

PIZZA SALES ANALYSIS USING SQL



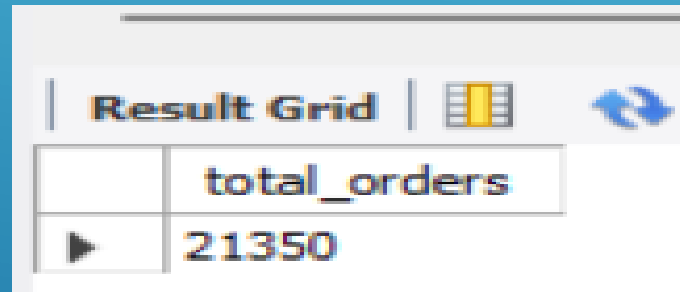
```
-- Retrieve the total number of orders placed.
```

```
SELECT
```

```
    COUNT(order_id) AS total_orders
```

```
FROM
```

```
    orders;
```



A screenshot of a database application's 'Result Grid' window. The window has a title bar and a toolbar with icons for grid view, a refresh button, and a sort button. The grid contains one column named 'total_orders' and one row with the value '21350'.

	total_orders
▶	21350

```
-- Calculate the total revenue generated from pizza sales.
```

```
SELECT
```

```
    ROUND(SUM(orders_details.quantity * pizzas.price),  
           2) AS total_sales
```

```
FROM
```

```
    orders_details
```

```
    JOIN
```

```
    pizzas ON pizzas.pizza_id = orders_details.pizza_id
```

Result Grid	
	total_sales
▶	817860.05

```
-- Identify the highest-priced pizza.
```

```
SELECT
```

```
    pizza_types.name, pizzas.price
```

```
FROM
```

```
    pizza_types
```

```
    JOIN
```

```
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
```

```
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.
```

```
SELECT
```

```
    pizzas.size,
```

```
    COUNT(orders_details.order_details_id) AS order_count
```

```
FROM
```

```
    pizzas
```

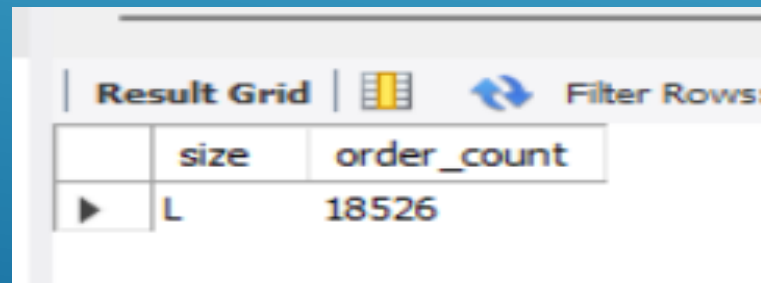
```
    JOIN
```

```
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
```

```
GROUP BY pizzas.size
```

```
ORDER BY order_count DESC
```

```
LIMIT 1;
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid has two columns: 'size' and 'order_count'. The first row shows 'L' with an order count of 18526. There are icons for a table structure and a refresh button above the grid, and a 'Filter Rows' label to the right.

	size	order_count
▶	L	18526

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

SELECT
    pizza_types.name, SUM(orders_details.quantity) AS quantity
FROM
    pizza_types
    JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
    orders_details ON orders_details.pizza_id = pizzas.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(orders_details.quantity) AS quantity
FROM
    pizza_types
    JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
    orders_details ON orders_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.
```

```
SELECT
```

```
    HOUR(order_time) AS hour, COUNT(order_id) AS order_count
```

```
FROM
```

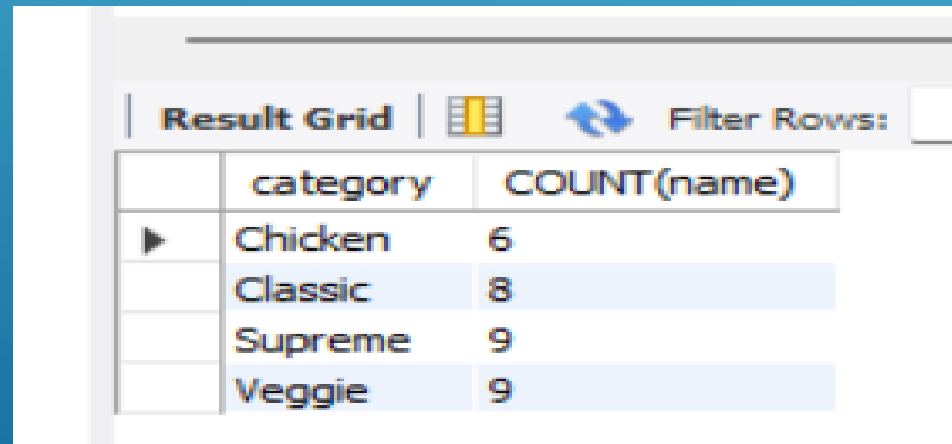
```
    orders
```

```
GROUP BY HOUR(order_time);
```

Result Grid			Filter Rows:
	hour	order_count	
▶	11	1231	
	12	2520	
	13	2455	
	14	1472	
	15	1468	
	16	1920	


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

```
SELECT  
    category, COUNT(name)  
FROM  
    pizza_types  
GROUP BY category
```

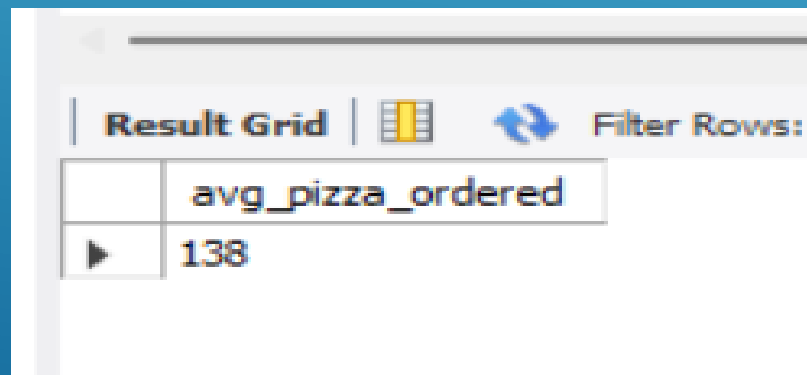


The screenshot shows a database interface with a 'Result Grid' tab. The grid displays the results of a SQL query, showing the category and the count of pizzas for each category. The categories are Chicken, Classic, Supreme, and Veggie. The counts are 6, 8, 9, and 9 respectively. The interface includes a 'Filter Rows' button and a 'Result Grid' tab.

	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.

SELECT
    ROUND(AVG(quantity), 0) AS avg_pizza_ordered
FROM
    (SELECT
        orders.order_date, SUM(orders_details.quantity) AS quantity
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```



The screenshot shows a database interface with a 'Result Grid' tab. The grid contains one column named 'avg_pizza_ordered' and one row with the value '138'. There are icons for grid view, refresh, and a filter row button.

	avg_pizza_ordered
▶	138

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

```
SELECT
```

```
    pizza_types.name,
```

```
    SUM(orders_details.quantity * pizzas.price) AS revenue
```

```
FROM
```

```
    pizza_types
```

```
    JOIN
```

```
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
```

```
    JOIN
```

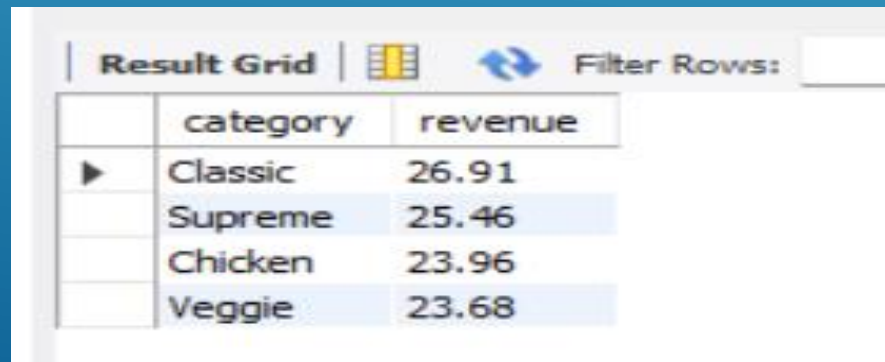
```
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
```

```
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.

select pizza_types.category,
round((sum(orders_details.quantity*pizzas.price) / (SELECT
    ROUND(SUM(orders_details.quantity * pizzas.price),
        2) AS total_sales
FROM
    orders_details
    JOIN
        pizzas ON pizzas.pizza_id = orders_details.pizza_id) ) * 100,2) as revenue
from pizza_types join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
join orders_details on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.category order by revenue DESC;
```



The screenshot shows a 'Result Grid' window with a table of pizza categories and their revenue. The table has two columns: 'category' and 'revenue'. The rows are ordered by revenue in descending order. The categories are Classic, Supreme, Chicken, and Veggie. The revenue values are 26.91, 25.46, 23.96, and 23.68 respectively. There is a 'Filter Rows' button and a 'Result Grid' tab at the top of the window.

	category	revenue
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

```
-- Analyze the cumulative revenue generated over time.
```

```
select order_date,  
sum(revenue) over(order by order_date) as cumulative_revenue  
from
```

```
(select orders.order_date,  
sum(orders_details.quantity * pizzas.price) as revenue  
from orders_details join pizzas  
on orders_details.pizza_id = pizzas.pizza_id  
join orders  
on orders.order_id = orders_details.order_id  
group by orders.order_date) as sales;
```

Result Grid			Filter Rows:
	order_date	cumulative_revenue	
▶	2015-01-01	2713.85000000000004	
	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	

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

```
select name, revenue from
```

```
(select category, name, revenue,  
rank() over(partition by category order by revenue desc) as ranks  
from
```

```
(select pizza_types.category, pizza_types.name,  
sum(orders_details.quantity * pizzas.price) as revenue  
from pizza_types join pizzas
```

```
on pizza_types.pizza_type_id = pizzas.pizza_type_id  
join orders_details
```

```
on orders_details.pizza_id = pizzas.pizza_id
```

```
group by pizza_types.category, pizza_types.name) as a) as b
```

```
where ranks <= 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		