**第二十一节知识点:**

**Mysql 服务器配置参数介绍**

**一部分参数要 根据 服务器硬件、应用类型、数据量的大小、和服务器的大小进行调整；**

**错误参数能够导致mysql系统崩溃所以要谨慎；**

**获取配置参数信息**

1. **命令行参数**
   1. **Mysqld\_safe –datadir=/data/sql\_data**
2. **读取配置文件**
   1. **Mysqld\_sqfe --help --verbose | grep -A 1 'Default options'**

**/etc/mysql/my.cnf /etc/my.cnf ~/.my.cnf**

**参数类型:**

1. **动态参数 :能够在mysql服务运行时动态调整，**
2. **静态参数 : 系统启动前进行配置,即运行时为只读;**

**如果修改会报错;**

**下面的配置都是配置动态参数的，**

**并且是在mysql命令行中而不是系统命令行中执行的**

1. **全局参数设置**
   1. **Set global 参数名=参数值**
   2. **Set @@global.参数名:=参数值**

**全局更改对全局有效，但是只对新的session（会话）有效对已有session（会话）是无效的，除非让其断开连接或者将session（会话）的值配置成更改过后全局的值；**

1. **会话参数设置**
   1. **Set [session] 参数名=参数值**
   2. **Set @@session.参数名:=参数值**

**会话参数在没有设置的情况下默认是用全局的参数值**

**注意:**

**上面的参数配置只是针对于动态的参数进行配置；对动态参数的修改都是针对此次服务的生命周期，而不会对参数配置文件进行修改，当重启服务时(如我们lnmp restart 一下) 所有的修改都会消失不管你是全局修改还是会话修改，所以要想做到完全修改必须修改配置文件**

**第二十二节知识点：**

**内存配置:**

**Mysql使用内存的情况可以分为两类**

1. **无法通过参数来配置**
   1. **如 mysql服务器的运行，解析，查询等内部使用**
2. **可以通过参数来配置**
   1. **如 缓存池，**

**配置时要注意的问题**

**1. 确定可以使用的内存的上限**

**可以使用的内存不能超过物理内存**

**操作系统对mysql使用内存也是有限制的，用64位的**

**2. 确定mysql每个链接使用的内存**

**就是合理配置独享线程的内存**

**mysql涉及下面操作的时候才会对以下参数分配内存。**

**并且一旦分配内存 就会分配参数值给定的大小;**

**内存由两个方面组成；**

1. **线程独享内存**
   1. **Sort\_buffer\_size 排序缓存区的大小（order by,group by）**
   2. **Join\_buffer\_size 连接缓冲区的大小(关联查询)**
      1. **当join三个表的时候 这个缓存区就会有三个**
   3. **Read\_****buffer\_size顺序读取数据缓冲区使用内存**
   4. **Read\_rnd\_buffer\_size随机读取数据缓冲区使用内存**

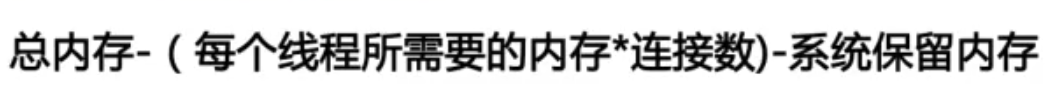
**以上四个参数 都是为某个线程分配的 。如果有100个链接就会分配100倍的内存，所以配大了就会导致mysql内存溢出 崩溃**

1. **线程共享内存**
   1. **Innodb\_buffer\_pool\_size （对innodb的索引和数据缓存）**

**缓存区越大，访问表中的数据需要的磁盘I/o越少**

**但是过大的话会导致物理内存竞争过大**

**当都为innodb表的话 内存可以设置为**



* 1. **Key\_buffer\_size（对myisam索引优化）**

**就算都用的innodb表也要给这个分配一个大小因为innodb的系统表还在使用Myisam的存储引擎**

**key\_buffer\_size指定索引缓冲区的大小，它决定索引处理的速度，尤其是索引读的速度，通过key\_read\_requests和key\_reads可以直到key\_baffer\_size设置是否合理。**

**可以通过show status like "key\_read%"查看**

|  |  |
| --- | --- |
|  | mysql> show variables like "key\_buffer\_size%";  +-----------------+---------+  | Variable\_name   | Value   |  +-----------------+---------+  | key\_buffer\_size | 8388608 |  +-----------------+---------+  1 row in set (0.00 sec) |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | mysql> show status like "key\_read%";  +-------------------+-------+  | Variable\_name     | Value |  +-------------------+-------+  | Key\_read\_requests | 6     |  | Key\_reads         | 3     |  +-------------------+-------+  2 rows in set (0.01 sec) |

一共有6个索引读取请求，有三个请求在内存中没有找到直接从硬盘中读取索引

**第二十三节知识点:**

**I/O相关的配置参数**

**一、 与innodb相关的配置参数  
 1. innodb\_log\_file\_size**

**指定重做日志文件大小**

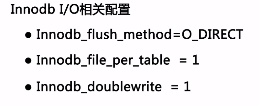
**2 . innodb\_log\_files\_in\_group**

**指定日志文件组中重做日志文件的个数，默认2**

1. **innodb\_mirrored\_log\_groups：**

**指定日志镜像文件组的数量，默认1**

1. **innodb\_log\_buffer\_size：对于控制支持缓冲区的大小  
   因为每秒就将缓存中的日志读到磁盘所以不要设置太大**
2. **innodb\_flush\_log\_at\_trx\_commit**



**事务日志的总大小就是 日志文件大小\*文件数，而根据经验，设置日志总大小要可以保持一个小时左右的服务器活动**

**事务日志是循环使用的，写满一个文件之后才会使用下一个，当所有的写满后又从第一个开始写**

**事务工作的流程:**

1. **重做日志先写入重做日志缓冲池**
2. **每秒会将重做日志缓冲写入磁盘的重做日志文件中**
3. **首先写入重做日志组中的日志文件1即ib\_logfile0，该文件写满后，继续向ib\_logfile2中写，日志文件2也写满后，继续向ib\_logfile0中写，循环执行**

**Innodb事务的介绍**

**InnoDB是一种是事务型的存储引擎，为了减少提交事务时产生的I/O开销，InnoDB写日志的方式就是在事务提交的时候，先写入事务日志中，而不是每次把修改后的数据刷新到数据文件中，这样做的是为了提高I/O的性能，因为事务的修改，使数据和索引文件通常都会映射到表空间的随机的位置，所以能刷新数据变更到数据文件，就会产生大量的随机I/O，而记录日志所需要的是顺序的I/O，所以相比于刷新脏数据到数据文件系统来说，记录事务日志的方法要快得多。**

**注: 随机I/O比顺序I/O性能要差，而日志是支持顺序I/O的**

**第二十四节知识点:**

**安全相关的配置:**

1. **expire\_logs\_days：指定自动清理binlog的天数**

**作用:**

* 1. 如果我们在启用binlog的日志，就应该打开这个选项，可以让服务器自动清理旧的二进制日志，以防止二进制日志占用太多空间，最终把服务器的空间耗尽的这种情况；

1. **max\_allowed\_packet 控制mysql可以接收的包的大小**

**主从配置的时候要确保这个参数的同步。不然会导致同步失败**

**这个参数的作用就是:**

**如果设置的太小，而客户端传送过来的添加或者修改的数据太大超过了这个参数值，就会导致写失败**

1. **skip\_name\_resolve 禁用DNS查找**

**解读:**

[**https://blog.csdn.net/qq\_25497867/article/details/81064682**](https://blog.csdn.net/qq_25497867/article/details/81064682)**’**

1. **sysdate\_is\_now 确保sysdate()返回确定性日期**
2. **read\_only 禁止非super权限的用户写权限**
   1. **这个参数开启,禁止没有super权限的用户在从库中进行数据变更，只接受在主库传来的数据变更，能够保持数据的一致性，所以强力建议开启，并且谨记在用户授权的时候不要轻易的给用户super权限;**
3. **Skip\_slave\_start 禁用slave自动恢复**
4. **Sql\_mode** 
   1. **默认下mysql对sql语法检查是非常轻松的；有些不符合语法的sql在mysql这里是不报错的。下面说一下sql\_mode的几个参数值，用来严格要求sql**
      1. **Strict\_trans\_tables**
      2. **No\_engine\_subtitution**
      3. **No\_zero\_date**
      4. **No\_zero\_in\_date**
      5. **Only\_full\_group\_by**

**上面的知识sql\_mode中的几个值，在这里强力提醒一下，不要轻易的修改这个参数，不然现有的程序有可能出现很多sql方面的错误；**

**第二十五节知识点:**

**其他常用的参数配置:**

1. **Sync\_binlog 控制mysql 怎么刷新二进制日志从cache到磁盘**
   1. **默认 为0;不会主动刷新，由操作系统决定**
   2. **大于0 则两次刷新日志到磁盘的次数**
   3. **如果为1 则每次事务完成之后就会往磁盘中写一下，虽然成本比较高，但是安全，在主从中，主服务器最好设置为1，这样可以避免主DB崩溃cache日志没有同步到二进制日志中，从而避免主从不同步；**
2. **Tmp\_table\_size 和 max\_heap\_table\_size;控制内存临时表的大小**
   1. **http://blog.sina.com.cn/s/blog\_59dd20310102yinv.html**
3. **Max\_connections 控制允许的最大链接数**