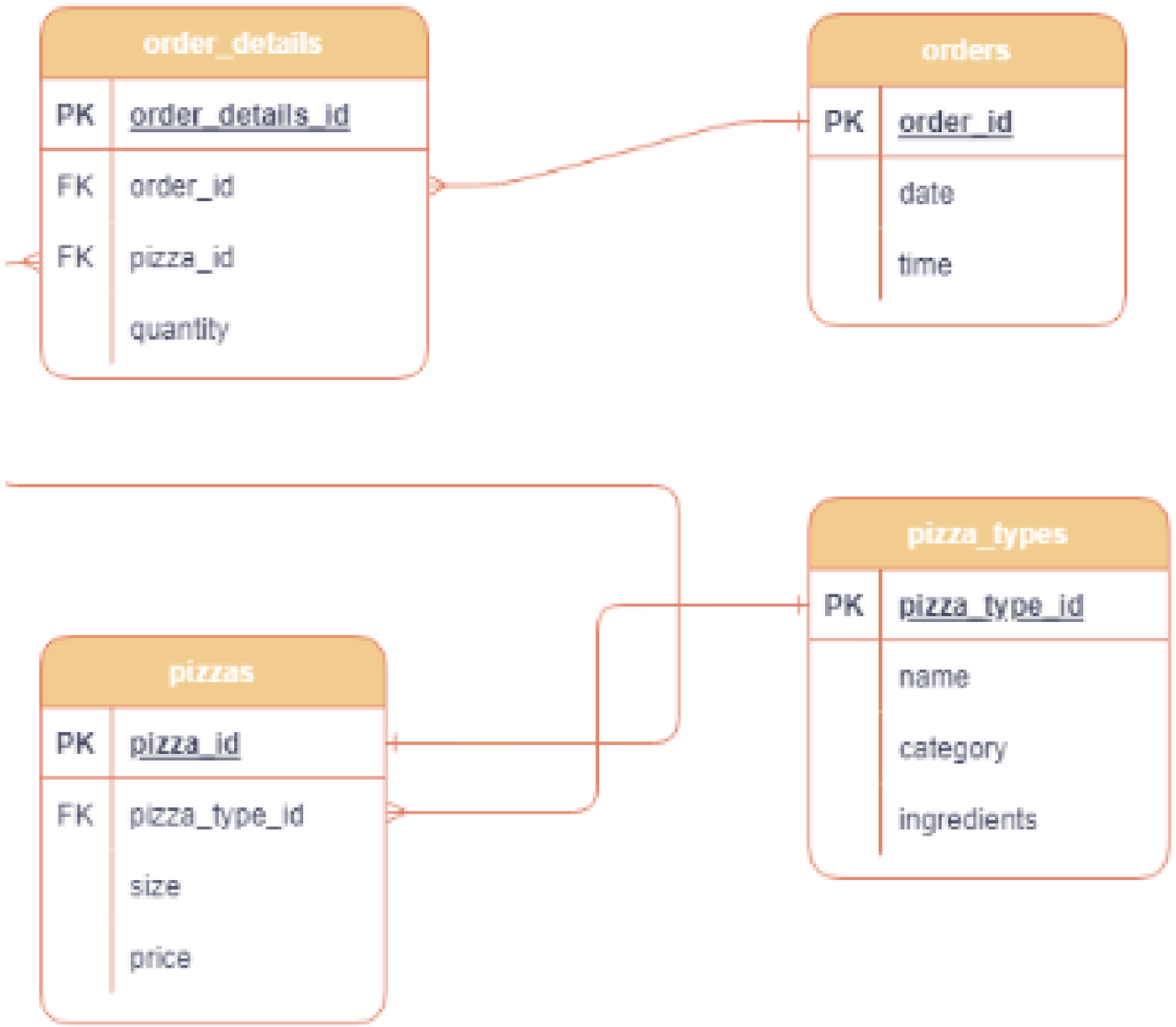
SQL PROJECT ON PIZZA’S SALES

Hello!

My name is **Indrajeet Kumar Verma** in this project i have utilized sql querey to solve this questions that were related to pizza sales.

Database schema :-



Short overview of Database schema :-

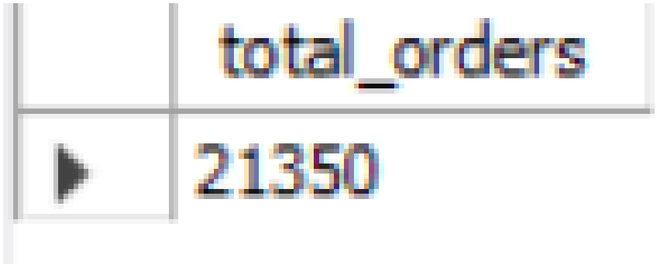
The database schema for pizza sales includes four main tables :"orders_table" for order details, "order_details table" for items in each order, "pizzas_table" for individual pizza specifications, and "pizza_types_table" for categorizing pizzas by type. Each table is interconnected with primary and foreign keys to recognise sales and their revenues efficiently.

Questions,queries,results and insights: -

-- Q1 - Retrieve the total number of orders placed.

```
SELECT
COUNT(order_id) AS total_orders
FROM
 orders;
```

insights - total orders during one year -

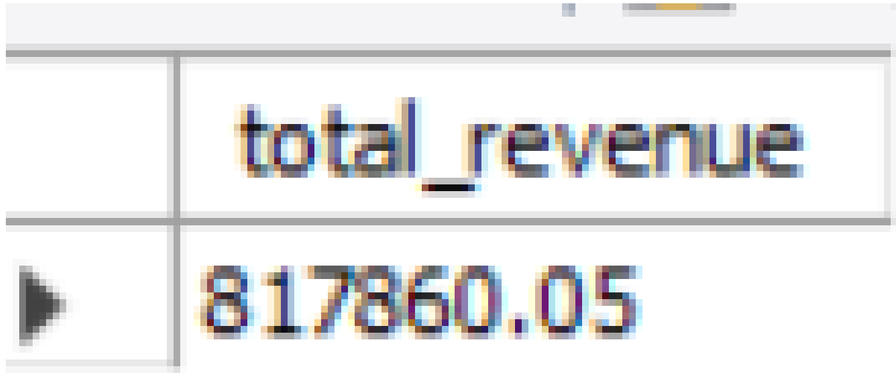


total_orders
21350

-- Q2 - Calculate the total revenue generated from pizza sales

```
select
round(sum(od.quantity * p.price),2) as total_revenue
from pizzas as p
inner join order_details as od
on p.pizza_id = od.pizza_id;
```

insights - total revenue generated in 1 year -



total_revenue
817860.05

-- Q3 - Identify the highest-priced pizza.

```
SELECT
pt.name, pt.category, MAX(p.price) AS max_price,
p.pizza_id
FROM
pizza_types AS pt
INNER JOIN
pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
GROUP BY pt.name , pt.category , p.pizza_id
ORDER BY max_price DESC
LIMIT 1;
```

-- BOTH WAY IS CORRECT

```
SELECT
pt.name, pt.category, p.price, p.pizza_id
FROM
pizza_types AS pt
INNER JOIN
pizzas AS p ON pt.pizza_type_id = p.pizza_type_id
ORDER BY p.price DESC
LIMIT 1;
```

	name	category	price	pizza_id
▶	The Greek Pizza	Classic	35.95	the_greek_xxl

-- Q4 - Identify the most common pizza size ordered.

```
SELECT
p.size, COUNT(od.quantity) AS total_number_of_pizza_ordered
FROM
pizzas AS p
 INNER JOIN
order_details AS od ON p.pizza_id = od.pizza_id
GROUP BY p.size
ORDER BY total_number_of_pizza_ordered DESC;
```

insights - commonly ‘L’ size pizza ordered during entire sales-

size	total_number_of_pizza_ordered
L	18526
M	15385
S	14137

-- Q5 - List the top 5 most ordered pizza types along with their quantities.

```
SELECT
pt.pizza_type_id,
pt.name,
SUM(od.quantity) AS total_quantity
FROM
pizzas AS p
 INNER JOIN
pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
INNER JOIN
order_details AS od ON p.pizza_id = od.pizza_id
GROUP BY pt.pizza_type_id , pt.name
ORDER BY total_quantity DESC
LIMIT 5;
```

pizza_type_id	name	total_quantity
classic_dlx	The Classic Deluxe Pizza	2453
bbq_ckn	The Barbecue Chicken Pizza	2432
hawaiian	The Hawaiian Pizza	2422
pepperoni	The Pepperoni Pizza	2418
thai_ckn	The Thai Chicken Pizza	2371

Q7 - Join the necessary tables to find the total quantity of each pizza category ordered.

```
SELECT
pt.category, SUM(od.quantity) AS total_quantity
FROM
pizzas AS p
 INNER JOIN
pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
INNER JOIN
order_details AS od ON p.pizza_id = od.pizza_id
GROUP BY pt.category
ORDER BY total_quantity desc;
```

‘insights’ - so i can say
classic category pizzas is ordered highest compare to others

category	total_quantity
Classic	14888
Supreme	11987
Veggie	11649
Chicken	11050

-- Q8 - Determine the distribution of orders by hour of the day.

```
SELECT
count(order_id),

EXTRACT(HOUR FROM order_time) AS hours
FROM
orders
group by hours;
```

insights - At the time of afternoon and evening order is very high.

count(order_id)	hours
1231	11
2520	12
2455	13
1472	14
1468	15
1920	16
2336	17
2399	18

-- Q9 - find the category-wise distribution of pizzas.

```
SELECT
    category,
    COUNT(name) AS total_number_of_pizza_in_each_category
FROM
    pizza_types
GROUP BY category
ORDER BY total_number_of_pizza_in_each_category DESC;
```

insights :-

in ‘supreme’ and ‘veggie’ category have hightest number of pizza types compare to others

category	total_number_of_pizza_in_each_category
Supreme	9
Veggie	9
Classic	8
Chicken	6

-- Q 10 - Join relevant tables to find the category-wise distribution of total ordered pizzas along their total prices.

```
SELECT
pt.category,
COUNT(od.quantity) AS total_grouped_quantity,
SUM(od.quantity * p.price) AS total_grouped_price
FROM
pizzas AS p
 INNER JOIN
pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
INNER JOIN
order_details AS od ON p.pizza_id = od.pizza_id
GROUP BY pt.category
ORDER BY total_grouped_price DESC;
```

category	total_grouped_quantity	total_grouped_price
Classic	14579	220053.1
Supreme	11777	208197
Chicken	10815	195919.5
Veggie	11449	193690.45

insights: -

you can see that the highest quantity of pizzas sales in “classic” category as well as revenue is also highest

-- Q11 - Group the orders by date and calculate the average number of pizzas ordered per day.

```
SELECT
ROUND(AVG(x.total_quantity), 0) AS average_pizza_ordered_per_day
FROM
(SELECT
o.order_date, SUM(od.quantity) AS total_quantity
FROM
orders AS o
INNER JOIN order_details AS od ON o.order_id = od.order_id
GROUP BY order_date) AS x;
```

average_pizza_ordered_per_day
138

-- Q12 - Determine the top 3 most ordered pizza types based on revenue.

```
select
pt.name,
sum(od.quantity * p.price) as total_revenue
from pizzas as p
inner join pizza_types as pt
on p.pizza_type_id = pt.pizza_type_id
inner join order_details as od
on p.pizza_id = od.pizza_id
group by pt.name
order by total_revenue desc limit 3;
```

name	total_revenue
The Thai Chicken Pizza	43434.25
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41409.5

insights: -

in “chicken” category most common ordered pizza is “The Thai Chicken Pizza compare to others Chicken Type Pizza”

-- Q13 - Calculate the percentage contribution of each pizza type to total revenue.

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

category	total_percent_revenue_of_each_category
Classic	26.91
Veggie	23.68
Supreme	25.46
Chicken	23.96

-- Q14 - Analyze the cumulative revenue/running total revenue generated over time.

```
select
x.order_date,
x.total_revenue,
sum(x.total_revenue) over(order by x.order_date) as cumulative_revenue
from(select
o.order_date,
round(sum(od.quantity * p.price),0) as total_revenue
from orders as o
inner join order_details as od
on o.order_id = od.order_id
inner join pizzas as p
on p.pizza_id = od.pizza_id
group by o.order_date) as x
order by cumulative_revenue asc;
```

order_date	total_revenue	cumulative_revenue
2015-01-01	2714	2714
2015-01-02	2732	5446
2015-01-03	2662	8108
2015-01-04	1755	9863
2015-01-05	2066	11929

insights:-

Revenue increase day by day over the peroid of time

-- Q15 - Determine the top 3 most ordered pizza types based on revenue for each pizza category.

```
select
y.category,
y.name,
round(y.total_revenue,2) as revenue
from(select
x.category,
x.name,
x.total_revenue,
dense_rank() over(partition by category order by total_revenue desc) as rn
from(select
pt.category as category,
pt.name as name,
sum(od.quantity * p.price) as total_revenue
from pizzas as p
inner join pizza_types as pt
on p.pizza_type_id = pt.pizza_type_id
inner join order_details as od
on p.pizza_id = od.pizza_id
group by pt.category , pt.name) as x
) as y
where rn in (1,2,3)
```

category	name	revenue
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

-- Q16 - Determine the total revenue between "14-08-2015" and "31-12-2015"(including) based on their each category.

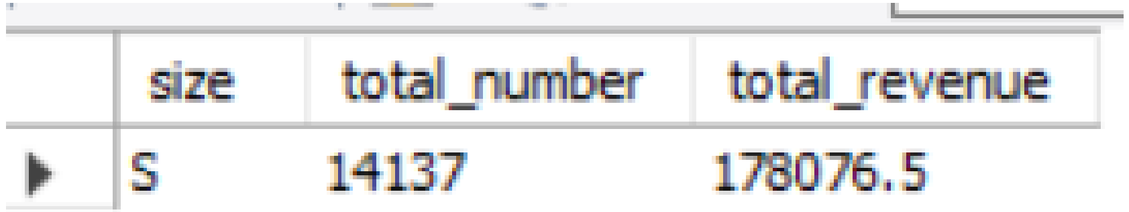
```
select
x.order_date,
x.category,
round(x.total_revenue,1) as total_revenue
from(
select
pt.category,
o.order_date,
sum(od.quantity * p.price) as total_revenue
from orders as o
inner join order_details as od
on o.order_id = od.order_id
inner join pizzas as p
on p.pizza_id = od.pizza_id
inner join pizza_types as pt
on p.pizza_type_id = pt.pizza_type_id
group by pt.category, o.order_date
) as x
where (x.order_date) > "2015-08-14" and  (x.order_date) <= "2015-12-31"
order by order_date asc;
```

order_date	category	total_revenue
2015-08-15	Classic	667.5
2015-08-15	Supreme	600.4
2015-08-15	Chicken	380.5
2015-08-15	Veggie	604
2015-08-16	Supreme	652.7
2015-08-16	Veggie	506.6
2015-08-16	Chicken	472.2
2015-08-16	Classic	507.8

-- Q17 - Determine the total number of small pizza size along with total revenue.

-- 'CASE WHEN' in 'SELECT' clause - executes after 'group by' and on aggregated data.
-- 'CASE WHEN' in 'HAVING' clause - also executes after 'group by' and on aggregated data.

```
SELECT
x.size, x.total_number, x.total_revenue
FROM
(SELECT
  p.size,
   CASE
WHEN COUNT(*) > 14000 AND COUNT(*) < 14500 THEN 1
ELSE 0
END AS small_size,
COUNT(*) AS total_number,
ROUND(SUM(od.quantity * p.price), 2) AS total_revenue
FROM
pizzas AS p
INNER JOIN order_details AS od ON p.pizza_id = od.pizza_id
INNER JOIN pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY p.size
ORDER BY p.size DESC) AS x
WHERE
x.small_size = 1;
```



size	total_number	total_revenue
S	14137	178076.5

-- Q18 - find all the details of only chicken pizza types along with their name.

```
select
category,
name
from pizza_types
where category = "chicken";
```

	category	name
▶	Chicken	The Barbecue Chicken Pizza
	Chicken	The California Chicken Pizza
	Chicken	The Chicken Alfredo Pizza
	Chicken	The Chicken Pesto Pizza
	Chicken	The Southwest Chicken Pizza
	Chicken	The Thai Chicken Pizza

-- Q19 - Determine the total revenue of March-Month along with their category,quantity.

```
SELECT
x.*
FROM
(SELECT
pt.category,
EXTRACT(MONTH FROM o.order_date) AS months,
COUNT(od.quantity) AS total_quantity,
ROUND(SUM(od.quantity * p.price), 2) AS total_revenue
FROM
orders AS o
INNER JOIN order_details AS od ON o.order_id = od.order_id
INNER JOIN pizzas AS p ON p.pizza_id = od.pizza_id
INNER JOIN pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY pt.category , months) AS x
WHERE
x.months = 3
```

category	months	total_quantity	total_revenue
Classic	3	1211	18116.6
Chicken	3	981	17625.5
Veggie	3	1023	17335.35
Supreme	3	971	17319.65

insights:-

In march-month
compare to all categories ,”classic category” have highest number of sales including revenue