**PostgreSQL配置优化**

|  |  |  |  |
| --- | --- | --- | --- |
| **您的评价**: |  | 力荐 | [**收藏该经验**](javascript:void(0)) |

* PostgreSQL配置优化
  + - 硬件和系统配置
    - 测试工具
    - 配置文件
    - 主要选项
    - 测试数据
    - 总结

**硬件和系统配置**

|  |  |
| --- | --- |
| 操作系统 | Ubuntu13.04 |
| 系统位数 | 64 |
| CPU | Intel(R) Core(TM)2 Duo CPU |
| 内存 | 4G |
| 硬盘 | Seagate ST2000DM001-1CH164 |
| 测试工具 | PostgreSQL-9.1.11 |

**测试工具**

|  |  |
| --- | --- |
| 工具名称 | pgbench |
| 数据量 | 200W（整个数据库大小约为300M） |
| 模拟客户端数 | 4 |
| 线程数 | 4 |
| 测试时间 | 60秒 |

* 准备命令：pgbench -i -s 20 pgbenchdb
* 测试命令：pgbench -r -j4 -c4 -T60 testdb

**配置文件**

默认的配置配置文件是保存在/etc/postgresql/VERSION/main目录下的postgresql.conf文件

* 如果想查看参数修改是否生效，可以用psql连接到数据库后，用 来查看。
* 如果要修改shared\_buffers， 在ubuntu下可能需要执行命令Managing Kernel Resources

**主要选项**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 选项 | 默认值 | 说明 | 是否优化 | 原因 |
| max\_connections | 100 | 允许客户端连接的最大数目 | 否 | 因为在测试的过程中，100个连接已经足够 |
| fsync | on | 强制把数据同步更新到磁盘 | 是 | 因为系统的IO压力很大，为了更好的测试其他配置的影响，把改参数改为off |
| shared\_buffers | 24MB | 决定有多少内存可以被PostgreSQL用于缓存数据（推荐内存的1/4) | 是 | 在IO压力很大的情况下，提高该值可以减少IO |
| work\_mem | 1MB | 使内部排序和一些复杂的查询都在这个buffer中完成 | 是 | 有助提高排序等操作的速度，并且减低IO |
| effective\_cache\_size | 128MB | 优化器假设一个查询可以用的最大内存，和shared\_buffers无关（推荐内存的1/2) | 是 | 设置稍大，优化器更倾向使用索引扫描而不是顺序扫描 |
| maintenance\_work\_mem | 16MB | 这里定义的内存只是被VACUUM等耗费资源较多的命令调用时使用 | 是 | 把该值调大，能加快命令的执行 |
| wal\_buffer | 768kB | 日志缓存区的大小 | 是 | 可以降低IO，如果遇上比较多的并发短事务，应该和commit\_delay一起用 |
| checkpoint\_segments | 3 | 设置wal log的最大数量数（一个log的大小为16M） | 是 | 默认的48M的缓存是一个严重的瓶颈，基本上都要设置为10以上 |
| checkpoint\_completion\_target | 0.5 | 表示checkpoint的完成时间要在两个checkpoint间隔时间的N%内完成 | 是 | 能降低平均写入的开销 |
| commit\_delay | 0 | 事务提交后，日志写到wal log上到wal\_buffer写入到磁盘的时间间隔。需要配合commit\_sibling | 是 | 能够一次写入多个事务，减少IO，提高性能 |
| commit\_siblings | 5 | 设置触发commit\_delay的并发事务数，根据并发事务多少来配置 | 是 | 减少IO，提高性能 |

**测试数据**

* 测试的数据是运行3次，取平均值。
* 关闭fsync是为了更好的体现出其他参数对PostgreSQL的影响。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数 | 修改值 | 事务总数 | tps(包括建立连接) | tps(不包括建立连接) |
| 默认设置 |  | 8464 | 140.999792 | 141.016182 |
| fsync | off | 92571 | 1479.969755 | 1480.163355 |
| shared\_buffers | 1GB | 100055 | 1635.759275 | 1635.977823 |
| work\_mem | 10MB | 101209 | 1665.804812 | 1666.04082 |
| effective\_cache\_size | 2GB | 98209 | 1636.733152 | 1636.970271 |
| maintenance\_work\_mem | 512MB | 92930 | 1548.029233 | 1548.223108 |
| checkpoint\_segments | 32 | 195982 | 3265.995 | 3266.471064 |
| checkpoint\_completion\_target | 0.9 | 194390 | 3239.406493 | 3239.842596 |
| wal\_buffer | 8MB | 198639 | 3310.241458 | 3310.724067 |
| 恢复fsync | off | 11157 | 185.883542 | 185.909849 |
| commit\_delay && commit\_siblings | 10 && 4 | 11229 | 187.103538 | 187.131747 |

**总结**

|  |  |  |  |
| --- | --- | --- | --- |
|  | 事务总数 | tps(包括建立连接) | tps(不包括建立连接) |
| 优化前 | 8464 | 140.999792 | 141.016182 |
| 优化后(fsync=on) | 11229 | 187.103538 | 187.131747 |
| 优化后(fsync=off) | 198639 | 3310.241458 | 3310.724067 |

在fsync打开的情况下，优化后性能能够提升30%左右。因为有部分优化选项在默认的SQL测试语句中没有体现出它的优势，如果到实际测试中，提升应该不止30%。  
测试的过程中，主要的瓶颈就在系统的IO，如果需要减少IO的负荷，最直接的方法就是把fsync关闭，但是这样就会在掉电的情况下，可能会丢失部分数据。