BORCELLE
RESTAURANT

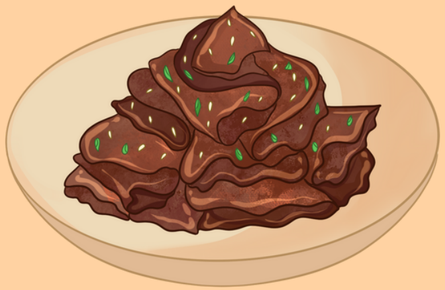
GRAB YOUR FAV. PIZZA

Pizza

BYTE

CATEGORY YOU PREFER

chicken



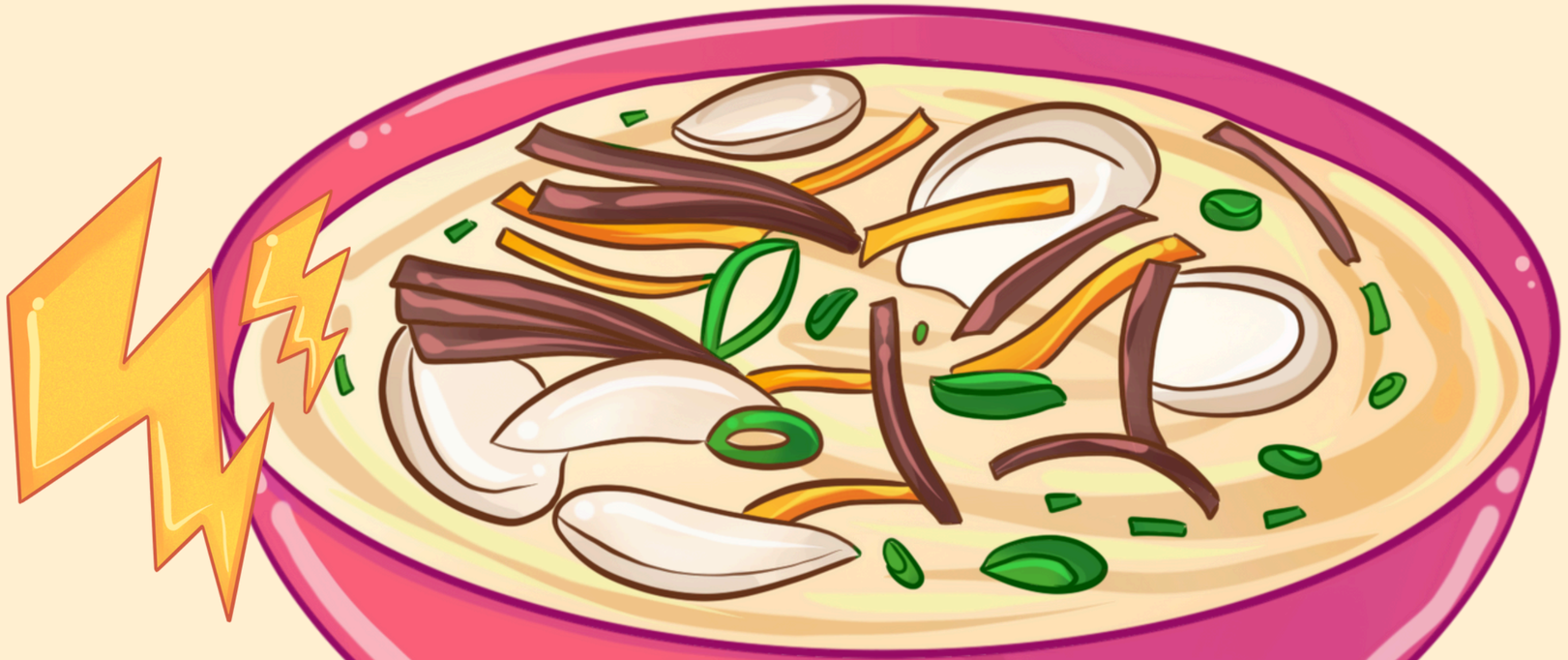
classic

veggie

supreme

ABOUT THE PIZZA SALES

"Pizza Byte’s SQL project focuses on analyzing sales data to provide valuable insights into inventory management, customer preferences, and peak sales periods. This comprehensive analysis helps in identifying top-selling pizzas and optimizing stock levels.



ABOUT

"Key queries for pizza sales include total sales calculation, top-selling items, sales trends over time, and customer purchase behavior,etc. SQL is crucial as it allows efficient data extraction, manipulation, and analysis from large databases. By executing these queries, businesses can gain insights into sales performance, optimize inventory, forecast demand, and make informed decisions, ultimately enhancing sales strategies and operational efficiency."

QUERY -Retrieve the total number of order placed.

```
select count(order_id) as total_orders from orders;
```

total_orders
21350

QUERY-Calculate the total revenue generated from pizza sales.(quantity*price)

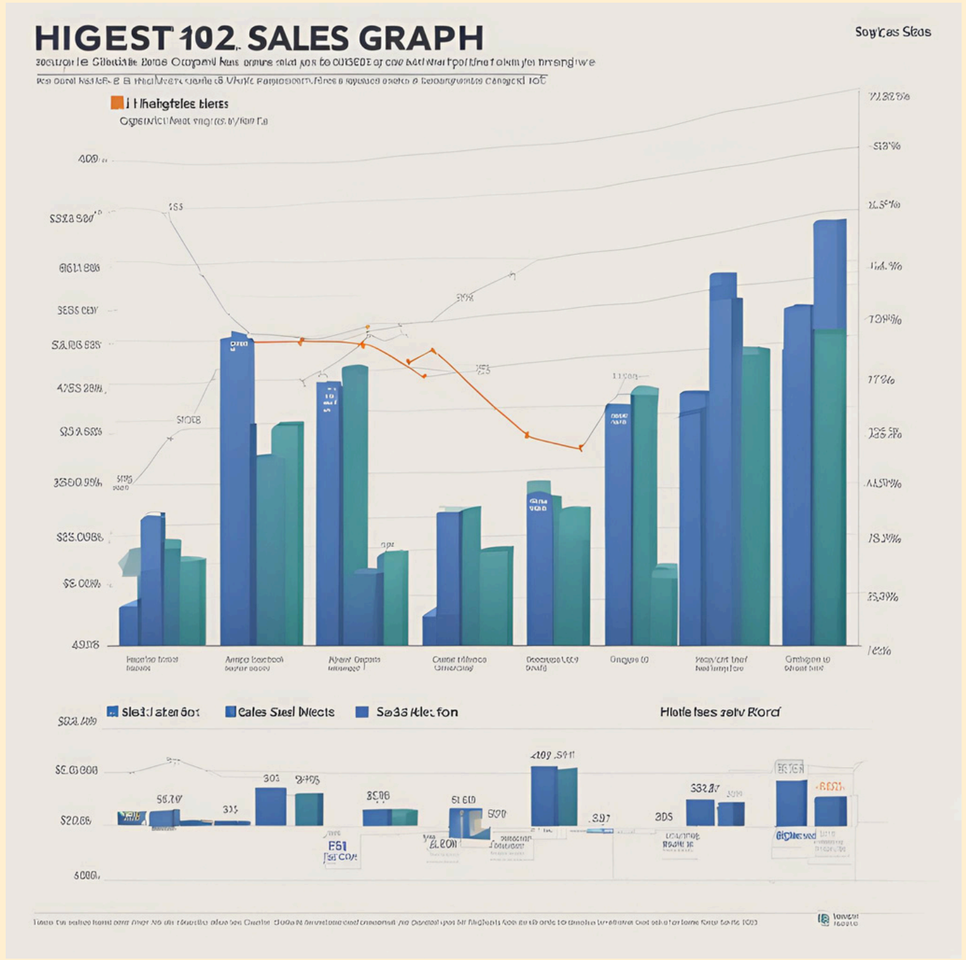
```
select * from order_details LIMIT 10;

describe order_details;

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;
```

total_sales
817860.05

pizza time

QUERY- 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;
```



OUTPUT

	name	price
▶	The Greek Pizza	35.95

QUERY- 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 ;
```

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

QUERY-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 ;
```

OUTPUT

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

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

```sql
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;
```

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

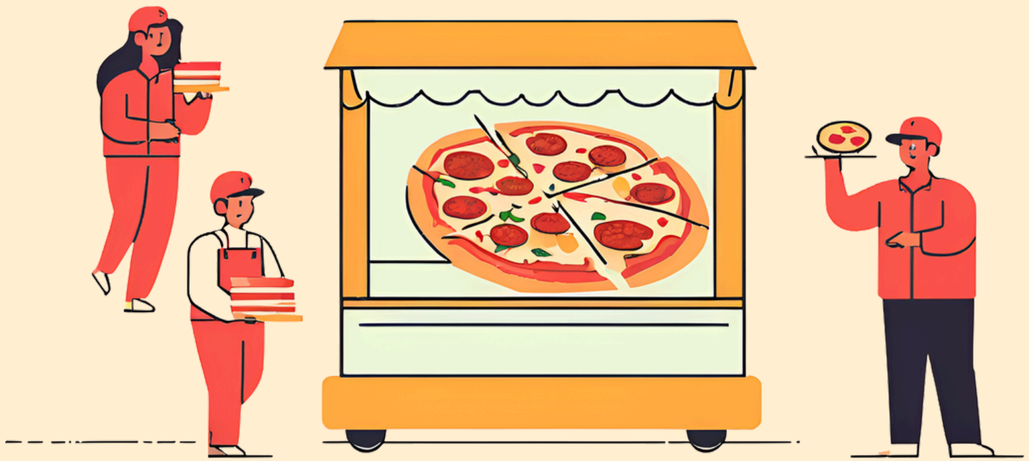
QUERY-Determine the distribution of orders by hour of the day.

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

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642

QUERY-Group the orders by date and calculate the average number of pizzas ordered pr day.

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



OUTPUT

avg_quantity
138

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

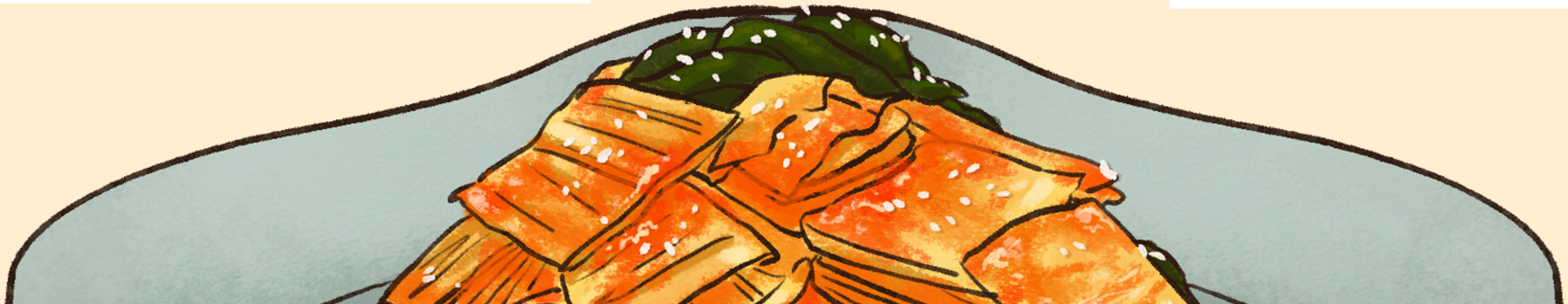
```sql
select pizza_types.name,
sum(orders_details.quantity * pizzas.price) as revenue
from pizza_types join pizzas
on pizzas.pizza_type_id = pizza_types.pizza_type_id
join orders_details
on orders_details.pizza_id = pizzas.pizza_id
group by pizza_types.name order by revenue desc limit 3;
```

name	revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

QUERY-Calculate the percentage contribution of each pizza type to total revnue.

```sql
select pizza_types.category,
(sum(orders_details.quantity * pizzas.price) / (SELECT ROUND(SUM(orders_details.quantity * pizza
FROM orders_details
JOIN pizzas ON pizzas.pizza_id = orders_details.pizza_id) )*100 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;
```

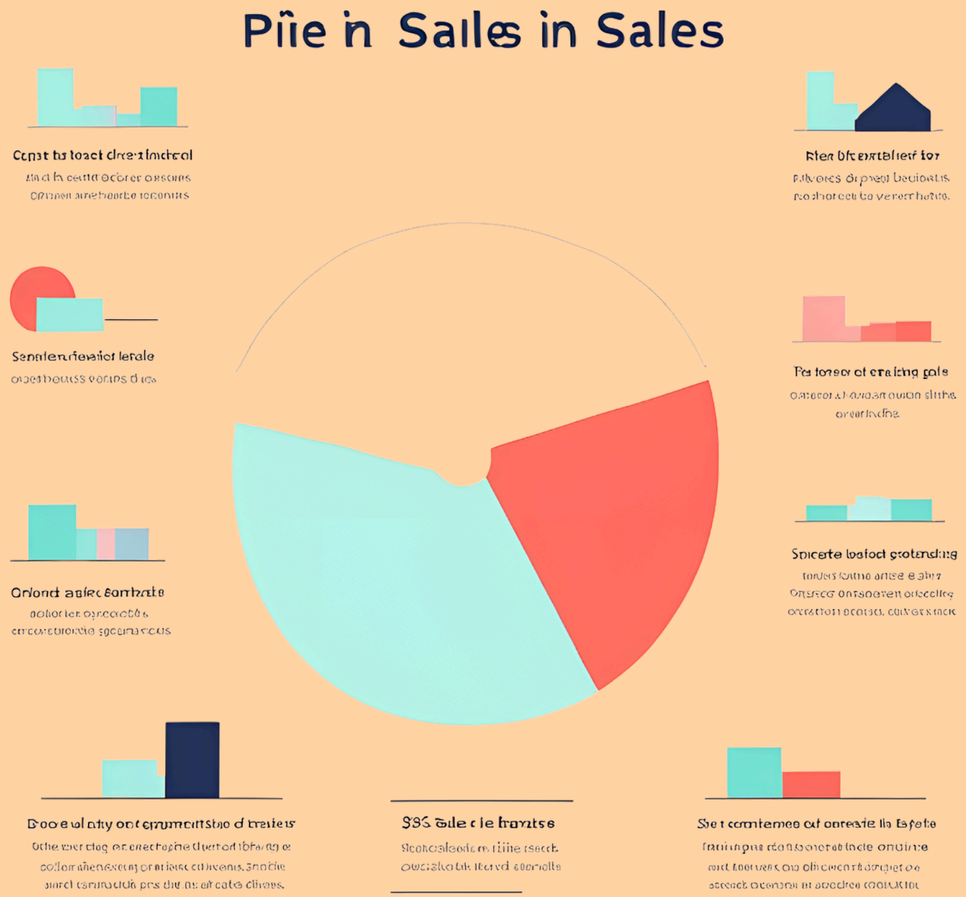
category	revenue
Classic	26.90596025566967
Supreme	25.45631126009862
Chicken	23.955137556847287
Veggie	23.682590927384577

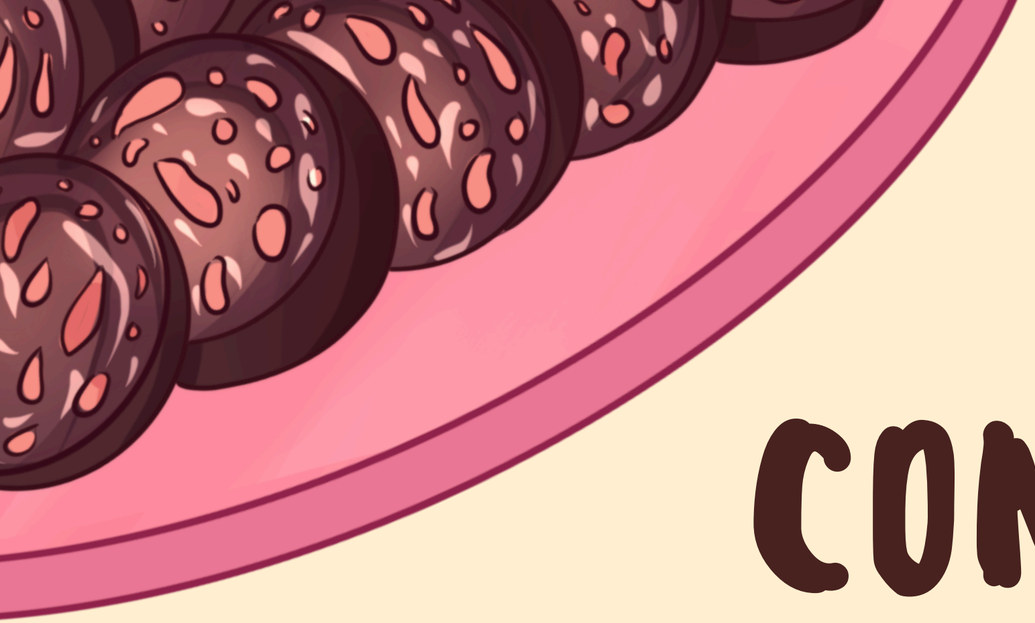


QUERY-Analyze the cumulative revenue generated over time.

```
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;
```

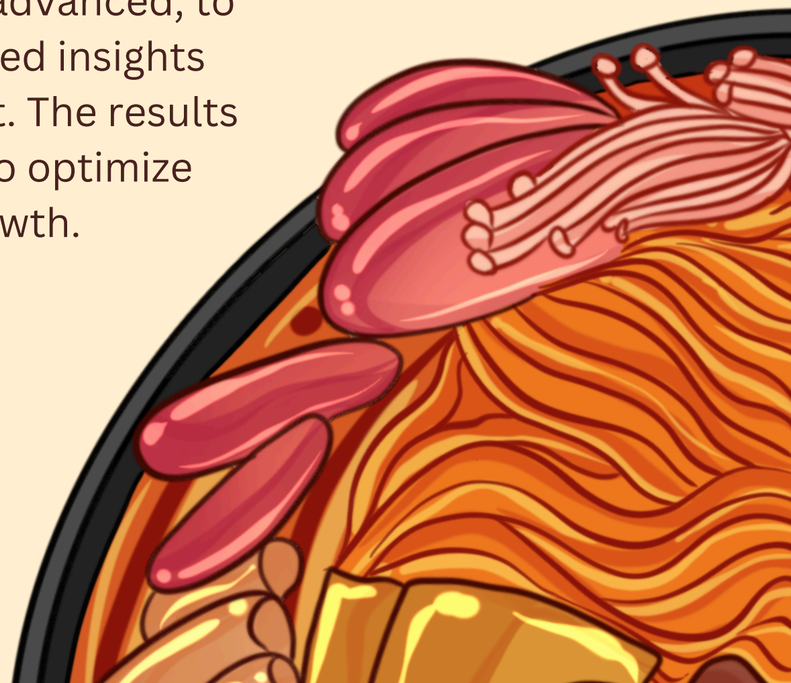
order_date	cum_revenue
2015-01-01	2713.8500000000004
2015-01-02	5445.75
2015-01-03	8108.15
2015-01-04	9863.6
2015-01-05	11929.55
2015-01-06	14358.5
2015-01-07	16560.7





CONCLUSION

In conclusion, our SQL project leveraged a range of queries, from basic to advanced, to thoroughly analyze sales data. This approach enabled us to extract detailed insights into sales performance, customer preferences, and inventory management. The results provided a robust foundation for data-driven decision-making, helping to optimize operations, enhance customer satisfaction, and drive business growth.




Thank you!
Vaibhavi Joshi