BCA – 502: Python Programming

Rahul Kumar Singh

In today's Class we have discussed on SQL Database connection using python.

Creating the 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:-

In this example, we will create the new table **Employee**. We have to mention the database name while establishing the connection object.

We can create the new table by using the **CREATE TABLE** statement of SQL. In our database **PythonDB**, the table Employee will have the four columns, i.e., name, id, salary, and department_id initially.

The following query is used to create the new table Employee.

```
> create table Employee (name varchar(20) not null, id int primary key, salary float not null, Dept_Id int not null)
```

```
import mysql.connector

#Create the connection object

myconn = mysql.connector.connect(

host = "localhost",

user = "root",

passwd = "admin",

database = "PythonDB"

)

#creating the cursor object

cur = myconn.cursor()

try:

   #Creating a table

   dbs = cur.execute("create table Employee(name varchar(20) not null, id int(20) not null primary key, salary float not null, Dept_id int not null)")

except:

   myconn.rollback()

myconn.close()
```

Output:-

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:

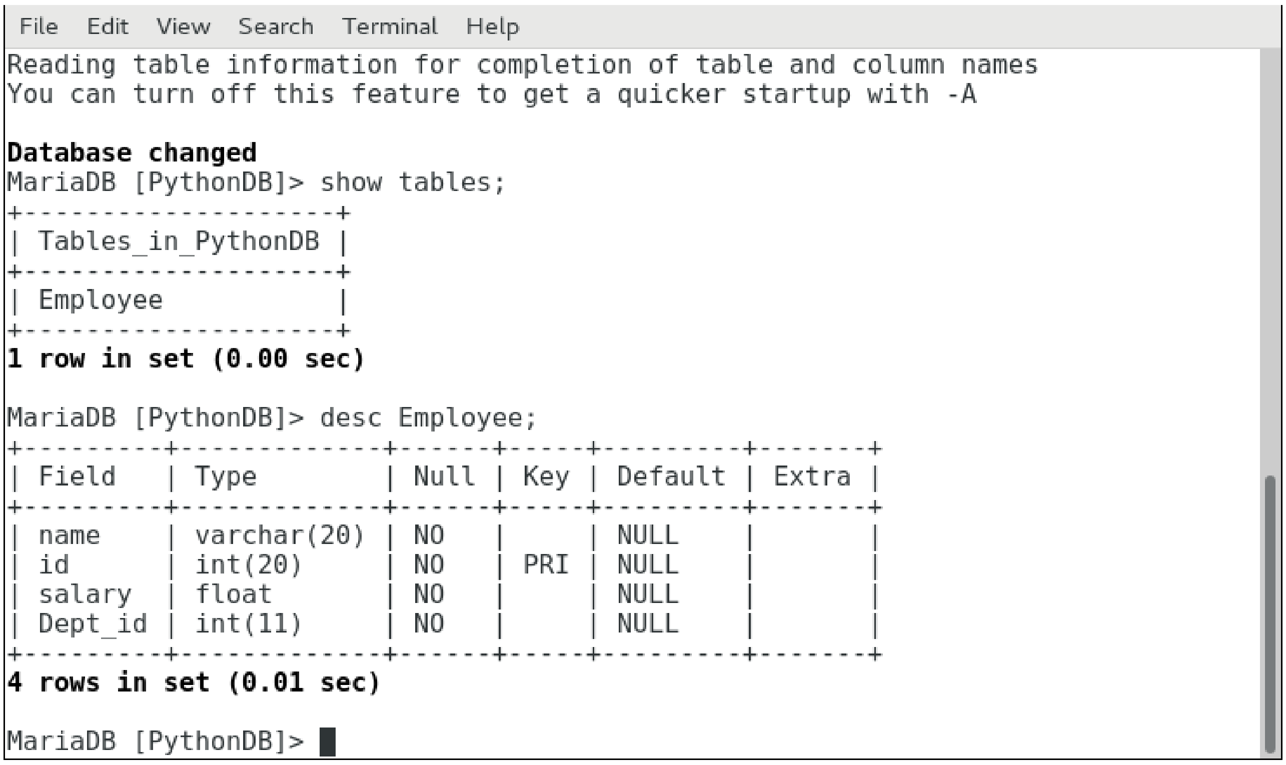
Example:-

Return a list of your system's databases:

```
import mysql.connector
myconn = mysql.connector.connect(
host = "localhost",
user = "root",
passwd = "admin",
database = "PythonDB"
)
mycursor = mydb.cursor()
mycursor.execute("SHOW TABLES")
for x in mycursor:
  print(x)
```

Output:-

```
Employee
```



```
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [PythonDB]> show tables;
+--------------------+
| Tables_in_PythonDB |
+--------------------+
| Employee           |
+--------------------+
1 row in set (0.00 sec)

MariaDB [PythonDB]> desc Employee;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | NO   |     | NULL    |       |
| id      | int(20)     | NO   | PRI | NULL    |       |
| salary  | float       | NO   |     | NULL    |       |
| Dept_id | int(11)     | NO   |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
4 rows in set (0.01 sec)

MariaDB [PythonDB]> 
```

Alter Table:-

Sometimes, we may forget to create some columns, or we may need to update the table schema. The alter statement used to alter the table schema if required.

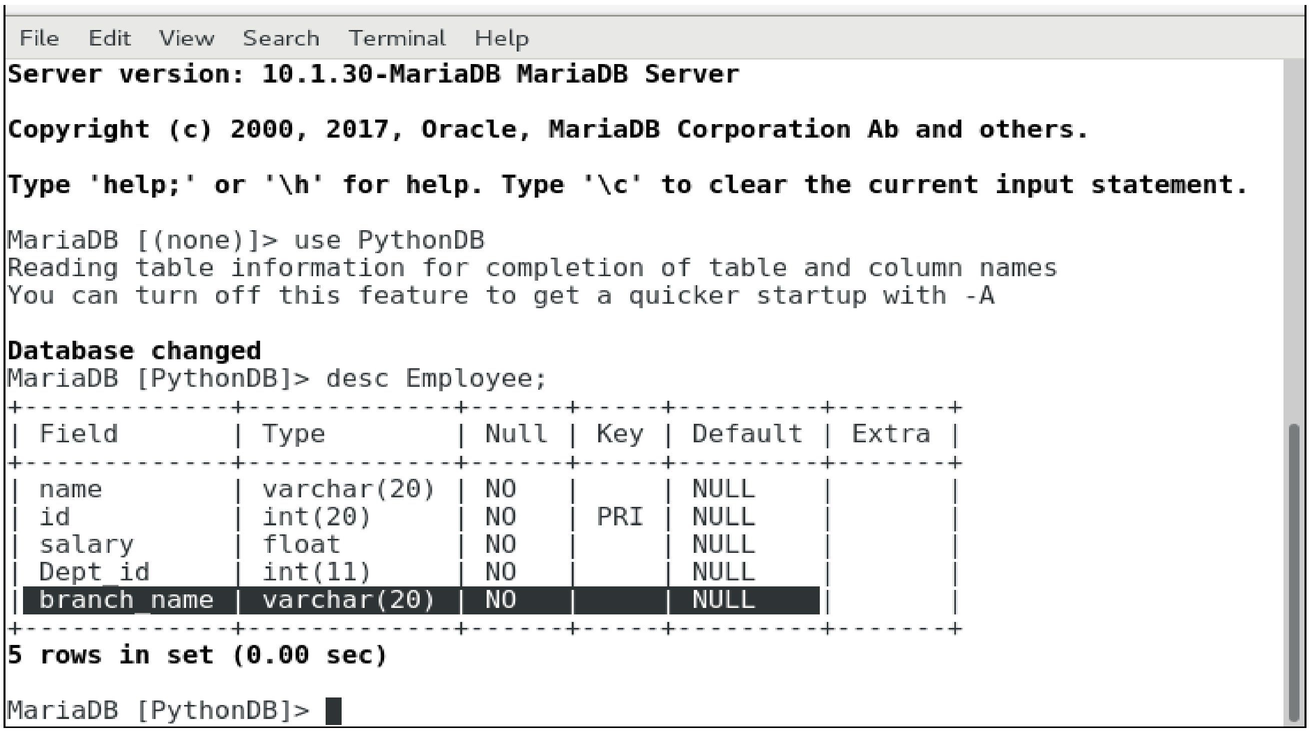
Here, we will add the column branch_name to the table Employee. The following SQL query is used for this purpose.

```
alter table Employee add branch_name varchar(20) not null
```

Example:-

```
import mysql.connector
#Create the connection object
myconn = mysql.connector.connect(
host = "localhost",
user = "root",
passwd = "admin",
database = "PythonDB"
)
#creating the cursor object
cur = myconn.cursor()
try:
    #adding a column branch name to the table Employee
    cur.execute("alter table Employee add branch_name varchar(20) not null")
except:
    myconn.rollback()
myconn.close()
```

Output:-



```
Server version: 10.1.30-MariaDB MariaDB Server

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use PythonDB
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [PythonDB]> desc Employee;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| name        | varchar(20) | NO   |     | NULL    |       |
| id          | int(20)     | NO   | PRI | NULL    |       |
| salary      | float       | NO   |     | NULL    |       |
| Dept_id     | int(11)     | NO   |     | NULL    |       |
| branch_name | varchar(20) | NO   |     | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)

MariaDB [PythonDB]> 
```

Insert Operation:-

Adding a record to the table:-

The **INSERT INTO** statement is used to add a record to the table. In python, we can mention the format specifier (%s) in place of values.

We provide the actual values in the form of tuple in the execute() method of the cursor.

Example:-

import mysql.connector

#Create the connection object

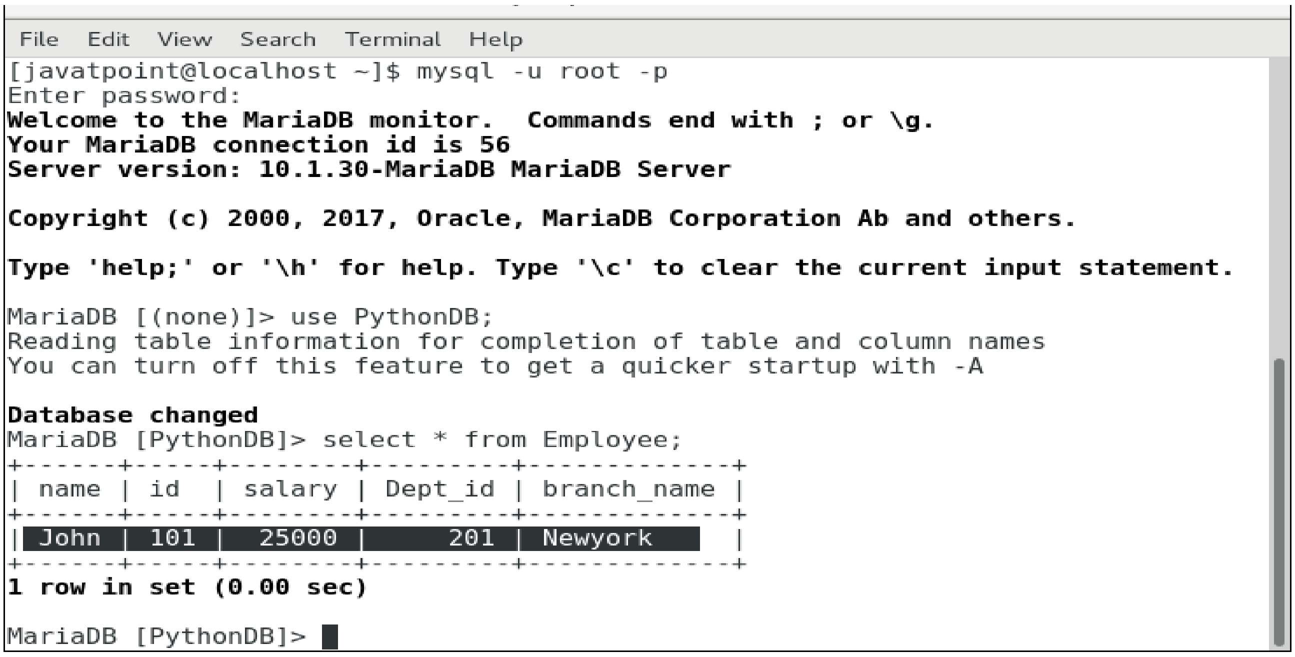
myconn = mysql.connector.connect(

host = "localhost",

```
user = "root",

passwd = "admin",

database = "PythonDB"

)

#creating the cursor object

cur = myconn.cursor()

sql = "insert into Employee(name, id, salary, dept_id, branch_name) values (%s, %s, %s, %s, %s)"

#The row values are provided in the form of tuple

val = ("Ram", 110, 25000.00, 201, "India")

try:

    #inserting the values into the table

    cur.execute(sql,val)

    #commit the transaction

    myconn.commit()

except:

    myconn.rollback()

print(cur.rowcount,"record inserted!")

myconn.close()
```

Output:-

```
1 record inserted!
```



```
[javatpoint@localhost ~]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 56
Server version: 10.1.30-MariaDB MariaDB Server

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use PythonDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [PythonDB]> select * from Employee;
+------+-----+--------+---------+-------------+
| name | id  | salary | Dept_id | branch_name |
+------+-----+--------+---------+-------------+
| John | 101 |  25000 |     201 | Newyork     |
+------+-----+--------+---------+-------------+
1 row in set (0.00 sec)

MariaDB [PythonDB]>
```

Insert multiple rows:-

We can also insert multiple rows at once using the python script. The multiple rows are mentioned as the list of various tuples.

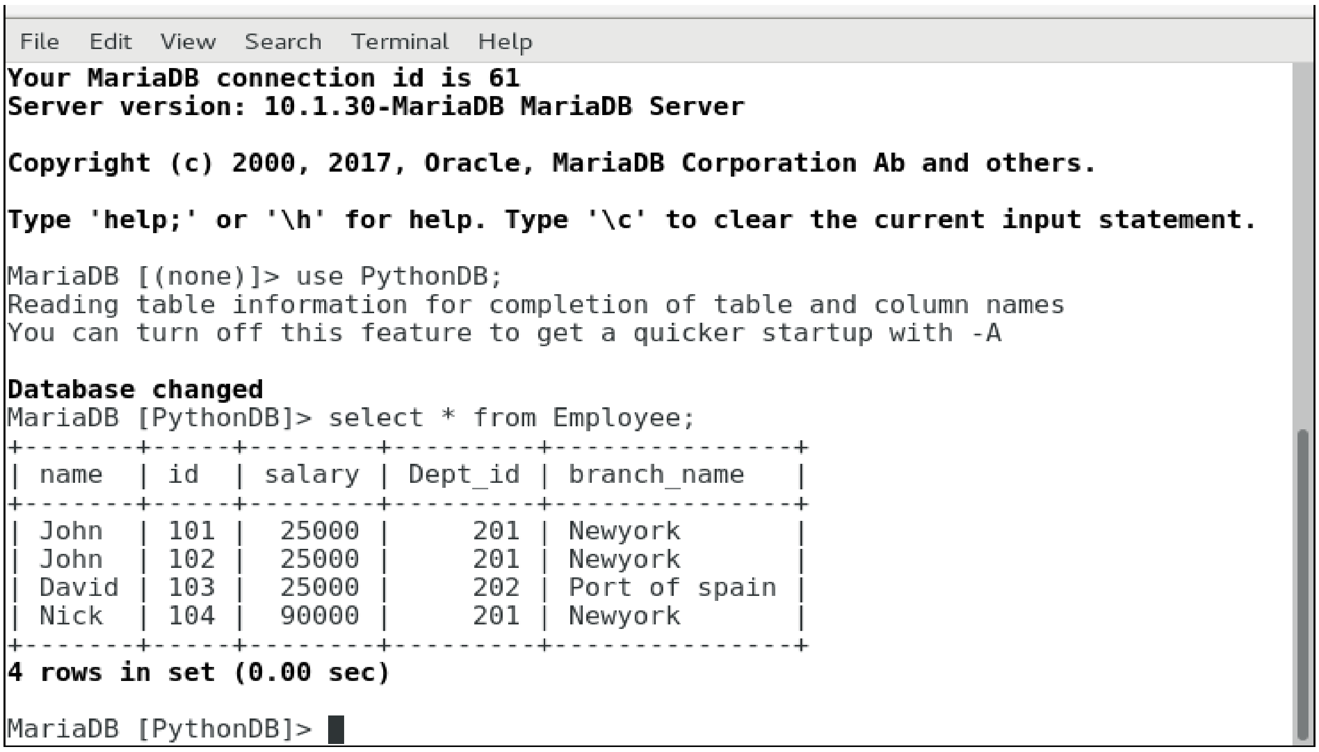
Each element of the list is treated as one particular row, whereas each element of the tuple is treated as one particular column value (attribute).

Example:-

```
import mysql.connector

#Create the connection object

myconn = mysql.connector.connect(

host = "localhost",

user = "root",
passwd = "admin",

database = "PythonDB"

)

#creating the cursor object

cur = myconn.cursor()

sql = "insert into Employee(name, id, salary, dept_id,
branch_name) values (%s, %s, %s, %s, %s)"

val = [("Ram", 102, 25000.00, 201,
"India"),("Shyam",103,25000.00,202,"India"),("Mohan",104,90
000.00,201,"India")]

try:

    #inserting the values into the table

    cur.executemany(sql,val)

    #commit the transaction

    myconn.commit()

    print(cur.rowcount,"records inserted!")

except:

    myconn.rollback()

myconn.close()
```

Output:-

```
3 records inserted!
```



```
Your MariaDB connection id is 61
Server version: 10.1.30-MariaDB MariaDB Server

Copyright (c) 2000, 2017, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> use PythonDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [PythonDB]> select * from Employee;
+-------+-----+--------+---------+---------------+
| name  | id  | salary | Dept_id | branch_name   |
+-------+-----+--------+---------+---------------+
| John  | 101 |  25000 |     201 | Newyork       |
| John  | 102 |  25000 |     201 | Newyork       |
| David | 103 |  25000 |     202 | Port of spain |
| Nick  | 104 |  90000 |     201 | Newyork       |
+-------+-----+--------+---------+---------------+
4 rows in set (0.00 sec)

MariaDB [PythonDB]> 
```

Row ID:-

In SQL, a particular row is represented by an insertion id which is known as row id. We can get the last inserted row id by using the attribute lastrowid of the cursor object.

Example:-

```
import mysql.connector

#Create the connection object

myconn = mysql.connector.connect(

host = "localhost",

user = "root",

passwd = "admin",

database = "PythonDB"
)
#creating the cursor object
cur = myconn.cursor()
sql = "insert into Employee(name, id, salary, dept_id, branch_name) values (%s, %s, %s, %s, %s)"
val = ("Sohan",105,28000,202,"India")
try:
    #inserting the values into the table
    cur.execute(sql,val)
    #commit the transaction
    myconn.commit()
    #getting rowid
    print(cur.rowcount,"
record inserted! id:",cur.lastrowid
)
except:
    myconn.rollback()
myconn.close()
```

Output:-

```
1 record inserted! Id: 0
```