

# PIZZA SALES ANALYSIS USING SQL



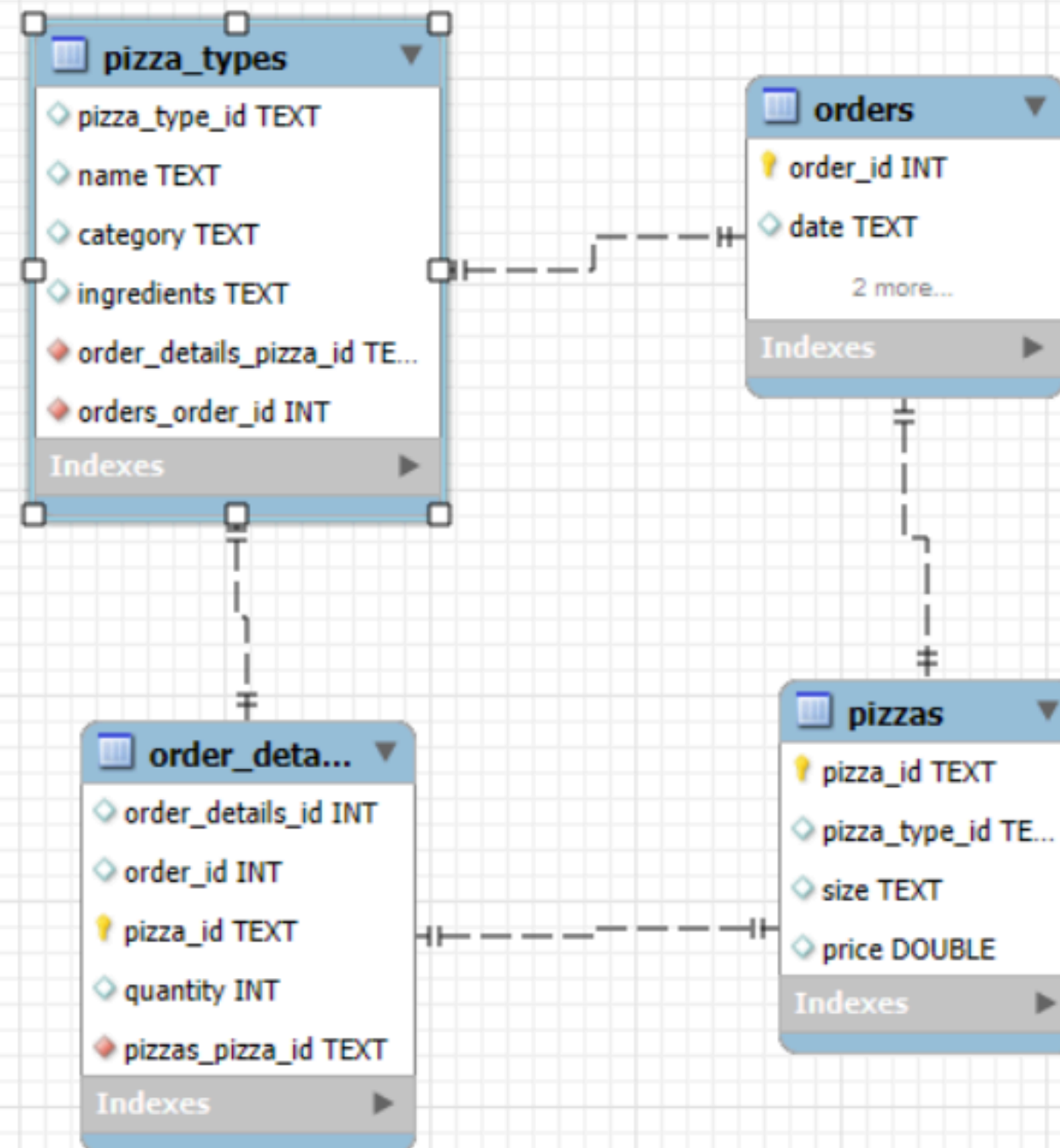


# OBJECTIVE

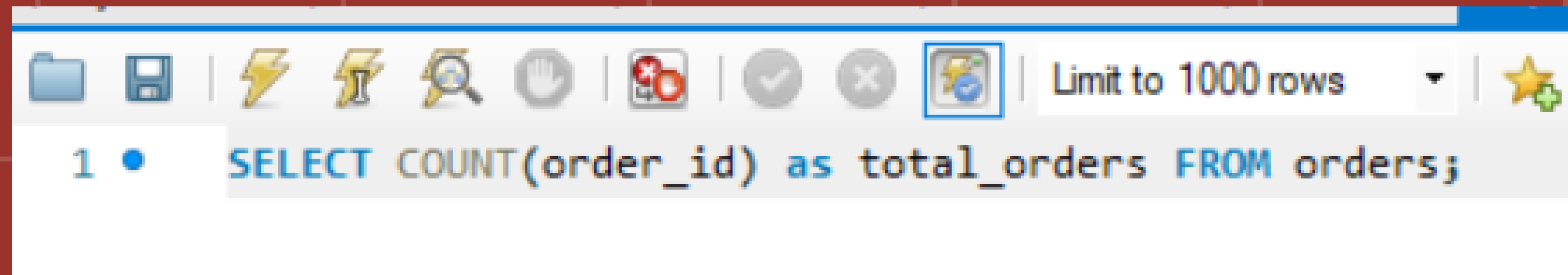
This project looks at pizza sales numbers to find out how well the restaurant is doing and what trends there are. It wants to use this information to make smarter decisions and plan better for the future.



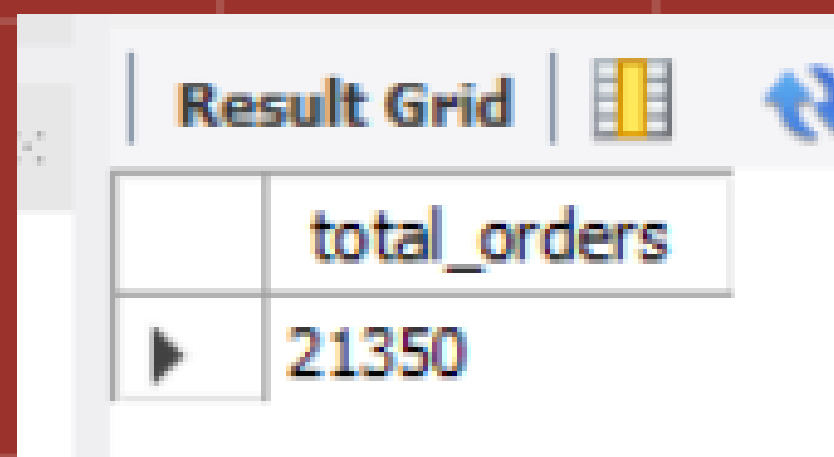
# DATA MODLE VIEW



Retrieve the total number of orders placed.



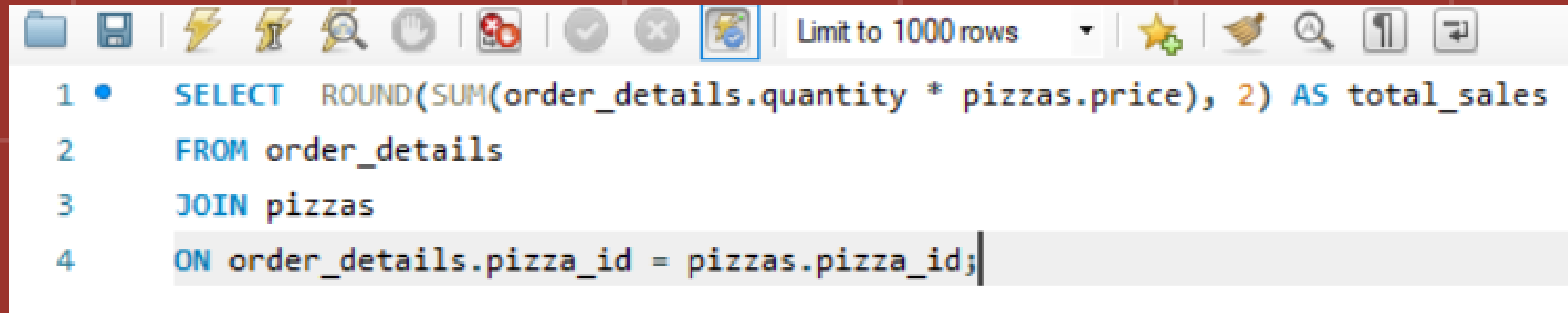
A screenshot of a SQL query editor window. The toolbar at the top includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The query text is: `1 • SELECT COUNT(order_id) as total_orders FROM orders;`



A screenshot of a 'Result Grid' window. It displays a single row with the column name 'total\_orders' and the value '21350'.

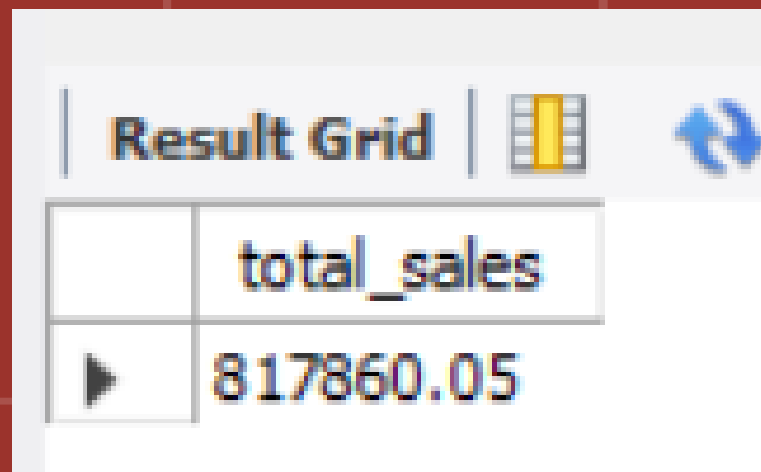
	total_orders
▶	21350

Calculate the total revenue generated from pizza sales.



A screenshot of a SQL query editor window. The window has a toolbar at the top with icons for file operations, execution, and search. The query text is as follows:

```
1 • SELECT ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_sales
2 FROM order_details
3 JOIN pizzas
4 ON order_details.pizza_id = pizzas.pizza_id;
```



A screenshot of a 'Result Grid' window. It displays a single row of results for the query. The column is labeled 'total\_sales' and the value is 817860.05.

	total_sales
▶	817860.05

# Identify the highest-priced pizza.

```
1  SELECT pizza_types.name , pizzas.price
2  FROM pizza_types
3  JOIN pizzas
4  ON pizza_types.pizza_type_id = pizzas.pizza_type_id
5  ORDER BY pizzas.price DESC LIMIT 1;
```

Result Grid

Filter Rows:

	name	price
▶	The Greek Pizza	35.95

# Identify the most common pizza size ordered.

```
1 • SELECT pizzas.size, COUNT(order_details.order_details_id) AS order_count
2 FROM pizzas
3 JOIN order_details
4 ON pizzas.pizza_id = order_details.pizza_id
5 GROUP BY pizzas.size ORDER BY order_count DESC ;
```

Result Grid			Filter Rows:
	size	order_count	
▶	L	18526	
	M	15385	
	S	14137	
	XL	544	
	XXL	28	

The background of the slide features two large, stylized pizza slices on the left and right sides. Each slice has a yellow-orange crust, a red tomato slice with black seeds, and a simple black smiley face. The background is a dark red color with a light red grid pattern.

# List the top 5 most ordered pizza types along with their quantities.

- ```
SELECT pizza_types.name, SUM(order_details.quantity) AS quantity
FROM pizza_types
JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC LIMIT 5;
```

| Result Grid |                            |          | Filter Rows: |
|-------------|----------------------------|----------|--------------|
|             | name                       | quantity |              |
| ▶           | The Classic Deluxe Pizza   | 2453     |              |
|             | The Barbecue Chicken Pizza | 2432     |              |
|             | The Hawaiian Pizza         | 2422     |              |
|             | The Pepperoni Pizza        | 2418     |              |
|             | The Thai Chicken Pizza     | 2371     |              |



Join the necessary tables to find the total quantity of each pizza category ordered.

```
SELECT pizza_types.category , SUM(order_details.quantity) AS quantity
FROM pizza_types
JOIN pizzas
ON pizzas.pizza_type_id = pizza_types.pizza_type_id
JOIN order_details
ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category ORDER BY quantity DESC;
```

| Result Grid |          |          | Filter Rows: |
|-------------|----------|----------|--------------|
|             | category | quantity |              |
| ▶           | Classic  | 14888    |              |
|             | Supreme  | 11987    |              |
|             | Veggie   | 11649    |              |
|             | Chicken  | 11050    |              |

# Determine the distribution of orders by hour of the day.

```
1 • SELECT HOUR(time) AS hour, COUNT(order_id) AS order_count
2   FROM orders
3   GROUP BY hour;
```

Result Grid

|   | hour | order_count |
|---|------|-------------|
| ▶ | 11   | 1231        |
|   | 12   | 2520        |
|   | 13   | 2455        |
|   | 14   | 1472        |
|   | 15   | 1468        |
|   | 16   | 1920        |
|   | 17   | 2336        |
|   | 18   | 2399        |
|   | 19   | 2009        |
|   | 20   | 1642        |
|   | 21   | 1198        |
|   | 22   | 663         |
|   | 23   | 28          |
|   | 10   | 8           |
|   | 9    | 1           |

Join relevant tables to find the category-wise distribution of pizzas.

```
1 • SELECT category , COUNT(name)
2   FROM pizza_types
3   GROUP BY category;
```

| Result Grid |          |             | Filter Rows: |
|-------------|----------|-------------|--------------|
|             | category | COUNT(name) |              |
| ▶           | Chicken  | 6           |              |
|             | Classic  | 8           |              |
|             | Supreme  | 9           |              |
|             | Veggie   | 9           |              |

Group the orders by date and calculate the average number of pizzas ordered per day.


```
1 • SELECT ROUND(AVG(quantity), 0) AS avg_pizza_ordered_per_day FROM
2 (SELECT orders.date, SUM(order_details.quantity) AS quantity
3  FROM orders JOIN order_details
4  ON orders.order_id = order_details.order_id
5  GROUP BY orders.date ) AS order_quantity
```

| Result Grid |                           | Filter Rows: |
|-------------|---------------------------|--------------|
|             | avg_pizza_ordered_per_day |              |
| ▶           | 138                       |              |



Determine the top 3 most ordered pizza types based on revenue.

- ```
SELECT pizza_types.name,  
SUM(order_details.quantity * pizzas.price) AS revenue  
FROM pizza_types JOIN pizzas  
ON pizzas.pizza_type_id = pizza_types.pizza_type_id  
JOIN order_details  
ON order_details.pizza_id = pizzas.pizza_id  
GROUP BY pizza_types.name ORDER BY revenue DESC LIMIT 3;
```



Result Grid			Filter Rows:
	name	revenue	
▶	The Thai Chicken Pizza	43434.25	
	The Barbecue Chicken Pizza	42768	
	The California Chicken Pizza	41409.5	


Calculate the percentage contribution of each pizza type to total revenue.

```
1 • SELECT pizza_types.category,  
2   ROUND(SUM(order_details.quantity * pizzas.price) /(  
3     SELECT ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_sales  
4     FROM order_details  
5     JOIN pizzas  
6     ON order_details.pizza_id = pizzas.pizza_id  
7   ) * 100 , 2) AS revenue  
8 FROM pizza_types JOIN pizzas  
9 ON pizzas.pizza_type_id = pizza_types.pizza_type_id  
10 JOIN order_details  
11 ON order_details.pizza_id = pizzas.pizza_id  
12 GROUP BY pizza_types.category ORDER BY revenue DESC;
```

Result Grid			Filter Rows:
	category	revenue	
▶	Classic	26.91	
	Supreme	25.46	
	Chicken	23.96	
	Veggie	23.68	

# Analyze the cumulative revenue generated over time.

```
1 • select date,  
2    sum(revenue) over (order by date) as cum_revenue  
3    from  
4    (select orders.date,  
5     sum(order_details.quantity * pizzas.price) as revenue  
6     from order_details join pizzas  
7     on order_details.pizza_id = pizzas.pizza_id  
8     join orders  
9     on orders.order_id = order_details.order_id  
10    group by orders.date) as sales;
```

Result Grid    Filter Rows: <input type="text"/>		
	date	cum_revenue
▶	2015-01-01	2713.8500000000004
	2015-01-02	5445.75
	2015-01-03	8108.15
	2015-01-04	9863.6
	2015-01-05	11929.55
	2015-01-06	14358.5
	2015-01-07	16560.7
	2015-01-08	19399.05
	2015-01-09	21526.4
	2015-01-10	23990.350000000002
	2015-01-11	25862.65
	2015-01-12	27781.7
	2015-01-13	29831.300000000003

Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
1 SELECT name, revenue FROM
2 (SELECT category, name, revenue,
3  rank() OVER(PARTITION BY category ORDER BY revenue DESC ) AS rn
4  FROM
5  (SELECT pizza_types.category , pizza_types.name,
6   SUM(order_details.quantity * pizzas.price) AS revenue
7   FROM pizza_types JOIN pizzas
8   ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9   JOIN order_details
10  ON order_details.pizza_id = pizzas.pizza_id
11  GROUP BY pizza_types.category, pizza_types.name ) AS a) AS b
12 WHERE rn <= 3;
```

Result Grid			Filter Rows:	Export
	name	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 Hawaiian Pizza	32273.25		
	The Pepperoni Pizza	30161.75		
	The Spicy Italian Pizza	34831.25		
	The Italian Supreme Pizza	33476.75		
	The Sicilian Pizza	30940.5		
	The Four Cheese Pizza	32265.70000000065		
	The Mexicana Pizza	26780.75		
	The Five Cheese Pizza	26066.5		





Thank You