Pizza Sales Analysis - SQL

Basic:

1. Retrieve the total number of orders placed.

```
SELECT

COUNT(order_id) AS total_orders

FROM

orders;

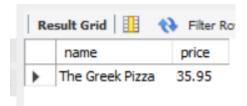
Result Grid 1 total_orders

total_orders

21350
```

2. Calculate the total revenue generated from pizza sales.

3. Identify the highest-priced pizza.



4. Identify the most common pizza size ordered.

```
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

LIMIT 1;

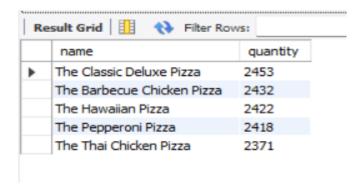
Result Grid  Fits

size order_count

L 18526
```

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

```
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;
```



Intermediate:

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

```
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;
```



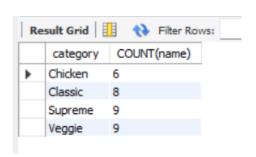
7. Determine the distribution of orders by hour of the day.

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



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

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



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

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

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

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



Advanced:

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

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



12. Analyze the cumulative revenue generated over time.

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

	order_date	cum_revenue
•	2015-01-01	2713.85000000000004
	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	36937.65000000001
	2015-01-17	39001.75000000001
	2015-01-18	40978.600000000006
	2015-01-19	43365.75000000001
	2015-01-20	45763.65000000001
	2015-01-21	47804.20000000001
	2015-01-22	50300.90000000001
	2015-01-23	52724.6000000000006
	2015-01-24	55013.8500000000006

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
3
     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;
```

	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