# 概述

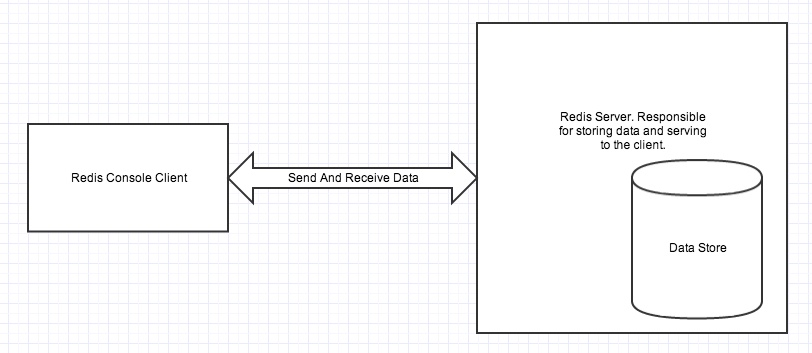
1. redis全称Remote Dictionary Server（远程字典服务器）；
2. Redis在TCP端口6379上监听到来的连接；
3. Value类型:string、hash、list、set、zset
4. Redis官网<https://redis.io/>

# 数据类型

# 架构

## 单服务结构

1. 采用C/S架构；



Client可以使控制台，也可以是基于Redis API的程序。

# 命令

参考中文手册<http://redisdoc.com/>

# 管道

1. 参考

<https://redis.io/topics/pipelining>

1. 定义：

This way it is possible to send multiple commands to the server without waiting for the replies at all, and finally read the replies in a single step.

1. 理解：

客户端在一次请求中发送多个命令而无需等待每一个命令的执行结果，这些命令的执行结果最终由服务器一次性发送给客户端。

1. 注意：

**IMPORTANT NOTE**: While the client sends commands using pipelining, the server will be forced to queue（排队） the replies, using memory. So if you need to send a lot of commands with pipelining, it is better to send them as batches having a reasonable number, for instance 10k commands, read the replies, and then send another 10k commands again, and so forth. The speed will be nearly the same, but the additional memory used will be at max the amount needed to queue the replies for this 10k commands.

1. 测试

参看本文档目录下的PipelineTest代码

# 恢复策略

有关Redis数据一致性的策略参考:

https://redis.io/topics/persistence

1. RDB机制

该策略采用定期备份（写快照snapshot）机制，每隔特定时间将内存中的数据写入外存。这种方式的缺点就是无法完全恢复最后一次备份到异常发生期间的数据。

1. AOF机制

该策略将所有在server端的write操作记录日志（包括写的数据），如果发生异常可以通过log恢复。这种策略的弊端就是log日志太大，恢复耗时。

1. Save命令

该策略主要依靠client端执行save命令将所有数据写入外存。

1. RDB+AOF

通过这两种策略恢复数据，如果发生异常，取出最近的一次snapshot和之后所有的操作日志进行恢复。

# 内存布局

# 传输协议

参考<https://redis.io/topics/protocol>

# 参考资料