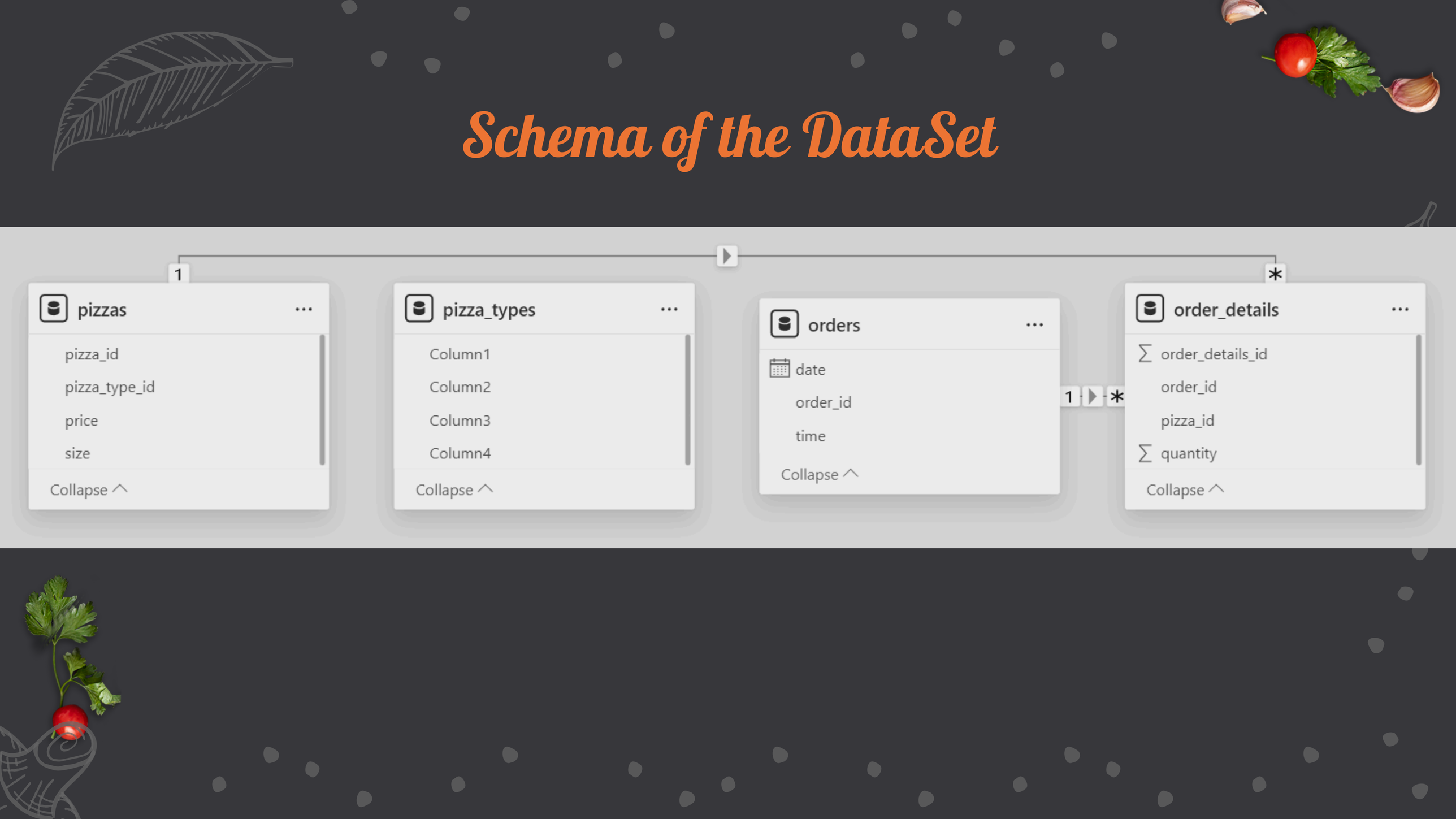
Slicing through Profits

An SQL Pizza Project

CIAO !

Ever wonder what sells best in the pizza world? I'm Vaibhav, and I used SQL to uncover the secrets of pizza sales. Let's get cheesy!

Schema of the DataSet



Retrieve the total number of orders placed.

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

	total_orders
▶	21350


Calculate the total revenue generated from pizza sales

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

total_revenue
569248.2

Identify the highest price pizza.

```
1   --  Identify the highest price pizza.
2
3 • SELECT
4       pizza_types.name, pizzas.price
5   FROM
6       pizza_types
7           JOIN
8       pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9   ORDER BY pizzas.price desc
10  LIMIT 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered

```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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

size	order_count
L	12890
M	10716
S	9821
XL	395
XXL	19

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

```sql
-- List the Top 5 ordered pizza types along with their quantities

SELECT
    pizza_types.name, SUM(order_details.quantity) AS quantity
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 quantity DESC
LIMIT 5;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	quantity
The Barbecue Chicken Pizza	1727
The Pepperoni Pizza	1689
The Classic Deluxe Pizza	1678
The Hawaiian Pizza	1655
The California Chicken Pizza	1647

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(order_details.quantity) AS quantity
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 quantity DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category	quantity
Classic	10340
Supreme	8330
Veggie	8171
Chicken	7654

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 order_count
FROM
    orders
GROUP BY HOUR(order_time)
ORDER BY order_count DESC;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

hours	order_count
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
10	8
9	1

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

```sql
-- join relavent tables to find the
-- category wise distribution of pizzas.

SELECT
    category, COUNT(name) types
FROM
    pizza_types
GROUP BY category
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category	types
Chicken	6
Classic	8
Supreme	9
Veggie	9

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
    orders.order_date, SUM(order_details.quantity) AS avg_qty
FROM
    orders
        JOIN
    order_details ON orders.order_id = order_details.order_id
GROUP BY orders.order_date
ORDER BY avg_qty DESC
LIMIT 10;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

order_date	avg_qty
2015-07-04	234
2015-07-03	213
2015-05-15	208
2015-07-24	196
2015-02-01	191
2015-08-14	188
2015-07-17	187
2015-06-01	184
2015-05-08	181
2015-05-29	181

Determine the top 3 most ordered pizza types ased on revenue

```sql
-- Determine the top 3 most ordered pizza types ased on revenue

SELECT
    *
FROM
    pizzahut.pizza_types;

SELECT
    pizza_types.name,
    SUM(order_details.quantity * pizzas.price) AS revenue
FROM
    pizza_types
        JOIN
    pizzas ON pizzas.pizza_type_id = pizza_types.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: | Export: | Wrap Cell Content:

name	revenue
The Barbecue Chicken Pizza	30307.25
The Thai Chicken Pizza	29498.5
The California Chicken Pizza	28743.25

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(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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category	revenue
Classic	26.88
Supreme	25.45
Veggie	23.87
Chicken	23.8

Analyse the cumulative revenue generated over time.

```sql
-- Analyse 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(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;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

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
-- Determmine 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((order_details.quantity) * pizzas.price) 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 , pizza_types.name) as a) as b
where rn <= 3;
select
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

name	revenue
The Barbecue Chicken Pizza	30307.25
The Thai Chicken Pizza	29498.5
The California Chicken Pizza	28743.25
The Classic Deluxe Pizza	26072
The Hawaiian Pizza	22076.75
The Pepperoni Pizza	21096
The Spicy Italian Pizza	24565.75
The Italian Supreme Pizza	23449
The Sicilian Pizza	21131.5
The Four Cheese Pizza	22334.550000000352
The Five Cheese Pizza	18666.5
The Mexicana Pizza	18384.25

Thanks!

Do you have any questions?

vaibhavmagdum1528@gmail.com

+91 8780054299

GITHUB : /pyvmag

CREDITS: This presentation template was created by **Vaibhav Magdum**, including icons by **Flaticon**, and infographics & images by **Freepik**