📄 Pizza Sales SQL Project:

Q1. Retrieve the total number of orders placed.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q2. Calculate the total revenue generated from pizza sales.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q3. Identify the highest-priced pizza.

SQL Query:

```
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;
```

📝 *Run this query in your SQL environment to view the output.*

Q4. Identify the most common pizza size ordered.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q5. List the top 5 most ordered pizza types along with their quantities.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q6. Join the necessary tables to find the total quantity of each pizza category ordered.

SQL Query:

```
SELECT pizza_types.category, SUM(orders_details.quantity) AS quantity
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY quantity DESC;
```

📝 *Run this query in your SQL environment to view the output.*

Q7. Determine the distribution of orders by hour of the day.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q8. Join relevant tables to find the category-wise distribution of pizzas.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q9. Group the orders by date and calculate the average number of pizzas ordered per day.

SQL Query:

```
SELECT ROUND(AVG(quantity), 2) AS Avg_of_orders
FROM (
  SELECT orders.order_date, SUM(orders_details.quantity) AS quantity
  FROM orders
  JOIN orders_details ON orders.order_id = orders_details.order_id
  GROUP BY orders.order_date
) AS order_quantity;
```

📝 *Run this query in your SQL environment to view the output.*

Q10. Determine the top 3 most ordered pizza types based on revenue.

SQL Query:

```
SELECT pizza_types.name, SUM(orders_details.quantity * pizzas.price) AS revenue
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
```

```
JOIN orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

📝 *Run this query in your SQL environment to view the output.*

Q11. Calculate the percentage contribution of each pizza type to total revenue.

SQL Query:

```
SELECT pizza_types.category,
  ROUND(SUM(orders_details.quantity * pizzas.price) /
  (SELECT ROUND(SUM(orders_details.quantity * pizzas.price), 2)
   FROM orders_details
   JOIN pizzas ON pizzas.pizza_id = orders_details.pizza_id) * 100, 2) AS revenue
FROM pizza_types
JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
JOIN orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

📝 *Run this query in your SQL environment to view the output.*

Q12. Analyze the cumulative revenue generated over time.

SQL Query:

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

📝 *Run this query in your SQL environment to view the output.*

Q13. Determine the top 3 most ordered pizza types based on revenue for each pizza category.

SQL Query:

```
SELECT category, 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(orders_details.quantity * pizzas.price) AS revenue
    FROM pizza_types
    JOIN pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
    JOIN orders_details ON orders_details.pizza_id = pizzas.pizza_id
    GROUP BY pizza_types.category, pizza_types.name
  ) AS a
) AS b
WHERE rn <= 3;
```

📝 *Run this query in your SQL environment to view the output.*