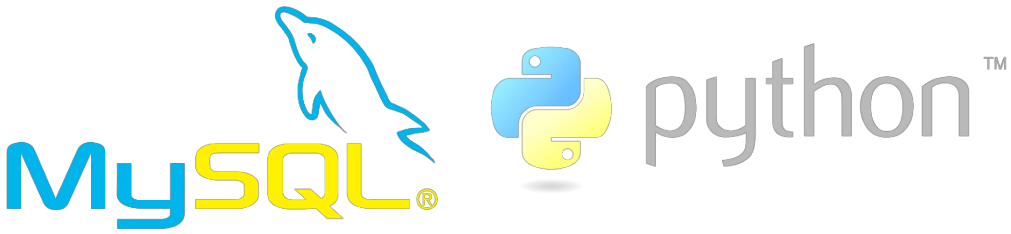
PYTHON MYSQL



To build the real world applications, connecting with the databases is the necessity for the programming languages. However, python allows us to connect our application to the databases like MySQL, SQLite, MongoDB, and many others.

- MySQL is a fast, easy to use relational database. It is currently the most popular open-source database
- MySQL is used for many small and big businesses. It is developed, marketed and supported by MySQL AB, a Swedish company. It is written in C and C++.
- MySQL is an open-source database, so you don't have to pay a single penny to use it.

MySQL Features:

- MySQL is a fast, easy to use relational database.
- MySQL is used for many small and big businesses.
- MySQL is an open-source database, so you don't have to pay for it.

Install MySQL

Download and install "MySQL Connector":

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

Test MySQL Connector

demo_mysql_test.py: import mysql.connector

Create Connection

```
demo_mysql_connection.py:
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword"
)
print(mydb)
```

Python needs a MySQL driver to access the MySQL database. We will use the driver "MySQL Connector". We recommend that you use PIP to install "MySQL Connector". PIP is most likely already installed in your Python environment.

To test if the installation was successful, or if you already have "MySQL Connector" installed, create a Python page with the content: **import mysql.connector**.

Creating Connection to the database. You should use the username and password from your MySQL database. After that you can start querying the database using SQL statements.

Python MySQL Create Database

Creating a Database	Check if Database Exists
import mysql.connector	import mysql.connector

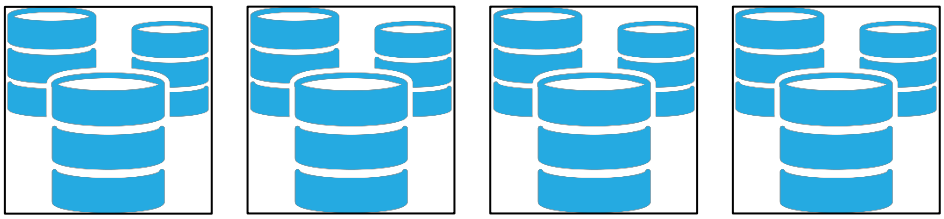
This document has been produced with the support of the EUROPEAN COMMISSION under the ERASMUS+ Programme, KA2 – Capacity Building in the Field of Higher Education: 598092-EPP-1-2018-1-BG-EPPKA2-CBHE-SP.
It reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword"
)
mycursor = mydb.cursor()
mycursor.execute("CREATE DATABASE mydatabase")
```

```
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword"
)
mycursor = mydb.cursor()
mycursor.execute("SHOW DATABASES")
for x in mycursor:
  print(x)
```

OR

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



After establishing connection with MySQL, to manipulate data in it you need to connect to a database. You can connect to an existing database or, create your own.

You would need special privileges to create or to delete a MySQL database. So if you have access to the root user, you can create any database.

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

Let's create a database named "mydatabase" as in the picture.

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

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

Python MySQL Create Table

Creating a Table

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
  database="mydatabase"
)
mycursor = mydb.cursor()
mycursor.execute("CREATE TABLE customers (name VARCHAR(255), address VARCHAR(255))")
```

Check if Table Exists

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
  database="mydatabase"
)
mycursor = mydb.cursor()
mycursor.execute("SHOW TABLES")
for x in mycursor:
  print(x)
```

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.

This document has been produced with the support of the EUROPEAN COMMISSION under the ERASMUS+ Programme, KA2 – Capacity Building in the Field of Higher Education: 598092-EPP-1-2018-1-BG-EPPKA2-CBHE-SP.
It reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

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

Python MySQL Create Table

Primary Key	Create primary key on an existing table
import mysql.connector mydb = mysql.connector.connect(host="localhost", user="yourusername", password="yourpassword", database="mydatabase") mycursor = mydb.cursor() mycursor.execute("**CREATE TABLE** customers (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(255), address VARCHAR(255))")	import mysql.connector mydb = mysql.connector.connect(host="localhost", user="yourusername", password="yourpassword", database="mydatabase") mycursor = mydb.cursor() mycursor.execute("**ALTER TABLE** customers ADD COLUMN id INT AUTO_INCREMENT 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.

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

Python MySQL Insert Into Table

Insert a record in the "customers" table:

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="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.")
```

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

Python MySQL Insert Into Table

Insert Multiple Rows	Get Inserted ID

This document has been produced with the support of the EUROPEAN COMMISSION under the ERASMUS+ Programme, KA2 – Capacity Building in the Field of Higher Education: 598092-EPP-1-2018-1-BG-EPPKA2-CBHE-SP.
It reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
...
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.")
```

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="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)
```

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.

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.

Python MySQL Select From

Select From a Table

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
  database="mydatabase"
)
mycursor = mydb.cursor()
mycursor.execute("SELECT * FROM
customers")
myresult = mycursor.fetchall()
for x in myresult:
  print(x)
```

Selecting Columns

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
  database="mydatabase"
)
mycursor = mydb.cursor()
mycursor.execute("SELECT name, address
FROM customers")
myresult = mycursor.fetchall()
for x in myresult:
  print(x)
```

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

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

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

This document has been produced with the support of the EUROPEAN COMMISSION under the ERASMUS+ Programme, KA2 – Capacity Building in the Field of Higher Education: 598092-EPP-1-2018-1-BG-EPPKA2-CBHE-SP.
It reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Python MySQL Order By

Sort the Result	ORDER BY DESC
import mysql.connector mydb = mysql.connector.connect(host="localhost", user="yourusername", password="yourpassword", database="mydatabase") mycursor = mydb.cursor() sql = "SELECT * FROM customers **ORDER BY** name" mycursor.execute(sql) myresult = mycursor.fetchall() for x in myresult: print(x)	import mysql.connector mydb = mysql.connector.connect(host="localhost", user="yourusername", password="yourpassword", database="mydatabase") mycursor = mydb.cursor() sql = "SELECT * FROM customers **ORDER BY** name **DESC**" mycursor.execute(sql) myresult = mycursor.fetchall() for x in myresult: print(x)

Use the ORDER BY statement to sort the result in ascending or descending order.

The ORDER BY keyword sorts the result ascending by default. To sort the result in descending order, use the DESC keyword.

Python MySQL Delete From By

Delete any record where the address is "Mountain 21"

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
  database="mydatabase"
)
mycursor = mydb.cursor()
sql = "DELETE FROM customers WHERE address = 'Mountain 21'"
mycursor.execute(sql)
mydb.commit()
print(mycursor.rowcount, "record(s) deleted")
```

You can delete records from an existing table by using the "**DELETE FROM**" statement.

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

Notice the **WHERE** clause in the DELETE syntax: The **WHERE** clause specifies which record(s) that should be deleted. If you omit the WHERE clause, all records will be deleted!

Python MySQL Drop Table

Delete the table "customers":

```
import mysql.connector
mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword",
```

This document has been produced with the support of the EUROPEAN COMMISSION under the ERASMUS+ Programme, KA2 – Capacity Building in the Field of Higher Education: 598092-EPP-1-2018-1-BG-EPPKA2-CBHE-SP.
It reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

```
  database="mydatabase"
)
mycursor = mydb.cursor()
sql = "DROP TABLE customers"
mycursor.execute(sql)
```

You can delete an existing table by using the "**DROP TABLE**" statement.

REFERENCES:

1. Link : [Copyright @ 2019 Learntek. All Rights Reserved]
2. Link: [https://www.w3schools.com/python/python_mysql_create_db.asp]

This document has been produced with the support of the EUROPEAN COMMISSION under the ERASMUS+ Programme, KA2 – Capacity Building in the Field of Higher Education: 598092-EPP-1-2018-1-BG-EPPKA2-CBHE-SP.
It reflects the views only of the authors, and the Commission cannot be held responsible for any use which may be made of the information contained therein.