非关系型数据库-NoSQL

1. 什么是 NoSQL？

NoSQL(NoSQL = Not Only SQL)，意为反 SQL 运动，是一项全新的数据库革命性运动，2000 年前就有人提出，发展至 2009 年趋势越发高涨。它是指运用非关系型的数据存储，相对于铺天盖地的关系型数据库运用，这一概念无疑是一种全新的思维的注入。

随着互联网 web2.0 网站的兴起，传统的关系数据库在应付 web2.0 网站，特别是超大规模和高并发的 SNS 类型的 web2.0 纯动态网站已经显得力不从心，暴露了很多难以克服的问题，而非关系型的数据库则由于其本身的特点得到了非常迅速的发展。NoSQL 数据库的产生就是为了解决大规模数据集合多重数据种类带来的挑战，尤其是大数据应用难题。

分类	Examples 举例	典型应用场景	数据模型	优点
键值（key-value）	Tokyo Cabinet/Tyrant, Redis, Voldemort, Oracle BDB	内容缓存，主要用于处理大量数据的高访问负载，也用于一些日志系统等等。	Key 指向 Value 的键值对，通常用 hash table 来实现	查找速度快
列存储数据库	Cassandra, HBase, Riak	分布式的文件系统	以列簇式存储，将同一列数据存在一起	查找速度快，可扩展性强，更容易进行分布式扩展
文档型数据库	CouchDB, MongoDb	Web 应用（与 Key-Value 类似，Value 是结构化的，不同的是数据库能够了解 Value 的内容）	Key-Value 对应的键值对，Value 为结构化数据	数据结构要求不严格，表结构可变，不需要像关系型数据库一样需要预先定义表结构
图形(Graph)数据库	Neo4J, InfoGrid, Infinite Graph	社交网络，推荐系统等。专注于构建关系图谱	图结构	利用图结构相关算法。比如最短路径寻址，N 度关系查找等

2. NoSQL 的特性？

NoSQL 是 key-value 形式存储，和传统的关系型数据库不一样,不一定遵循传统数据库的一些基本要求，比如说遵循 SQL 标准、ACID 属性、表结构等等。

这类数据库主要有以下特点：

非关系型的、分布式、开源的、水平可扩展的

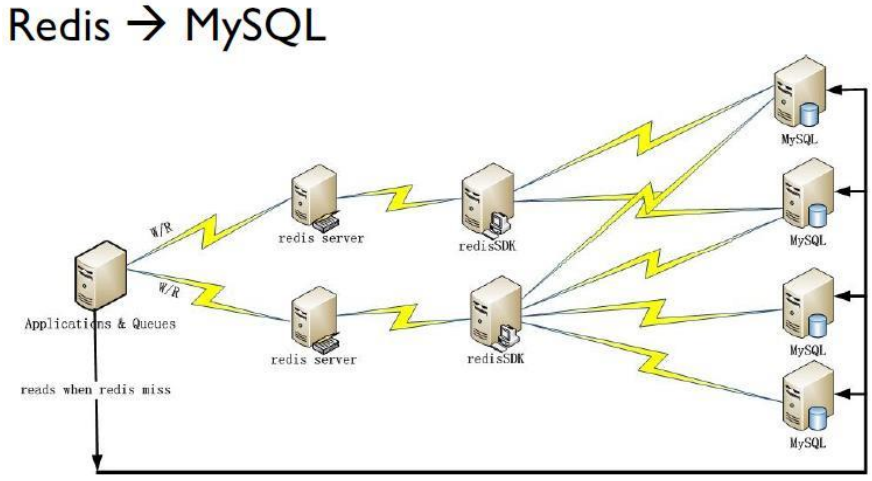
处理超大量数据

击碎了性能瓶颈
对数据高并发读写
对海量数据的高效率存储和访问
对数据的高扩展性和高可用性

3. 什么是 Redis?

Redis 是一个开源的，先进的 key-value 存储。它通常被称为数据结构服务器，因为键可以包含 string（字符串）、hash（哈希）、list（链表）、set（集合）和 zset（sorted-set--有序集合）。这些数据类型都支持 push/pop、add/remove 及取交集并集和差集及更丰富的操作。

Redis 和 Memcached 类似，它支持存储的 value 类型相对更多，与 memcached 一样，为了保证效率，数据都是缓存在内存中，区别是 Redis 会周期性的把更新的数据写入磁盘或者把修改操作写入追加的记录文件，并且在此基础上实现了 master-slave(主从)同步。



4. Redis 安装部署

Redis 的官方网站是：http://redis.io

下载安装包：

wget http://download.redis.io/releases/redis-2.8.6.tar.gz

编译安装：

tar –zxvf redis-2.8.6.tar.gz
cd redis-2.8.6
make
make PREFIX=/usr/local/redis install

指定安装位置，如果没有指定安装位置 PREFIX=/usr/local/redis，则 make install 会把 redis 安装到/usr/local/bin/目录下

mkdir /usr/local/redis/etc
cp ./redis.conf /usr/local/redis/etc/

复制 Redis 的配置文件到/usr/local/redis/etc/下，便于管理。

修改配置文件：

vi /usr/local/redis/etc/redis.conf

daemonize no 修改为 yes #后台启动

启动服务：

/usr/local/redis/bin/redis-server 配置文件

必须制定配置文件位置，否则会提示警告

客户端连接：

/usr/local/redis/bin/redis-cli

-h	IP：	连接指定的 redis 服务器
-p	6379：	指定 redis 服务器的端口
-a	密码：	使用密码登录
-n	数据库号：	指定连接哪个数据库
--raw：		redis 支持存储中文

停止 Redis：

/usr/local/redis/bin/redis-cli shutdown

或

pkill -9 redis

5. Redis 常用命令

1) string 类型及操作

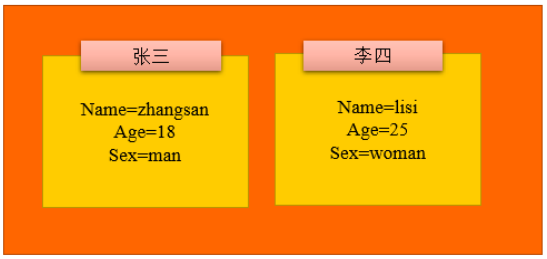
string 是最简单的类型，一个 key 对应一个 value，string 类型是二进制安全的。redis 的 string 可以包含任何数据。

1. set：设置 key 对应的值为 string 类型
   例如：我们添加一个 name=atguigu 的键值对应
   redis127.0.0.1:6379>set name atguigu
2. setnx：设置 key 对应的值为 string 类型，如果 key 已经存在，返回 0，nx 是 not exist 的意思
3. get：获取 key 对应的 string 值，如果 key 不存在返回 nil
4. mset & mget 同时设置和获取多个键值对
5. incrby：对 key 的值做加加（指定值）操作，并返回新的值
6. del：删除一个已创建的 key

2) hash 类型及操作

Redis hash 是一个 string 类型的 field（字段）和 value 的映射表，它的添加、删除操作都是 0(1)平均；hash 特别适合用于存储对象，相较于将对象的每个字段存成单个 string 类型，将一个对象存储在 hash 类型中会占用更少的内存，并且可以更方便的存取整个对象。

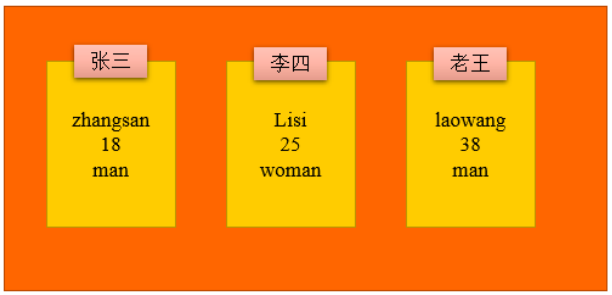
1. hset：设置 hash field 为指定值，如果 key 不存在，则先创建。
   例如：为 num1 表创建一个叫 name 字段（key），键值是 liuchuan
   redis127.0.0.1:6379>hset num1 name liuchuan



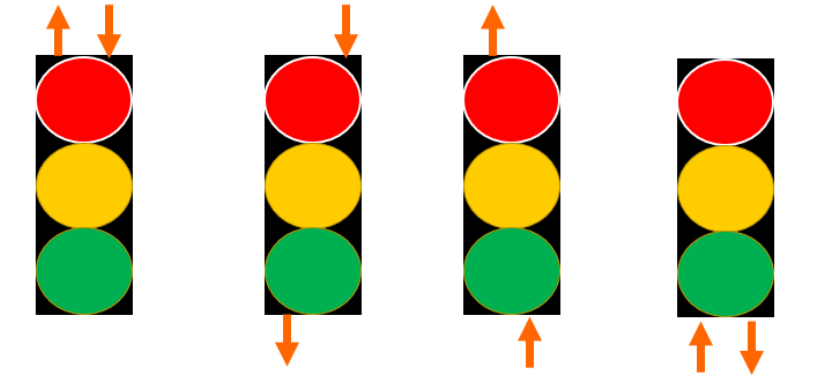
2. hget、hmset、hmget 意义同上近似
3. hdel：删除制定表中的某一个键值对
4. hgetall：列出表中的所有键值对

3) list 类型及操作

list 是一个链表结构，主要功能是 push、pop、获取一个范围内的所有值等等，操作中 key 理解为链表的名字。Redis 的 list 类型其实就是一个每个子元素都是 string 类型的双向链表。我们可以通过 push、pop 操作从链表的头部或尾部添加删除元素。



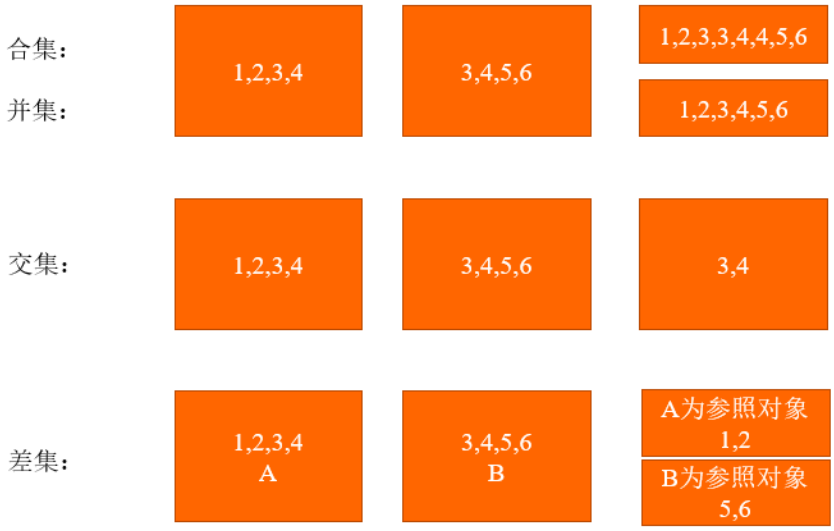
1. lpush：在 key 对应 list 的头部添加字符串元素。
   redis127.0.0.1:6379>lpush zhangsan zhangsan
   redis127.0.0.1:6379>lpush zhangsan 18
2. lrange：从指定链表中获取指定范围的元素
   redis127.0.0.1:6379>lrange zhangsan 0 -1
   0 -1：此范围代表全部元素，意为从头到尾
3. lpush、rpush、lpop、rpop、lrange 详见图示



lpush&lpop	lpush&rpop	rpush&lpop	rpush&rpop
写：绿黄红 删：红黄绿 先写入的后删除	写：绿黄红 删：绿黄红 先写入的先删除	写：红黄绿 删：红黄绿 先写入的先删除	写：红黄绿 删：绿黄红 先写入的后删除

4) Set 类型及操作

set 是集合，他是 string 类型的无序集合。Set 是通过 hash table 实现的，对集 、交集、差集。通过这些操作我们可以实现社交网站中的好友推荐和 blog 的 tag 功能。集合不允许有重复值。



1. sadd：添加一个或多个元素到集合中
   redis127.0.0.1:6379>sadd mset 1 2 3 4
2. smembers：获取集合里面所有的元素
   redis127.0.0.1:6379> smembers mset
3. srem：从集合中删除指定的一个或多个元素
4. spop：随机从集合中删除一个元素，并返回
5. scard：获取集合里面的元素个数
6. sdiff：返回集合 1 与集合 2 的差集。以集合 1 为主
   redis127.0.0.1:6379>sdiff mset1 mset2
7. sinter：获得两个集合的交集

8. sunion：获得指定集合的并集

5) zset 类型及操作

zset 是 set 的一个升级版本，它在 set 的基础上增加了一个顺序属性，这一属性在添加修改元素的时候可以指定，每次指定后，zset 会自动重新按新的值调整顺序。可以理解为有两列的 mysql 表，一列存的 value，一列存的顺序。操作中 key 理解为 zset 的名字。

有序集合的排序方式：

1. 先按照预设设置好的分数进行先后排序

2. 相同分数的值，按照值的ASCII码表的顺序排序

1. zadd：向一个指定的有序集合中添加元素，每一个元素会对应的有一个分数。你可以指定多个分数/成员组合。如果一个指定的成员已经在对应的有序集合中了，那么其分数就会被更新成最新的，并且该成员会重新调整到正确的位置，以确保集合有序。分数的值必须是一个表示数字的字符串。

redis127.0.0.1:6379>zadd zset 2 zhangsan 1 lisi 1 wangwu

2. zrange：返回有序集合中，指定区间内的成员。其中成员按照 score（分数）值从小到大排序。具有相同 score 值的成员按照字典顺序来排列。

redis127.0.0.1:6379>zrange zset 0 -1 withscores

注：withscores 返回集合中元素的同时，返回其分数（score）

3. zrem：删除有序集合中指定的值

redis127.0.0.1:6379>zrem zset zhangsan

4. zcard：返回有序集合元素的个数

6) 其他相关命令

1. keys：按照键名查找指定的键。支持通配符（* ?等）

redis127.0.0.1:6379>keys h*llo

2. exists：确认一个键是否存在（1 表示存在）

3. del：删除一个键（通用）

4. expire：设置一个键（已存在）的过期时间，如果键已经过期，将会被自动删除

5. ttl：以秒为单位，返回指定键的剩余有效时间

当 key 不存在时，返回 -2 。
当 key 存在但没有设置剩余生存时间时，返回 -1 。
否则，以秒为单位，返回 key 的剩余生存时间。
6. select：选择一个数据库，默认连接的数据库是 0，可以支持共 16 个数据库。在配置文件中，通过 databases 16 关键字定义。
7. move：将当前数据库的键移动到指定的数据库中
8. type：返回键的类型
9. dbsize：返回当前库中键的数量（所有类型）
10. save：保存所有的数据。很少在生产环境直接使用 SAVE 命令，因为它会阻塞所有的客户端的请求，可以使用 BGSAVE 命令代替. 如果在 BGSAVE 命令的保存数据的子进程发生错误的时,用 SAVE 命令保存最新的数据是最后的手段。
11. info：获取服务器的详细信息
12. config get：获取 redis 服务器配置文件中的参数。支持通配符
13. flushdb：删除当前数据库中所有的数据
14. flushall：删除所有数据库中的所有数据

6. Redis 高级应用

1) 密码防护

给 redis 服务器设置密码
1、修改配置文件
vi /usr/local/redis/etc/redis.conf
requirepass 123456
2、重启 redis
pkill redis
…/bin/redis-server …/etc/redis.conf
3、客户端登录
…/bin/redis-cli -a 123456
或
交互模式下使用【auth 密码】命令

2) 主从同步

Redis 主从复制过程：

- Slave 与 master 建立连接，发送 sync 同步命令
- Master 会启动一个后台进程，将数据库快照保存到文件中，同时 master 主进程会开始收集新的写命令并缓存。
- 后台完成保存后，就将此文件发送给 slave
- Slave 将此文件保存到硬盘上

1. 主服务器给自己设置好密码即可（iptables&SELinux 关闭）

2. 从服务器修改配置文件，用来连接主服务器

老版本：

从：

slaveof <masterip> <msterport>　　#主服务器的 IP 和端口

masterauth <masterpass>　　#主服务器的密码(主服务器要设置好密码)

新版本 redis 5.* 以上：

主：

找到 bind 127.0.0.1 注释掉，或者修改为本机的 IP 地址(重启)

从：

replicaof <masterip> <msterport>　　#主服务器的 IP 和端口

masterauth <masterpass>　　#主服务器的密码(主服务器要设置好密码)

3. 重启从服务器，然后测试（可通过 info 命令获取当前服务器身份类型）

3) 数据持久化

Redis 是一个支持持久化的内存数据库，也就是说需要经常将内存中的数据同步到硬盘来保证持久化。

snapshotting（快照）--默认方式

RDB 持久化方式能够在指定的时间间隔能对你的数据进行快照存储。是默认的持久化方式。这种方式是将内存中数据以快照的方式写入到二进制文件中，默认的文件名为 dump.rdb。这种持久化方式被称为快照 snapshotting（快照）。

过了 900 秒并且有 1 个 key 发生了改变 就会触发 save 动作

过了 300 秒并且有 10 个 key 发生了改变 就会触发 save 动作

过了 60 秒并且至少有 10000 个 key 发生了改变 也会触发 save 动作

结论：在 redis.conf 文件中 dir ./定义了数据库文件的存放位置，默认是当前目录。所以每次重启 redis 服务所在的位置不同，将会生成新的 dump.rdb 文件；建议服务器搭建完成时先修改快照文件保存位置。

append-only file（缩写 aof）

使用 AOF 会让你的 Redis 更加耐久：你可以使用不同的持久化策略：每次写的时候备份、每秒备份、无备份。使用默认的每秒备份策略,Redis 的性能依然很好(备份是由后台线程进行处理的,主线程会尽力处理客户端请求),一旦出现故障，你最多丢失 1 秒的数据。

打开 redis.conf 配置文件开启 AOF 持久化

appendonly no

默认不使用 AOF 持久化（450 行）将 no 改成 yes。

appendfsync always

有写操作，就马上写入磁盘。效率最慢，但是最安全

appendfsync everysec

默认，每秒钟写入磁盘一次。

appendfsync no

不进行 AOF 备份，将数据交给操作系统处理。最快，最不安全

测试：重启 redis 服务，登录 client 添加一个键值，退出然后 ls 命令查看下是否生成 appendonly.aof。

可以用 cat 查看。