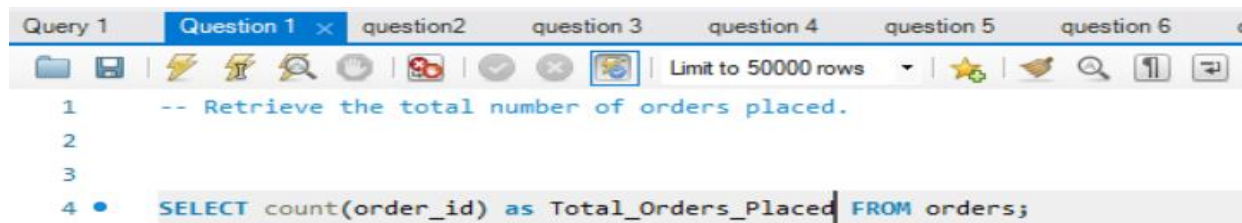


Question 1 With Output:-



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

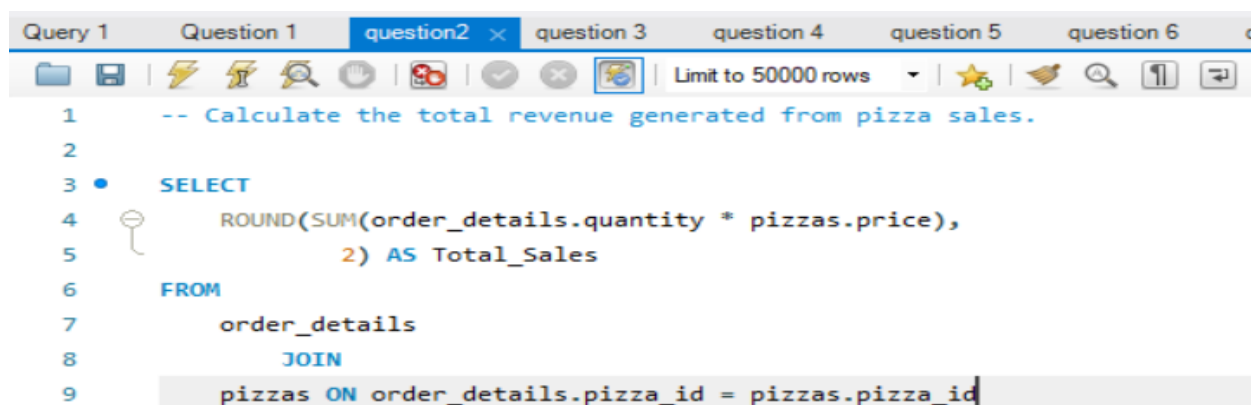
```
1  -- Retrieve the total number of orders placed.
2
3
4  •  SELECT count(order_id) as Total_Orders_Placed FROM orders;
```



The screenshot shows the result grid for the first query. It has a toolbar with 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content' options. The table below shows the result:

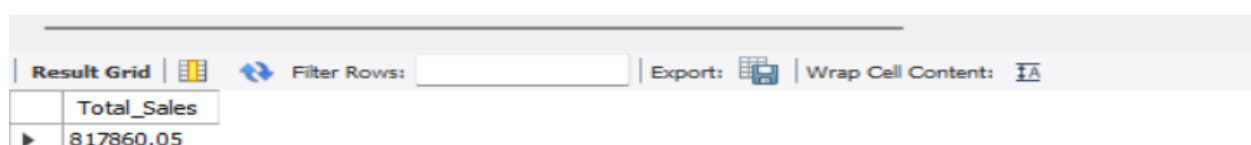
	Total_Orders_Placed
▶	21350

Question 2 With Output:-



The screenshot shows a SQL query editor with a toolbar at the top. The query is as follows:

```
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
```



The screenshot shows the result grid for the second query. It has a toolbar with 'Result Grid', 'Filter Rows', 'Export', and 'Wrap Cell Content' options. The table below shows the result:

	Total_Sales
▶	817860.05

Question 3 With Output:-

```
Query 1  Question 1  question2  question 3 x question 4  question 5  question 6
[Icons] Limit to 50000 rows [Icons]
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 price DESC
10 LIMIT 1;
```

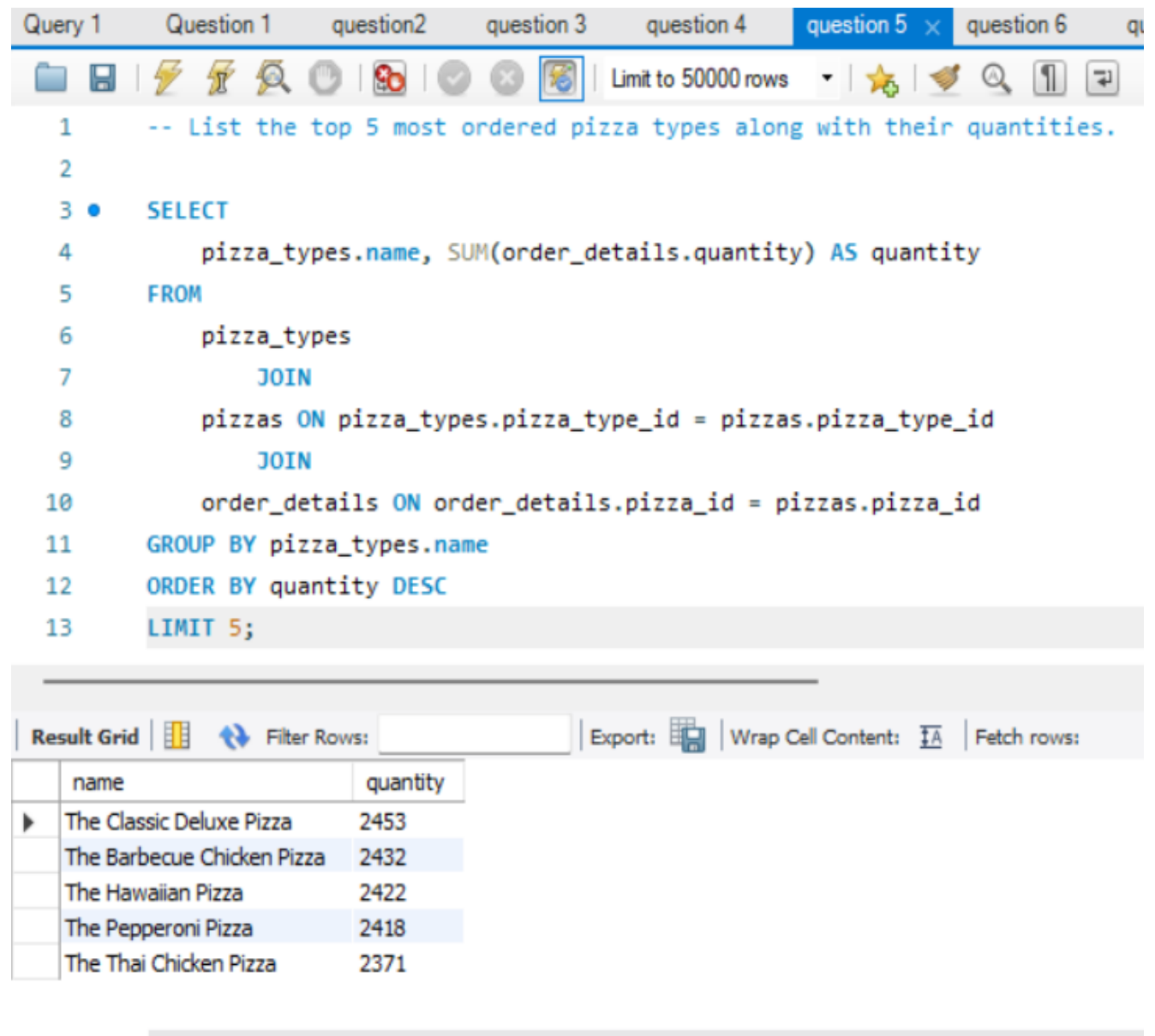
Result Grid	[Icons]	Filter Rows: <input type="text"/>	Export: [Icon]	Wrap Cell Content: <input type="checkbox"/>	Fetch rows: <input type="text"/>
	name	price			
▶	The Greek Pizza	35.95			

Question 4 With Output:-

```
Query 1  Question 1  question2  question 3  question 4 x question 5  questi
[Icons] Limit to 50000 rows [Icons]
1  -- Identify the most common pizza size ordered.
2
3 • SELECT
4     pizzas.size,
5     COUNT(order_details.order_details_id) AS orders_count
6 FROM
7     pizzas
8     JOIN
9     order_details ON pizzas.pizza_id = order_details.pizza_id
10 GROUP BY pizzas.size
11 ORDER BY orders_count DESC;
```

Result Grid	[Icons]	Filter Rows: <input type="text"/>	Export: [Icon]	Wrap Cell Content: <input type="checkbox"/>
	size	orders_count		
▶	L	18526		
	M	15385		
	S	14137		
	XL	544		
	XXL	28		

Question 5 With Output:-



The screenshot shows a SQL IDE interface with a query editor and a results grid. The query editor has tabs for 'Query 1', 'Question 1', 'question2', 'question 3', 'question 4', 'question 5' (selected), and 'question 6'. The query in the editor is as follows:

```
1  -- List the top 5 most ordered pizza types along with their quantities.
2
3  •  SELECT
4      pizza_types.name, SUM(order_details.quantity) AS quantity
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9      JOIN
10     order_details ON order_details.pizza_id = pizzas.pizza_id
11  GROUP BY pizza_types.name
12  ORDER BY quantity DESC
13  LIMIT 5;
```

Below the query editor, the results grid is displayed. It has a toolbar with 'Result Grid', 'Filter Rows', 'Export', 'Wrap Cell Content', and 'Fetch rows'. The results grid shows the following data:

	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

Question 6 With Output:-

Query 1 Question 1 question2 question 3 question 4 question 5 **question 6** × question 7 quest

Limit to 50000 rows

```
1  -- Join the necessary tables to find the total quantity of each pizza category ordered.
2  • SELECT
3      pizza_types.category, SUM(order_details.quantity) AS quantity
4  FROM
5      pizza_types
6      JOIN
7      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8      JOIN
9      order_details ON order_details.pizza_id = pizzas.pizza_id
10 GROUP BY pizza_types.category
11 ORDER BY quantity DESC;
```

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

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Question 7 With Output:-

Query 1 Question 1 question2 question 3 question 4 question 5 que

Limit to 50000 rows

```
1  -- Determine the distribution of orders by hour of the day.
2
3  • SELECT
4      HOUR(order_time) AS hours, COUNT(order_id) AS count_orders
5  FROM
6      orders
7  GROUP BY hours
8  ORDER BY count_orders DESC;
```

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

	hours	count_orders
▶	12	2520
	13	2455
	18	2399
	17	2336
	19	2009
	16	1920
	20	1642

Question 8 With Output:-

```
Query 1  Question 1  question2  question 3  question 4  question 5  question 6  qu
-- Join relevant tables to find the category-wise distribution of pizzas.
1
2
3 • SELECT
4     category, COUNT(name) AS count
5 FROM
6     pizza_types
7 GROUP BY category;
```

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

	category	count
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

Question 9 With Output:-

```
Query 1  Question 1  question2  question 3  question 4  question 5  question 6  question 7  questio
-- Group the orders by date and calculate the average number of pizzas ordered per day.
1
2
3 • select round(avg(quantity),0)as avg_pizzas_orderd_per_day
4 from
5
6 (select orders.order_date,sum(order_details.quantity) as quantity
7  from orders join order_details
8  on orders.order_id=order_details.order_id
9  group by orders.order_date) as order_quantity;
```

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

	avg_pizzas_orderd_per_day
▶	138

Question 10 With Output:-

The screenshot shows a SQL query editor interface with a toolbar at the top containing icons for file operations, execution, and settings. The query is written in a multi-line editor with line numbers 1 through 14. The query text is as follows:

```
1  -- Determine the top 3 most ordered pizza types based on revenue.  
2  
3  ●  SELECT  
4      pizza_types.name,  
5      sum(pizzas.price * order_details.quantity) AS Revenue  
6  FROM  
7      pizza_types  
8      JOIN  
9      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id  
10     JOIN  
11     order_details ON order_details.pizza_id = pizzas.pizza_id  
12 GROUP BY pizza_types.name  
13 ORDER BY Revenue desc  
14 LIMIT 3;
```

Below the query editor, there is a toolbar for the results section, including a 'Result Grid' button, a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' button. The results are displayed in a table with two columns: 'name' and 'Revenue'.

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

Question 11 solution:-

```
question 3 question 4 question 5 question 6 question 7 question 8 question 9 question 10
Limit to 50000 rows
















1  -- Calculate the percentage contribution of each pizza type to total revenue.
2  • SELECT
3      pizza_types.category,
4      ROUND(SUM(pizzas.price * order_details.quantity) / (SELECT
5          ROUND(SUM(order_details.quantity * pizzas.price),
6              2) AS Total_Sales
7          FROM
8              order_details
9              JOIN
10                 pizzas ON order_details.pizza_id = pizzas.pizza_id) * 100,
11          2) AS revenue
12  FROM
13      pizza_types
14      JOIN
15      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
16      JOIN
17      order_details ON order_details.pizza_id = pizzas.pizza_id
18  GROUP BY pizza_types.category
19  ORDER BY Revenue DESC;
```

Output:-





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

Question 12 With Outpu :-

question 3 question 4 question 5 question 6 question 7 question 8 question 9 question 10 que

          Limit to 50000 rows     

```
1  -- Analyze the cumulative revenue generated over time.
2
3
4  •  select order_date,sum(Revenue) over (order by order_date) as Cumullative_revenue
5     from
6     (select orders.order_date,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: 

	order_date	Cumullative_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

Result 1 x

Question 13 With Output :-

Question 3 question 4 question 5 question 6 question 7 question 8 question 9 question 10 question 12

Limit to 50000 rows

```
1  -- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
2  • select name,Revenue
3  from
4  (select category,name,Revenue,
5   rank() over(partition by category order by Revenue desc) as rnk
6   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
14  where rnk<=3;
```

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

	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

Result 1 x

---THE END---

