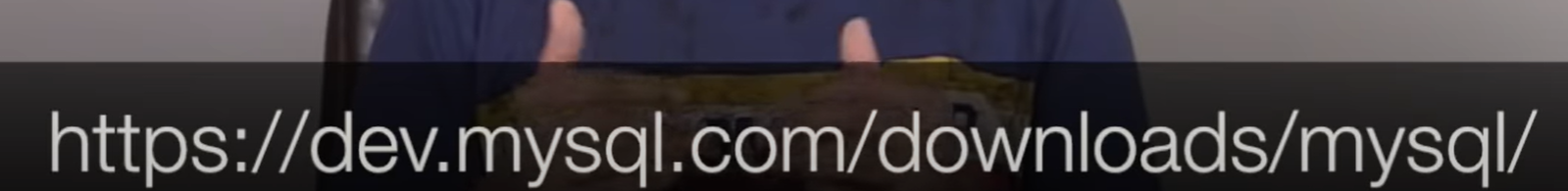
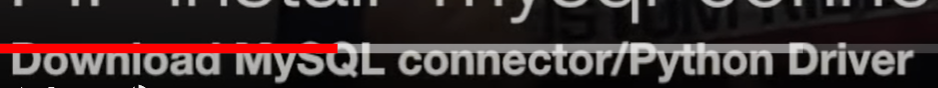
**MY SQL Database Connection with Python**

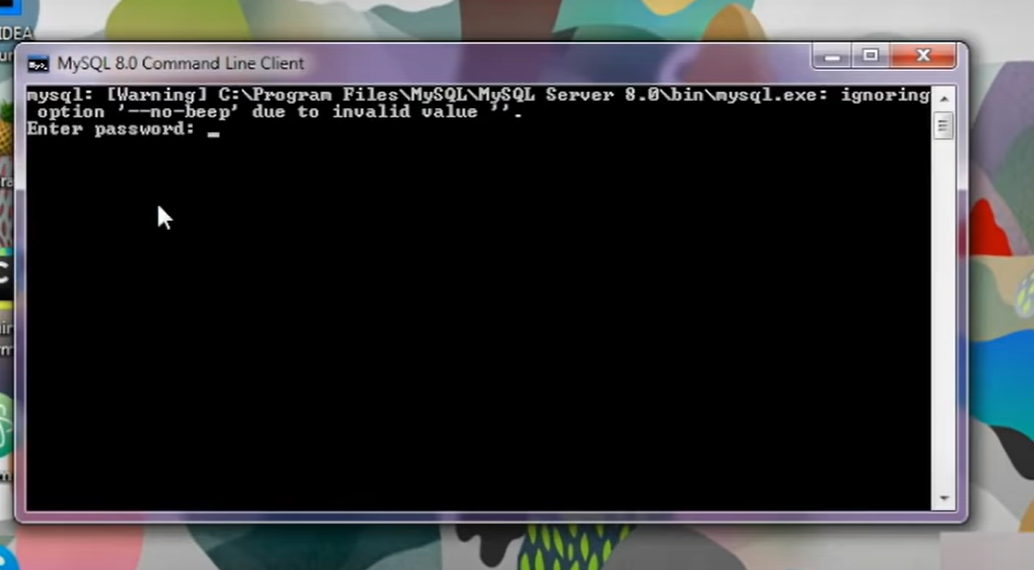
1. Download MYSQL from the following link



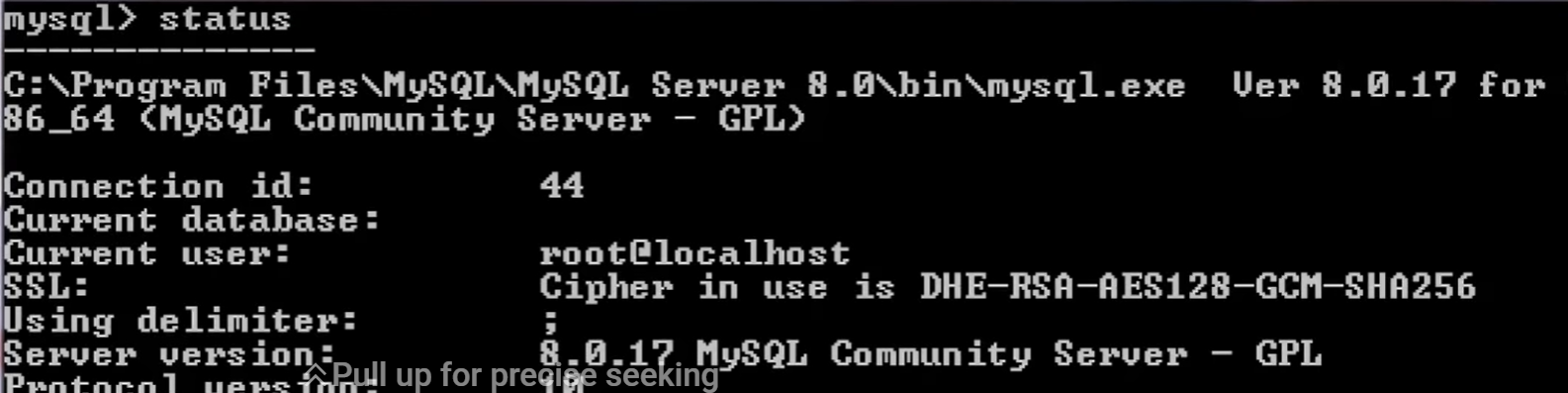
1. After downloading it install the MYSQL on your system
2. Download python drivers



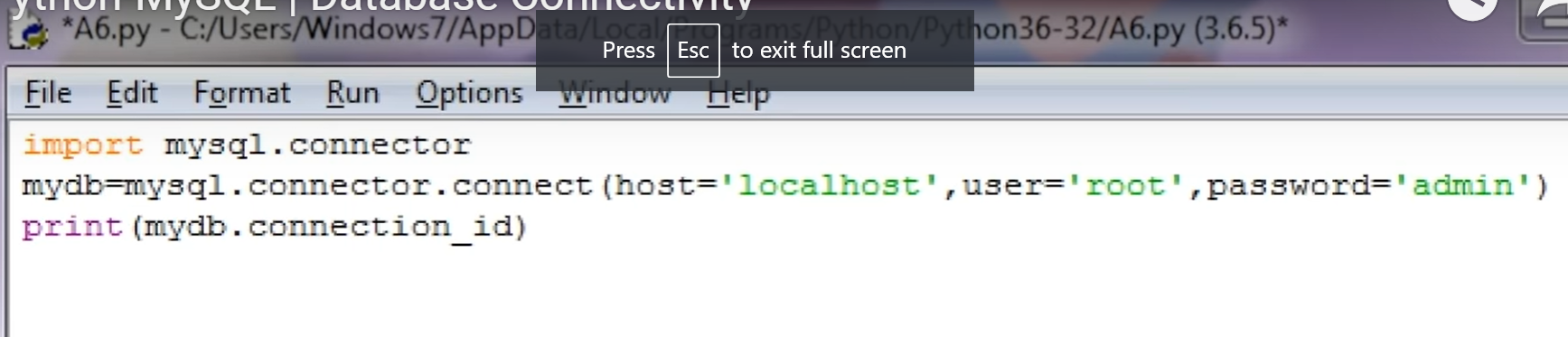
1. Then check wheterher the MYSQL client softgware is installed properly on the system or not by opening MYSQL client software from the start menu
2. A command window will open where it will ask you for login here user name is by default ‘root’ and password is ‘admin ’ which you have given at the time of intstallation.



1. Then type the command here to check wheteher it working properly or not checking the status by giving command STATUS on the prompt

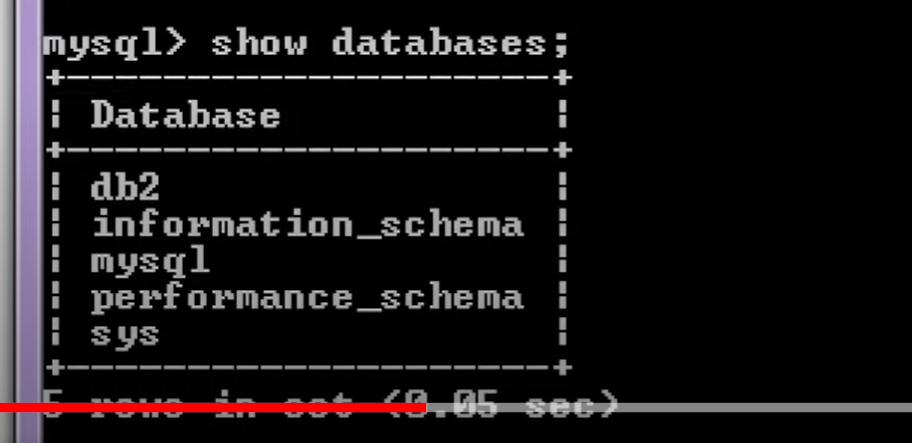


1. If you want to check the connection from the python IDLE then type the code to check it

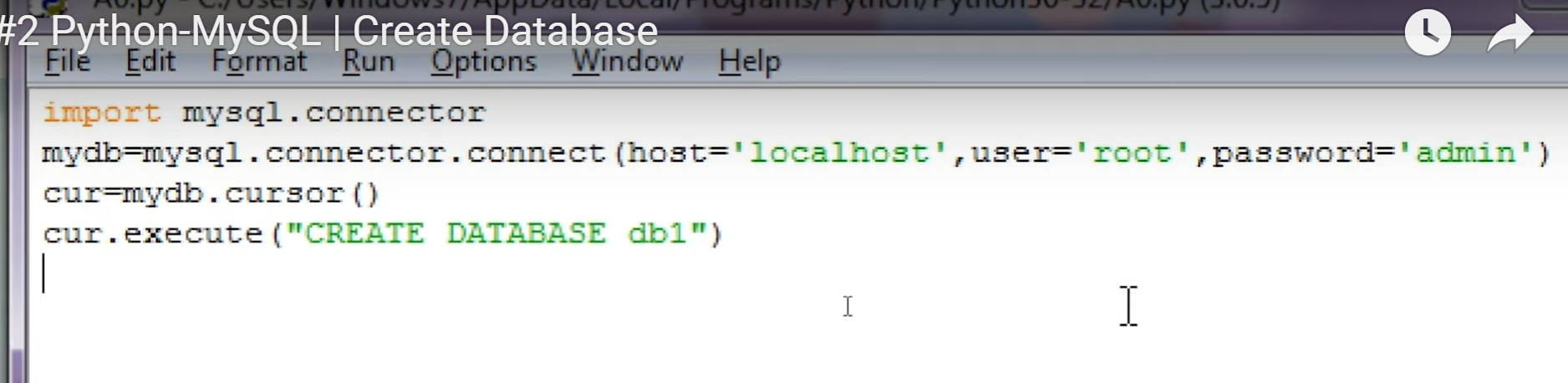


1. Now create a new database in to python using IDLE

Go on MYSQL client and show how many databases are already created in MYSQL by typing the command



Now create a new database DB1 in MYSQL



Now the DB1 database is created in MYSQL check it by going on the MYSQL client and cheking the command

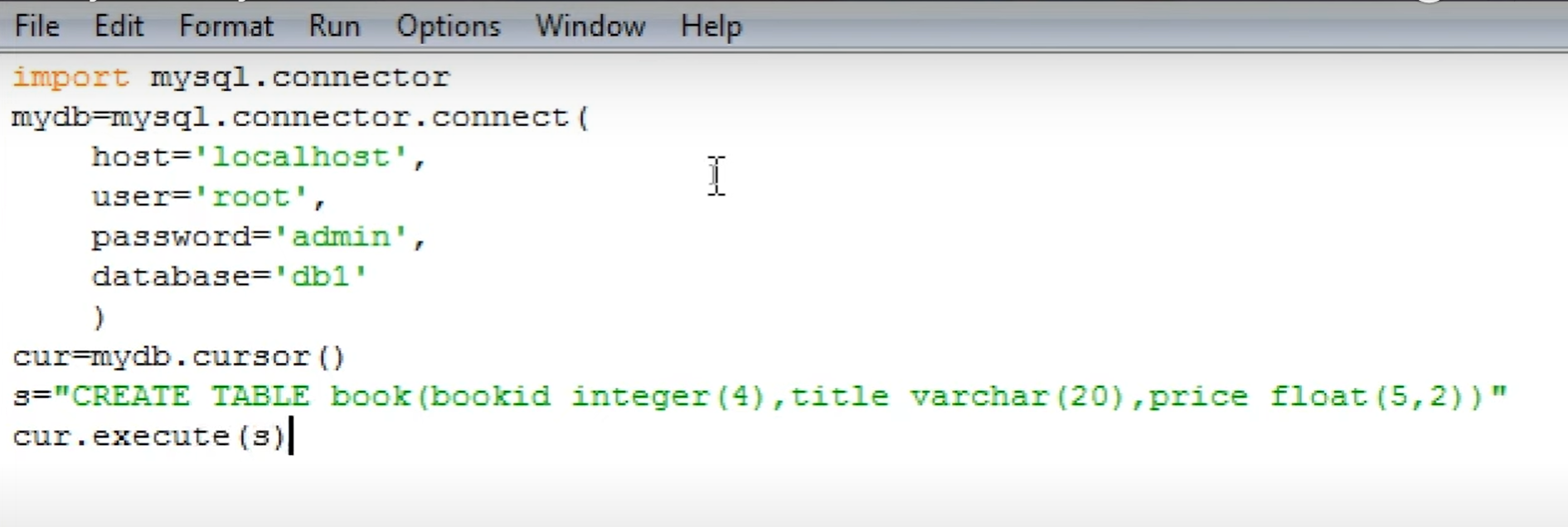
Show databae



Now see the table created in the database DB1



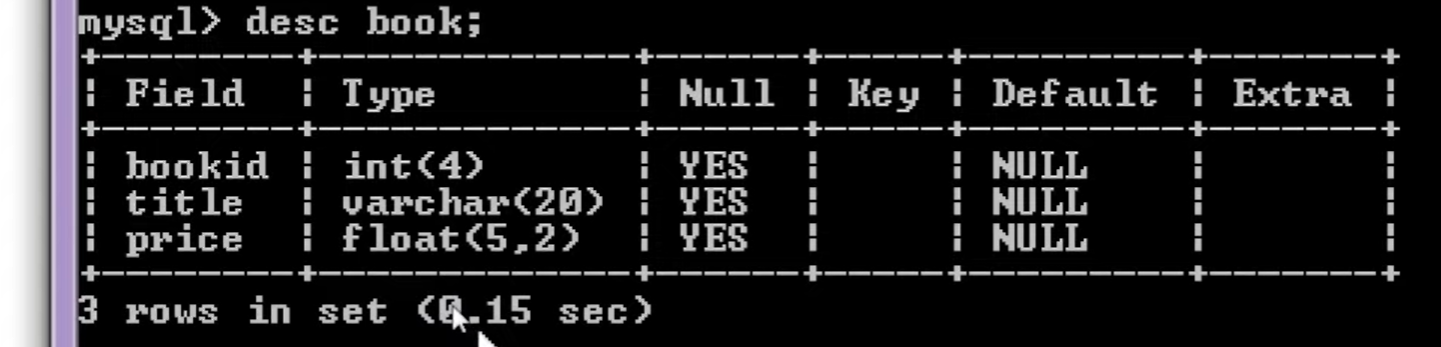
Now to create the table write the code below



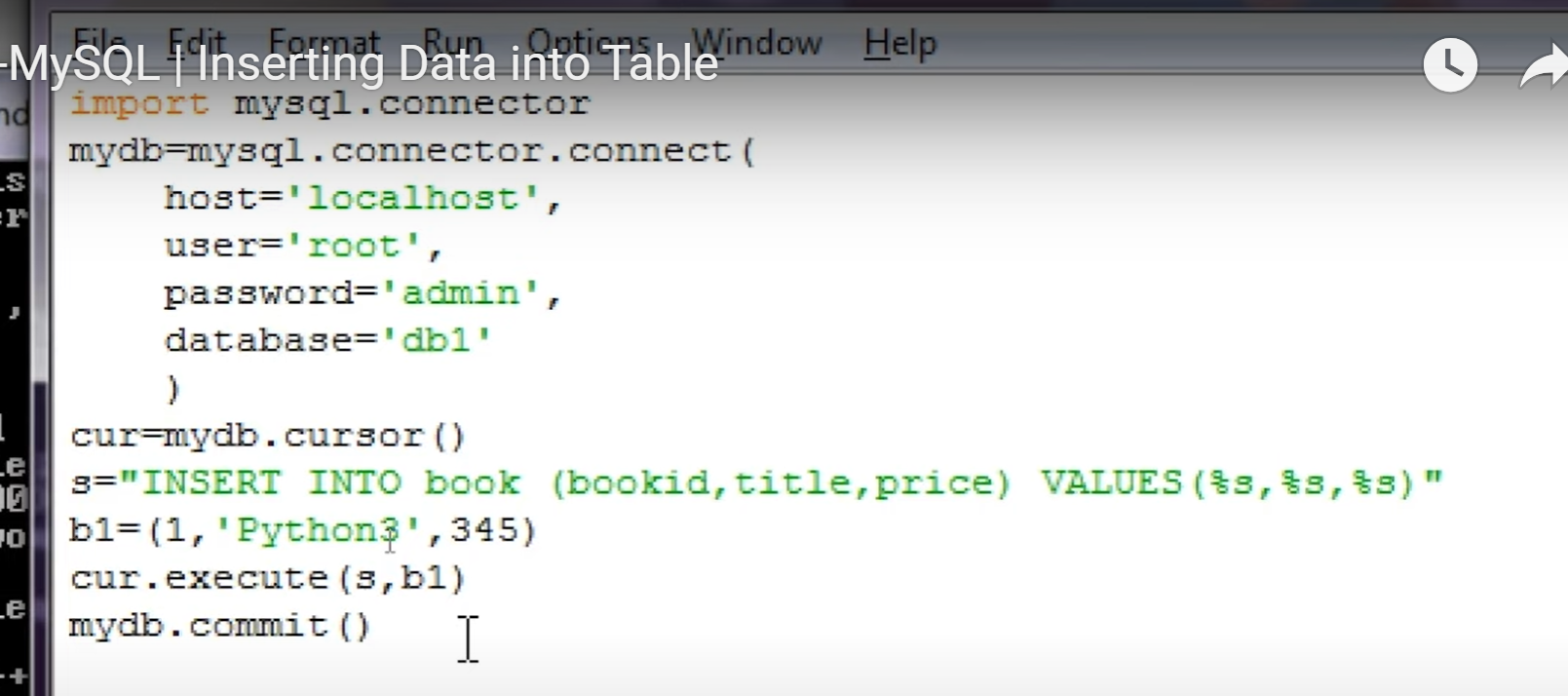
To see wheteher table is create or not write the command



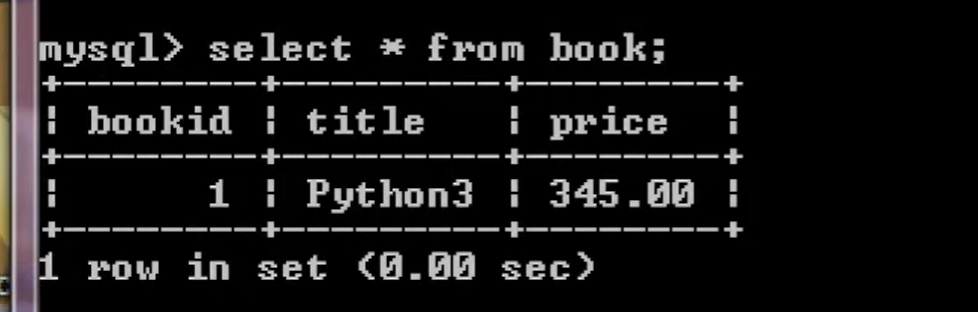
To see the table contents write the command



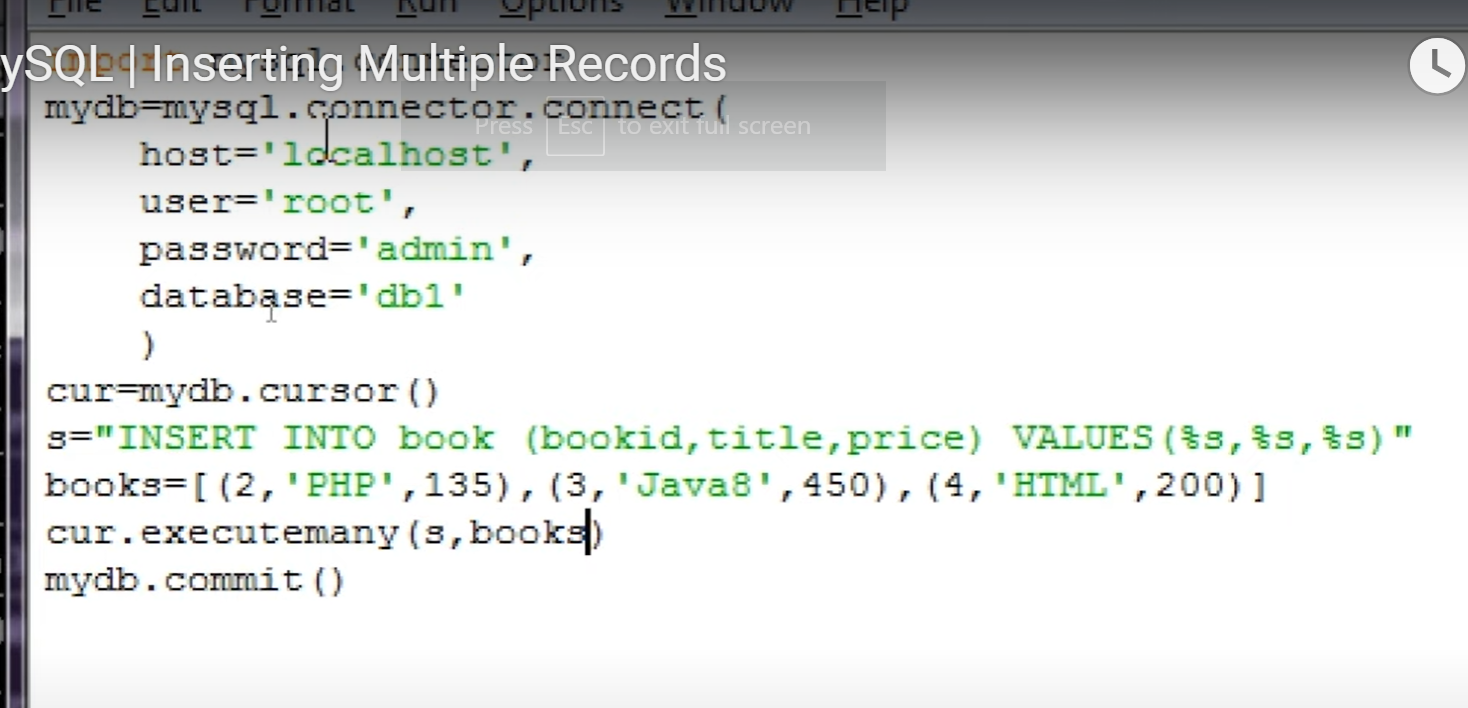
No to insert data into the table



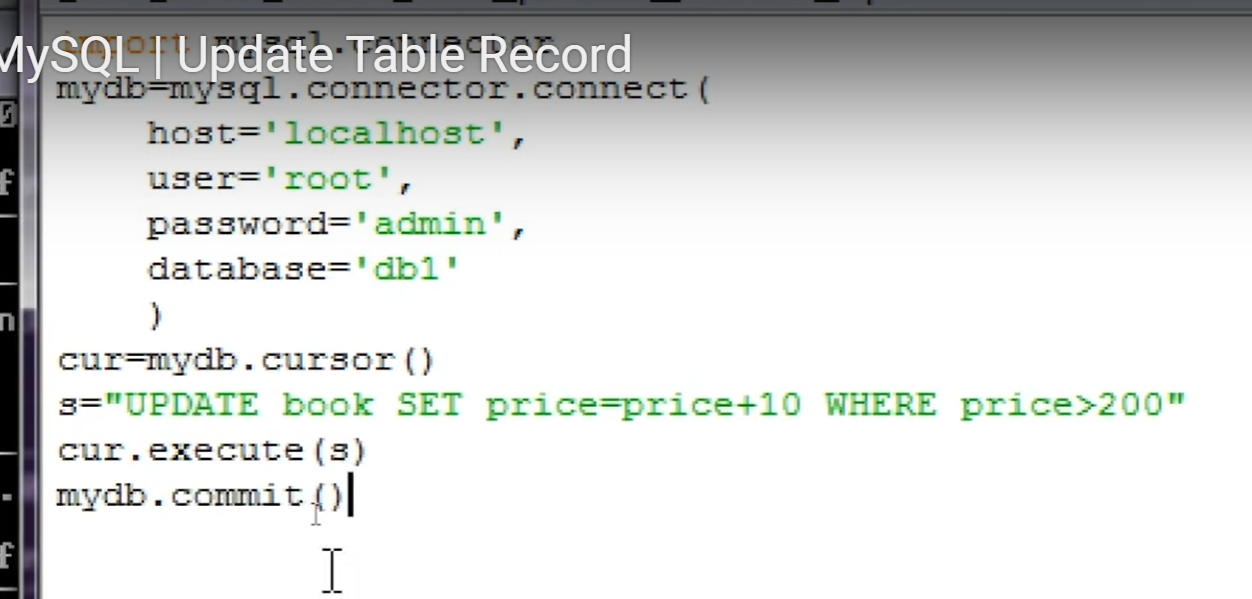
To see entered record



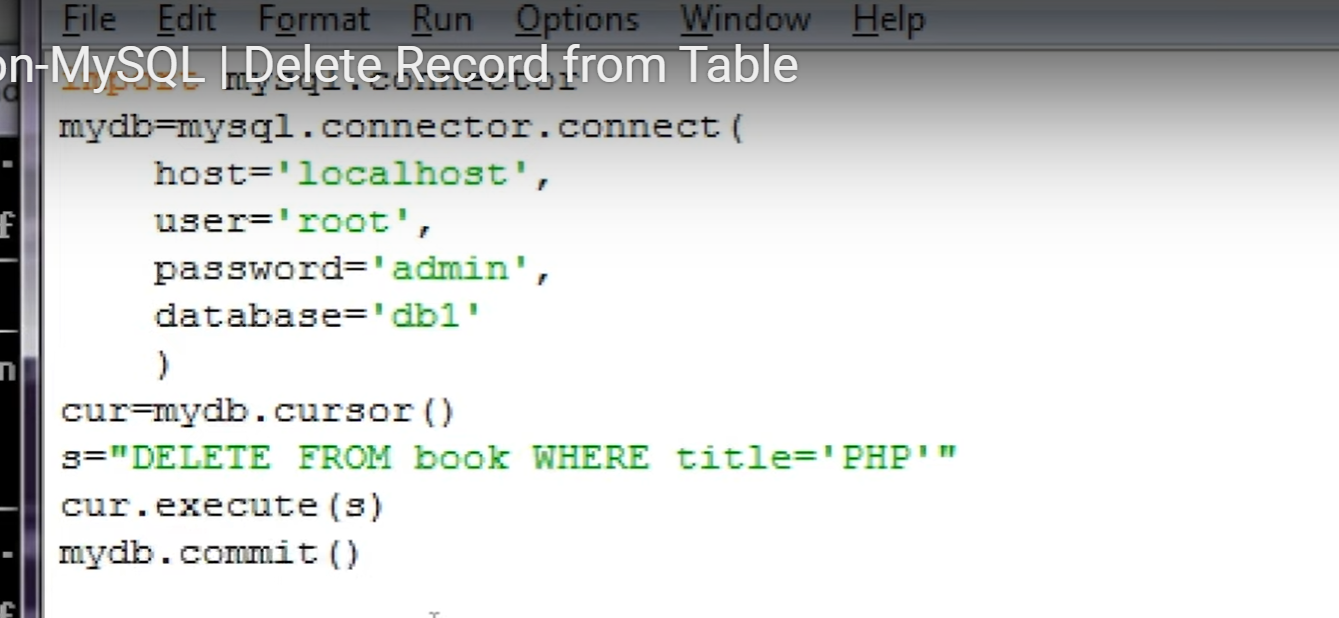
To insert Multiple Records into table



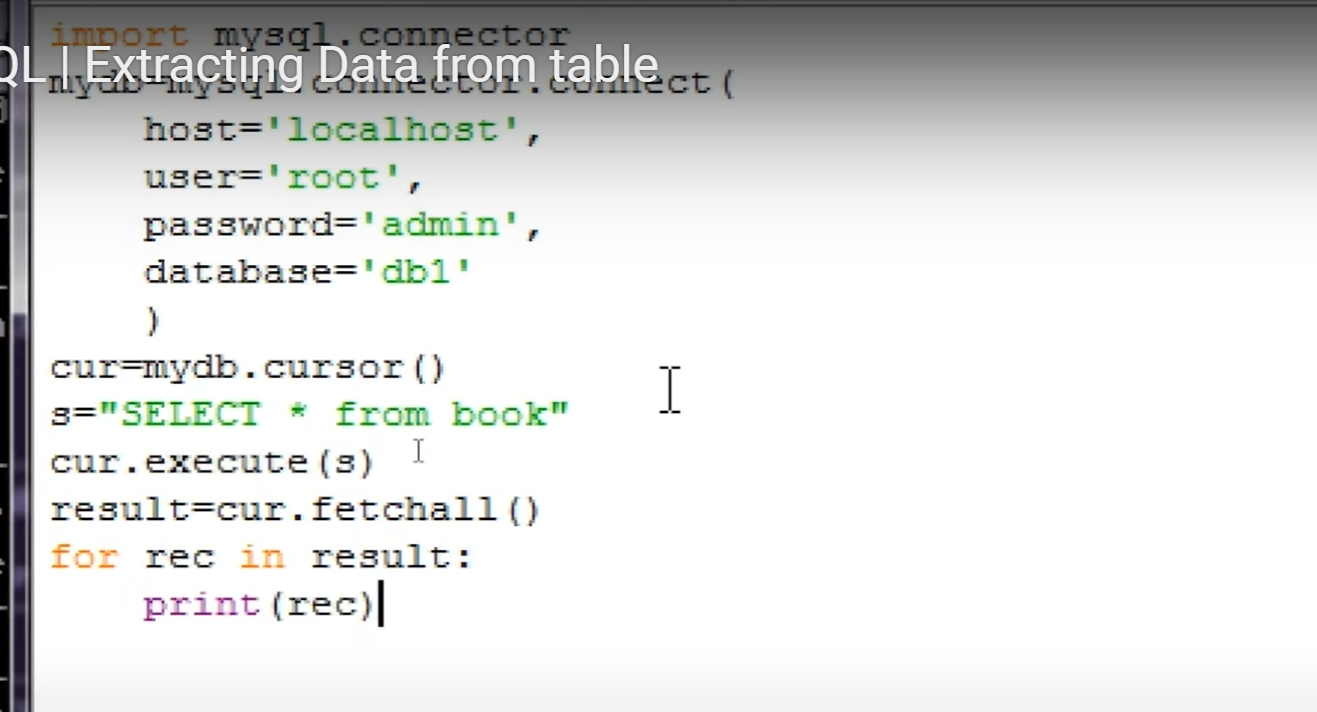
To Update existing records from the table



To delete existing record/s from the table



Reading and displaying records from the table and displaying on the screen



Python MySQL

Python can be used in database applications. One of the most popular databases is MySQL.

## MySQL Database

Firstly you should have MySQL installed on your computer. You can download a MySQL database at [*https://www.mysql.com/downloads/*](https://www.mysql.com/downloads/). From this link and install it on your computer

## Install MySQL Driver

Python needs a MySQL driver to access the MySQL database.

In this tutorial 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. Navigate your command line to the location of PIP, and type the following:

Download and install "MySQL Connector":

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

## 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",#(bydefault root is the user)  
  password="yourpassword" # Password is admin or which you have given   
)  
print(mydb)

## Creating a Database

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

create a database named "mydatabase":

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

### Example

Try connecting to the database "mydatabase":

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

## 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

Create a table named "customers":

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))")

## 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",  
  password="yourpassword",  
  database="mydatabase"  
)  
mycursor = mydb.cursor()  
mycursor.execute("CREATE TABLE customers (id INT AUTO\_INCREMENT PRIMARY KEY, name VARCHAR(255), address VARCHAR(255))")

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("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. 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.")

## 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",  
  password="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.")

## Select From a Table

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

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

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",  
  password="yourpassword",  
  database="mydatabase"  
)  
mycursor = mydb.cursor()  
mycursor.execute("SELECT name, address FROM customers")  
myresult = mycursor.fetchall()  
for x in myresult:  
  print(x)

## Select With a Filter

When selecting records from a table, you can filter the selection by using the "WHERE" statement:

Select record(s) where the address is "Park Lane 38": result:

import mysql.connector  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="*yourusername*",  
  password="*yourpassword*",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "SELECT \* FROM customers WHERE address ='Park Lane 38'"  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
for x in myresult:  
  print(x)

## Sort the Result

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. Sort the result alphabetically by name: result:

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"

or sql = "SELECT \* FROM customers ORDER BY name Desc”  
mycursor.execute(sql)  
myresult = mycursor.fetchall()  
for x in myresult:  
  print(x)

## Delete Record

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

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")

The mysql.connector module uses the placeholder %s to escape values in the delete statement:

### Example

Escape values by using the placeholder %s method:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "DELETE FROM customers WHERE address = %s"  
adr = ("Yellow Garden 2", )  
mycursor.execute(sql, adr)  
mydb.commit()  
print(mycursor.rowcount, "record(s) deleted")

## Delete a Table

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

Delete the table "customers":

import mysql.connector  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="*yourusername*",  
  password="*yourpassword*",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
sql = "DROP TABLE customers"  
or sql = "DROP TABLE IF EXISTS customers"  
mycursor.execute(sql)

## Update Table

You can update existing records in a table by using the "UPDATE" statement:

Overwrite the address column from "Valley 345" to "Canyon 123":

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="*yourusername*",  
  password="*yourpassword*",  
  database="mydatabase"  
)  
mycursor = mydb.cursor()  
sql = "UPDATE customers SET address = 'Canyon 123' WHERE address = 'Valley 345'"  
mycursor.execute(sql)  
mydb.commit()  
print(mycursor.rowcount, "record(s) affected")

Escape values by using the placeholder %s method:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="*yourusername*",  
  password="*yourpassword*",  
  database="mydatabase"  
)  
mycursor = mydb.cursor()  
sql = "UPDATE customers SET address = %s WHERE address = %s"  
val = ("Valley 345", "Canyon 123")  
mycursor.execute(sql, val)  
mydb.commit()  
print(mycursor.rowcount, "record(s) affected")

## Limit the Result

You can limit the number of records returned from the query, by using the "LIMIT" statement:

### Example

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
mycursor = mydb.cursor()  
mycursor.execute("SELECT \* FROM customers LIMIT 5")  
myresult = mycursor.fetchall()  
for x in myresult:  
  print(x)

## Join Two or More Tables

You can combine rows from two or more tables, based on a related column between them, by using a JOIN statement.Consider you have a "users" table and a "products" table:

{ id: 1, name: 'John', fav: 154},  
{ id: 2, name: 'Peter', fav: 154},  
{ id: 3, name: 'Amy', fav: 155},  
{ id: 4, name: 'Hannah', fav:},  
{ id: 5, name: 'Michael', fav:}

### products

{ id: 154, name: 'Chocolate Heaven' },  
{ id: 155, name: 'Tasty Lemons' },  
{ id: 156, name: 'Vanilla Dreams' }

These two tables can be combined by using users' fav field and products' id field.

### Example

Join users and products to see the name of the users favorite product:

import mysql.connector  
  
mydb = mysql.connector.connect(  
  host="localhost",  
  user="yourusername",  
  password="yourpassword",  
  database="mydatabase"  
)  
  
mycursor = mydb.cursor()  
  
sql = "SELECT \  
  users.name AS user, \  
  products.name AS favorite \  
  FROM users \  
  INNER JOIN products ON users.fav = products.id"  
  
mycursor.execute(sql)  
  
myresult = mycursor.fetchall()  
  
for x in myresult:  
  print(x)

## LEFT JOIN

In the example above, Hannah, and Michael were excluded from the result, that is because INNER JOIN only shows the records where there is a match.

If you want to show all users, even if they do not have a favorite product, use the LEFT JOIN statement:

### Example

Select all users and their favorite product:

sql = "SELECT \  
  users.name AS user, \  
  products.name AS favorite \  
  FROM users \  
  LEFT JOIN products ON users.fav = products.id"

## RIGHT JOIN

If you want to return all products, and the users who have them as their favorite, even if no user have them as their favorite, use the RIGHT JOIN statement:

### Example

Select all products, and the user(s) who have them as their favorite:

sql = "SELECT \  
  users.name AS user, \  
  products.name AS favorite \  
  FROM users \  
  RIGHT JOIN products ON users.fav = products.id"