第一天：

MongoDB

数据存储阶段

文件管理系统（.txt .doc .xls）

优点：使用简单，展现直观

可以长期保存数据

可存储数据量比较大

缺点：查找不方便，容易造成数据冗余，数据格式不规范

数据库管理阶段

优点：将数据结构化存储，降低冗余，提高了增删改查效率，方便扩展，方便程序调用。

缺点：数据库往往需要指令或语句操作，相对复杂。

几个概念

数据：能够输入到计算机中并被识别处理的信息集合。

数据结构：组成一个数据集合的数据之间的关系

数据库：按照一定的数据结构，存储数据的仓库。数据库是在数据库管理系统管理和控制下，在一定介质上的数据集合。

数据库管理系统：数据库管理软件，用于建立维护操作数据库。

关系型数据库：采用关系模型（二维表）来组织数据结构的数据库

Oracle DB2 SQLServer Mysql SQLite（唯一一个Python标准库支持的）

优点：\*容易理解，逻辑类似常见的表格

\*使用方便，都是用SQL语句，SQL语句非常成熟，数据一致性高，冗余低，数据完整性好，便于操作，技术成熟，功能强大，支持很多复杂操作

缺点：\*每次操作都要进行SQL语句解析，消耗较大

\*不能很好的满足并发需求，特便是海量数据爆发，关系型数据库读写能力会显得不足

\*关系型数据库往往每一步都要进行加锁的操作，也造成了数据库的负担

\*数据一致性高，有时也会使数据的存储不灵活。

非关系型数据库（NoSQL---）not only sql

优点：高并发，读写能力强

弱化数据结构一致性，使用更加灵活

有良好的可扩展性

缺点：通用性差，没有SQL语句那样同于的语句

操作灵活导致容易出错和混乱

没有外键关联等复杂的操作

Nosql的使用情况

1. 对数据存储灵活性要求高，一致性要求低
2. 数据处理海量并发，要求瞬间效率数据比较高
3. 数据比较容易建立nosql模型
4. 网站临时缓冲存储，爬虫应用

NoSQL分类：

1. 键值型数据库 Redis
2. 文档型数据库 MongoDB
3. 列存储数据库 HBASE
4. 图形数据库

MongoDB数据库

标签：非关系型数据库 文档型数据库 最像关系型的非关系型数据库

特点：

1. 是由c++编写的数据库管理系统
2. 支持丰富的数据操作，增删改查索引聚合
3. 支持丰富的数据类型
4. 使用方便，可以很好地扩展，相对比较成熟
5. 支持众多的编程语言接口（Python PHP c++ c#）

要求：

1. 关系型数据库和非关系型数据库各自有什么特点

1、关系型数据库优点就是他用二维表的形式，同时用SQL语句，数据的一致高，SQL技术成熟功能强大，功能强大。缺点就是不能处理那种高并发，每次执行都得解析，消耗大，读写能力不足。同时每一步有加锁操作。增加负担。

2、非关系型数据库，操作灵活，有扩展性，高并发。但是缺点是通用性差，没有外间关联等复杂操作。由于灵活性导致有时出行数据混乱。

1. MongoDB是一个什么样的数据库

MongoDB是一个非关系型数据库，是有c++编写，有丰富的数据操作，增删改查索引聚合，可以很好地扩展，支持众多编程语言的接口。

MongoDB的安装

自动安装

sudo apt-get install mongodb

默认安装位置：/var/lib/mongodb

配置文件位置：/etc/mongodb.conf

命令集：/usr/bin /uer/local/bin

手动下载

1. 下载安装包

[www.mongdb.com](http://www.mongdb.com)

1. 解压安装包

/usr/local /opt

1. 将解压后的MongoDB文件中的bin目录添加到环境变量里：

PATH=$PATH:/opt、mongo...../bin

export PATH

将两句写入启动脚本 /etc/rc.local

1. 重启

MongoDB命令

设置数据库存储位置

mongod -- dbpath dir(目录) 设置存储位置

e.g 将存储路径设置为dbs

mongod --dbpath dbs

设置数据库监听端口

mongd --port 8080

\*默认监听端口为27017

mongo

进入数据库交互操作界面

mongo shell:用来操作mongdb数据库的界面，在这里可以使用mongodb语句操作数据库内容

退出mongdb shell：quit（） exit ctrl-c

组织结构：键值对--》文档--》 集合--》数据库

-----------------------------------

ID | name | age

-----------------------------------

1 | Lucy | 17

-----------------------------------

2 | Tom | 18

-----------------------------------

｛"ID":1,"name":'lucy',"age":17｝,{ "ID":2,"name":'tom',"age":18}

mysql和mongodb概念对比

mysql mongodb 含义

database database 数据库

table collection 表/集合

column field 字段/域

row docment 记录/文档

index index 索引/索引

数据库的创建;

use databaseName

e.g．创建一个名字为stu的数据库

use stu

\*use实际为选择使用哪个数据库，当数据库不存在时会自动创建。

\*use后并不会立即创建出数据库，而是西药等到插入数据时数据库才会创建。

查看系统中的数据库

show dbs

系统数据库说明

admin：存储用户信息

local：存储本地数据

config：存储分片信息

数据库命名规则

1. 使用utf-8字符（mongo默认支持utf-8）
2. 不能含有空格 . / \ '\0'字符
3. 长度不能超过64字节
4. 不能和系统数据库重名

db：mongodb的全局量，代表你当前正在使用的数据库

* 如果不选择使用任何数据库db代表test，字节插入数据就会建立test数据库

数据库的备份和恢复

备份： mongodump -h host =d dbname -o bak

e.g. 将本机test数据库备份到把bak目录下

mongodump -h 127.0.0.1 -d test -o bak

恢复mongorestore -h dbhost:port -d dbname path

e.g.将test数据库恢复到本机 的res数据库中（res不存在自动创建）

mongorestore -h 127.0.0.1:27017 -d res bak/test

数据库检测

mongostat



insert query update delete :每秒增查改删的次数

flushes 每秒和磁盘交互次数

vsize 虚拟内存

res 物理内存

time 时间

mongotop监控数据库读写时长



ns ns数据表

total 总时间

read 读时间

write 写时间

删除数据库

首先use库，

然后

db.dropDatabase()

删除db所代表的数据库

创建集合

方法1

db.createCollection(collection\_name)

e.g. 创建class1集合

db.createCollection('class1')

方法2

当向一个集合插入数据的时候如果集合不存在则自动创建

db.collection\_name.insert(…)

查看数据库中集合

show collections

show tables

集合命名规则

1. 合法的UTF-8字符
2. 不能有‘\0’
3. 不能以system.开头，因为这是系统保留集合前缀
4. 不能和关键字重名

删除集合

db.collection.drop()

e.g. 删除class2集合

db.class2.drop()

集合的重命名

db.collection.renameCollection("new\_name")

e.g. 将class集合重命名为class0

db.class.renameCollection("class0")

文档

mongodb中数据的组织形式--》文档

mongodb文档：是以键值对的形式组成的一组数据。类似Python中字典描述数据的方式

键：即文档的域，表达了一个键值对的含义

键的命名规则：

1. utf-8格式字符串
2. 不能使用‘\0’
3. 一个文档中的键不能重复

值：即文档存储的数据。

\*文档中键值对是有序的

\*文档中键值对严格区分大小写

class.collectionname.find()

类型 值

整型 整数1 2 3

布尔类型 true false

浮点型 小数

Array 数组

Data 时间日期

Timestamp 时间戳

String 字符串

Symbol 特殊字符串

Bingary data 二进制子串

Null null空值

Object 内部文档（对象）

code js代码

regex 正则子串

ObiectId 自动生成ID标记

集合中的文档

1. 集合中的文档不一定有相同的域

\*个数不同

\*域不相同

\*数据类型不同

2、集合中文档各自比较独立，相互不影响

集合创建原则

1. 集合中的文档要描述同一类事物
2. 数据库中统一类数据尽量集中存放在相同的集合
3. 集合中的文档嵌套层数不要太多

插入文档

db.collection.insert()

功能：插入一个文档

参数：要插入的文档

db.class0.insert({'name':'LucY',"age":18,'sex':'w'})

\*插入操作中键可以不加引号

\*查看插入结果db.class0.find()

\*\_id值可以自己 插入，但是不能重复

插入多条文档

插入多条文档时，参数用中括号里面放入多个文档

e.g.

db.class0.insert([{name:"Alex",age:18,sex:'m'},{name:'Abby',age:19,sex:'w'}])

其他插入方法：

insertOne

db.class0.insertOne({name:'xiao',age:23,sex:'m'})

insertMany

db.class0.insertMany([{name:'john',age:15,sex:'m'},{name:'bai',age:22,sex:'m'}])

save插入文档

db.collection.seve()

如果正常插入与insert用法相同

e.g.

db.class0.save({name:'allen',age:18,sex:'m'})

db.class0.save([{name:'Eve',age:18,sex:'w'},{name:"sunny",age:19,sex:'w'}])

如果插入数据是有\_id域 ，且\_id域值存在时则会修改原有文档，如果该值不存在则正常插入

db.class0.save({"\_id" : ObjectId("5ba0bd993dd674ce1936a73c"),name:'Mary',age:20,sex:'w'})

获取集合对象方法

db.class0 🡺db.getCollection(‘class0’)

e.g.

db.getCollection(‘class0’).find()

