Redis是一种基于客户端-服务端模型以及请求/响应协议的TCP服务。
这意味着通常情况下一个请求会遵循以下步骤：

- 客户端向服务端发送一个查询请求，并监听Socket返回，通常是以阻塞模式，等待服务端响应。
- 服务端处理命令，并将结果返回给客户端。

Redis 管道技术

Redis 管道技术可以在服务端未响应时，客户端可以继续向服务端发送请求，
并最终一次性读取所有服务端的响应。

实例

查看 redis 管道，只需要启动 redis 实例并输入以下命令：

```
$(echo -en "PING\r\n SET runoobkey redis\r\nGET runoobkey\r\nINCR visitor\r\nINCR visitor\r\nINCR visitor\r\n"; sleep 10) | nc localhost 6379

+PONG
+OK
redis
:1
:2
:3
```

以上实例中我们通过使用 PING 命令查看redis服务是否可用， 之后我们设置了 runoobkey 的值为 redis，然后我们获取 runoobkey 的值并使得 visitor 自增 3 次。
在返回的结果中我们可以看到这些命令一次性向 redis 服务提交，并最终一次性读取所有服务端的响应

管道技术的优势

管道技术最显著的优势是提高了 redis 服务的性能。

一些测试数据

在下面的测试中，我们将使用Redis的Ruby客户端，支持管道技术特性，测试管道技术对速度的提升效果。

```
require 'rubygems'
require 'redis'
def bench(descr)
start = Time.now
yield
```

```
puts "#{descr} #{Time.now-start} seconds"
end
def without_pipelining
r = Redis.new
10000.times {
    r.ping
}
end
def with_pipelining
r = Redis.new
r.pipelined {
    10000.times {
        r.ping
    }
}
end
bench("without pipelining") {
    without_pipelining
}
bench("with pipelining") {
    with_pipelining
}
```

从处于局域网中的Mac OS X系统上执行上面这个简单脚本的数据表明，开启了管道操作后，往返时延已经被改善得相当低了。

```
without pipelining 1.185238 seconds
with pipelining 0.250783 seconds
```

如你所见，开启管道后，我们的速度效率提升了5倍。