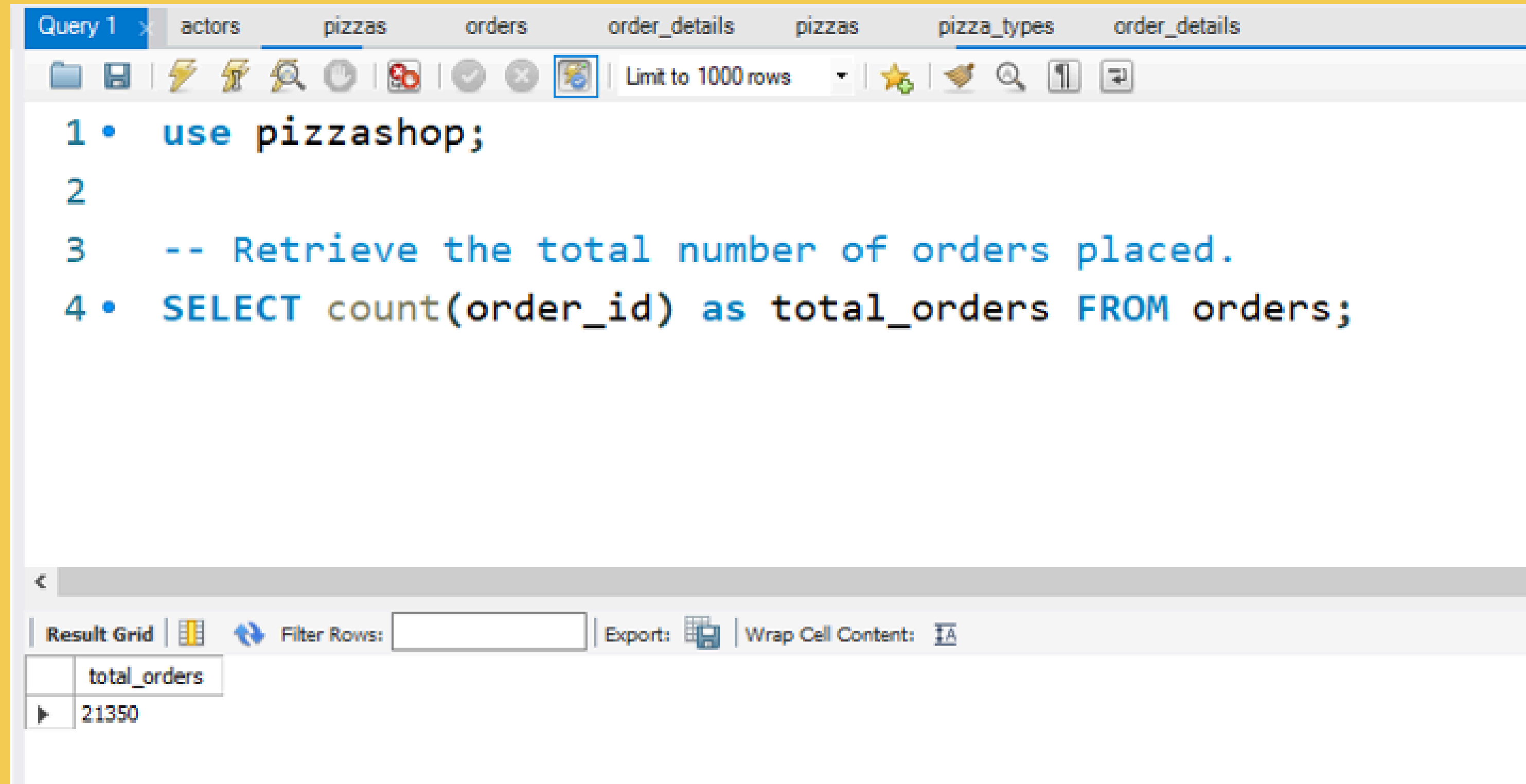


PIZZA SHOP DATA ANALYTICS



SQL

Retrieve the total number of orders placed.



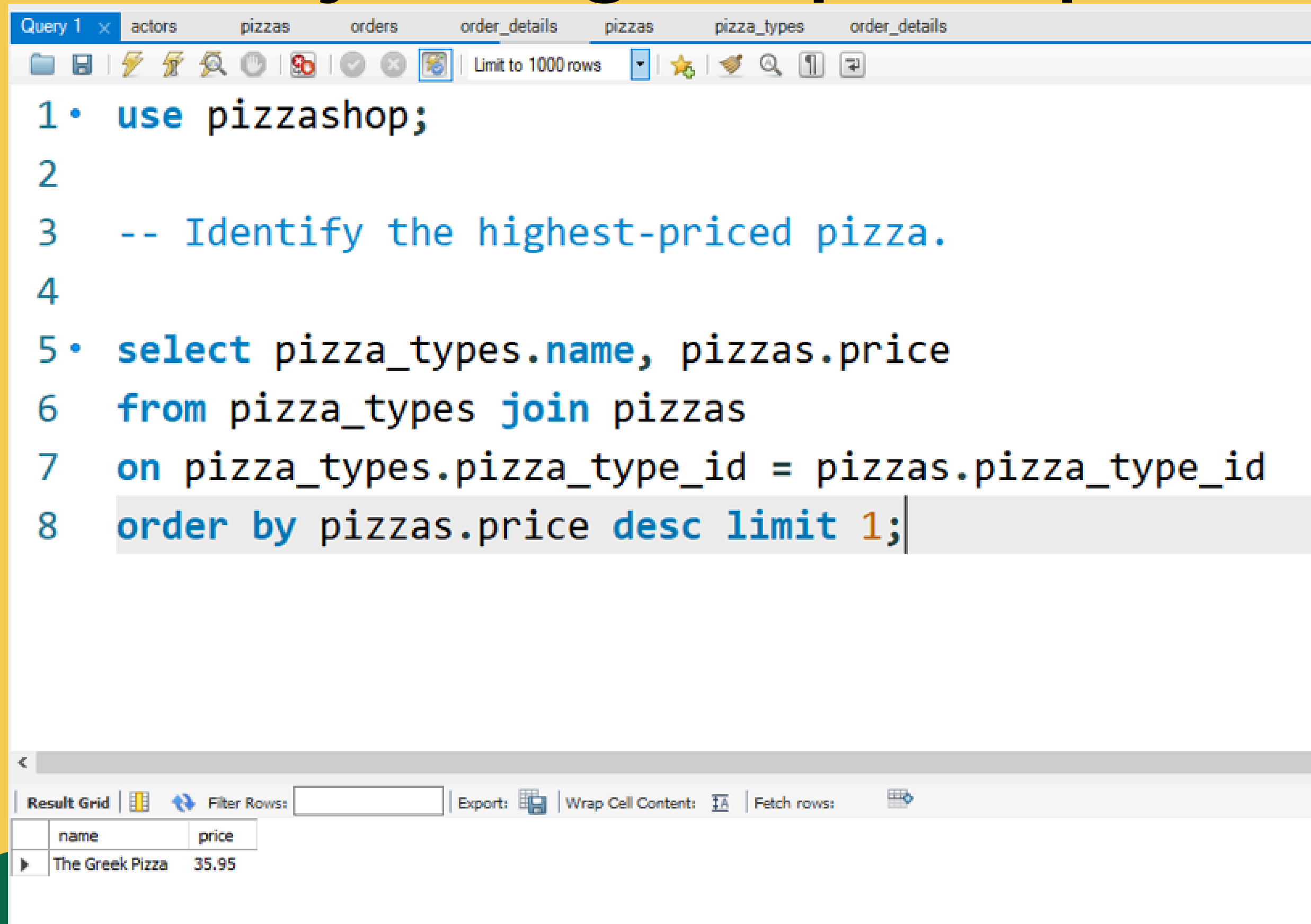
The screenshot shows a SQL query editor with a toolbar at the top containing icons for file operations, execution, and settings. The query text is as follows:

```
1 • use pizzashop;  
2  
3 -- Retrieve the total number of orders placed.  
4 • SELECT count(order_id) as total_orders FROM orders;
```

Below the query editor, there is a result grid section. It includes a 'Result Grid' tab, a 'Filter Rows' input field, and an 'Export' button. The result grid displays the following data:

	total_orders
▶	21350

Identify the highest-priced pizza.



The screenshot shows a database query editor interface. At the top, there are tabs for 'Query 1', 'actors', 'pizzas', 'orders', 'order_details', 'pizzas', 'pizza_types', and 'order_details'. Below the tabs is a toolbar with various icons, including a 'Limit to 1000 rows' dropdown. The main area contains a SQL query with line numbers 1 through 8. The query is as follows:

```
1 • use pizzashop;  
2  
3 -- Identify the highest-priced pizza.  
4  
5 • select pizza_types.name, pizzas.price  
6 from pizza_types join pizzas  
7 on pizza_types.pizza_type_id = pizzas.pizza_type_id  
8 order by pizzas.price desc limit 1;
```

At the bottom of the editor, there is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, a 'Wrap Cell Content' checkbox, and a 'Fetch rows' button. Below these controls is a table with the following data:

	name	price
▶	The Greek Pizza	35.95

Identify the most common pizza size ordered.

```
3  -- Identify the most common pizza size ordered.
4
5 • select pizzas.size, count(order_details.order_details_id) as order_count
6 from pizzas join order_details
7 on pizzas.pizza_id = order_details.pizza_id
8 group by pizzas.size order by order_count desc limit 1;
9
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
size	order_count				
L	18526				

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



```
3  -- List the top 5 most ordered pizza types along with their quantities.
4
5 • select pizza_types.name,
6    sum(order_details.quantity) as quantity
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.name order by quantity desc limit 5;
```

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

```
3  -- Join the necessary tables to find the
4  -- total quantity of each pizza category ordered.
5
6 • select pizza_types.category, sum(order_details.quantity) as quantity
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 order by quantity desc;
```

<
Result Grid
Filter Rows: <input type="text"/>
Export: 
Wrap Cell Content: 
category
quantity
Classic
14888
Supreme
11987
Veggie
11649
Chicken
11050

Determine the distribution of orders by hour of the day.

```
2
3 -- Determine the distribution of orders by hour of the day.
4
5 • select hour(order_time) as HOUR, count(order_id) AS ORDER_COUNT
6    from orders
7    group by hour(order_time);
```

<

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

	HOUR	ORDER_COUNT
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468
	16	1920
	17	2336

Result 1 x

Output

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

```
2
3  -- Join relevant tables to find the category-wise distribution of pizzas.
4
5 • select category, count(name) from pizza_types
6  group by category;
```

	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.






```
2
3 -- Group the orders by date and calculate the average number of pizzas ordered per day.
4
5
6 • select ROUND(avg(quantity),0) from
7   (select orders.order_date, sum(order_details.quantity) as quantity
8    from orders join order_details
9    on orders.order_id = order_details.order_id
10   group by orders.order_date) as order_quantity;
11
```

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

	ROUND(avg(quantity),0)
▶	138

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

```
4
5 • select pizza_types.name,
6    sum(order_details.quantity * pizzas.price) as revenue from pizza_types join pizzas
7    on pizzas.pizza_type_id = pizza_types.pizza_type_id
8    join order_details
9    on order_details.pizza_id = pizzas.pizza_id
10   group by pizza_types.name order by revenue desc limit 3;
11
```

Result Grid			Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 	Fetch 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.

```
2
3  -- Calculate the percentage contribution of each pizza type to total revenue.
4
5
6 • select pizza_types.category,
7    sum(order_details.quantity * pizzas.price) / (SELECT round(sum(order_details.quantity * pizzas.price),2) as total_sales
8  from order_details JOIN pizzas ON pizzas.pizza_id = order_details.pizza_id) *100 as revenue
9  from pizza_types join pizzas
10 on pizzas.pizza_type_id = pizza_types.pizza_type_id
11 join order_details
12 on order_details.pizza_id = pizzas.pizza_id
13 group by pizza_types.category order by revenue desc;
14
```

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

	category	revenue
▶	Classic	26.90596025566967
	Supreme	25.45631126009862
	Chicken	23.955137556847287
	Veggie	23.682590927384577

Analyze the cumulative revenue generated over time.

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

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

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

result 13 x

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

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

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

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