



**Vidyavardhini's College of Engineering and Technology**  
**Department of Artificial Intelligence & Data Science**

---

<b>Experiment No.3</b>
Create a database using Data Definition Language(DDL) and apply integrity constraints for the specified system
Date of Performance:
Date of Submission:



## Vidyavardhini's College of Engineering and Technology

### Department of Artificial Intelligence & Data Science

---

**Aim:-** Write a query to create tables for each relation in the relational schema of experiment no.2. Apply drop and alter commands on those tables.

**Objective:-** To learn commands of Data Definition Language(DDL) to create and define databases, and also learn to apply integrity constraints for the specified system.

#### **Theory:**

DDL Commands & Syntax:-

Data Definition Language (DDL) is a subset of SQL and a part of DBMS(Database Management System). DDL consist of Commands to commands like CREATE, ALTER, TRUNCATE and DROP. These commands are used to create or modify the tables in SQL. DDL Commands:

1. Create
2. Alter
3. truncate
4. drop
5. Rename

#### **CREATE:**

This command is used to create a new table in SQL. The user must give information like table name, column names, and their data types.

Syntax –CREATE TABLE table\_name

(  
column\_1 datatype,  
column\_2 datatype,  
column\_3 datatype,  
....



### ALTER :

This command is used to add, delete or change columns in the existing table. The user needs to know the existing table name and can add, delete, or modify tasks easily.

Syntax –

ALTER TABLE table\_name

ADD column\_name datatype;

### TRUNCATE :

This command is used to remove all rows from the table, but the structure of the table still exists.

Syntax –

TRUNCATE TABLE table\_name;

### DROP :

This command is used to remove an existing table along with its structure from the Database. Syntax –

DROP TABLE table\_name;

### RENAME :

It is possible to change name of table with or without data in it using simple RENAME command. We can rename any table object at any point of time.

Syntax –

RENAME TABLE <Table Name> To <New\_Table\_Name>;



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

### Implementation:

### Program:-

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'SCHEMAS' panel with a search filter 'Filter objects'. Below it, a tree view shows the 'farm\_database' schema. The main editor area displays the following SQL queries:

```
1 * CREATE DATABASE IF NOT EXISTS FARM_DATABASE;
2
3 * CREATE TABLE IF NOT EXISTS FMS (
4     F_ID INT PRIMARY KEY,
5     F_NAME VARCHAR(25),
6     L_NAME VARCHAR(25),
7     AGE INT,
8     MOB_NO INT,
9     COUNTRY VARCHAR(50),
10    LOAN INT
11 );
12
13 * CREATE TABLE IF NOT EXISTS FARM(
14     farm_id INT PRIMARY KEY,
15     farm_size int,
16     farm_type varchar(50),
17     farm_location varchar(50)
18 );
19
20 * CREATE TABLE IF NOT EXISTS SUPPLIERS (
21     S_ID INT PRIMARY KEY,
22     F_NAME VARCHAR(25),
23     L_NAME VARCHAR(25),
24     MOB_NO INT,
25     DATES DATE,
26     EMAIL VARCHAR(50),
27     FOREIGN KEY (S_ID)
28         REFERENCES FMS (F_ID)
29 );
```



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

```
Query 1 x
Limit to 1000 rows
29 );
30
31 CREATE TABLE IF NOT EXISTS TOOLS(
32     T_ID INT PRIMARY KEY,
33     T_NAME VARCHAR(25),
34     T_PRICE INT,
35     T_DESC VARCHAR(200)
36 );
37
38 ALTER TABLE FMS ADD LOAN VARCHAR(10);
39 ALTER TABLE FMS DROP LOAN;
40
41 ALTER TABLE FMS
42 RENAME COLUMN COUNTRY TO PLACE;
43
44 TRUNCATE TABLE FMS;
45 TRUNCATE TABLE SUPPLIERS;
46 TRUNCATE TABLE FARM;
47 TRUNCATE TABLE TOOLS;
48
49 DROP TABLE FMS;
50 DROP TABLE SUPPLIERS;
51 DROP TABLE FARM;
52 DROP TABLE TOOLS;
53
54 SELECT * FROM FMS;
55 SELECT * FROM SUPPLIERS;
56 SELECT * FROM FARM;
57 SELECT * FROM TOOLS;
```



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

### Output:

Create Farmers Table:

The screenshot shows the SQL Developer interface with the 'FMS' table selected. The 'Output' pane displays the following actions and messages:

#	Time	Action	Message	Duration / Fetch
7	01:24:35	CREATE TABLE IF NOT EXISTS TOOLS( T_ID INT PRIMARY KEY, T_NAME VARCHAR(25), T_PRICE...	0 row(s) affected	0.032 sec
8	01:24:58	ALTER TABLE FMS ADD LOAN VARCHAR(10)	Error Code: 1060, Duplicate column name 'LOAN'	0.016 sec
9	01:25:17	ALTER TABLE FMS RENAME COLUMN COUNTRY TO PLACE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
10	01:25:26	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
11	01:25:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
12	01:25:53	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec

Create Suppliers Table:

The screenshot shows the SQL Developer interface with the 'SUPPLIERS' table selected. The 'Output' pane displays the following actions and messages:

#	Time	Action	Message	Duration / Fetch
8	01:24:58	ALTER TABLE FMS ADD LOAN VARCHAR(10)	Error Code: 1060, Duplicate column name 'LOAN'	0.016 sec
9	01:25:17	ALTER TABLE FMS RENAME COLUMN COUNTRY TO PLACE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
10	01:25:26	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
11	01:25:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
12	01:25:53	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
13	01:26:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

### Create Farm Table

Result Grid	Filter Rows	Edit	Export/Import	Wrap Cell Contents
farm_id	farm_size	farm_type	farm_location	
INT	INT	INT	TEXT	

  

FARM 5				
Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
9	01:25:17	ALTER TABLE FMS RENAME COLUMN COUNTRY TO PLACE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
10	01:25:26	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
11	01:25:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
12	01:25:53	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
13	01:26:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
14	01:27:03	SELECT * FROM FARM LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec

### Create Tools Table

Result Grid	Filter Rows	Edit	Export/Import	Wrap Cell Contents
T_ID	T_NAME	T_PRICE	T_DESC	
INT	TEXT	INT	TEXT	

  

TOOLS 6				
Output				
Action Output				
#	Time	Action	Message	Duration / Fetch
10	01:25:26	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
11	01:25:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
12	01:25:53	SELECT * FROM FMS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
13	01:26:41	SELECT * FROM SUPPLIERS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
14	01:27:03	SELECT * FROM FARM LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec
15	01:27:27	SELECT * FROM TOOLS LIMIT 0, 1000	0 row(s) returned	0.000 sec / 0.000 sec



# Vidyavardhini's College of Engineering and Technology

## Department of Artificial Intelligence & Data Science

---

### Conclusion:

The completion of the practical exercise involving Data Definition Language (DDL) commands on the Farmers Management System offers valuable insights into the foundational aspects of database management. By applying various DDL commands, including CREATE TABLE, ALTER TABLE, and DROP TABLE, we gained hands-on experience in defining, modifying, and deleting the structure of the database schema.

1. Explain the concept of constraints in DDL. How are constraints used to enforce data integrity?

Ans. Constraints in Data Definition Language (DDL) are rules or conditions applied to the structure of a database schema to enforce data integrity and maintain consistency. They define limitations and restrictions on the data that can be stored in the database, ensuring that only valid, accurate, and meaningful data is allowed. Constraints play a crucial role in database design by preventing data inconsistencies, errors, and anomalies.

Here's how constraints are used to enforce data integrity in a database:

1. **Primary Key Constraint:** A primary key constraint ensures that each row in a table is uniquely identifiable by a primary key attribute or combination of attributes. It prohibits duplicate and null values in the primary key column(s), thereby enforcing entity integrity and ensuring that each record is uniquely identifiable.
2. **Unique Constraint:** The unique constraint ensures that the values in one or more columns of a table are unique across all rows. Unlike primary keys, unique constraints allow null values, but if a column is marked as unique, only one row may contain a null value in that column. This constraint helps maintain data integrity by preventing the insertion of duplicate values in specified columns.
3. **Foreign Key Constraint:** Foreign key constraints establish relationships between tables by enforcing referential integrity. A foreign key in one table references the primary key in another table, ensuring that every foreign key value must match a primary key value.
4. **Check Constraint:** Check constraints define conditions that must be true for every row in a table. They allow you to specify rules that restrict the values allowed in certain columns. For example, a check constraint can ensure that values in a "age" column are greater than zero and less than 120. Check constraints help enforce domain integrity by validating the data against predefined conditions.





## Vidyavardhini's College of Engineering and Technology

### Department of Artificial Intelligence & Data Science

---

integrity and reliability. Constraints help maintain the accuracy, consistency, and validity of data, making them an essential component of database design and management.

2. What is the significance of data types in DDL? Provide examples of commonly used data types in DDL.

Ans. Data types in Data Definition Language (DDL) specify the type of data that can be stored in a column of a table in a relational database. They define the format, size, and range of values that a particular attribute can hold. Data types are essential for ensuring data integrity, optimizing storage, and facilitating efficient data retrieval and manipulation operations.

The significance of data types in DDL can be summarized as follows:

**Data Integrity:** Data types enforce constraints on the values that can be stored in a column, helping to maintain data integrity. By specifying the appropriate data type for each attribute, DDL ensures that only valid and meaningful data is stored in the database.

**Storage Optimization:** Different data types require different amounts of storage space. Choosing the most appropriate data type for each attribute can optimize storage utilization and reduce storage requirements. For example, using a smaller data type like INTEGER instead of BIGINT for a column that only needs to store small integers can conserve storage space.

**Data Retrieval and Manipulation Efficiency:** Data types affect the efficiency of data retrieval and manipulation operations. Using appropriate data types can improve query performance and reduce processing overhead. For example, indexing columns with suitable data types can speed up search and retrieval operations.

**Data Validation:** Data types help validate the format and range of values entered into a column. They prevent the insertion of invalid or incompatible data, thereby enhancing data quality and consistency. For example, using a DATE data type for a column ensures that only valid dates can be stored in that column.

Examples of commonly used data types in DDL include:

**INTEGER:** Stores whole numbers (e.g., 1, 10, -5).

**FLOAT:** Stores floating-point numbers (e.g., 3.14, -0.001).

**CHAR(n):** Stores fixed-length character strings of length n (e.g., 'hello').

**VARCHAR(n):** Stores variable-length character strings of maximum length n (e.g., 'world').

**DATE:** Stores date values in YYYY-MM-DD format (e.g., '2022-04-19').

**TIME:** Stores time values in HH:MM:SS format (e.g., '14:30:00').

**BOOLEAN:** Stores true/false or 1/0 values.

NUMERIC(p, s): Stores fixed-point numbers with precision p and scale s (e.g., NUMERIC(10, 2) for monetary values).