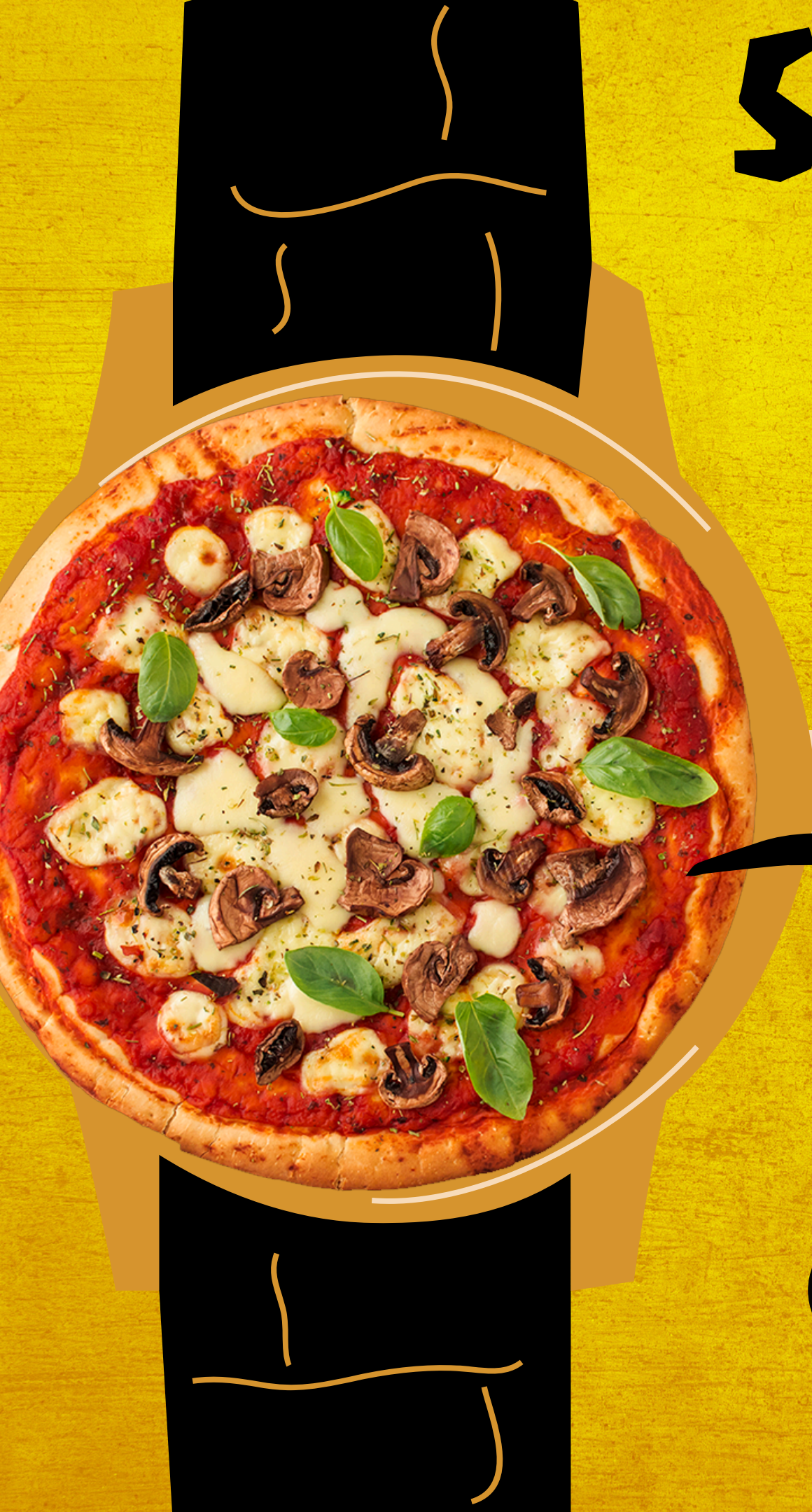
SQL

project portfolio

ON Pizza_Hut

sales analysis



BY: HARSHIT RAJ

PIZZA_SALES _ANALYSIS

Objective: The objective of this project is to analyze sales data from a pizza_hut restaurant chain to gain insights into customer preferences, sales trends, popular menu items, peak hours, and other relevant metrics.

DATA SOURCES:

1.Orders Table: The ordes table contains orders_id, Date of order and the time of order placed

2.Orders_details Table: The table contain details of order placed in four column having order_details_id,order_id, Pizza_id, quantity.

3.Pizza_types Table: the table contains varieties in pizzas which specifies the type and categories of pizza.The coloumn contains pizza_type_id,name, categories and pizza_id.

4.pizzas Table: The table contains the coloumn pizza_id,pizza_type_id and size.

1.Retrieve the total number of orders placed.

```sql
-- retrieve the total number of order placed
use pizzahut;
SELECT
    COUNT(order_id) AS total_orderplaced
FROM
    orders;
```

Result Grid

total_orderplaced
21350

2.Calculate the total revenue generated from pizza sales.

```sql
-- Calculate the total revenue generated from pizza sales

SELECT
    SUM(p.price * o.quantity) AS total_revenue
FROM
    pizzas p
        INNER JOIN
    order_details o ON p.pizza_id = o.pizza_id;
```

Result Grid | Filter Rows:

total_revenue
817860.049999993

3.Identify the highest-priced pizza.

```sql
-- Identify the highest-priced pizza.

SELECT
    pizza_type_id,price
FROM
    pizzas
WHERE
    price = (SELECT
            MAX(price)
        FROM
            pizzas)
```

Result Grid | Filter Rows:

	pizza_type_id	price
▶	the_greek	35.95

4.Identify the most common pizza size ordered.

```sql
-- Identify the most common pizza size ordered.

SELECT
    p.size, COUNT(o.quantity) AS total_orders
FROM
    pizzas p
        INNER JOIN
    order_details o ON p.pizza_id = o.pizza_id
GROUP BY p.size
ORDER BY total_orders DESC
LIMIT 1
```

Save the script to a file.

Result Grid | Filter Rows:

size	total_orders
L	18526

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

```sql
-- List the top 5 most ordered pizza types along with their quantities
select * from order_details;
select * from pizza_types;
select * from pizzas;
SELECT
    pizza_types.name,
    SUM(order_details.quantity) AS total_quantity
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.name
ORDER BY total_quantity DESC
LIMIT 5
```

Result Grid | Filter Rows:

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

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

```sql
SELECT
    p1.category, COUNT(o.quantity) AS total
FROM
    pizzas p
        INNER JOIN
    order_details o ON o.pizza_id = p.pizza_id
        JOIN
    pizza_types p1 ON p.pizza_type_id = p1.pizza_type_id
GROUP BY p1.category
```

Result Grid | Filter Rows:

category	total
Classic	14579
Veggie	11449
Supreme	11777
Chicken	10815

7.Determine the distribution of orders by hour of the day.

```sql
-- Determine the distribution of orders by hour of the day.
SELECT
    HOUR(o2.time) AS time, COUNT(o2.order_id) AS total_orders
FROM
    order_details o1
        INNER JOIN
    orders o2 ON o1.order_id = o2.order_id
GROUP BY HOUR(o2.time)
```

Result Grid | Filter Ro

time	total_orders
11	2672
12	6543
13	6203
14	3521
15	3170
16	4185
17	5143
18	5359
19	4350
20	3487
21	2528

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

```sql
-- Join relevant tables to find the category-wise distribution of pizzas.
  SELECT
      category, COUNT(name)
FROM
      pizza_types
GROUP BY category
```

Result Grid | Filter Rows:

category	COUNT(name)
Chicken	6
Classic	8
Supreme	9
Veggie	9

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

```sql
-- Group the orders by date and calculate the average number of pizzas ordered per day.
SELECT
    ROUND(AVG(quantity))
FROM
    (SELECT
        o1.date, COUNT(o2.quantity) AS quantity
    FROM
        orders o1
    INNER JOIN order_details o2 ON o1.order_id = o2.order_id
    GROUP BY o1.date) AS order_quantity
```

Result Grid | Filter

	average_number
▶	136

136

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

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

select p1.name,round(sum((p.price * o.quantity))) as revenue from pizzas p inner join  order_details o
on o.pizza_id=p.pizza_id join pizza_types p1
on p.pizza_type_id=p1.pizza_type_id
group by p1.name
order by revenue desc
limit 3
```

Result Grid | Filter Rows:

name	revenue
The Thai Chicken Pizza	43434
The Barbecue Chicken Pizza	42768
The California Chicken Pizza	41410

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

```sql
-- Calculate the percentage contribution of each pizza type to total revenue.
SELECT
    p1.category,
    ROUND(SUM(p.price * o.quantity * 100) / (SELECT
                    SUM(p.price * o.quantity) AS total_revenue
                FROM
                    pizzas p
                        INNER JOIN
                    order_details o ON p.pizza_id = o.pizza_id)) AS percentage
FROM
    pizzas p
        INNER JOIN
    order_details o ON o.pizza_id = p.pizza_id
        JOIN
    pizza_types p1 ON p.pizza_type_id = p1.pizza_type_id
GROUP BY p1.category
```

Result Grid | Filter Rows:

category	percentage
Classic	27
Veggie	24
Supreme	25
Chicken	24

12.Analyze the cumulative revenue generated over time.

```
2
3   select date, sum(revenue) over(order by date) as cumm_revenue from
4   (select o2.date,round(sum((o1.quantity*p.price))) as revenue from order_details o1 inner join orders o2 on
5   o1.order_id=o2.order_id join  pizzas p on
6   o1.pizza_id=p.pizza_id
7   group by o2.date) as sales
```

Result Grid | Filter Rows:

date	cumm_revenue
2015-01-01	2714
2015-01-02	5446
2015-01-03	8108
2015-01-04	9863
2015-01-05	11929
2015-01-06	14358
2015-01-07	16560
2015-01-08	19398
2015-01-09	21525
2015-01-10	23989

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

```sql
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
select name , revenue from
(select name, category,revenue, rank()
over(partition by category order by revenue desc) as rn from
(select p1.name,p1.category,
round(sum((p.price * o.quantity))) as revenue
 from pizzas p inner join  order_details o
on o.pizza_id=p.pizza_id join pizza_types p1
on p.pizza_type_id=p1.pizza_type_id
group by p1.name,p1.category) as total) as ranking
where rn<4;
```

Result Grid | Filter Rows:

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