图文 23 生产经验：在生产环境中，如何基于机器配置来合理设置Buffer Pool?





**生产经验：在生产环境中，如何基于机器配置来合理设置Buffer Pool?**

**如何提问**：每篇文章都有评论区，大家可以尽情留言提问，我会逐一答疑
**如何加群**：购买狸猫技术窝专栏的小伙伴都可以加入狸猫技术交流群，一个非常纯粹的技术交流的地方

具体加群方式，请参见目录菜单下的文档：**《MySQL专栏付费用户如何加群》**（购买后可见）

**1、生产环境中应该给buffer pool设置多少内存?**

今天这篇文章我们接着上一次讲解的Buffer Pool的一些内存划分的原理，来给大家最后总结一下，在生产环境中到底应该如何设置Buffer Pool的大小呢。

首先考虑第一个问题，我们现在数据库部署在一台机器上，这台机器可能有个8G、16G、32G、64G、128G的内存大小，那么此时buffer pool应该设置多大呢?

有的人可能会想，假设我有32G内存，那么给buffer pool设置个30GB得了，这样的话，MySQL大量的crud操作都是基于内存来执行的，性能那是绝对高!

但是这么想就大错特错了，你要知道，虽然你的机器有32GB的内存，但是你的操作系统内核就要用掉起码几个GB的内存!

然后你的机器上可能还有别的东西在运行，是不是也要内存?然后你的数据库里除了buffer pool是不是还有别的内存数据结构，是不是也要内存?所以上面那种想法是绝对不可取的!

如果你胡乱设置一个特别大的内存给buffer，会导致你的mysql启动失败的，他启动的时候就发现操作系统的内存根本不够用了!

所以通常来说，我们建议一个比较合理的、健康的比例，是给buffer pool设置你的机器内存的50%~60%左右

比如你有32GB的机器，那么给buffer设置个20GB的内存，剩下的留给OS和其他人来用，这样比较合理一些。

假设你的机器是128GB的内存，那么buffer pool可以设置个80GB左右，大概就是这样的一个规则。

**2、buffer pool总大小=(chunk大小 * buffer pool数量)的2倍数**

接着确定了buffer pool的总大小之后，就得考虑一下设置多少个buffer pool，以及chunk的大小了

此时要记住，有一个很关键的公式就是：buffer pool总大小=(chunk大小 * buffer pool数量)的倍数

比如默认的chunk大小是128MB，那么此时如果你的机器的内存是32GB，你打算给buffer pool总大小在20GB左右，那么你得算一下，此时你的buffer pool的数量应该是多少个呢?

假设你的buffer pool的数量是16个，这是没问题的，那么此时chunk大小 * buffer pool的数量 = 16 * 128MB = 2048MB，然后buffer pool总大小如果是20GB，此时buffer pool总大小就是2048MB的10倍，这就符合规则了。

当然，此时你可以设置多一些buffer pool数量，比如设置32个buffer pool，那么此时buffer pool总大小（20GB）就是（chunk大小128MB * 32个buffer pool）的5倍，也是可以的。

那么此时你的buffer pool大小就是20GB，然后buffer pool数量是32个，每个buffer pool的大小是640MB，然后每个buffer pool包含5个128MB的chunk，算下来就是这么一个结果了。

**3、一点总结**

我们再来做一点总结，就是说你的数据库在生产环境运行的时候，你必须根据机器的内存设置合理的buffer pool的大小，然后设置buffer pool的数量，这样的话，可以尽可能的保证你的数据库的高性能和高并发能力。

然后在线上运行的时候，buffer pool是有多个的，每个buffer pool里多个chunk但是共用一套链表数据结构，然后执行crud的时候，就会不停的加载磁盘上的数据页到缓存页里来，然后会查询和更新缓存页里的数据，同时维护一系列的链表结构。

然后后台线程定时根据lru链表和flush链表，去把一批缓存页刷入磁盘释放掉这些缓存页，同时更新free链表。

如果执行crud的时候发现缓存页都满了，没法加载自己需要的数据页进缓存，此时就会把lru链表冷数据区域的缓存页刷入磁盘，然后加载自己需要的数据页进来。

整个buffer pool的结构设计以及工作原理，就是上面我们总结的这套东西了，大家只要理解了这个，首先你对MySQL执行crud的时候，是如何在内存里查询和更新数据的，你就彻底明白了。

接着我们后面继续探索undo log、redo log、事务机制、事务隔离、锁机制，这些东西，一点点就把MySQL他的数据更新、事务、锁这些原理，全部搞清楚了，同时中间再配合穿插一些生产经验、实战案例。

**4、SHOW ENGINE INNODB STATUS**

当你的数据库启动之后，你随时可以通过上述命令，去查看当前innodb里的一些具体情况，执行SHOW ENGINE INNODB STATUS就可以了。此时你可能会看到如下一系列的东西：

```
Total memory allocated xxxx;
Dictionary memory allocated xxx
Buffer pool size   xxxx
Free buffers       xxx
Database pages     xxx
Old database pages xxxx
Modified db pages  xx
Pending reads 0
Pending writes: LRU 0, flush list 0, single page 0
Pages made young xxxx, not young xxx
xx youngs/s, xx non-youngs/s
Pages read xxxx, created xxx, written xxx
xx reads/s, xx creates/s, 1xx writes/s
Buffer pool hit rate xxx / 1000, young-making rate xxx / 1000 not xx / 1000
Pages read ahead 0.00/s, evicted without access 0.00/s, Random read ahead 0.00/s
LRU len: xxxx, unzip_LRU len: xxx
I/O sum[xxx]:cur[xx], unzip sum[16xx:cur[0]
```

下面我们给大家解释一下这里的东西，主要讲解这里跟buffer pool相关的一些东西。

（1）Total memory allocated，这就是说buffer pool最终的总大小是多少
（2）Buffer pool size，这就是说buffer pool一共能容纳多少个缓存页
（3）Free buffers，这就是说free链表中一共有多少个空闲的缓存页是可用的
（4）Database pages和Old database pages，就是说lru链表中一共有多少个缓存页，以及冷数据区域里的缓存页数量
（5）Modified db pages，这就是flush链表中的缓存页数量

（6）Pending reads和Pending writes，等待从磁盘上加载进缓存页的数量，还有就是即将从lru链表中刷入磁盘的数量、即将从flush链表中刷入磁盘的数量

（7）Pages made young和not young，这就是说已经lru冷数据区域里访问之后转移到热数据区域的缓存页的数量，以及在lru冷数据区域里1s内被访问了没进入热数据区域的缓存页的数量

（8）youngs/s和not youngs/s，这就是说每秒从冷数据区域进入热数据区域的缓存页的数量，以及每秒在冷数据区域里被访问了但是不能进入热数据区域的缓存页的数量

（9）Pages read xxxx, created xxx, written xxx，xx reads/s, xx creates/s, 1xx writes/s，这里就是说已经读取、创建和写入了多少个缓存页，以及每秒钟读取、创建和写入的缓存页数量

（10）Buffer pool hit rate xxx / 1000，这就是说每1000次访问，有多少次是直接命中了buffer pool里的缓存的

（11）young-making rate xxx / 1000 not xx / 1000，每1000次访问，有多少次访问让缓存页从冷数据区域移动到了热数据区域，以及没移动的缓存页数量

（12）LRU len：这就是lru链表里的缓存页的数量

（13）I/O sum：最近50s读取磁盘页的总数

（14）I/O cur：现在正在读取磁盘页的数量

**5、今日实践思考题**

今天留给大家的作业，就是每个人都对自己线上在运行的数据库执行上述命令，然后分析一下数据库的buffer pool的使用情况

这里要尤为关注的是free、lru、flush几个链表的数量的情况，然后就是lru链表的冷热数据转移的情况，然后你的缓存页的读写情况，这些代表了你当前buffer pool的使用情况。

最关键的是两个东西，一个是你的buffer pool的千次访问缓存命中率，这个命中率越高，说明你大量的操作都是直接基于缓存来执行的，性能越高。

第二个是你的磁盘IO的情况，这个磁盘IO越多，说明你数据库性能越差。

大家可以去观察一下，把自己的分析和思考发布在评论区里一起交流

**End**



---