

- 是什么
 - 上述问题的解决思路
 - 官网：
 - 定义：
 - 一句话：
 - 案例演示
 - 小总结

是什么

上述问题的解决思路

管道(pipeline)可以一次性发送多条命令给服务端，服务端依次处理完毕后，通过一条响应一次性将结果返回，通过减少客户端与**redis**的通信次数来实现降低往返延时时间。pipeline实现的原理是队列，先进先出特性就保证数据的顺序性。

官网：

<https://redis.io/docs/manual/pipelining/>

定义：

pipeline是为了解决RTT往返时，仅仅是将命令打包一次性发送，对整个Redis的执行不造成其他任何影响

一句话：

批处理命令变种优化措施，类似Redis的原生批命令(mget和mset)

案例演示

```

[myredis]# ls -l cmd.txt
-rw-r--r-- 1 root root 87 11月 18 11:34 cmd.txt
[myredis]#
[myredis]# cat cmd.txt
set k100 v100
set k200 v200
hset k300 name haha
hset k300 age 20
hset k300 gender male
[myredis]# cat cmd.txt | redis-cli -a 111111 --pipe
Warning: Using a password with '-a' or '-u' option on the command line interface may not be safe.
All data transferred. Waiting for the last reply...
Last reply received from server.
errors: 0, replies: 5
[myredis]# redis-cli -a 111111
Warning: Using a password with '-a' or '-u' option on the command line interface may not be safe.
127.0.0.1:6379> get k100
"v100"
127.0.0.1:6379> hget k300 name
"haha"
127.0.0.1:6379>

```

将所有命令保存到一个文件中

将文件内容通过管道的形式传入到redis

小总结

- pipeline与原生批量命令对比

1. 原生批量命令是原子性(例如: mset、mget), *pipeline*是非原子性的
2. 原生批量命令一次只能执行一种命令, pipeline支持批量执行不同命令
3. 原生批量命令是服务端实现, 而pipeline需要服务端与客户端共同完成

- pipeline与事务对比

1. 事务具有原子性, 管道不具有原子性
2. 管道一次性将多条命令发送到服务器, 事务是一条一条的发, 事务只有在接收到exec命令后才会执行, 管道不会
3. 执行事务时会阻塞其他命令的执行, 而执行管道中的命令时不会

- 使用pipeline注意事项

1. pipeline缓冲的指令只是会依次执行, 不保证原子性, 如果执行中指令发生异常, 将会继续执行后续的指令
2. 使用pipeline组装的命令个数不能太多, 不然数量过大客户端阻塞的时间可能过久, 同时服务端此时也被迫回复一个队列答复, 占用很多内存