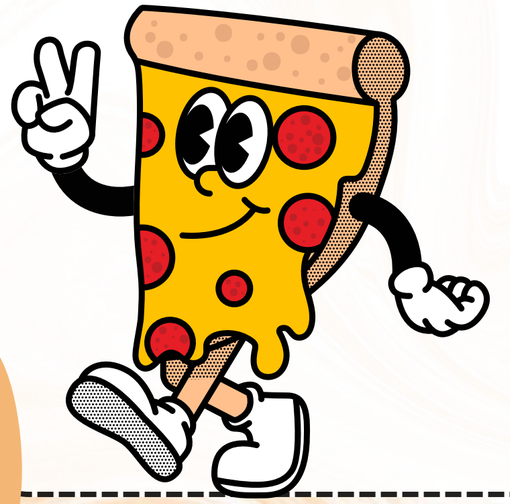
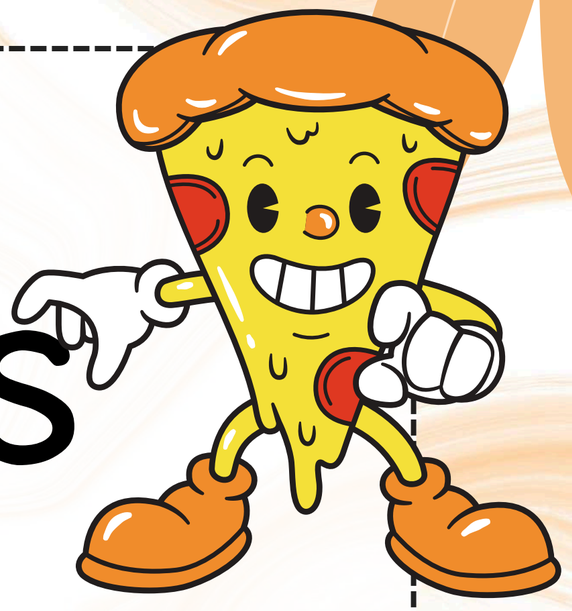
PIZZA SALES



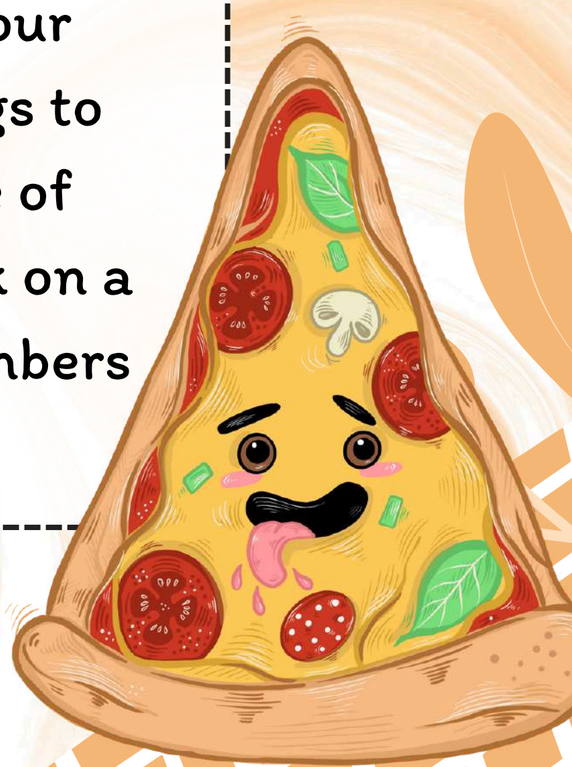


Using SQL

- by Ishika Kaur

INTRODUCTION

Welcome to Ishika's Pizza Sales Project, where SQL magic meets the delicious world of pizza! 🍕 Using the power of SQL, I has dived deep into sales data to uncover the secrets behind our favorite cheesy delights. From tracking top-selling toppings to identifying peak pizza hours, this project serves up a slice of data-driven insights with a side of fun. Get ready to embark on a flavorful journey through queries, and mouth-watering numbers that reveal the story of pizza sales like never before!



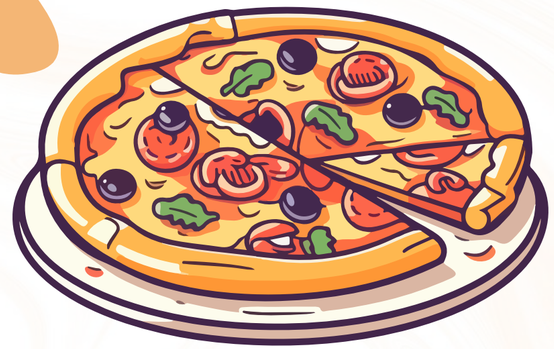
RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
1 • SELECT
2       COUNT(order_id)
3       AS total_orders
4   FROM
5       orders;
```

Result Grid

	total_orders
▶	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.



Result Grid

	total_sales
▶	817860.05

Limit to 1000 rows

```sql
-- Calculate the total revenue generated from pizza sales.
SELECT
    ROUND(SUM(order_details.quantity * pizzas.price),2)
    AS total_sales
FROM
    order_details
        JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

IDENTIFY THE HIGHEST-PRICED PIZZA.

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

	name	price
▶	The Greek Pizza	35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

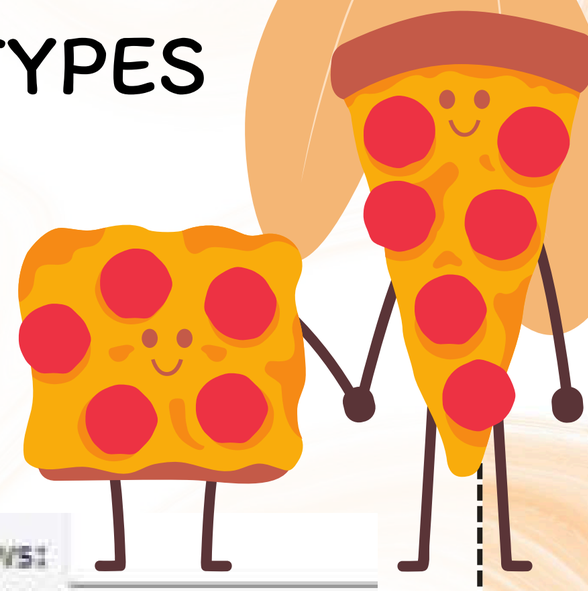
Result Grid | Filte

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

Limit to 1000 rows

```sql
-- Identify the most common pizza size ordered.
SELECT
    pizzas.size,
    COUNT(order_details.order_details_id)
    as order_count
FROM
    pizzas
        JOIN
    order_details
ON pizzas.pizza_id = order_details.pizza_id
GROUP BY pizzas.size
ORDER BY order_count DESC;
```

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.
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 pizzas.pizza_id = order_details.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
-- quantity of each pizza category ordered.
SELECT
    pizza_types.category,
    SUM(order_details.quantity)
    AS category_qantity
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 category_qantity DESC;
```

Result Grid | Filter Rows:

category	category_qantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

Result Grid

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

Limit to 1000 rows

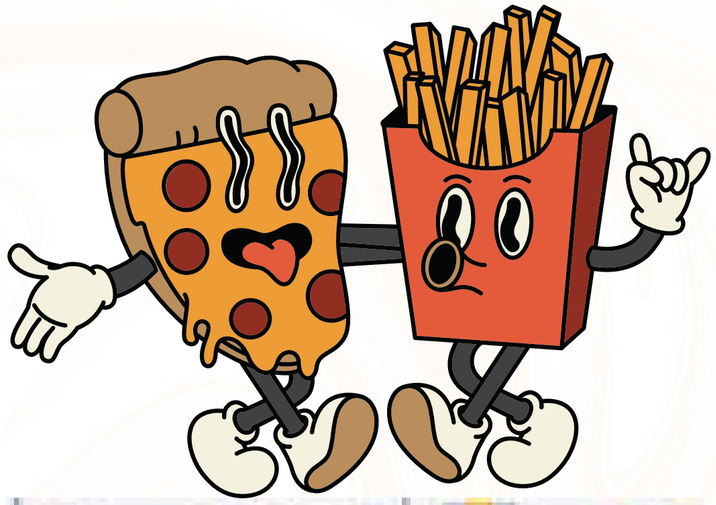
```sql
-- Determine the distribution of
-- orders by hour of the day.
SELECT
    HOUR(order_time) AS hours,
    COUNT(order_id) AS orders
FROM
    orders
GROUP BY hours;
```

JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

category	quantity_name
Chicken	6
Classic	8
Supreme	9
Veggie	9

```
1   -- Join relevant tables to find the
2   -- category-wise
3   -- distribution of pizzas.
4   SELECT
5       category, COUNT(name)
6       AS quantity_name
7   FROM
8       pizza_types
9   GROUP BY category;
```

GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.



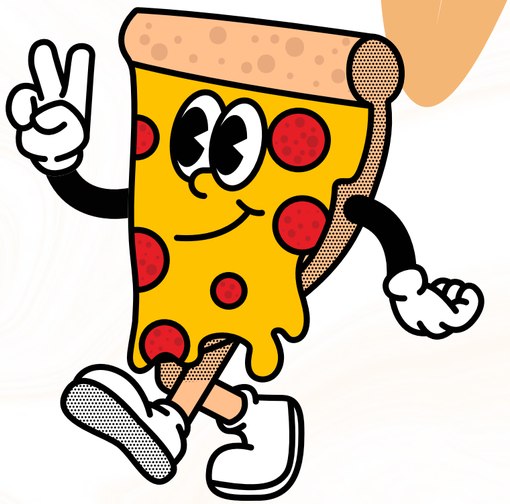
Result Grid

	average_orders
▶	138

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

SELECT
    ROUND(AVG(quantity)) AS average_orders
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 quantity_orders;
```

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,
    ROUND(SUM(order_details.quantity * pizzas.price),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.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.

Result Grid

category	revenue
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Limit to 1000 rows

```sql
-- Calculate the percentage contribution of each pizza type to total revenue.
select pizza_types.category,
round(sum(order_details.quantity* pizzas.price)/
(select round(sum(order_details.quantity * pizzas.price),2)
as total_sales
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;
```

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

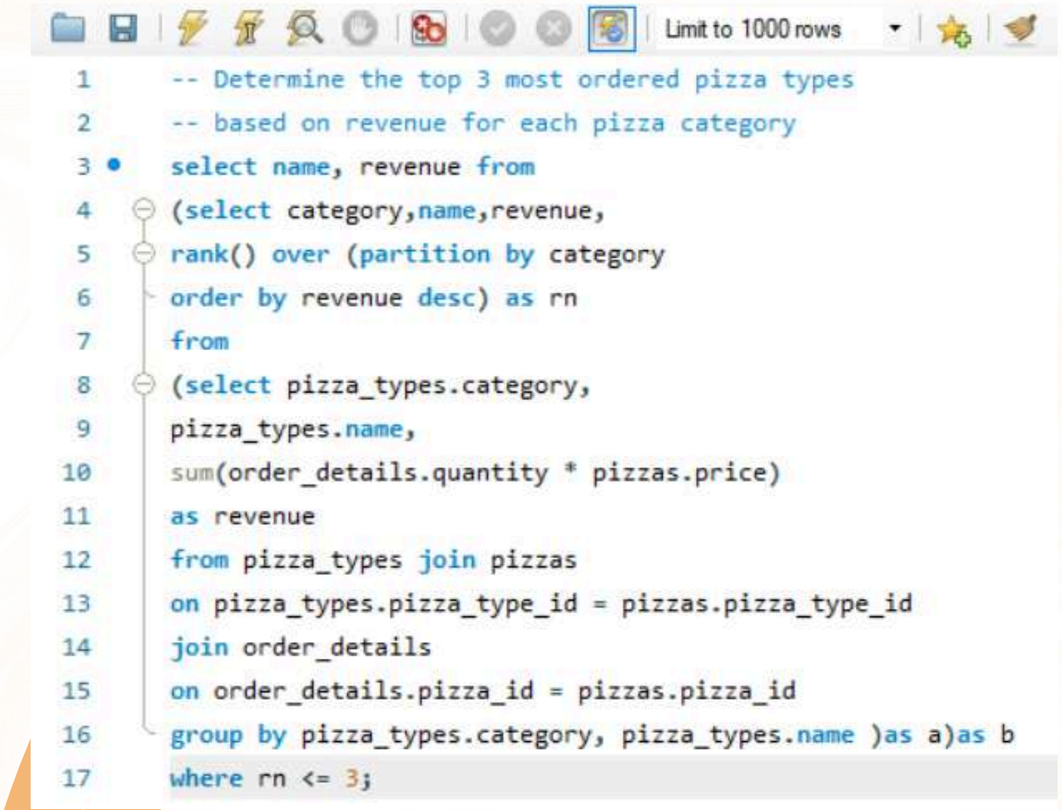
Result Grid | Filter Rows:

order_date	cum_revenue
2015-01-04	9864
2015-01-05	11930
2015-01-06	14358
2015-01-07	16561
2015-01-08	19399
2015-01-09	21526
2015-01-10	23990
2015-01-11	25863
2015-01-12	27782
2015-01-13	29831
2015-01-14	32359

Limit to 1000 rows

```sql
-- Analyze the cumulative revenue generated over time.
select order_date,
round(sum(revenue) over (order by order_date) ,0)as cum_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;
```

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.



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

THANK YOU