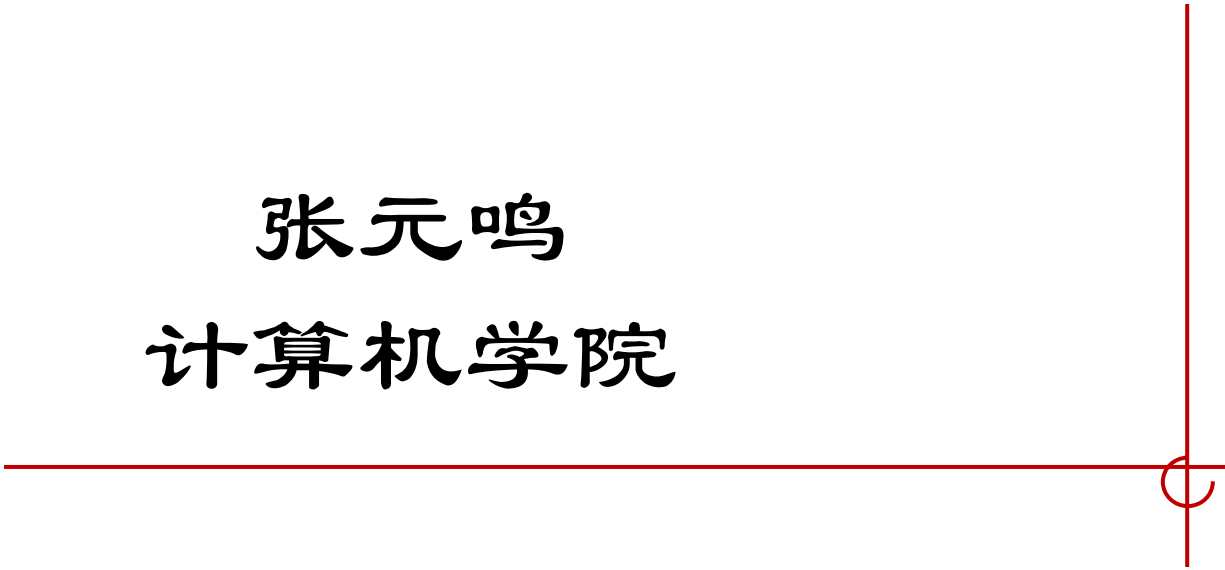
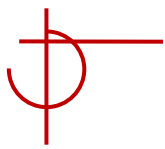




第二章 键值数据库

张元鸣
计算机学院





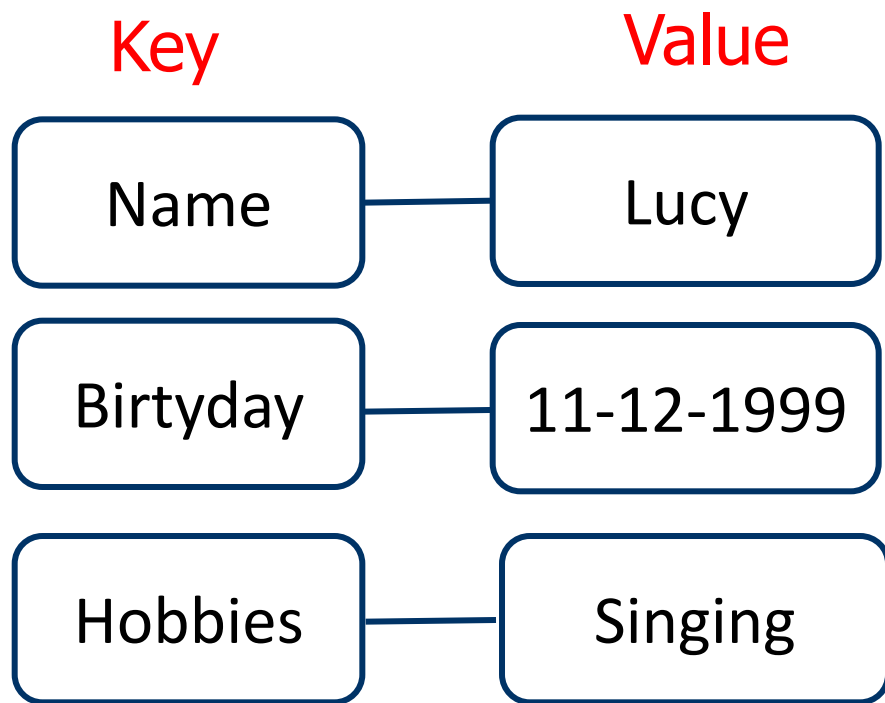
本章内容

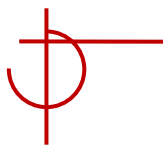
- 2.1 键值数据库的数据结构
- 2.2 键的设计与分区
- 2.3 值的设计与结构化
- 2.4 键值数据库的特点
- 2.5 Redis键值数据库



2.1 键值数据库的数据结构

键值数据库是一种最简单的NoSQL数据库，它是一个键值对的集合，能够存储大量数据，这种简单性也让它成为NoSQL中最具扩展性的数据库类型。





2.1 键值数据库的数据结构

数组是程序设计中一种最简单的数据结构，它把具有**相同数据类型**的若干元素**按无序的方式**组织起来。要求：

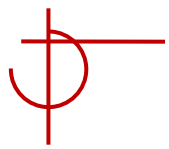
- 标识符（数组下标）只能是整数。
- 元素值是相同数据类型

0	7.21
1	10.22
2	19.2
3	29.4
4	30.29

0	True
1	False
2	False
3	True
4	False

0	Word
1	Hello
2	Good
3	Test
4	Order

三个不同的数组



2.1 键值数据库的数据结构

关联数组是对数组概念的泛化，其标识符是任意的标量，称为键（Key），用于检索其对应的值（Value）：

- 键可以是任意的字符串或整数；
- 键不可以重复，但值可以重复。

Key Value

S01	张杰
S02	李玮
S03	王韩
S04	蒋珊
S05	陆阳

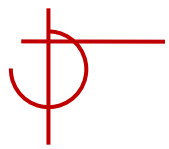
Key Value

张杰	1.68
李玮	1.68
王韩	1.82
蒋珊	1.64
陆阳	1.79

Key Value

PI	3.14
17234	34468
CapitalChina	Beijing
Foo	Bar
Start	1

三个不同的关联数组



2.1 键值数据库的数据结构

键值数据库的数据结构就是**关联数组**，也称为Hash表，形式上是一个键值对（Key，Value）。

- **键（Key）**：用来查询值的唯一标识符。
- **值（Value）**：是一个实例，与唯一的Key相关联，可以是任意的数据类型。
- **命名空间键（namespace）**：键值对构成的集合，也称为桶（bucket）或数据库（database）。
 - 同一个命名空间中的键不允许相同，不同命名空间中的键可以相同。
- **分区**：根据键名，把数据分割成不同的单元，存储在集群中的不同服务器上，实现负载均衡。



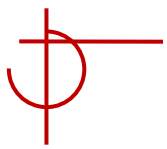
2.1 键值数据库的数据结构

三个键值对数据库

桶1	
S01	张杰
S02	李玮
S03	王韩

桶2	
S01	岑水
S03	蒋珊
S05	陆阳

桶3	
S01	黄河
S04	王杰
S07	李飞



2.2 键的设计与分区

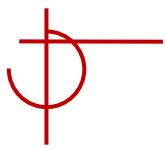
1、键的设计

在键值数据库中，键是指向值的引用，用来找到值的唯一标识符，其命名基本原则是使用有意义的内容作为键名。

键命名一般方法：

实体名：实体标识符：实体属性

如：表名：主键值：属性



2.2 键的设计与分区

将关系型数据库转换为键值数据库存储

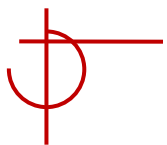
Students
基本表

Sno	Sname	Ssex	Sage	Dno
S01	王建平	男	21	D01
S02	刘华	女	19	D01
S03	范林军	女	18	D02
S04	李伟	男	19	D03
S05	黄河	男	18	D03
S06	长江	男	20	D03



键值数据库

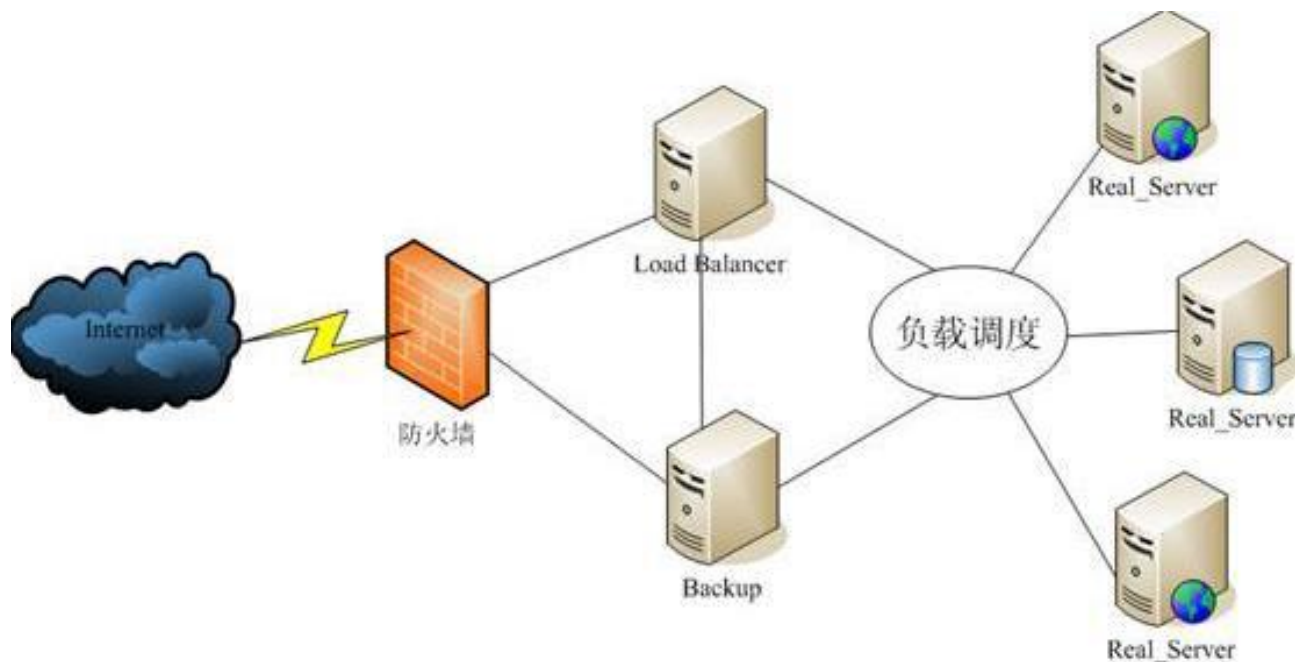
- ① (Students:S01:Sname, 王建平)
- ② (Students:S02:Sname, 刘华)
- ③ (Students:S03:Sname, 范林军)
- ④ (Students:S04:Sname, 李伟)
- ⑤ (Students:S05:Sname, 黄河)
- ⑥ (Students:S06:Sname, 长江)



2.2 键的设计与分区

2、键的分区

当键值对的数量非常巨大时，一台服务器将难以应对，此时需要集群（具有多个服务器节点）来处理这些键值对。集群是一组相互连接的计算机。

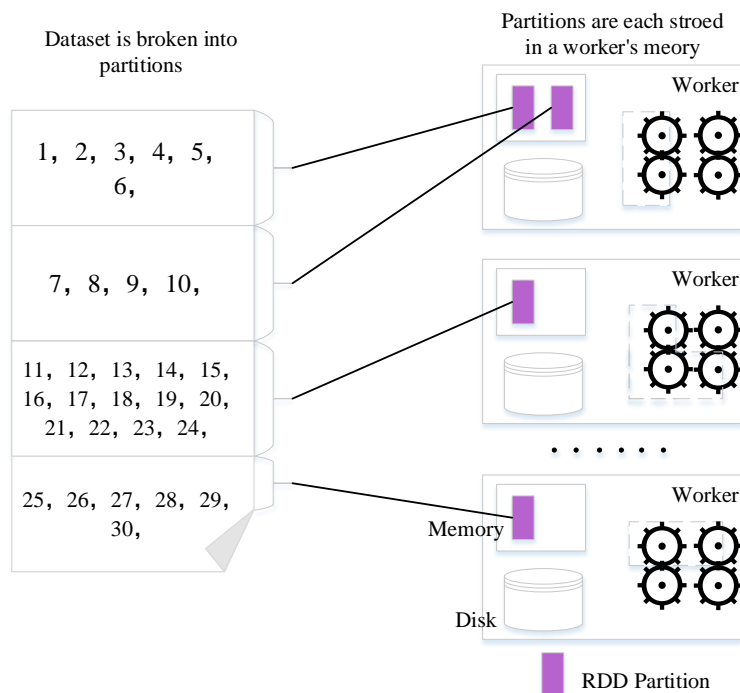




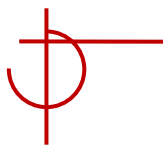
2.2 键的设计与分区

2、键的分区

集群中的一组或一个服务器称为集群的分区。为了实现集群负载平衡，希望将数量巨大的键值对均匀地存储到不同分区。



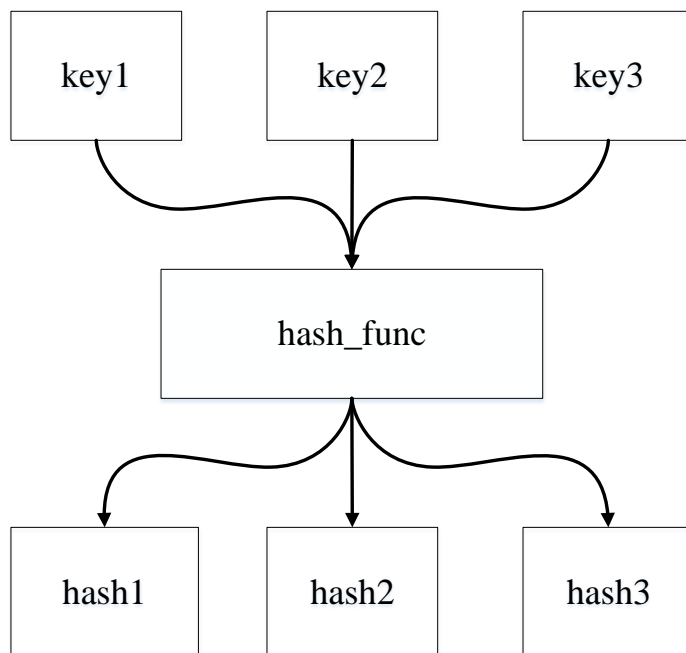
目标：尽可能将键值对均衡地分配给集群中的不同分区。

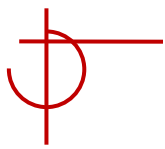


2.2 键的设计与分区

2、键的分区

- 键值数据库中一般以“键”作为分区的依据，通过所定义的Hash哈希函数直接将键值对映射到相应分区。





2.2 键的设计与分区

键名及其哈希码

键名	哈希函数	哈希值
Students:S01:Sname	SHA-1	683755b70d5f432c765b298b17552e6431dbbb7e
Students:S02:Sname		23a4c4bfec423a57018d61325d3151c78925835a
Students:S03:Sname		5aeefcea47ae7a2911bcc833414623c0b930649
Students:S04:Sname		911e30f616d8215ac1f752b2ac18a61c97dec40
Students:S05:Sname		96963c2e2cad771cb213d88bc16e7ab7ed511f59

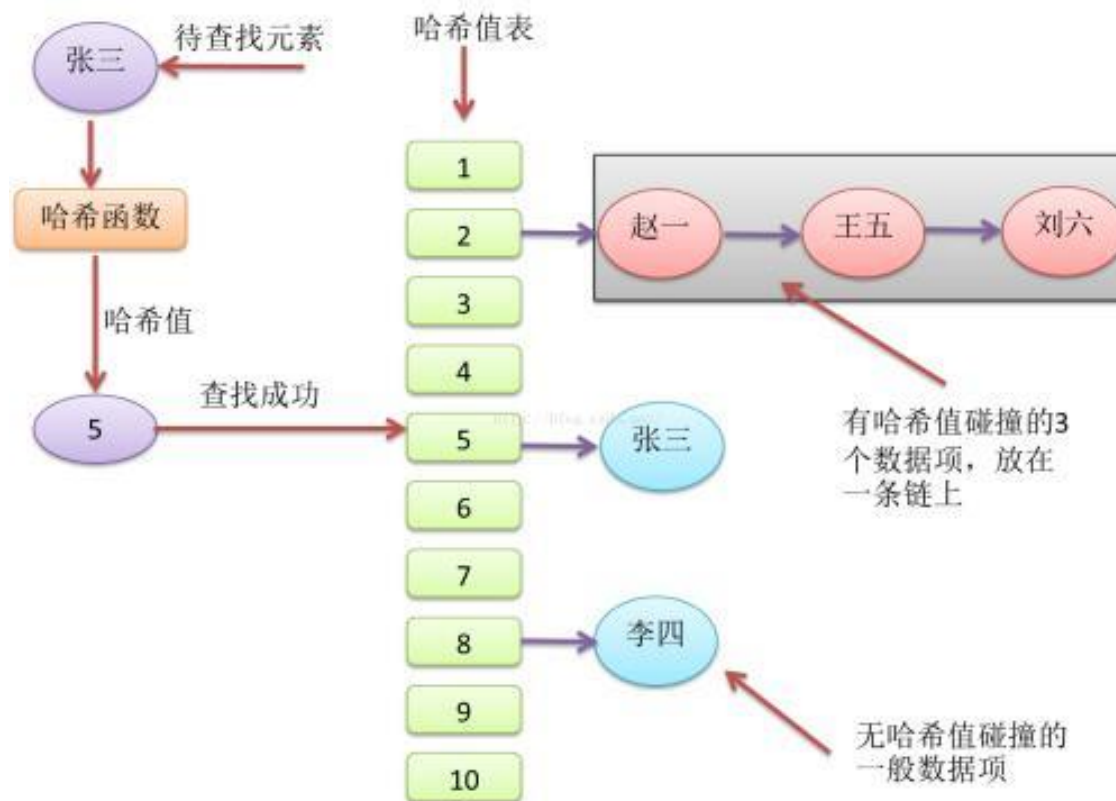


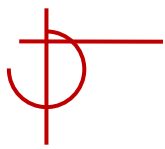
2.2 键的设计与分区

2、键的分区

- 如果两个不同的键产生了相同的哈希值，则称为**碰撞**。

解决的基本方法是在对应的存储空间中另外设置一个列表，类似于一个链表。



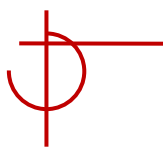


2.2 键的设计与分区

2、键的分区

- 如果某个键无法实现负载均衡，可以采取的办法：
 - 另选合适的分区键，使得分区结果更加均衡。
 - 改变哈希分区算法。





2.2 键的设计与分区

第42卷 第8期
2019年8月

计 算 机 学 报
CHINESE JOURNAL OF COMPUTERS

Vol. 42 No. 8
August 2019

面向 MapReduce 的迭代式数据均衡分区策略

张元鸣 蒋建波 陆佳炜 徐俊 肖刚

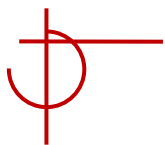
(浙江工业大学计算机科学与技术学院 杭州 310023)

摘 要 MapReduce 是一种适用于大数据处理的重要并行计算框架,然而,由于难以提前全面获得中间数据的分布规律,默认的数据分区策略往往会造成 Reducer 端的数据倾斜,会直接影响 MapReduce 的整体性能.为了实现数据均衡分区,本文提出一种迭代式数据均衡分区策略,将每个 Mapper 节点要处理的数据块细分后以迭代方式循环处理,根据已迭代轮次的微分区分配结果决定当前迭代轮次的微分区分配方案,以不断调整历次迭代产生的数据倾斜,逐步实现数据均衡分区.给出了迭代式数据分区策略的分配时机、分配准则、分配评价模型和分配算法.基于公开的数据集,对迭代式数据均衡分区策略进行了详细测评,结果表明,该策略能够得到更均衡的数据分区结果,当数据集本身倾斜比较显著时,MapReduce 整体性能比默认分区策略平均提高了 11.1% 和 19.7%.

关键词 MapReduce; 大数据; 数据倾斜; 迭代式数据分区; 微分区; 均衡分区

中图法分类号 TP311

DOI 号 10.11897/SP.J.1016.2019.01873



2.2 键的设计与分区

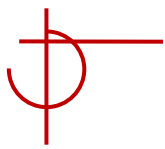
3、命名空间

由互不相同的键值对所构成集合称为命名空间，也称为桶。

- 同一个命名空间中的键互不相同；
- 不同命名空间中的键可以相同。

custMgmt:prod:12986:name

ordMgmt:prod:12986:name



2.2 键的设计与分区

4、无纲要模式

- 对于关系型数据库，一般需要先构建**数据库纲要**（数据库结构），然后再添加数据。
- 在键值数据库中，则采用**无纲要模型**：
 - 不需要在添加数据之前定义所有的键，也无需指定值的数据类型；
 - 可以任意的修改数据。

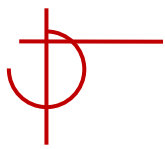


2.3 值的设计与结构化

1、值的数据类型

键值数据库中的值可以存放多种类型的数据，包括字符串(String)，哈希(Hash)，列表(list)，集合(sets)和有序集合(sorted sets)等类型，例如：

- 整数：用来存储一个整数值，如350、600等；
- 浮点数：用来存储一个浮点数值，如234.23、9832.23等；
- 字符串：字符串是一种常见的数据类型，通过双引号括起来，如“Li si”、“Zhejiang”、“Database”等；
- 列表：字符串构成的**有序集合**，**元素可以重复**，按照插入顺序排序，用一对小括号括起来，如（“Zhang”，“Li”，“Wang”，“Li”）；



2.3 值的设计与结构化

1、值的数据类型

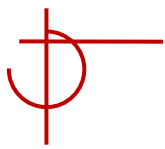
- 集合：集合是字符串类型的无序集合，集合成员是唯一的，**不能出现重复的数据**，用一对大括号括起来，如 { “Zhang” , “Li” , “Wang” , “Liu” }；
- 有序集：有序集合和集合一样也是字符串类型元素的集合，**且不允许重复的成员**。不同的是**每个元素都会关联一个double类型的数**，通过这个数来为集合中的成员进行从小到大的排序；
- 哈希映射：该数据类型是一个字符串类型的Key和Value的映射表，特别适合用于存储对象；
- 位数组：二进制整数构成的数组。



2.3 值的设计与结构化

2、值的操作

根据键对值进行读取、写入和删除，**查询操作则需要开发者编写程序来实现**，或调用API实现相应查询。



2.3 值的设计与结构化

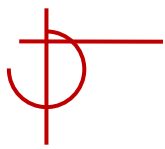
3、值的结构化

为了提高访存性能，可以采用结构化的数据作为值键值数据库的值，比如Json格式。

结构化的数据（Json）作为一个键值：

```
{"firstName": "Brett",  
  "lastName": "McLaughlin"}
```

但是，过于复杂的结构也不利于数据的存取，因为许多数据不是必须的。



2.4 键值数据库的特点

1、数据模型简单

通过键值对存储数据，不需要设计复杂的数据模型、纲要，也不需要为每个属性指定数据类型。

(Students:S01:Sname, 张利)

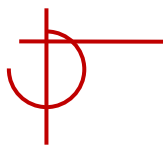
(Students:S02:Sname, 王芳)

(Students:S03:Sname, 范诚欣)

(Students:S04:Sname, 李铭)

(Students:S05:Sname, 黄佳宇)

(Students:S06:Sname, 仇星星)

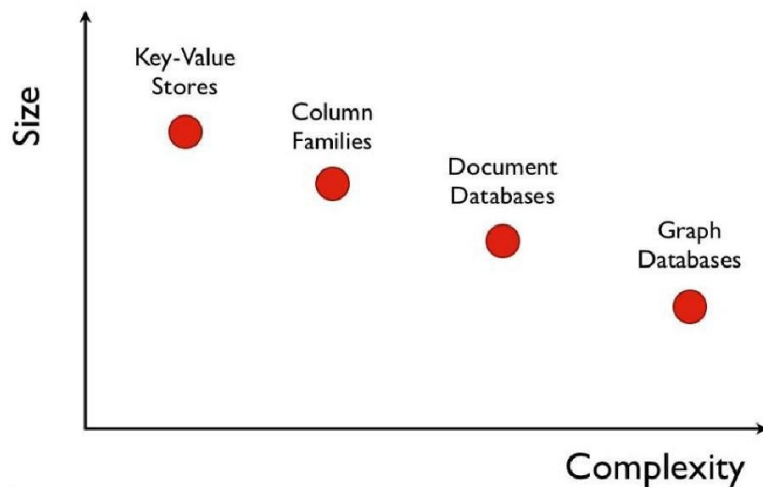


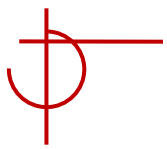
2.4 键值数据库的特点

2、速度快

采用简单的数据结构，并通过缓存提高了访问速度，适合于高吞吐量的数据密集型操作。

Focus Of Different Data Models



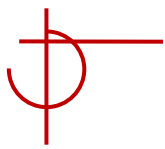


2.4 键值数据库的特点

3、易于缩放

能够根据系统的负载量，动态地增加和减少服务器节点，而服务器节点之间的关系可以有：

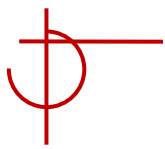
- 主从式复制：一个主服务器，多个从服务器。
- 无主式复制：服务器之间平等。



2.4 键值数据库的特点

4、自身的局限性

- 1) 只能通过键来查询数据：
 - 键值数据库不支持在不知道键名的情况下查找相关信息。
 - 通过辅助工具来实现查找。
- 2) 不支持某个范围的值。
- 3) 不支持SQL式的标准查询语言。

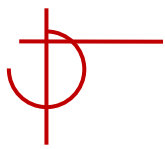


2.5 Redis键值数据库

1、Redis概述

Redis是基于内存与可持久化的键值数据库，多用在缓存方面。

- 支持数据的持久化，可以将内存中的数据保存在磁盘中，重启的时候可以再次加载进行使用。
- 支持丰富多样的数据类型，如list，set，zset，hash等数据结构。
- 支持master-slave模式的数据备份。
- 可视化管理工具：Redis-desktop-manager



2.5 Redis键值数据库

2、Redis的数据类型

Redis键值数据库支持的数据类型包括：

- 字符串：字符串或数值型的单个值
- 列表：字符串构成的**有序集合，允许重复**
- 集合：不同元素构成的**无序集合，不允许重复**
- 有序集：不同的元素按照特定顺序构成的集合
- 哈希映射：带有键值型特征的数据结构
- 位数组：二进制整数构成的数组。



2.5 Redis键值数据库

1) String数据类型

String 是 Redis 最基本的数据类型，一个 key 对应一个 value，可以存储任何数据，比如jpg图片或者序列化的对象，最大能存储 512MB。数值型数据作为字符串处理。

–SET key value: 设置指定 key 的值

- SET firstkey “redis” //firstkey是键, redis是值

–GET key: 获取指定 key 的值。

- GET firstkey

–GETRANGE key start end:返回 key 中字符串值的子串

- GETRANGE firstkey 0 5



2.5 Redis键值数据库

2) Hash数据类型

哈希Hash是一个String类型的field和value的映射表，用于存储对象,每个hash可以存储 $2^{32}-1$ 键值对（40多亿）。

–HSET key field value:将哈希表 key 中的 field 的值设为 value

- HSET secondkey name “Zhang”

–HGET key field: 获取指定 key 的指定字段的值。

- HGET secondkey name

–HMSET key field1 value1 field2 value2...: 同时设置多个Hash

- HMSET secondkey name “redis tutorial” description “redis basic commands for caching” likes 20 visitors 23000: 设置指定field和value的值

–HGETALL key:获得哈希表中指定key的所有字段和值。

- HGETALL secondkey



2.5 Redis键值数据库

3) List数据类型

List列表是简单的字符串列表，按照插入顺序排序。你可以添加一个元素到列表的头部或者尾部，一个列表最多可以存储40多亿个成员。

–LPUSH key value1 [value2]:将一个或多个值插入到列表头部（左部）

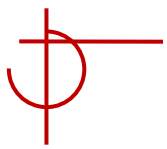
- Lpush `thirdkey` 'C++' 'database'

–LRANGE key start stop:获取列表指定范围内的元素

- Lrange `thirdkey` 0 2

–RPUSH key value1 [value2]:将一个或多个值插入到列表尾部（右部）

- Rpush `thirdkey` orange



2.5 Redis键值数据库

4) Set集合

Set是String类型的无序集合，**集合成员是唯一的**，不能出现重复的数据。通过哈希表实现，每个集合可存储40多亿个成员。

–SADD key member1 [member2]: 向集合添加一个或多个成员

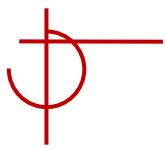
- Sadd fourthkey redis mysql:添加了两个

–SMEMBERS key:返回集合中的所有成员。

- Smembers fourthkey

–SREM key member1 [member2]: 移除集合中一个或多个成员

- Srem fourthkey redis



2.5 Redis键值数据库

5) Sorted Set有序集合

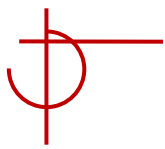
Sorted Set有序集合和集合一样也是String类型元素的集合，且不允许重复的成员。不同的是每个元素都会关联一个double类型的数，Redis通过这个数来为集合中的成员进行从小到大的排序。

–ZADD key **score1 member1** [score2 member2]: 向有序集合添加一个或多个成员，或者更新已存在成员的数值。

- Zadd **fifthkey** 1 redis
- Zadd **fifthkey** 2 mysql

–ZRANGE key start stop [WITHSCORES]:通过索引区间返回有序集合指定区间内的成员

- Zrange **fifthkey** 0 5

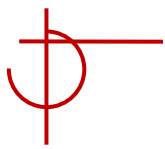


2.5 Redis键值数据库

6) 键命令

Redis 键命令用于管理 redis 的键。

- Keys *: 列出所有的key
- DEL key : 用于在 key 存在时删除 key。
 - Del firstkey
- EXISTS key : 检查给定 key 是否存在。
- MOVE key db: 将当前数据库的 key 移动到给定的数据库 d b 当中
- RENAME key newkey: 修改 key 的名称
- TYPE key: 返回 key 所储存的值的类型



2.5 Redis键值数据库

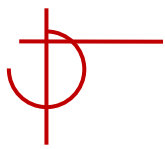
7) 数据库命令

Redis数据库的默认命名都是从0开始递增的数字（不支持自定义数据库名），默认支持16个数据库，默认使用0号数据库。

–SELECT index: 切换到指定的数据库，index 表示数据库索引号，以0作为起始索引值。

–FLUSHALL: 清空所有数据库的所有数据

–FLUSHDB: 清空当前所在数据库的数据



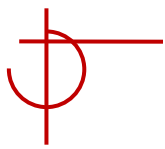
2.5 Redis键值数据库

3、Redis事务

Redis事务的定义方法：

- 执行MULTI：开始一个事务。
- 命令入队。
- 执行EXEC：依次执行队列中的命令。

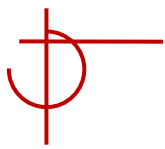
单个 Redis 命令的执行是原子性的，但 Redis 没有在事务上增加任何维持原子性的机制，所以 Redis 事务的执行并不是原子性的。**事务可以理解为一个打包的批量执行脚本，但批量指令并非原子化的操作，中间某条指令的失败不会导致前面已做指令的回滚，也不会造成后续的指令不做。**



2.5 Redis键值数据库

```
redis 127.0.0.1:7000> multi
OK
redis 127.0.0.1:7000> set a aaa
QUEUED
redis 127.0.0.1:7000> set b bbb
QUEUED
redis 127.0.0.1:7000> set c ccc
QUEUED
redis 127.0.0.1:7000> exec
1) OK
2) OK
3) OK
```

如果在 set b bbb 处失败，set a 已成功不会回滚，set c 还会继续执行。

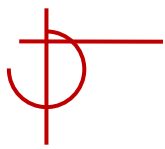


2.5 Redis键值数据库

3、Redis事务

Redis事务可以一次执行多个命令，并且带有以下三个重要的保证：

- 1) 批量操作在发送 EXEC 命令前被放入队列缓存；
- 2) 收到 EXEC 命令后进入事务执行，事务中任意命令执行失败，其余的命令依然被执行（非原子性）；
- 3) 在事务执行过程，其他客户端提交的命令请求不会插入到事务执行命令序列中。



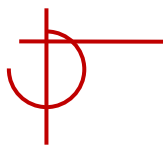
2.5 Redis键值数据库

4、嵌入在JAVA语言中

将Redis作为后台数据库，以JAVA作为开发语言，可以开发数据库应用程序，此时只需要在JAVA中安装Redis 驱动程序（一个Jar文件）

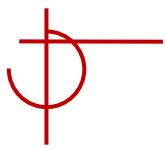
```
import redis.clients.jedis.Jedis;

public class RedisJava {
    public static void main(String[] args) {
        //连接本地的 Redis 服务
        Jedis jedis = new Jedis("localhost");
        System.out.println("连接成功");
        //查看服务是否运行
        System.out.println("服务正在运行: "+jedis.ping());
    }
}
```



2.5 Redis键值数据库

```
public class RedisListJava {  
    public static void main(String[] args) {  
        //连接本地的 Redis 服务  
        Jedis jedis = new Jedis("localhost");  
        System.out.println("连接成功");  
        //存储数据到列表中  
        jedis.lpush("site-list", "Runoob");  
        jedis.lpush("site-list", "Google");  
        jedis.lpush("site-list", "Taobao");  
        // 获取存储的数据并输出  
        List<String> list = jedis.lrange("site-list", 0 ,2);  
        for(int i=0; i<list.size(); i++) {  
            System.out.println("列表项为: "+list.get(i));  
        }  
    }  
}
```

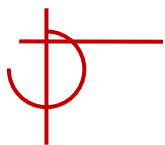



2.6 应用案例

将关系型数据库转换为Redis键值数据库

Students基本表

Sno	Sname	Ssex	Sage	Dno
S01	王建平	男	21	D01
S02	刘华	女	19	D01
S03	范林军	女	18	D02
S04	李伟	男	19	D03
S05	黄河	男	18	D03
S06	长江	男	20	D03



2.6 应用案例

Reports基本表

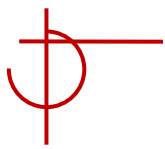
Sno	Cno	Grade
S01	C01	92
S01	C03	84
S02	C01	90
S02	C02	94
S02	C03	82
S03	C01	72
S03	C02	90
S04	C03	75

Courses基本表

Cno	Cname	Pre_Cno	Credits
C01	英语		4
C02	数据结构	C05	2
C03	数据库	C02	2
C04	DB_设计	C03	3
C05	C++		3
C06	网络原理	C07	3
C07	操作系统	C05	3

Depts基本表

Dno	Dname
D01	自动化
D02	计算机
D03	数学
D04	通信
D05	电子



2.6 应用案例

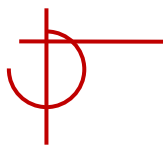
Students基本表对应的键值对

(Students:S01:Sname, 王建平)
(Students:S02:Sname, 刘华)
(Students:S03:Sname, 范林军)
(Students:S04:Sname, 李伟)
(Students:S05:Sname, 黄河)
(Students:S06:Sname, 长江)

(Students:S01:Sex, 男)
(Students:S02:Sex, 女)
(Students:S03:Sex, 女)
(Students:S04:Sex, 男)
(Students:S05:Sex, 男)
(Students:S06:Sex, 男)

(Students:S01:Sage, 21)
(Students:S02:Sage, 19)
(Students:S03:Sage, 18)
(Students:S04:Sage, 19)
(Students:S05:Sage, 18)
(Students:S06:Sage, 20)

(Students:S01:Sage, D01)
(Students:S02:Sage, D01)
(Students:S03:Sage, D02)
(Students:S04:Sage, D03)
(Students:S05:Sage, D03)
(Students:S06:Sage, D03)



2.6 应用案例

Reports基本表对应的键值对

```
(Reports:S01:Cno:Grade, C01 92, C03 84)
(Reports:S02:Cno:Grade, C01 90, C02 94,C03 82)
(Reports:S03:Cno:Grade, C01 72, C02 90)
(Reports:S04:Cno:Grade, C03 75)
```

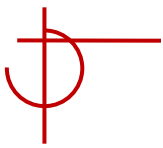
Courses基本表对应的键值对

```
(Courses:C01:Cname,英语)
(Courses:C02:Cname,数据结构)
(Courses:C03:Cname,数据库)
(Courses:C02:Pre_Cno,C05)
(Courses:C03:Pre_Cno,2)
(Courses:C04:Pre_Cno,3)
```

.....

Depts基本表对应的键值对

```
(Depts:D01:Dname, 自动化)
(Depts:D02:Dname, 计算机)
(Depts:D03:Dname, 数学)
(Depts:D04:Dname, 通信)
(Depts:D05:Dname, 电子)
```



Chapter over