School of Economics ang Management, Beihang University

现代程序设计技术

赵吉昌
jichang@buaa.edu.cn

本周内容

- 面向对象编程
  - 数据库连接与操作
  - ORM简介
  - Web开发框架简介

数据库连接与操作

- 数据库
  - 关系数据库
    - PostgreSQL
    - MySQL
  - 非关系数据库
    - MongoDB
- 所需第三方库
  - 关系数据库
    - psycopg2
    - pymysql (建议自学并了解)
  - 非关系数据库
    - pymongo

数据库连接与操作

- psycopg2
  - implementing the DB API 2.0 protocol
  - 一般的使用逻辑 (Demo: pys.py)
    - import psycopg2
    - 创建数据库连接(会话）
      - conn = psycopg2.connect("dbname=test user=postgres password=secret")
    - 创建游标并通过游标执行SQL语句
      - cur = conn.cursor()
    - 执行SQL语句（创建，插入，查询等）
      - cur.execute("CREATE TABLE test (id serial PRIMARY KEY, num integer, data varchar);")
      - cur.execute("INSERT INTO test (num, data) VALUES (%s, %s)",(100, "abc'def"))
      - cur.execute("SELECT * FROM test;")

数据库连接与操作

- psycopg2
  - 通过会话完成事务的提交或回滚
    - conn.commit()
    - conn.rollback()
  - 关闭数据库会话
    - cur.close()
    - conn.close()

数据连接与操作

- psycopg2
  - 利用with语句进行连接和游标的管理
  - with psycopg2.connect(DSN) as conn:
    - with conn.cursor() as curs:
      - SQL)
  - conn = psycopg2.connect(DSN)#多次使用
    - with conn:
  - with conn.cursor() as curs:
  - curs.execute(SQL1)
    - with conn:
  - with conn.cursor() as curs:
  - curs.execute(SQL2)
    - conn.close() #注意退出with上下文时并不关闭连接

数据库连接与操作

- psyconpg2
  - 给SQL语句传参
    - using %s placeholders in the SQL statements
    - cur.execute("""
    - INSERT INTO some_table (an_int, a_date, a_string)
    - VALUES (%s, %s, %s);
    - """,
    - (10, datetime.date(2005, 11, 18), "O'Reilly"))
    - cur.execute("""
      - INSERT INTO some_table (an_int, a_date, another_date, a_string)
      - VALUES (%(int)s, %(date)s, %(date)s, %(str)s); """,
      - {'int': 10, 'str': "O'Reilly", 'date': datetime.date(2005, 11, 18)})

数据库连接与操作

- psyconpg2
  - 给SQL语句传参
    - Python字符串操作的%不能使用
      - cur.execute(“INSERT INTO numbers VALUES (%s, %s)” % (10, 20)) #错误的写法
    - the execute() method accepts a tuple or dictionary of values as second parameter
      - cur.execute(“INSERT INTO foo VALUES (*%s*)” , (“bar”))
    - The placeholder *must not be quoted*
      - cur.execute("INSERT INTO numbers VALUES ('*%s*')", (10,))
      - *must always be a* %s
    - 表名或者列名等不能直接作为参数传入（动态查询）
      - import psycopg2.sql
      - cur.execute(SQL("INSERT INTO {} VALUES (%s)").format(Identifier('numbers')),(10,))

数据库连接与操作

- psyconpg2
  - 结果的获取
    - cursor实例本身可迭代
      - cur.execute("SELECT * FROM test;")
      - for record in cur:
        - » print record
    - fetechone()
      - 返回tuple
    - fetchmany([size])
      - 返回tuple的list，如果无数据即返回[]
    - fetchall()
      - 返回tuple的list

数据库连接与操作

- psyconpg2
  - 以字典形式返回执行结果
    - dict_cur = conn.cursor(cursor_factory=psycopg2.extras.DictCursor)
    - dict_cur.execute("SELECT * FROM test")
    - rec = dict_cur.fetchone()
    - Demo: psy_dict_cur.py

数据库连接与操作

- psyconpg2
  - 批量操作
    - 直接使用cursor.executemany()效率并不佳
    - 建议使用psycopg2.extras.execute_values
    - Demo: psy_batch.py
  - 高级特征
    - 协程支持
      - psycopg2.extras.wait_select(*conn*)
    - 异步IO
      - aiopg

数据库连接与操作

- pymongo
  - 创建连接
    - from pymongo import MongoClient
    - client = MongoClient()
    - client = MongoClient('localhost', 27017)
    - client = MongoClient('mongodb://localhost:27017/’)
  - 指定数据库
    - db = client.test_database
    - db = client['test-database’]
  - 指定集合(collection)
    - collection = db.test_collection
    - collection = db['test-collection’]

数据库连接连接与操作

- pymongo
  - 插入文档
    - insert_one()
  - 批量插入
    - insert_many()
  - Demo: mongo_insert_batch.py

数据库连接与操作

- pymongo
  - returns a single document matching a query (or None if there are no matches)
    - find_one([query])
    - 返回结果为字典
  - To get more than a single document as the result of a query
    - find([query])
    - returns a Cursor instance that can be iterated
    - find().limit(size)#控制返回的数目
  - Demo: mongo_find.py

数据库连接与操作

- 计数
  - 返回满足要求的文档数
    - count_documents({})
  - Demo: mongo_count.py
- 排序
  - 对查询结果进行排序
  - sort(“name”,1) #ascending
  - sort(” name”,-1) #descending
  - Demo: mongo_sort.py

数据库连接与操作

- pymongo
  - 删除文档
    - delete_one(myquery)
    - delete_many(myquery)
    - delete_many({})#删除collection中的所有文档
    - drop()#删除整个collection
  - 更新
    - myquery = { "address": "Valley 345" }
    - newvalues = { "$set": { "address": "Canyon 123" } }
    - update_one(myquery, newvalues)
    - myquery = { "address": { "$regex": "^S" } }
    - newvalues = { "$set": { "name": "Minnie" } }
    - x = mycol.update_many(myquery, newvalues)

ORM简介

- 面向关系数据库的ORM
  - sqlalchemy
  - peewee
  - PonyORM
  - Django ORM
- 一般的逻辑
  - 创建Mapping
    - 业务逻辑中的实体类与数据库的表建立对应关系
  - 构建数据加会话并进行存储或查询
  - Demo: sqla.py

ORM简介

- 面向非关系数据库的ORM
  - Django ORM
  - MongoEngine
  - MongoKit
  - Ming

Web开发框架简介

web framework	Bottle	Flask	Flask	Django
ORM	Peewee	Pony ORM	SQLAlchemy	Django ORM
database connector	psycopg	psycopg	psycopg	psycopg
relational database	PostgreSQL	PostgreSQL	PostgreSQL	PostgreSQL

Web开发框架简介

- Flask
  - Demo: fdemo.py