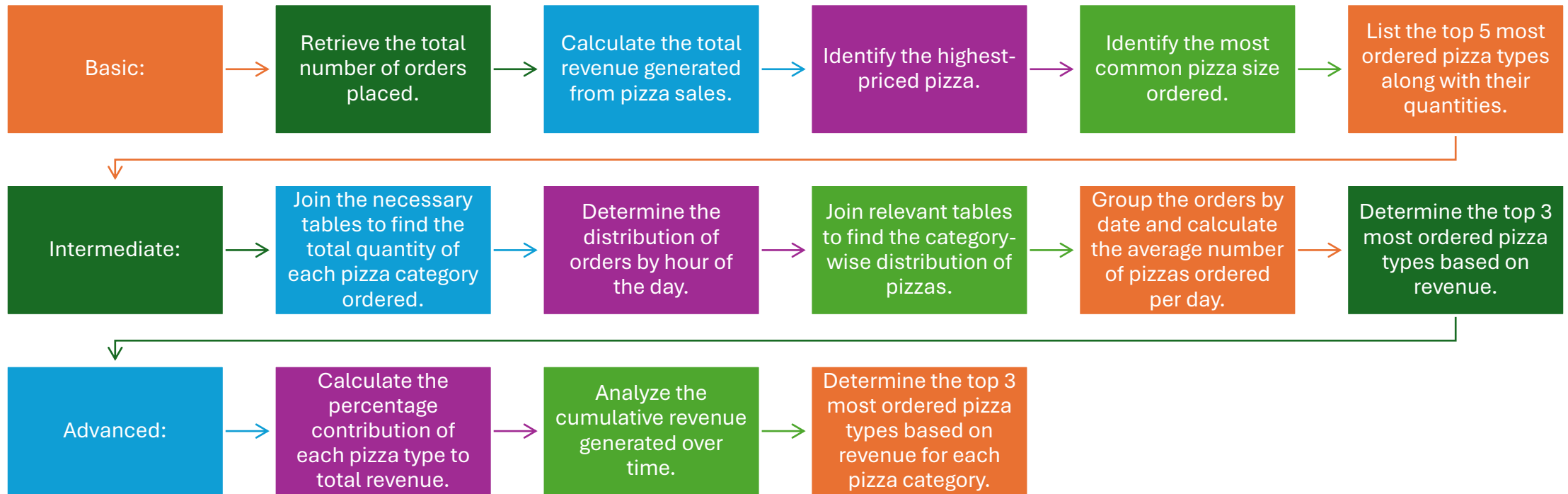
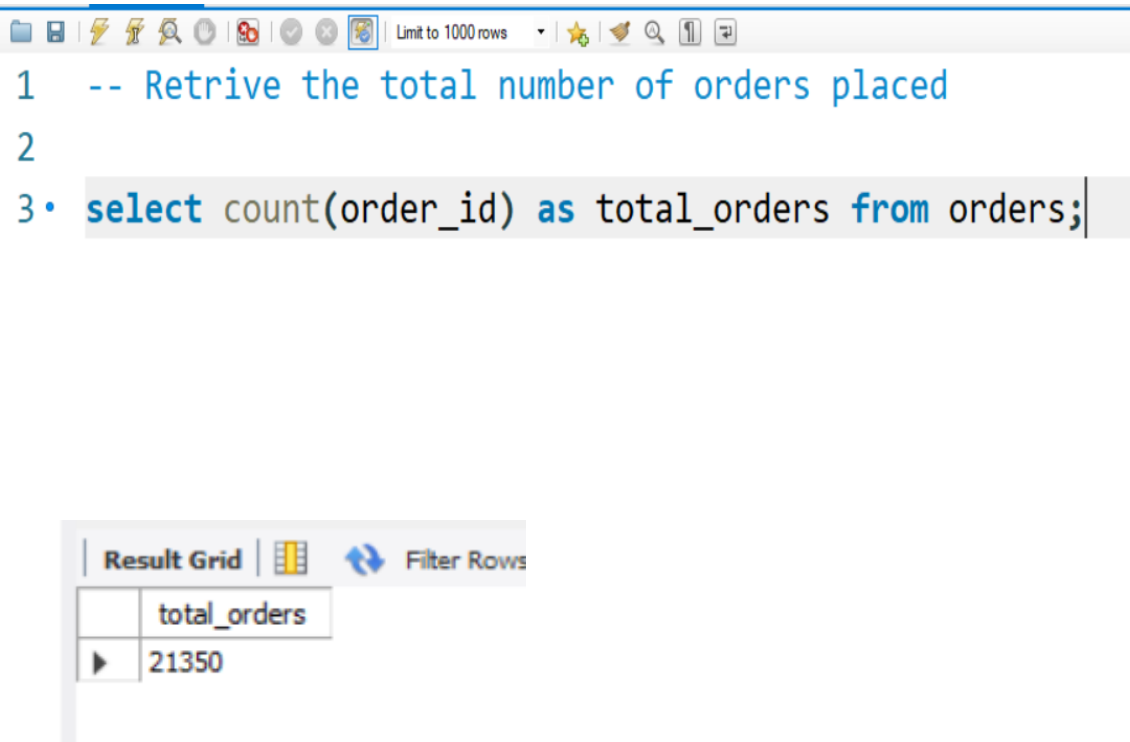


SQL Queries for Pizza Sales



Retrieve the
total number
of order
placed

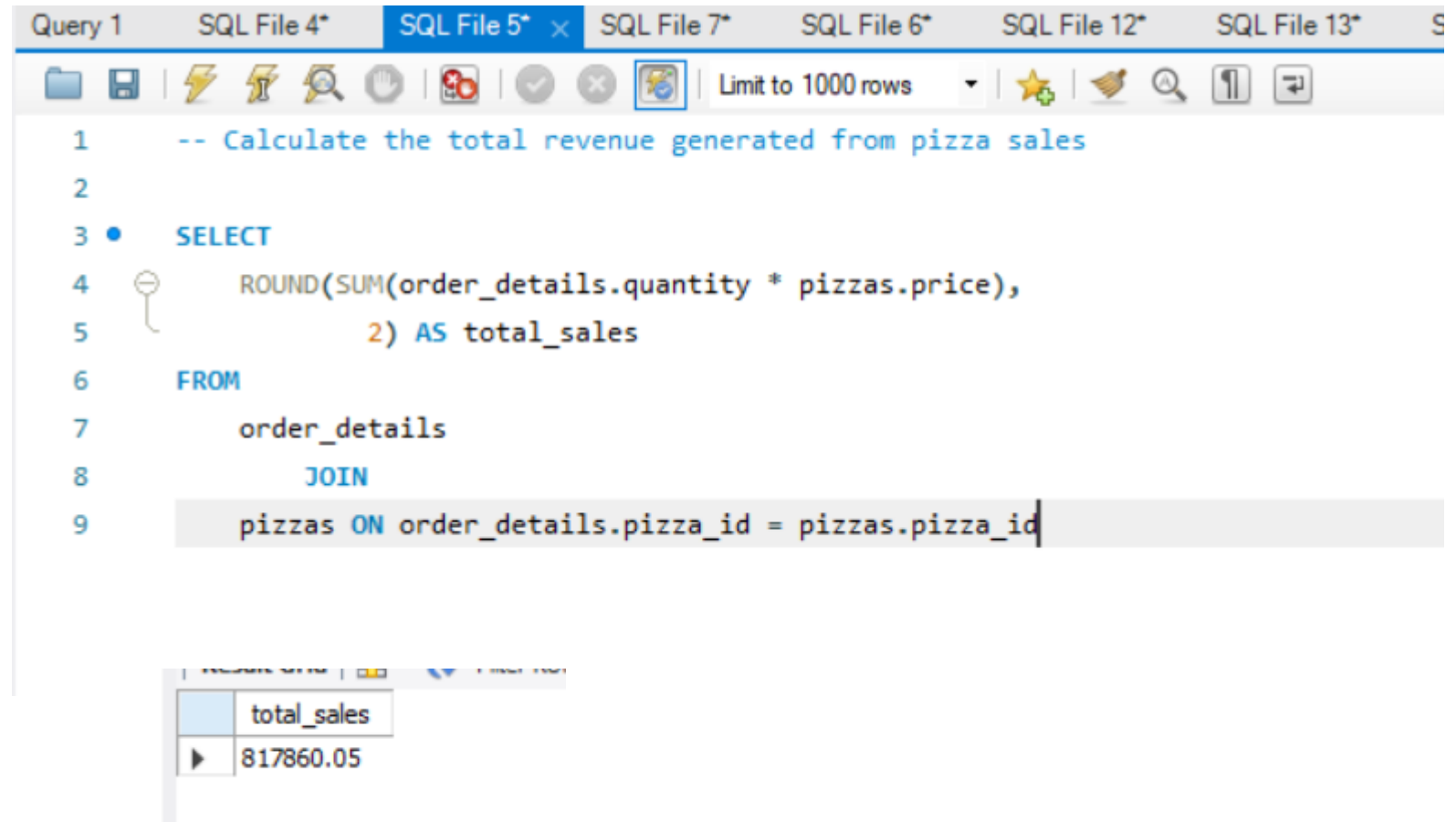


The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and a 'Limit to 1000 rows' dropdown. The query editor contains three lines of SQL code. The third line is highlighted. Below the editor, the 'Result Grid' tab is active, displaying a single row with the column 'total_orders' and the value '21350'.

```
1  -- Retrieve the total number of orders placed
2
3 • select count(order_id) as total_orders from orders;
```

	total_orders
▶	21350

Calculate
total revenue
generated
from pizza
sales



The screenshot shows a SQL IDE interface with multiple tabs at the top: "Query 1", "SQL File 4*", "SQL File 5*" (selected), "SQL File 7*", "SQL File 6*", "SQL File 12*", and "SQL File 13*". Below the tabs is a toolbar with various icons, including a "Limit to 1000 rows" dropdown. The main editor area contains the following SQL query:

```
1  -- Calculate the total revenue generated from pizza sales
2
3  •  SELECT
4      ROUND(SUM(order_details.quantity * pizzas.price),
5             2) AS total_sales
6  FROM
7      order_details
8      JOIN
9      pizzas ON order_details.pizza_id = pizzas.pizza_id
```

Below the query editor, a small window titled "Results of the query" displays the result of the query:

total_sales
817860.05

Identify the
highest
priced pizza

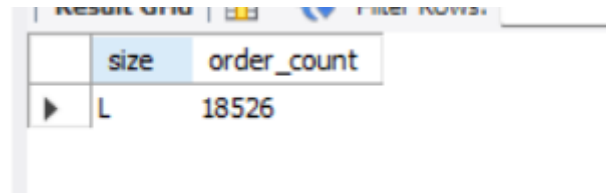
```
1  -- Identify the highest priced pizza
2
3  • SELECT
4      pizza_types.name, pizzas.price
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9  ORDER BY pizzas.price DESC
10 LIMIT 1;
```

Result Grid		Filter Rows:
	name	price
▶	The Greek Pizza	35.95

Identify the
most
common
pizza size
ordered

-- 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-- ;-- this is mandatory for if we have count or aggregate without this it will error out
order by order_count desc-- ;
limit 1;`



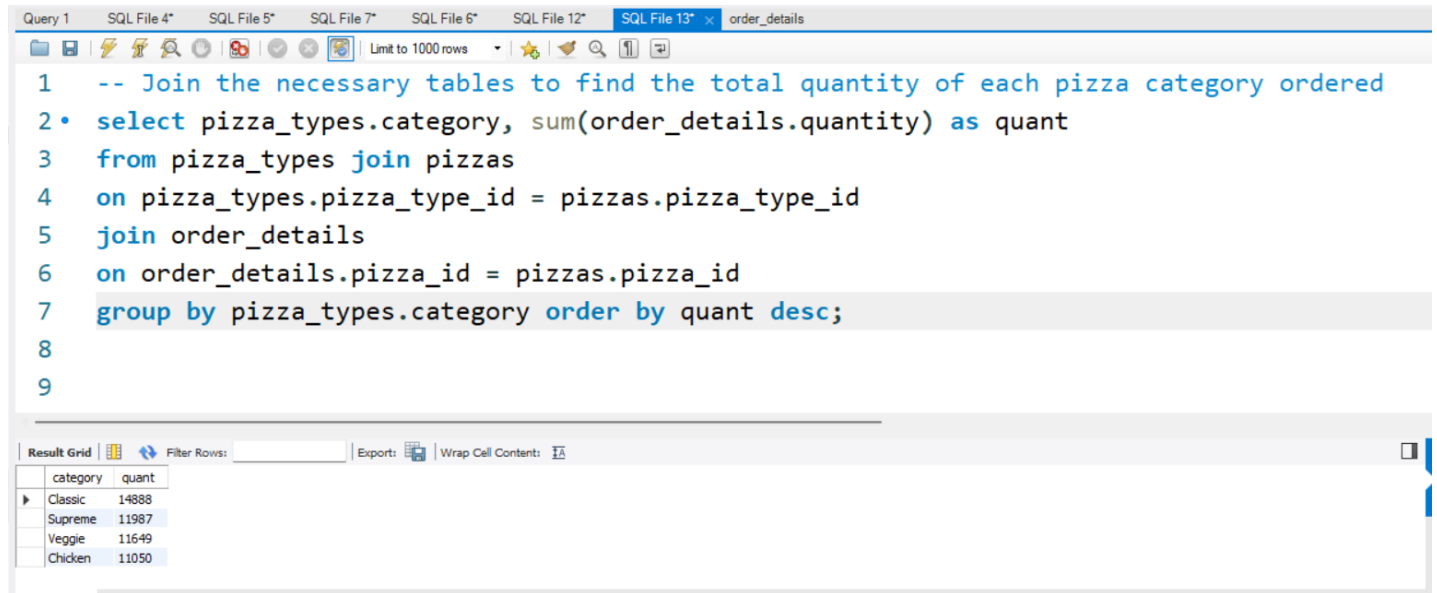
	size	order_count
▶	L	18526

List the top 5
most ordered
pizzas types
along with their
quantities

```
1  -- list the top 5 most ordered pizza types along with their quantities
2
3  • select pizza_types.name, sum(order_details.quantity) as quantity
4  from pizza_types join pizzas
5  on pizza_types.pizza_type_id=pizzas.pizza_type_id
6  join order_details
7  on order_details.pizza_id = pizzas.pizza_id
8  group by pizza_types.name order by quantity desc limit 5;
9
```

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



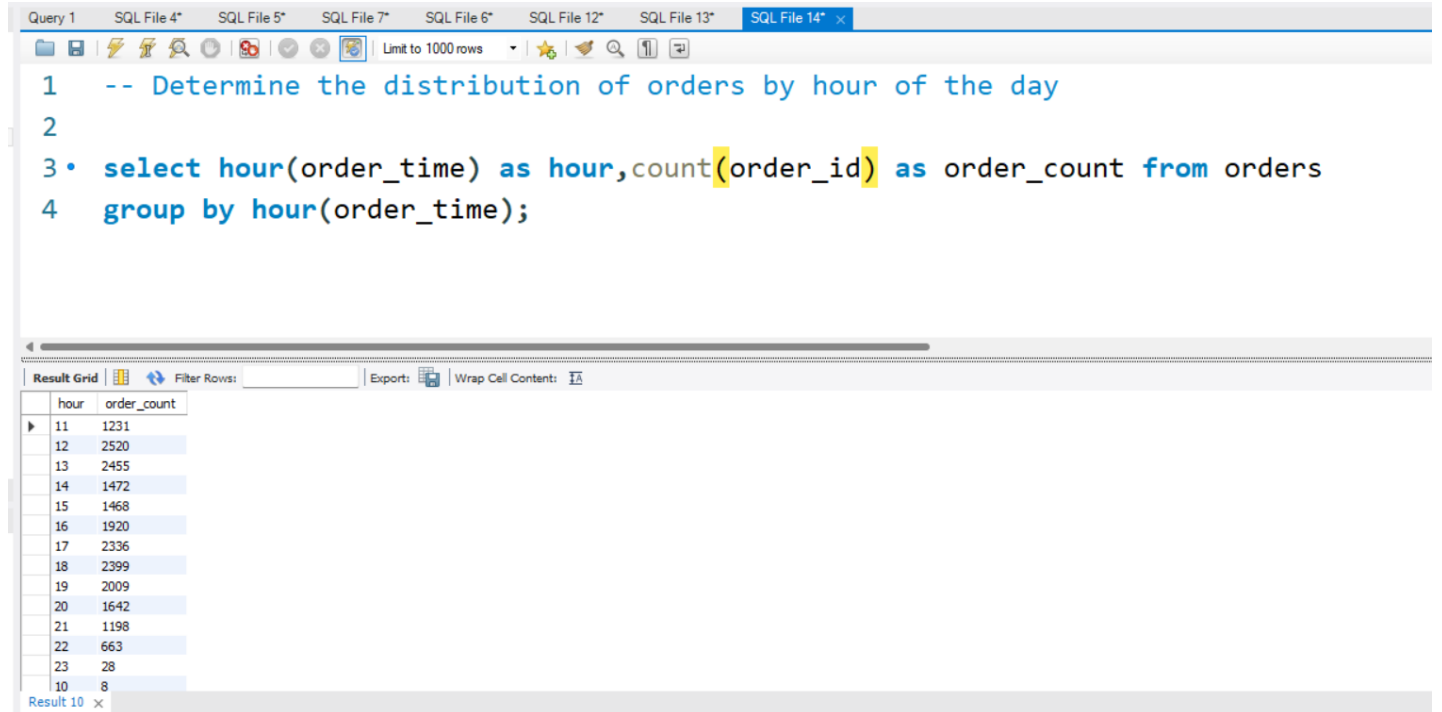
The screenshot shows a SQL IDE window with multiple tabs. The active tab is 'SQL File 13*' with the filename 'order_details'. The query editor contains the following SQL code:

```
1  -- Join the necessary tables to find the total quantity of each pizza category ordered
2 • select pizza_types.category, sum(order_details.quantity) as quant
3  from pizza_types join pizzas
4  on pizza_types.pizza_type_id = pizzas.pizza_type_id
5  join order_details
6  on order_details.pizza_id = pizzas.pizza_id
7  group by pizza_types.category order by quant desc;
8
9
```

Below the query editor, the 'Result Grid' is visible, showing the results of the query. The table has two columns: 'category' and 'quant'. The results are as follows:

category	quant
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the
distribution of
orders by hour
of the day



The screenshot shows a SQL IDE interface with multiple tabs. The active tab is 'SQL File 14*'. The query editor contains the following SQL code:

```
1 -- Determine the distribution of orders by hour of the day
2
3 • select hour(order_time) as hour, count(order_id) as order_count from orders
4 group by hour(order_time);
```

Below the query editor is a 'Result Grid' showing the results of the query. The grid has two columns: 'hour' and 'order_count'. The results are as follows:

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

The interface also includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' checkbox.

Join relevant
tables to find the
category wise
distribution of
pizzas

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

Limit to 1000 rows

Result Grid

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

Group the order
by date and
calculate the
average number
of pizzas ordered
per day

```
1  -- Group the order by date and calculate the average
2  -- number of pizzas ordered per day
3  • select round(avg(quantity)) as avg_pizza_ordered_per_day from
4  (select orders.order_date, sum(order_details.quantity) as quantity
5   from orders join order_details
6   on orders.order_id = order_details.order_id
7   group by orders.order_date) as order_quantity
```

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

	avg_pizza_ordered_per_day
▶	138

Determine the
top 3 most
ordered pizza
type based on
revenue

```
1  -- Determine the top 3 most ordered pizza types based on revenue
2
3 • select pizza_types.name,
4    sum(order_details.quantity * pizzas.price) as revenue
5  from pizza_types join pizzas
6    on pizzas.pizza_type_id = pizza_types.pizza_type_id
7  join order_details
8    on order_details.pizza_id = pizzas.pizza_id
9  group by pizza_types.name order by revenue desc limit 3;
10
```

Result Grid

	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

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

Result 6 x

Analyze the cumulative revenue generated over time

```
1  -- Analyze the cumulative revenue generated over time
2 • select order_date,
3     sum(revenue) over (order by order_date) as cum_revenue
4  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;
```

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

	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

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

```
1  -- Determine the top 3 most ordered pizza types based on revenue for each pizza category
2 • select name, revenue from
3  (select category, name, revenue,
4   rank() over (partition by category order by revenue desc) as rn
5   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:	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		