11. Python SQLAlchemy

3ikakke

Friday, March 1st, 2022

Outline

- Objectives
- Relational Database Management Systems (RDBMS)
- Requirements for connection
- from sqlalchemy import create_engine
- The single most important query!
- References
- Conclusion
- Review of objectives
- Q&A
- Gist of the day

Objectives

- Understand how to connect to and retrieve data from a relational database management system

Relational Database Management Systems (RDBMS)

- These are database stores that can be queried using structured query language (SQL)
- They are the most popular large data stores along with NoSQL
- Proprietary RDBMS engines include:
  - Microsoft SQL (MSSQL)
  - Oracle
- Free and open source RDBMS engines inlcude:
  - MySQL
  - PostgreSQL
  - SQLite3

Requirements for connection

- Except for SQLite3 which is a database engine in a file the others usually require that you have some detail about them before you can connect

- Connection parameters include:
  - The database engine type (example MySQL, Oracle etc)

  - The server that the database is hosted on. This may be an IP address

  - The port to listen on the server for a connection (example 3306 for MySQL, 1433 for Microsoft SQL, 5432 for PostgreSQL, and sometimes there may be custom port numbers)

  - The database name

  - The database username

  - The database password

from sqlalchemy import create_engine

- Python has the SQLAlchemy module that offers a generic module for connecting to virtually andy RDBMS.

- The SQLAlchemy module comes with a method that allows for the creation of an active connection and its called create_engine

- Let's see how we may achieve this connection

```
from sqlalchemy import create_engine
import pandas as pd

rdbms = 'postgresql'
server = '127.0.0.1'
port = 5432
database = 'musigma'
username = 'admin'
password = 'shokolokobangoshe'

engine = create_engine(f"{rdbms}://{username}:{password}@{server}:{port}/{database}")
```

The single most important query!

- Next stop is to get the actual data fromt the database.

- Assuming the database name = 'students'

```
dataset = pd.read_sql("SELECT * FROM students", engine)
```

References

- Pandas getting data using read_sql

- Connecting to MySQL
- Connecting to PostgreSQL
- Connecting to MicrosoftSQL
- Connecting to Oracle

Conclusion

- For the rest, of manipulating the data you can refer to the previous class on Pandas
- Next round of conversations will be on the basics of statistics
- We will be using the Statistics and the SciPy modules along with Pandas
- We will be introducing the Matplotlib and Seaborn modules
- Remember to set them up!

```
pip install scipy
pip install matplotlib
pip install seaborn
#statistics comes pre-installed with python
```

Review of objectives

- Understand how to connect to and retrieve data from a relational database management system

Q&A

Gist of the day

- The gist for the day can be found here
- The pdf version of this class can be found here
- The Jupyter Notebook will be uploaded

Next stop: Statistics!