```
-- Here are 30 SQL question prompts** based on the dataset:

### ◇ Basic Level
-- 1. List all unique pizza sizes available.
SELECT DISTINCT size from pizzas;

-- 2. Retrieve all orders placed on a specific date.
SELECT * FROM orders WHERE date = '2015-06-19';

-- 3. Find the total number of pizzas sold.
SELECT COUNT(*) FROM orders;

-- 4. Show the name and category of all pizza types.
SELECT name, category FROM pizza_types;

-- 5. Display all pizzas with a price greater than $15.
SELECT * FROM pizzas WHERE price > 15;

-- 6. List all pizzas along with their size and price.
SELECT pt.name,pt.category,p.size,p.price FROM pizzas as p
join pizza_types as pt
on pt.pizza_type_id=p.pizza_type_id;

-- 7. Count the number of different pizza types.
SELECT COUNT(DISTINCT(pizza_type_id)) as 'NUMBER OF DIFERENT TYPE OF PIZZA' FROM
  pizza_types;

-- 8. Show the total quantity of each pizza sold.
SELECT pizza_id,sum(quantity) as 'Total Quantity' FROM order_details GROUP BY
  pizza_id;

-- 9. Find the details of the most expensive pizza.
SELECT pt.*,p.price FROM pizza_types as pt
join pizzas as p
on pt.pizza_type_id=p.pizza_type_id
order by price desc limit 1;

-- 10. List all distinct ingredients used in any pizza.
SELECT DISTINCT(ingredients) FROM pizza_types;
-----------------------------------------------------------------------------------
  -----------------------------------------
### ◇ Intermediate Level
-- 11. Find total revenue generated from all orders.
SELECT ROUND(SUM(od.quantity*p.price),2) AS 'total revenue'
FROM order_details AS od
join pizzas AS p on od.pizza_id=p.pizza_id;

-- 12. Display the top 5 most ordered pizza types.
SELECT pt.*
```

```sql
FROM pizza_types AS pt JOIN pizzas AS p ON p.pizza_type_id=pt.pizza_type_id
JOIN order_details AS od ON od.pizza_id = p.pizza_id
GROUP BY p.pizza_type_id
ORDER BY COUNT(od.order_id) DESC LIMIT 5;

-- 13. Calculate the average price of pizzas by size.
SELECT ROUND(AVG(price),2) AS 'Average Price', size FROM pizzas
GROUP BY size;

-- 14. Get all pizza types that contain “cheese” in their ingredients .
SELECT * FROM pizza_types
WHERE ingredients LIKE '%cheese%';

-- 15. List the number of orders per day.
SELECT date, COUNT(order_id) FROM orders
GROUP BY date;

-- 16. Find the pizza with the highest total sales (quantity × price).
SELECT pt.*
FROM pizza_types AS pt JOIN pizzas AS p ON p.pizza_type_id=pt.pizza_type_id
JOIN order_details AS od ON od.pizza_id = p.pizza_id
GROUP BY p.pizza_type_id
ORDER BY COUNT(od.order_id) DESC LIMIT 1;

-- 17. Show the number of times each pizza type was ordered.
SELECT pt.*,COUNT(od.order_id) AS 'Count of Order'
FROM pizza_types AS pt JOIN pizzas AS p ON p.pizza_type_id=pt.pizza_type_id
JOIN order_details AS od ON od.pizza_id = p.pizza_id
GROUP BY p.pizza_type_id;

-- 18. Retrieve all orders that include a "Veggie" category pizza.
SELECT od.* FROM order_details AS od
LEFT JOIN pizzas as p on od.pizza_id = p.pizza_id
JOIN pizza_types as pt on pt.pizza_type_id = p.pizza_type_id
WHERE pt.category = 'Veggie';

-- 19. List pizzas that were never ordered.
SELECT p.pizza_id, pt.name AS pizza_name
FROM pizzas AS p
JOIN pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
LEFT JOIN order_details AS od ON p.pizza_id=od.pizza_id
WHERE od.order_id IS NULL;

-- 20. Get the time of the first and last order on a specific date.
SELECT
    MIN(time) AS 'First Order time',
    MAX(time) AS 'Last Order time'
FROM
    orders
```

```sql
WHERE date = '2015-07-15';
-------------------------------------------------------------------------------------------------------------------------------------------------------
### ◇ Advanced Level
-- 21. Calculate total sales (in $) per pizza category.
SELECT ROUND(SUM(p.price*od.quantity),2) AS 'Total Sales', pt.category
FROM pizza_types AS pt
JOIN pizzas AS p ON p.pizza_type_id = pt.pizza_type_id
JOIN order_details AS od ON od.pizza_id = p.pizza_id
GROUP BY pt.category;

-- 22. Find the most popular pizza size for each category.
WITH size_popularity AS (
    SELECT pt.category, p.size, SUM(od.quantity) AS Total_Quantity,
    ROW_NUMBER() OVER(PARTITION BY pt.category ORDER BY SUM(od.quantity) DESC) AS Rank_in_category
    FROM order_details AS od
    JOIN pizzas AS p ON od.pizza_id=p.pizza_id
    JOIN pizza_types as pt ON p.pizza_type_id = pt.pizza_type_id
    GROUP BY pt.category, p.size
)
SELECT category, size AS 'Most popular size', Total_Quantity
FROM size_popularity WHERE Rank_in_category = 1;

-- 23. Rank pizzas by revenue generated.
WITH revenue AS (
    SELECT ROUND(SUM(p.price*od.quantity),2) AS Total_revenue ,pt.name
    FROM pizza_types AS pt
    JOIN pizzas AS p ON pt.pizza_type_id=p.pizza_type_id
    JOIN order_details AS od ON od.pizza_id=p.pizza_id
    GROUP BY pt.name
),
ranked_pizzas AS (
SELECT name, Total_revenue, RANK() OVER (ORDER BY Total_revenue DESC) AS revenue_rank FROM revenue
)
SELECT * FROM ranked_pizzas;

-- 24. Retrieve customers who ordered more than 10 pizzas in a single order.
SELECT od_summary.order_id, od_summary.total_pizzas
FROM (
    SELECT order_id, SUM(quantity) AS total_pizzas FROM order_details
    GROUP BY order_id HAVING SUM(quantity) > 10
) AS od_summary
JOIN orders AS o ON od_summary.order_id= o.order_id;

-- 25. Display the monthly revenue trend.
SELECT DATE_FORMAT(o.date, '%Y-%m') AS month,
ROUND(SUM(od.quantity*p.price),2) AS total_revenue
```

```
FROM order_details AS od
JOIN orders AS o ON od.order_id=o.order_id
JOIN pizzas AS p ON od.pizza_id=p.pizza_id
GROUP BY DATE_FORMAT(o.date,'%Y-%m')
ORDER BY month;

-- 26. Find the pizza type with the widest variety of sizes.
WITH size_counts AS (
    SELECT pt.name AS pizza_type, COUNT(DISTINCT p.size) AS size_count
    FROM pizzas AS p
    JOIN pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
    GROUP BY pt.name
),
max_size_count AS (
    SELECT MAX(size_count) AS max_count
    FROM size_counts
)
SELECT sc.pizza_type,sc.size_count
FROM size_counts AS sc
JOIN max_size_count AS msc ON sc.size_count = msc.max_count;

-- 27. Calculate average quantity ordered per pizza per day.
WITH daily_pizza_totals AS (
    SELECT od.pizza_id, o.date, SUM(od.quantity) AS total_quantity_per_day
    FROM order_details AS od
    JOIN orders as o ON od.order_id = o.order_id
    GROUP BY od.pizza_id, o.date
)
SELECT p.pizza_id,pt.name AS pizza_name, ROUND(AVG(dpt.total_quantity_per_day),2)
  AS avg_quantity_per_day
FROM daily_pizza_totals AS dpt
JOIN pizzas AS p ON dpt.pizza_id = p.pizza_id
JOIN pizza_types AS pt ON p.pizza_type_id = pt.pizza_type_id
GROUP BY p.pizza_id,pt.name ORDER BY avg_quantity_per_day DESC;

-- 28. Determine the category with the highest revenue.
WITH category_revenue AS (
    SELECT pt.category, SUM(od.quantity*p.price) AS total_revenue
    FROM order_details AS od
    JOIN pizzas AS p ON od.pizza_id=p.pizza_id
    JOIN pizza_types AS pt ON p.pizza_type_id=pt.pizza_type_id
    GROUP BY pt.category
),
max_revenue AS (
    SELECT MAX(total_revenue) AS max_rev
    FROM category_revenue
)
SELECT cr.category, cr.total_revenue
FROM category_revenue AS cr
```

```sql
    JOIN max_revenue AS mr ON cr.total_revenue =  mr.max_rev;

-- 29. Show the hourly distribution of pizza orders.
SELECT 
    HOUR(o.time) AS order_hour,
    SUM(od.quantity) AS total_pizzas_ordered
FROM
    orders AS o
        JOIN
    order_details AS od ON o.order_id = od.order_id
GROUP BY order_hour
ORDER BY order_hour;

-- 30. Find the least ordered pizza type and its total sales.
WITH type_sales AS (
SELECT pt.name AS pizza_type, SUM(od.quantity) AS total_quantity
FROM order_details AS od
JOIN pizzas AS p ON od.pizza_id=p.pizza_id
JOIN pizza_types AS pt ON p.pizza_type_id=pt.pizza_type_id
GROUP BY pt.name
),
min_type AS (
SELECT MIN(total_quantity) AS min_qty
FROM type_sales
)
SELECT ts.pizza_type,ts.total_quantity
FROM type_sales AS ts
JOIN min_type AS mt on ts.total_quantity = mt.min_qty;
```