OwenStory
Pizza Store

SALES REPORT ANALYSIS

Retrieve the total number of orders placed.

```sql
-- Retrieve the total number of orders placed.

SELECT count(order_id) AS total_orders
FROM orders
```

total_orders
21350

Calculate the total revenue generated from pizza sales..

```sql
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS total_Revenue
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id
```

Result Grid | Filter Rows:

total_Revenue
817860.05

Identify the highest-priced pizza..

```sql
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;
```

Result Grid | Filter Rows:

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

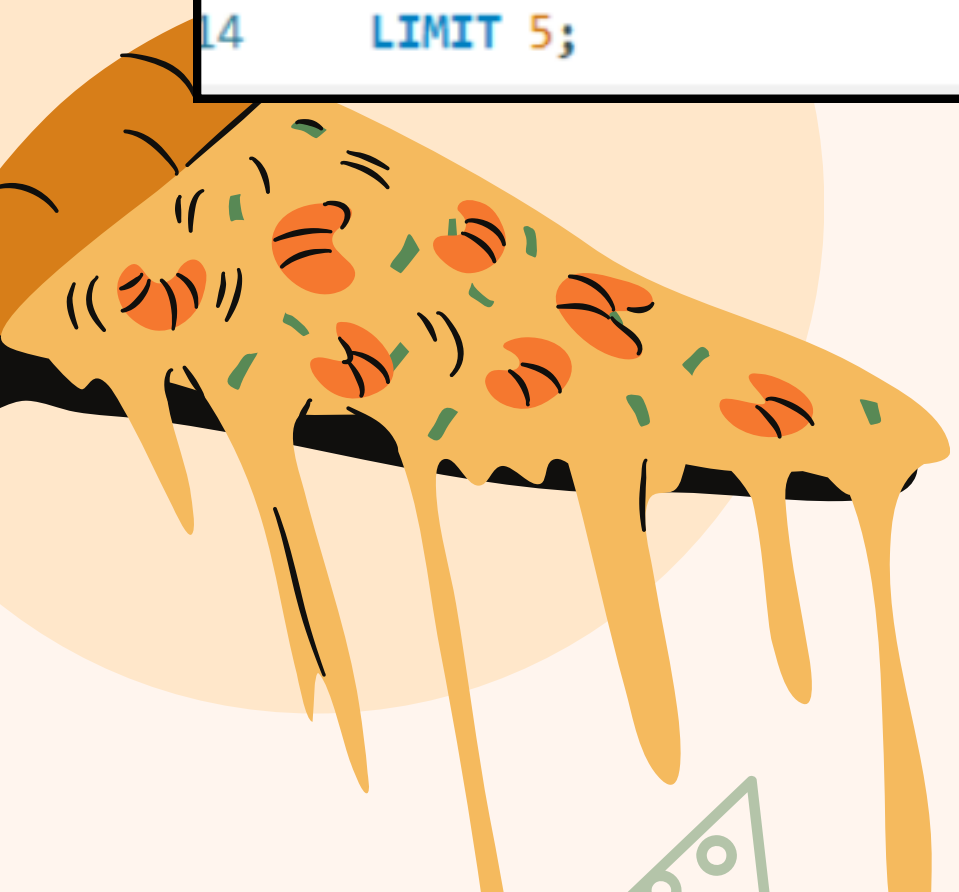
```
2
3 •   SELECT
4         pizzas.size,
5         COUNT(order_details.order_details_id) AS order_count
6     FROM
7         pizzas
8             JOIN
9         order_details ON pizzas.pizza_id = order_details.pizza_id
10    GROUP BY pizzas.size
11    ORDER BY order_count DESC
```

Result Grid | Filter Rou

size	order_count
L	18526
M	15385
S	14137
XL	544
XXL	28

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

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



Result Grid | Filter Rows:

name	order_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.

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

Result Grid | Filt

category	quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

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

Result Grid

hour	order_count
11	2
11	1
12	1
12	3
12	1
12	1
12	1
12	1
12	1
12	1
13	2
13	1
13	2
13	1
13	2
13	2

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

```
3 •   SELECT
4         category, COUNT(name)
5     FROM
6         pizza_types
7     GROUP BY category
```

Result Grid | Filter Rows:

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.

```sql
SELECT
    ROUND(AVG(quantity), 0)
FROM
    (SELECT
        orders.order_date, SUM(order_details.quantity) AS quantity
    FROM
        orders
    JOIN order_details ON orders.order_id = order_details.order_id
    GROUP BY orders.order_date) AS order_quantity;
```

Result Grid | Filter Rows:

	Average_pizza_per_day
▶	138

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

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

Result Grid

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

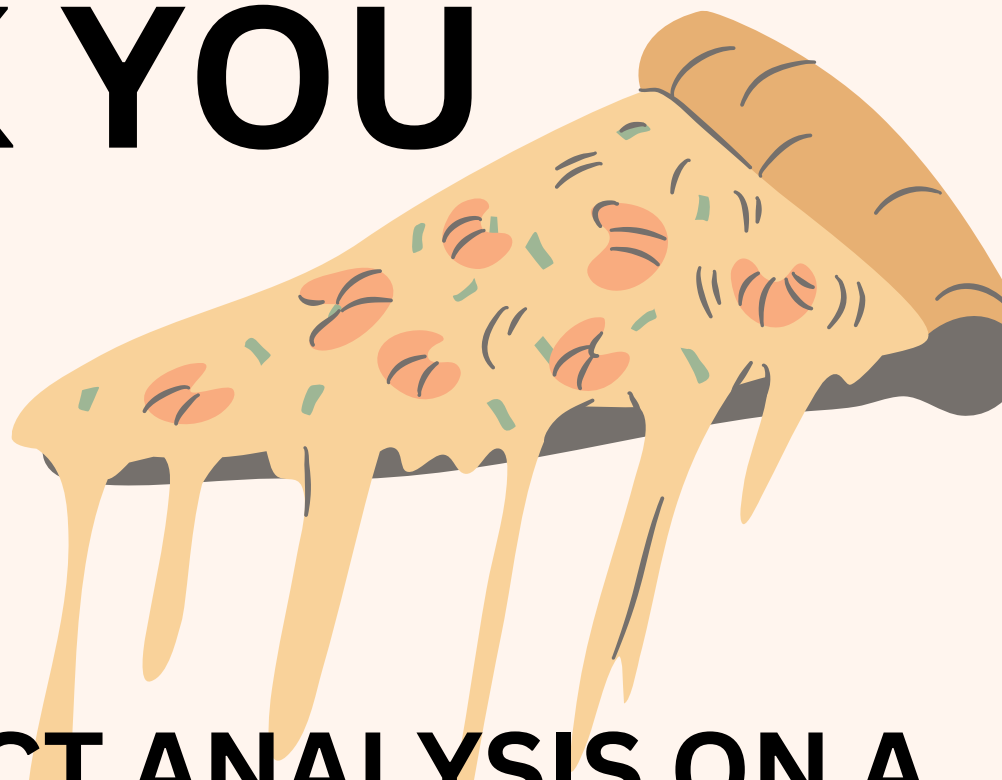
Analyze the cumulative revenue generated over time.

```sql
SELECT order_date,
sum(revenue) over (ORDER BY order_date) AS Cummlative_Revenue
FROM
(SELECT orders.order_date,
sum(order_details.quantity * pizzas.price) as revenue
FROM order_details JOIN pizzas
ON order_details.pizza_id = pizzas.pizza_id
JOIN orders
ON orders.order_id = order_details.order_id
GROUP BY orders.order_date) AS Sales
```

Result Grid | Filter Rows:

order_date	Cummlative_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
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	34343.50000000001
2015-01-16	36937.65000000001

THANK YOU



THIS WAS MY PROJECT ANALYSIS ON A PIZZA STORE SALES

RUSHIKESH GARATE