**Hexaware Technical Training Program**

**Day 6 - March 18 , 2025**

**MySQL**

**Data Query Language (DQL) in SQL**

DQL is used to retrieve data from a database. The main command in DQL is the **SELECT** statement.

**Basic SELECT Statement**

SELECT \* FROM Students;

Retrieves all columns and rows from the **Students** table.

**Selecting Specific Columns**

SELECT Student\_ID, Name, Age FROM Students;

Retrieves only **Student\_ID**, **Name**, and **Age** from the **Students** table.

**Using WHERE to Filter Data**

SELECT \* FROM Students WHERE Age > 20;

Retrieves only students whose **Age** is greater than **20**.

SELECT \* FROM Students WHERE Course = 'Computer Science';

Retrieves students who are in **Computer Science**.

**Using ORDER BY for Sorting**

SELECT \* FROM Students ORDER BY Age ASC;

Sorts students by **Age** in **ascending order**.

SELECT \* FROM Students ORDER BY Age DESC;

Sorts students by **Age** in **descending order**.

**Using DISTINCT to Remove Duplicates**

SELECT DISTINCT Course FROM Students;

Retrieves unique **Course** names from the **Students** table.

**Using LIMIT to Fetch a Fixed Number of Records**

SELECT \* FROM Students LIMIT 5;

Retrieves **only 5** records from the **Students** table.

**Using LIKE for Pattern Matching**

SELECT \* FROM Students WHERE Name LIKE 'A%';

Retrieves students whose **Name starts with 'A'**.

SELECT \* FROM Students WHERE Name LIKE '%n';

Retrieves students whose **Name ends with 'n'**.

SELECT \* FROM Students WHERE Name LIKE '%sh%';

Retrieves students whose **Name contains 'sh'**.

**Using IN for Multiple Conditions**

SELECT \* FROM Students WHERE Course IN ('AI', 'Data Science');

Retrieves students who are in **AI** or **Data Science**.

**Using BETWEEN for Range Queries**

SELECT \* FROM Students WHERE Age BETWEEN 18 AND 25;

Retrieves students whose **Age is between 18 and 25**.

**Using GROUP BY for Aggregation**

SELECT Course, COUNT(\*) AS Student\_Count FROM Students GROUP BY Course;

Retrieves the **number of students** in each course.

**Using HAVING for Filtering Groups**

SELECT Course, COUNT(\*) AS Student\_Count FROM Students

GROUP BY Course HAVING COUNT(\*) > 5;

Retrieves courses that have **more than 5 students**.

**Using JOIN to Combine Tables**

SELECT Students.Name, Courses.Course\_Name

FROM Students

JOIN Courses ON Students.Course\_ID = Courses.Course\_ID;

Retrieves student names along with their course names.

**Aggregate Functions in SQL**

Aggregate functions perform calculations on a set of values and return a single value. They are commonly used with GROUP BY to summarize data.

**List of Aggregate Functions**

| **Function** | **Description** |
| --- | --- |
| COUNT() | Counts the number of rows |
| SUM() | Returns the total sum of a column |
| AVG() | Returns the average value |
| MAX() | Returns the highest value |
| MIN() | Returns the lowest value |

**COUNT() - Counting Records**

SELECT COUNT(\*) AS Total\_Students FROM Students;

Returns the **total number of students**.

SELECT COUNT(DISTINCT Course) FROM Students;

Returns the **number of unique courses**.

**SUM() - Summing Values**

SELECT SUM(Fee) AS Total\_Fees FROM Students;

Returns the **total sum of all fees**.

SELECT Course, SUM(Fee) FROM Students GROUP BY Course;

Returns the **total fees collected per course**

**AVG() - Average Calculation**

SELECT AVG(Age) AS Average\_Age FROM Students;

Returns the **average age of students**.

SELECT Course, AVG(Fee) FROM Students GROUP BY Course;

Returns the **average fee for each course**.

**MAX() - Maximum Value**

SELECT MAX(Age) AS Oldest\_Student FROM Students;

Returns the **oldest student's age**.

SELECT Course, MAX(Fee) FROM Students GROUP BY Course;

Returns the **highest fee for each course**.

**MIN() - Minimum Value**

SELECT MIN(Age) AS Youngest\_Student FROM Students;

Returns the **youngest student's age**.

SELECT Course, MIN(Fee) FROM Students GROUP BY Course;

Returns the **lowest fee for each course**.

**Using Multiple Aggregate Functions**

SELECT

COUNT(\*) AS Total\_Students,

AVG(Age) AS Average\_Age,

MIN(Age) AS Youngest,

MAX(Age) AS Oldest

FROM Students;

Returns **total students, average age, youngest and oldest student**.

**Using HAVING with Aggregate Functions**

SELECT Course, COUNT(\*) AS Student\_Count

FROM Students

GROUP BY Course

HAVING COUNT(\*) > 5;

Returns only **courses with more than 5 students**.