

# SQL PROJECT

## PIZZA SHOP SALES ANALYSIS

Analyzing Pizza Shop Sales Data

**Project Objective:**  
**Analyzing Sales Data & customer preferences for refining inventory management & elevate customer satisfaction.**



# Key Analysis

---

01

TOTAL SALES FOR  
EACH MONTH

02

SALES DIFFERENCE BETWEEN  
CURRENT AND PRE-MONTH

03

TOTAL ORDERS FOR EACH  
MONTH & MOM GROWTH

04

SALES ANALYSIS  
ON WEEKENDS &  
WEEKDAYS

05

SALES ANALYSIS BY STORE  
LOCATION

06

DAILY SALES ANALYSIS WITH  
AVERAGE LINE

07

SALES BY PRODUCT CATEGORY

08

SALES ANALYSIS BY DAYS AND HOURS

- To calculate the total sales for each month.
- 

## Syntax:

```
Select date_format(transaction_date,'%M') as Monthname,  
round(sum(unit_price*transaction_qty),2) as total_sales  
from coffee_shop_sales  
group by 1,2;
```

## Output:

Monthname	total_sales
January	81677.74
February	76145.19
March	98834.68
April	118941.08
May	156727.76

- To calculate the Sales Difference between current & previous month

Syntax:

with aa as

(

```
select month(transaction_date) as Monthwise, date_format(transaction_date,'%M') as Monthname, round(sum(unit_price*transaction_qty),2) as totalsales from coffee_shop_sales group by 1,2
```

)

```
select Monthwise,Monthname,totalsales,LAG(totalsales) OVER (ORDER BY Monthwise) AS prem_sales,  
round((totalsales-LAG(totalsales) OVER (ORDER BY Monthwise)),2) as mom_growth from aa;
```

Output:

Monthname	totalsales	prem_sales	mom_growth
January	81677.74	NULL	NULL
February	76145.19	81677.74	-5532.55
March	98834.68	76145.19	22689.49
April	118941.08	98834.68	20106.4
May	156727.76	118941.08	37786.68

- To calculate total orders for each month and MOM Order Growth

Syntax:

with aa as

(

```
select month(transaction_date) as Monthwise, date_format(transaction_date,'%M') as Monthname,  
count(transaction_id) as total_orders from coffee_shop_sales  
group by 1,2
```

)

```
select Monthwise,Monthname,total_orders,LAG(total_orders) OVER (ORDER BY Monthwise)  
AS pmonthods,  
round((total_orders-LAG(total_orders) OVER (ORDER BY Monthwise)),2) as  
mom_ordergrowth from aa;
```

Output:

Monthname	total_orders	pmonthods	mom_ordergrowth
January	17314	NULL	NULL
February	16359	17314	-955
March	21229	16359	4870
April	25335	21229	4106
May	33527	25335	8192

- To calculate Total sales, orders, quantities on Weekends & Weekdays

### Syntax:

```
select case when DAYOFWEEK(transaction_date) IN (1,7) THEN 'WEEKEND' ELSE  
'WEEKDAYS' END AS day_type,  
round(sum(unit_price*transaction_qty),2) as totalsales,count(transaction_id) as  
total_orders,  
sum(transaction_qty) as Totqty  
from coffee_shop_sales  
where month(transaction_date)=6  
group by case  
when DAYOFWEEK(transaction_date) IN (1,7) THEN 'WEEKEND' ELSE 'WEEKDAYS'  
END;
```

### Output:

day_type	totalsales	total_orders	Totqty
WEEKDAYS	121484.08	25883	37278
WEEKEND	45001.8	9469	13664

# • Calculate Total Sales,Orders,Quantites by various Store Location

---

## Syntax:

```
select store_location,  
round(sum(unit_price*transaction_qty),2) as totalsales,  
count(transaction_id) as total_orders,  
sum(transaction_qty) as Totqty from coffee_shop_sales  
group by 1  
order by 2 desc;
```

## Output:

store_location	totalsales	total_orders	Totqty
Hell s Kitchen	236511.17	50735	71737
Astoria	232243.91	50599	70991
Lower Manhattan	230057.25	47782	71742

# DAILY SALES ANALYSIS WITH AVERAGE LINE AND AVERAGE STATUS

## Syntax:

```
with bb as
(
  with aa as
  (
    select day(transaction_date) as Daywsie,month(transaction_date) as monthh,
    round(sum(unit_price*transaction_qty),2) as Revenue from coffee_shop_sales
    where month(transaction_date)=5
    group by 1,2
  )
  select monthh,Daywsie,Revenue, round(avg(Revenue) over (partition by monthh),2) as
  dailiy_avg_of_month from aa
)
select * , CASE WHEN Revenue > dailiy_avg_of_month THEN "ABOVE AVERAGE" ELSE "BELOW
AVERAGE" END AS Average_Status from bb;
```

## Output:

monthh	Daywsie	Revenue	dailiy_avg_of_month	Average_Status
5	1	4731.45	5055.73	BELOW AVERAGE
5	2	4625.5	5055.73	BELOW AVERAGE
5	3	4714.6	5055.73	BELOW AVERAGE
5	4	4589.7	5055.73	BELOW AVERAGE
5	5	4701	5055.73	BELOW AVERAGE

- To Calculate TotalSales by Product Category

---

Syntax:

```
select product_category,  
       round(sum(unit_price*transaction_qty),2) as Revenue from  
coffee_shop_sales  
group by 1  
order by 2 desc;
```

Output:

product_category	Revenue
Coffee	269952.45
Tea	196405.95
Bakery	82315.64
Drinking Chocolate	72416
Coffee beans	40085.25

## • To Calculate TotalSales By Days and Hours

---

Syntax:

```
select dayname(transaction_date) as Dayname,  
hour(transaction_time) as Byhour,  
round(sum(unit_price*transaction_qty),2) as Revenue from  
coffee_shop_sales  
where month(transaction_date)=5  
group by 1,2  
order by 3 desc  
limit 7;
```

Output:

Dayname	Byhour	Revenue
Tuesday	9	3234.06
Sunday	9	3231.08
Tuesday	10	3124.87
Wednesday	10	3107.4
Wednesday	9	3011.6

# Percentage % contribution of each pizza Category to Total Revenue

Syntax:

```
with aa as
(
select a.category as cat,round(sum(c.quantity*b.price),2) as revenue
from pizza_types as a
join pizzas as b on a.pizza_type_id=b.pizza_type_id
join order_details as c on b.pizza_id=c.pizza_id
group by 1
)
select cat, (revenue/(select sum(revenue) from aa))*100.00 as
percentage from aa;
```

Output:

cat	percent
Classic	26.9059
Veggie	23.6825
Supreme	25.4563

- Analyze the Cumulative Revenue generated over time.

Syntax:

```
with aa as
(
  select a.date,round(sum(d.quantity*c.price),2) as revenue from orders as a
  join order_details as d on a.order_id=d.order_id
  join pizzas as c on d.pizza_id=c.pizza_id
  group by 1
)
select date,revenue,round(sum(revenue) over (order by date),2) as
cum_revenue from aa;
```

Output:

<b>date</b>	<b>revenue</b>	<b>cum_revenue</b>
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15

- Top 3 most ordered pizza based on revenue for each pizza category.

Syntax:

```
with bb as
(
with aa as
(
select a.category as cat,a.name as pizza_name,sum(c.quantity*b.price) as revenue from
pizza_types as a
join pizzas as b on a.pizza_type_id=b.pizza_type_id
join order_details as c on b.pizza_id=c.pizza_id
group by cat,pizza_name
order by cat, 2 desc
)
select cat,pizza_name,revenue, row_number() over(partition by cat order by revenue desc) as
ranking from aa
)
select cat,pizza_name,revenue,ranking from bb
where ranking <=3;
```

Output:

cat	pizza_name	revenue	ranking
Chicken	The Thai Chicken Pizza	43434.25	1
Chicken	The Barbecue Chicken Pizza	42768	2
Chicken	The California Chicken Pizza	41409.5	3