iNeuron

SQL Query

Interview Questions

1. Differentiate between VARCHAR and VARCHAR2 data types.
Both VARCHAR and VARCHAR2 are used to store variable-length character strings. VARCHAR is a standard SQL data type which works across different relational database systems. Whereas, VARCHAR2 is specific to Oracle.

VARCHAR 2 has several advantages. It is more storage efficient and, unlike VARCHAR, it does not store trailing spaces at the end of a string so avoids potential unexpected results when comparing strings. However, VARCHAR2 might not be supported non-Oracle database systems.

2. How would you explain database roles and privileges in Oracle SQL security? How do you grant and revoke privileges to users and roles in Oracle?
Database roles are named groups of related privileges. They allow for assigning multiple privileges to a role and granting or revoking the role to users, simplifying security management. The GRANT statement is used to grant, and the REVOKE statement is used to revoke privileges.

3. Write an Oracle SQL query to find the average salary of employees within each department.

```
SELECT department_id, AVG(salary) AS average_salary

FROM employees

GROUP BY department_id;
```

4. Write an Oracle SQL query to find employees who earn more than their managers.

```
SELECT emp.*

FROM Employee emp

INNER JOIN Employee mgr ON emp.manager_id = mgr.employee_id

WHERE emp.salary > mgr.salary;
```

5. How would you update the status column of the orders table to set all orders with a total amount greater than 1,000 to High Value?

```
UPDATE orders

SET status = 'High Value'

WHERE total_amount > 1000;
```

6. Write an Oracle SQL query to get the date and time of the last 10 logins for a specific user.

```
SELECT login_time

FROM UserLogins

WHERE user_id = 'specific_user_id'

ORDER BY login_time DESC

FETCH FIRST 10 ROWS ONLY;
```

7. Retrieve the top five highest-rated products based on customer reviews from the product_reviews table.

```
SELECT product_id, product_name, AVG(review_rating) AS
average_rating

FROM product_reviews

GROUP BY product_id, product_name

ORDER BY average_rating DESC

FETCH FIRST 5 ROWS ONLY;
```

8. Calculate the total revenue generated by each customer in the last three months.

```
SELECT customer_id, SUM(revenue) AS total_revenue

FROM sales

WHERE transaction_date >= TRUNC(SYSDATE) - INTERVAL '3' MONTH

GROUP BY customer_id;
```

9. Calculate the percentage of total sales each product contributes to the overall revenue.

```
SELECT product_id, SUM(total_amount) / (SELECT
SUM(total_amount) FROM sales) * 100 AS percentage_contribution

FROM sales

GROUP BY product_id;
```

10. Write an Oracle SQL query to find the names of employees not assigned to any project.

```
SELECT employee_name

FROM employees

WHERE employee_id NOT IN (SELECT DISTINCT employee_id FROM projects);
```

11. Write an Oracle SQL query to find the five most common names in the Employee table.

```
SELECT name, COUNT(*) AS name_count

FROM Employee

GROUP BY name

ORDER BY name_count DESC

FETCH FIRST 5 ROWS ONLY;
```

12. Write an Oracle SQL query to ensure only users with the manager role can insert rows into the performance_reviews table.

```
CREATE OR REPLACE TRIGGER enforce_manager_insert

BEFORE INSERT ON performance_reviews

FOR EACH ROW

DECLARE
```

```
BEGIN

  IF NOT (IS_ROLE_ENABLED('manager')) THEN

    RAISE_APPLICATION_ERROR(-20001, 'Only users with the
"manager" role can insert into this table.');

  END IF;

END;

/
```

13. You have an Employees table with columns for employee names and their respective managers. How will you find the longest chain of reporting for each employee?

```
WITH RECURSIVE ReportingChain AS (

    SELECT employee_id, manager_id, employee_name, 1 AS
chain_length

    FROM Employees

    WHERE manager_id IS NOT NULL

    UNION ALL

    SELECT e.employee_id, e.manager_id, e.employee_name,
rc.chain_length + 1
```

```
    FROM Employees e

    INNER JOIN ReportingChain rc ON e.manager_id = rc.employee_id

)

SELECT employee_id, employee_name, MAX(chain_length) AS
longest_chain

FROM ReportingChain

GROUP BY employee_id, employee_name;
```

14. Imagine that you have a students table with the columns student_id, student_name, and birthdate. Write an Oracle SQL query to find each student's age (in years) as of today.

```
SELECT student_id, student_name,

      EXTRACT(YEAR FROM SYSDATE) - EXTRACT(YEAR FROM birthdate)
AS age

FROM students;
```

15. In a database containing information about books and authors, write an SQL query to identify the author with the most published books.

```
SELECT author_id, author_name, COUNT(book_id) AS total_books

FROM Authors
```

JOIN Books ON Authors.author_id = Books.author_id

GROUP BY author_id, author_name

ORDER BY total_books DESC

FETCH FIRST 1 ROWS ONLY;

16. Imagine you have an Inventory table with product_id and quantity columns. Write an Oracle SQL query to find the products that have experienced an increase in quantity compared to the previous month.
SELECT product_id

FROM (

SELECT product_id, quantity, LAG(quantity) OVER (ORDER BY month) AS prev_quantity

FROM Inventory

)

WHERE quantity > prev_quantity;

17. Case Study: Sales Analysis System. The dataset contains information about sales transactions in a company. The "sales" table includes the following columns:
transaction_id: Unique identifier for each transaction.

customer_id: Unique identifier for each customer.

product_id: Unique identifier for each product sold.

transaction_date: The date when the transaction occurred.

quantity: The quantity of the product sold in the transaction.

unit_price: The price of one unit of the product.

You're tasked with finding the top 5 customers who made the highest total purchase amount in the last quarter (last three months) and displaying their names and total purchase amounts. Write an Oracle SQL query to retrieve this information.

```
WITH LastQuarterSales AS (

   SELECT customer_id, SUM(quantity * unit_price) AS
total_purchase_amount

   FROM sales

   WHERE transaction_date >= TRUNC(SYSDATE) - INTERVAL '3'
MONTH

   GROUP BY customer_id

)

SELECT c.customer_id, c.customer_name,
lqs.total_purchase_amount
```

```
FROM LastQuarterSales lqs

JOIN customers c ON lqs.customer_id = c.customer_id

ORDER BY lqs.total_purchase_amount DESC

FETCH FIRST 5 ROWS ONLY;
```

18. Case Study: Employee Performance Evaluation System.
The dataset contains information about employees' performance evaluations in a company. The "employees" table includes the following columns:

employee_id: Unique identifier for each employee.

employee_name: The name of the employee.

department: The department to which the employee belongs (e.g., HR, Finance, Sales).

rating: The employee's performance rating on a scale of 1 to 5 (5 being the highest).

years_of_experience: The number of years of experience of the employee.

salary: The salary of the employee.

manager_id: The ID of the employee's manager.

19. Imagine you're an HR manager and want to get an overview of the average performance rating for each department. Write an Oracle SQL query to retrieve the department and the average performance rating for each department.

```
SELECT department, AVG(rating) AS avg_rating

FROM employees

GROUP BY department;
```

20. Say you're preparing a report for the management to identify employees who have shown consistently high performance. Write an Oracle SQL query to retrieve the names and performance ratings of employees with a rating of 5 in all their performance evaluations.

```
SELECT employee_name

FROM employees

WHERE rating = 5

GROUP BY employee_name

HAVING COUNT(*) = (SELECT COUNT(*) FROM employees);
```