Aggregate Functions and Grouping

Concept-Based

Q1. What are aggregate functions?
A1. Functions that operate on a set of values and return a single value: COUNT, SUM, AVG, MIN, MAX.

Q2. What is the use of GROUP BY?
A2. It groups rows that have the same values into summary rows.

Q3. Can we use WHERE with GROUP BY?
A3. Yes, WHERE filters rows before grouping.

Q4. What is the difference between COUNT(*) and COUNT(column_name)?
A4. COUNT(*) counts all rows, COUNT(column) ignores NULLs.

Q5. What does HAVING clause do?
A5. Filters groups created by GROUP BY.

Code-Based

Q6. Count number of employees.

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

Q7. Get average salary department-wise.

```
SELECT department, AVG(salary) FROM employees GROUP BY department;
```

Q8. Get max and min salaries.

```
SELECT MAX(salary), MIN(salary) FROM employees;
```

Q9. Find departments with more than 5 employees.

```
SELECT department, COUNT(*) FROM employees GROUP BY department HAVING C
OUNT(*) > 5;
```

Q10. Get total salary of employees in each job role.

```
SELECT job_title, SUM(salary) FROM employees GROUP BY job_title;
```

Scenario-Based

Q11. Find departments with average salary greater than 60000.

```
SELECT department, AVG(salary) AS avg_sal FROM employees GROUP BY departmen
t HAVING AVG(salary) > 60000;
```

Scalar Functions

Concept-Based Questions

1. What are scalar functions in MySQL?

   Scalar functions operate on a single value and return a single value. They are often used in SELECT, WHERE, or other clauses.

2. Name some commonly used scalar functions in MySQL.

   Common scalar functions include:
   - UPPER(), LOWER()
   - NOW(), CURDATE()
   - LENGTH(), CHAR_LENGTH()
   - ROUND(), FLOOR(), CEIL()

3. What is the difference between CHAR_LENGTH() and LENGTH() in MySQL?

   CHAR_LENGTH() returns the number of characters in a string, while LENGTH() returns the number of bytes.

4. Can scalar functions be used in WHERE clauses?

   Yes, scalar functions can be used in WHERE clauses to filter results based on computed values.

5. Are scalar functions deterministic in MySQL?

   Some are deterministic (e.g., UPPER()), and others like NOW() are non-deterministic as they return a different result every time.

Code-Based Questions

1. Write a query to convert all customer names to uppercase.

   ```
   SELECT UPPER(CustomerName) FROM Customers;
   ```

2. Write a query to get the current date and time.

   ```
   SELECT NOW();
   ```

3. Write a query to get the length (in bytes) of a product name.

   ```
   SELECT LENGTH(ProductName) FROM Products;
   ```

4. Write a query to round the price of products to the nearest integer.

   ```
   SELECT ROUND(Price) FROM Products;
   ```

5. Write a query to get the first 5 characters of each employee's name.

   ```
   SELECT LEFT(EmployeeName, 5) FROM Employees;
   ```

Expressions in MySQL

Concept-Based Questions

1. What is an expression in MySQL?

   An expression is a combination of one or more values, operators, and SQL functions that evaluate to a single value.

2. What types of expressions are available in MySQL?

   Types include:
   - Arithmetic expressions
   - Comparison expressions
   - Logical expressions
   - String expressions

3. Can expressions be used in SELECT statements?

   Yes, expressions are commonly used in SELECT statements to derive new values or transform data.

4. What is the result of a logical expression in MySQL?

   Logical expressions return either TRUE, FALSE, or NULL.

5. How does operator precedence affect expressions in MySQL?

   Operator precedence determines the order in which operations are evaluated. For example, multiplication is evaluated before addition.

Code-Based Questions

1. Write an expression to calculate the total price as quantity × price.

```
SELECT Quantity * Price AS TotalPrice FROM OrderDetails;
```

2. Write a query to return employees whose salary is greater than 50000.

```
SELECT * FROM Employees WHERE Salary > 50000;
```

3. Write a query that uses a logical expression to find active users.

```
SELECT * FROM Users WHERE Status = 'Active' AND LastLogin IS NOT NULL;
```

4. Write a query to return 'High' if marks > 80, 'Medium' if >50, else 'Low'.

```
SELECT Name,
CASE
  WHEN Marks > 80 THEN 'High'
  WHEN Marks > 50 THEN 'Medium'
  ELSE 'Low'
END AS Grade
FROM Students;
```

5. Write a query to concatenate first and last names of employees.

```
SELECT CONCAT(FirstName, ' ', LastName) AS FullName FROM Employees;
```