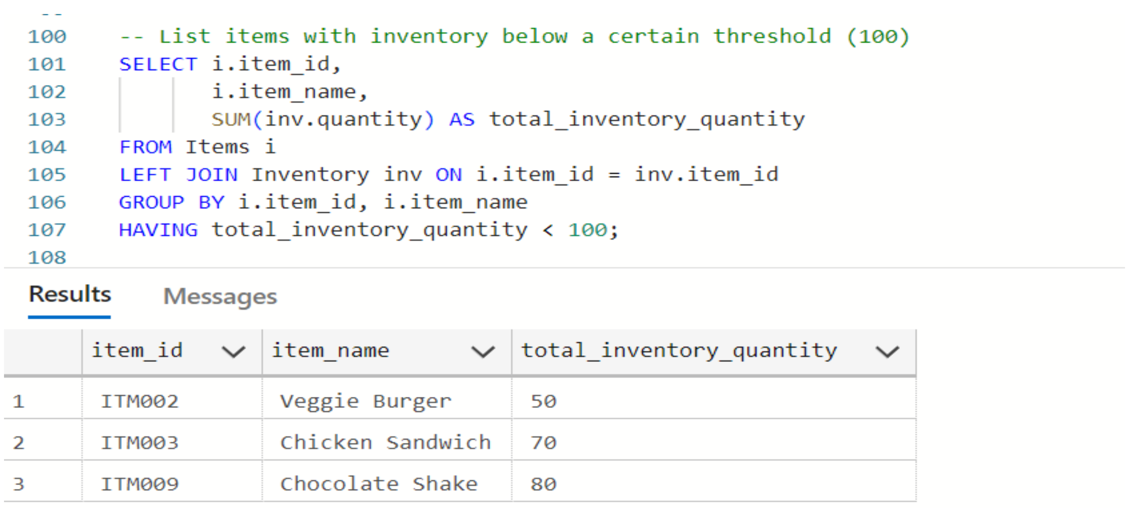
9. Calculate the average price of items in each category, rounded to two decimal points.



```sql
-- Calculate the average price of items in each category, rounded to two decimal points.
SELECT item_category,
       ROUND(AVG(item_price), 2) AS average_price
FROM Items
GROUP BY item_category;
```

	item_category	average_price
1	Burger	5.49
2	Sandwich	6.49
3	Side	3.59
4	Beverage	3.19
5	Dessert	3.49

10. List items with inventory below a certain threshold (100)



```sql
-- List items with inventory below a certain threshold (100)
SELECT i.item_id,
       i.item_name,
       SUM(inv.quantity) AS total_inventory_quantity
FROM Items i
LEFT JOIN Inventory inv ON i.item_id = inv.item_id
GROUP BY i.item_id, i.item_name
HAVING total_inventory_quantity < 100;
```

	item_id	item_name	total_inventory_quantity
1	ITM002	Veggie Burger	50
2	ITM003	Chicken Sandwich	70
3	ITM009	Chocolate Shake	80