Contents

[Sqlite3 1](#_Toc457150759)

[创建sql文件 1](#_Toc457150760)

[打开或创建数据库 2](#_Toc457150761)

[导入sql文件 2](#_Toc457150762)

[CRUD操作 2](#_Toc457150763)

[事务操作 2](#_Toc457150764)

[Redis (remote dictionary service) 3](#_Toc457150765)

[MongoDB 6](#_Toc457150766)

[PostgreSQL 7](#_Toc457150767)

# Sqlite3

SQLite是一个进程中关系数据库，SQLite设计为嵌入在应用中，而不是作为一个单独的数据库服务器程序

支持数据库大小至2TB

ACID事务

$sudo apt-get install sqlite3 安装sqlite3

## 创建sql文件

create table project (

name text primary key,

description text,

deadline date

);

create table task (

id integer primary key autoincrement not null,

priority integer default 1,

details text,

status text,

deadline date,

completed\_on date,

project text not null references project(name)

);

insert into project (name, description, deadline)

values ('pymotw', 'Python MOdule of the Week', '2010-11-01');

insert into task (details, status, deadline, project)

values('write about select', 'done', '2010-10-03', 'pymotw');

insert into task (details, status, deadline, project)

values('write about random', 'waiting', '2010-10-10', 'pymotw');

insert into task (details, status, deadline, project)

values('write about sqlite3', 'actie', '2010-10-17', 'pymotw');

## 打开或创建数据库

$sqlite3 todo.db3

## 导入sql文件

sqlite> .read todo\_schema.sql; 执行sql文件

## CRUD操作

sqlite> select \* from task; 查询

sqlite> delete from task where id=2; 删除

sqlite> insert into task (details, status, deadline, project) 插入

values('write about random', 'waiting', '2010-10-10', 'pymotw');

sqlite> update task set status='done' where id=4; 更新

## 事务操作

sqlite> begin;

sqlite> … //crud operation

sqlite> commit; 递交

(或者sqlite> rollback; 撤销)

# Redis (remote dictionary service)

也称为数据结构服务器，发布订阅系统，作为有用的数据结构算法和程序的工具包。作为栈，队列或优先队列， 对象存储系统（通过hashtable），集合；支持集合操作（并，交，差），提供原子命令，和多步命令的事务机制；键到期及缓存

1. Download and install [Redis-x64-3.0.500-rc2.msi](https://github.com/MSOpenTech/redis/releases/download/win-3.0.500-rc2/Redis-x64-3.0.500-rc2.msi) from <https://github.com/MSOpenTech/redis/releases>
2. Run Redis server:

C:\Program Files\redis > redis-server redis.windows.conf

1. Run Redis cli:

C:\Program Files\redis > redis-cli

127.0.0.1:6379> ping

// 键值

>set key value //mset key1 value1 key2 value2 …

> get key //mget key1 key2

//hash table (map) 嵌套的Redis对象

mset key1 value1 key2 value2

mget key1 key2

hmset key key1 value1 key2 value2 …

hvals key

hkeys key

for example:

mset user:qzlin:name “qizhong lin” user:qzlin:password changeit //使用:作为分离的键

hmset user:qzlin name “qizhong lin” password changeit //不使用分离的键

hvals user:qzlin

hkeys user:qzlin

// 列表

rpush qzlin:wishlist google yahoo baidu //建立列表

lrange qzlin:wishlist 0 -1 //取子集

lrem qzlin:wishlist 0 google //删除元素

rpoplpush qzlin:wishlist qzlin:visited //从左列表弹出元素， 在右列表压入元素

// set 集合

sad key value1 value2 … //创建集合

smembers key //获取集合

sinterstore key key1 key2 //求交集

sdiffstore key key1 key2 //差集

sunionstore key key1 key2 //并集key1, key2, 结果存入key中

for example:

sad news nytimes.com baidu.com

smembers news

// 有序集合 (随机存取的优先级队列)

zadd key score1 key1 score2 key2

zrange key idx0 idx 1

zrangebyscore key score0 score1

// 加权后求并集

zunionstore destination numkeys key1 key2 … [weights weight1 weight2 …] [aggregate sum|min|max]

for example:

zadd visits 500 google 9 yahoo 10 baidu

zunionstore importance 2 visits votes weights 1 2 aggregate sum

事务: multi … exec ; multi和exec内多命令按事务执行，在使用multi命令时，命令在实际上并不执行，而是排入队列，然后按顺序执行。

到期：有助于避免总的键集无限增长，做法是安排Redis经过一定的时间就删除一个键-值对

expire key seconds

setex key second value // 设置key value的生命

ttl key //获取key的生命

persist key //最近使用缓存算法(MRU, Most Recently Used)： 每当你检索一个值时，更新它的到期时间，确保你最近使用的键将继续保留在Redis中，而被忽视的键将正常到期

Redis命名空间称为数据库(database),以数字为键。默认的命名空间0（也称为数据库0）交互

// select 0

set greeting hello

get greeting

select 1 //通过select切换到数据库1

get greeting //返回(nil)

所有的数据库都运行在同一服务器实例内，Redis就允许用move命令，在不同命令空间之间移动键. 若单个Redis服务器运行不同的应用程序，又要允许这些应用程序相互之间交换数据

move greeting 2

select 2

get greeting

别的操作命令： rename, type, del, flushdb, flushall.

//发布-接收模式

127.0.0.1:6379> brpop comments 300 //某客户端监听 key=comments, 超时为5分钟,

127.0.0.1:6379> lpush comments “hello world!” //某客户端发布 key-values

Key points:

redis server启动后，独占进程，能否改为后台服务？

redis server服务是单线程的，能否在同一台机器上启动多个实例？也就是修改默认端口6379？

$redis-server --port 6378 --daemonize yes 服务端

$redis-cli -p 6378 客户端

Redis允许在运行的过程中，在不重启服务器的情况下更改服务器配置

127.0.0.1:6378> config set tcp-keepalive 60

127.0.0.1:6378> config set port 6380

127.0.0.1:6380>

注意：先把心跳包发送时间间隔修改成60秒。

Redis主从复制

特性：

* redis使用异步复制
* 一个master拥有多个slave
* master死了，会自举一个slave作为master

做法：

打开三个终端启动三个redis 服务实例，然后启动三个redis client连接

$redis-server --daemonize yes

$redis-server --daemonize yes --slaveof 127.0.0.1 10000

$redis-server --daemonize yes --slaveof 127.0.0.1 10000

从而两个slave会同步一个主

Redis集群

集群技术是构建高性能网站架构的重要手段，试想在网站承受高并发访问压力的同时，还需要从海量数据中查询出满足条件的数据，并快速响应，我们必然想到的是将数据进行切片，把数据根据某种规则放入多个不同的服务器节点，来降低单节点服务器的压力。

集群主要是利用切片技术来组建的集群

集群要实现的目的是要将不同的 key 分散放置到不同的 redis 节点，这里我们需要一个规则或者算法.

Redis 集群中内置了 16384 个哈希槽，当需要在 Redis 集群中放置一个 key-value 时，redis 先对 key 使用 crc16 算法算出一个结果，然后把结果对 16384 求余数，这样每个 key 都会对应一个编号在 0-16383 之间的哈希槽，redis 会根据节点数量大致均等的将哈希槽映射到不同的节点。

使用哈希槽的好处就在于可以方便的添加或移除节点。

当需要增加节点时，只需要把其他节点的某些哈希槽挪到新节点就可以了；

当需要移除节点时，只需要把移除节点上的哈希槽挪到其他节点就行了；

在新增或移除节点的时候不要让我们先停掉所有的 redis 服务

做法：

先启动三个redis服务实例

创建集群，使三个实例互相通讯, （具体见网上）

$./redis-tri.rb create –replicas 0 127.0.0.1:9001 127.0.0.1:9002 127.0.0.1:9003

连接127.0.0.1:9001的客户端，设置或获取的数据，可能被redis cluster自动重定向到127.0.0.1:9003。

# MongoDB

安装

Install and set env

install mongoDB in ubuntu

https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/

Import the public key used by the package management system.

$sudo apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv EA312927

Create a list file for MongoDB

$echo "deb http://repo.mongodb.org/apt/ubuntu trusty/mongodb-org/3.2 multiverse" | sudo tee /etc/apt/sources.list.d/mongodb-org-3.2.list

Reload local package database.

$sudo apt-get update

Install the MongoDB packages.

$sudo apt-get install -y mongodb-org

启动mongod --ubuntu

The MongoDB instance stores its data files in /var/lib/mongodb and its log files in /var/log/mongodb by default, and runs using the mongodb user account. You can specify alternate log and data file directories in /etc/mongod.conf. See systemLog.path and storage.dbPath for additional information.

Start MongoDB

$sudo service mongod start

Verify that the mongod process has started successfully by checking the contents of the log file at /var/log/mongodb/mongod.log for a line reading

[initandlisten] waiting for connections on port <port>

Stop MongoDB

$sudo service mongod stop

Restart MongoDB

$sudo service mongod restart

启动mongod --window

>mongod --dbpath ./data 默认是c:/data/db

默认情况下，MongoDB监听27017端口，mongod还启动一个HTTP服务器，监听数字比主端口号高1000的端口，也就是28017端口

$curl localhost:28017

MongoDB shell

MongoDB自带JavaScript shell,可以shell中使用命令行与MongoDB实例交互。Shell是一个功能完备的JavaScript解释器，可运行任意JavaScript程序

>mongo localhost:27017/test

启动时，shell会连到MongoDB服务器的test数据库，并将数据库连接赋值给全局变量db.这个变量是通过shell访问MongoDB的主要入口点

为了方便习惯SQL shell用户，shell包含一些非JavaScript语法的扩展

>help

> show dbs //show all database

> use test // select database

> db.dropDatabase() // remove database

> show collections //show collections of selected database

> db.customer.drop() //delete collection

>db.customer.find() // read all documents of one collection of one database

>db.customer.find({‘firstname’:’Alice’}) // read

>db.customer.insert({‘firstname’:’QiZhong’, ‘lastname’: ‘Lin’}) // post

>db.customer.update({‘firstname’:’Alice’}, {‘firstname’:’QiZhong’, ‘lastname’:’Lin’}) // put

>db.customer.remove({‘lastname’:’Lin’}) //delete

基本数据类型

Null, true/false, 数值（默认使用64位浮点型，对于整数用NumberInt()/NumberLong()）, 字符串，日期(new Date()), 正则表达式(/foobar/i), 数组([“a”, “b”]), 内嵌文档({“foo”: “bar”}, 对象id(ObjectId()), 二进制数据，代码(Javascript代码function(){…})

使用shell执行脚本

$mongo ip:port/db script1.js script2.js script3.js

>load(“script1.js”)

注意：在脚本中，可以访问db变量，以及其他全局变量，但不能使用shell语法糖（如use db; show collections;等），只能使用对应的shell辅助函数

db.getSisterDB(“foo”) use foo

db.getMongo().getDBs() show dbs

db.getCollectionNames() show collections

# PostgreSQL

Download and install postgresql-9.5.0-rc1-windows-x64.exe

安装过程会要求设置管理员密码：qzlin （默认管理员:postgre）

数据库默认地址: C:\Program Files\PostgreSQL\9.5\data

Open pgAdmin III

连接服务器： right click PostgreSQL 9.5 (localhost:5432) -> 连接服务器

数据库 -> 创建数据库 //创建数据库

登录角色 -> 创建登录角色 //创建登录角色

Psql:

CREATE ROLE qzlin LOGIN PASSWORD ‘pass@123’ VALID UNTIL ‘infinity’;

then login with qzlin

CREATE DATABASE blog;