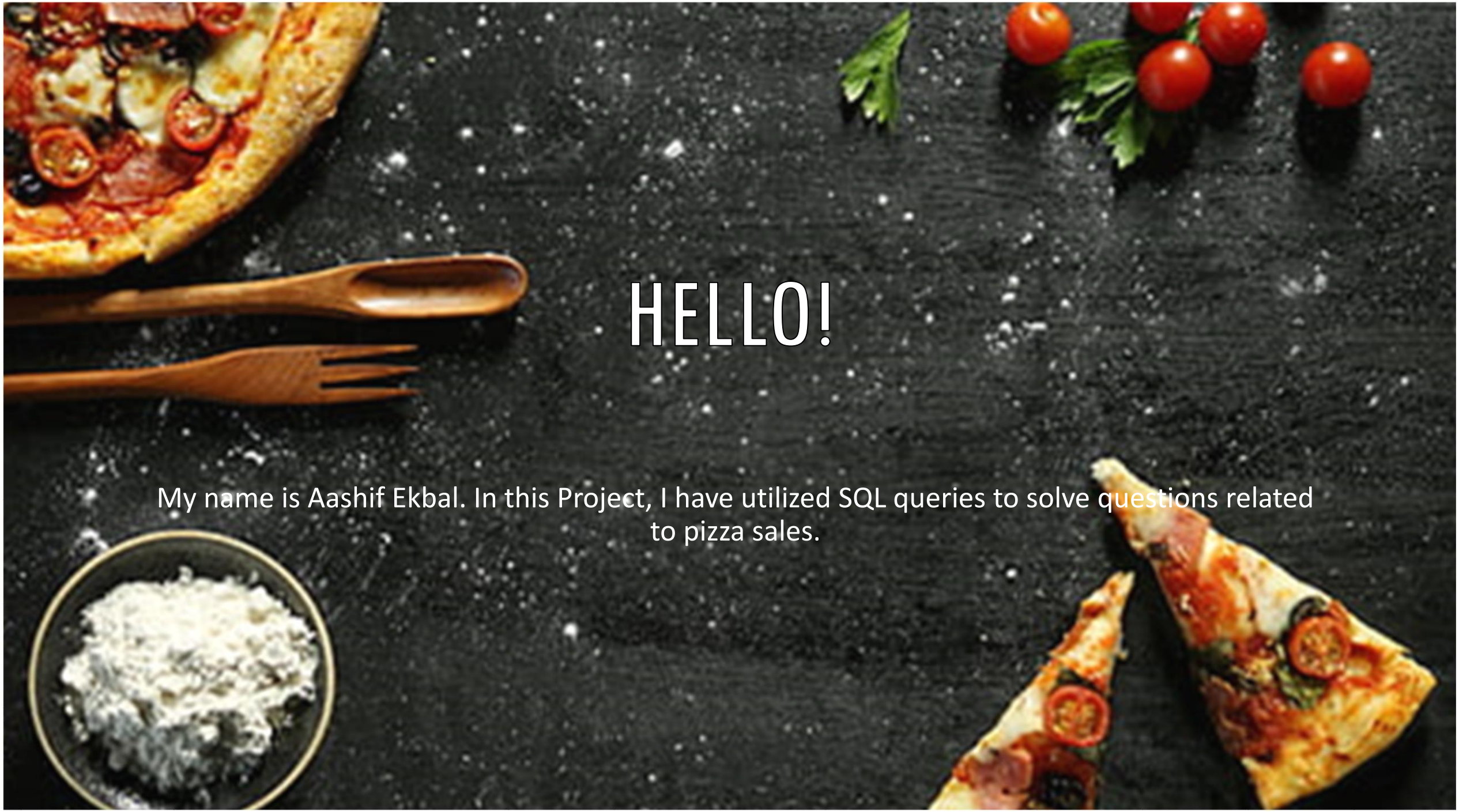
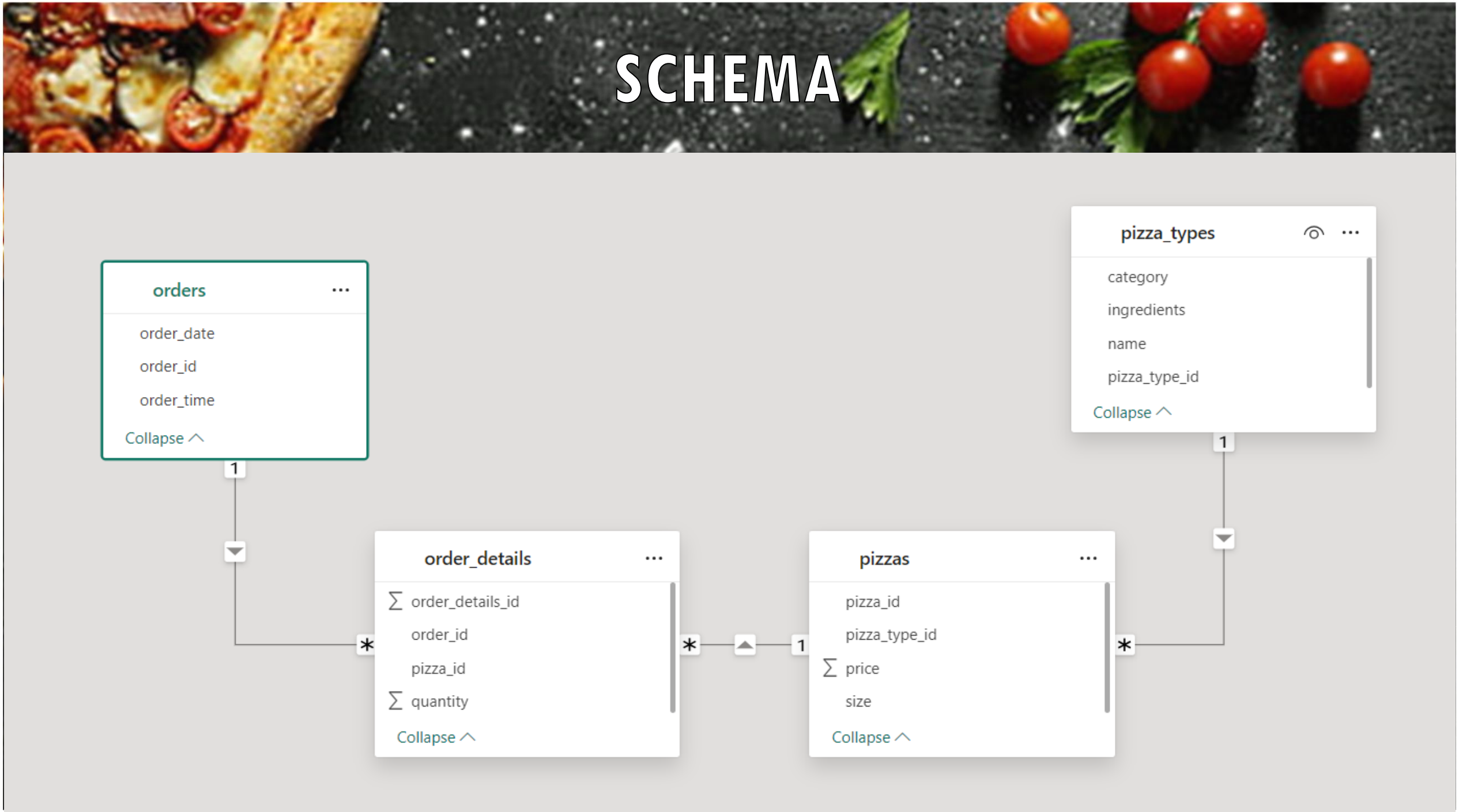

PIZZA SALES ANALYSIS
PRESENTED BY
AASHIF EKBAL


HELLO!
My name is Aashif Ekbal. In this Project, I have utilized SQL queries to solve questions related to pizza sales.


SCHEMA
orders
order_date
order_id
order_time
Collapse
1
order_details
order_details_id
order_id
pizza_id
quantity
Collapse
pizzas
pizza_id
pizza_type_id
price
size
Collapse
pizza_types
category
ingredients
name
pizza_type_id
Collapse
1

PIZZAS TABLE

Table: pizzas

Columns:

pizza_id	text
pizza_type_id	text
size	text
price	double

pizzas:

- **pizza_id**: Unique identifier for each pizza.
- **pizza_type_id**: Identifier linking to the type of pizza (foreign key referencing the pizza_types table).
- **size**: Size of the pizza (e.g., small, medium, large).
- **price**: Price of the pizza.

PIZZA_TYPES TABLE

Table: pizza_types

Columns:

pizza_type_id	text
name	text
category	text
ingredients	text

pizza_types:

- **pizza_type_id**: Unique identifier for each type of pizza.
- **name**: Name of the pizza type (e.g., Greek, Pepperoni, Italian).
- **category**: Category of the pizza (e.g., Chicken, Classic, Veggie).
- **ingredients**: Ingredients used in the pizza type.

ORDERS TABLE

Table: orders

Columns:

order_id	int
order_date	text
order_time	text

orders:

- **order_id**: Unique identifier for each order.
- **date**: Date when the order was placed.
- **time**: Time when the order was placed.

ORDER_DETAILS TABLE

Table: order_details

Columns:

order_details_id	int
order_id	int
pizza_id	text
quantity	int

order_details:

- **order_details_id**: Unique identifier for each order detail.
- **order_id**: Identifier linking to the order (foreign key referencing the orders table).
- **pizza_id**: Identifier linking to the pizza ordered (foreign key referencing the pizzas table).
- **quantity**: Quantity of the specific pizza ordered in the order.

Retrieve the total number of orders placed.

```
6 •    SELECT COUNT(order_id) AS Total_no_of_orders FROM orders;
```

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

Total_no_of_orders
21350

Calculate the total revenue generated from pizza sales.

```
6 •    SELECT ROUND(SUM(p.price * od.quantity),2) AS Total_revenue
7      FROM pizzas p
8      INNER JOIN order_details od
9      ON p.pizza_id = od.pizza_id;
```

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

Total_revenue
817860.05

Identify the highest-priced pizza.

```sql
SELECT pt.name, p.price
FROM pizzas p
INNER JOIN pizza_types pt
ON p.pizza_type_id = pt.pizza_type_id
WHERE p.price = (SELECT MAX(price) FROM pizzas) ;
```

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

name	price
The Greek Pizza	35.95

Identify the most common pizza size ordered.

```
6 •  SELECT p.size, COUNT(od.pizza_id) AS No_of_orders
7    FROM pizzas p
8    INNER JOIN order_details od
9    ON p.pizza_id = od.pizza_id
10   GROUP BY p.size
11   ORDER BY No_of_orders DESC ;
```

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

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

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

```
6 •   SELECT p.pizza_type_id, SUM(od.quantity) AS No_of_quantity_ordered
7     FROM pizzas p
8     INNER JOIN order_details od
9     ON p.pizza_id = od.pizza_id
10    GROUP BY p.pizza_type_id
11    ORDER BY No_of_quantity_ordered DESC
12    LIMIT 5 ;
```

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

pizza_type_id	No_of_quantity_ordered
classic_dlx	2453
bbq_ckn	2432
hawaiian	2422
pepperoni	2418
thai_ckn	2371

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

```
Limit to 1000 rows

 6 •  SELECT pt.category, SUM(od.quantity) AS Total_quantity_ordered
 7    FROM pizza_types pt
 8    INNER JOIN pizzas p
 9    ON pt.pizza_type_id = p.pizza_type_id
10    INNER JOIN order_details od
11    ON p.pizza_id = od.pizza_id
12    GROUP BY pt.category
13    ORDER BY Total_quantity_ordered DESC ;
```

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

category	Total_quantity_ordered
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 Total_no_of_orders
FROM orders
GROUP BY Hour
ORDER BY Total_no_of_orders DESC ;
```

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

Hour	Total_no_of_orders
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

Result 11 | Read Only

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

```
6 •   SELECT category, COUNT(pizza_type_id) AS No_of_pizza_varieties
7     FROM pizza_types
8     GROUP BY category
9     ORDER BY No_of_pizza_varieties DESC ;
10
```

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

category	No_of_pizza_varieties
Supreme	9
Veggie	9
Classic	8
Chicken	6

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

```sql
SELECT o.order_date, SUM(od.quantity) AS No_of_quantity_ordered
FROM orders o
INNER JOIN order_details od
ON o.order_id = od.order_id
GROUP BY order_date
ORDER BY No_of_quantity_ordered DESC ;
```

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

order_date	No_of_quantity_ordered
2015-11-26	266
2015-11-27	264
2015-10-15	262
2015-07-04	234
2015-07-03	213
2015-05-15	208
2015-07-24	196
2015-10-01	194
2015-02-01	191
2015-11-06	190
2015-08-14	188
2015-07-17	187
2015-06-01	184
2015-05-08	181
2015-05-29	181

esult 17 ×

Read Only

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

```sql
SELECT pt.pizza_type_id, pt.name, ROUND(SUM(p.price * od.quantity),2) AS Revenue
FROM pizza_types pt
INNER JOIN pizzas p
ON pt.pizza_type_id = p.pizza_type_id
INNER JOIN order_details od
ON p.pizza_id = od.pizza_id
GROUP BY p.pizza_type_id, pt.name
ORDER BY Revenue DESC
LIMIT 3 ;
```

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

pizza_type_id	name	Revenue
thai_ckn	The Thai Chicken Pizza	43434.25
bbq_ckn	The Barbecue Chicken Pizza	42768
cali_ckn	The California Chicken Pizza	41409.5

Calculate the percentage contribution of each pizza type to total revenue.

```sql
SELECT pt.category,
 CONCAT(ROUND((SUM(p.price * od.quantity)/
 (SELECT  SUM(od.quantity * p.price) AS Total_revenue
  FROM  pizzas p
  INNER JOIN order_details od
  ON p.pizza_id = od.pizza_id)*100),2), "%")
AS Percentage_contribution
FROM pizza_types pt
INNER JOIN pizzas p
ON pt.pizza_type_id = p.pizza_type_id
INNER JOIN order_details od
ON p.pizza_id = od.pizza_id
GROUP BY pt.category ;
```

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

category	Percentage_contribution
Classic	26.91%
Veggie	23.68%
Supreme	25.46%
Chicken	23.96%

Result 28 | Read Only

Analyze the cumulative revenue generated over time.

```sql
SELECT order_date,
SUM(Revenue_generated) OVER (ORDER BY order_date) AS Cum_revenue
FROM
(SELECT o.order_date, SUM(od.quantity * p.price) AS Revenue_generated
FROM  pizzas p
INNER JOIN order_details od
ON p.pizza_id = od.pizza_id
INNER JOIN orders o
ON o.order_id = od.order_id
GROUP BY o.order_date) AS revenue ;
```

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

Result 34 | Read Only

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

```sql
SELECT category, name, Revenue_generated
FROM
(SELECT category, name, Revenue_generated,
RANK() OVER (PARTITION BY category ORDER BY Revenue_generated DESC ) AS  r_n
FROM
(SELECT pt.category, pt.name, SUM(od.quantity * p.price) AS Revenue_generated
FROM pizza_types pt
INNER JOIN pizzas p
ON pt.pizza_type_id = p.pizza_type_id
INNER JOIN order_details od
ON p.pizza_id = od.pizza_id
INNER JOIN orders o
ON o.order_id = od.order_id
GROUP BY pt.category,pt.name) AS revenue) AS R_rank
WHERE r_n <= 3 ;
```

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

category	name	Revenue_generated
Chicken	The Thai Chicken Pizza	43434.25
Chicken	The Barbecue Chicken Pizza	42768
Chicken	The California Chicken Pizza	41409.5
Classic	The Classic Deluxe Pizza	38180.5
Classic	The Hawaiian Pizza	32273.25
Classic	The Pepperoni Pizza	30161.75
Supreme	The Spicy Italian Pizza	34831.25
Supreme	The Italian Supreme Pizza	33476.75
Supreme	The Sicilian Pizza	30940.5
Veggie	The Four Cheese Pizza	32265.70000000066

Result 41 | Read Only

CONCLUSION

- A total of 21,350 orders have been placed, generating a revenue of $817,860.
- 'The Greek Pizza' is the most expensive pizza and 'L' size is the most common size pizza ordered.
- The most popular pizza category is 'classic' and the most ordered pizza type is 'classic_dlx’.
- The busiest times for ordering are between 12:00-14:00 and 16:00-19:00.
- "The Thai Chicken Pizza" generated $434,34 in revenue, followed by "The Barbecue Chicken Pizza".
- ‘Classic’ and ‘Supreme pizzas comprise the largest revenue share, followed by ‘Chicken’ and ‘Veggie’.


THANK YOU
Thank you for your attention. I appreciate your time and interest in my Pizza Sales Analysis project.
If you have any questions, feedback, or suggestions, please feel free to contact me. Your insights and comments are valuable to me as I continue to improve my analysis.
Have a great day!