目录

[P28 28\_redis持久化之理论介绍 1](#_Toc128337165)

[为啥需要持久化 1](#_Toc128337166)

[RDB(Redis Database) &AOF(Append Only File) 2](#_Toc128337167)

[P29 29\_redis持久化之RDB简介 2](#_Toc128337168)

[1.是什么 2](#_Toc128337169)

[2.能干啥 2](#_Toc128337170)

[3.案例演示 2](#_Toc128337171)

[3.1Redis6.0.16以下（找redis的redis.conf文件） 3](#_Toc128337172)

[3.2Redis6.2以及Redis-7.0.0（保存时间的间隔频率变长） 4](#_Toc128337173)

[P30 30\_redis持久化之RDB配置说明 4](#_Toc128337174)

[3.3手动触发和被动触发 4](#_Toc128337175)

[P31 31\_redis持久化之RDB自动触发 5](#_Toc128337176)

[P32 32\_redis持久化之RDB手动触发 6](#_Toc128337177)

[P33 33\_redis持久化之RDB优缺点及数据丢失案例 8](#_Toc128337178)

[4.优势 8](#_Toc128337179)

[官网说明 8](#_Toc128337180)

[小总结 8](#_Toc128337181)

[5.劣势 9](#_Toc128337182)

[官网说明 9](#_Toc128337183)

[小总结 9](#_Toc128337184)

[P34 34\_redis持久化之RDB修复命令简介 9](#_Toc128337185)

[6.如何检查和修复dump.rdb文件 9](#_Toc128337186)

[P35 35\_redis持久化之RDB触发小结和快照禁用 10](#_Toc128337187)

[7.那些情况会触发RDB快照 10](#_Toc128337188)

[8.如何禁用快照 10](#_Toc128337189)

[P36 36\_redis持久化之RDB优化参数 10](#_Toc128337190)

[9.配置文件SNAPSHOTTING模块 10](#_Toc128337191)

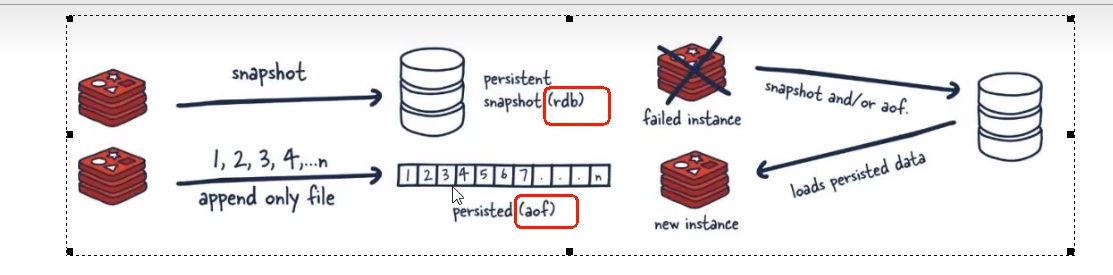
# P28 28\_redis持久化之理论介绍

官网：https:/redis.io/docs/manual/persistence/

## 为啥需要持久化

为什么需要持久化:所有数据都在内存中，一断电就没有了

## RDB(Redis Database) &AOF(Append Only File)

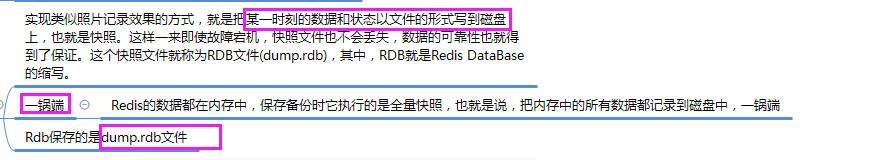


# P29 29\_redis持久化之RDB简介

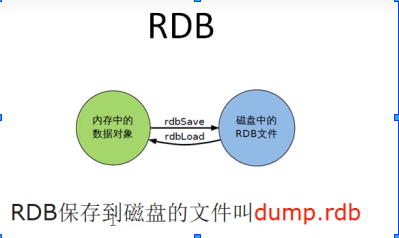
## 1.是什么

RDB持久性以指定的时间间隔执行数据集的时间点快照。

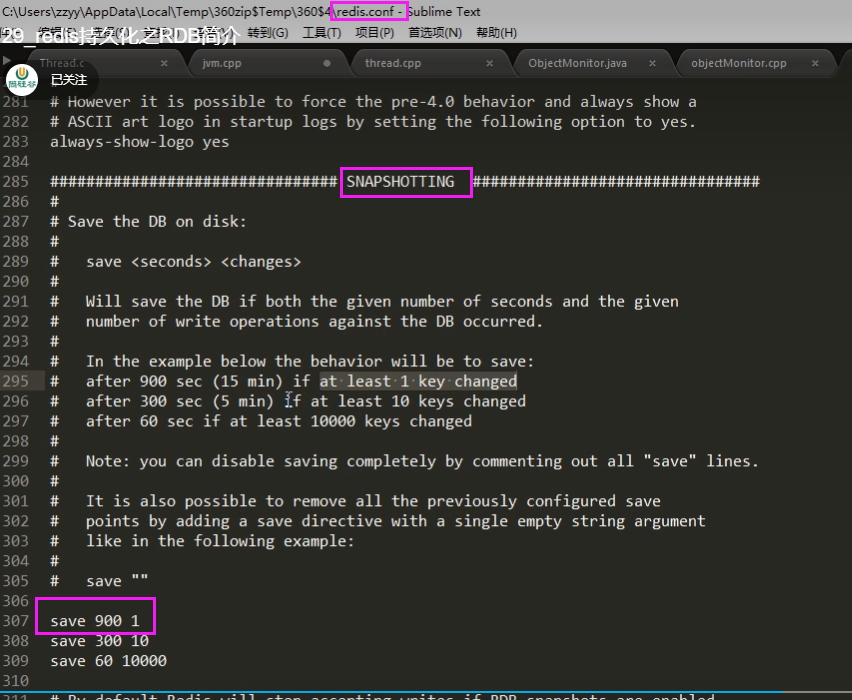
## 2.能干啥



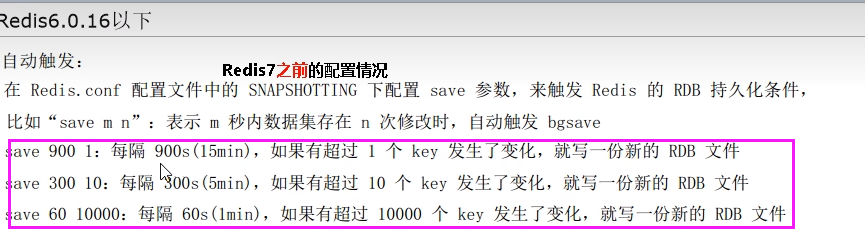
## 3.案例演示



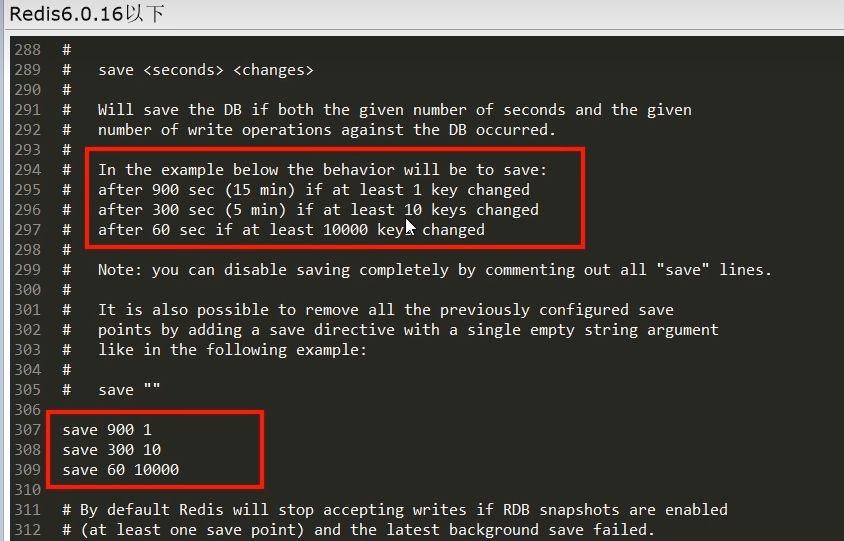
### 3.1Redis6.0.16以下（找redis的redis.conf文件）



900sec内有一次写入，就保存快照



### 3.2Redis6.2以及Redis-7.0.0（保存时间的间隔频率变长）



# P30 30\_redis持久化之RDB配置说明

### 3.3手动触发和被动触发

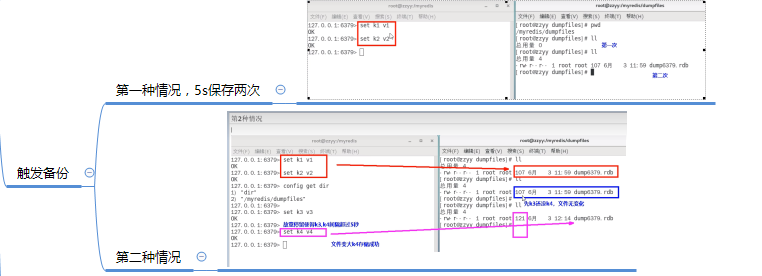
#### 3.3.1配置说明



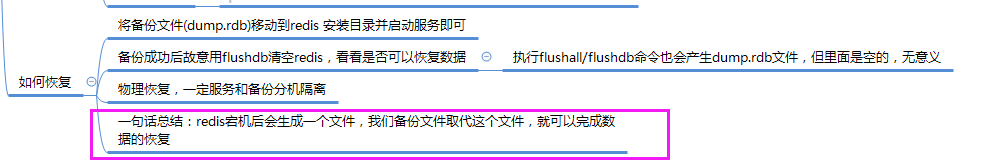
# P31 31\_redis持久化之RDB自动触发

#### 3.3.2自动触发

##### 触发备份



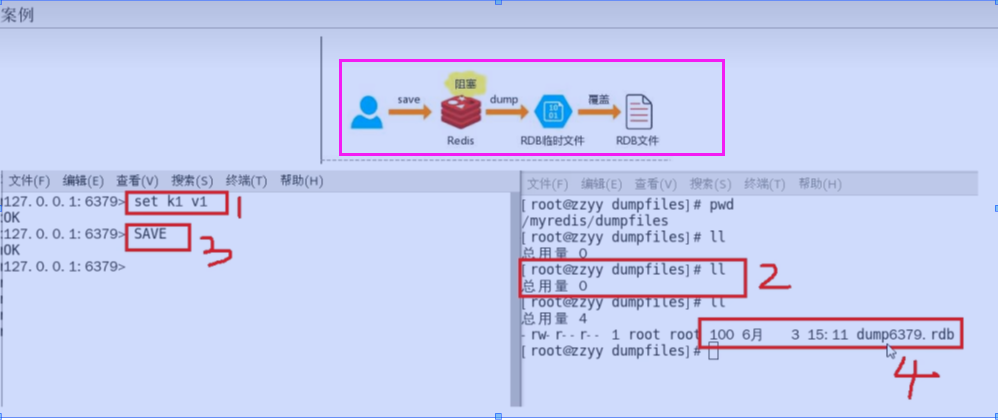
##### 如何恢复（不可以把备份文件dump.rdb和生产redis服务器放在同一台机器，必须分开各自存储，以防生产机物理损坏后备份文件也挂了）



# P32 32\_redis持久化之RDB手动触发

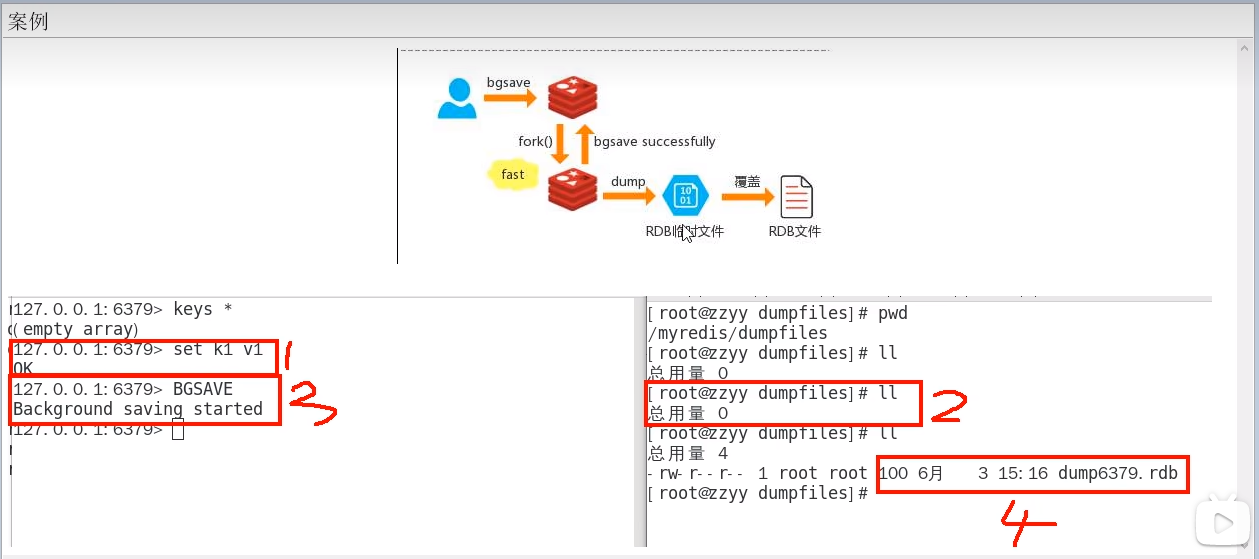
#### 3.3.3手动触发

##### Save



在主程序中执行会阻塞当前redis服务器，直到持久化工作完成执行save命令期间，Redis不能处理其他命令，线上禁止使用

##### Bgsave



Redis会在后台异步进行快照操作，不阻塞快照同时还可以响应客户端请求,该触发方式会fork一个子进程由子进程复制持久化过程.

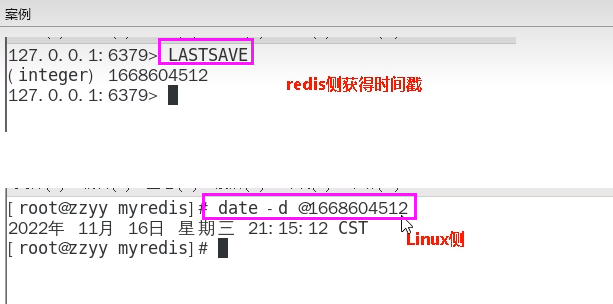
Redis会使用bgsave对当前内存中的所有数据做快照，这个操作是子进程在后台完成的，这就允许主进程同时可以修改数据。

##### Fork操作

在Linux程序中，fork ()会产生一个和父进程完全相同的子进程，但子进程在此后多会exec系统调用，出于效率考虑，尽量避免膨胀。

##### Lastsave

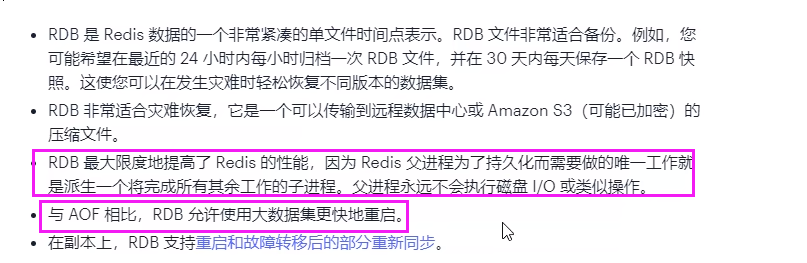
可以通过lastsave命令获取最后一次成功执行快照的时间



# P33 33\_redis持久化之RDB优缺点及数据丢失案例

## 4.优势

### 官网说明



### 小总结

适合大规模的数据恢复

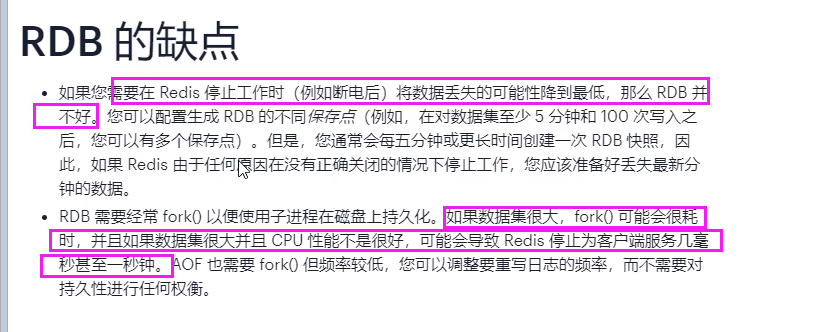
按照业务定时备份

对数据完整性和一致性要求不高

RDB文件在内存中的加载速度要比AOF快得多

## 5.劣势

### 官网说明



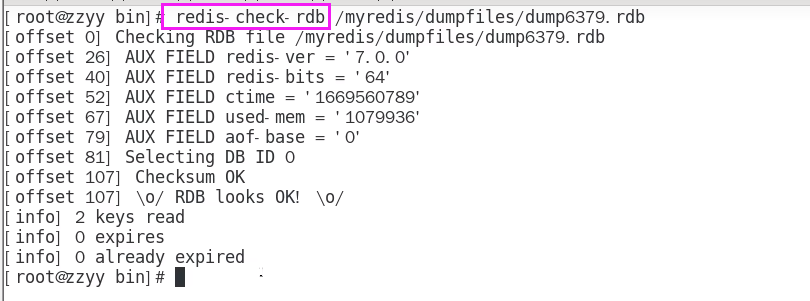
### 小总结

在一定间隔时间做一次备份，所以如果redis意外down掉的话，就会丢失从当前至最近一次快照期间的数据，快照之间的数据会丢失内存数据的全量同步，如果数据量太大会导致I/O严重影响服务器性能。

RDB依赖于主进程的fork，在更大的数据集中，这可能会导致服务请求的瞬间延迟。fork的时候内存中的数据被克隆了一份，大致2倍的膨胀性，需要考虑

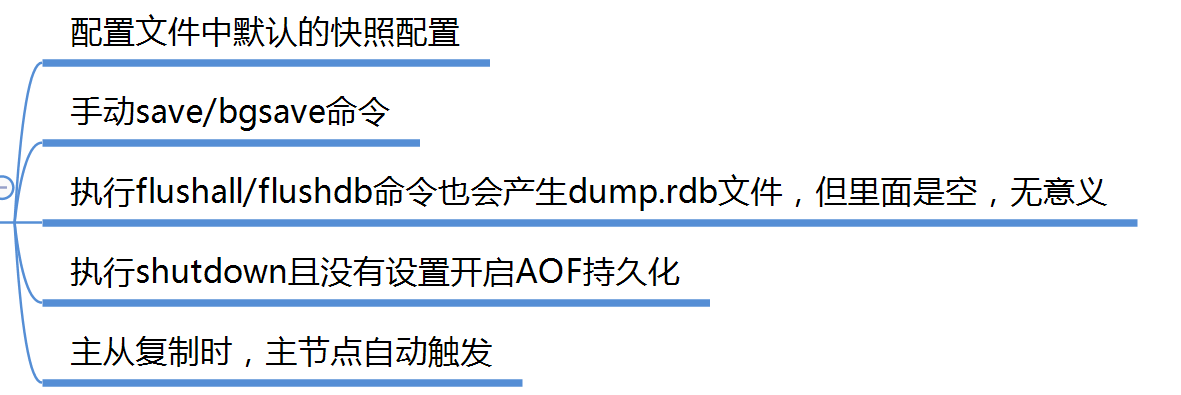
# P34 34\_redis持久化之RDB修复命令简介

## 6.如何检查和修复dump.rdb文件

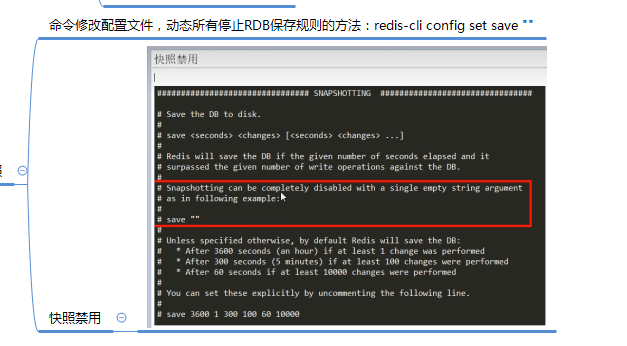


# P35 35\_redis持久化之RDB触发小结和快照禁用

## 7.那些情况会触发RDB快照



## 8.如何禁用快照



# P36 36\_redis持久化之RDB优化参数

## 9.配置文件SNAPSHOTTING模块



## 10．小总结