Python 基础

Day 08. Python 访问数据库

1. SQLite

SQLite

```
#!/usr/bin/env python

import sqlite3

connection = sqlite3.connect('test.db')

cursor = connection.cursor()

cursor.execute('''
    create table if not exists user(
        id int primary  key,
        name varchar(20)
    )
''')

cursor.execute("insert into user(id, name) values(1, 'Tom')")

print(cursor.rowcount)

cursor.execute('select * from user where id = ?', (1,))

values = cursor.fetchall()

print(values)
```

```
cursor.execute('update user set name = ? where id = ?', ('Jerry', 1))

cursor.execute('select * from user where id = ?', (1,))

values = cursor.fetchall()

print(values)

cursor.execute('delete from user where id = ?', (1,))

cursor.execute('select * from user where id = ?', (1,))

values = cursor.fetchall()

print(values)

cursor.close()

connection.close()
```

2. MySQL

Install `mysql-connector`

```
pip install mysql-connector-python
```

Connection

```python
import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
    # host='localhost'
    # database='db_python'
)
```

Create Database

- `cursor` 游标

```python
import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
    # host='localhost'
    # database='db_python'
)

cursor = connection.cursor()

cursor.execute('drop database if exists db_python')

cursor.execute('create database db_python')

cursor.execute('show databases')

for db in cursor:
    print(db)
```

Create Table

```
#!/usr/bin/env python

import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()

cursor.execute('drop table if exists db_python.user')

cursor.execute('''
    create table db_python.user(
      id int auto_increment primary key comment 'id PK',
      email varchar(255) not null comment 'email NN',
      password varchar(255) not null comment 'password NN'
    ) comment 'user table'
''')

cursor.execute('drop table if exists db_python.book')

cursor.execute('''
    create table db_python.book(
      id int auto_increment primary key comment 'id PK',
      title varchar(255) not null comment 'title NN',
      author varchar(255) not null comment 'author NN',
        userId int comment 'user id FK'
    ) comment 'book table'
''')

cursor.execute('''
    alter table db_python.book
    add constraint
    book_fk_userId
    foreign key (userId)
    references db_python.user(id)
''')

cursor.execute('show tables from db_python')

for table in cursor:
    print(table)
```

Insert

```
#!/usr/bin/env python

import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()

cursor.execute('''
    insert into db_python.user(email,password)
    values
        ('tom@web.com','123'),
        ('jerry@web.com','456')
''')

print(cursor.rowcount)

sql = 'insert into db_python.user(email, password) values(%s, %s)'
val = ('spike@web.com', '789')

cursor.execute(sql, val)

print(cursor.rowcount)

cursor.execute('''
    insert into db_python.book(title, author, userId)
    values
        ('HTML','author-1', 1),
        ('CSS','author-2', 2),
        ('JavaScript','author-3', 2),
        ('MyBatis','author-4', 3),
        ('Spring','author-5', 3),
        ('Python 编程基础','author-6', 3)
''')

print(cursor.rowcount)

connection.commit()
```

Select

```python
#!/usr/bin/env python

import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()

cursor.execute('select * from db_python.user')

rows = cursor.fetchall()

for row in rows:
    print(row)

print('-----------------')

cursor.execute('''
  select * from db_python.book
  where id < 10
  order by title desc
  limit 5 offset 0
''')

rows = cursor.fetchall()

for row in rows:
    print(row)
```

Join

```
#!/usr/bin/env python

import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()

cursor.execute('''
  select u.email, b.title from
  db_python.user u inner join db_python.book b
  on u.id = b.userId
  # where u.email = 'spike@web.com'
''')

rows = cursor.fetchall()

for row in rows:
    print(row)
```

Update

```
#!/usr/bin/env python

import mysql.connector
```

```python
connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()

cursor.execute('''
  update db_python.book
  set title = 'JavaScript 高级编程'
  where title = 'JavaScript'
''')

print(cursor.rowcount)

connection.commit()
```

Delete

```python
#!/usr/bin/env python

import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()
```

```
###
cursor.execute('alter table db_python.book drop foreign key book_fk_userId')

cursor.execute('''
    alter table db_python.book
    add constraint
    book_fk_userId
    foreign key (userId)
    references db_python.user(id)
    on delete set null
''')
###

sql = 'delete from db_python.user where id = %s'

val = ('3',)

cursor.execute(sql, val)

print(cursor.rowcount)

connection.commit()
```

Drop Table

```
#!/usr/bin/env python

import mysql.connector

connection = mysql.connector.connect(
    user='root',
    password='system'
)

cursor = connection.cursor()

# cursor.execute('drop table db_python.user')
# cursor.execute('drop table db_python.book')

cursor.execute('show tables from db_python')

for table in cursor:
    print(table)
```

3. SQLAlchemy

`['ælkɪmɪ]`

ORM

- Object-Relational Mapping

Install `sqlalchemy`

```
pip install sqlalchemy
```

O-R 映射

```python
#!/usr/bin/env python

from sqlalchemy import Column, Integer, String
from sqlalchemy.ext.declarative import declarative_base

# Base Class
Base = declarative_base()


# User class
class User(Base):
    """ O-R Mapping """

    # class - table mapping
    __tablename__ = 'user'

    # attribute - column mapping
    id = Column(Integer(), autoincrement=True, primary_key=True)
    email = Column(String(255), nullable=False)
    password = Column(String(255), nullable=False)
```

连接数据库

```python
#!/usr/bin/env python

from sqlalchemy import Column, Integer, String, create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base

# Base Class
Base = declarative_base()
```

```python
# User class
class User(Base):
    """ O-R Mapping """

    # class - table mapping
    __tablename__ = 'user'

    # attribute - column mapping
    id = Column(Integer(), autoincrement=True, primary_key=True)
    email = Column(String(255), nullable=False)
    password = Column(String(255), nullable=False)


# database connection
# 数据库类型+数据库驱动名://账号:口令@服务器:端口号/数据库名'
engine = create_engine('mysql+mysqlconnector://root:system@localhost:3306/db_python')


# create DBSession type
DBSession = sessionmaker(bind=engine)
```

DML

```python
#!/usr/bin/env python

from sqlalchemy import Column, Integer, String, create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base

# Base Class
```

```python
Base = declarative_base()


# User class
class User(Base):
    """ O-R Mapping """

    # class - table mapping
    __tablename__ = 'user'

    # attribute - column mapping
    id = Column(Integer(), autoincrement=True, primary_key=True)
    email = Column(String(255), nullable=False)
    password = Column(String(255), nullable=False)


# database connection
# 数据库类型+数据库驱动名://账号:口令@服务器:端口号/数据库名'
engine = create_engine('mysql+mysqlconnector://root:system@localhost:3306/db_python')


# create DBSession type
DBSession = sessionmaker(bind=engine)

spike = User(email='spike@web.com', password='789')

# create DBSession object
session = DBSession()  # session as connection

session.add(spike)

session.commit()

session.close()
```

DQL

```python
#!/usr/bin/env python

from sqlalchemy import Column, Integer, String, create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy.ext.declarative import declarative_base

# Base Class
Base = declarative_base()


# User class
class User(Base):
    """ O-R Mapping """

    # class - table mapping
    __tablename__ = 'user'

    # attribute - column mapping
    id = Column(Integer(), autoincrement=True, primary_key=True)
    email = Column(String(255), nullable=False)
    password = Column(String(255), nullable=False)

    def __str__(self):
        return 'id = ' + str(self.id) + '\nemail = ' + self.email + '\npassword = ' + self.password


# database connection
# 数据库类型+数据库驱动名://账号:口令@服务器:端口号/数据库名'
```

```python
engine = create_engine('mysql+mysqlconnector://root:system@localhost:3306/db_python')

# create DBSession type
DBSession = sessionmaker(bind=engine)

# create DBSession object
session = DBSession()  # session as connection

# query: filter as WHERE; one() for one row; all() from multiple rows
user = session.query(User).filter(User.id == '1').one()

# print(user.id)
# print(user.email)
# print(user.password)

print(user)

session.close()
```

4. 作业

1. 使用 `SQLAlchemy` 实现关联查询 `user - books`

> `books = relationship('Book')`