

# PIZZA PROJECT



[NEXT >](#)



## ABOUT US

My name is Nikhil Kumar, and I am a passionate Data Analyst with a strong interest in turning raw data into meaningful insights. I recently worked on a project analyzing Pizza Sales Data using SQL, where I explored order patterns, customer behavior, and sales trends.

[< BACK](#)[NEXT >](#)

# 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.

< BACK

NEXT >



LET SOLVE ONE BY ONE

< BACK

NEXT >




RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

QUERY -

```
SELECT
    COUNT (ORDER_ID)
FROM
    ORDERS;
```

ANSWER -

	count bigint 
1	21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

QUERY -

```
SELECT
  SUM(P.PRICE * O.QUANTITY) AS TOTAL_REVENUE
FROM
  PIZZA P
JOIN ORDERS_DETAIL O ON O.PIZZA_ID = P.PIZZA_ID;
```

ANSWER -

	total_revenue money 
1	? 817,860.05

IDENTIFY THE HIGHEST-PRICED PIZZA.

QUERY -

```
SELECT
  P1.NAME,
  P2.PRICE
FROM
  PIZZA_CATEGORY P1
  JOIN PIZZA P2 ON P2.PIZZA_TYPE_ID = P1.PIZZA_TYPE_ID
ORDER BY
  P2.PRICE DESC LIMIT
  1;
```

ANSWER -

	name character varying (100) 🔒	price money 🔒
1	The Greek Pizza	? 35.95

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

QUERY -

```
SELECT
  P.SIZE,
  COUNT(O.ORDER_DETAILS_ID) AS MOSTLY_ORDERED
FROM
  PIZZA P
  JOIN ORDERS_DETAIL O ON P.PIZZA_ID = O.PIZZA_ID
GROUP BY
  P.SIZE
ORDER BY
  MOSTLY_ORDERED DESC LIMIT
  1;
```

ANSWER -

	size character varying (10) 🔒	mostly_ordered bigint 🔒
1	L	18526



LIST THE TOP 5 MOST ORDERED PIZZA TYPES ALONG WITH THEIR QUANTITIES.

QUERY -

```
SELECT
  P2.NAME,
  SUM(O.QUANTITY) AS TOTAL
FROM
  PIZZA_CATEGORY P2
JOIN PIZZA P1 ON P1.PIZZA_TYPE_ID = P2.PIZZA_TYPE_ID
JOIN ORDERS_DETAIL O ON O.PIZZA_ID = P1.PIZZA_ID
GROUP BY
  P2.NAME
ORDER BY
  TOTAL DESC LIMIT
  5;
```

ANSWER -

	name character varying (100)	total bigint
1	The Classic Deluxe Pizza	2453
2	The Barbecue Chicken Pizza	2432
3	The Hawaiian Pizza	2422
4	The Pepperoni Pizza	2418
5	The Thai Chicken Pizza	2371

JOIN THE NECESSARY TABLES TO FIND THE TOTAL QUANTITY OF EACH PIZZA CATEGORY ORDERED.

QUERY -

```
SELECT
  P1.CATEGORY,
  SUM(O.QUANTITY) AS TOTAL_SALE
FROM
  PIZZA_CATEGORY P1
  JOIN PIZZA P2 ON P1.PIZZA_TYPE_ID = P2.PIZZA_TYPE_ID
  JOIN ORDERS_DETAIL O ON P2.PIZZA_ID = O.PIZZA_ID
GROUP BY
  P1.CATEGORY
ORDER BY
  TOTAL_SALE DESC;
```

ANSWER -

	category character varying (100)	total_sale bigint
1	Classic	14888
2	Supreme	11987
3	Veggie	11649
4	Chicken	11050

JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

QUERY -

```
SELECT
  CATEGORY,
  COUNT(NAME)
FROM
  PIZZA_CATEGORY
GROUP BY
  CATEGORY
ORDER BY
  COUNT DESC;
```

ANSWER -


	category character varying (100) 🔒	count bigint 🔒
1	Supreme	9
2	Veggie	9
3	Classic	8
4	Chicken	6

GROUP THE ORDERS BY DATE AND CALCULATE THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

QUERY -

```
SELECT
    AVG(QUANTITY)::NUMERIC(10, 2)
FROM
    (
        SELECT
            O1.ORDER_DATE,
            SUM(O2.QUANTITY) AS QUANTITY
        FROM
            ORDERS O1
            JOIN ORDERS_DETAIL O2 ON O1.ORDER_ID = O2.ORDER_ID
        GROUP BY
            O1.ORDER_DATE
    ) AS ORDER_QUANTITY;
```

ANSWER -

	avg numeric (10,2) 
1	138.47



DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

QUERY -

```
SELECT
  P2.NAME,
  SUM(O.QUANTITY * P1.PRICE) AS REVENUE
FROM
  PIZZA_CATEGORY P2
  JOIN PIZZA P1 ON P1.PIZZA_TYPE_ID = P2.PIZZA_TYPE_ID
  JOIN ORDERS_DETAIL O ON P1.PIZZA_ID = O.PIZZA_ID
GROUP BY
  P2.NAME
ORDER BY
  REVENUE DESC LIMIT
  3;
```

ANSWER -

	name character varying (100)	revenue money
1	The Thai Chicken Pizza	? 43,434.25
2	The Barbecue Chicken Pizza	? 42,768.00
3	The California Chicken Pizza	? 41,409.50

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

QUERY -

```
SELECT
  P2.category,
  (SUM(O.QUANTITY * P1.PRICE) / (SELECT
    SUM(P1.PRICE * O.QUANTITY) AS TOTAL_REVENUE
  FROM
    PIZZA P1
    JOIN ORDERS_DETAIL O ON O.PIZZA_ID = P1.PIZZA_ID) *100)::numeric(10,2) as revenue
FROM
  PIZZA_CATEGORY P2
  JOIN PIZZA P1 ON P1.PIZZA_TYPE_ID = P2.PIZZA_TYPE_ID
  JOIN ORDERS_DETAIL O ON P1.PIZZA_ID = O.PIZZA_ID
GROUP BY
  P2.category
ORDER BY
  REVENUE DESC LIMIT
  3;
```

ANSWER -

	category character varying (100)	revenue numeric (10,2)
1	Classic	26.91
2	Supreme	25.46
3	Chicken	23.96

## ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

QUERY -

```
SELECT
  ORDER_DATE,
  SUM(REVENUE) OVER (
    ORDER BY
      ORDER_DATE
  ) AS CUM_REVENUE
FROM
  (
    SELECT
      O1.ORDER_DATE,
      SUM(O2.QUANTITY * P.PRICE) AS REVENUE
    FROM
      ORDERS_DETAIL O2
    JOIN PIZZA P ON O2.PIZZA_ID = P.PIZZA_ID
    JOIN ORDERS O1 ON O1.ORDER_ID = O2.ORDER_ID
    GROUP BY
      O1.ORDER_DATE
  ) AS SALES;
```

ANSWER -

	order_date date	cum_revenue money
1	2015-01-01	? 2,713.85
2	2015-01-02	? 5,445.75
3	2015-01-03	? 8,108.15
4	2015-01-04	? 9,863.60
5	2015-01-05	? 11,929.55
6	2015-01-06	? 14,358.50

DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE FOR EACH PIZZA CATEGORY.

QUERY -

```
select category, name, revenue, rank
from
(SELECT CATEGORY,NAME,REVENUE,
RANK() OVER (PARTITION BY CATEGORY ORDER BY REVENUE ) AS RANK
FROM
(
SELECT
P2.CATEGORY,
P2.NAME,
SUM(O.QUANTITY * P1.PRICE) AS REVENUE
FROM
PIZZA_CATEGORY P2
JOIN PIZZA P1 ON P1.PIZZA_TYPE_ID = P2.PIZZA_TYPE_ID
JOIN ORDERS_DETAIL O ON O.PIZZA_ID = P1.PIZZA_ID
GROUP BY
P2,
CATEGORY,
P2.NAME) AS A) as B
where rank <=3;
```

ANSWER -

	category character varying (100)	name character varying (100)	revenue money	rank bigint
1	Chicken	The Chicken Pesto Pizza	716,701.75	1
2	Chicken	The Chicken Alfredo Pizza	716,900.25	2
3	Chicken	The Southwest Chicken Pizza	734,705.75	3
4	Classic	The Pepperoni, Mushroom, and Peppers Pizza	718,834.50	1
5	Classic	The Big Meat Pizza	722,568.00	2
6	Classic	The Napolitana Pizza	724,087.00	3



# THANK YOU!

< BACK