

# 1.关于Flask数据库

---

Flask中没有指定使用的数据库，可以自由选择不管你是使用关系型数据库，还是非关系型数据库都可以，不像django提供了orm 数据库抽象层，可以直接采用对象的方式操作数据库。

但是为了开发效率，在开发Flask项目中我们一般会选择 SQLAlchemy 来操作数据库，类似于Django的ORM。

SQLAlchemy实际上是对数据库的抽象，让开发者不直接使用sql语句做开发，而是通过Python对象来操作数据库，在舍弃一些性能开销的同时，换来的是开发效率的较大提升。

SQLAlchemy是一个关系型数据库框架，它提供了高层的ORM和底层的原生数据库的操作。

为了简化配置和操作，我们使用 Flask-SQLAlchemy，这个 Flask 扩展封装了 SQLAlchemy 框架。

安装 Flask-SQLAlchemy

```
pip install Flask-SQLAlchemy
```

要连接mysql数据库还需要安装flask-mysqldb

```
pip install flask-mysqldb # 目前不建议使用，特别是windows
```

```
# 推荐使用 pymysql进行连接
```

```
pip install pymysql -i
https://pypi.tuna.tsinghua.edu.cn/simple
```

如果Linux安装报下面错误

```
Command "python setup.py egg_info" failed with error code 1
```

执行下面这条命令：

```
sudo apt-get install mysql-server libmysqld-dev
```

## 2.设置

设置数据库连接：

```
app.config['SQLALCHEMY_DATABASE_URI'] =
'mysql://root:mysql@127.0.0.1:3306/py'
```

### Flask-SQLAlchemy 配置键：

配置键	说明
SQLAL CHEM Y_DAT ABASE _URI	设置连接数据库 mysql://username:password@server/db
SQLAL CHEM Y_BIN DS	一个映射绑定 (bind) 键到 SQLAlchemy 连接 URIs 的字典。更多的信息请参阅 绑定多个数据库。

配置键	说明
SQLALCHEMY_ECHO	如果设置成 True, SQLAlchemy 将会记录所有 发到标准输出(stderr)的语句, 这对调试很有帮助。
SQLALCHEMY_RECORD_QUERIES	可以用于显式地禁用或者启用查询记录。查询记录 在调试或者测试模式下自动启用。更多信息请参阅get_debug_queries()。
SQLALCHEMY_NATIVE_UNICODE	可以用于显式地禁用支持原生的 unicode。这是 某些数据库适配器必须的 (像在 Ubuntu 某些版本上的 PostgreSQL), 当使用不合适的指定无编码的数据库 默认值时。
SQLALCHEMY_POOL_SIZE	数据库连接池的大小。默认是数据库引擎的默认值 (通常是5) 。
SQLALCHEMY_POOL_TIMEOUT	指定数据库连接池的超时时间。默认是 10。

配置键	说明
SQLALCHEMY_POOL_RECYCLE	自动回收连接的秒数。这对 MySQL 是必须的，默认情况下MySQL会自动移除闲置 8 小时或者以上的连接。需要注意的是如果使用 MySQL 的话，Flask-SQLAlchemy 会自动地设置这个值为 2 小时。
SQLALCHEMY_MAX_OVERFLOW	控制在连接池达到最大值后可以创建的连接数。当这些额外的连接回收到连接池后将会被断开和抛弃。
SQLALCHEMY_TRACK_MODIFICATIONS	如果设置成 True (默认情况)，Flask-SQLAlchemy 将会追踪对象的修改并且发送信号。这需要额外的内存，如果不必要的可以禁用它。
SQLALCHEMY_COMMIT_ON_TEARDOWN	每次请求结束后会自动提交数据库中的改动

URI：统一资源标识符，用于标识某一互联网资源名称的字符串

常用数据库的链接URI

数据库	URI
MySQL	mysql://username:password@hostname/database
Postgres	postgresql://username:password@hostname/database
SQLite (Unix)	sqlite:///python/data/database
SQLite (Windows)	sqlite:///c:/db/data/database
Oracle	oracle://scott:tiger@127.0.0.1:1521/sidname

### 字段说明:

username: 登录数据库的用户名

password: 登录数据库的密码

hostname: 服务器主机ip, 可以是本地主机 (localhost) 也可以是远程服务器

database: 表示要使用的数据库

连接mysql在ubuntu中先创建数据库:

先在mysql中创建数据库:

```
# 连接mysql
mysql -uroot -pmysql
# 查看所有数据库
show databases;
```

```
# 创建数据库名为 py
create database py charset=utf8;
```

```
# -*- encoding: utf-8 -*-
from flask import Flask
from flask_sqlalchemy import SQLAlchemy

app = Flask(__name__) # type:Flask

# 设置连接的数据库 注意采用pymysql连接 mysql+pymysql
app.config['SQLALCHEMY_DATABASE_URI'] =
'mysql+pymysql://root:qwe123@127.0.0.1:3306/py'
# 创建数据库对象
db = SQLAlchemy(app)

class User(db.Model):
    """创建User模型类"""
    # SQLAlchemy 需要手动执行主键列，第一个参数是 字段类型，第二个
    参数是约束条件
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(20), unique=True)

if __name__ == '__main__':
    # 创建手动应用上下文（激活） --- 避免异步处理 RuntimeError:
    Working outside of application context
    with app.app_context(): # 应用上下文app context，存储的是应
    用级别的信息，比如数据库连接信息
        # 删除所有表，注意这条是危险命令，会将模型类对应数据库中的
        表物理删除。在实际生产环境下勿用。
        db.drop_all()
        # 创建所有表
        db.create_all()
```



image