SQL Pizza Sales Analysis Project Queries

Database and Table Creation

```
CREATE DATABASE pizza;
USE pizza;

CREATE TABLE orders(
   order_id INT NOT NULL,
   order_date DATE NOT NULL,
   order_time TIME NOT NULL,
   PRIMARY KEY(order_id)
);

CREATE TABLE order_details(
   order_details_id INT NOT NULL,
   order_id INT NOT NULL,
   pizza_id TEXT NOT NULL,
   quantity INT NOT NULL,
   PRIMARY KEY(order_details_id)
);

 CREATE TABLE pizzas (
    pizza_id TEXT NOT NULL PRIMARY KEY,
    pizza_type_id TEXT NOT NULL,
    size VARCHAR(10) NOT NULL,
    price DOUBLE NOT NULL
 );


 CREATE TABLE pizza_types (
    pizza_type_id TEXT NOT NULL PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    category VARCHAR(50) NOT NULL,
    ingredients TEXT
 );
```

Analysis Questions and Queries

1. Retrieve the total number of orders placed.

```
SELECT
   COUNT(order_id) AS total_orders
FROM
   orders;
```

2. Calculate the total revenue 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;
```

3. 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;
```

4. Identify the most common pizza size ordered.

```
SELECT
   pizzas.size,
   COUNT(order_details.order_details_id) AS order_count
FROM
   pizzas
JOIN
   order_details ON pizzas.pizza_id = order_details.pizza_id
GROUP BY
   pizzas.size
ORDER BY
   order_count DESC;
```

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

```
SELECT
   pizza_types.name,
   SUM(order_details.quantity) AS 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
   Quantity DESC
LIMIT 5;
```

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

```
SELECT
   pizza_types.category,
   SUM(order_details.quantity) AS 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.category
ORDER BY
   Quantity DESC;
```

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

```
SELECT
   HOUR(order_time) AS hour,
   COUNT(order_id) AS order_count
FROM
   orders
GROUP BY
   HOUR(order_time);
```

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

```
SELECT
   category,
   COUNT(name)
FROM
   pizza_types
GROUP BY
   category;
```

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

```
SELECT
   ROUND(AVG(quantity), 0) AS avg_pizza_ordered_per_day
FROM
   (SELECT
      orders.order_date,
      SUM(order_details.quantity) AS Quantity
   FROM
      orders
   JOIN
      order_details ON orders.order_id = order_details.order_id
   GROUP BY
      orders.order_date) AS order_quantity;
```

10. 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 pizzas.pizza_type_id = pizza_types.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;
```

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

```
SELECT
   pizza_types.category,
   ROUND(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) * 100,2) AS
   revenue_percentage
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_percentage DESC;
```

12. 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;
```

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

```
SELECT
    category,
    name,
    revenue
FROM (
    SELECT
        pizza_types.category,
        pizza_types.name,
        SUM(order_details.quantity * pizzas.price) AS revenue,
        RANK() OVER (PARTITION BY pizza_types.category ORDER BY
SUM(order_details.quantity * pizzas.price) DESC) AS rn
    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
WHERE rn <= 3; -- Corrected from 'rn < 33' based on typical "top 3" requests
```