



PIZZA SALES ANALYSIS

Objective

A purpose of this project is to examine the data set with SQL and understand business of pizza sales growth by answering simple questions.

Q.1 Retrieve the total number of orders placed.

```
3 •     SELECT  
4         COUNT(order_id) AS total_orders  
5     FROM  
6         orders;
```

The screenshot shows a MySQL query editor interface. At the top, there is a code editor window containing the SQL query. Below the code editor is a toolbar with various buttons: 'Result Grid' (selected), 'Filter Rows' (with an input field), 'Export' (with a grid icon), and 'W'. The main area displays the results of the query in a table format. The table has two columns: an empty column on the left and a column labeled 'total_orders' on the right. The single row contains the value '21350'.

	total_orders
▶	21350

Q.2 Calculate the total revenue generated from pizza sales.

```
3   ●    SELECT
4   ⚋      ROUND(SUM(order_details.quantity * pizzas.price),
5                   2) AS total_revenue
6   FROM
7   order_details
8   JOIN
9   pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

The screenshot shows the MySQL Workbench interface with the results of the executed SQL query. The results are displayed in a table with one row, showing the total revenue.

	total_revenue
▶	817860.05

Below the table, there are several buttons: 'Result Grid' (selected), 'Filter Rows:', 'Export:', and 'Wrap Cell Content:'.

Q.3 Identify the highest-priced pizza.

```
3 • select pizza_types.name, pizzas.price  
4   from pizza_types join pizzas  
5     on pizza_types.pizza_type_id = pizzas.pizza_type_id  
6   order by pizzas.price desc  
7   limit 1;
```

< []

Result Grid | Filter Rows: Export: Wrap Cell Content:

	name	price
▶	The Greek Pizza	35.95

Q.4 Identify the most common pizza size ordered.

```
3 •   SELECT pizzas.size,  
4       COUNT(order_details.order_details_id) AS order_count  
5       FROM pizzas JOIN order_details  
6       ON pizzas.pizza_id = order_details.pizza_id  
7       GROUP BY pizzas.size  
8       ORDER BY order_count DESC;  
9
```

Result Grid | Filter Rows: Export: Wrap Cell Content:

	size	order_count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28

Q.5 List the top 5 most ordered pizza type along with their quantities.

```
4 •     SELECT pizza_types.name,  
5         SUM(order_details.quantity) AS total_quantity  
6     FROM pizza_types JOIN pizzas  
7     ON pizza_types.pizza_type_id = pizzas.pizza_type_id  
8     JOIN order_details  
9     ON order_details.pizza_id = pizzas.pizza_id  
10    GROUP BY pizza_types.name  
11    ORDER BY total_quantity DESC  
12    LIMIT 5;
```

	name	total_quantity
	The Classic Deluxe Pizza	2453
	The Barbecue Chicken Pizza	2432
	The Hawaiian Pizza	2422
	The Pepperoni Pizza	2418
	The Thai Chicken Pizza	2371

Q.6 Join the necessary tables to find the total quantity of each pizza category ordered.

```
4 •   SELECT pizza_types.category,  
5       SUM(order_details.quantity) AS total_quantity  
6   FROM pizza_types JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id  
7   JOIN order_details ON order_details.pizza_id = pizzas.pizza_id  
8   GROUP BY pizza_types.category  
9   ORDER BY total_quantity DESC;
```

Result Grid | Filter Rows:

	category	total_quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Q.7 Determine the distribution of orders by hour of the day.

```
3 •     SELECT  
4         HOUR(order_time) AS hour, COUNT(order_id) AS order_count  
5     FROM  
6     orders  
7     GROUP BY HOUR(order_time);
```

Result Grid | Filter

	hour	order_count
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336
	18	2399
	19	2009

Result 1 ×

Q.8 Join relevant tables to find the category-wise distribution of pizzas.

```
3 •      SELECT  
4          category, COUNT(name)  
5      FROM  
6          pizza_types  
7      GROUP BY category;
```

The screenshot shows a MySQL Workbench interface. At the top, there is a code editor window containing the SQL query. Below it is a toolbar with buttons for 'Result Grid' (selected), 'Grid View', 'Edit Rows', and 'Filter Rows'. The main area displays the results of the query in a tabular format.

	category	COUNT(name)
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

Q.9 Group the orders by date and calculate the average number of pizzas ordered per day.

```
4 • select round(avg(quantity), 0) from
5   (select orders.order_date, sum(order_details.quantity) as quantity
6     from orders join order_details
7       on orders.order_id = order_details.order_id
8     group by orders.order_date) as order_quantity;
```



Result Grid | Filter Rows: Export: Wrap Cell Content:

	round(avg(quantity), 0)
▶	138

Q.10 Determine the top 3 most ordered pizza types based on revenue.

```
3 • select pizza_types.name,  
4     sum(order_details.quantity * pizzas.price) as revenue  
5     from pizza_types join pizzas on pizzas.pizza_type_id = pizza_types.pizza_type_id  
6     join order_details  
7     on order_details.pizza_id = pizzas.pizza_id  
8     group by pizza_types.name  
9     order by revenue desc  
10    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

Q.11 Calculate the percentage contribution of each pizza type to total revenue.

```
3 •    select pizza_types.category,  
4     round(sum(order_details.quantity * pizzas.price) / (select round(sum(order_details.quantity * pizzas.price),  
5          2) as total_sales  
6      from order_details join pizzas  
7      on pizzas.pizza_id = order_details.pizza_id)*100,2) as revenue  
8      from pizza_types join pizzas  
9      on pizza_types.pizza_type_id = pizzas.pizza_type_id  
10     join order_details  
11     on order_details.pizza_id = pizzas.pizza_id  
12     group by pizza_types.category  
13     order by revenue desc;
```

Result Grid | Filter F

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

Q.12 Analyze the cumulative revenue generated over time.

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

	order_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

Q.13 Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
3 •    select name, revenue from
4     (select category, name, revenue,
5      rank() over(partition by category order by revenue desc) as rn from
6     (select pizza_types.category, pizza_types.name,
7      sum(order_details.quantity * pizzas.price) as revenue
8     from pizza_types join pizzas
9     on pizza_types.pizza_type_id = pizzas.pizza_type_id
10    join order_details
11    on order_details.pizza_id = pizzas.pizza_id
12   group by pizza_types.category, pizza_types.name) as a) as b
13 where rn <= 3;
```

Result Grid		Filter Rows:
	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

Thank You