PIZZA SALES

SQL-BASED ANALYSIS OF PIZZA SALES DATA

BY - SHEIKH MOHMED SAMEER

Introduction

Project Overview:

This project focuses on analyzing sales data from PizzaHut to derive meaningful business insights such as total revenue, order patterns, and popular pizza types.

Data Description:

The dataset contains information about pizza types, orders, prices, and quantities.
Data tables used: pizza_types, pizzas, orders_details,orders.

Objective:

To understand the performance of various pizza types and categories based on sales revenue. Identify top-selling pizzas, revenue distribution, and order trends that can be used for business decision-making.

Tools Used:

SQL for querying and analyzing the data.
MySQL Workbench (or any other SQL environment).
Canvas for creating this presentation.

Scope of Analysis:

Basic metrics like total orders and revenue.
Intermediate queries for distribution analysis.
Advanced metrics for deeper insights into pizza sales performance.

Questions

Basic:

Retrieve the total number of orders placed.
Calculate the total revenue generated from pizza sales.
Identify the highest-priced pizza.
Identify the most common pizza size ordered.
List the top 5 most ordered pizza types along with their quantities.

Intermediate:

Join the necessary tables to find the total quantity of each pizza category ordered.
Determine the distribution of orders by hour of the day.
Join relevant tables to find the category-wise distribution of pizzas.
Group the orders by date and calculate the average number of pizzas ordered per day.
Determine the top 3 most ordered pizza types based on revenue.

Advanced:

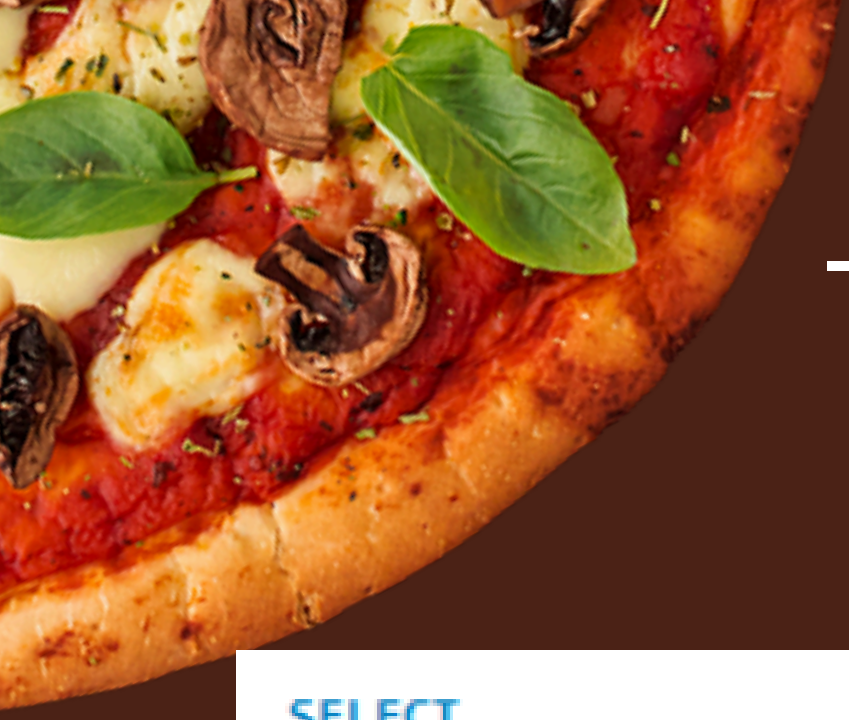
Calculate the percentage contribution of each pizza type to total revenue.
Analyze the cumulative revenue generated over time.
Determine the top 3 most ordered pizza types based on revenue for each pizza category.

-- Retrieve the total number of orders placed.

```
-- 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(orders_details.quantity * pizzas.price),
            2) AS total_sales
FROM
    orders_details
        JOIN
pizzas ON orders_details.pizza_id = pizzas.pizza_id
```

Result Grid

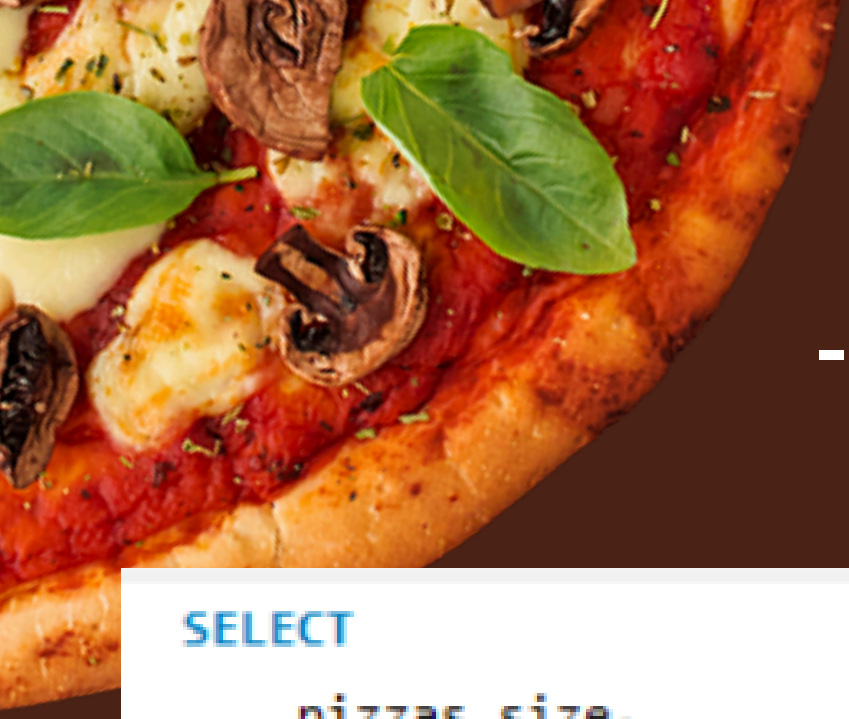
	total_sales
▶	817860.05

-- 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

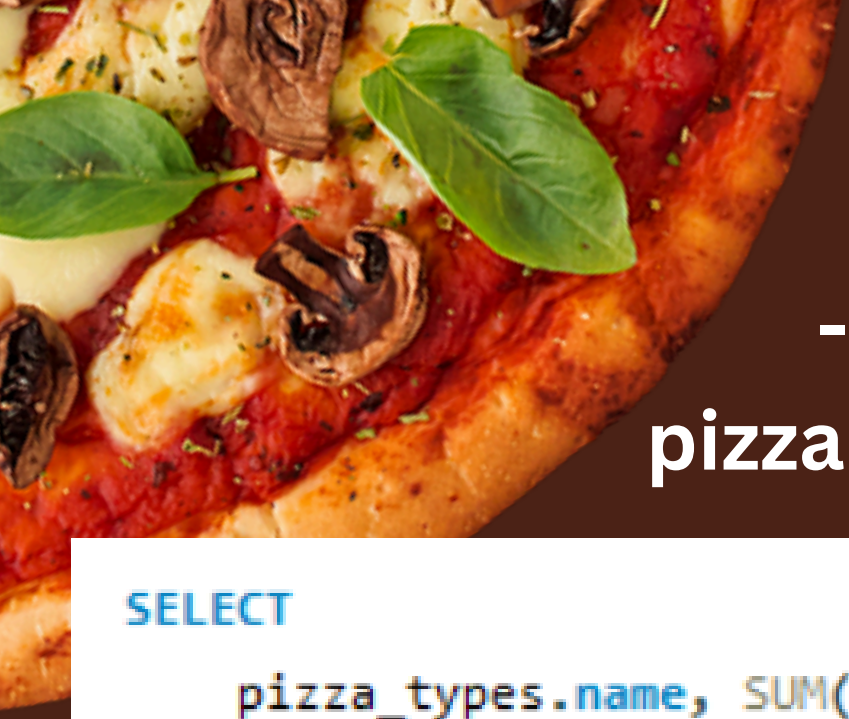


-- Identify the most common pizza size ordered.

```sql
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 | Filter

	size	order_count
▶	L	18526

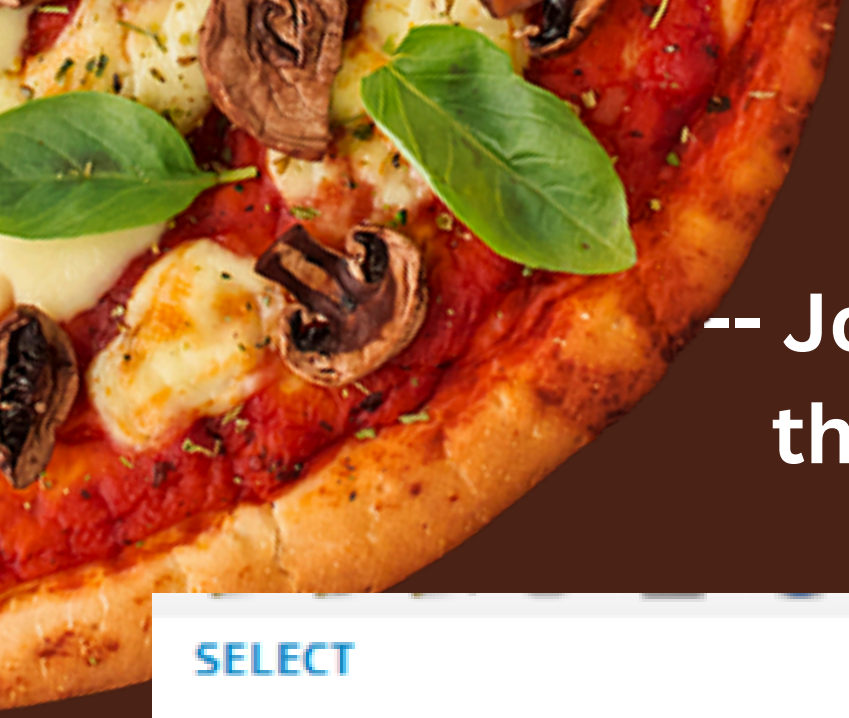


-- List the top 5 most ordered pizza types along with their quantities.

```sql
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;
```

Result Grid | Filter Rows:

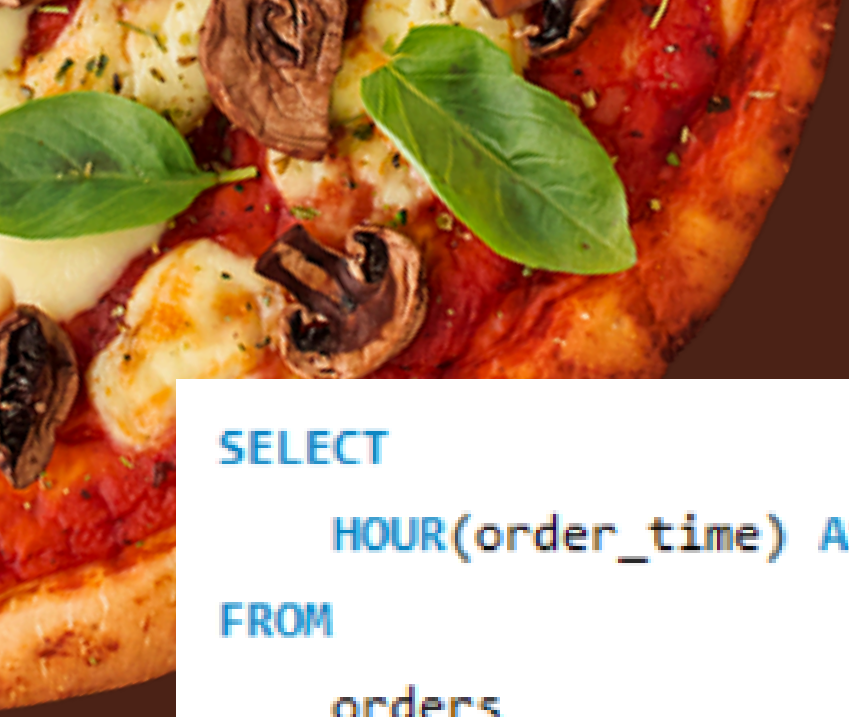
name	quantity
The Classic Deluxe Pizza	2453
The Barbecue Chicken Pizza	2432
The Hawaiian Pizza	2422
The Pepperoni Pizza	2418
The Thai Chicken Pizza	2371



-- Join the necessary tables to find the total quantityof 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



-- 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);
```

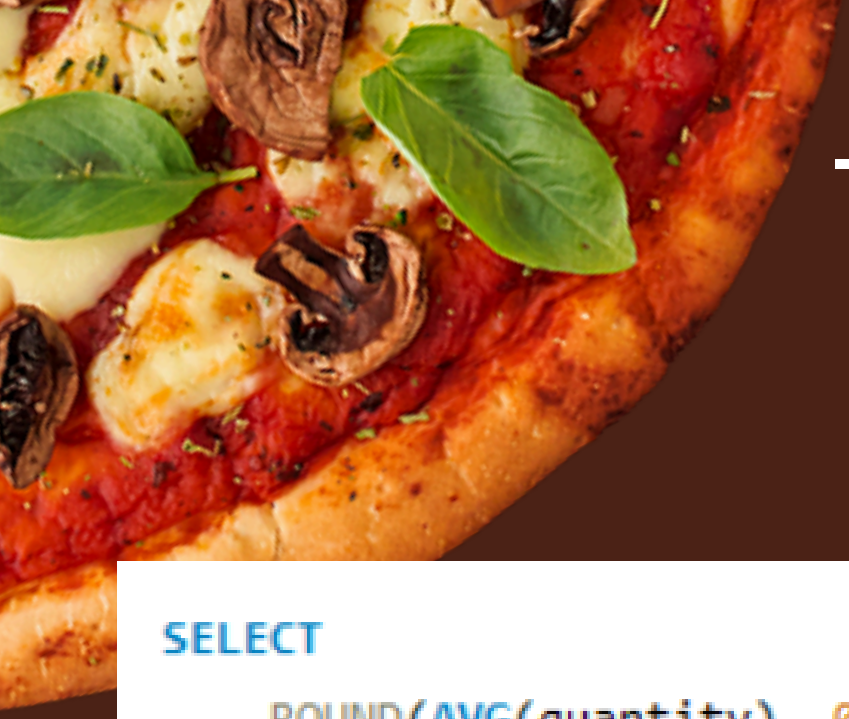
Result Grid | Filter Ro

hour	order_count
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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

```sql
select category, count(name) from pizza_types
group by category;
```

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

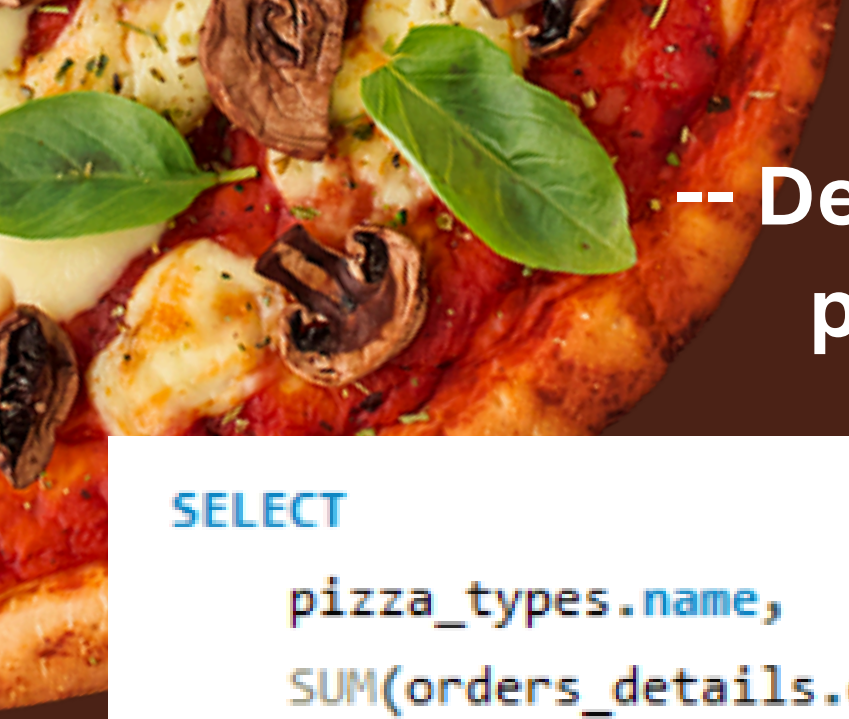


-- Group the orders by date and calculate the average number of pizzas ordered per day.

```sql
SELECT
    ROUND(AVG(quantity), 0) 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;
```

Result Grid | Filter Rows:

avg_pizza_ordered_per_day
138



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

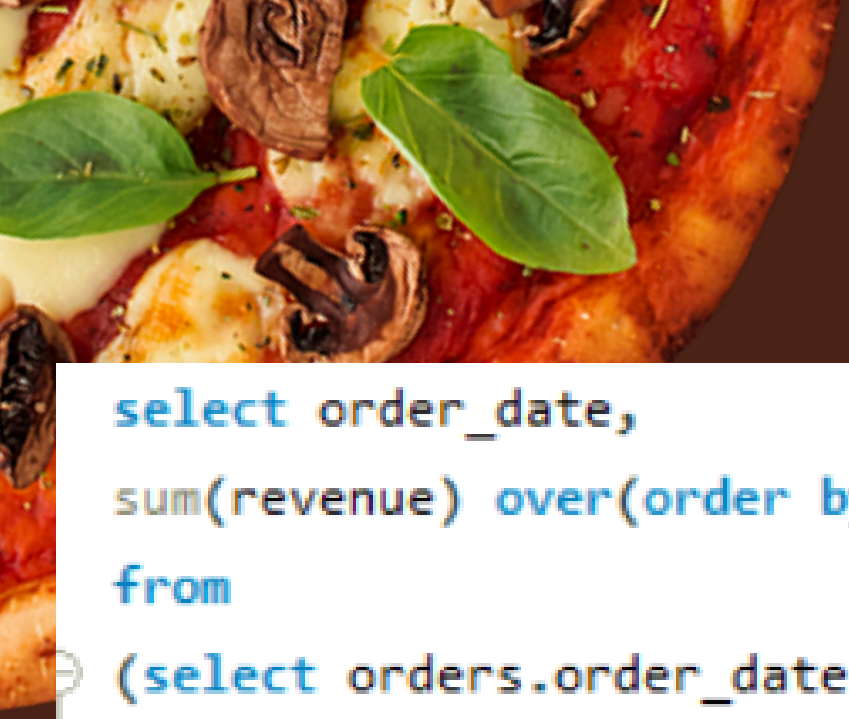
```sql
SELECT
    pizza_types.name,
    SUM(orders_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
        JOIN
    orders_details ON orders_details.pizza_id = pizzas.pizza_id
GROUP BY pizza_types.name
ORDER BY revenue DESC
LIMIT 3;
```

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

-- Calculate the percentage contribution of each pizza type

```sql
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 orders_details.pizza_id = pizzas.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;
```

	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 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
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

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

```sql
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;
```

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

Results and Insights :

As company is getting less orders between 8PM to 12 AM ,so to increase sales during the lower-traffic hours ,
the store can consider implementing the following strategies:

- Offer Special Discounts: Introduce "Late-Night Happy Hours" with discounts on specific pizza types or categories (e.g., 20% off on all pizzas between 8 PM to 12 AM).

- Bundle Deals: Create combo offers, such as “Buy 1 Get 1 Free” or family packs, that are available exclusively during the low-sales hours.

- Loyalty Points: Offer double loyalty points for orders placed during the late hours to encourage repeat customers.

--THANK YOU !!!! --