The `PyMySQL` Package

Reference:

- Package
- Docs
- Source Repo
- Example Usage

The `PyMySQL` package provides a way for Python to interface with MySQL databases.

Run a `PyMySQL` application "in development" using a database server on your local machine, and/or "in production" using a remote database server hosted by a provider like Heroku. If you run it in development, you should be able to connect via localhost, whereas if you run it in production, you should be able to connect using the production server's credentials. The professor recommends using Sequel Pro as a GUI interface to your MySQL databases, local or remote.

Installation

As a prerequisite: install MySQL on your local machine, if necessary. If you are on a Mac, use Homebrew: `brew install mysql` and follow the post-installation instructions. Make sure you can connect to your local MySQL installation via a GUI or command-line interface. If attempting to connect from the command-line, try running `mysql -uroot` or `mysql -uroot -p`, depending on whether or not your "root" user has a password. Note the username and password you are using to connect.

After demonstrating your ability to connect to a local MySQL installation, install `PyMySQL`, if necessary:

```
pip install PyMySQL
```

Usage

Place the following contents inside a new Python script:

```
import os
import pymysql

MYSQL_ROOT_USER_PASSWORD = os.environ["MYSQL_ROOT_USER_PASSWORD"] # store your password as an environment variable to keep is secret!

# OPEN DATABASE CONNECTION

connection = pymysql.connect(
  host='localhost',
  port=3306,
  user='root', # or perhaps a different MySQL username, depending on your installation
  passwd=MYSQL_ROOT_USER_PASSWORD
)

# PERFORM A DATABASE OPERATION

try:
    with connection.cursor() as cursor:
        my_query = "SELECT host, user, select_priv, super_priv FROM mysql.user" # a.k.a "what local database users exist?". NOTE: `mysql.user` is a built-in table, but you can execute any SQL here instead.

        cursor.execute(my_query)

        print("host, user, select_priv, super_priv")
        print("------------------------------------")
        for row in cursor.fetchall():
           print(row)
finally:
    # CLOSE DATABASE CONNECTION
    connection.close()
```

Finally, run the script to see the results of your SQL query output into the terminal. Oh yea!

Now that you know how to use Python to execute a SQL query, practice using Python to manage databases and tables, then populate them and query them.