Pizza Sales SQL Queries

Basic Queries:

Retrieve the total number of orders placed.

SQL: SELECT COUNT(*) AS total_orders FROM orders;

Calculate the total revenue generated from pizza sales.

SQL: SELECT SUM(order_details.quantity * pizzas.price) AS total_revenue FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id;

Identify the highest-priced pizza.

SQL: SELECT name, price FROM pizzas ORDER BY price DESC LIMIT 1;

Identify the most common pizza size ordered.

SQL: SELECT size, COUNT(*) AS count FROM pizzas JOIN order_details ON pizzas.pizza_id = order_details.pizza_id GROUP BY size ORDER BY count DESC LIMIT 1;

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

SQL: SELECT pizzas.name, SUM(order_details.quantity) AS total_quantity FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id GROUP BY pizzas.name ORDER BY total_quantity DESC LIMIT 5;

Intermediate Queries:

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

SQL: SELECT pizza_types.category, SUM(order_details.quantity) AS total_quantity FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id GROUP BY pizza_types.category;

Determine the distribution of orders by hour of the day.

SQL: SELECT EXTRACT(HOUR FROM order_time) AS hour, COUNT(*) AS order_count FROM orders GROUP BY hour ORDER BY hour;

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

SQL: SELECT pizza_types.category, COUNT(DISTINCT pizzas.pizza_id) AS pizza_count FROM pizzas JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id GROUP BY pizza_types.category;

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

SQL: SELECT order_date, AVG(order_details.quantity) AS avg_pizzas_per_order FROM orders JOIN

order_details ON orders.order_id = order_details.order_id GROUP BY order_date;

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

SQL: SELECT pizza_types.name, SUM(order_details.quantity * pizzas.price) AS revenue FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id GROUP BY pizza_types.name ORDER BY revenue DESC LIMIT 3;

Advanced Queries:

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

SQL: SELECT pizza_types.name, ROUND(SUM(order_details.quantity * pizzas.price) * 100.0 / (SELECT SUM(order_details.quantity * pizzas.price) FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id), 2) AS revenue_percentage FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id GROUP BY pizza_types.name ORDER BY revenue_percentage DESC;

Analyze the cumulative revenue generated over time.

SQL: SELECT order_date, SUM(order_details.quantity * pizzas.price) OVER (ORDER BY order_date) AS cumulative_revenue FROM orders JOIN order_details ON orders.order_id = order_details.order_id JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id;

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

SQL: SELECT category, name, revenue FROM (SELECT pizza_types.category, pizza_types.name, SUM(order_details.quantity * pizzas.price) AS revenue, ROW_NUMBER() OVER (PARTITION BY pizza_types.category ORDER BY SUM(order_details.quantity * pizzas.price) DESC) AS rank FROM order_details JOIN pizzas ON order_details.pizza_id = pizzas.pizza_id JOIN pizza_types ON pizzas.pizza_type_id = pizza_types.pizza_type_id GROUP BY pizza_types.category, pizza_types.name) AS ranked WHERE rank <= 3;