Redis事务

可以一次执行多个命令，本质是一组命令的集合。一个事务中的

所有命令都会序列化，按顺序地串行化执行而不会被其它命令插入，不许加塞

# 常用指令

## multi

标志一个事务块的开始

## exec

执行事务块内的所有命令

## discard

取消事务，放弃执行命令

## watch key(key . . . )

监听一个或多个key，在事务被执行前，这个值被改动，则事务被打断

## unwatch

取消对所有Key的监视

# 使用情况

## 正常执行



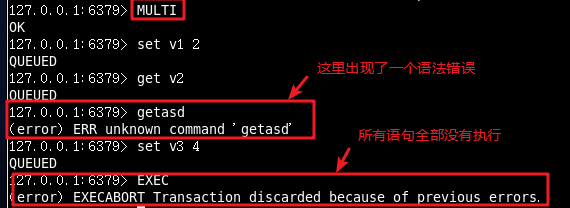
每条语句不会被立即执行，会被加入到队列，发出exec指令后会把这些指令依次执行

## 放弃事务



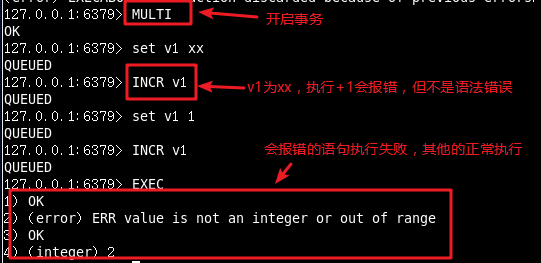
放弃事务后，所有语句不会被执行

## 全体连坐



出现一个语法错误，所有语句全部不会执行

## 冤头债主



如果有语句执行有运行时错误（不是语法错误），则该条语句会执行失败，其他语句不会受其影响

# switch监听

监控了key，如果key被修改了，后面一个事务的执行失效

## 悲观锁

 悲观锁(Pessimistic Lock), 顾名思义，就是很悲观，每次去拿数据的时候都认为别人会修改，所以每次在拿数据的时候都会上锁，这样别人想拿这个数据就会block直到它拿到锁。传统的关系型数据库里边就用到了很多这种锁机制，比如行锁，表锁等，读锁，写锁等，都是在做操作之前先上锁

## 乐观锁

  乐观锁(Optimistic Lock), 顾名思义，就是很乐观，每次去拿数据的时候都认为别人不会修改，所以不会上锁，但是在更新的时候会判断一下在此期间别人有没有去更新这个数据，可以使用版本号等机制。乐观锁适用于多读的应用类型，这样可以提高吞吐量，

乐观锁策略:提交版本必须大于记录当前版本才能执行更新

## Witch指令

Watch指令，类似乐观锁，事务提交时，如果Key的值已被别的客户端改变，比如某个list已被别的客户端push/pop过了，整个事务队列都不会被执行

通过WATCH命令在事务执行之前监控了多个Keys，倘若在WATCH之后有任何Key的值发生了变化，EXEC命令执行的事务都将被放弃，同时返回Nullmulti-bulk应答以通知调用者事务执行失败

## 取消witch监听

1. unwatch

这个指令执行后，所有的监听都会被取消

1. 一旦执行了exec之前加的监控锁都会被取消掉了