

PIZZA HUT SALES SQL ANALYSIS

BUSINESS INSIGHTS USING SQL QUERIES



TOTAL ORDERS PLACED



THIS QUERY COUNTS THE TOTAL NUMBER OF UNIQUE ORDERS IN THE DATASET.

```
1  -- Retrieve the total number of orders placed.
2  • use pizzahut;
3  • SELECT
4      COUNT(order_id) AS total_orders
5  FROM
6      orders;
```

Result Grid		Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	total_orders			
▶	66207			

TOTAL REVENUE



CALCULATES THE TOTAL REVENUE GENERATED FROM ALL PIZZA SALES.

```
1  -- Calculate the total revenue generated from pizza sales.
2  • SELECT
3      ROUND(SUM(order_details.quantity * pizzas.price),
4             2) AS total_revenue
5  FROM
6      order_details
7      JOIN
8      pizzas ON pizzas.pizza_id = order_details.pizza_id
9
```

Result Grid			Filter Rows: <input type="text"/>	Export:	Wrap Cell Content:
	total_revenue				
	817860.05				

Result Grid

Filter Rows:

Export:

Wrap Cell Content:

Fetch rows:

	name	price
▶	The Greek Pizza	35.95

	name	price
▶	The Greek Pizza	35.95

MOST COMMON PIZZA SIZE



Finds the most frequently ordered pizza size.

```
1  -- Identify the most common pizza size ordered.
2  ●  SELECT
3      pizzas.size,
4      COUNT(order_details.order_details_id) AS order_count
5  FROM
6      pizzas
7      JOIN
8      order_details ON pizzas.pizza_id = order_details.pizza_id
9  GROUP BY pizzas.size
10 ORDER BY order_count DESC;
```

Result Grid | | Filter Rows: | Export: | Wrap Cell Content:

	size	order_count
▶	L	18526
	M	15385
	S	14137
	XL	544
	XXL	28

TOP 5 PIZZA TYPES BY QUANTITY

LISTS THE FIVE MOST ORDERED PIZZA TYPES BASED ON TOTAL QUANTITY SOLD.

```
1  -- List the top 5 most ordered pizza types along with their quantities.
2  • SELECT
3      pizza_types.name, SUM(order_details.quantity) AS quantity
4  FROM
5      pizza_types
6      JOIN
7      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8      JOIN
9      order_details ON order_details.pizza_id = pizzas.pizza_id
10 GROUP BY pizza_types.name
11 ORDER BY quantity DESC
12 LIMIT 5;
```

Result Grid		Filter Rows:	Exports:	Wrap Cell Content:	Fetch rows:
	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			

TOTAL QUANTITY BY CATEGORY



SUMMARIZES THE TOTAL QUANTITY SOLD FOR EACH PIZZA CATEGORY.

```
1  -- Join the necessary tables to find the total quantity of each pizza category ordered
2  • SELECT
3      pizza_types.category,
4      SUM(order_details.quantity) AS total_qty
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
9      JOIN
10     order_details ON order_details.pizza_id = pizzas.pizza_id
11     GROUP BY pizza_types.category
12     ORDER BY total_qty
13
```

Result Grid	Filter Rows:	Exports	Wrap Cell Content:
	category	total_qty	
▶	Chicken	11050	
	Veggie	11649	
	Supreme	11987	
	Classic	14888	

CATEGORY-WISE PIZZA DISTRIBUTION

SHOWS HOW PIZZA ORDERS ARE DISTRIBUTED ACROSS DIFFERENT CATEGORIES.

```
1  -- Join relevant tables to find the category-wise distribution of pizzas.
2  • SELECT
3      pizza_types.category, COUNT(order_details.order_id)
4  FROM
5      pizza_types
6      JOIN
7      pizzas ON pizza_types.pizza_type_id = pizzas.pizza_type_id
8      JOIN
9      order_details ON pizzas.pizza_id = order_details.pizza_id
10 GROUP BY category;
```

Result Grid		Filter Rows:	Export:	Wrap Cell Content:
category	COUNT(order_details.order_id)			
Classic	14579			
Veggie	11449			
Supreme	11777			
Chicken	10815			

AVERAGE PIZZAS ORDERED PER DAY

CALCULATES THE AVERAGE NUMBER OF PIZZAS ORDERED PER DAY.

```
1  -- Group the orders by date and calculate the average number of pizzas ordered per day.
2  • SELECT
3      ROUND(AVG(quantity), 0) as avg_pizza_ordered_per_day
4  FROM
5      (SELECT
6          orders.date, SUM(order_details.quantity) AS quantity
7      FROM
8          orders
9      JOIN order_details ON orders.order_id = order_details.order_id
10     GROUP BY orders.date) AS order_quantity;
```

Result Grid			Filter Rows: <input type="text"/>	Export: 	Wrap Cell Content: 
	avg_pizza_ordered_per_day				
▶	429				

TOP 3 PIZZAS BY REVENUE

DISPLAYS THE THREE PIZZA TYPES THAT GENERATED THE HIGHEST REVENUE.

```
1  -- Determine the top 3 most ordered pizza types based on revenue.
2  • SELECT
3      pizza_types.name,
4      SUM(order_details.quantity * pizzas.price) AS Revenue
5  FROM
6      pizza_types
7      JOIN
8      pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
9      JOIN
10     order_details ON order_details.pizza_id = pizzas.pizza_id
11  GROUP BY pizza_types.name
12  ORDER BY revenue DESC
13  LIMIT 3;
```

Result Grid	Filter Rows:	Export:	Wrap Cell Content:	Fetch rows:
name		Revenue		
▶	The Thai Chicken Pizza	43434.25		
	The Barbecue Chicken Pizza	42768		
	The California Chicken Pizza	41409.5		

REVENUE CONTRIBUTION BY CATEGORY

SHOWS WHAT PERCENTAGE OF TOTAL REVENUE EACH CATEGORY CONTRIBUTES.

```
1  -- Calculate the percentage contribution of each pizza type to total revenue.
2  ● SELECT
3      pizza_types.category,
4      ROUND(SUM(order_details.quantity * pizzas.price) / (SELECT
5          ROUND(SUM(order_details.quantity * pizzas.price),
6              2) AS total_revenue
7      FROM
8          order_details
9          JOIN
10             pizzas ON pizzas.pizza_id = order_details.pizza_id) * 100,
11          2) AS revenue
12  FROM
13      pizza_types
14      JOIN
15          pizzas ON pizzas.pizza_type_id = pizza_types.pizza_type_id
16      JOIN
17          order_details ON order_details.pizza_id = pizzas.pizza_id
18  GROUP BY pizza_types.category
19  ORDER BY revenue DESC;
```

Result Grid			Filter Rows:
	category	revenue	
▶	Classic	26.91	
	Supreme	25.46	
	Chicken	23.96	
	Veggie	23.68	

CUMULATIVE REVENUE OVER TIME

ANALYZES HOW REVENUE HAS ACCUMULATED OVER TIME.

```
1  -- Analyze the cumulative revenue generated over time.
2  • SELECT
3      orders.date,
4      SUM(order_details.quantity * pizzas.price) AS daily_revenue,
5      SUM(SUM(order_details.quantity * pizzas.price)) OVER (ORDER BY orders.date)
6      AS cumulative_revenue
7  FROM
8      order_details
9  JOIN
10     pizzas ON order_details.pizza_id = pizzas.pizza_id
11  JOIN
12     orders ON orders.order_id = order_details.order_id
13  GROUP BY
14     orders.date
15  ORDER BY
16     orders.date;
```

Result Grid				Filter Rows:	Export:
	date	daily_revenue	cumulative_revenue		
▶	2015-01-01	8376.55	8376.55		
	2015-01-02	8655.199999999999	17031.75		
	2015-01-03	1999999997	25083.699999999997		
	2015-01-04	5516.8499999999985	30600.549999999996		
	2015-01-05	6418.3499999999985	37018.899999999994		
	2015-01-06	7637.749999999999	44656.649999999994		
	2015-01-07	7126.249999999997	51782.899999999994		
	2015-01-08	8790.05	60572.95		
	2015-01-09	6634.299999999997	67207.25		
	2015-01-10	7503.0999999999985	74710.35		

TOP 3 PIZZAS BY CATEGORY REVENUE

RANKS THE TOP 3 PIZZA TYPES BY REVENUE WITHIN EACH CATEGORY USING WINDOW FUNCTIONS.

```
-- Determine the top 3 most ordered pizza types based on revenue for each pizza category.
SELECT
  category,
  name,
  revenue
FROM (
  SELECT
    pt.category,
    pt.name,
    SUM(od.quantity * p.price) AS revenue,
    RANK() OVER (PARTITION BY pt.category ORDER BY SUM(od.quantity * p.price) DESC) AS rank_within_category
  FROM
    pizza_types pt
  JOIN
    pizzas p ON pt.pizza_type_id = p.pizza_type_id
  JOIN
    order_details od ON od.pizza_id = p.pizza_id
  GROUP BY
    pt.category, pt.name
) AS ranked
WHERE
  rank_within_category <= 3
ORDER BY
  category, revenue DESC limit 3;
```

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