The Great Pizza Analy Challenge

SQL-Based Data Analysis Mini Project By Asha Maddi

Agenda

Overview of Key Phases and Insights

Data Exploration

In this phase, we will focus on analyzing the dataset, identifying trends, and uncovering meaningful insights that will guide our SQL analysis and business decision-making.

SQL Analysis

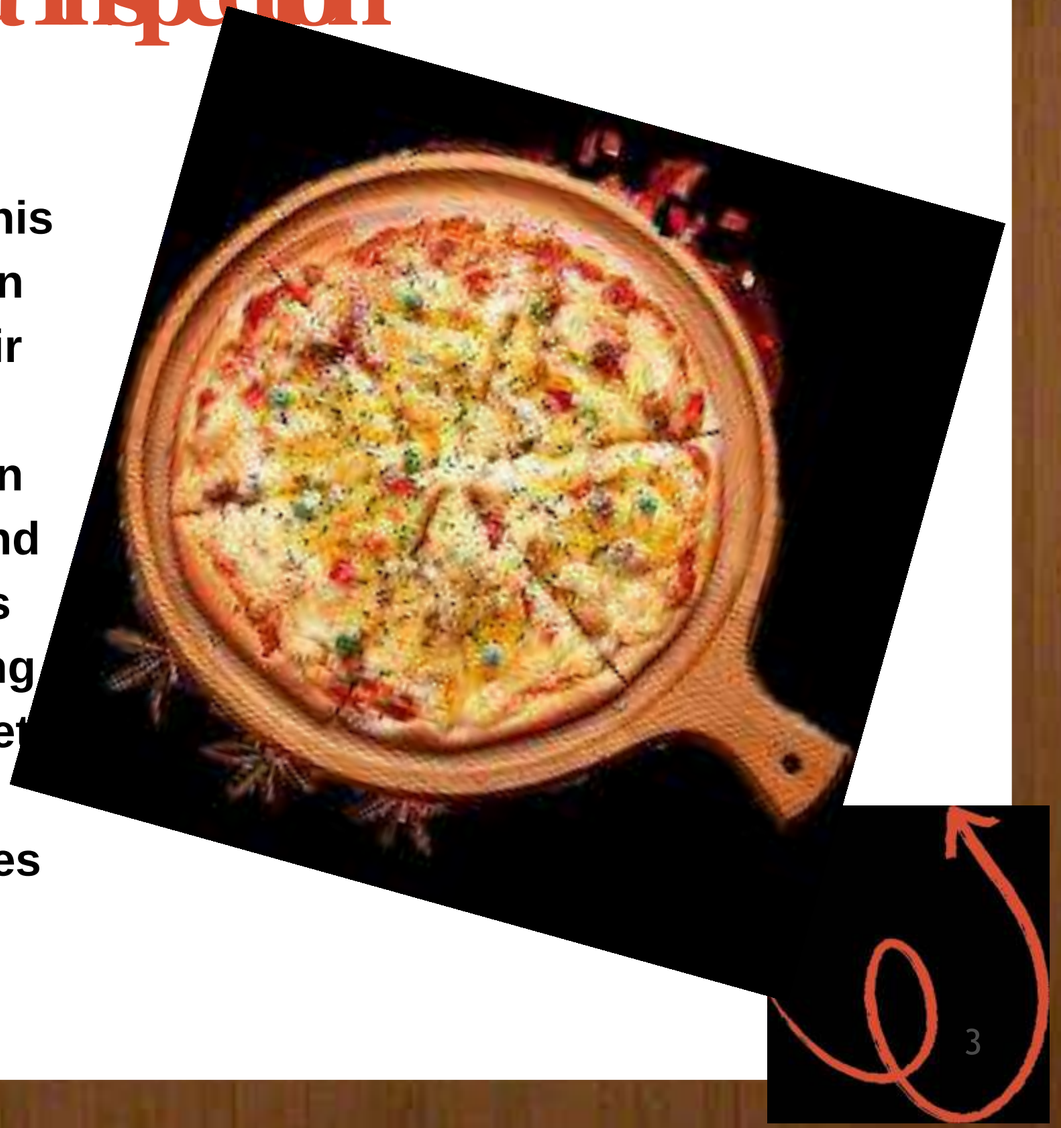
This section emphasizes applying SQL queries to extract, manipulate, and aggregate data, allowing us to derive actionable insights from our dataset for informed business strategies.

Business Insights

We will interpret the results from our analysis, discussing their implications and how they can influence decision-making processes, ultimately driving growth and innovation in our pizza analytics challenge.

PHASE 1: Foundation & Data Inspection

Begin by installing the database and conducting essential quality checks to verify its integrity. During this phase, it is crucial to explore unique categories within the dataset, such as different types of pizzas and their respective attributes— Pay particular attention to handling any missing data, as incomplete records can skew analysis results. It's also important to identify and address any records that lack pricing information, as this is a critical component for most analyses involving sales or profitability. By carefully preparing the dataset through these steps, you set a solid foundation for accurate and reliable analysis in the subsequent stages of your project.



Q1.List all unique pizza categories (DISTINCT).

```
SELECT DISTINCT category from pizza_types;;
```

Result Grid

	category
▶	Chicken
	Classic
	Supreme
	Veggie

Q2: Display pizza_type_id,name and ingredients, replacing NULL ingredients withMissing Data", Show first 5 rows.

```
SELECT pizza_type_id, name, COALESCE(ingredients, 'Missing Data') AS ingredients
FROM pizza_types LIMIT 5;
```

pizza_type_id	name	ingredients
bbq_ckn	The Barbecue Chicken Pizza	Barbecued Chicken, Red Peppers, Green Peppe...
cali_ckn	The California Chicken Pizza	Chicken, Artichoke, Spinach, Garlic, Jalapeno P...
ckn_alfredo	The Chicken Alfredo Pizza	Chicken, Red Onions, Red Peppers, Mushrooms...
ckn_pesto	The Chicken Pesto Pizza	Chicken, Tomatoes, Red Peppers, Spinach, Garl...
southw_ckn	The Southwest Chicken Pizza	Chicken, Tomatoes, Red Peppers, Red Onions, ...

Q3: Check for pizzas missing a price (IS NULL).

```
SELECT pizza_id ,price
from pizzas WHERE price IS NULL;
```

sdd

PHASE 2: Filtering & Exploratory SQL Analysis

SQL filtering techniques help answer business questions by extracting orders based on dates and times. Filtering pizzas by size, keyword, and price range tailors offerings, enhancing satisfaction and profitability. This sharpens your skills in refining datasets and conducting analyses.

Q1: Orders placed on '2015-01-01'

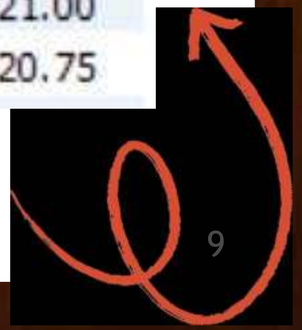
```
SELECT * from orders
WHERE date = '01-01-2015';
```

order_id	date	time
1	01-01-2015	11:38:36
2	01-01-2015	11:57:40
3	01-01-2015	12:12:28
4	01-01-2015	12:16:31
5	01-01-2015	12:21:30
6	01-01-2015	12:29:36

Q2: List pizzas with price descending.

```
SELECT pt.pizza_type_id,pt.name,pt.category,p.size,p.price
    FROM pizza_types AS pt LEFT JOIN pizzas AS p ON
        pt.pizza_type_id=p.pizza_type_id
    ORDER BY p.price desc;
```

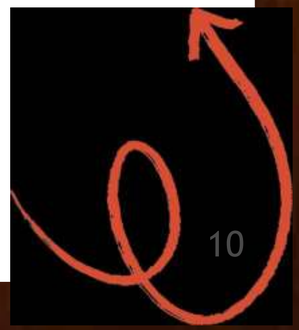
pizza_type_id	name	category	size	price
the_greek	The Greek Pizza	Classic	XXL	35.95
the_greek	The Greek Pizza	Classic	XL	25.50
brie_carre	The Brie Carre Pizza	Supreme	S	23.65
ital_veggie	The Italian Vegetables Pizza	Veggie	L	21.00
bbq_ckn	The Barbecue Chicken Pizza	Chicken	L	20.75



Q3: Pizzas sold in sizes 'L' or 'XL'

```
SELECT * FROM pizzas WHERE size IN('L','XL');
```

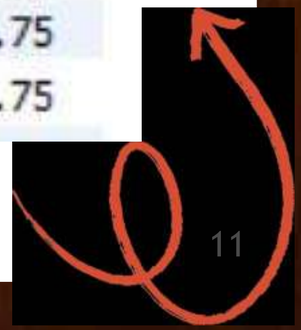
pizza_id	pizza_type_id	size	price
bbq_ckn_l	bbq_ckn	L	20.75
cali_ckn_l	cali_ckn	L	20.75
ckn_alfredo_l	ckn_alfredo	L	20.75
ckn_pesto_l	ckn_pesto	L	20.75
southw_ckn_l	southw_ckn	L	20.75



Q4: Pizzas priced between $15.00 and $17.00.

```
SELECT pt.pizza_type_id,pt.name,p.size,p.price
FROM pizza_types AS pt LEFT JOIN pizzas AS p ON
pt.pizza_type_id=p.pizza_type_id
WHERE price BETWEEN 15.00 AND 17.00;
```

pizza_type_id	name	size	price
bbq_ckn	The Barbecue Chicken Pizza	M	16.75
cali_ckn	The California Chicken Pizza	M	16.75
ckn_alfredo	The Chicken Alfredo Pizza	M	16.75
ckn_pesto	The Chicken Pesto Pizza	M	16.75
southw_ckn	The Southwest Chicken Pizza	M	16.75

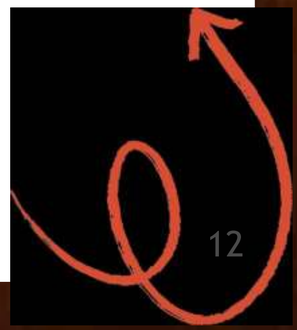


Q5: Pizzas with "Chicken" in the name.

```
SELECT name AS Pizza_name from pizza_types
    WHERE name LIKE '%Chicken%';
```

sdd

Pizza_name
The Barbecue Chicken Pizza
The California Chicken Pizza
The Chicken Alfredo Pizza
The Chicken Pesto Pizza
The Southwest Chicken Pizza



Q6: Orders on '2015-02-15' or placed after 8 PM.

```
SELECT * from orders
WHERE DATE = '15-02-2015' AND time >'20:00:00';
```

order_id	date	time
2769	15-02-2015	20:14:22
2770	15-02-2015	20:41:38
2771	15-02-2015	20:54:46
2772	15-02-2015	21:02:21
2773	15-02-2015	21:02:35

PHASE 3: Sales Performance & Business Insights

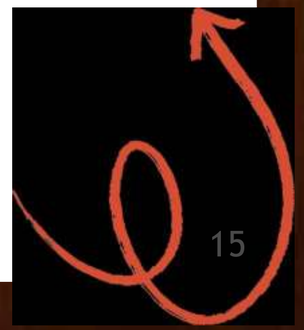
SQL queries are vital for extracting insights from sales data. You calculate total sales and analyze category performance to identify which product categories contribute most.

Detecting pizzas that were never ordered uncovers menu gaps for optimization.

Q1:Total quantity of pizzas sold (SUM).

```
SELECT SUM(quantity) AS Total_pizzas_sold
FROM order_details;
```

	Total_pizzas_sold
▶	49574



Q2: Average pizza price (AVG).

```
SELECT
ROUND(AVG(price),2) AS  Avg_pizza_price
FROM pizzas;
```

sdd

Result Grid

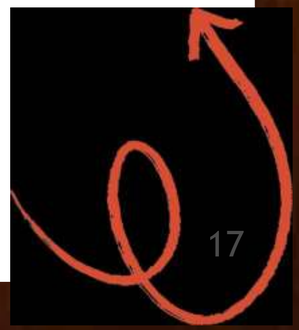
Avg_pizza_price
16.44

Q3: Total order value per order (JOIN, SUM, GROUP BY).

```
SELECT od.order_id,
SUM(od.quantity)*SUM(p.price) AS Total_value
FROM Pizzas p JOIN order_details od
ON od.pizza_id=p.pizza_id GROUP BY od.order_id
ORDER BY order_id;
```

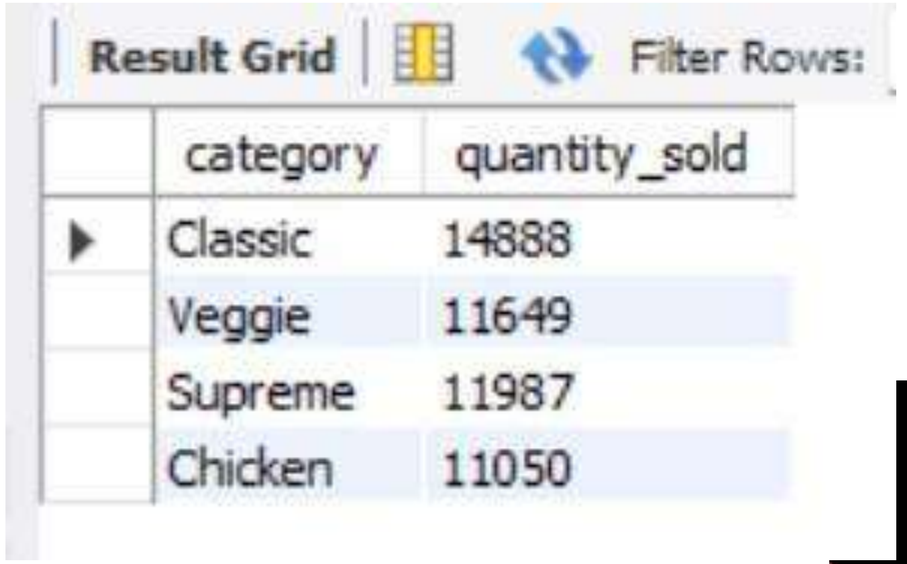
Result Grid | Filter Row

order_id	Total_value
1	13.25
2	460.00
3	74.50
4	16.50
5	16.50
6	49.50
7	12.50



Q4: Total quantity sold per pizza category (JOIN, GROUP BY).

```
SELECT DISTINCT pt.category,
SUM(od.quantity) AS quantity_sold
FROM pizzas p JOIN order_details od
ON p.pizza_id=od.pizza_id
JOIN pizza_types pt ON
pt.pizza_type_id=p.pizza_type_id GROUP BY
pt.category;
```



Q5: Categories with more than 5,000 pizzas sold (HAVING).

```
SELECT DISTINCT pt.category,
SUM(od.quantity) AS quantity_sold
FROM pizzas p JOIN order_details od
ON p.pizza_id=od.pizza_id
JOIN pizza_types pt ON pt.pizza_type_id=p.pizza_type_id
GROUP BY pt.category
HAVING quantity_sold > 5000;
```

Result Grid | Filter Rows:

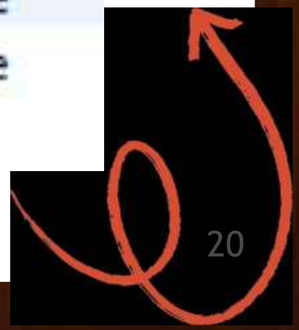
category	quantity_sold
Classic	14888
Veggie	11649
Supreme	11987
Chicken	11050

Q6: Pizzas never ordered (LEFT/RIGHT JOIN).

```
SELECT pt.pizza_type_id, pt.name ,pt.category
    FROM pizza_types pt LEFT JOIN pizzas p
        ON pt.pizza_type_id=p.pizza_type_id
            LEFT JOIN order_details od ON
                p.pizza_id=od.pizza_id
            WHERE od.pizza_id IS NULL;
```

Result Grid | Filter Rows:

pizza_type_id	name	category
big_meat	The Big Meat Pizza	Classic
big_meat	The Big Meat Pizza	Classic
five_cheese	The Five Cheese Pizza	Veggie
five_cheese	The Five Cheese Pizza	Veggie
four_cheese	The Four Cheese Pizza	Veggie



Q7: Price differences between different sizes of the same pizza (SELF JOIN).

```
SELECT  p1.pizza_type_id, p1.size AS from_size,
        p2.size AS to_size,
    ABS(p1.price - p2.price) AS price_difference
    FROM pizzas p1 JOIN pizzas p2
     ON p1.pizza_type_id = p2.pizza_type_id
     AND p1.pizza_id < p2.pizza_id
    ORDER BY price_difference DESC;
```

Result Grid | Filter Rows:

pizza_type_id	name	category
big_meat	The Big Meat Pizza	Classic
big_meat	The Big Meat Pizza	Classic
five_cheese	The Five Cheese Pizza	Veggie
five_cheese	The Five Cheese Pizza	Veggie
four_cheese	The Four Cheese Pizza	Veggie

Thank You

For your attention and interest

Thank You

For your attention and interest