***Oracle SQL Assignment Questions***

1. Basic SELECT Query

- Write an SQL query to retrieve all columns from the `employees` table.

Sol: SELECT \* FROM employees

2. Filtering Data

- Write an SQL query to find all employees who are working in the "Sales" department.

Sol: SELECT \* FROM employees WHERE department\_name = 'Sales';

### 3. \*\*Sorting Data\*\*

- Write an SQL query to get the names and salaries of employees in the "Marketing" department, sorted by their salaries in descending order.

Sol: SELECT e.employee\_name, e.salary FROM employees e JOIN departments d ON e.department\_id = d.department\_id WHERE d.department\_name = 'Marketing' ORDER BY e.salary DESC;

### 4. \*\*Using Aggregate Functions\*\*

- Write an SQL query to calculate the average salary of employees in the "HR" department.

Sol: SELECT AVG(salary) AS average\_salary FROM employees WHERE department\_id = ( SELECT department\_id FROM departments WHERE department\_name = 'HR' );

### 5. \*\*Group By Clause\*\*

- Write an SQL query to find the total number of employees in each department.

Sol: SELECT d.department\_name, COUNT(e.employee\_id) AS total\_employees FROM employees e JOIN departments d ON e.department\_id = d.department\_id GROUP BY d.department\_name;

### 6. \*\*Using DISTINCT\*\*

- Write an SQL query to list all unique job titles from the `employees` table.

Sol: SELECT DISTINCT job\_title FROM employees;

### 7. \*\*Using LIKE Operator\*\*

- Write an SQL query to retrieve all employees whose names start with the letter "J".

SOL: SELECT employee\_name FROM employees WHERE employee\_name LIKE 'J%';

### 8. \*\*Using AND/OR Conditions\*\*

- Write an SQL query to find employees who are either in the "IT" department or have a salary greater than $50,000.

Sol: SELECT \*

FROM employees

WHERE department\_id = (

SELECT department\_id

FROM departments

WHERE department\_name = 'IT'

) OR salary > 50000;

### 9. \*\*Joining Tables (Inner Join)\*\*

- Write an SQL query to display employee names along with their department names by joining the `employees` and `departments` tables.

Sol: SELECT e.employee\_name, d.department\_name

FROM employees e

JOIN departments d ON e.department\_id = d.department\_id;

### 10. \*\*Joining Tables (Left Join)\*\*

- Write an SQL query to display all employees and their department names, including those employees who are not assigned to any department.

Sol: SELECT e.employee\_name, d.department\_name

FROM employees e

LEFT JOIN departments d ON e.department\_id = d.department\_id;

### 11. \*\*Subqueries\*\*

- Write an SQL query to find employees whose salary is greater than the average salary in the `employees` table.

Sol: SELECT employee\_name, salary

FROM employees

WHERE salary > (SELECT AVG(salary) FROM employees);

### 12. \*\*Using IN Operator\*\*

- Write an SQL query to list all employees who belong to the departments "Sales", "Marketing", or "HR".

Sol: SELECT e.employee\_name

FROM employees e

WHERE e.department\_id IN (

SELECT department\_id

FROM departments

WHERE department\_name IN ('Sales', 'Marketing', 'HR')

);

### 13. \*\*Using BETWEEN Operator\*\*

- Write an SQL query to find employees whose salaries are between $40,000 and $60,000.

Sol: SELECT employee\_name, salary

FROM employees

WHERE salary BETWEEN 40000 AND 60000;

### 14. \*\*Using EXISTS\*\*

- Write an SQL query to find departments that have at least one employee with a salary greater than $70,000.

Sol: SELECT department\_name

FROM departments d

WHERE EXISTS (

SELECT 1

FROM employees e

WHERE e.department\_id = d.department\_id AND e.salary > 70000

);

### 15. \*\*Date Functions\*\*

- Write an SQL query to find all employees who joined after January 1, 2020.

Sol: SELECT employee\_name, hire\_date

FROM employees

WHERE hire\_date > TO\_DATE('2020-01-01', 'YYYY-MM-DD');

### 16. \*\*Updating Data\*\*

- Write an SQL query to increase the salary of all employees in the "IT" department by 10%.

Sol: UPDATE employees

SET salary = salary \* 1.10

WHERE department\_id = (

SELECT department\_id

FROM departments

WHERE department\_name = 'IT'

);

### 17. \*\*Deleting Data\*\*

- Write an SQL query to delete all employees who are no longer with the company.

Sol: DELETE FROM employees

WHERE status = 'Inactive'; -- Assumes a 'status' column exists

### 18. \*\*Creating a Table\*\*

- Write an SQL query to create a table called `customers` with columns `customer\_id`, `first\_name`, `last\_name`, `email`, and `phone\_number`.

Sol: CREATE TABLE customers (

customer\_id NUMBER PRIMARY KEY,

first\_name VARCHAR2(50),

last\_name VARCHAR2(50),

email VARCHAR2(100),

phone\_number VARCHAR2(15)

);

### 19. \*\*Modifying a Table (ALTER)\*\*

- Write an SQL query to add a new column `hire\_date` to the `employees` table.

Sol: ALTER TABLE employees

ADD hire\_date DATE;

### 20. \*\*Dropping a Table\*\*

- Write an SQL query to drop the `temporary\_employees` table if it exists.

Sol: BEGIN

EXECUTE IMMEDIATE 'DROP TABLE temporary\_employees';

EXCEPTION

WHEN OTHERS THEN

IF SQLCODE != -942 THEN

RAISE;

END IF;

END;