SOLUTIONS

#BASIC

1. **Retrieve the total number of orders placed.**

```
SELECT
    COUNT(*)
FROM
    order_details;
```

2. **Calculate the total revenue generated from pizza sales.**

```
SELECT
    ROUND(SUM(quantity * price), 2) AS revenue
FROM
    order_details
        JOIN
    pizzas USING (pizza_id);
```

3. **Identify the highest-priced pizza.**

```
SELECT
    name,
    price
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
ORDER BY price DESC
LIMIT 1;
```

4. **Identify the most common pizza size ordered.**

```
SELECT
    size,
    COUNT(quantity) AS most_ordered
FROM
    order_details
        JOIN
    pizzas USING (pizza_id)
GROUP BY size
ORDER BY most_ordered DESC
LIMIT 1;
```

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

```
SELECT
    name,
    SUM(    quantity) AS total_ordered_qty
FROM
    order_details
        JOIN
    pizzas USING (pizza_id)
        JOIN
    pizza_types USING (pizza_type_id)
GROUP BY name
ORDER BY total_ordered_qty DESC
LIMIT 5;
```

SOLUTIONS

#INTERMEDIATE

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

```
SELECT
    category,
    SUM(quantity) AS total_qty
FROM
    order_details
        JOIN
    pizzas USING (pizza_id)
        JOIN
    pizza_types USING (pizza_type_id)
GROUP BY category;
```

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

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

8. **Find the category-wise distribution of pizzas.**

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

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

```
WITH total_qty_by_date AS
(
SELECT
    order_date,
    SUM(quantity) AS total_qty
FROM
    orders
        JOIN
    order_details USING (order_id)
GROUP BY order_date
)
SELECT
    ROUND(AVG(total_qty),0)  AS avg_orders_per_day
FROM
    total_qty_by_date ;
```

SOLUTIONS

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

```
SELECT
    name,
    ROUND(SUM(quantity * price), 2) AS total_revenue
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    order_details USING (pizza_id)
GROUP BY name
ORDER BY total_revenue DESC
LIMIT 3;
```

#ADVANCED

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

```
WITH total_revenue AS (
SELECT
    name,
    ROUND(SUM(price * quantity), 2) AS revenue
FROM
    pizza_types
        JOIN
    pizzas USING (pizza_type_id)
        JOIN
    order_details USING (pizza_id)
GROUP BY name
)
SELECT
    name,
    CONCAT(
            ROUND(100 * revenue / sum(revenue) over() ,2) ,
            '%'
              )     AS pct_contribution
FROM
    total_revenue;
```

12. Analyze the cumulative revenue generated over time.

```
WITH total_revenue AS
(
SELECT
    order_date,
    ROUND(SUM(price * quantity), 2) AS revenue
FROM
    orders
        JOIN
    order_details USING (order_id)
        JOIN
    pizzas USING (pizza_id)
GROUP BY order_date
)

SELECT
        *,
```

SOLUTIONS

```
        ROUND(
              SUM(revenue) OVER (ORDER BY order_date),
              2
                  )  AS cum_revenue
FROM total_revenue;
```

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

```
WITH revenue_cte AS (
SELECT
    category,
    name as pizza_type,
    ROUND(SUM(price * quantity),2) AS revenue
FROM
    orders
        JOIN
    order_details USING (order_id)
        JOIN
    pizzas USING (pizza_id)
        JOIN
    pizza_types USING (pizza_type_id)
GROUP BY category,name
),
rank_cte AS (
select
    *,
    Rank() OVER(PARTITION BY category ORDER BY revenue DESC) as rnk
FROM
    revenue_cte
)
SELECT
    category,
    pizza_type,
    revenue
FROM
    rank_cte
WHERE
    rnk<=3;
```