

Pizza sales dashboard

1. total revenue

The screenshot shows a SQL query editor with a tab labeled 'pizza'. The query is as follows:

```
1 SELECT
2     SUM(total_price) AS total_revenue
3 FROM
4     pizza;
```

Below the query editor, there is a toolbar with icons for saving, running, and other functions. A dropdown menu shows 'Limit to 1000 rows'. Below the toolbar, there is a 'Result Grid' section with a table showing the result of the query:

total_revenue
817860.0508384705

2. Avg. order value

The screenshot shows a SQL query editor with a tab labeled 'pizza'. The query is as follows:

```
1 SELECT
2     SUM(total_price) / COUNT(DISTINCT order_id) AS Avg_Order_value
3 FROM
4     pizza;
```

Below the query editor, there is a toolbar with icons for saving, running, and other functions. A dropdown menu shows 'Limit to 1000 rows'. Below the toolbar, there is a 'Result Grid' section with a table showing the result of the query:

Avg_Order_value
38.30726233435459

3.Total pizza sold

The screenshot shows a SQL IDE window with a tab labeled 'pizza'. The query editor contains the following SQL code:

```
1 SELECT
2     SUM(quantity) AS Total_pizza_sold
3 FROM
4     pizza;
5
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query:

Total_pizza_sold
49574

The interface includes a toolbar with various icons and a 'Limit to 1000 rows' dropdown menu.

4.Total order

The screenshot shows a SQL IDE window with a tab labeled 'pizza'. The query editor contains the following SQL code:

```
1 SELECT
2     count( DISTINCT order_id) as total_orders
3 FROM
4     pizza;
5
6
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query:

total_orders
21350

The interface includes a toolbar with various icons and a 'Limit to 1000 rows' dropdown menu.

5. Avg pizza

The screenshot shows a SQL IDE window titled 'pizza'. The query editor contains the following SQL code:

```
1 SELECT
2   cast(CAST(SUM(quantity) AS DECIMAL (10 , 2 )) / CAST(COUNT(DISTINCT order_id) AS DECIMAL (10 , 2 )) as decimal (10,2)) as Avg_pizza_per_order
3 FROM
4   pizza;
5
6
```

Below the query editor, the 'Result Grid' tab is active, displaying a single row of results:

Avg_pizza_per_order
2.32

6. Daily trends for pizza orders

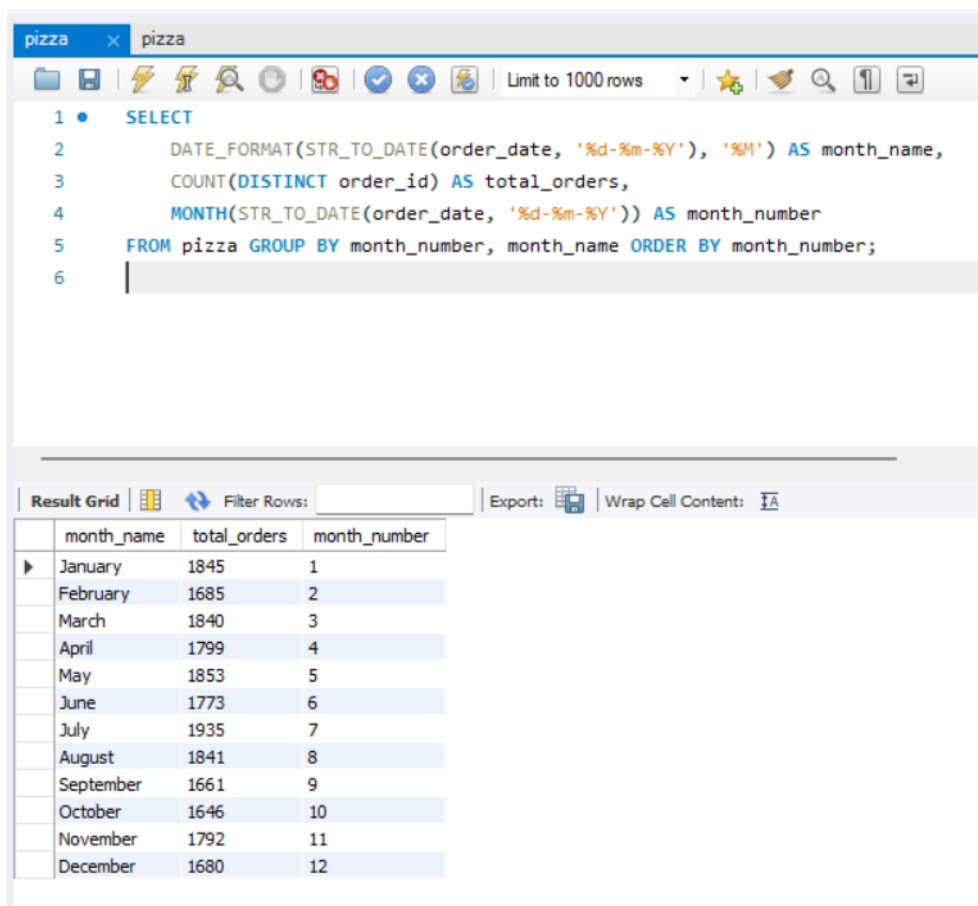
The screenshot shows a SQL IDE window titled 'pizza'. The query editor contains the following SQL code:

```
1 SELECT
2   DAYNAME(STR_TO_DATE(order_date, '%d-%m-%Y')) AS order_day,
3   COUNT(DISTINCT order_id) AS total_orders,
4   DAYOFWEEK(STR_TO_DATE(order_date, '%d-%m-%Y')) AS weekday_number
5 FROM pizza GROUP BY weekday_number, order_day ORDER BY weekday_number;
6
```

Below the query editor, the 'Result Grid' tab is active, displaying a table with three columns: 'order_day', 'total_orders', and 'weekday_number'.

order_day	total_orders	weekday_number
Sunday	2624	1
Monday	2794	2
Tuesday	2973	3
Wednesday	3024	4
Thursday	3239	5
Friday	3538	6
Saturday	3158	7

7. Monthly Trends for orders



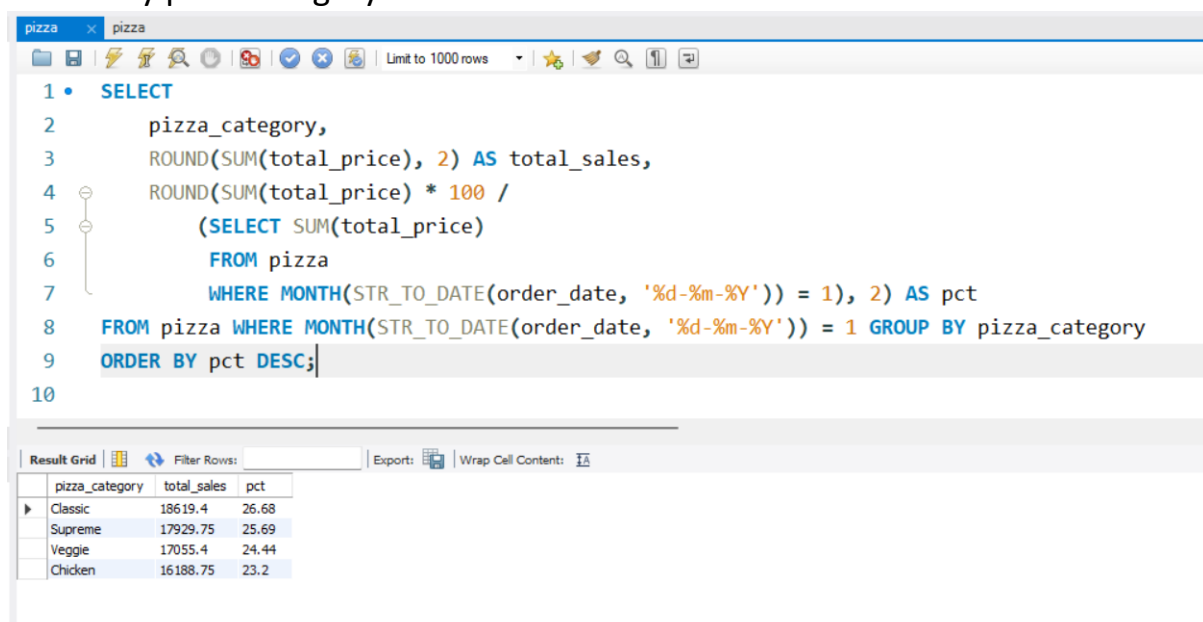
The screenshot shows a SQL IDE window titled 'pizza'. The query editor contains the following SQL code:

```
1 • SELECT
2     DATE_FORMAT(STR_TO_DATE(order_date, '%d-%m-%Y'), '%M') AS month_name,
3     COUNT(DISTINCT order_id) AS total_orders,
4     MONTH(STR_TO_DATE(order_date, '%d-%m-%Y')) AS month_number
5 FROM pizza GROUP BY month_number, month_name ORDER BY month_number;
6
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query. The table has four columns: month_name, total_orders, and month_number. The data is sorted by month_number.

month_name	total_orders	month_number
January	1845	1
February	1685	2
March	1840	3
April	1799	4
May	1853	5
June	1773	6
July	1935	7
August	1841	8
September	1661	9
October	1646	10
November	1792	11
December	1680	12

8. Sales by pizza category



The screenshot shows a SQL IDE window titled 'pizza'. The query editor contains the following SQL code:

```
1 • SELECT
2     pizza_category,
3     ROUND(SUM(total_price), 2) AS total_sales,
4     ROUND(SUM(total_price) * 100 /
5         (SELECT SUM(total_price)
6          FROM pizza
7          WHERE MONTH(STR_TO_DATE(order_date, '%d-%m-%Y')) = 1), 2) AS pct
8 FROM pizza WHERE MONTH(STR_TO_DATE(order_date, '%d-%m-%Y')) = 1 GROUP BY pizza_category
9 ORDER BY pct DESC;
10
```

Below the query editor, the 'Result Grid' tab is active, displaying the results of the query. The table has three columns: pizza_category, total_sales, and pct. The data is sorted by pct in descending order.

pizza_category	total_sales	pct
Classic	18619.4	26.68
Supreme	17929.75	25.69
Veggie	17055.4	24.44
Chicken	16188.75	23.2

9.Sales by pizza size

The screenshot shows a SQL IDE window titled 'pizza'. The query editor contains the following SQL code:

```
1 • SELECT
2     pizza_size,
3     ROUND(SUM(total_price), 2) AS total_sales,
4     ROUND(SUM(total_price) * 100 /
5         (SELECT SUM(total_price)
6             FROM pizza
7             WHERE MONTH(STR_TO_DATE(order_date, '%d-%m-%Y')) = 1), 2) AS pct
8 FROM pizza WHERE MONTH(STR_TO_DATE(order_date, '%d-%m-%Y')) = 1 GROUP BY pizza_size
9 ORDER BY pct DESC;
10
```

Below the query editor is the 'Result Grid' tab, which displays the following data:

	pizza_size	total_sales	pct
▶	L	32399.4	46.42
	M	20943.5	30.01
	S	15103.5	21.64
	XL	1275	1.83
	XXL	71.9	0.1

10.Bottom 5 pizza by Revenue

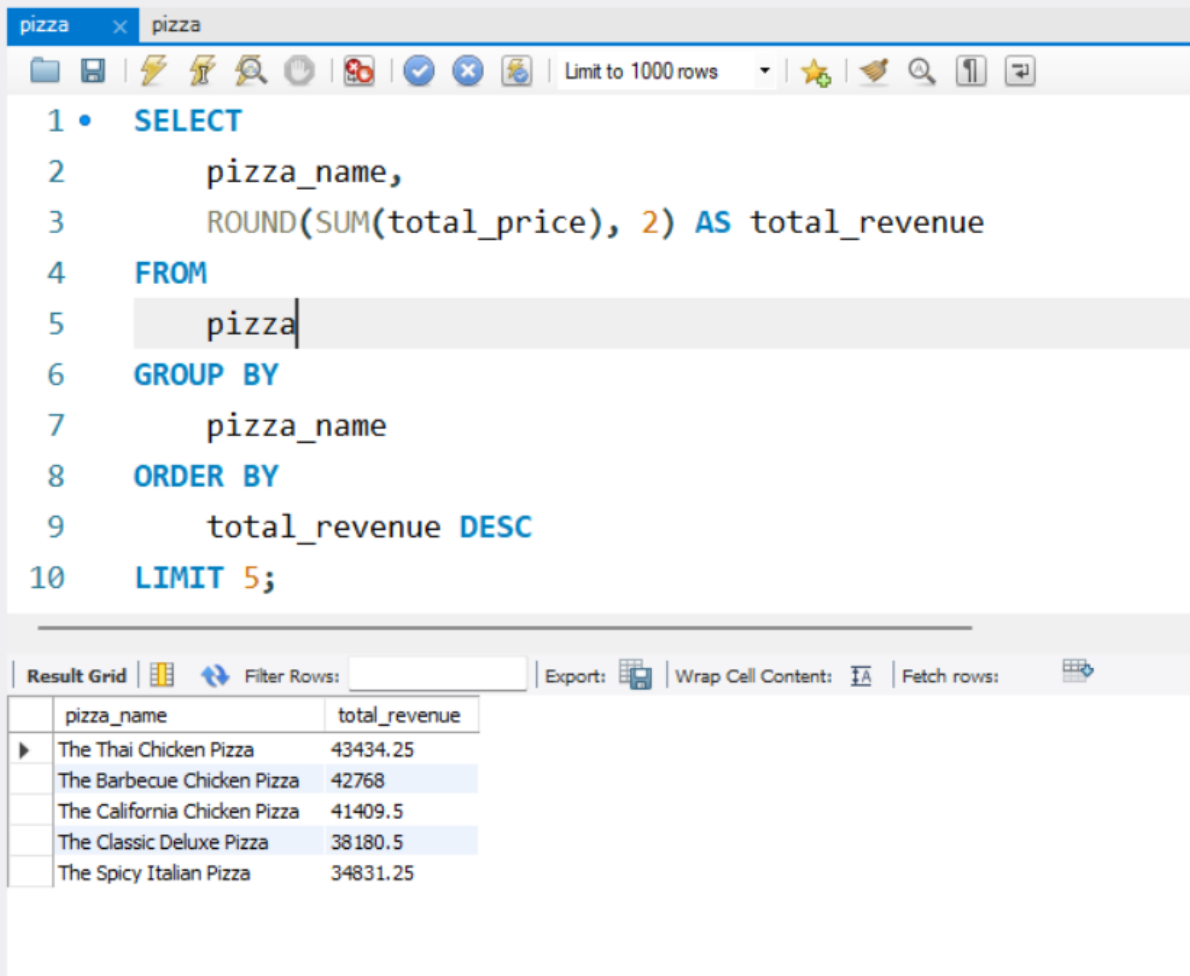
The screenshot shows a SQL IDE window titled 'pizza'. The query editor contains the following SQL code:

```
1 • SELECT
2     pizza_name,
3     ROUND(SUM(total_price), 2) AS total_revenue
4 FROM
5     pizza_sales
6 GROUP BY
7     pizza_name
8 ORDER BY
9     total_revenue ASC
10 LIMIT 5;
```

Below the query editor is the 'Result Grid' tab, which displays the following data:

	pizza_size	total_sales	pct
▶	L	32399.4	46.42
	M	20943.5	30.01
	S	15103.5	21.64
	XL	1275	1.83
	XXL	71.9	0.1

11. Top 5 pizza by Revenue



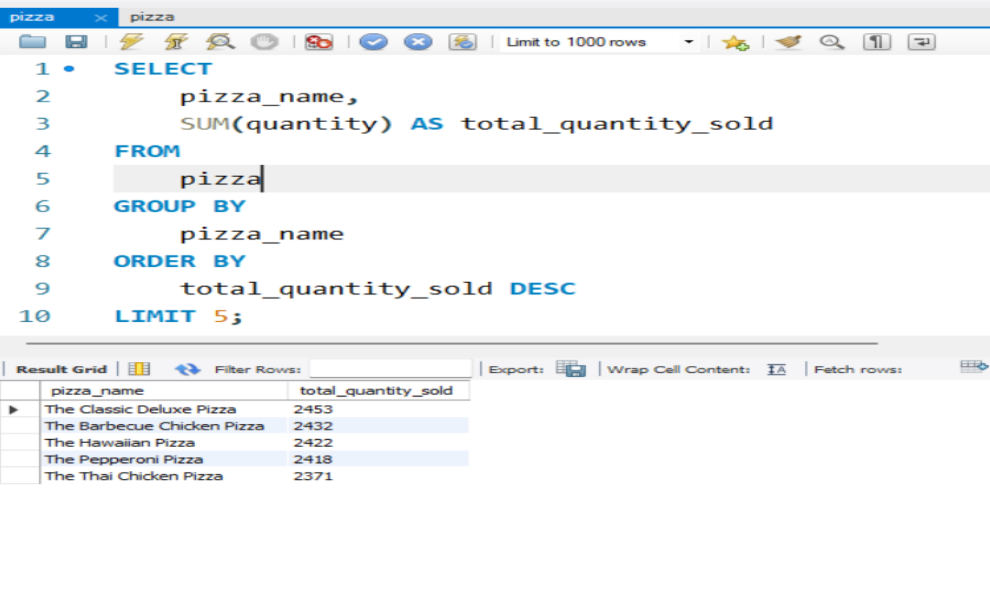
The screenshot shows a SQL query editor with a query to find the top 5 pizzas by revenue. The query is as follows:

```
1 • SELECT
2     pizza_name,
3     ROUND(SUM(total_price), 2) AS total_revenue
4 FROM
5     pizza
6 GROUP BY
7     pizza_name
8 ORDER BY
9     total_revenue DESC
10    LIMIT 5;
```

Below the query, the result grid shows the top 5 pizzas by revenue:

pizza_name	total_revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Spicy Italian Pizza	34831.25

12. Top 5 pizza by Quantity



The screenshot shows a SQL query editor with a query to find the top 5 pizzas by quantity. The query is as follows:

```
1 • SELECT
2     pizza_name,
3     SUM(quantity) AS total_quantity_sold
4 FROM
5     pizza
6 GROUP BY
7     pizza_name
8 ORDER BY
9     total_quantity_sold DESC
10    LIMIT 5;
```

Below the query, the result grid shows the top 5 pizzas by quantity:

pizza_name	total_quantity_sold
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371

13. . Bottom 5 pizza by Quantity

The screenshot shows a SQL IDE with a query editor and a result grid. The query is as follows:

```
1 • SELECT
2     pizza_name,
3     SUM(quantity) AS total_quantity_sold
4 FROM
5     pizza
6 GROUP BY
7     pizza_name
8 ORDER BY
9     total_quantity_sold ASC
10 LIMIT 5;
```

The result grid displays the following data:

pizza_name	total_quantity_sold
The Brie Carre Pizza	490
The Mediterranean Pizza	934
The Calabrese Pizza	937
The Spinach Supreme Pizza	950
The Soppressata Pizza	961

14.Top 5 pizza by total order

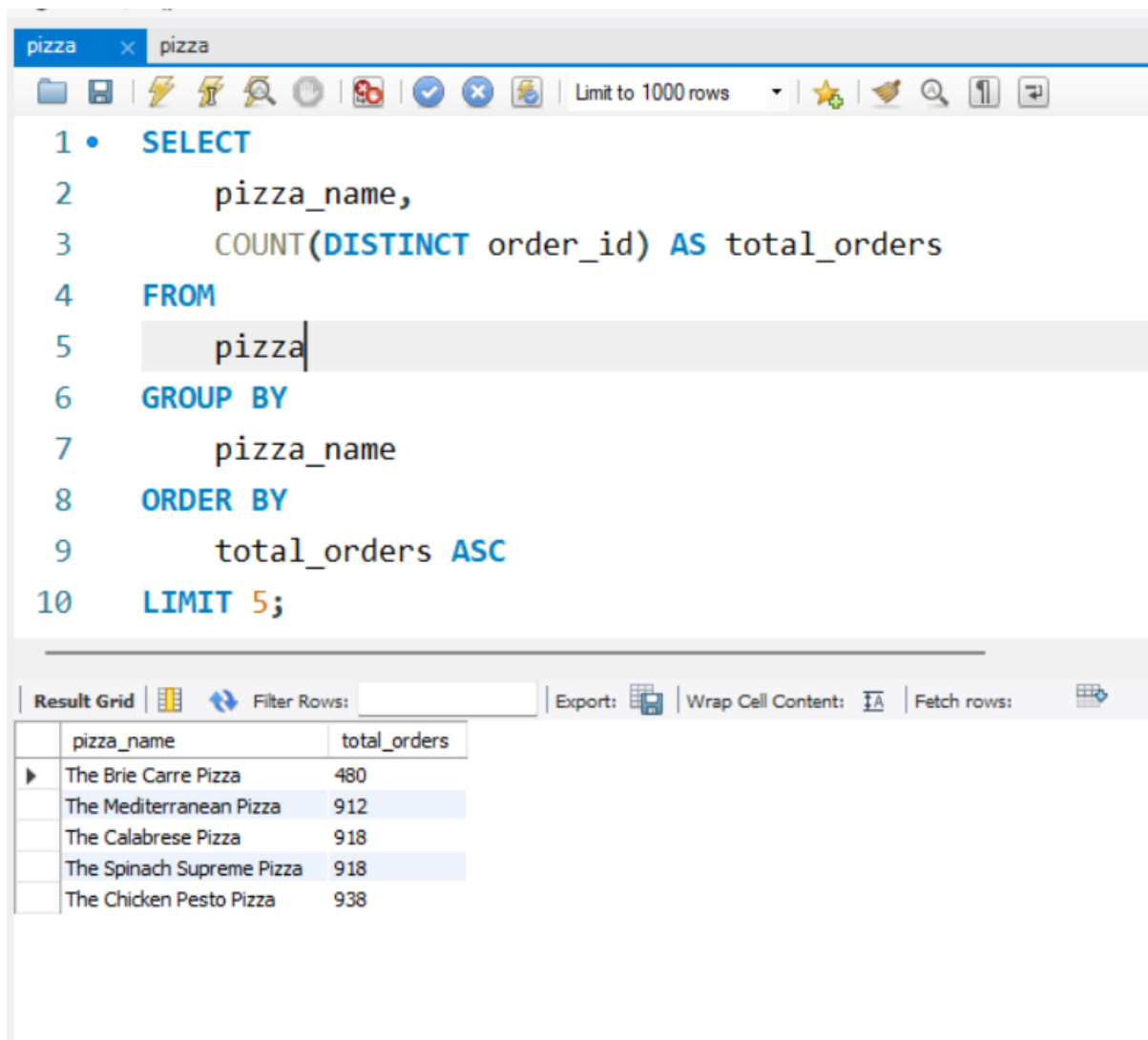
The screenshot shows a SQL IDE with a query editor and a result grid. The query is as follows:

```
1 • SELECT
2     pizza_name,
3     COUNT(DISTINCT order_id) AS total_orders
4 FROM
5     pizza
6 GROUP BY
7     pizza_name
8 ORDER BY
9     total_orders DESC
10 LIMIT 5;
```

The result grid displays the following data:

pizza_name	total_orders
The Classic Deluxe Pizza	2329
The Hawaiian Pizza	2280
The Pepperoni Pizza	2278
The Barbecue Chicken Pizza	2273
The Thai Chicken Pizza	2225

15. Bottom 5 pizza by total order



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor contains the following SQL code:

```
1 • SELECT
2     pizza_name,
3     COUNT(DISTINCT order_id) AS total_orders
4 FROM
5     pizza
6 GROUP BY
7     pizza_name
8 ORDER BY
9     total_orders ASC
10    LIMIT 5;
```

The results grid displays the following data:

	pizza_name	total_orders
▶	The Brie Carre Pizza	480
	The Mediterranean Pizza	912
	The Calabrese Pizza	918
	The Spinach Supreme Pizza	918
	The Chicken Pesto Pizza	938