# RDB持久化

Redis是一个简直对数据库服务器，服务器中通常包含着任意个非空数据库，而每个非空数据库中又可以包含任意个键值对，为了方便起见，**我们将服务器中的非空数据库以及它们的键值对**统称为数据库状态。

Redis是内存数据库，它将数据库状态保存在内存中。因此，如果不想办法将内存中的数据保存到磁盘里面，一旦服务器退出，服务器中的数据库状态也会消失不见。

为了解决这个功能，redis提供了RDB持久化功能，将redis内存中的数据库状态保存到硬盘里面，避免数据以外丢失。

RDB持久化功能所生成的RDB文件时一个经过压缩的二进制文件，通过该文件可以还原生成RDB文件时的数据库状态。

## RDB文件的创建与载入

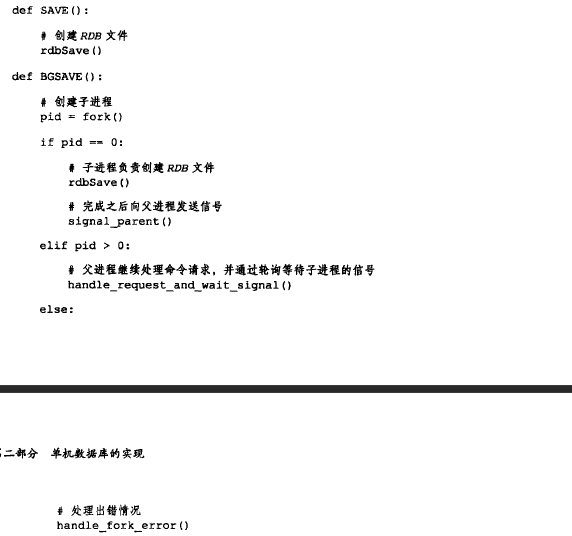
### RDB创建

有两个Redis命令可以用于生成RDB文件，一个时SAVE,一个时BGSAVE。

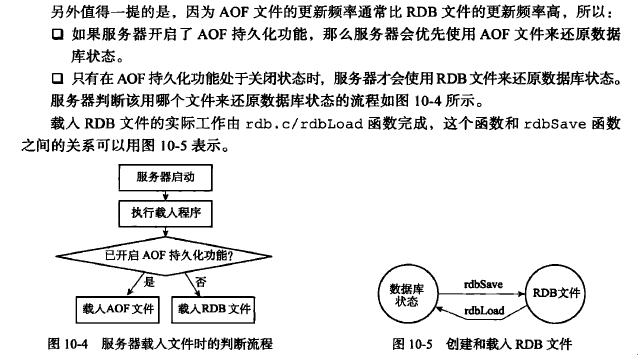
SAVE命令会阻塞Redis服务器进程，知道RDB文件创建完毕为止，在服务器进程阻塞期间，服务器不能处理任何命令请求。

BGSAVE命令会派生出一个子进程，然后由子进程负责创建RDB文件。

伪代码：



### RDB载入



### BGSAVE命令执行时的服务器状态

在BGSAVE命令执行期间，服务i其处理SAVE，BGSAVE，BGREWRITEAOF的方式会何平时有所不同。

SAVE： 会被服务器拒绝

BGSAVE： BGSAVE也会被服务器拒绝

BGREWRITEAOF：该命令和BGSAVE不能同时执行，因为这两个命令都会执行大量的磁盘写入操作。出于性能考虑，并发出两个这样的子进程并不合适。

## 自动间隔性保存

因为BGSAVE命令可以在不阻塞服务器的条件下进行，所以Redis允许用户通过设置服务器配置的save选项，让服务器每隔一段时间自动执行一次VGSAVE命令。

用户可以通过save选项设置多个保存条件，只要任意一个条件满足，服务器就会执行BGSAVE命令。

### 设置保存条件

Save选项保存在redisServer结构中：

Struct redisServer{

//…

//记录了保存条件的数组

Struct saveparam \*saveparams;

//…

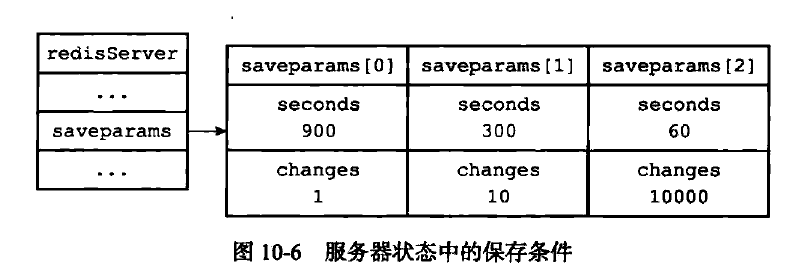
}

Struct saveparam{

Time\_t seconds; //秒数

Int changes; //修改数

}



### Dirty计数器和lastsave属性

Struct redisServer{

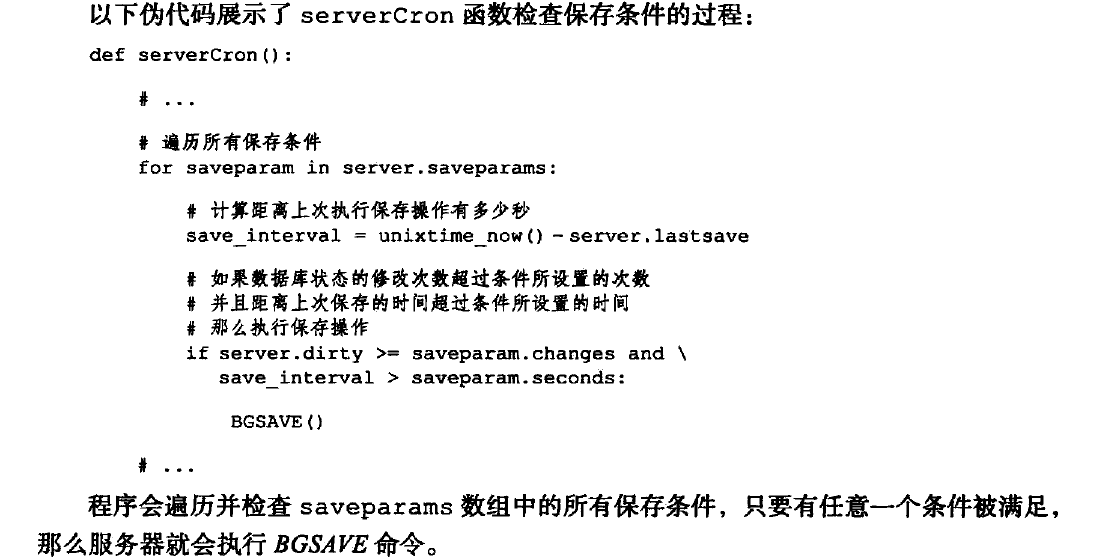
//…

Long long dirty; //修改器 ，记录距离上一次成功执行SAVE或者BGSAVE命令后，服务器对数据库状态（所有数据库）进行了多少次修改

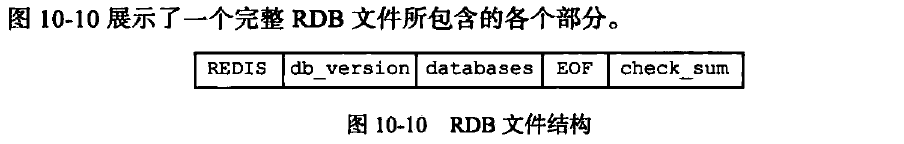
Time\_t lastsave; //上一次执行保存的时间

}

### 检查保存条件是否满足



## RDB文件结构



注意：

图中全大写的为常量，全小写的为变量和数据

RDN文件保存的是二进制数据

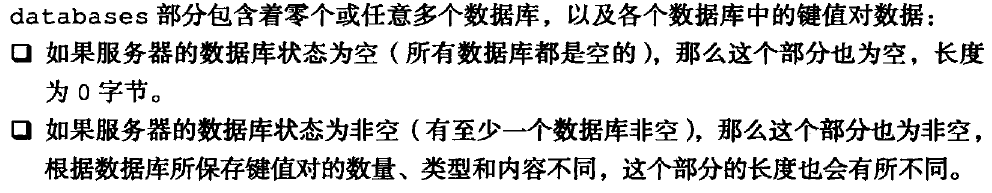
1. REDIS:

长度为5字节，保存“REDIS”五个字符，用来全速检查所载入的文件时RDB文件

1. db\_version:

长度为4字节，保存一个字符串整数，记录版本号

1. databases：



1. EOF：

长度为1字节，标志着RDB文件正文内容的结束，当读入程序读入这个值的时候，代表所有数据库的键值对都已经载入完毕了。

1. checksum：

长度为8字节，是一个无符号整数，保存着一个校验和。这个校验和是程序通过对REDIS，db\_version，databases，EOF四个部分的内容进行计算得出的。通过将载入数据计算出的校验和和check\_sum记录的校验和对比来检查是否出错或损坏。

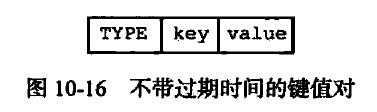
### database部分

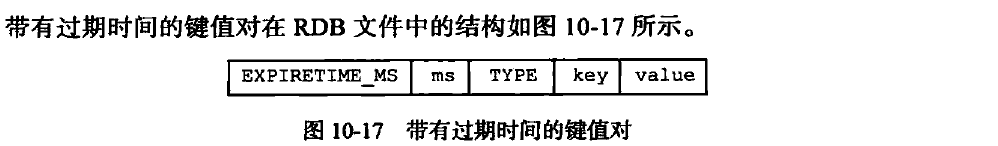


1. SELECTDB常量的长度为1字节，当读入这个常量时，代表接下来读入的时一个数据库号码。
2. db\_number保存着一个数据库号码，根据号码大小的不同，这个部分可以是1，2，5字节。读入后，服务器就会调用SELECT命令，进行数据库切换。
3. key\_value\_pairs保存了数据库中的所有键值对数据。

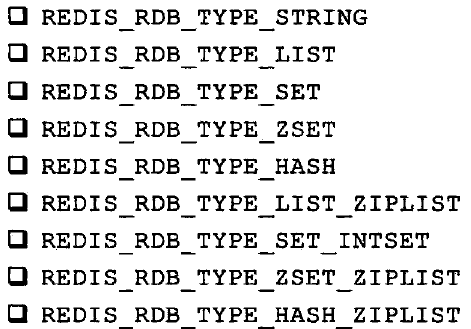
### key\_value\_pairs部分

RDB文件中的每一个key\_value\_pairs部分都保存了一个以上的简直对。如果键值对包含过期时间，那么键值对的过期时间也会被保存在内。





1. TYPE记录了value的类型,长度为1字节，值可以是以下常量中的一个



每个常量都代表一种对象类型或者底层编码。

1. key总是字符串对象，编码类型和REDIS\_RDB\_TYPE\_STRING类型value一样。
2. 根据TYPE类型的不同，保存value的结构和长度也会有所不同
3. EXPIRETIME\_MS常量的长度为1字节，它告诉服务器，接下来要读的僵尸一个以毫秒为单位的过期时间
4. ms是8字节长的带符号整数，记录以毫秒为单位的UNIX时间戳

### value的编码

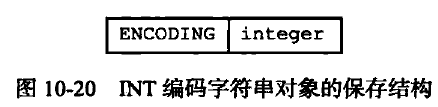
#### 字符串对象

TYPE值：REDIS\_RDB\_TYPE\_STRING

对象编码：REDIS\_ENCODING\_INT 或者 REDIS\_ENCODING\_RAW

保存结构：

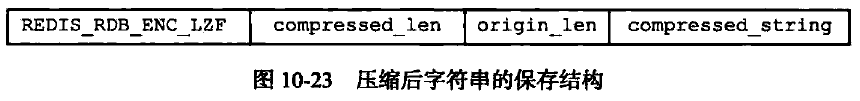
REDIS\_ENCODING\_INT编码：



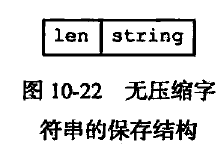
ENCODING可以是REDIS\_RDB\_ENC\_INT8, REDIS\_RDB\_ENC\_INT16, REDIS\_RDB\_ENC\_INT32

REDIS\_ENCODING\_RAW编码：

压缩（长度大于20字节会被压缩）：



不压缩（长度小于等于20字节）：

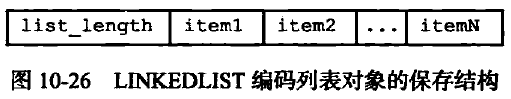


#### 列表对象

TYPE值：REDIS\_RDB\_TYPE\_LIST

ENCODING：REDIS\_ENCODING\_LINKEDLIST

保存结构：

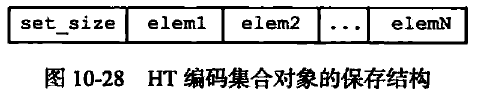


#### 集合对象

TYPE值：REDIS\_RDB\_TYPE\_SET

ENCODING：REDIS\_ENCODING\_HT

保存结构：

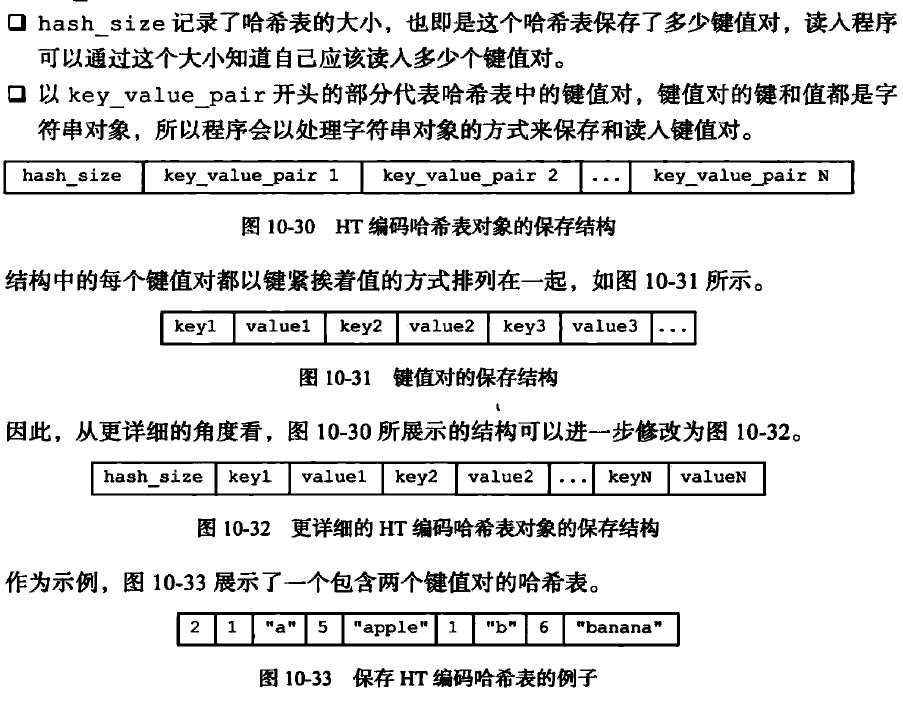


#### 哈希表对象

TYPE值：REDIS\_RDB\_TYPE\_HASH

ENCODING：REDIS\_ENCODING\_HT

保存结构：

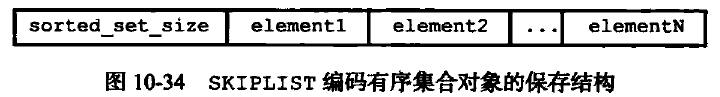


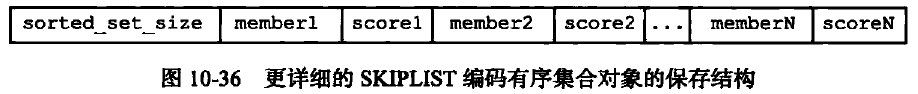
#### 有序集合对象

TYPE值：REDIS\_RDB\_TYPE\_ZSET

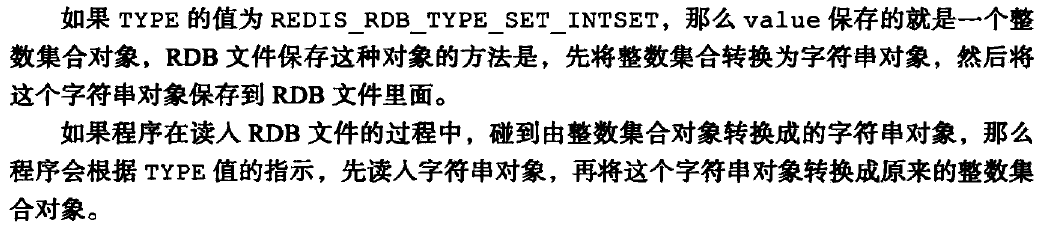
ENCODING：REDIS\_ENCODING\_SKIPLIST

保存结构：member为字符串对象，score为double浮点数





#### INTSET编码的集合



#### ZIPLIST编码的列表、哈希表或者有序集合

