Tasks Assigned Today

20-02-2025
Suneetha.V

Commands:

In MySQL, **DDL** (Data Definition Language) and **DML** (Data Manipulation Language) are categories of SQL commands used to define and manipulate data in a database, respectively. Here is a breakdown of these commands:

1. DDL Commands (Data Definition Language):

These commands are used to define, modify, and manage the structure of the database and its objects (like tables, indexes, views, etc.).

- **CREATE**: Used to create databases, tables, views, and other database objects.

Example:

```
CREATE DATABASE my_database;
CREATE TABLE students ( student_id INT PRIMARY KEY, name VARCHAR(100), age INT );
```

ALTER: Used to modify the structure of an existing database object, such as adding, deleting, or modifying columns in a table.

- `ALTER TABLE students ADD COLUMN grade VARCHAR(10);`
- `ALTER TABLE students MODIFY COLUMN age INT(3);`

DROP: Used to delete databases, tables, views, or other objects.

```
DROP TABLE students;

DROP DATABASE my_database;
```

TRUNCATE: Removes all rows from a table without deleting the table structure.

- TRUNCATE TABLE students;

Tasks Assigned Today

20-02-2025
Suneetha.V

RENAME: Used to rename a database object, like a table or column.

- RENAME TABLE students TO alumni;

2. DML Commands (Data Manipulation Language):

These commands are used to manipulate or modify the data stored in the database. DML commands deal with data operations like inserting, updating, and deleting records.

- **INSERT**: Used to insert new records into a table.
    -

Example:

INSERT INTO students (student_id, name, age) VALUES (1, 'John Doe', 20);

SELECT: Used to query and retrieve data from one or more tables.

- Example:
  SELECT * FROM students; SELECT name, age FROM students WHERE age > 18;

UPDATE: Used to modify existing records in a table.

UPDATE students SET age = 21 WHERE student_id = 1;

DELETE: Used to delete records from a table.

DELETE FROM students WHERE student_id = 1;

1. WHERE Clause:

The `WHERE` clause is used to filter records based on a specified condition.

- **Syntax**:

```
SELECT column1, column2 FROM table_name WHERE condition;
```

```
example:SELECT * FROM students WHERE age > 18;
```

Tasks Assigned Today

20-02-2025
Suneetha.V

2. ORDER BY Clause:

The ORDER BY clause is used to sort the result set in ascending (ASC) or descending (DESC) order.

- **Syntax**:

```
SELECT column1, column2 FROM table_name ORDER BY column_name
[ASC | DESC];
```

Example:

```
SELECT * FROM students ORDER BY name ASC; SELECT * FROM
students ORDER BY age DESC;
```

- **ASC** (Ascending) is the default if you don’t specify anything.
- **DESC** (Descending) sorts in reverse order.

3. LIMIT Clause:

The LIMIT clause is used to specify the number of records to return.

- **Syntax**:

```
SELECT column1, column2 FROM table_name LIMIT
number_of_records;
```

Example:

```
SELECT * FROM students LIMIT 5;  -- Returns the first 5
records
```

You can also use LIMIT with an offset

```
SELECT * FROM students LIMIT 5 OFFSET 10;  -- Skip the first
10 records and return the next 5
```

Tasks Assigned Today

20-02-2025
Suneetha.V

4. MIN() Function:The MIN() function returns the smallest value in a column.

Syntax:
```
SELECT MIN(column_name) FROM table_name;
```

Example:
```
SELECT MIN(age) FROM students;  -- Returns the minimum age
from the students table
```

5. MAX() Function:

The MAX() function returns the largest value in a column.

Syntax:
```
SELECT MAX(column_name) FROM table_name;
```

Example:

```
SELECT MAX(age) FROM students;  -- Returns the maximum age
from the students table
```

6. SUM() Function:

The SUM() function returns the total sum of a numeric column.

Syntax:
```
SELECT SUM(column_name) FROM table_name;
```

Example:
```
SELECT SUM(age) FROM students;  -- Returns the sum of all ages
in the students table
```

Tasks Assigned Today

20-02-2025
Suneetha.V

7. COUNT() Function:

The COUNT() function returns the number of rows that match a specified condition.

Syntax:

```
SELECT COUNT(column_name) FROM table_name;

SELECT COUNT(*) FROM table_name;  -- Counts all rows,
regardless of column values
```

Example:

```
SELECT COUNT(*) FROM students;  -- Returns the number of
records in the students table

SELECT COUNT(age) FROM students WHERE age > 18;  -- Returns
the count of students older than 18
```

8. AVG() Function:

The AVG() function returns the average value of a numeric column.

Syntax:

```
SELECT AVG(column_name) FROM table_name;
```

Example:

```
SELECT AVG(age) FROM students;  -- Returns the average age
from the students table
```

Tasks Assigned Today

20-02-2025
Suneetha.V

9. BETWEEN Operator:

The BETWEEN operator is used to filter the result set within a range of values. It is inclusive of the boundary values.

Syntax:
```
SELECT column1, column2 FROM table_name WHERE column_name
BETWEEN value1 AND value2;
```

Example:
```
SELECT * FROM students WHERE age BETWEEN 18 AND 25;  --
Returns students aged between 18 and 25.
```