Connecting to a SQL Server Database

How to connect to a SQL Server database

Connecting to a SQL Server database is slightly more involved than the process for connecting to a SQLite database. In order to connect to a SQL Server 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:

- **DRIVER** Defines the ODBC driver used to connect to the database server.
- **SERVER** Defines the location of the 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 SQL Server Database

How to connect to a SQL Server database

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

```
conn = pyodbc.connect("DRIVER={ODBC Driver 17 for SQL Server};SERVER=
webdb,1400;DATABASE=vc-helpdesk;UID=login_extracts;PWD=XXXXXXX;")
```

Here, the connect() method of the pyodbc module is used to connect to the vc-helpdesk database on webdb,1400 using the ODBC Driver 17 for SQL Server. We use the login_extracts username with a specific password to connect. The result will be an object variable called conn that we can use to perform CRUD operations with.

Connecting to a SQL Server Database

How to connect to a SQL Server database

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

```
server = "webdb,1400"
database = "vc-helpdesk"
username = "login_extracts"
password = "XXXXXXX"
conn = pyodbc.connect("DRIVER={ODBC Driver 17 for SQL Server};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 SQL Server Database

How to close an open SQL Server 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()
```