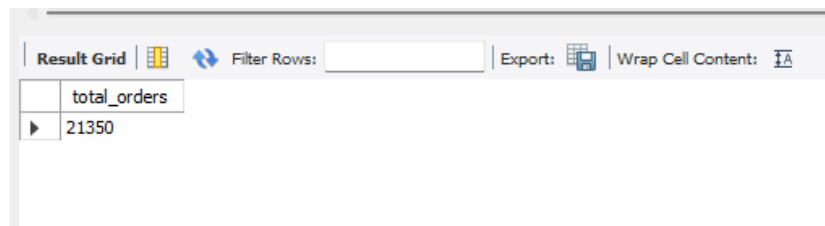


Pizza Sales Project

1) Retrieve the total number of orders placed.

```
SELECT COUNT(order_id) as total_orders from orders;
```



The screenshot shows a database query result grid. The header row is labeled 'total_orders'. The first data row shows the value '21350'. The interface includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' option.

total_orders
21350

2) Calculate the total revenue generated from pizza sales.

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



The screenshot shows a database query result grid. The header row is labeled 'total_sales'. The first data row shows the value '1635720.1'. The interface includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' option.

total_sales
1635720.1

3) 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;

```

The screenshot shows a database interface with a 'Result Grid' tab. It includes a 'Filter Rows' search bar, an 'Export' button, and a 'Wrap Cells' option. The table has two columns: 'name' and 'price'. The first row is 'The Greek Pizza' with a price of 35.95.

	name	price
▶	The Greek Pizza	35.95

4) Identify the most common pizza size ordered.

```

SELECT
    pizzas.size,
    COUNT(order_details.order_details_id) AS order_count
FROM
    pizzas
JOIN
    order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;

```

The screenshot shows a database interface with a 'Result Grid' tab. It includes a 'Filter Rows' search bar, an 'Export' button, and a 'Wrap Cells' option. The table has two columns: 'size' and 'order_count'. The rows are ordered by count in descending order: L (37052), M (30770), S (28274), XL (1088), and XXL (56).



	size	order_count
▶	L	37052
	M	30770
	S	28274
	XL	1088
	XXL	56

5) 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 order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;

```

Result Grid  Filter Rows: <input type="text"/> Export:  Wrap Cell Center		
	name	quantity
▶	The Classic Deluxe Pizza	4906
	The Barbecue Chicken Pizza	4864
	The Hawaiian Pizza	4844
	The Pepperoni Pizza	4836
	The Thai Chicken Pizza	4742

6) 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 pizza_types.pizza_type_id = pizzas.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:	Export:	Wrap Cell Co
	category	quantity			
▶	Classic	29776			
	Supreme	23974			
	Veggie	23298			
	Chicken	22100			

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

```

SELECT
    HOUR(time) AS hour,
    COUNT(order_id) AS order_count
FROM orders
GROUP BY HOUR(time);

```

Result Grid			Filter Rows:	Export:	Wrap Cell Content:
	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			

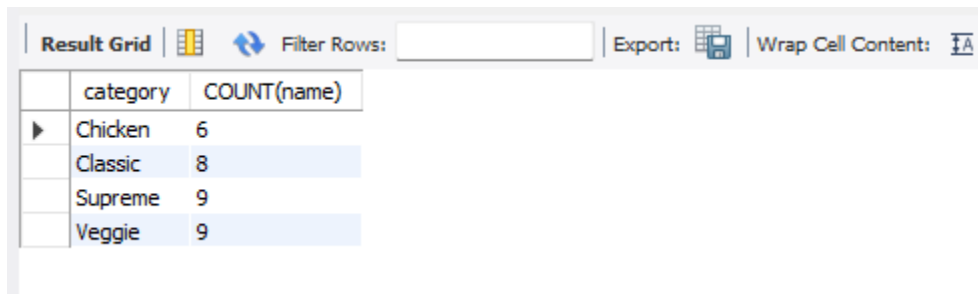
8) Join relevant tables to find the category-wise distribution of pizzas.

```

SELECT
    category, COUNT(name)
FROM

```

pizza_types
GROUP BY category;



The screenshot shows a database query result grid with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. The table has two columns: 'category' and 'COUNT(name)'. The data rows are: Chicken (6), Classic (8), Supreme (9), and Veggie (9).

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

- 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_per_day
FROM (
    SELECT orders.date, SUM(order_details.quantity) AS quantity
    FROM orders
    JOIN order_details
    ON orders.order_id = order_details.order_id
    GROUP BY orders.date
) AS order_quantity;
```



The screenshot shows a database query result grid with a toolbar at the top containing 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content'. The table has one column: 'avg_pizza_ordered_per_day'. The data row shows the value 277.

avg_pizza_ordered_per_day
277

- 10) 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 = pizzas.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:	
	Export:	Wrap Cell Content:
	name	revenue
▶	The Vegetables + Vegetables Pizza	1635720.0999999042
	The Spinach and Feta Pizza	1635720.0999999042
	The Spinach Pesto Pizza	1635720.0999999042

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

```

SELECT pizza_types.category,
(SUM(order_details.quantity*pizzas.price) / (select
ROUND(SUM(order_details.quantity * pizzas.price), 2) AS total_sales
FROM order_details
JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id) ) * 100 as
revenue
from pizza_types JOIN pizzas
ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details
ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category order by revenue desc;

```




Result Grid		
	Filter Rows:	
	Export:	Wrap Cell Content:
	category	revenue
▶	Classic	26.90596025566968
	Supreme	25.456311260099167
	Chicken	23.955137556847287
	Veggie	23.6825909273853

12) Analyze the cumulative revenue generated over time.

```

SELECT
    date,
    SUM(revenue) OVER (ORDER BY date) AS cum_revenue
FROM (
    SELECT
        orders.date,
        SUM(order_details.quantity * pizzas.price) AS revenue
    FROM order_details
    JOIN pizzas
        ON order_details.pizza_id = pizzas.pizza_id
    JOIN orders
        ON orders.order_id = order_details.order_id
    GROUP BY orders.date
) AS sales;

```

Result Grid  Filter Rows: <input type="text"/> Export:  Wrap Cell Content: 		
	date	cum_revenue
	2015-05-12	603658.3
	2015-05-13	608181.9
	2015-05-14	613570.9
	2015-05-15	620343.2000000001
	2015-05-16	624905.4
	2015-05-17	628562.2000000001
	2015-05-18	632981.5000000001
	2015-05-19	636955.5000000001

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

```

SELECT
    pizza_types.category,
    ROUND(
        SUM(order_details.quantity * pizzas.price) /
        (
            SELECT SUM(order_details.quantity * pizzas.price)

```

```

FROM order_details
JOIN pizzas
    ON pizzas.pizza_id = order_details.pizza_id
) * 100,
2) AS revenue
FROM pizza_types
JOIN pizzas
    ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN order_details
    ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;

```

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