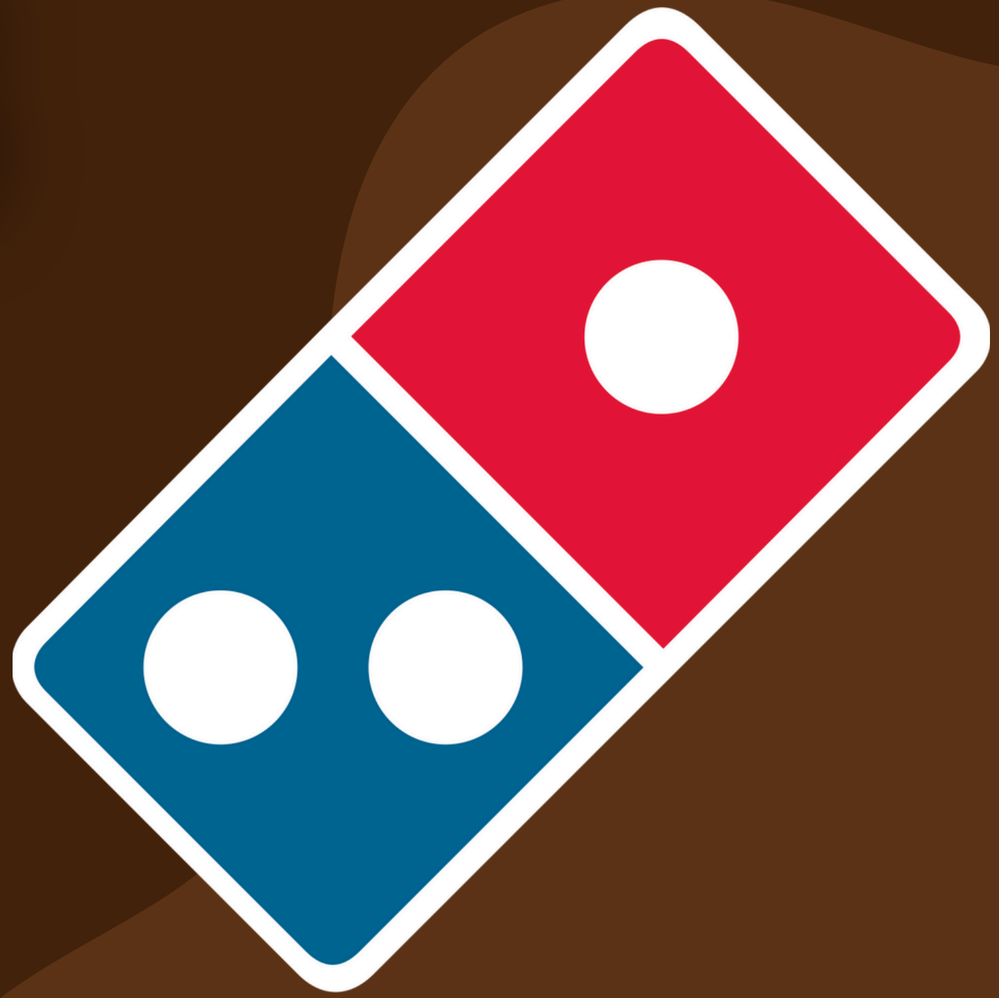
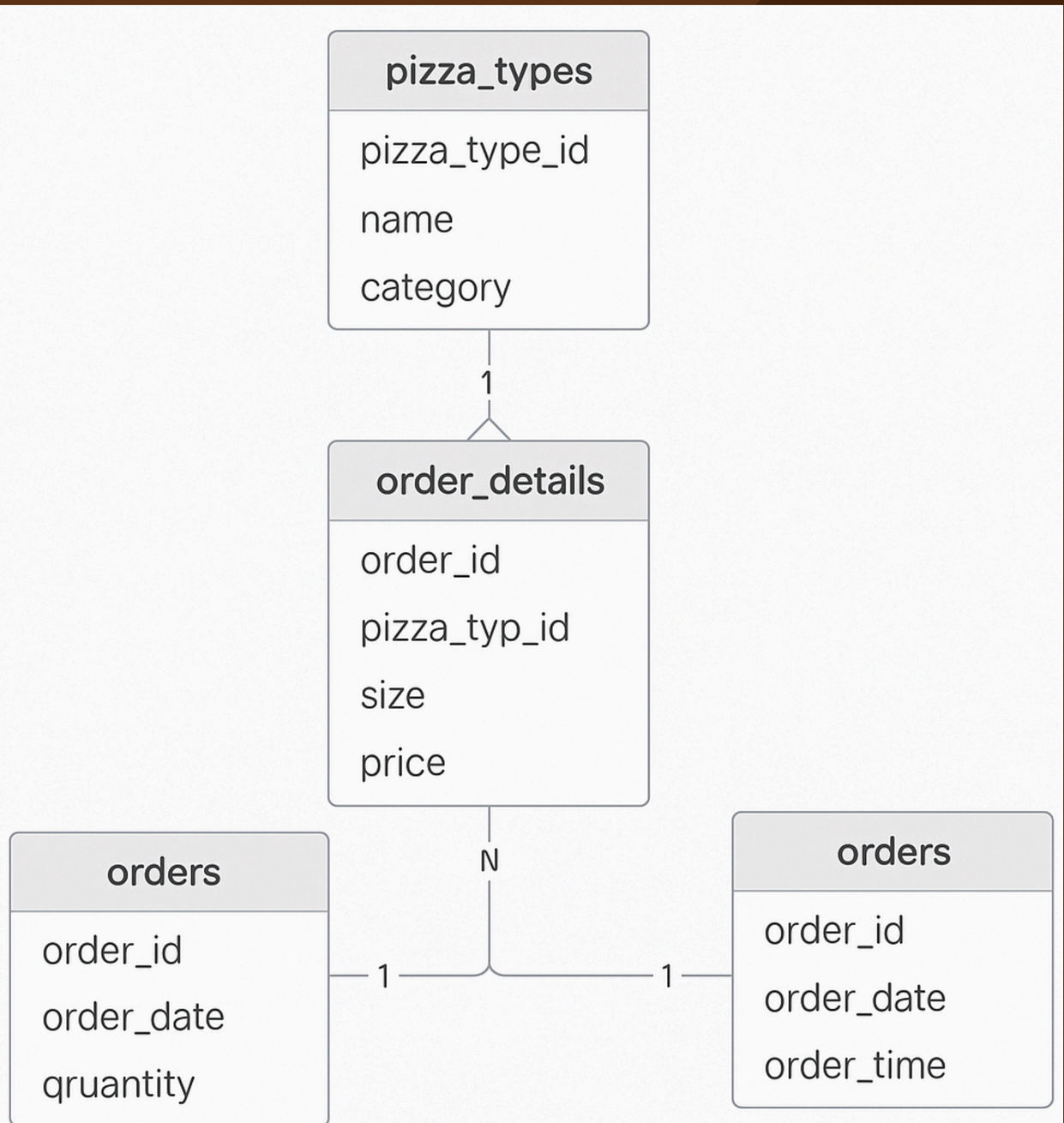
PIZZA SALES ANALYSIS

https://www..dominos.co.in





SCHEMA

This schema represents a relational database for a pizza ordering system, comprising four core tables:

orders: Contains information about each order, including the order ID,date, and time.

order_details: Acts as a junction table connecting orders to the pizzas ordered. It tracks quantity and links to both the orders and pizzas tables.

pizzas: Represents individual pizza offerings with attributes such as size and price, and a foreign key to pizza_types.

pizza_types: Describes the general category and name of the pizza, such as "Veggie" "Meat Lovers".

QUESTIONS COVERED

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.

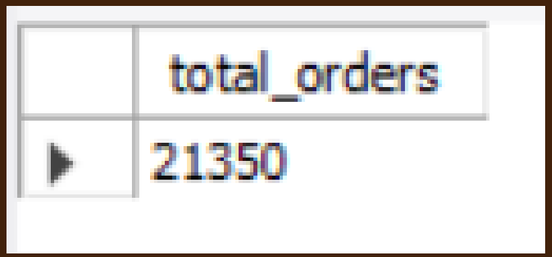
ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

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

Retrieve the total number of orders placed.

```
select count(order_id) total_orders
from orders;
```

Result:



	total_orders
▶	21350

Calculate the total revenue generated from pizza sales.

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

total_revenue
817860.05

Identify the highest-priced pizza.

```sql
select pizza_types.name, pizzas.price
from pizzas
inner join pizza_types
on pizza_types.pizza_type_id = pizzas.pizza_type_id
order by pizzas.price desc
limit 1;
```

	name	price
▶	The Greek Pizza	35.95

Identify the most common pizza size ordered.

```
select pizzas.size, count(order_details.quantity) order_cnt
from pizzas
inner join order_details
on pizzas.pizza_id = order_details.pizza_id
group by pizzas.size
order by order_cnt desc ;
```

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

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

```
select sum(order_details.quantity), pizza_types.name
from pizza_types
inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.name
order by count(order_details.quantity) desc
limit 5;
```

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

```
select  pizza_types.category, sum(order_details.quantity) quantity
from pizza_types
inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.category
order by quantity desc
```

	category	quantity
▶	Classic	14888
	Supreme	11987
	Veggie	11649
	Chicken	11050

Determine the distribution of orders by hour of the day.

```
select hour(order_time), count(order_id)
from orders
group by hour(order_time);
```

hour(order_time)	count(order_id)
11	1231
12	2520
13	2455
14	1472
15	1468
16	1920
17	2336
18	2399
19	2009
20	1642
21	1198
22	663
23	28
10	8
9	1

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

```
select count(pizza_type_id), category
from pizza_types
group by category;
```

	count(pizza_type_id)	category
▶	6	Chicken
	8	Classic
	9	Supreme
	9	Veggie

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

```
select round(avg(no_of_pizza),0) order_per_day
from(
select orders.order_date date, sum(order_details.quantity) no_of_pizza
from orders
inner join order_details
on orders.order_id = order_details.order_id
group by orders.order_date) tbl;
```

	order_per_day
▶	138

Determine the top 3 most ordered pizza based on revenue.

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

	name	revenue
▶	The Thai Chicken Pizza	43434.25
	The Barbecue Chicken Pizza	42768
	The California Chicken Pizza	41409.5

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

```
select category, round((revenue/sum(revenue) over()*100),2)
from(
select pizza_types.category category, round(sum(pizzas.price * order_details.quantity),2) revenue
from pizza_types
inner join pizzas
on pizza_types.pizza_type_id = pizzas.pizza_type_id
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
group by pizza_types.category
order by revenue desc) tbl
group by category;
```

	category	round((revenue/sum(revenue) over()*100),2)
▶	Classic	26.91
	Supreme	25.46
	Chicken	23.96
	Veggie	23.68

25.46

Analyze the cumulative revenue generated over time.

```
select order_date, revenue, sum(revenue) over(order by order_date) cumulative_renenue
from(
select orders.order_date, round(sum(pizzas.price * order_details.quantity),2) revenue
from order_details
inner join pizzas
on order_details.pizza_id = pizzas.pizza_id
inner join orders
on orders.order_id = order_details.order_id
group by orders.order_date) tbl;
```

order_date	revenue	cumulative_renenue
2015-01-01	2713.85	2713.85
2015-01-02	2731.9	5445.75
2015-01-03	2662.4	8108.15
2015-01-04	1755.45	9863.6
2015-01-05	2065.95	11929.55

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

```sql
select category, name, rnk
from(
select category, name, revenue, rank() over(partition by category order by revenue desc) rnk
from(
select pizza_types.category, pizza_types.name , round(sum(pizzas.price * order_details.quantity), 2) revenue
from pizzas
inner join order_details
on order_details.pizza_id = pizzas.pizza_id
inner join pizza_types
on pizza_types.pizza_type_id = pizzas.pizza_type_id
group by pizza_types.category,  pizza_types.name) tbl) tbl2
where rnk<=3;
```

category	name	rnk
Chicken	The Thai Chicken Pizza	1
Chicken	The Barbecue Chicken Pizza	2
Chicken	The California Chicken Pizza	3
Classic	The Classic Deluxe Pizza	1
Classic	The Hawaiian Pizza	2
Classic	The Pepperoni Pizza	3