Third Lab

Data Definition Language: DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.

Some of the DDL commands are:

Create: It is used to create the database or its objects (like table, index, function, views, store procedure and triggers).

Creating a database:

Syntax:

```
CREATE DATABASE database_name;
```

Example: CREATE DATABASE kits;

- Show command is used to list all the databases on the server.

Syntax:

```
show databases;
```

==========

Creating the table:

Syntax:

CREATE TABLE [IF NOT EXISTS] TableName (fieldname dataType [optional parameters]) or

```
CREATE TABLE table_name
(
column1 data_type(size),
column2 data_type(size),
column3 data_type(size),
....
);
```

Example:

CREATE TABLE student(sno int,name char(50),address varchar(50));

==========

Describe command: DESCRIBE means to show the information in detail. Since we have tables in MySQL, so we will use the **DESCRIBE command to show the structure of our table**, such as column names, constraints on column names, etc. The **DESC** command is a short form of the DESCRIBE command. Both DESCRIBE and DESC command is equivalent and case sensitive.

Syntax:
DESC | DESCRIBE table_name;

Example:
DESC student;

==========

Alter command: This command is used to modify the structure of the schema or table. It can even used to add, modify or delete the columns in the table.

Adding a column in the table:

This command is used to add columns in the table.
Syntax:

ALTER TABLE table_name ADD [column] new_column_name column_definition [FIRST | AFTER column_name];

Example:
1. ALTER TABLE student add age int(20);
2. ALTER TABLE student add clgname varchar(50) first;
3. ALTER TABLE student add gender char(2) after sno;
4. ALTER TABLE student add column sec char(5) after gender;

==========

Adding multiple columns in the table(ADD):

Syntax:

ALTER TABLE table_name ADD
new_column_name column_definition [FIRST | AFTER column_name],
ADD new_column_name column_definition [FIRST | AFTER column_name],
……
;

Example:

ALTER TABLE student ADD fname char(20),add lname char(20);

==========

Modify column in the table (MODIFY):

The MODIFY command is used to change the column definition of the table.

Syntax:

ALTER TABLE table_name MODIFY column_name column_definition;

Example:

ALTER TABLE student MODIFY clgname char(100);

==========

Modifying multiple columns in the table:

Syntax: ALTER TABLE table_name MODIFY column_name column_definition,
MODIFY column_name column_definition;

Example:

ALTER TABLE student MODIFY gender varchar(100), MODIFY sec varchar(100);

==========

Changing the column name (CHANGE):

Used to change the column name in the table or rename the column name in a table.

Syntax:

ALTER TABLE table_name CHANGE old_name new_name column_definition

Example:

1. ALTER TABLE student CHANGE sno rollno int(50);
2. ALTER TABLE student CHANGE lname lastname char(20);

==========

Renaming the table or relation name(RENAME):

Used to change the name of the relation or table.

Syntax:

ALTER TABLE old_table_name RENAME TO new_table_name; or
RENAME TABLE old_table_name to new_table_name;

Example:

1. ALTER TABLE student RENAME TO studentsection;
2. RENAME TABLE studentsection to student;

==========

Drop column in the table: It is used to drop / delete the column in the table.

Syntax:

ALTER TABLE table_name DROP [COLUMN] column_name;

Example:

First way: ALTER TABLE student DROP COLUMN lastname;

Second way: ALTER TABLE student DROP fname;

==========

Drop multiple columns in the table:

Syntax:

ALTER TABLE table_name DROP columnname1, DROP columnname2….;

Example:

ALTER TABLE student DROP age, DROP address;

==========

Deleting the table or drop the table(DROP):

Syntax:

DROP TABLE table_name;

Example:

DROP TABLE student;

==========

Deleting the database:

Syntax:

DROP DATABASE database_name;

Example:

DROP DATABASE kits;

Truncate: This command is used to remove the content of the table, but keeps the structure of the table. This simply removes all the records from the table. No partial removal of the data is possible here; it also removes all the spaces allocated for the data.

No where clause is used here.

Syntax:

Truncate table table_name;

5. Data Manipulation Language (DML): When we want to insert records into the table or if we want to change / modify some records or delete some records or perform any other action on records in the database, we need to have some media to perform it. DML helps user request it helps to insert, delete, update and retrieve the data from the data.

Some of the DML Commands are:

1. INSERT
2. SELECT (Data Query Language)
3. UPDATE
4. DELETE.

INSERT: Insert statement is used to store or add new records / rows in the table.

Syntax:

INSERT INTO table_name (field1, field2,fieldN) VALUES(value1, value2, ..valueN);

CREATE DATABASE kits;

CREATE TABLE student(rollno int(20),name varchar(50),gender varchar(10), address varchar(50));

Example:

INSERT INTO student (rollno,name,gender,address)
VALUES(101,'karhik','M','hzb');
//inserting data with field names

❖ ***To retrieve / display the data we use select command***

Example: select * from student;

INSERT INTO student VALUES(102,"Raju","M","HNK");

//Inserting data without field's names.

Select * from student;

Inserting multiple records;

INSERT INTO student values(103,'kavitha','f','knr'),(104,'ravi','m','hyd');

Select * from student;

INSERT INTO student values('rakesh','knr');//check

INSERT INTO student(name,address) values('rakesh','knr');//check

UPDATE QUERY:

UPDATE query is a DML statement used to modify the data of the table within the database. In a real-life scenario, records are changed over a period of time. So, we need to make changes in the values of the tables also. To do so, it is required to use the UPDATE query.

The UPDATE statement is used with the **SET** and **WHERE** clauses. The SET clause is used to change the values of the specified column. We can update single or multiple columns at a time.

Syntax:

UPDATE table_name SET column_name1 = new-value1, column_name2= new-value2, [WHERE Clause];

Example:

- UPDATE student SET name=”Ranjith”;//**All the rows are effected**
- UPDATE student SET name=”Srikanth” where rollno=101;

WHERE Clause is used with SELECT, INSERT, UPDATE and DELETE clause to filter the results. It specifies a specific position where you have to do the operation.

Updating multiple columns:

Example:

UPDATE student SET name="kavitha", gender="F" where rollno=104;

DELETE statement:

DELETE statement is used to remove records from the table that is no longer required in the database. **This query deletes a full row from the table**. It also allows us to delete more than one record from the table within a single query, which is beneficial while removing large numbers of records from a table. By using the delete statement, we can also remove data based on conditions.

Syntax:

DELETE FROM table_name WHERE condition;

Example:

DELETE FROM student where rollno=101;

DELETE FROM student; //**deletes all the rows**

LAB WORK-3(PART-2)

DML COMMANDS

1. CREATE TABLE EMPLOYEE (ID INT , NAME VARCHAR(30) , DEPT VARCHAR(20), SALARY INT , DOB DATE , GENDER CHAR(1)) ;
2. DESC EMPLOYEE ;
3. INSERT INTO EMPLOYEE VALUES(1,'PRAVEEN' , 'CSE' , 4000 , '1980-12-12' , 'M') , (2 , 'RAJU ' , 'CSE ' , 5000 , '1982-02-01' , 'M') , (3 , 'RANI' , 'CSE' , 6000 , '1985-12-25' , 'F') , (4 , 'RAJITHA' , 'ECE' , 6500 , '1989-01-24' , 'F') ,(5 , 'KARUN' , 'EEE' , 4500 , '1990-08-14' , 'M') ;
4. SELECT * FROM EMPLOYEE ;
5. SELECT NAME FROM EMPLOYEE ;
6. SELECT NAME,SALARY FROM EMPLOYEE ;
7. SELECT * FROM EMPLOYEE WHERE ID =1 ;
8. SELECT * FROM EMPLOYEE WHERE DEPT = 'CSE' ;
9. SELECT * FROM EMPLOYEE WHERE YEAR(DOB) = '1980' ;

ID	NAME	DEPT	SALARY	DOB	GENDER
1	PRAVEEN	CSE	4000	1980-12-12	M

10. SELECT * FROM EMPLOYEE WHERE SALARY < 5000 ;
11. SELECT * FROM EMPLOYEE WHERE DEPT<> 'CSE' ;
12. SELECT * FROM EMPLOYEE WHERE SALARY BETWEEN 4000 AND 5000 ;

ID	NAME	DEPT	SALARY	DOB	GENDER
1	PRAVEEN	CSE	4000	1980-12-12	M
2	RAJU	CSE	5000	1982-02-01	M
5	KARUN	EEE	4500	1990-08-14	M

13. SELECT * FROM EMPLOYEE WHERE ID BETWEEN 3 AND 4 ;
14. SELECT * FROM EMPLOYEE WHERE DEPT = 'CSE' OR ID =4 ;

ID	NAME	DEPT	SALARY	DOB	GENDER
1	PRAVEEN	CSE	4000	1980-12-12	M
2	RAJU	CSE	5000	1982-02-01	M
3	RANI	CSE	6000	1985-12-25	F
4	RAJITHA	ECE	6500	1989-01-24	F

15. SELECT * FROM EMPLOYEE WHERE ID NOT BETWEEN 1 AND 3 ;
16. SELECT * FROM EMPLOYEE WHERE LENGTH(NAME) = 4 ;

ID	NAME	DEPT	SALARY	DOB	GENDER
3	RANI	CSE	6000	1985-12-25	F

17. SELECT * FROM EMPLOYEE WHERE NAME LIKE '_R%' ;
18.

ID	NAME	DEPT	SALARY	DOB	GENDER
1	PRAVEEN	CSE	4000	1980-12-12	M

NOTE : THE MOST COMMONLY USED OPERATION ON STRING IS PATTERN MATCHING USING THE OPERATOR **LIKE .** WE DESCRIBE PATTERN BY USING TWO SPECIAL CHARACTERS .

a) PERCENT (%) : THE % CHARACTER MATCHES ANY SUBSTRING.
b) UNDERSCORE(_) : THE _ CHARACTER MATCHES ANY CHARACTER .

PATTERNS ARE CASE SENSITIVE

19. SELECT * FROM EMPLOYEE WHERE NAME LIKE 'R%' ;
20. SELECT * FROM EMPLOYEE WHERE NAME LIKE '%N' ;
21. UPDATE EMPLOYEE SET NAME ='MOHAN' WHERE ID =1 ;
22. SELECT * FROM EMPLOYEE ;
23. DELETE FROM EMPLOYEE WHERE ID =2 ;
24. SELECT * FROM EMPLOYEE ;

25. TRUNCATE TABLE EMP ;