Tedu.cn
达内教育

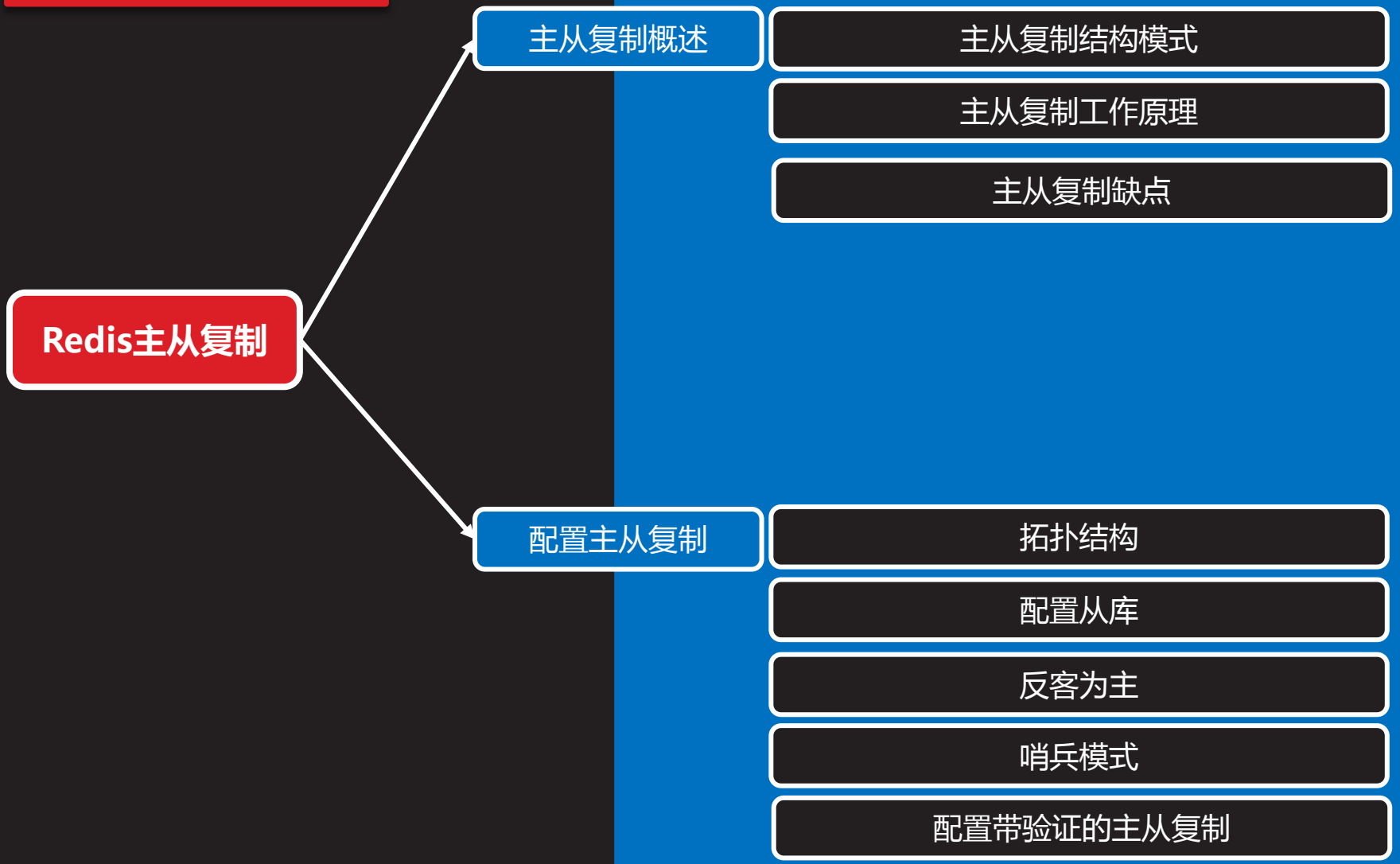
数据库管理

NSD NoSQL DAY03

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	Redis主从复制
	10:30 ~ 11:20	
	11:30 ~ 12:00	持久化(RDB/AOF)
下午	14:00 ~ 14:50	
	15:00 ~ 15:50	数据类型
	16:10 ~ 17:00	
	17:10 ~ 18:00	总结和答疑

Redis主从复制

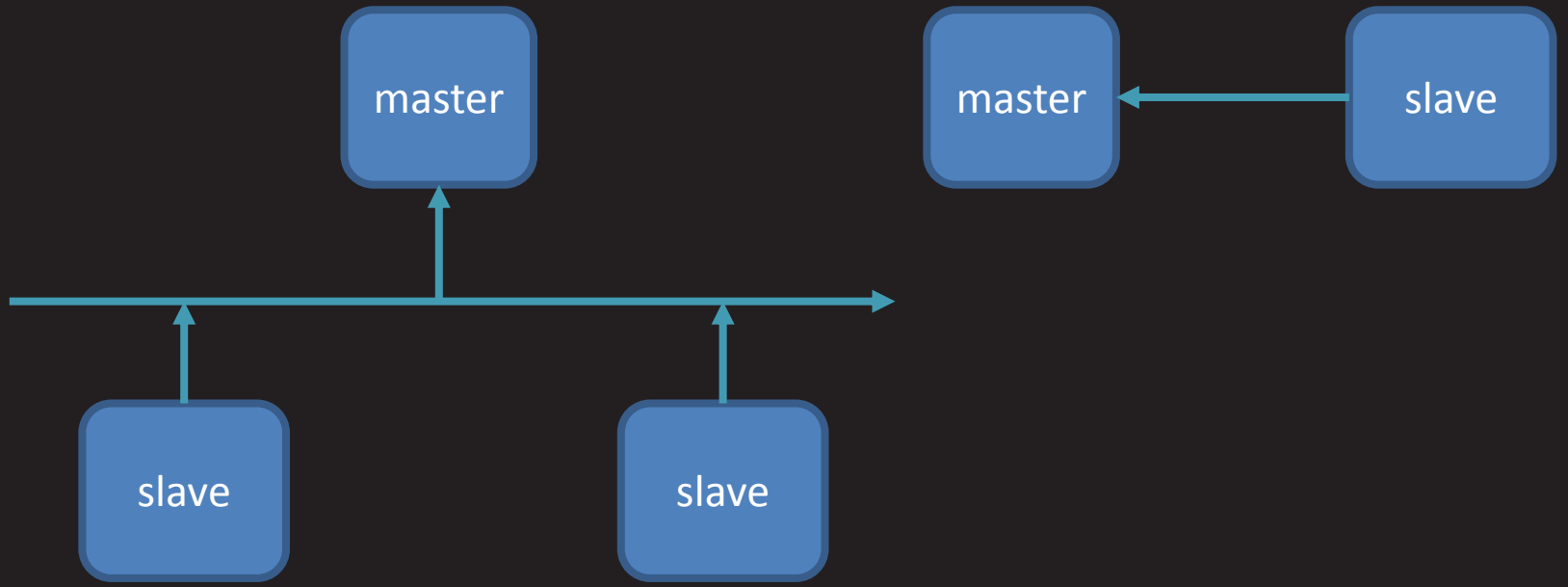

Redis主从复制
主从复制概述
主从复制结构模式
主从复制工作原理
主从复制缺点
配置主从复制
拓扑结构
配置从库
反客为主
哨兵模式
配置带验证的主从复制

Tedu.cn
达内教育

主从复制概述

主从复制结构模式

- 结构模式
  - 一主一从
  - 一主多从
  - 主从从



知识讲解

主从复制工作原理

- 工作原理
  - Slave向maste发送sync命令
  - Master启动后台存盘进程，同时收集所有修改数据命令
  - Master执行完后台存盘进程后，传送整个数据文件到slave。
  - Slave接收数据文件后，将其存盘并加载到内存中完成首次完全同步
  - 后续有新数据产生时，master继续将新的所以收集到的修改命令依次传给slave，完成同步。

知识讲解

主从复制缺点

- 缺点
  - 网络繁忙，会产生数据同步延时问题
  - 系统繁忙，会产生数据同步延时问题

知识讲解

Tedu.cn
达内教育

配置主从复制

拓扑结构

- 主服务器数据自动同步到从服务器

知识讲解

配置从库

- 配置从库192.168.4.52/24
  - redis服务运行后，默认都是master 服务器
  - 修改服务使用的IP地址　bind　192.168.4.X

知识讲解

```
[root@redis52 ~]# redis-cli  -h 192.168.4.52
192.168.4.52:6379> info  replication    //查看主从配置信息
# Replication
role:master
connected_slaves:0
.....
192.168.4.52:6379>  SLAVEOF    192.168.4.51    6379
OK
192.168.4.52:6379> info replication
# Replication
role:slave
master_host:192.168.4.51
master_port:6379
```

命令行指定主库
SLAVEOF　主库IP地址　端口号

反客为主

- 反客为主
  - 主库宕机后，手动把从库设置为主库

[root@redis52 ~]# redis-cli -h 192.168.4.52

192.168.4.52:6379> SLAVEOF no one //设置为主库
OK

192.168.4.52:6379> info replication
Replication
role:master

知识讲解

哨兵模式

- 哨兵模式
  - 主库宕机后，从库自动升级为主库
  - 在slave主机编辑sentinel.conf文件
  - 在slave主机运行哨兵程序

知识讲解

```
[root@redis52 ~]# vim  /etc/sentinel.conf
sentinel  monitor  redis51  192.168.4.51   6379   1
:wq
[root@redis52 ~]# redis-sentinel   /etc/sentinel.conf
```

sentinel　monitor　主机名　ip地址　端口　票数
主机名：自定义
IP地址：master主机的IP地址
端　口：master主机 redis服务使用的端口
票　数：主库宕机后， 票数大于1的主机被升级为主库

配置带验证的主从复制

- 配置master主机
  - 设置连接密码 ，启动服务，连接服务

知识讲解

```
[root@redis51 ~]# sed  -n '70p;501p' /etc/redis/6379.conf
bind 192.168.4.51
requirepass 123456   //密码
[root@redis51 ~]#

[root@redis51 ~]# /etc/init.d/redis_6379 start
Starting Redis server...

[root@redis51 ~]# redis-cli  -h 192.168.1.111 -a 123456 -p 6379
192.168.4.51:6379>
```

配置带验证的主从复制(续1)

- 配置slave主机
  - 指定主库IP ，设置连接密码，启动服务

```
[root@redis52 ~]#  sed -n '70p;282p;289p' /etc/redis/6379.conf
bind 192.168.4.52
slaveof 192.168.4.51 6379 //主库IP 与端口
masterauth 123456  //主库密码
[root@redis52 ~]#
[root@redis52 ~]# /etc/init.d/redis_6379 start
Starting Redis server...
[root@redis52 ~]# redis-cli  -h 192.168.4.52
192.168.4.52:6379> INFO replication
# Replication
role:slave
master_host:192.168.4.51
master_port:6379
```

知识讲解

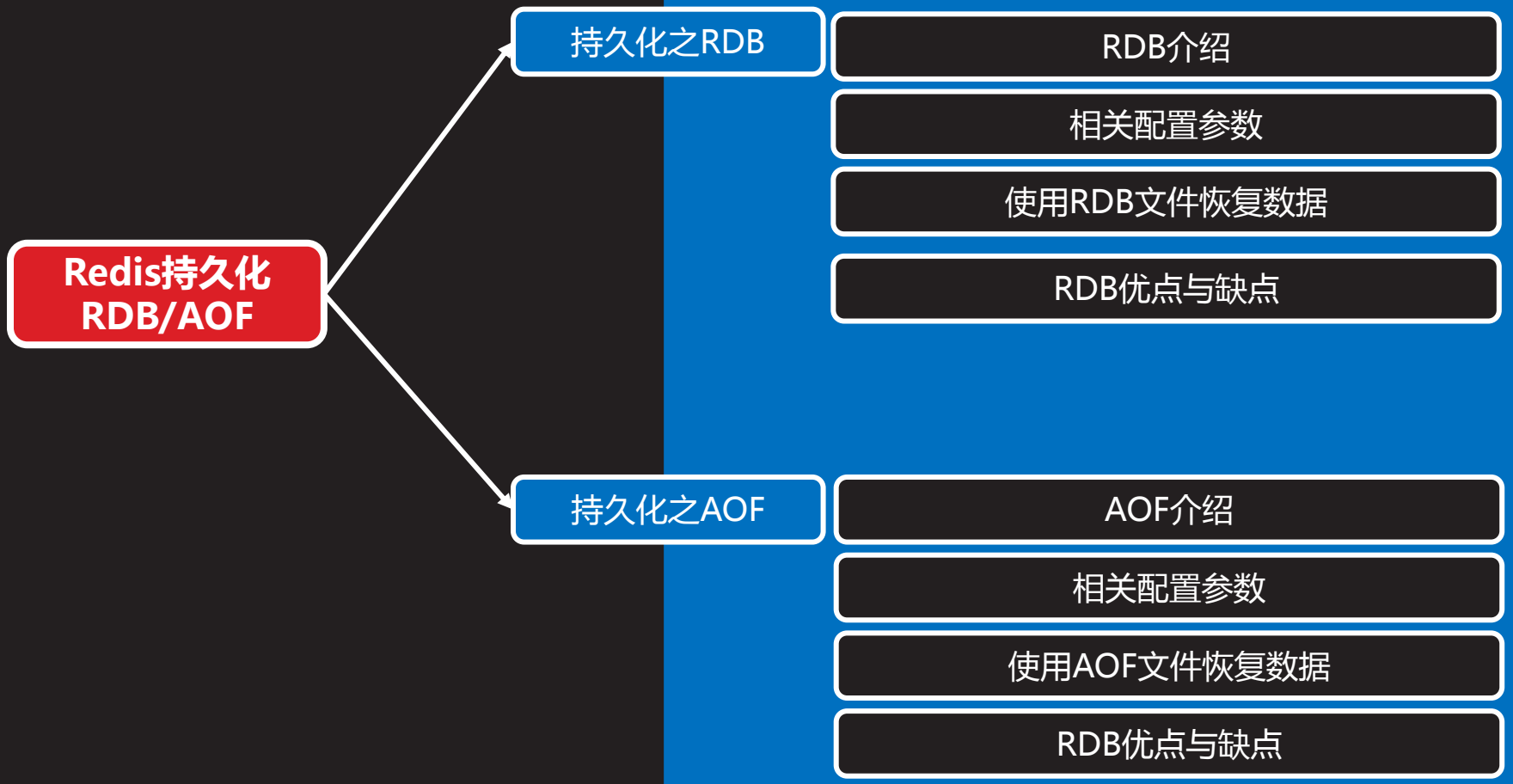
案例1：配置redis主从复制

具体要求如下：

- 啊
- 啊啊啊
- 啊啊啊啊啊
- 啊啊啊啊啊
- 啊啊啊啊啊啊啊

课堂练习

Redis持久化RDB/AOF



Tedu.cn
达内教育

持久化之RDB

RDB介绍

- 全称 Reids DataBase
  - 数据持久化方式之一
  - 在指定时间间隔内，将内存中的数据集快照写入硬盘。
  - 术语叫Snapshot快照。
  - 恢复时，将快照文件直接读到内存里。

知识讲解

相关配置参数

- 文件名
  - dbfilename “dump.rdb” //文件名
  - save “” //禁用RDB
- 数据从内存保存到硬盘的频率
  - save 900 1 // 900秒内且有1次修改存盘
  - save 300 10 //300秒内且有10次修改存盘
  - save 60 10000 //60秒内且有10000修改存盘
- 手动立刻存盘
  - > save //阻塞写存盘
  - > bgsave //不阻塞写存盘

知识讲解

相关配置参数(续1)

- 压缩
  - rdbcompression yes | no

- 在存储快照后，使用crc16算法做数据校验
  - rdbchecksum yes|no

- bgsave出错停止写操作,对数据一致性要求不高设置为no
  - stop-writes-on-bgsave-error yes|no

知识讲解

使用RDB文件恢复数据

知识讲解

- 备份数据
  - 备份dump.rdb 文件到其他位置
  - ~]# cp 数据库目录/dump.rdb 备份目录
- 恢复数据
  - 把备份的dump.rdb文件拷贝回数据库目录,重启redis服务
  - cp 备份目录/dump.rdb 数据库目录/
  - /etc/redid/redis_端口 start

RDB优点/缺点

- RDB优点
  - 持久化时，Redis服务会创建一个子进程来进行持久化，会先将数据写入到一个临时文件中，待持久化过程都结束了，再用这个临时文件替换上次持久化好的文件；整个过程中主进程不做任何IO操作，这就确保了极高的性能。
  - 如果要进程大规模数据恢复，且对数据完整行要求不是非常高，使用RDB比AOF更高效。
- RDB的缺点
  - 意外宕机，最后一次持久化的数据会丢失。

知识讲解

案例2 ：使用RDB文件恢复数据

要求如下：

- 啊啊啊
- 啊啊啊啊组
- 啊水水

课堂练习

持久化之AOF

AOF介绍

- 只追加操作的文件
  - Append Only File
  - 记录redis服务所有写操作。
  - 不断的将新的写操作，追加到文件的末尾。
  - 使用cat 命令可以查看文件内容

知识讲解

相关配置参数

- 文件名
  - appendfilename "appendonly.aof" //文件名
  - appendonly yes //启用aof ，默认no

- AOF文件记录，写操作的三种方式
  - appendfsync always //有新的写操作立即记录，性能差，完整性好。
  - appendfsync everysec //每秒记录一次，宕机时会丢失1秒的数据
  - appendfsync no //从不记录

知识讲解

相关配置参数(续1)

- 日志重写(日志文件会不断增大)，何时会触发日志重写？
  - redis会记录上次重写时AOF文件的大小，默认配置是当aof文件是上次rewrite后大小的1倍且文件大于64M时触发。
  - auto-aof-rewrite-percentage 100
  - auto-aof-rewrite-min-size 64mb

相关配置参数(续2)

- 修复AOF文件，
  - 把文件恢复到最后一次的正确操作

知识讲解

```
[root@redis53 6379]# redis-check-aof --fix appendonly.aof
0x              83: Expected \r\n, got: 6166
AOF analyzed: size=160, ok_up_to=123, diff=37
This will shrink the AOF from 160 bytes, with 37 bytes, to 123
bytes
Continue? [y/N]: y
Successfully truncated AOF
```

使用AOF文件恢复数据

知识讲解

- 备份数据
  - 备份dump.rdb 文件到其他位置
  - ~]# cp 数据库目录/appendonly.aof 备份目录
- 恢复数据
  - 把备份的dump.rdb文件拷贝回数据库目录,重启redis服务
  - Cp 备份目录/appendonly.aof 数据库目录/
  - /etc/redid/redis_端口 start

AOF优点/缺点

- RDB优点
  - 可以灵活的设置同步持久化appendfsync alwayls 或异步持久化appendfsync verysec
  - 宕机时，仅可能丢失1秒的数据

- RDB的缺点
  - AOF文件的体积通常会大于RDB文件的体积。执行fsync策略时的速度可能会比RDB 慢。

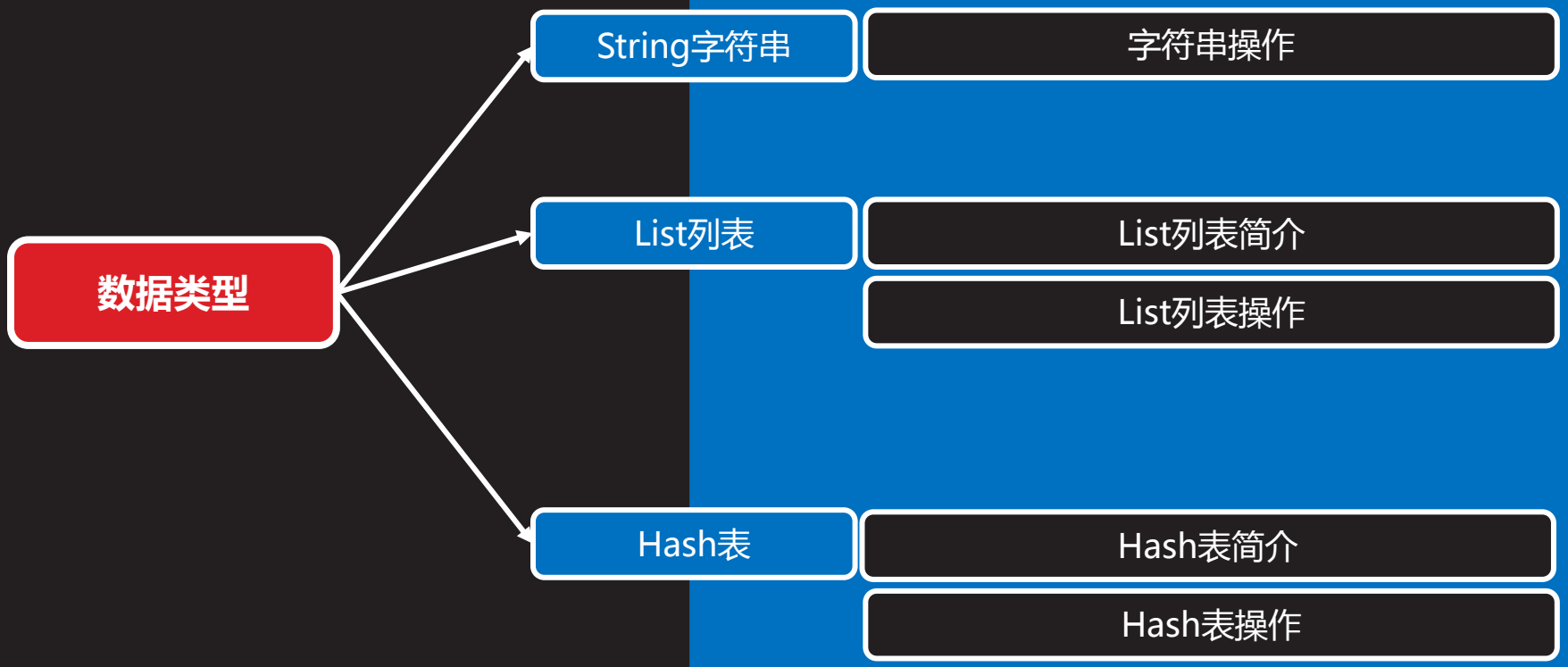
案例 3：使用AOF文件恢复数据

要求如下：

- 啊啊啊
- 啊啊啊啊组
- 啊水水

课堂练习

数据类型


数据类型
String字符串
字符串操作
List列表
List列表简介
List列表操作
Hash表
Hash表简介
Hash表操作

String字符串

字符串操作

- set key value [ex seconds] [px milliseconds] [nx|xx]
  - 设置key及值，过期时间可以设置为秒或毫秒为单位
  - nx只有key不存在，才对key进行操作
  - xx只有key已存在，才对key进行操作
- setrange key offset value
  - 从偏移量开始复写key的特定位的值

```
>set first "hello world"
>setrange first 6 “Redis”          //改写为hello Redis
```

- strlen key
  - 统计字串长度

```
>strlen first
```

知识讲解

字符串操作（续1）

知识讲解

- append key value
  - 字符存在则追加，不存在则创建key及value
  - 返回值为key的长度

```
>append myname jacob
```

- setbit key offset value
  - 对key所存储字串，设置或清除特定偏移量上的位(bit)
  - Value值可以为1或0，offset为0~2^32之间
  - key不存在，则创建新key

```
>setbit bit 0 1
>setbit bit 1 0
```

bit:第0位为1，第一位为0

字符串操作（续2）

- bitcount key
  - 统计字串中被设置为1的比特位数量

```
>setbit bits 0 1          //0001
>setbit bits 3 1          //1001
>bitcount bits            //结果为2
```

记录网站用户上线频率，如用户A上线了多少天等类似的数据

如用户在某天上线，则使用setbit，以用户名为key，将网站上线日为offset，并在该offset上设置1，最后计算用户总上线次数时，使用bitcount用户名即可

这样，即使网站运行10年，每个用户仅占用10*365比特位即456字节即可

```
>setbit peter 100 1        //网站上线100天用户登录了一次
>setbit peter 105 1        //网站上线105天用户登录了一次
>bitcount peter
```

知识讲解

字符串操作（续3）

知识讲解

- decr key
  - 将key中的值减1，key不存在则先初始化为0，再减1

```
>set test 10
>decr test
```

- decrby key decrement
  - 将key中的值，减去decrement

```
>set count 100
>decrby count 20
```

- get key
  - 返回key所存储的字符串值
  - 如果key不存在则返回特殊值nil
  - 如果key的值不是字串，则返回错误，get只能处理字串

字符串操作（续4）

- getrange key start end
  - 返回字串值中的子字串，截取范围为start和end
  - 负数偏移量表述从末尾计数，-1表示最后一个字符，-2表示倒数第二个字符

```
>set first  “hello,the world”
>getrange first -5 -1
>getrange first 0 4
```

知识讲解

字符串操作（续5）

- incr key
  - 将key的值加1，如果key不存在，则初始为0后再加1
  - 主要应用为计数器

    >set page 20

    >incr page
- incrby key increment
  - 将key的值增加increment

知识讲解

字符串操作（续6）

- incrbyfloat key increment
  - 为key中所储存的值加上浮点数增量 increment

```
>set num 16.1
>incrbyfloat num 1.1
```

- mget key [key...]
  - 一次获取一个或多个key的值，空格分隔，<具有原子性>
- mset key value [key value ...]
  - 一次设置多个key及值，空格分隔，<具有原子性>

知识讲解

Hash表

Hash表简介

- Redis hash是一个string类型的field和value的映射表
- 一个key可对应多个field，一个field对应一个value
- 将一个对象存储为hash类型，较于每个字段都存储成string类型更能节省内存

知识讲解

Hash表操作

- hset key field value
  - 将hash表中field值设置为value

```
>hset site google 'www.g.cn  '
>hset site baidu 'www.baidu.com'
```

- hget key filed
  - 获取hash表中field的值

```
>hget site google
```

知识讲解

Hash表操作（续1）

- hmset key field value [field value...]
  - 同时给hash表中的多个field赋值
    >hmset site google www.g.cn baidu www.baidu.com
- hmget key field [field...]
  - 返回hash表中多个field的值
    >hmget site google baidu
- hkeys key
  - 返回hash表中所有field名称
    >hmset site google www.g.cn baidu www.baidu.com
    >hkeys site

知识讲解

Hash表操作（续2）

知识讲解

- hgetall key
  - 返回hash表中所有field的值
- hvals key
  - 返回hash表中所有filed的值
    - >hvals key
- hdel key field [field...]
  - 删除hash表中多个field的值，不存在则忽略
    - >hdel site google baidu

Tedu.cn
达内教育

List列表

List列表简介

- Redis的list是一个字符队列
- 先进后出
- 一个key可以有多个值

知识讲解

List列表操作

知识讲解

- lpush key value [value...]
  - 将一个或多个值value插入到列表key的表头
  - Key不存在，则创建key

```
>lpush list a b c  //list1值依次为c b a
等同于lpush list a; lpush list b; lpush list c
```

- lrange key start stop
  - 从开始位置读取key的值到stop结束

```
>lrange list 0 2      //从0位开始，读到2位为止
>lrange list 0 -1     //从开始读到结束为止
>lrange list 0 -2     //从开始读到倒数第2位值
```

List列表操作（续1）

- lpop key
  - 移除并返回列表头元素数据，key不存在则返回nil

```
>lpop list                    //删除表头元素，可以多次执行
```

- llen key
  - 返回列表key的长度

知识讲解

List列表操作（续2）

- lindex key index
  - 返回列表中第index个值

    如lindex key 0 ; lindex key 2; lindex key -2
- lset key index value
  - 将key中index位置的值修改为value

    >lset list 3 test //将list中第3个值修改为test

知识讲解

List列表操作（续3）

- rpush key value [value...]
  - 将value插入到key的末尾

```
>rpush list3  a b c          //list3值为a b c
>rpush list3 d               //末尾插入d
```

- rpop key
  - 删除并返回key末尾的值
  - >rpush list3 a b c //list3值为a b c

```
>rpush list3 d               //末尾插入d
```

知识讲解

Tedu.cn
达内教育

其他操作

其他操作指令

- del key [key...]
  - 删除一个或多个key
- exists key
  - 测试一个key是否存在
- expire key seconds
  - 设置key的生存周期
- persist key
  - 设置key永不过期
- ttl key
  - 查看key的生存周期

知识讲解

其他操作指令（续1）

知识讲解

- keys 匹配
  - 找符合匹配条件的key，特殊符号用\屏蔽

```
>keys *                    //显示所有key
>keys h?llo        //匹配hello,hallo,hxllo等
>keys h*llo        //匹配hllo或heeello等
>keys h[ae]lo      //匹配hello和hallo
```

- flushall
  - 清空所有数据
- select id
  - 选择数据库，id用数字指定，默认数据库为0

```
>select 0
>select 2
```

其他操作指令（续2）

- move key db_id
  - 将当前数据库的key移动到db_id数据库中

```
>move key 1      //将key移动到1数据库中
```

- rename key newkey
  - 给key改名为newkey，newkey已存在时，则覆盖其值
- renamenx key newkey
  - 仅当newkey不存在时，才将key改名为newkey

知识讲解

其他操作指令（续3）

知识讲解

- sort key
  - 对key进行排序

```
>lpush cost 1 8 7 2 5
>sort cost                    //默认对数字排序，升序
>sort cost desc               //降序
>lpush test  “about”  “site”  “rename”
>sort test alpha              //对字符排序
>sort cost alpha limit 0 3          //排序后提取0-3位数据
>sort cost alpha limit 0 3 desc
>sort cost STORE cost2   //对cost排序并保存为cost2
```

- type key
  - 返回key的数据类型

案例2：常用Redis数据库操作指令

- 对Redis数据库各数据类型进行增删改查操作
  - 数据类型分别为Strings、Hash表、List列表
  - 设置数据缓存时间
  - 清空所有数据
  - 对数据库操作

课堂练习