

PYTHON SQL DATABASE COMMAND

Download and install "MySQL Connector":

```
C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>python -m pip install mysql-connector
```

Now you have downloaded and installed a MySQL driver.

Test MySQL Connector

To test if the installation was successful, or if you already have "MySQL Connector" installed, create a Python page with the following content:

demo_mysql_test.py:

```
import mysql.connector
```

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

Create Connection

Start by creating a connection to the database.

Use the username and password from your MySQL database:

demo_mysql_connection.py:

```
import mysql.connector
```

```
mydb = mysql.connector.connect(  
    host="localhost",  
    user="yourusername",  
    passwd="yourpassword"  
)  
print(mydb)
```

Creating a Database

To create a database in MySQL, use the "CREATE DATABASE" statement:

Example

create a database named "mydatabase":

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword"
)

mycursor = mydb.cursor()

mycursor.execute("CREATE DATABASE mydatabase")
```

Check if Database Exists

You can check if a database exist by listing all databases in your system by using the "SHOW DATABASES" statement:

Example

Return a list of your system's databases:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword"
)

mycursor = mydb.cursor()

mycursor.execute("SHOW DATABASES")

for x in mycursor:
    print(x)
```

Or you can try to access the database when making the connection:

Example

Try connecting to the database "mydatabase":

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)
```

If the database does not exist, you will get an error.

Creating a Table

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

Example

Create a table named "customers":

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()

mycursor.execute("CREATE TABLE customers (name VARCHAR(255), address VARCHAR(255))")
```

If the above code was executed with no errors, you have now successfully created a table.

Check if Table Exists

You can check if a table exist by listing all tables in your database with the "SHOW TABLES" statement:

Example

Return a list of your system's databases:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()

mycursor.execute("SHOW TABLES")

for x in mycursor:
    print(x)
```

Primary Key

When creating a table, you should also create a column with a unique key for each record.

This can be done by defining a PRIMARY KEY.

We use the statement "INT AUTO_INCREMENT PRIMARY KEY" which will insert a unique number for each record. Starting at 1, and increased by one for each record.

Example

Create primary key when creating the table:

```
import mysql.connector
```

```
mydb = mysql.connector.connect(  
    host="localhost",  
    user="yourusername",  
    passwd="yourpassword",  
    database="mydatabase"  
)
```

```
mycursor = mydb.cursor()
```

```
mycursor.execute("CREATE TABLE customers (id INT AUTO_INCREMENT PRIMARY  
KEY, name VARCHAR(255), address VARCHAR(255))")
```

If the table already exists, use the ALTER TABLE keyword:

Example

Create primary key on an existing table:

```
import mysql.connector
```

```
mydb = mysql.connector.connect(  
    host="localhost",  
    user="yourusername",  
    passwd="yourpassword",  
    database="mydatabase"  
)
```

```
mycursor = mydb.cursor()
```

```
mycursor.execute("ALTER TABLE customers ADD COLUMN id INT  
AUTO_INCREMENT PRIMARY KEY")
```

Insert Into Table

To fill a table in MySQL, use the "INSERT INTO" statement.

Example

Insert a record in the "customers" table:

```
import mysql.connector
```

```
mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"
val = ("John", "Highway 21")
mycursor.execute(sql, val)

mydb.commit()

print(mycursor.rowcount, "record inserted.")
```

Important!: Notice the statement: `mydb.commit()`. It is required to make the changes, otherwise no changes are made to the table.

Insert Multiple Rows

To insert multiple rows into a table, use the `executemany()` method.

The second parameter of the `executemany()` method is a list of tuples, containing the data you want to insert:

Example

Fill the "customers" table with data:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()
```

```

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"
val = [
    ('Peter', 'Lowstreet 4'),
    ('Amy', 'Apple st 652'),
    ('Hannah', 'Mountain 21'),
    ('Michael', 'Valley 345'),
    ('Sandy', 'Ocean blvd 2'),
    ('Betty', 'Green Grass 1'),
    ('Richard', 'Sky st 331'),
    ('Susan', 'One way 98'),
    ('Vicky', 'Yellow Garden 2'),
    ('Ben', 'Park Lane 38'),
    ('William', 'Central st 954'),
    ('Chuck', 'Main Road 989'),
    ('Viola', 'Sideway 1633')
]

mycursor.executemany(sql, val)

mydb.commit()

print(mycursor.rowcount, "was inserted.")

```

Get Inserted ID

You can get the id of the row you just inserted by asking the cursor object.

Note: If you insert more than one row, the id of the last inserted row is returned.

Example

Insert one row, and return the ID:

```

import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

```

```
mycursor = mydb.cursor()

sql = "INSERT INTO customers (name, address) VALUES (%s, %s)"
val = ("Michelle", "Blue Village")
mycursor.execute(sql, val)

mydb.commit()

print("1 record inserted, ID:", mycursor.lastrowid)
```

Select From a Table

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

Example

Select all records from the "customers" table, and display the result:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()

mycursor.execute("SELECT * FROM customers")

myresult = mycursor.fetchall()

for x in myresult:
    print(x)
```

Note: We use the `fetchall()` method, which fetches all rows from the last executed statement.

Selecting Columns

To select only some of the columns in a table, use the "SELECT" statement followed by the column name(s):

Example

Select only the name and address columns:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
    user="yourusername",
    passwd="yourpassword",
    database="mydatabase"
)

mycursor = mydb.cursor()

mycursor.execute("SELECT name, address FROM customers")

myresult = mycursor.fetchall()

for x in myresult:
    print(x)
```

Using the fetchone() Method

If you are only interested in one row, you can use the `fetchone()` method.

The `fetchone()` method will return the first row of the result:

Example

Fetch only one row:

```
import mysql.connector

mydb = mysql.connector.connect(
    host="localhost",
```

```
user="yourusername",  
passwd="yourpassword",  
database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
mycursor.execute("SELECT * FROM customers")  
  
myresult = mycursor.fetchone()  
  
print(myresult)
```