SQL Interview Questions (Intermediate to Advance)

1. Find the top 3 customers with the highest order value.

```
SELECT
  CustomerName,
  SUM(OrderValue) AS 'Total Order Value'
FROM
  Orders
GROUP BY
  CustomerName
ORDER BY
  TotalOrderValue DESC
LIMIT
  3;
```

2. Find the average order value for each product category.

```
SELECT
  ProductCategory,
  AVG(OrderValue) AS 'Average Order Value'
FROM
  Orders
GROUP BY
  ProductCategory;
```

3. Find the number of orders placed by each customer in the last 30 days.

```
SELECT
  CustomerName,
  COUNT(*) AS 'Number of Orders'
FROM
  Orders
WHERE
  OrderDate > CURRENT_DATE - INTERVAL 30 DAY
GROUP BY
  CustomerName;
```

4. Find the customers who have placed orders for more than 5 different products.

```
SELECT
  CustomerName,
  COUNT(DISTINCT ProductID) AS 'Number of Products Ordered'
FROM
  Orders
GROUP BY
  CustomerName
HAVING
  COUNT(DISTINCT ProductID) > 5;
```

5. Find the products that have been ordered more than 100 times.

```
SELECT
  ProductID,
  COUNT(*) AS 'Number of Orders'
FROM
  Orders
GROUP BY
  ProductID
HAVING
  COUNT(*) > 100;
```

6. Find the products that have been ordered by more than 100 different customers.

```
SELECT
  ProductID,
  COUNT(DISTINCT CustomerName) AS 'Number of Customers'
FROM
  Orders
GROUP BY
  ProductID
HAVING
  COUNT(DISTINCT CustomerName) > 100;
```

7. Find the customers who have placed orders for more than $10,000 worth of products.

```
SELECT
  CustomerName,
```

```
  SUM(OrderValue) AS 'Total Order Value'
FROM
  Orders
GROUP BY
  CustomerName
HAVING
  SUM(OrderValue) > 10000;
```

8. Find the products that have been ordered more than once by the same customer.

```
SELECT
  ProductID,
  COUNT(*) AS 'Number of Orders'
FROM
  Orders
GROUP BY
  ProductID, CustomerName
HAVING
  COUNT(*) > 1;
```

9. Find the customers who have placed orders for products in more than 2 different product categories.

```
SELECT
  CustomerName,
  COUNT(DISTINCT ProductCategory) AS 'Number of Product
Categories'
FROM
  Orders
GROUP BY
  CustomerName
HAVING
  COUNT(DISTINCT ProductCategory) > 2;
```

10. Find the customers who have placed orders for products in all 5 product categories.

```
SELECT
  CustomerName
```

```
FROM
  Orders
GROUP BY
  CustomerName
HAVING
  COUNT(DISTINCT ProductCategory) = 5;
```

11. Find the top 3 products that have been ordered the most in the past 30 days.

```
SELECT
  ProductID,
  COUNT(*) AS 'Number of Orders'
FROM
  Orders
WHERE
  OrderDate > CURRENT_DATE - INTERVAL 30 DAY
GROUP BY
  ProductID
ORDER BY
  Number of Orders DESC
LIMIT
  3;
```

12. Find the customers who have placed orders for more than $10,000 worth of products in the past year.

```
SELECT
  CustomerName,
  SUM(OrderValue) AS 'Total Order Value'
FROM
  Orders
WHERE
  OrderDate > CURRENT_DATE - INTERVAL 1 YEAR
GROUP BY
  CustomerName
HAVING
  SUM(OrderValue) > 10000;
```

SQL Interview Questions (Intermediate to Advance)

13. Find the products that have been ordered by more than 100 different customers in the past 6 months.

```
SELECT
  ProductID,
  COUNT(DISTINCT CustomerName) AS 'Number of Customers'
FROM
  Orders
WHERE
  OrderDate > CURRENT_DATE - INTERVAL 6 MONTH
GROUP BY
  ProductID
HAVING
  COUNT(DISTINCT CustomerName) > 100;
```

14. Find the customers who have placed orders for products in all 5 product categories.

```
SELECT
  CustomerName
FROM
  Orders
GROUP BY
  CustomerName
HAVING
  COUNT(DISTINCT ProductCategory) = 5;
```

15. Find the products that have been ordered more than once by the same customer, but not more than 3 times.

```
SELECT
  ProductID,
  COUNT(*) AS 'Number of Orders'
FROM
  Orders
GROUP BY
  ProductID, CustomerName
HAVING
  COUNT(*) > 1 AND COUNT(*) <= 3;
```

SQL Interview Questions (Intermediate to Advance)

16. Find the products that have been ordered more than once by the same customer, and the average number of times they have been ordered by each customer.

```
SELECT
  ProductID,
  COUNT(*) AS 'Number of Orders',
  AVG(COUNT(*)) OVER (PARTITION BY CustomerName) AS 'Average
Number of Orders'
FROM
  Orders
GROUP BY
  ProductID, CustomerName;
```

17. Find the products that have been ordered more than once by the same customer, and the customer who has ordered each product the most number of times.

```
SELECT
  ProductID,
  CustomerName,
  COUNT(*) AS 'Number of Orders',
  MAX(COUNT(*)) OVER (PARTITION BY ProductID) AS 'Max Number
of Orders'
FROM
  Orders
GROUP BY
  ProductID, CustomerName;
```

18. Find the products that have been ordered more than once by the same customer, and the customer who has ordered the most different products.

```
SELECT
  ProductID,
  CustomerName,
  COUNT(DISTINCT ProductID) AS 'Number of Different Products
Ordered',
```

```
  MAX(COUNT(DISTINCT ProductID)) OVER (PARTITION BY
CustomerName) AS 'Max Number of Different Products Ordered'
FROM
  Orders
GROUP BY
  ProductID, CustomerName;
```

19. Find the products that have been ordered more than once by the same customer, and the customer who has spent the most money on each product.

```
SELECT
  ProductID,
  CustomerName,
  SUM(OrderValue) AS 'Total Order Value',
  MAX(SUM(OrderValue)) OVER (PARTITION BY ProductID) AS 'Max
Order Value'
FROM
  Orders
GROUP BY
  ProductID, CustomerName;
```

20. Find the products that have been ordered more than once by the same customer, and the customer who has spent the most money in total on all products.

```
SELECT
  ProductID,
  CustomerName,
  SUM(OrderValue) AS 'Total Order Value',
  MAX(SUM(OrderValue)) OVER (PARTITION BY CustomerName) AS
'Max Order Value'
FROM
  Orders
GROUP BY
  ProductID, CustomerName;
```

21. Write a query to join the `Customers` and `Orders` tables, and return the customer name, order date, and order value for all orders placed by customer `123456`.

```
SELECT
  c.CustomerName,
  o.OrderDate,
  o.OrderValue
FROM
  Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
WHERE
  c.CustomerID = 123456;
```

22. Write a query to join the `Customers`, `Orders`, and `Products` tables, and return the customer name, order date, product name, and order value for all orders placed by customer `123456` for product `1234`.

```
SELECT
  c.CustomerName,
  o.OrderDate,
  p.ProductName,
  o.OrderValue
FROM
  Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
INNER JOIN Products p ON o.ProductID = p.ProductID
WHERE
  c.CustomerID = 123456 AND
  o.ProductID = 1234;
```

23. Write a query to join the `Customers`, `Orders`, and `Products` tables, and return the customer name, order date, product name, order value, and the number of days between the order date and today's date for all orders placed by customer `123456` for product `1234`.

```
SELECT
  c.CustomerName,
  o.OrderDate,
  p.ProductName,
  o.OrderValue,
```

```
  CURRENT_DATE - o.OrderDate AS 'Days Since Order'
FROM
  Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
INNER JOIN Products p ON o.ProductID = p.ProductID
WHERE
  c.CustomerID = 123456 AND
  o.ProductID = 1234;
```

24. Write a query to join the `Customers`, `Orders`, and `Products` tables, and return the customer name, order date, product name, order value, and the total number of orders placed by the customer for that product.

```
SELECT
  c.CustomerName,
  o.OrderDate,
  p.ProductName,
  o.OrderValue,
  COUNT(*) OVER (PARTITION BY c.CustomerID, o.ProductID) AS
'Total Orders'
FROM
  Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
INNER JOIN Products p ON o.ProductID = p.ProductID;
```

25. Write a query to join the `Customers`, `Orders`, and `Products` tables, and return the customer name, order date, product name, order value, and the average order value for all orders placed for that product.

```
SELECT
  c.CustomerName,
  o.OrderDate,
  p.ProductName,
  o.OrderValue,
  AVG(o.OrderValue) OVER (PARTITION BY o.ProductID) AS
'Average Order Value'
FROM
  Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
```

```
INNER JOIN Products p ON o.ProductID = p.ProductID;
```

26. Write a query to join the `Customers`, `Orders`, and `Products` tables, and return the customer name, order date, product name, order value, and the total order value for all orders placed for that product by customers who live in the state of California.

```
SELECT
  c.CustomerName,
  o.OrderDate,
  p.ProductName,
  o.OrderValue,
  SUM(o.OrderValue) OVER (PARTITION BY o.ProductID, c.State)
AS 'Total Order Value'
FROM
  Customers c
INNER JOIN Orders o ON c.CustomerID = o.CustomerID
INNER JOIN Products p ON o.ProductID = p.ProductID
WHERE
  c.State = 'CA';
```