数据库存取

北京理工大学计算机学院　高玉金

2019年3月

SQL与SQLite3

- SQL是一种声明式语言，它是关系型数据库的通用语言。SQL包含了客户端发送给数据库服务器的文本字符串，指明需要执行的具体操作。SQL语言不区分大小写。
- SQL语句有两种主要的类型：
  - DDL（数据定义语言）， 处理用户、数据库以及表单的创建、删除、约束和权限等
  - DML（数据操作语言）， 处理数据插入、选择、更新和删除
- SQLite是一种轻量级、优秀的开源关系型数据库
- SQLite使用Python的标准库实现，并且把数据库存储在普通文件中
- 这些普通文件在不同机器和操作系统之间是可移植的，使得SQLite成为简易关系型数据库应用的可移植的解决方案
- SQLite仅仅支持原生SQL以及多用户并发操作。
- 浏览器、智能手机和其他应用把SQLite作为嵌入数据库使用

连接对象和游标对象

- 连接对象可以是硬盘上面的数据库文件，也可以是建立在内存中的
  - ✓创建在硬盘： conn = sqlite3.connect('test.db')
  - ✓创建在内存： conn = sqlite3.connect(':memory:')
- 所有sql语句的执行都要在游标对象的参与下完成

commit()	--事务提交
rollback()	--事务回滚
close()	--关闭一个数据库链接
cursor()	--创建一个游标

execute()	--执行一条sql语句
executemany()	--执行多条sql语句
close()	--游标关闭
fetchone()	--从结果中取出一条记录
fetchmany()	--从结果中取出多条记录
fetchall()	--从结果中取出所有记录
scroll()	--游标滚动

数据库SQLite3操作示例

```
from faker import Faker
import sqlite3

fake = Faker("zh_cn")
conn=sqlite3.connect(":memory:")
c=conn.cursor()
c.execute("create table user (id varchar(20) \
        primary key, name varchar(20))")
for i in range(10):
            c.execute("insert into user (id, name) \
                    values ('{}','{}')".format(i,fake.name()))
c.execute("select * from user")
result = c.fetchall()
print(result)
```

```
[('0', '束丹'), ('1', '班冬梅'), ('2', '广玉梅'), ('3', '俞宇'), ('4', '祁健'), ('5', '谈春梅'), ('6', '宰鑫'), ('7', '段倩'), ('8', '石玉华'), ('9', '谯丽丽')]
```

NoSQL数据库

- 随着web的快速发展，非关系型、分布式数据存储得到了快速的发展，它们不保证关系数据的事务特性（ACID原子性，一致性，独立性和持久性）
- NoSQL概念在2009年被提出。NoSQL最常见的解释是“non-relational”或“Not Only SQL”
- NoSQL类型数据中用得最多的是key-value存储，此外还有其他的文档型的、列存储、图型数据库、xml数据库等
- MySQL和NoSQL都有各自的特点和使用的应用场景，两者的紧密结合将会给新一代web的数据库发展带来新的思路
- 关系数据库关注在关系上，NoSQL关注在存储上

常见的NoSQL数据库

类型	部分代表	特点
列存储	Hbase Cassandra Hypertable	顾名思义，是按列存储数据的。最大的特点是方便存储结构化和半结构化数据，方便做数据压缩，对针对某一列或者某几列的查询有非常大的IO优势。
文档存储	MongoDB CouchDB	文档存储一般用类似json的格式存储，存储的内容是文档型的。这样也就有机会对某些字段建立索引，实现关系数据库的某些功能。
key-value存储	Tokyo Cabinet / Tyrant Berkeley DB MemcacheDB Redis	可以通过key快速查询到其value。一般来说，存储不管value的格式，照单全收。（Redis包含了其他功能）
图存储	Neo4J FlockDB	图形关系的最佳存储。使用传统关系数据库来解决的话性能低下，而且设计使用不方便。
对象存储	db4o Versant	通过类似面向对象语言的语法操作数据库，通过对象的方式存取数据。
xml数据库	Berkeley DB XML BaseX	高效的存储XML数据，并支持XML的内部查询语法，比如XQuery,Xpath,

MongoDB

- MongoDB 是一个基于分布式文件存储的数据库，由 C++ 语言编写
- MongoDB旨在为 WEB 应用提供可扩展的高性能数据存储解决方案
- MongoDB 是一个介于关系数据库和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的
- MongoDB提出的是文档、集合的概念，使用BSON（类JSON）作为其数据模型结构，其结构是面向对象的而不是二维表
- 使用MongoDB，无需掌握SQL语言，直接使用面向对象的方式，按照类似Json的方式进行存取

MongoDB简单示例

pip install pymongo，确保MongoDB已经安装且可以正常运行

```
from pymongo import MongoClient

conn = MongoClient('192.168.0.113', 27017)
db = conn.mydb  #连接mydb数据库，没有则自动创建
my_set = db.test_set    #使用test_set集合，没有则自动创建
```

```
my_set.insert({"name":"zhangsan","age":18})
#或
my_set.save({"name":"zhangsan","age":18})
```

插入多条

```
#添加多条数据到集合中
users=[{"name":"zhangsan","age":18},{"name":"lisi","age":20}]
my_set.insert(users)
#或
my_set.save(users)
```

MongoDB的增删改查

```
my_set.update({"name":"zhangsan"},{'$set':{"age":20}})
```

```
#     (>)  大于 - $gt
#     (<)  小于 - $lt
#     (>=)  大于等于 - $gte
#     (<= )  小于等于 - $lte
```

```
#例：查询集合中age大于25的所有记录
for i in my_set.find({"age":{"$gt":25}}):
    print(i)
```

MongoDB的增删改查

```
#查询全部
for i in my_set.find():
    print(i)
#查询name=zhangsan的
for i in my_set.find({"name":"zhangsan"}):
    print(i)
print(my_set.find_one({"name":"zhangsan"}))
```

```
#删除name=lisi的全部记录
my_set.remove({'name': 'zhangsan'})

#删除name=lisi的某个id的记录
id = my_set.find_one({"name":"zhangsan"})["_id"]
my_set.remove(id)

#删除集合里的所有记录
db.users.remove()
```