

# Level: Basic

**Query 1:** Retrieve the total number of orders placed.

```
SELECT
    COUNT(order_id) AS total_orders
FROM
    orders;
```

Result Grid	
	total_orders
▶	21350

**Query 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
```

Result Grid	
	total_sales
▶	817860.05

**Query 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;
```

Result Grid		
	name	price
▶	The Greek Pizza	35.95

**Query 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;
```

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

**Query 5:** List the top 5 most ordered pizza types along with their quantities.

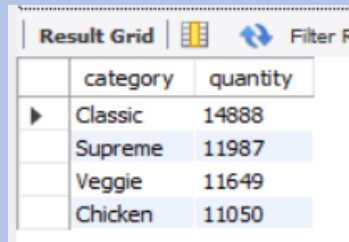
```
SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
FROM
    pizza_types
    JOIN
        pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN
        order_details ON order_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;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

	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

## Level: Intermediate

**Query 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;
```

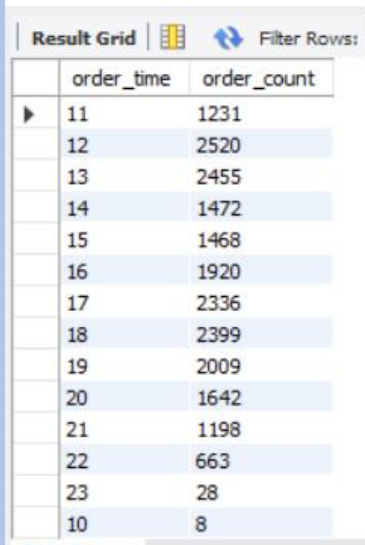


The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays the results of the SQL query for Query 6. The columns are 'category' and 'quantity'. The data is sorted in descending order of quantity. The categories and their corresponding quantities are: Classic (14888), Supreme (11987), Veggie (11649), and Chicken (11050).

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

**Query 7:** Determine the distribution of orders by hour of the day.

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



The screenshot shows a database interface with a 'Result Grid' tab selected. The grid displays the results of the SQL query for Query 7. The columns are 'order\_time' and 'order\_count'. The data shows the number of orders for each hour of the day. The hours and their corresponding order counts are: 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), and 10 (8).

	order_time	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

**Query 8:** Join relevant tables to find the category-wise distribution of pizzas.

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

	category	name_count
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

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

```
SELECT
    ROUND(AVG(quantity), 0)
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;
```

	ROUND(AVG(quantity), 0)
▶	138

**Query 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 = 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;
```

	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5

## Level: Advanced

**Query 10:** Calculate the percentage contribution of each pizza type to total revenue.

```
SELECT
    pizza_types.category,
    ROUND((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,
    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
	category	revenue	
▶	Classic	26.91	
	Supreme	25.46	
	Chicken	23.96	
	Veggie	23.68	

**Query 12:** Analyze the cumulative revenue generated over time.

```
select `date`,
sum(revenue) over(order by `date`) as cum_rvenue
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:
	date	cum_rvenue	
▶	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	
	2015-01-14	32358.700000000004	
	2015-01-15	34343.500000000001	
	2015-01-16	36937.650000000001	
	2015-01-17	39001.750000000001	
	2015-01-18	40978.600000000006	
	2015-01-19	43365.750000000001	
	2015-01-20	45763.650000000001	
	2015-01-21	47804.200000000001	
	2015-01-22	50300.900000000001	
	2015-01-23	52774.600000000006	

**Query 13:** 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 rn
from
(select pizza_types.category, pizza_types.name,
sum((order_details.quantity) * pizzas.price) 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, pizza_types.name) as a) as b
where rn <= 3;
```

Result Grid			Filter Rows:	Expo
	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		