Connecting to a MySQL Database

How to connect to a MySQL database

Connecting a MySQL database is slightly more involved than the process for connecting to a SQLite database. In order to connect to a MySQL database, a **connection string** must be created that sets certain parameters that establishes how Python (and pyodbc) should connect to the database server. Specifically, the following parameters must be set to establish a working connection string:

- **Login Prompt** Defines whether or not the user should be prompted with a login in order to connect to the database.
- **DRIVER** Defines the ODBC driver used to connect to the MySQL database server.
- **SERVER** Defines the location of the MySQL server.
- **DATABASE** Defines the default database that we want to connect to on the server.
- **UID** Sets the username used to log into the database.
- **PWD** Sets the password used to log into the database.

Connecting to a MySQL Database

How to connect to a MySQL database

A connection string can be constructed using generic parameters as follows:

```
conn = pyodbc.connect('Login Prompt=False;DRIVER={Devart ODBC Driver for
MySQL};SERVER=localhost;DATABASE=vc-helpdesk;UID=root;PWD=;')
```

Here, the connect() method of the pyodbc module is used to connect to the vc-helpdesk database on localhost using the Devart ODBC Driver for MySQL. We use the default root username with an empty password to connect. The result will be an object variable called conn that we can use to perform CRUD operations with.

Connecting to a MySQL Database

How to connect to a MySQL database

Another option is to define variables for all of your parameters and then construct the connection string using those variables as follows:

```
server = "localhost"
database = "vc-helpdesk"
username = "root"
password = ""
conn = pyodbc.connect("Login Prompt=False;DRIVER={Devart ODBC Driver for MySQL};SERVER=" +
server + ";DATABASE=" + database + ";UID=" + username + ";PWD=" + password)
```

In this case, if changes to the server name, database, username, or password needed to be made quickly, you can make them easily by simply changing the variable values without fumbling through a long connection string.

Connecting to a MySQL Database

How to close an open MySQL database connection

Once you've opened a database connection, you'll perform whatever tasks you need to perform. Ultimately though, you'll need to close the database connection so that it doesn't remain open and consume valuable application resources. To properly close a database connection, you begin by checking that the conn object exists. if it does, then you call the close() method of the connection object to close the connection as follows:

```
if conn:
    conn.close()
```