BY PIYUSH TIKIYA

Pizza Store Analysis

PIZZA STORE

BY PIYUSH TIKIYA

Menu Highlights

Thai Chicken

BBQ Chicken

California Chicken

Italian Chicken

SITUATION:

- THE BUSINESS WANTED TO EVALUATE THE PERFORMANCE OF THEIR PIZZA STORE OPERATIONS.
- FOCUS AREAS INCLUDED UNDERSTANDING WHICH CATEGORIES AND TOPPINGS WERE MOST POPULAR.
- THEY ALSO WANTED INSIGHTS INTO SALES PATTERNS DURING DIFFERENT TIMES OF THE DAY.
- THE GOAL WAS TO DISCOVER TRENDS AND PAIN POINTS TO HELP THE BUSINESS GROW.

TASK:

- MY RESPONSIBILITY WAS TO ANALYZE THE SALES DATA USING SQL.
- I NEEDED TO UNCOVER ACTIONABLE INSIGHTS TO HELP MAKE SMARTER, DATA-BACKED BUSINESS DECISIONS.
- THE FINAL OUTPUT WAS TO BE A CLEAR AND INSIGHTFUL REPORT SHOWING TRENDS, TOP ITEMS, AND IMPROVEMENT AREAS.

ACTION

- COLLECTED AND STUDIED THE DATA FROM FOUR TABLES: ORDERS, ORDER_DETAILS, PIZZAS, AND PIZZA_TYPES.
- USED JOINS, GROUP BY, CTES, SUBQUERIES, AND WINDOW FUNCTIONS LIKE ROW_NUMBER, RANK, AND DENSE_RANK FOR DEEP ANALYSIS.

CALCULATED:

- CATEGORY-WISE TOTAL SALES
- TOP-SELLING PIZZAS BY QUANTITY AND REVENUE
- PEAK SALES HOURS
- LEAST-PERFORMING ITEMS
- PERFORMED TIME-BASED COMPARISONS TO IDENTIFY CHURN TRENDS AND SALES GROWTH PATTERNS.
- COMPILED ALL FINDINGS INTO A FINAL REPORT WITH INSIGHTS, VISUALS, AND RECOMMENDATIONS.

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

pizza_id	pizza_type_id	size	price
bbq_ckn_s	bbq_ckn	S	12.75
bbq_ckn_m	bbq_ckn	M	16.75
bbq_ckn_l	bbq_ckn	L	20.75
cali_ckn_s	cali_ckn	S	12.75
cali_ckn_m	cali_ckn	M	16.75
cali_ckn_l	cali_ckn	L	20.75

```sql
select * from orders;
select * from order_details;
select * from pizzas;
select * from pizza_types;
```

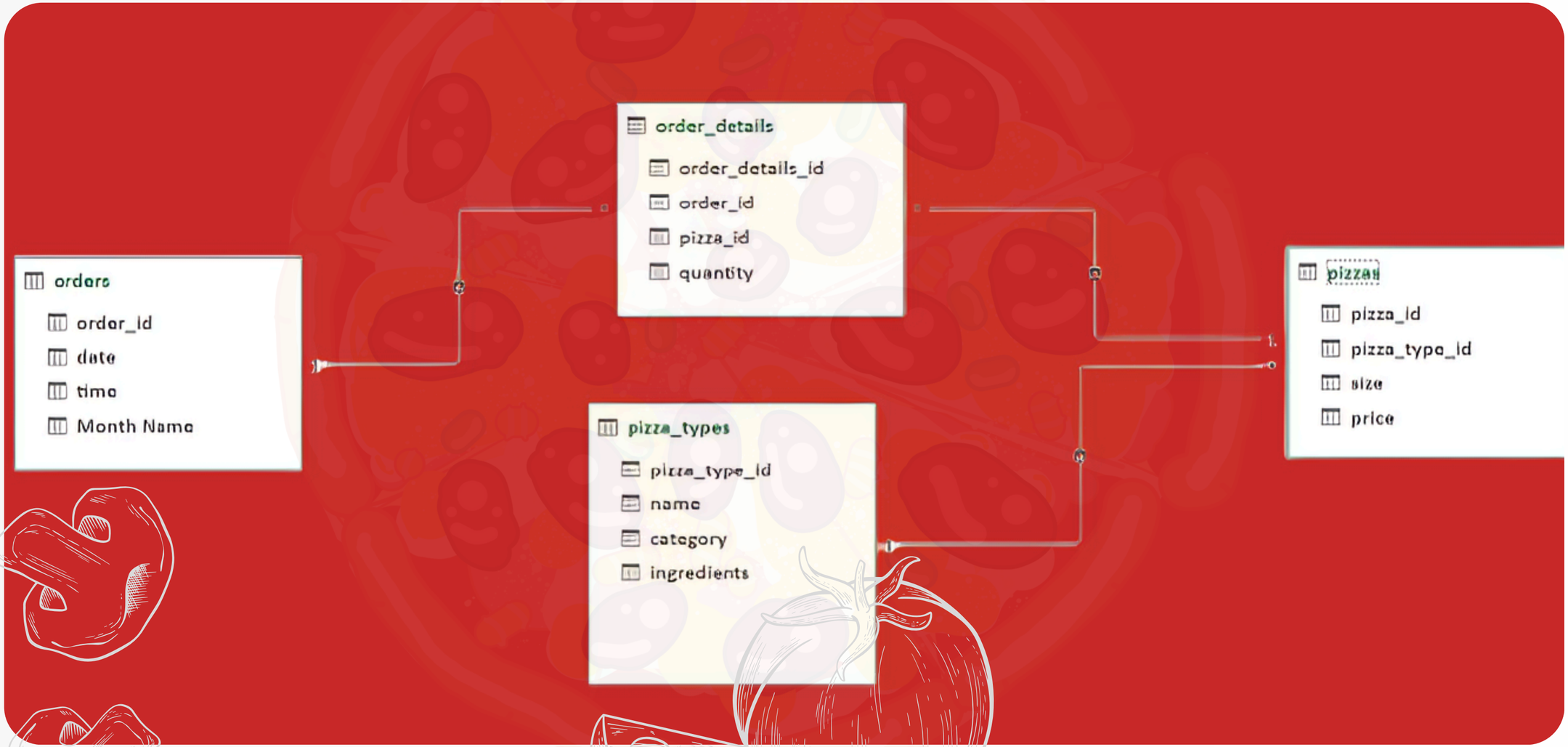
order_details_id	order_id	pizza_id	quantity
1	1	hawaiian_m	1
2	2	classic_dlx_m	1
3	2	five_cheese_l	1
4	2	ital_supr_l	1
5	2	mexicana_m	1
6	2	thai_ckn_l	1

pizza_type_id	name	category	ingredients
bbq_ckn	The Barbecue Chicken Pizza	Chicken	Barbecued Chicken, Red Peppers, Green Peppers, Tomatoes, Red Onions, Barbecue Sauce
cali_ckn	The California Chicken Pizza	Chicken	Chicken, Artichoke, Spinach, Garlic, Jalapeno Peppers, Fontina Cheese, Gouda Cheese
ckn_alfredo	The Chicken Alfredo Pizza	Chicken	Chicken, Red Onions, Red Peppers, Mushrooms, Asiago Cheese, Alfredo Sauce
ckn_pesto	The Chicken Pesto Pizza	Chicken	Chicken, Tomatoes, Red Peppers, Spinach, Garlic, Pesto Sauce
southw_ckn	The Southwest Chicken Pizza	Chicken	Chicken, Tomatoes, Red Peppers, Red Onions, Jalapeno Peppers, Corn, Cilantro, Chipotle Sauce
thai_ckn	The Thai Chicken Pizza	Chicken	Chicken, Pineapple, Tomatoes, Red Peppers, Thai Sweet Chilli Sauce

PIZZA STORE

BY PIYUSH TIKIYA

ER DIAGRAM



RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
select distinct(count(order_id)) count_of_orders from orders;
```

count_of_orders
21350

CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
selecT round(Sum(price*quantity),2) as Total__pizza_sales from vw_combined_pizza_data
```

Total__pizza_sales
817860.05

PIZZA STORE

Customer Engagement

IDENTIFY THE HIGHEST-PRICED PIZZA.

```sql
SELECT
    pizza_type_name, MAX(price) AS Higest_priced_pizza
FROM
    vw_combined_pizza_data
GROUP BY pizza_type_name
ORDER BY Higest_priced_pizza DESC
LIMIT 1;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.18800000

	pizza_type_name	Higest_priced_pizza
▸	The Greek Pizza	35.95

PIZZA STORE

Customer Engagement

IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
SELECT
    size, COUNT(order_details_id) AS commom_size_pizza_count
FROM
    vw_combined_pizza_data
GROUP BY size
ORDER BY commom_size_pizza_count DESC;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.15600000

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

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

PIZZA STORE

Customer Engagement

```sql
SELECT
    pizza_types.NAME,
    SUM(order_details.quantity) AS total_quantity
FROM
    pizzas
        JOIN
    pizza_types 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 total_quantity DESC
LIMIT 5;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.15700000

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

PIZZA STORE

Customer Engagement

```sql
SELECT
    Category, COUNT(name) AS pizza_type
FROM
    pizza_types
GROUP BY Category;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.0000000

	Category	pizza_type
▶	Chicken	6
	Classic	8
	Supreme	9
	Veggie	9

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

PIZZA STORE

Customer Engagement

```sql
SELECT
    pizza_types.category,
    SUM(order_details.quantity) AS Total_quantity,
    ROUND(SUM(order_details.quantity * pizzas.price),
            2) AS Total_sales
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 Total_sales DESC , Total_quantity DESC
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.15600000

category	Total_quantity	Total_sales
Classic	14888	220053.1
Supreme	11987	208197
Chicken	11050	195919.5
Veggie	11649	193690.45

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

```sql
SELECT
    ROUND(AVG(daily_pizzas), 0) AS avg_pizzas_per_day
FROM
    (SELECT
        o.order_date, SUM(od.quantity) AS daily_pizzas
    FROM
        orders o
    JOIN order_details od ON o.order_id = od.order_id
    GROUP BY o.order_date) AS daily_summary;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.09300000

	avg_pizzas_per_day
▶	138

DETERMINE THE TOP 5 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

BY PIYUSH TIKIYA

PIZZA STORE

Customer Engagement

```sql
SELECT
    (pizza_types.name) AS N,
    SUM(pizzas.price * order_details.quantity) AS Total_sales_of_pizzas
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 N
ORDER BY Total_sales_of_pizzas DESC
LIMIT 5;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.15600000

N	Total_sales_of_pizzas
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5
The Classic Deluxe Pizza	38180.5
The Spicy Italian Pizza	34831.25

DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY.

BY PIYUSH TIKIYA

Customer Engagement

```sql
SELECT
    DATE_FORMAT(order_time, '%h %p') AS hour_am_pm,
    COUNT(DISTINCT order_id) AS total_orders
FROM orders
GROUP BY hour_am_pm
ORDER BY total_orders DESC;
```

Query Statistics

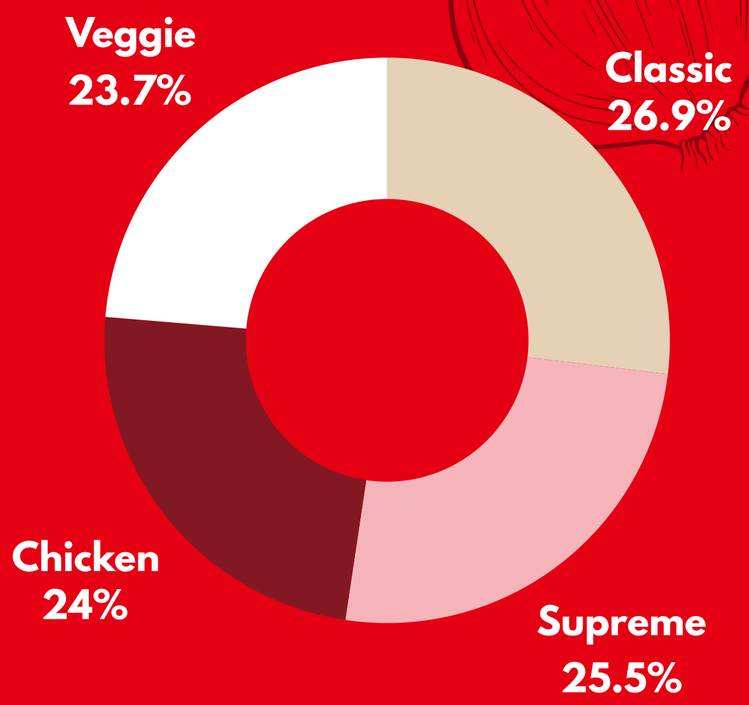
Timing (as measured at client side):
Execution time: 0:00:0.06200000

hour_am_pm	total_orders
12 PM	2520
01 PM	2455
06 PM	2399
05 PM	2336
07 PM	2009
04 PM	1920
08 PM	1642
02 PM	1472
03 PM	1468
11 AM	1231

PIZZA STORE

CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE

```sql
select c, Percentage_contribution from
(
SELECT (pizza_types.category) as c,
round(sum(pizzas.price*order_details.quantity),0) as Category_wise_Total_sales_of_pizzas,
round(SUM(sum(pizzas.price * order_details.quantity)) over(),0) as Total_sale_across_all_category,
round(
sum(pizzas.price*order_details.quantity)/SUM(SUM(pizzas.price * order_details.quantity)) over() * 100,
2) as Percentage_contribution
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  C
ORDER BY Category_wise_Total_sales_of_pizzas desc) as A;
```



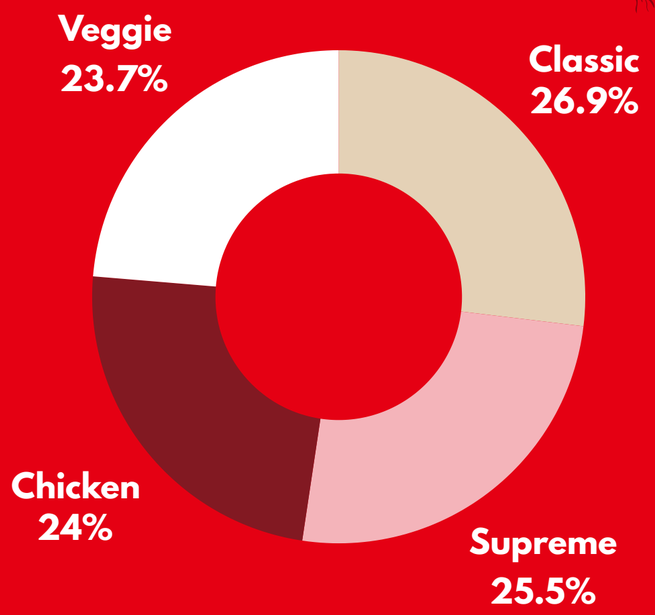
c	Percentage_contribution
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.34300000

2ND SOLUTION WITH QUERY STATS COMPARSION WITH 1ST SOLUTION

```sql
WITH category_sales AS (
    SELECT
        pizza_types.category AS c,
        ROUND(SUM(pizzas.price * order_details.quantity), 0) AS category_total
    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
),
total_sales AS (
    SELECT
        ROUND(SUM(pizzas.price * order_details.quantity), 0) AS total
    FROM pizzas
    JOIN order_details ON order_details.pizza_id = pizzas.pizza_id
)
SELECT
    cs.c AS Category,
    -- cs.category_total AS Category_wise_Total_sales_of_pizzas,
    -- ts.total AS Total_sale_across_all_category,
    ROUND((cs.category_total / ts.total) * 100, 2) AS Percentage_contribution
FROM category_sales cs
CROSS JOIN total_sales ts
ORDER BY cs.category_total DESC;
```



Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.50000000

c	Percentage_contribution
Classic	26.91
Supreme	25.46
Chicken	23.96
Veggie	23.68

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

PIZZA STORE

```sql
with cte as
(
SELECT
    pizza_types.name n,
    pizza_types.category c,
    ROUND(SUM(order_details.quantity * pizzas.price),
            0) 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 n , c
),
 cte1 as (
select n,c,revenue,
row_number() over(partition by c order by revenue desc) as dr
from cte
)
select n,c, revenue from cte1 where dr <=3;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.18700000

name	category	revenue
The Thai Chicken Pizza	Chicken	43434
The Barbecue Chicken Pizza	Chicken	42768
The California Chicken Pizza	Chicken	41410
The Classic Deluxe Pizza	Classic	38180
The Hawaiian Pizza	Classic	32273
The Pepperoni Pizza	Classic	30162
The Spicy Italian Pizza	Supreme	34831
The Italian Supreme Pizza	Supreme	33477
The Sicilian Pizza	Supreme	30940
The Four Cheese Pizza	Veggie	32266
The Mexicana Pizza	Veggie	26781
The Five Cheese Pizza	Veggie	26066

2ND SOLUTION WITH QUERY STATS COMPARSION WITH 1ST SOLUTION

BY PIYUSH TIKIYA

PIZZA STORE

```sql
select name,category,revenue from
(
select name, category, revenue,
rank() over(partition by category order by revenue desc) as Rn
from(
SELECT
    pizza_types.name ,
    pizza_types.category ,
    ROUND(SUM(order_details.quantity * pizzas.price),
            0) 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 name, category
) as a) as b
where rn <= 3
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.23500000

name	category	revenue
The Thai Chicken Pizza	Chicken	43434
The Barbecue Chicken Pizza	Chicken	42768
The California Chicken Pizza	Chicken	41410
The Classic Deluxe Pizza	Classic	38180
The Hawaiian Pizza	Classic	32273
The Pepperoni Pizza	Classic	30162
The Spicy Italian Pizza	Supreme	34831
The Italian Supreme Pizza	Supreme	33477
The Sicilian Pizza	Supreme	30940
The Four Cheese Pizza	Veggie	32266
The Mexicana Pizza	Veggie	26781
The Five Cheese Pizza	Veggie	26066

ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME AND DATE MUST BE IN DD-MM-YYYY

BY PIYUSH TIKIYA

PIZZA STORE

```sql
select date_format(d,'%D-%M-%Y') as Period , revenue,
round(sum(revenue) over(order by d),0 )as Cumulative_revenue
from
(
 select (orders.order_date) as d,
round(sum(order_details.quantity*pizzas.price),0) 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 d) as Inside_query
order by d;
```

Query Statistics

Timing (as measured at client side
Execution time: 0:00:0.14000000

Period	revenue	Cumulative_revenue
1st-January-2015	2714	2714
2nd-January-2015	2732	5446
3rd-January-2015	2662	8108
4th-January-2015	1755	9863
5th-January-2015	2066	11929
6th-January-2015	2429	14358
7th-January-2015	2202	16560
8th-January-2015	2838	19398
9th-January-2015	2127	21525
10th-January-2015	2464	23989
11th-January-2015	1872	25861
12th-January-2015	1919	27780

PIZZA STORE

2ND SOLUTION WITH QUERY STATS COMPARSION WITH 1ST SOLUTION

BY PIYUSH TIKIYA

```sql
with cumulative_revenue as
(
select (orders.order_date) as d,
round(sum(order_details.quantity*pizzas.price),0) 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 d)

select date_format(d,'%D-%M-%Y') as Period, revenue,
round(sum(revenue) over(order by d),0 )as Cumulative_revenue
from cumulative_revenue
order by d ;
```

Query Statistics

Timing (as measured at client side):
Execution time: 0:00:0.12500000

Period	revenue	Cumulative_revenue
1st-January-2015	2714	2714
2nd-January-2015	2732	5446
3rd-January-2015	2662	8108
4th-January-2015	1755	9863
5th-January-2015	2066	11929
6th-January-2015	2429	14358
7th-January-2015	2202	16560
8th-January-2015	2838	19398
9th-January-2015	2127	21525
10th-January-2015	2464	23989
11th-January-2015	1872	25861
12th-January-2015	1919	27780

BY PIYUSH TIKIYA

FINAL RECOMMENDATIONS FOR BUSINESS GROWTH (BASED ON ANALYSIS)

REVAMP THE MENU:

REPLACE OR REWORK UNDERPERFORMING PIZZAS WITH LOW SALES AND LOW CUSTOMER INTEREST.

PROMOTE FAST-GROWING LESSER-KNOWN PIZZAS (E.G., THAI OR SOUTHWEST VARIETIES) — THEY SHOW UNTAPPED POTENTIAL.

TIME-BASED PROMOTIONS:

LAUNCH TARGETED OFFERS DURING EVENING HOURS AND WEEKENDS, WHEN SALES ARE NATURALLY HIGH, TO BOOST REVENUE EVEN FURTHER.

CONSIDER WEEKDAY LUNCH DEALS TO ATTRACT TRAFFIC DURING LOW-ACTIVITY PERIODS.

BUNDLE HIGH & LOW SELLERS:

COMBINE TOP-SELLERS (LIKE DELUXE OR CLASSIC) WITH LOWER-SELLING ITEMS IN COMBO OFFERS , TO INCREASE OVERALL PRODUCT MOVEMENT.

FINAL RECOMMENDATIONS FOR BUSINESS GROWTH (BASED ON ANALYSIS)

BY PIYUSH TIKIYA

CATEGORY-BASED INVENTORY PLANNING:

USE SALES DATA TO FORECAST DEMAND BY CATEGORY (E.G., VEG VS CHICKEN VS BEEF) — REDUCE WASTAGE, IMPROVE STOCKING EFFICIENCY.

LOYALTY & RETENTION PROGRAM:

- USE CHURN RATE INSIGHTS TO IDENTIFY REPEAT VS LOST CUSTOMERS.
- INTRODUCE LOYALTY REWARDS OR PERSONALIZED DISCOUNTS TO RETAIN HIGH-VALUE BUYERS.

FEEDBACK LOOP FOR LOW-SELLERS:

COLLECT CUSTOMER FEEDBACK ON PIZZAS THAT AREN'T PERFORMING WELL — FIND OUT WHETHER IT'S TASTE, INGREDIENTS, OR PRICING.

GEO & EVENT-BASED CAMPAIGNS :

ANALYZE LOCATION-WISE PERFORMANCE AND RUN HYPERLOCAL OFFERS.

ALIGN CAMPAIGNS WITH LOCAL EVENTS OR SEASONS, E.G., DISCOUNTS DURING SPORTS FINALS OR FESTIVE WEEKENDS.

PIZZA STORE

BY PIYUSH TIKIYA