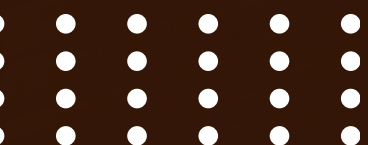


SQL PROJECT ON PIZZA SALES ANALYSIS REPORT

Start Your Slide

**ORDER
NOW**



HELLO MY NAME DEEPAK PRAJAPATI

In this project, I have utilized SQL queries
to solve questions related to pizza sales

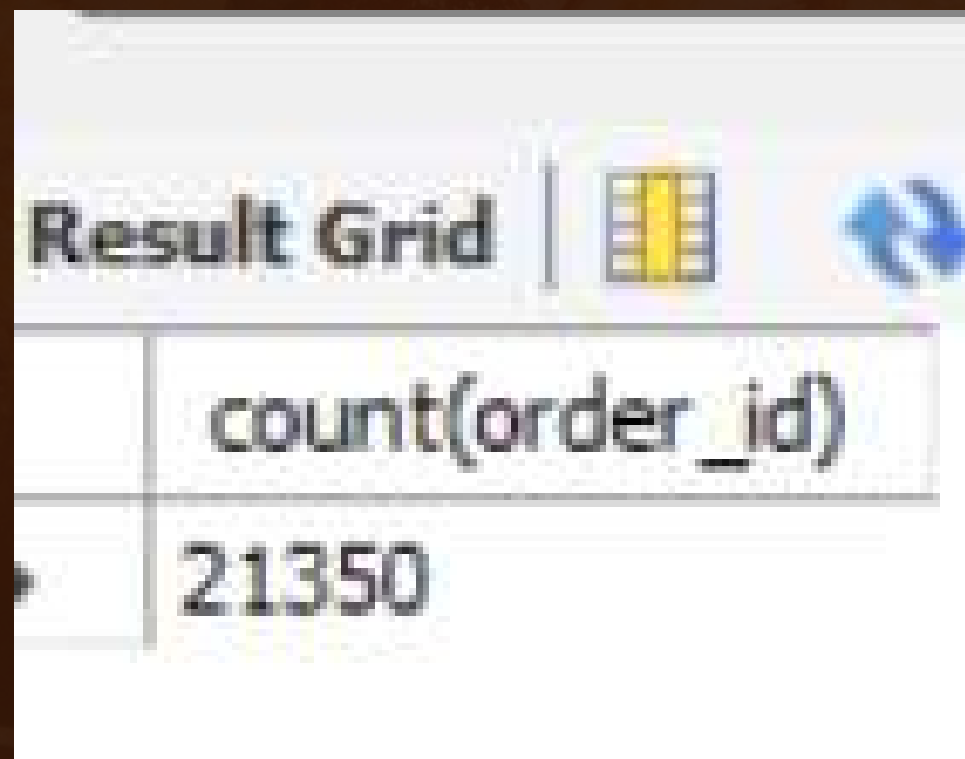


1.RETRIEVE THE TOTAL NUMBER OF ORDERS PLACED.

```
use pizzahut;
```

```
-- 1. Retrieve the total number of orders placed
```

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



The screenshot shows a 'Result Grid' window with a yellow header bar and a blue refresh icon. The grid contains one column labeled 'count(order_id)' and one row with the value '21350'.

count(order_id)
21350

2.CALCULATE THE TOTAL REVENUE GENERATED FROM PIZZA SALES.

```
SELECT
    ROUND(SUM((order_details.quantity * pizzas.price)),
          2) AS total_revenue
FROM
    order_details
    JOIN
    pizzas ON pizzas.pizza_id = order_details.pizza_id;
```

Result Grid	
	total_revenue
▶	817860.05

3. IDENTIFY THE MOST COMMON PIZZA SIZE ORDERED.

```
select pizzas.size, count(order_details.order_details_id) as max_sell_size
from pizzas join order_details
on pizzas.pizza_id = order_details.pizza_id
group by pizzas.size order by max_sell_size desc limit 1;
```

Result Grid			Filter Rows
	size	max_sell_size	
▶	L	18526	

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

```
select pizza_types.name,  
sum(order_details.quantity) as top_5  
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 top_5 desc limit 5;
```

Result Grid   Filter Rows:		
	name	top_5
•	The Classic Deluxe Pizza	2453
	The Barbecue Chicken Pizza	2432
	The Hawaiian Pizza	2422
	The Pepperoni Pizza	2418
	The Thai Chicken Pizza	2371

5.IDENTIFY THE HIGHEST-PRICED PIZZA.

```
select  
pizza_types.name ,pizzas.price  
from pizza_types  
join pizzas  
on pizza_types.pizza_type_id = pizzas.pizza_type_id  
ORDER BY pizzas.price DESC LIMIT 1;
```

Result Grid			Filter Rows:
	name	price	
▶	The Greek Pizza	35.95	

6. DETERMINE THE DISTRIBUTION OF ORDERS BY HOUR OF THE DAY

```
select hour(order_time) as hour1, count(order_id) from orders  
group by hour1;
```

	hour1	count(order_id)
▶	11	1231
	12	2520
	13	2455
	14	1472
	15	1468

7. JOIN RELEVANT TABLES TO FIND THE CATEGORY-WISE DISTRIBUTION OF PIZZAS.

```
select pizza_types.category as category, sum(order_details.order_details_id) as sell
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 category order by sell desc;
```

Result Grid		Filter Rows:
	category	sell
•	Classic	355542220
	Supreme	286412905
	Ve Supreme	16348618
	Chicken	263672767

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

```
select avg(avgvise) from
(select orders.order_date , sum(order_details.quantity) as avgvise from
orders join order_details
on orders.order_id = order_details.order_id
group by orders.order_date) as order_quantity ;
```

Result Grid 	
	avg(avgvise)
▶	138.4749

9.DETERMINE THE TOP 3 MOST ORDERED PIZZA TYPES BASED ON REVENUE.

```
select pizza_types.name,  
sum(order_details.quantity * pizzas.price) 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 pizza_types.name order by revenue desc limit 3;
```

Result Grid			Filter Rows:
	name	revenue	
▶	The Thai Chicken Pizza	43434.25	
	The Barbecue Chicken Pizza	42768	
	The California Chicken Pizza	41409.5	

10. CALCULATE THE PERCENTAGE CONTRIBUTION OF EACH PIZZA TYPE TO TOTAL REVENUE.

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

Result Grid			Filter Rows:
	category	revenue	
▶	Classic	26.90596025566967	
	Supreme	25.45631126009862	
	Chicken	23.955137556847287	
	Veggie	23.682590927384577	

11.ANALYZE THE CUMULATIVE REVENUE GENERATED OVER TIME.

```
select order_date, sum(revenue)
over(order by order_date) as cum_revenue from
(select orders.order_date,
sum(order_details.quantity * pizzas.price) 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 orders.order_date) as sales;
```

Result Grid		Filter Rows:	
	order_date	cum_revenue	
▶	2015-01-01	2713.8500000000000004	
	2015-01-02	5445.75	5445.75
	2015-01-03	8108.15	
	2015-01-04	9863.6	
	2015-01-05	11929.55	

Result 26

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

```
select name, revenue from
(select category, name, revenue ,
rank() over(partition by category order by revenue desc) as rn
from
(select pizza_types.category, pizza_types.name,
sum((order_details.quantity) * pizzas.price) 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 pizza_types.category, pizza_types.name) as a) as b
where rn <= 3;
```

RESULT GRID			Filter Rows:	
	name	revenue		
▶	The Thai Chicken Pizza	43434.25		
	The Barbecue Chicken Pizza	42768		
	The California Chicken Pizza	41409.5		
	The Classic Deluxe Pizza	38180.5		
	The Hawaiian Pizza	32273.25		

**THANK YOU
FOR ATTENTION**

