

31 MySQL整体优化思路

更新时间：2019-10-29 11:06:52



“ 如果不想在世界上虚度一生，那就要学习一辈子。

——高尔基 ”

在前面的很多节中，我们都讲到了 MySQL 优化，这节就一起总结一下 MySQL 整体优化的思路。

1 硬件相关优化

在 MySQL 整体的优化环节中，硬件相关的优化必不可少，因此首先来聊聊这一方面的优化策略：

1.1 CPU 相关

A.关闭 CPU 节能，设定为最大性能模式。

原因是：考虑到在高并发之前没有任何连接的情况，机器可能会处于节电模式，高并发场景来临时可能导致处理不过来新的请求。

B.配置合理的 CPU 核数和选择合适的 CPU 主频。

原因是：

CPU 核数越多，支持的并发也越高；

CPU 主频越高，处理任务的速度越快。

1.2 内存相关

内存对 MySQL 数据库影响是非常大的。InnoDB 使用 InnoDB buffer pool 缓存数据、索引等内容，从而加快访问速度。因此 MySQL 运行的物理机上，内存配置也是比较重要的。

在应用数据库实例前，应该预估活跃的数据大小，然后根据这个合理配置数据库服务器内存的大小。

1.3 磁盘相关

对于 OLTP 的数据库，一般场景是 IO 密集型的操作。因此，对于这类情况，应该把更多的注意力放在提高磁盘 IO 上。

对于磁盘相关的优化，这里聊聊几种方法：

A. 使用 SSD(固态硬盘) 或者 Pcle SSD 设备；

为什么 SSD 比传统机械硬盘快？

传统的机械硬盘需要耗费长时间的磁头旋转和定位来查找数据。

而 SSD 其内部是由闪存组成的。闪存延迟低、功耗低。

因此 SSD 比传统机械硬盘更快。

B. 尽可能选择 RAID 10；

原因是 RAID 10 是先镜像在分区数据，将所有的硬盘分为两组，看作 RAID 0 的最低组合，然后将两组各自看做 RAID 1 运作。

RAID 10 有着不错的读取速度，而且拥有比 RAID 0 更高的数据保护性。

C. 设置阵列写策略为 Write Back。

有阵列卡时，建议设置阵列写策略为 Write Back(简称：WB)，WB 是指 RAID 控制器能够将写入的数据放入自身的缓存中，并把他们安排到后面再执行。

其优点是：不需要每次写入都等物理磁盘实际写入才完成，因此大大提升了写入速度。

2 系统层面优化

当硬件层优化的差不多了之后，系统层部分配置也应该去做一些优化。这里就来介绍部分系统层面的优化方法：

2.1 调整 I/O 调度算法

MySQL 运行的物理机上，I/O 调度算法建议使用：deadline/noop，尽量不使用 CFQ。

原因是 CFQ 把 I/O 请求按照进程分别放入进程对应的队列中。CFQ 的公平是针对进程而言，每一个提交 I/O 请求的进程都会有自己的 I/O 队列，以时间算法为前提，轮转调动队列，默认从当前队列中取出 4 个请求处理，然后处理下一个队列的 4 个请求，确保每个进程享有的 I/O 资源是均衡的。因此高并发场景，CFQ 很可能导致 I/O 的响应缓慢。（参考《深入浅出 MySQL》第二版：22.6 调整 I/O 调度算法）

2.2 文件系统选择

优先选用 xfs 或 ext4，坚决不用 ext3。

原因是：ext3 在 fsck 时需要耗费大量时间，文件越多，时间越长。

2.3 调整内核参数

```
vm.swappiness ≤ 10
```

降低使用 swap 的概率，但是尽量不要设置为 0，可能引起 OOM。

```
vm.dirty_ratio ≤ 5
```

这个参数指定了当文件系统缓存脏页数量达到系统内存百分之多少时（如 10%），系统不得不开始处理缓存脏页（因为此时脏页数量已经比较多，为了避免数据丢失需要将一定脏页刷入外存）；在此过程中很多应用进程可能会因为系统转而处理文件 IO 而阻塞。

```
vm.dirty_background_ratio ≤ 10
```

避免因为 IO 压力瞬间飙升导致内核进程卡死，操作系统 hung 住。这个参数指定了当文件系统缓存脏页数量达到系统内存百分之多少时，就会触发 pdflush/flush/kdmflush 等后台回写进程运行，将一定缓存的脏页异步地刷入外存。

3 MySQL 层优化

在之前的部分章节中，我们知道了，MySQL 自身很多地方是可以优化的，这里再对 MySQL 层的优化做一个总结：

3.1 参数优化

在启动 MySQL 之前，一些参数合理的设置，可以大大提升 MySQL 的性能。这里就来介绍一部分相对比较重要的参数：

```
innodb_buffer_pool_size
```

该参数控制 InnoDB 缓存表和索引数据的内存区域大小。对性能影响非常大，建议设置为机器内存的 50-80%。

```
innodb_flush_log_at_trx_commit
```

InnoDB 的 redo 日志刷新方式，对 InnoDB 的影响会很大。参数详细介绍可以复习[第 19 节](#)。

```
sync_binlog
```

控制累积多少个事务后才将二进制日志 **fsync** 到磁盘。参数详细介绍可以复习[第 19 节](#)。

```
innodb_file_per_table
```

开启独立表空间。在[第 29 节](#)详细介绍了参数的意义。

```
max_connection
```

最大连接数。不能设置的过小，防止客户端连接失败；也不能设置的过大，防止数据库内存资源过多消耗。

```
long_query_time
```

慢查询时间阈值。在[第 2 节](#)中有详细介绍和建议设置的大小。

```
query_cache_type  
query_cache_size
```

建议这两个参数都设置为 0。具体原因可以复习[第 26 节](#)。

3.2 MySQL 设计优化

- A. 使用 InnoDB 存储引擎，不建议使用 MyISAM 存储引擎；
- B. 预估表数据量和访问量，如果数据量或者访问量比较大，则需要提前考虑分库分表；
- C. 指定合适的数据库规范，在设计表、执行 SQL 语句时按照数据库规范来进行。（MySQL 操作规范将在下一节详细讲解）。

4 总结

这一节讲解了 MySQL 整体优化的思路。

具体从三个大方面聊到了优化方式：硬件、操作系统、MySQL server。

硬件方面需要考虑的优化内容是：

- 1、CPU；
- 2、内存；
- 3、磁盘。

操作系统层面需要考虑的优化为：

- 1、调整合适的 I/O 调度算法；
- 2、选择合适的文件系统；

3、调整部分会影响 MySQL 的内核参数。

在 MySQL 层

1、调整部分会影响性能的参数；

2、MySQL 设计时的优化。

5 问题

你觉得我们在使用 MySQL 时，还需要考虑哪些方面的优化？欢迎在留言区讨论。

6 参考资料

《MySQL 技术内幕：InnoDB 存储引擎》第 9 章：性能优化

《深入浅出 MySQL》（第二版）22.6：调整 I/O 调度算法

}

