1. Retrieve the customer names who have placed orders (order number + order date). Display the customer name, order number, and order date.

```
SELECT
   c.customerName,
   o.orderNumber,
   o.orderDate
FROM
   customers c
INNER JOIN
   orders o ON c.customerNumber = o.customerNumber;
```

2. Retrieve the customer details and order dates for all customers, including those who have not placed any orders. Display the customer name, phone number, and order date if available. (LEFT JOIN)

```
SELECT
   c.customerName,
   c.phone,
   o.orderDate
FROM
   customers c
LEFT JOIN
   orders o ON c.customerNumber = o.customerNumber;
```

3. Retrieve the customer details and order dates for all customers, including those who have not placed any orders. Display the customer name, phone number, and order date if available. (Right JOIN)

```
SELECT
   c.customerName,
   c.phone,
   o.orderDate
FROM
```

```
    orders o
RIGHT JOIN
    customers c ON c.customerNumber = o.customerNumber
```

4. Retrieve the details of products, their order quantities, and the customer names for orders containing those products. Display the product name, order quantity, and customer name.

```
SELECT
    p.productName,
    od.quantityOrdered,
    c.customerName
FROM
    products p
JOIN
    orderdetails od ON p.productCode = od.productCode
JOIN
    orders o ON od.orderNumber = o.orderNumber
JOIN
    customers c ON o.customerNumber = c.customerNumber;
```

5. Retrieve the details of all products along with the total quantity ordered for each product, even if there are no orders for that product. Display the product name, product code, and total quantity ordered. (LEFT JOIN)

```
SELECT
    p.productName,
    p.productCode,
    SUM(od.quantityOrdered) AS totalQuantityOrdered
FROM
    products p
LEFT JOIN
    orderdetails od ON p.productCode = od.productCode
```

```
GROUP BY
   p.productName, p.productCode;
```

6. Retrieve the details of all products along with the total quantity ordered for each product, even if there are no orders for that product. Display the product name, product code, and total quantity ordered. (RIGHT JOIN)

```
SELECT
   p.productName,
   p.productCode,
   SUM(od.quantityOrdered) AS totalQuantityOrdered
FROM
   orderdetails od
RIGHT JOIN
   products p ON p.productCode = od.productCode
GROUP BY
   p.productName, p.productCode;
```

7. Retrieve the details of all customers and the total payments they have made, even if a customer has not made any payments. Display the customer name, customer number, and total payments.(LEFT JOIN)

```
SELECT
   c.customerName,
   c.customerNumber,
   SUM(p.amount) AS totalPayments
FROM
   customers c
LEFT JOIN
   payments p ON c.customerNumber = p.customerNumber
GROUP BY
   c.customerName, c.customerNumber;
```

8. Retrieve the details of all customers and the total payments they have made, even if a customer has not made any payments. Display the customer name, customer number, and total payments. (Right JOIN)

```
SELECT
   c.customerName,
   c.customerNumber,
   SUM(p.amount) AS totalPayments
FROM
   payments p
RIGHT JOIN
   customers c ON c.customerNumber = p.customerNumber
GROUP BY
   c.customerName, c.customerNumber;
```

9. Retrieve the distinct product lines present in the "products" table. To display each unique product line only once. In two Different Ways/Queries

```
SELECT DISTINCT(productLine) FROM `products`;

=============================================================

SELECT productLine
FROM products
GROUP BY productLine;
```

10. Retrieve the total quantity ordered for each product. Display the productcode and the total quantity ordered, ordered by the total quantity in descending order.

```
SELECT
   productCode,
   SUM(quantityOrdered) AS totalQuantityOrdered
FROM
   orderdetails
GROUP BY
```

```
    productCode
ORDER BY
    totalQuantityOrdered DESC;
```

11. Display the number of orders in each status presented in the “orders” table

```
SELECT count(*), year(orderDate)
 FROM `orders`
group BY year(orderDate)
```

12. Retrieve the count of orders for each year from the "orders" table. Group the results by the year of the order date, and display the count along with the corresponding year. Sort the results in descending order based on the count of orders.

```
SELECT count(*), year(orderDate) as Year_Ordered
 FROM `orders`
group BY Year_Ordered
order by Year_Ordered desc
```

13. Retrieve the total value of each product in stock by multiplying the quantity in stock with the corresponding price from the "products" table. Display the product name along with the calculated total value for each product. Group the results by the product name.

```
SELECT p.productName,
  SUM(p.quantityInStock * od.priceEach) AS totalValue
FROM products p
JOIN orderdetails od ON p.productCode = od.productCode
GROUP BY p.productName;
```

14. Retrieve the product names and their total values for products currently in stock. Calculate the total value for each product by multiplying the quantity in stock with the corresponding price from the "products" table. Display only those products with a total value greater than $10,000,000. Group the results by the product name.

```
SELECT p.productName,
  SUM(p.quantityInStock * od.priceEach) AS totalValue
FROM products p
JOIN orderdetails od ON p.productCode = od.productCode
GROUP BY p.productName
HAVING totalValue > 10000000;
```

15. Retrieve the total sales amount for each product line and each year. Calculate the total sales by multiplying the quantity ordered by the price each. Display the product line, the year of the order date, and the total sales for each combination. Sort the results by product line in ascending order and then by year in descending order.

```
SELECT productLine, YEAR(orderDate) AS orderYear,
    SUM(quantityOrdered * priceEach) AS totalSales
FROM orderdetails od
JOIN products p ON od.productCode = p.productCode
JOIN orders o ON od.orderNumber = o.orderNumber
GROUP BY productLine, orderYear
ORDER BY productLine ASC, orderYear DESC;
```

16. Retrieve the total quantity of products sold and the total sales amount for each country. Display the country, the total quantity of products sold, and the total sales amount ((quantityOrdered * priceEach)) . Include only countries where the total quantity sold is greater than 2000. Sort the results by the total sales amount in descending order.

```
SELECT
    c.country,
   SUM(od.quantityOrdered) AS totalQuantitySold,
   SUM(od.quantityOrdered * od.priceEach) AS totalSalesAmount
FROM customers c
JOIN orders o ON c.customerNumber = o.customerNumber
JOIN orderdetails od ON o.orderNumber = od.orderNumber
GROUP BY c.country
HAVING totalquantitysold > 2000
ORDER BY totalsalesamount DESC;
```

17. Retrieve the number of products in each product lines their text descriptions. Display the product line, the number of products in each line, and the text description. Include only those product lines where the count of products is greater than 20.

```
SELECT
   p.productLine,
   COUNT(*) AS productCount,
   pl.textDescription
FROM
   products p
JOIN
   productlines pl USING(productLine)
GROUP BY
   p.productLine, pl.textDescription
HAVING
   productCount > 20;
```

18. Retrieve the product details for the product with the lowest stock quantity. Display the product name, product code, and the current stock quantity.

```
SELECT
   productName,
   productCode,
   quantityInStock
```

```
FROM
   products
WHERE
   quantityInStock = (SELECT MIN(quantityInStock) FROM products);

==========================================================

SELECT
   productName,
   productCode,
   quantityInStock
FROM
   products
ORDER BY quantityInStock ASC
limit 1;
```

19. Retrieve the details of the product with the highest profit margin. Display the product name, product code, and profit margin (calculated as (buyPrice - MSRP) / MSRP).

```
SELECT
   productName,
   productCode,
   (buyPrice - MSRP) / MSRP AS profitMargin
FROM
   products
HAVING
   profitMargin = (SELECT MAX((buyPrice - MSRP) / MSRP) FROM products);
```

20. Retrieve the details of the customer who has the largest credit limit. Display the customer name, customer number, and the credit limit.

```
SELECT
   customerName,
   customerNumber,
   creditLimit
FROM
```

```
   customers
WHERE
   customerNumber = (SELECT customerNumber FROM customers ORDER BY
creditLimit DESC LIMIT 1);



=================================================

SELECT customerNumber, customerName , creditLimit
FROM `customers`
order BY creditLimit DESC
LIMIT 1;
```

21. Retrieve the customer names, order numbers, and product codes for products ordered by customers, but only for orders where the total order price (quantity ordered * price each) is greater than $3,000. Show the results in a single table.

```
SELECT c.customerName, o.orderNumber, od.productCode , od.quantityOrdered *
p.buyPrice
FROM customers c
JOIN orders o ON c.customerNumber = o.customerNumber
JOIN orderdetails od ON o.orderNumber = od.orderNumber
JOIN products p ON od.productCode = p.productCode
WHERE (od.quantityOrdered * p.buyPrice) > 3000;

============================================================

SELECT c.customerName, o.orderNumber, od.productCode, (od.quantityOrdered *
p.buyPrice) AS Total_order_price
FROM customers c
JOIN orders o ON c.customerNumber = o.customerNumber
JOIN orderdetails od ON o.orderNumber = od.orderNumber
JOIN products p ON od.productCode = p.productCode
HAVING Total_order_price > 3000;
```