

python :

Table Creation And Data Manipulation Commands

Database In MySQL

The database acts as a central point of administration for the tables in the database. The actual data is stored in the table which provide a structured organization for the data and maintain the integrity of the data. To create tables, either you can create a new database or use an existing database. MySQL provides an empty database by the name test, which you can use to store your tables in. But it is recommended that you create new database before you create a group of some [www.cbsepythonedu.com] tables pertaining to a specific application.

Creating Database

Creating databases is an easier task relatively. In simplest form the create Database command takes [www.cbsepythonedu.com] the following syntax.

Syntax:

Create Database [If Not Exists] < database name > ;

The IF NOT EXISTS clause, if used will first test whether a database by the mentioned name already exists or not. If it does, then create database command is simply ignored, otherwise a [www.cbsepythonedu.com] database with the mentioned name is created.

```
CREATE DATABASE myDB ;
```

```
CREATE DATABASE IF NOT EXISTS myDB ;
```

Opening Databases

To open a database, you simply need to write the statement as per following syntax.

Syntax:

```
USE < databasename > ;
```

Removing Databases

Sometimes, you need to remove a database when you don't need it anymore. But before making this decision, do make sure that you don't need data stored in different tables of the database. This is because, when you drop a database, all its tables also get removed along with the database.

[www.cbsepythonedu.com] To remove a database, you need to issue a command with following syntax.

Syntax:

```
DROP DATABASE < database name > ;
```

Creating tables

Tables are defined with the CREATE TABLES command. When a table is created, its columns are named, data types and sizes are supplied for each column. Each table must have at least one [www.cbsepythonedu.com] column.

The syntax of CREATE TABLE command is :

Syntax:

python :

```
CREATE TABLE <table-name>
    (<column name> <data type>[(<size>)]
    <columnname> <data type> [(size).....]);
```

Data integrity Through Constraints

A constraint, in general, refers to a condition or a check that is applied to a column (field) or set of columns in a table. The constraints applied to maintain data integrity are also known as integrity constraints. Once an integrity constraint is enabled, all data in the table must confirm to the rule that **[www.cbsepythonedu.com]** it specifies. Any subsequent SQL statement, that tries to enter or modify data in the table, is successfully carried out if and only if the data being entered or modified satisfies the integrity constraints. The two basic types of constraints are column constraints and Table constraints. The difference between the two is that column constraints apply only to individual column, whereas table constraints apply to groups of one or more columns. The following is the syntax for the CREATE TABLE command, expanded to include constraints :

Syntax:

```
CREATE TABLE <table name>
    <column name> <data type> [(<size>)] <column constraint>,
    <column name> <data type> (<size>)] <column constraint>,
    <table constraint> (<column name>, [, <column name>...])...);
```

Different Constraints

These constraints ensure database integrity, thus are sometimes called database integrity **[www.cbsepythonedu.com]** constraints. A few of them are :

- | | | |
|-----------------------|-----------------------------|--------------------------|
| (i) Unique constraint | (ii) Primary key constraint | (iii) Default constraint |
| (iv) Check constraint | (v) Foreign constraint | |

Unique constraint

This constraint ensures that no two rows have the same value in the specified column (s).

Primary key constraint

This constraint declares a column as the primary key of the table. This constraint is similar to unique constraint except that only one column (or one group of column) can be applied in this constraint. The primary keys cannot allow NULL values, thus, this constraint must be applied to column **[www.cbsepythonedu.com]** declared as NOT NULL.

Default constraint

A default value can be specified for a column using the DEFAULT clause. When a user does not enter a value for the column (having default value), automatically the defined default value is inserted in the field.

Check constraint

This constraint limits values that can be inserted into a column of a table. This statement compares two columns ROL and QOH, thus, these two columns must be defined before this CHECK **[www.cbsepythonedu.com]** constraint. Check constraint can consist of :

python :

- * A list of constant expressions specified using IN. For example,
descp char (20) CHECK
(descp IN('NUT', 'BOLT', 'SCREW', 'WHRENCH', 'NAIL'))
- * Range of constant expressions specified using BETWEEN. The upper and lower boundary values [www.cbsepythonedu.com] are included in the range. For example, :
prince decimal CHECK (price BETWEEN 253.00 and 7770.00)
- * A pattern specified using LIKE. For example, if we give a date in Char Format, then we may [www.cbsepythonedu.com] check :
ordate char (10) NOT NULL CHECK (ordate LIKE '----/--/----')
- * Multiple conditions using OR, AND etc. For example,
CHECK ((discount = 0.15 and city = 'HISSAR') OR
(discount = 0.13 and city = 'JAIPUR') OR
(discount = 0.15 and city = 'MOHALI'))

Foreign Key Constraint

In a RDBMS, tables reference one another through common fields and to ensure validity of references, referential integrity is enforced. Referential integrity is a system of rules that a DBMS use to ensure that relationships between records in related tables are valid, and that users don't accidentally delete or change related data. Referential integrity is ensured through FOREIGN KEY [www.cbsepythonedu.com] constraint. Whenever two tables are related by a common column (or set of columns), then the related column(s) in the parent table (or primary table) should be either declared a PRIMARY KEY or UNIQUE key and the related column (s) in the child table (or related table) should have [www.cbsepythonedu.com] FOREIGN KEY constraint.

Role of FOREIGN KEY/ REFERENCES constraint

- * It results into the rejection of INSERT (statement used to insert new data) or UPDATE (statement used to modify existing data) if a corresponding value does not currently exist in the primary table.
- * If ON DELETE CASCADE option has been set then upon deleting a tuple in the parent table, all its related record in the child table get deleted. In the absence of this clause, it rejects the deletion [www.cbsepythonedu.com] operation if corresponding records in the child table exist.
- * The foreign key column in the child table must reference a PRIMARY KEY or UNIQUE column in the parent table.
- * Both the related columns (in parent table and in child table) should have the same data type.
- * May reference to the same table in which it is being defined i.e., it may reference the same table name as in the CREATE TABLE command.

Applying Table constraints

When a constraint is to be applied on a group of columns of the table, it is called table constraint. The table constraints appear in the end definition. For instance, if you want combination of icode

python :

[www.cbsepythonedu.com] and desc of table items to be unique, you may write it as follows :

```
CREATE TABLE items
(
    icode          char (5)      NOT NULL,
    descp          char (20)     NOT NULL,
    ROL            integer,
    QOH            integer,
    CHECK          (ROL < QOH),
    UNIQUE         (icode, descp) );
```

The above statement ensures that the combination of icode and desc in each row must be unique. Similarly, if you want to define primary key that contains more than one column, you can use PRIMARY KEY table constraint. For instance, if you want to declare a primary key for the table members as the combination of columns [www.cbsepythonedu.com] firstname and lastname, it can be done as follows:

```
CRATE TABLE members
(
    firstname      char (15)     NOT NULL,
    lastname       char (15)     NOT NULL,
    city           char (20),
    Primary Key (firstname, lastname) );
```

To define foreign key constraint for a group of columns, FOREIGN KEY table constraints should be used. For example, if a group of columns say orderno and itemno, is to be declared foreign key, it [www.cbsepythonedu.com] should be done as follows :

```
CREATE TABLE orders
(
    orderno Number (6,0)        NOT NULL,
    itemno char (5),
    FOREIGN KEY (orderno, itemno) REFERENCES Ordmaster
                                     (OrderNo, ItemNo)
);
```

Assigning Names to Constraints

By default, MySQL assigns a unique name to each constraint defined by you. MySQL names constraints as :

SYS_Cn

where n is an integer that makes the constraint name unique. For instance, SYS_C003217, SYS_C001592 etc. are constraint names generated by MySQL. However, you, yourself, can name [www.cbsepythonedu.com] a constraint created by you. This can be done as per following syntax :

Syntax:

CONSTRAINT <name-of-constraint> <definition-of-constraint>

For example, consider the following two CREATE TABLE statements :

```
CREATE TABLE Items
(
    Itemno char (5) CONSTRAINT p_Itemkey PRIMARY KEY,
```

python :

```
);  
CREATE TABLE Orders  
(  
    Orderno,  
    CONSTRAINT Group_fkey_orders FOREIGN KEY (Orderno, Itemno)  
    REFERENCES OrdMaster (Order #, Item#) );
```

Viewing a table Structure

Once you have created a table, you may want to view its structure. Anytime, if you want to view an already created/existing table's structure, you may use DESC [RIBE] command of MySQL * Plus. The syntax of this command is as [www.cbsepythonedu.com] follows :

Syntax:
DESC[RIBE] <tablename>;

For [www.cbsepythonedu.com] example, if you write
DESC Pet ;
DESCRIBE Pet ;

Creating Table from Existing Table

You can define a table and put data into it without going through the usual data definition process. This can be done by using SELECT statement with CREATE TABLE. The new table stores the result produced by the SELECT statement. The name of the new table must be unique. Following [www.cbsepythonedu.com] query illustrates this :

```
CREATE TABLE orditem AS  
(  
    SELECT icode, descp  
    FROM items  
    WHERE QOH <ROL  
);
```

This will create a new table called orditem that stores two columns : itemcode and description for [www.cbsepythonedu.com] the items that have their QOH less than ROL in the relation items.

Changing Data with DML commands

As you know that DML (Data Manipulation Language) division of SQL, is dedicated to manipulating data in one way or another. This section is going to talk about various DML commands that are used for changing data in tables. Of these commands, INSERT INTO command is used for entering [www.cbsepythonedu.com] data into tables.

Inserting Data into Table

Values are placed in and removed from attributes of a relation with three DML commands : INSERT, DELETE and UPDATE. These are all referred to in SQL as update commands in a generic sense. In our text, lowercase “update” will indicate these commands generically and the [www.cbsepythonedu.com] uppercase for the keyword UPDATE.

Insert Into Command

The rows (tuples) are added to relations using INSERT command of SQL. In its simplest form,

python :

INSERT takes the following syntax :

Syntax:

```
INSERT INTO <tablename> [<column list>]  
VALUES (<value>, <value>...);
```

Example Add the following data in the above(www.cbsepythonedu.com) table (Employee) as instructed.

ID	First_Name	Last_Name	User_ID	Salary
1	Dim	Joseph	Jdim	5000
2	Jagannath	Mishra	Jnmishra	4000
3	Siddharth	mishra	Smishra	8000
4	Shankar	Giri	sgiri	7000
5	Gautam	Buddha	bgautam	2000

- (i) Populate table with record mentioning the column list in the insert clause.
- (ii) Populate table with next two records without mentioning the column list in the insert clause.
- (iii) Populate (www.cbsepythonedu.com) table with 4th record and enter only ID and First_Name
- (iv) Populate table with 5th record and enter ID, User_ID and Last_Name only

Solution.

- (i) INSERT INTO Employee (ID, First_name, Last_name, User_ID, Salary)
 Values (1, 'Dim', 'Joseph', 'Jdim', 5000);
- (ii) INSERT INTO Employee
 Values (2, 'Jagannath', 'Mishra', 'jnmishra', 4000);
 INSERT INTO Employee
 Values (3, 'Siddharth', 'mishra', 'smishra', 8000);
- (iii) INSERT INTO Employee (ID, First_Name)
 Values (4, 'Shankar');
- (iv) INSERT INTO Employee (ID, User_ID, Last_Name)
 Values (5, 'bgautam', 'Buddha');

Inserting NULL values

To insert value NULL in a specific column, you can type NULL without quotes and NULL will be (www.cbsepythonedu.com) inserted in that column. Consider the following statement :

```
INSERT INTO EMPL (Empno, Ename, Job, Mgr, Hiredate, Sal, Comm, Depton)  
Values (8100, 'YASH', 'ANALYST', NULL, '10-MAY-03', 6000, NULL, 20);
```

Inserting Dates

Dates are by default entered in 'YYYY-MM-DD' format i.e., first four digits depicting year, followed by a hyphen, followed by 2 digits month, followed by a hyphen and a two digit day. All this is (www.cbsepythonedu.com) enclosed in single quotes.

Inserting Data from Another Table

INSERT command can also be used to take or derive values from one table and place them in

python :

another by using it with a query. To do this, simply replace the VALUES clause with an appropriate query as shown in the following example :

```
INSERT INTO branch1
SELECT * FROM branch2
WHERE gross > 7000.00;
```

Modifying data with UPDATE command

Sometimes you need to change some or all of the values in an existing row. This can be done using the UPDATE command of SQL. The UPDATE command specifies the rows to be changed using the WHERE clause, and the new data using the SET keyword. The new data can be a specified [www.cbsepythonedu.com] constant, an expression or data from other tables. For example, to change the reorder level ROL of all items to 250, you would write.

```
UPDATE items
SET ROL = 250;
```

If you want to change ROL to 400 only for those items that have ROL as 300, you use the [www.cbsepythonedu.com] command

```
UPDATE items
SET ROL = 400
WHERE ROL = 300 ;
```

Updating Multiple Columns

To update multiple columns, multiple column assignments can be specified with SET clause, separated by commas. All of the said assignments will still be made to the table, a single row at a time. To update the ROL and QOH of items having icode less than I040', we shall write

```
UPDATE items
SET ROL = 400, QOH = 700
WHERE icode < ' I040' ;
```

Using Expressions in Update

Scalar expressions can also be used in the SET clause of the UPDATE command. Suppose, if you want to increase the gross pay of all the employees by Rs. 900, [www.cbsepythonedu.com] you could use the following expression :

```
UPDATE employee
SET gross = gross + 900 ;
```

To double the gross pay of employees of grade 'E3' and 'E4' you use the command :

```
UPDATE employee
SET gross = gross * 2
```

python :

WHERE (GRADE = 'E3' OR grade = 'E4');

Updating to NULL Values

The NULL values can also be entered just as other values. For example, a new grade is to be introduced and all the employees with grade 'E4' have to be promoted to it. But for the time being this grade is not known, thus NULL values are to be inserted for grades 'E4'. This can be done as [www.cbsepythonedu.com] follows :

```
UPDATE employees
SET grade = NULL
WHERE grade = 'E4'
```

Deleting Data with DELETE Command

The DELETE command removes rows from a table. This removes the entire rows, not individual field value, so no field argument is needed or accepted. The DELETE statement takes the following [www.cbsepythonedu.com] general form :

```
DELETE FROM <tablename>
[WHERE <predicate>];
```

To remove all the contents of items table, you use the command :

```
DELETE FROM items ;
```

The table would now be empty and could be destroyed with a DROP TABLE command. The DDL division of SQL is responsible for defining redefining, modifying dropping various database objects. DDL commands for altering tables, adding comments, enabling/disabling constraints, truncating [www.cbsepythonedu.com] tables, renaming objects, dropping tables, views and synonyms etc.

Alter Table Command

The ALTER TABLE command is used to change definitions of existing tables. Usually, it can add columns to a table. Sometimes it can delete columns (depending on privileges) or change their sizes.

In general, in MySQL SQL, ALTER TABLE command is used :

- * to add a column
- * to add an integrity constraint
- * to redefine a column [www.cbsepythonedu.com] (datatype, size, default value).

Adding columns

Syntax:

```
ALTER TABLE <table name> ADD <column name> <data type> <size>
[<constraint name>];
```

The new column will be added with NULL values for all rows currently in the table. It is generally possible to add several new columns, separated by commas, in a single command. It increasing their

python :

size. For instance, to add a new column tel_number of type integer in table Empl you may give :

Modifying Column Definitions

You can use the MODIFY clause to change any of the following parts of a column definition :

- | | | |
|------------------------------|--------|-------------------|
| * datatype | * size | * default value |
| * NOT NULL column constraint | | * Order of column |

The MODIFY clause need only specify the column name and the modified part of the definition, [www.cbsepythonedu.com] rather than the entire column definition.

Datatypes and Size

You can change a CHAR column to VARCHAR and a VARCHAR TO CHAR only if the column contains nulls in all rows or if you do not attempt to change the column size. You can change any column's datatype or decrease any column's size if all rows for the column contain nulls. However, you can always increase the size of a CHAR OR VARCHAR column or the precision of a numeric [www.cbsepythonedu.com] column.

Default values

A change to a column's default value only affects rows subsequently inserted into the table. Such [www.cbsepythonedu.com] change does not change default values previously inserted.

Integrity constraints

The only type of integrity constraint that you can add to an existing column using the MODIFY clause with the column constraint syntax is a NOT NULL constraint. However, you can define other types of integrity constraints (UNIQUE, PRIMARY KEY, [www.cbsepythonedu.com] referential integrity, and CHECK constraints) on existing columns using the ADD clause and the table constraint syntax.

Order of Column

With MODIFY clause you can also reorder the columns within a table. For this you need to use FIRST OR AFTER <column> clause to specify the position of the column. To modify existing columns of table, ALTER TABLE command can be used according to following syntax :

Syntax:

```
ALTER TABLE <tablename>
MODIFY (columnname new datatype (newsize)) [FIRST|AFTER column];
```

Changing a Column Name

Sometime you may need to change the name of one of your columns. For this you can use [www.cbsepythonedu.com] CHANGE clause of ALTER TABLE command as per following syntax :

```
ALTER TABLE
CHANGE [COLUMN] old_col_name column_definition ;
```

python :

For instance, to change the existing column namely First_Name of table student, to FirstName you may write :

```
ALTER TABLE Customers  
CHANGE First_name FirstName VARCHAR (20);
```

Removing table Components

If you plan to remove a component of a table, then you may use the DROP clause of ALTER TABLE. The keywords mostly used with DROP clause of ALTER TABLE command are :

Drops the table's primary key constraint. Removes mentioned column from the table Removes the **[www.cbsepythonedu.com]** mentioned foreign key constraint from the table. Now consider the following examples :

```
ALTER TABLE Empl  
DROP PRIMARY KEY, DROP FOREIGN KEY fk_1, DROP COLUMN  
depton ;
```

The DROPTABLE Command

The DROP TABLE command of SQL lets you drop a table from the database. The syntax for using a DROP TABLE command is :

```
DROP TABLE [IF EXISTS] <tablename>
```

That is, to drop a table items, you need to write :

```
DROP TABLE items ;
```