Database Handling in Python

It is very important to understand the database before learning MySQL. A database is an application that stores the organized collection of records. It can be accessed and manage by the user very easily. It allows us to organize data into tables, rows, columns, and indexes to find the relevant information very quickly. Each database contains distinct API for performing database operations such as creating, managing, accessing, and searching the data it stores. Today, many databases available like MySQL, Sybase, Oracle, MongoDB, PostgreSQL, SQL Server, etc. In this section, we are going to focus on MySQL mainly.

What is MySQL?

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database.

MySQL is a Relational Database Management System (RDBMS) software that provides many things, which are as follows:

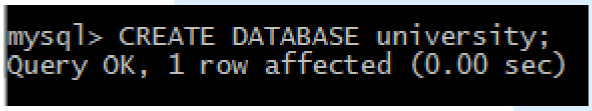
- It allows us to implement database operations on tables, rows, columns, and indexes.
- It defines the database relationship in the form of tables (collection of rows and columns), also known as relations.
- It provides the Referential Integrity between rows or columns of various tables.
- It allows us to updates the table indexes automatically.
- It uses many SQL queries and combines useful information from multiple tables for the end-users.

MySQL Queries for Database

A database is used to store the collection of records in an organized form. It allows us to hold the data into tables, rows, columns, and indexes to find the relevant information frequently. We can access and manage the records through the database very easily.

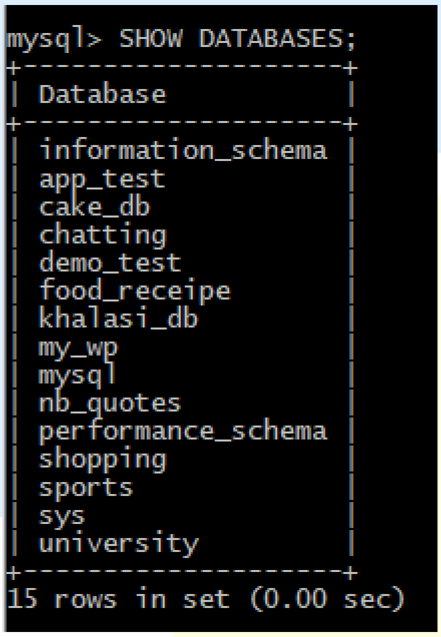
To create a database, type the following command: -

- CREATE DATABASE database_name;
- Eg:- CREATE DATABASE university;



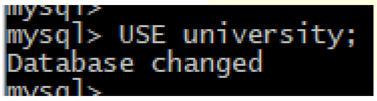
```
mysql> CREATE DATABASE university;
Query OK, 1 row affected (0.00 sec)
```

1. To show all the Databases in MySQL:-
   SHOW DATABASES;



```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| app_test           |
| cake_db            |
| chatting           |
| demo_test          |
| food_receipe       |
| khalasi_db         |
| my_wp              |
| mysql              |
| nb_quotes          |
| performance_schema |
| shopping           |
| sports             |
| sys                |
| university         |
+--------------------+
15 rows in set (0.00 sec)
```

2. To use a Database, we will use:
   USE university;



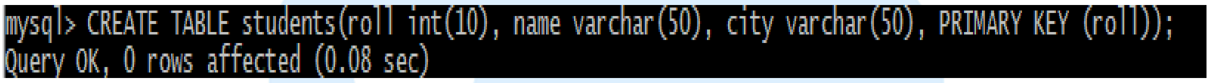
```
mysql>
mysql> USE university;
Database changed
mysql>
```

MySQL Queries for Tables

Create a Table

MySQL create query is used to create a table, view, and function. For example:

CREATE TABLE students (roll int(10), name varchar(50), city varchar(50), PRIMARY KEY (roll));

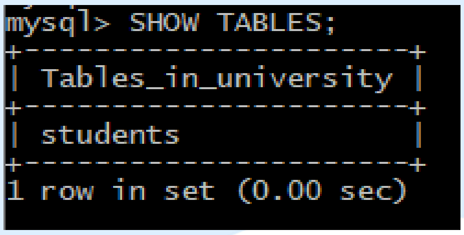


```
mysql> CREATE TABLE students(roll int(10), name varchar(50), city varchar(50), PRIMARY KEY (roll));
Query OK, 0 rows affected (0.08 sec)
```

To Check if Table is Created

To check what Tables exists in our Database, type the following command:-

- SHOW TABLES;



```
mysql> SHOW TABLES;
+----------------------+
| Tables_in_university |
+----------------------+
| students             |
+----------------------+
1 row in set (0.00 sec)
```

INSERT Statement Syntax

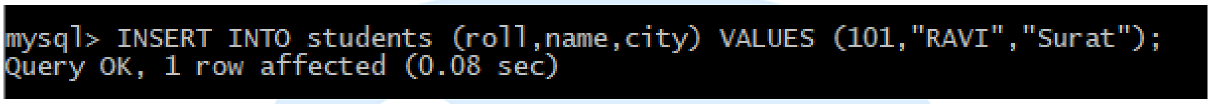
MySQL INSERT statement is used to store or add data in MySQL table within the database. We can perform insertion of records in two ways using a single query in MySQL:

- Insert record in a single row
- Insert record in multiple rows

Insert record in a single row

The below is syntax of SQL INSERT INTO command to insert a single record in MySQL table:

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



```
mysql> INSERT INTO students (roll,name,city) VALUES (101,"RAVI","Surat");
Query OK, 1 row affected (0.08 sec)
```

Insert record in multiple rows

If we want to insert multiple records within a single command, use the following statement:

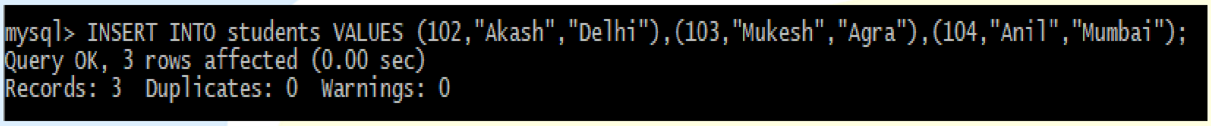
INSERT INTO table_name VALUES

(value1, value2,...valueN),

(value1, value2,...valueN),

....

(value1, value2,...valueN);



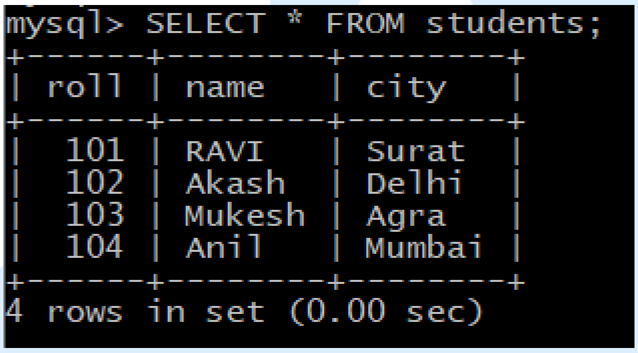
```
mysql> INSERT INTO students VALUES (102,"Akash","Delhi"),(103,"Mukesh","Agra"),(104,"Anil","Mumbai");
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
```

SELECT Statement Syntax

The SELECT statement in MySQL is used to fetch data from one or more tables. We can retrieve records of all fields or specified fields that match specified criteria using this statement.

To retrieve all values from a Table: -

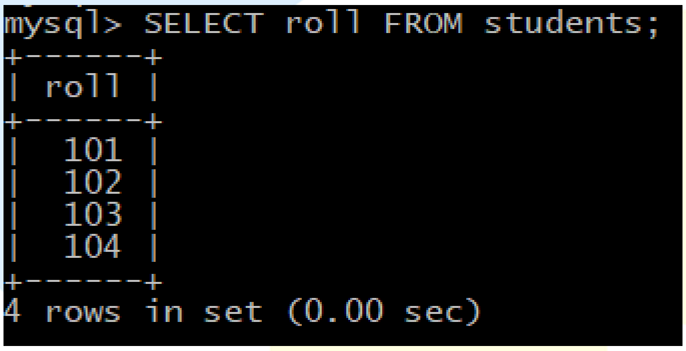
SELECT * FROM table_name;



```
mysql> SELECT * FROM students;
+------+--------+--------+
| roll | name   | city   |
+------+--------+--------+
|  101 | RAVI   | Surat  |
|  102 | Akash  | Delhi  |
|  103 | Mukesh | Agra   |
|  104 | Anil   | Mumbai |
+------+--------+--------+
4 rows in set (0.00 sec)
```

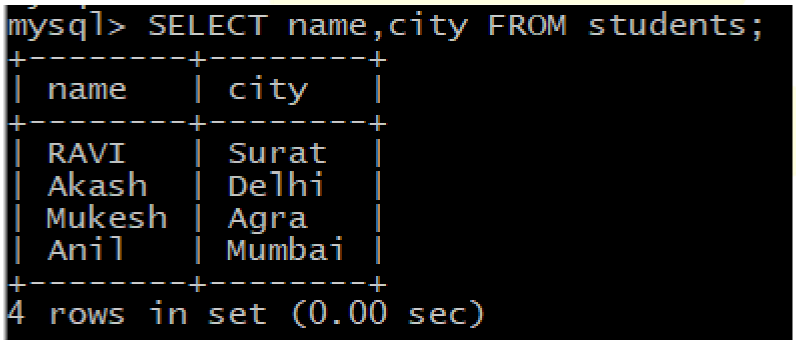
To retrieve particular/multiple values from a table: -

SELECT value1 FROM table_name;



```
mysql> SELECT roll FROM students;
+------+
| roll |
+------+
|  101 |
|  102 |
|  103 |
|  104 |
+------+
4 rows in set (0.00 sec)
```

SELECT value1,value2,….,valueN FROM table_name;

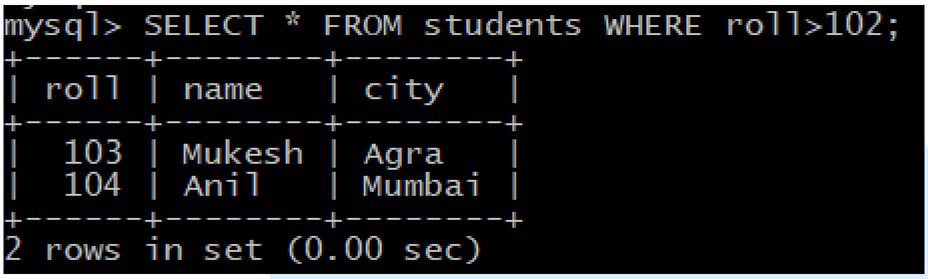


```
mysql> SELECT name,city FROM students;
+--------+--------+
| name   | city   |
+--------+--------+
| RAVI   | Surat  |
| Akash  | Delhi  |
| Mukesh | Agra   |
| Anil   | Mumbai |
+--------+--------+
4 rows in set (0.00 sec)
```

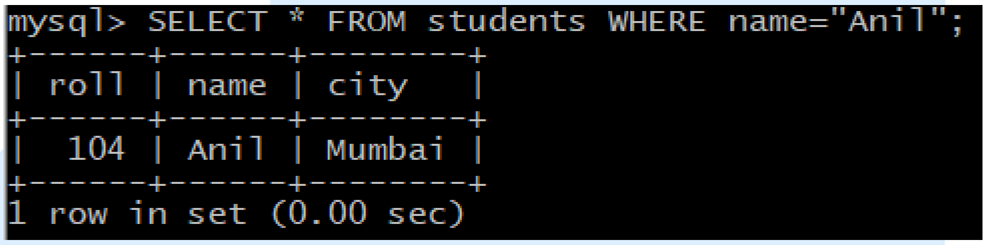
To retrieve particular/multiple values from a table using CONDITION: -

We will retrieve values from table where we should satisfy condition: -

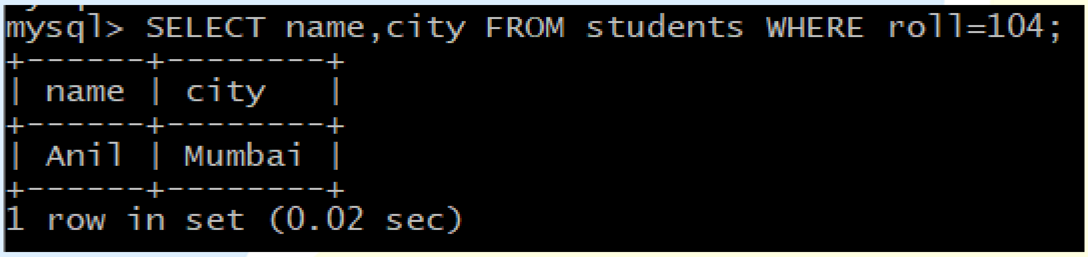
SELECT * FROM table_name WHERE condition;



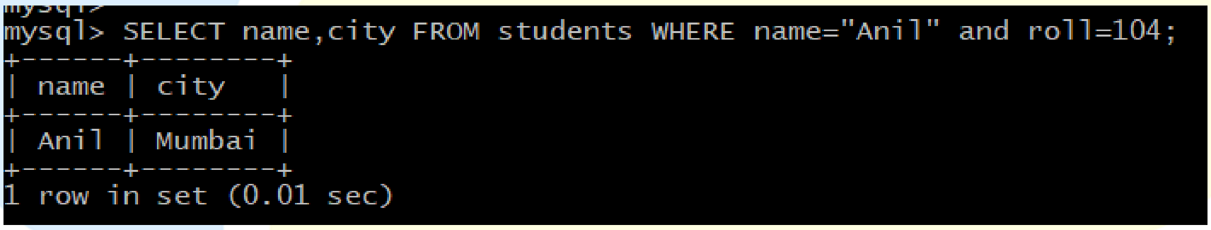
```
mysql> SELECT * FROM students WHERE roll>102;
+------+--------+--------+
| roll | name   | city   |
+------+--------+--------+
|  103 | Mukesh | Agra   |
|  104 | Anil   | Mumbai |
+------+--------+--------+
2 rows in set (0.00 sec)
```



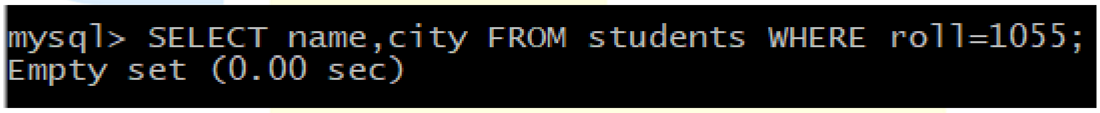
```
mysql> SELECT * FROM students WHERE name="Anil";
+------+------+--------+
| roll | name | city   |
+------+------+--------+
|  104 | Anil | Mumbai |
+------+------+--------+
1 row in set (0.00 sec)
```



```
mysql> SELECT name,city FROM students WHERE roll=104;
+------+--------+
| name | city   |
+------+--------+
| Anil | Mumbai |
+------+--------+
1 row in set (0.02 sec)
```



```
mysql> SELECT name,city FROM students WHERE name="Anil" and roll=104;
+------+--------+
| name | city   |
+------+--------+
| Anil | Mumbai |
+------+--------+
1 row in set (0.01 sec)
```

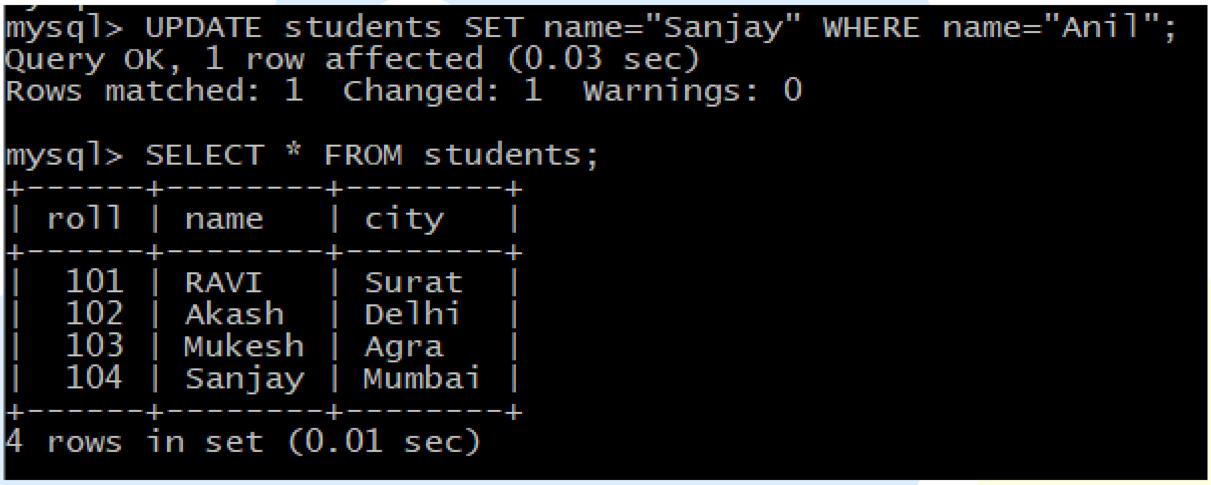


```
mysql> SELECT name,city FROM students WHERE roll=1055;
Empty set (0.00 sec)
```

UPDATE Statement Syntax

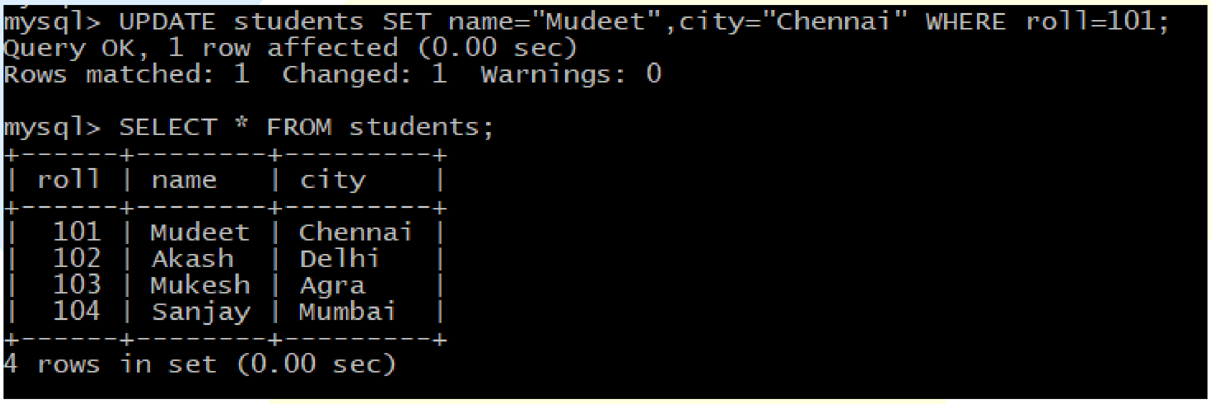
The UPDATE statement is used with the **SET** and **WHERE** clauses. The SET clause is used to change the values of the specified column. We can update single or multiple columns at a time.

UPDATE table_name SET column_name1 = new-value1, column_name2=new-value2, ... column_name3=new-value3 WHERE condition;



```
mysql> UPDATE students SET name="Sanjay" WHERE name="Anil";
Query OK, 1 row affected (0.03 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM students;
+------+--------+--------+
| roll | name   | city   |
+------+--------+--------+
|  101 | RAVI   | Surat  |
|  102 | Akash  | Delhi  |
|  103 | Mukesh | Agra   |
|  104 | Sanjay | Mumbai |
+------+--------+--------+
4 rows in set (0.01 sec)
```

Updating Multiple columns at same time



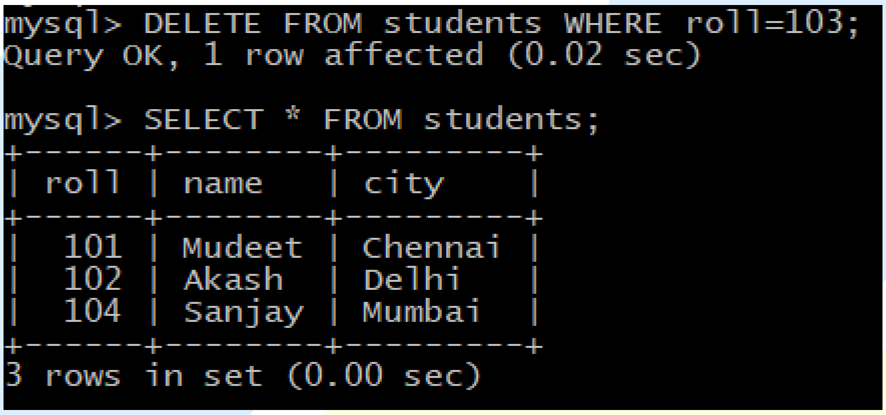
```
mysql> UPDATE students SET name="Mudeet",city="Chennai" WHERE roll=101;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM students;
+------+--------+---------+
| roll | name   | city    |
+------+--------+---------+
|  101 | Mudeet | Chennai |
|  102 | Akash  | Delhi   |
|  103 | Mukesh | Agra    |
|  104 | Sanjay | Mumbai  |
+------+--------+---------+
4 rows in set (0.00 sec)
```

DELETE Statement Syntax

MySQL DELETE statement is used to remove records from the MySQL table that is no longer required in the database. This query in MySQL deletes a full row from the table and produces the count of deleted rows. 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.

Once we delete the records using this query, we cannot recover it.

DELETE FROM table_name WHERE condition;



```
mysql> DELETE FROM students WHERE roll=103;
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM students;
+------+---------+---------+
| roll | name    | city    |
+------+---------+---------+
|  101 | Mudeet  | Chennai |
|  102 | Akash   | Delhi   |
|  104 | Sanjay  | Mumbai  |
+------+---------+---------+
3 rows in set (0.00 sec)
```

MySQL using Python

Python needs a MySQL driver to access the MySQL database.

To install MySQL Driver type -> **pip install mysql-connector-python**

To test if installation is successful create empty Python file and type: -

import mysql.connector

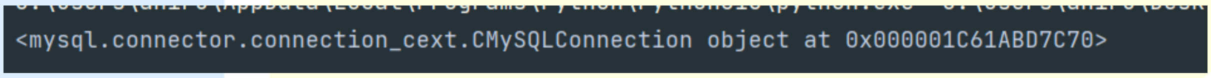
If the above code was executed with no errors, "MySQL Connector" is installed and ready to be used.

Create a Connection

Start by creating a connection to the database. Use the username and password from your MySQL database:

```python
import mysql.connector

mydb = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password=""
)

print(mydb)
```

Output



```
<mysql.connector.connection_cext.CMySQLConnection object at 0x000001C61ABD7C70>
```

Create a table through python

To create a table in MySQL, use the "CREATE TABLE" statement.

Make sure you define the name of the database when you create the connection.

```
import mysql.connector

mydb = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password="",
    database="university",
)

mycursor = mydb.cursor()
mycursor.execute("CREATE TABLE students (name VARCHAR(255),
address VARCHAR(255), roll INT PRIMARY KEY)")

mydb.close()
```

SELECT Statement through Python

To select from a table in MySQL, use the "SELECT" statement:

```python
import mysql.connector

mydb = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password="",
    database="university"
)

mycursor = mydb.cursor()
mycursor.execute("SELECT * FROM students")

# fetchall fetches all the rows.
myresult = mycursor.fetchall()

for x in myresult:
    print(x)

print("\nFetching using fetchone()")

mycursor.execute("SELECT * FROM students")
# fetchone only fetches one row.
myresult2 = mycursor.fetchone()
print(myresult2)

mydb.close()
```

Output

```
('Elon', 'new york', 33)
('Anirudh', 'xyz, surat, gujarat', 55)
('Ananya', 'Mumbai', 100)

Fetching using fetchone()
('Elon', 'new york', 33)
```

UPDATE Statement through Python

To update from a table in MySQL, use the "UPDATE" statement:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password="",
    database="university"
)

mycursor = mydb.cursor()
mycursor.execute("UPDATE students SET name='xyz' WHERE
roll=55")

# To save the changes
mydb.commit()

mydb.close()
```

DELETE Statement through Python

To delete from a table in MySQL, use the "DELETE" statement:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password="",
    database="university"
)

mycursor = mydb.cursor()
mycursor.execute("DELETE FROM students WHERE name='Elon'")

# To save the changes
mydb.commit()

mydb.close()
```