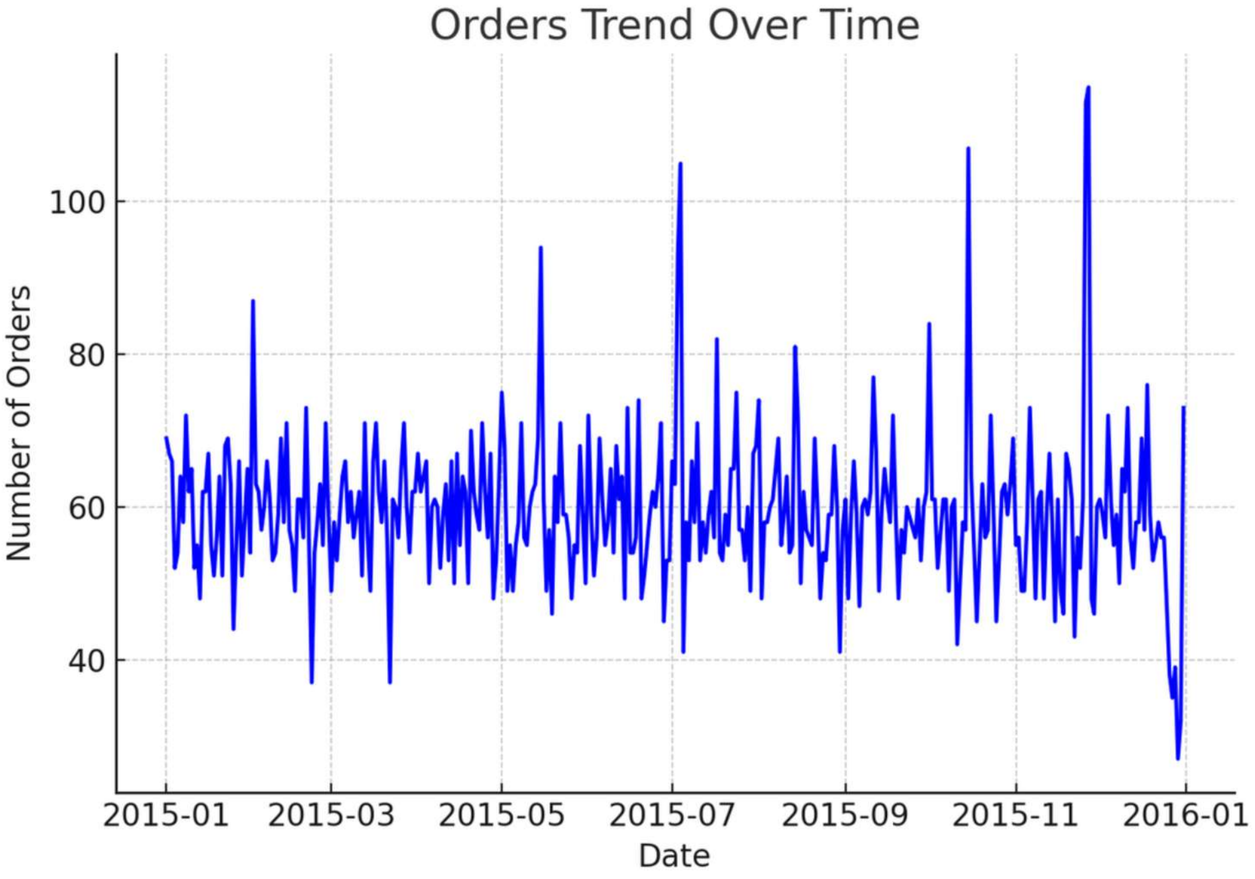
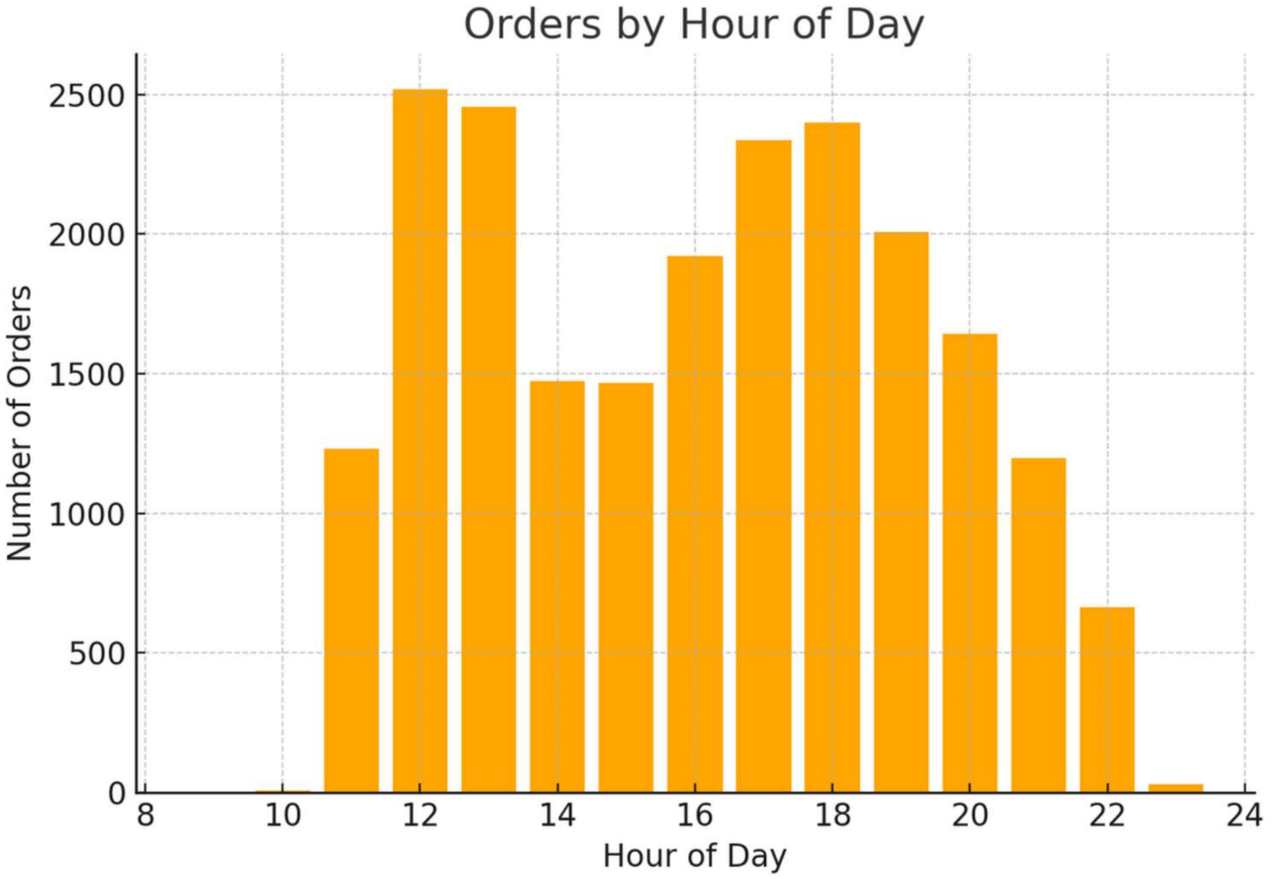
PIZZA SALES

Marketing Analytics

UNDERSTANDING THE
SALES
THROUGH SQL

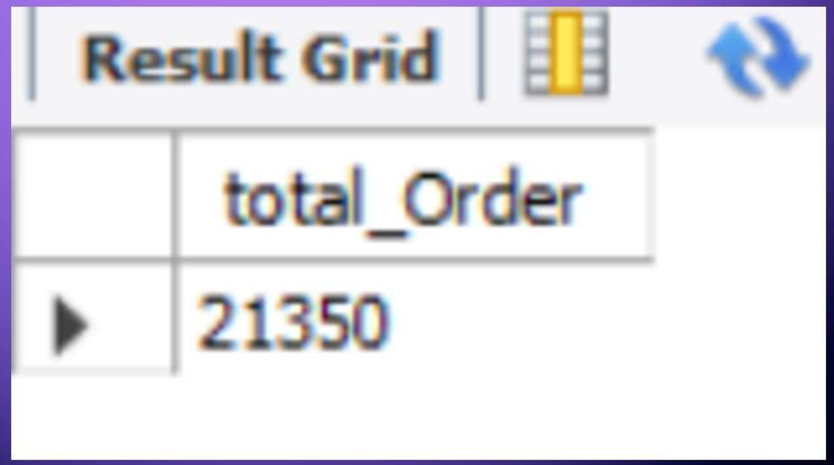




Total Orders
21,350

RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
1      -- Retrieve the total number of orders placed.
2
3 •    select * from orders;
4 •    select count(order_id) from orders;
5 •    select count(order_id) as total_Order from orders;
6
```

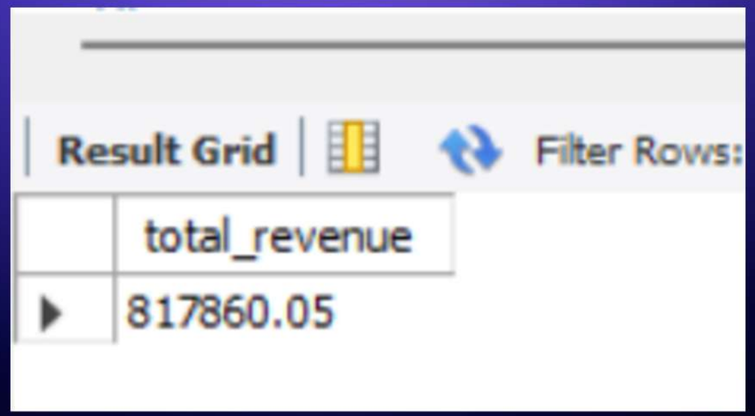


CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```sql
-- Calculate the total revenue generated from pizza sales.

-- solution
-- Now the problem is the data is not in the single table so u can make a table
-- or else directly fetch the data from the both tables with the help of common thing(column) i.e. pizza_id

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

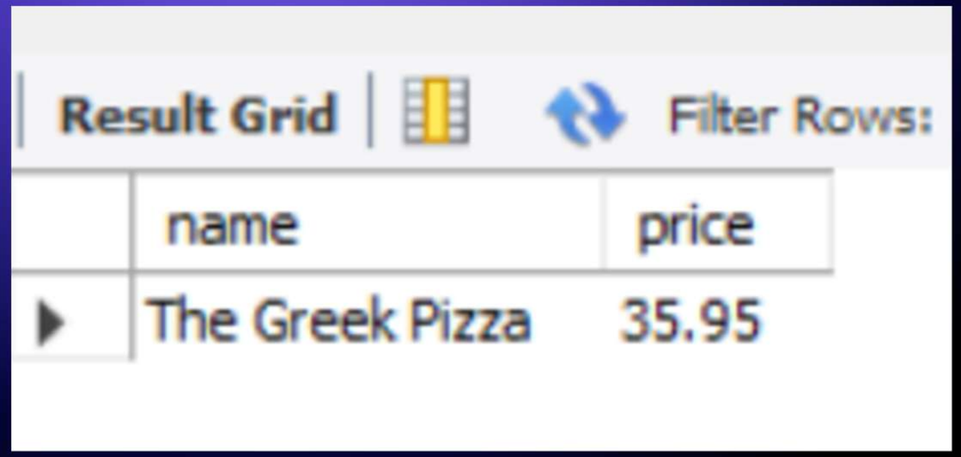


Identify the highest-priced pizza.

```sql
-- Identify the highest-priced pizza.

-- In this we have to find the highest price pizza Name so take the name from the pizza_type table and price from the pizzas
-- after that just arracnge in desc order on the basis of prices and then limit by 1.

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;
```



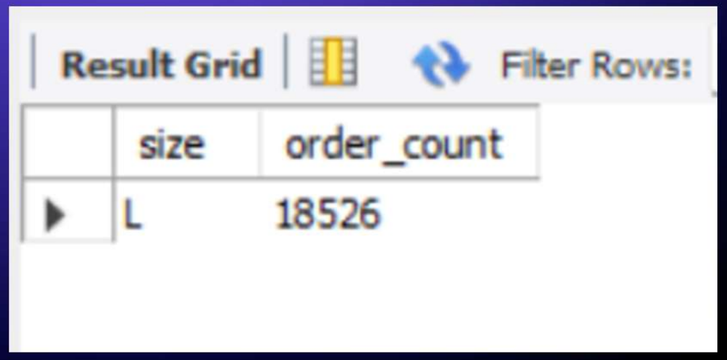
Identify the most common pizza size ordered.

```sql
-- Identify the most common pizza size ordered.

-- group by size and take the pizzaid common from both table pizzas and from order_details
-- then count the order_details_id with size of pizzas
-- please make a point always most selling thing is not a most common thing here u need to find most common not most selling

SELECT
    pizzas.size,
    COUNT(orders_details.order_details_id) AS order_count
FROM
    pizzas
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC
LIMIT 1;

```



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

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

-- Now here most ordered means most selling in quantity
-- just take quantity from order details, here u have to pick the name.
-- So grp the quantity to pizza id and join to the pizza's table with pizza's id and then take the name from pizza types having same pizza_type_id

-- Note* Here we join three tables one by one  with common columns in it ok

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 pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizza_types.name
ORDER BY quantity DESC
LIMIT 5;
```

Result Grid | Filter 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.

```sql
-- Join the necessary tables to find the total quantity of each pizza category ordered.

SELECT
    pizza_types.category,
    SUM(orders_details.quantity) AS total_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 total_Quantity DESC;
```

Result Grid | Filter Rows:

category	total_Quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

Determine the distribution of orders by hour of the day.

```sql
-- Determine the distribution of orders by hour of the day.

select hour(order_time) as hours, count(order_id) as orders from orders
group by hours order by orders desc;
```

Result Grid

hours	orders
12	2520
13	2455
18	2399
17	2336
19	2009
16	1920
20	1642
14	1472
15	1468
11	1231
21	1198
22	663
23	28

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

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

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

Result Grid | Filter Rows:

category	COUNT(name)
Supreme	9
Veggie	9
Classic	8
Chicken	6

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

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.

SELECT
    ROUND(AVG(total_orders), 0) as Average_Pizza_perDay
FROM
    (SELECT
        orders.order_date,
            SUM(orders_details.quantity) AS total_orders
    FROM
        orders
    JOIN orders_details ON orders.order_id = orders_details.order_id
    GROUP BY orders.order_date) AS averageNumber_perday;
```

Result Grid | Filter Rows:

Average_Pizza_perDay
138

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

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

Result Grid | Filter Rows:

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.

```sql
-- 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_revenue
FROM
    orders_details
        JOIN
    pizzas ON orders_details.pizza_id = pizzas.pizza_id))* 100, 0) As revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
        JOIN
    orders_details ON pizzas.pizza_id = orders_details.pizza_id
GROUP BY pizza_types.category
ORDER BY revenue DESC;
```

Result Grid | Filter Rows:

category	revenue
Classic	27
Supreme	25
Veggie	24
Chicken	24

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

```sql
-- 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
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;
```

Result Grid | Filter Rows:

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

Analyze the cumulative revenue generated over time.

```sql
-- Analyze the cumulative revenue generated over time.


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 orders_details.pizza_id = pizzas.pizza_id
join orders
on orders.order_id = orders_details.order_id
group by orders.order_date) as sales;
```

Result Grid | Filter Rows:

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
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