MySQL运维及开发规范

一.基础规范

(1) 使用INNODB存储引擎
(2) 表字符集使用UTF8
(3) 所有表都需要添加注释
(4) 单表数据量建议控制在5000W以内
(5) 不在数据库中存储图、文件等大数据
(6) 禁止在线上做数据库压力测试
(7) 禁从测试、开发环境直连数据库

二.命名规范

(1) 库名表名字段名必须有固定的命名长度，12个字符以内
(2) 库名、表名、字段名禁止超过32个字符。须见名之意
(3) 库名、表名、字段名禁止使用MySQL保留字
(4) 临时库、表名必须以tmp为前缀，并以日期为后缀
(5) 备份库、表必须以bak为前缀，并以日期为后缀

三.库、表、字段开发设计规范

(1) 禁使用分区表
(2) 拆分大字段和访问频率低的字段，分离冷热数据
(3) 用HASH进散表，表名后缀使进制数，下标从0开始
(4) 按日期时间分表需符合YYYY[MM][DD][HH]格式
(5) 采用合适的分库分表策略。例如千库十表、十库百表等
(6) 尽可能不使用TEXT、BLOB类型
(7) 用DECIMAL代替FLOAT和DOUBLE存储精确浮点数
(8) 越简单越好：将字符转化为数字、使用TINYINT来代替ENUM类型
(9) 所有字段均定义为NOT NULL
(10) 使用UNSIGNED存储非负整数
(11) INT类型固定占用4字节存储
(12) 使用timestamp存储时间
(13) 使用INT UNSIGNED存储IPV4
(14) 使用VARBINARY存储大小写敏感的变长字符串
(15) 禁止在数据库中存储明文密码，把密码加密后存储
(16) 用好数值类型字段
    Tinyint   (1Byte)
    smallint   (2Byte)
    mediumint   (3Byte)
    int   (4Byte)
    bigint   (8Byte)
    如果数值字段没有那么大，就不要用 bigint
(17) 存储ip最好用int存储而非char(15)
(18) 不允许使用ENUM
(19) 避免使用NULL字段
    NULL字段很难查询优化，NULL字段的索引需要额外空间，NULL字段的复合索引无效
(20) 少用text/blob，varchar的性能会比text高很多，实在避免不了blob，请拆表
(21) 数据库中不允许存储大文件，或者照片，可以将大对象放到磁盘上，数据库中存储它的路径

四.索引规范

1、索引的数量要控制：

```
(1) 单张表中索引数量不超过5个
(2) 单个索引中的字段数不超过5个
(3) 对字符串使用前缀索引，前缀索引长度不超过8个字符
(4) 建议优先考虑前缀索引，必要时可添加伪列并建立索引
```

2、主键准则

```
(1) 表必须有主键
(2) 不使用更新频繁的列作为主键
(3) 尽量不选择字符串列作为主键
(4) 不使用UUID MD5 HASH这些作为主键(数值太离散了)
(5) 默认使非空的唯一键作为主键
(6) 建议选择自增或发号器
```

3、重要的SQL必须被索引，比如：

```
(1) UPDATE、DELETE语句的WHERE条件列
(2) ORDER BY、GROUP BY、DISTINCT的字段
```

4、多表JOIN的字段注意以下：
(1) 区分度最大的字段放在前面
(2) 核SQL优先考虑覆盖索引
(3) 避免冗余和重复索引
(4) 索引要综合评估数据密度和分布以及考虑查询和更新比例

5、索引禁忌

```
(1) 不在低基数列上建立索引，例如“性别”
(2) 不在索引列进行数学运算和函数运算
```

6、尽量不使用外键

```
(1) 外键用来保护参照完整性，可在业务端实现
(2) 对父表和子表的操作会相互影响，降低可用性
```

7、索引命名：非唯一索引必须以 idx_字段1_字段2命名，唯一所以必须以uniq_字段1_字段2命名，索引名称必须全部小写

8、新建的唯一索引必须不能和主键重复

9、索引字段的默认值不能为NULL，要改为其他的default或者空。NULL非常影响索引的查询效率

10、反复查看与表相关的SQL，符合最左前缀的特点建立索引。多条字段重复的语句，要修改语句条件字段的顺序，为其建立一条联合索引，减少索引数量

11、能使用唯一索引就要使用唯一索引，提高查询效率

12、研发要经常使用explain，如果发现索引选择性差，必须让他们学会使用hint

五.SQL规范

```
(1) sql语句尽可能简单
    大的sql想办法拆成小的sql语句(充分利用QUERY CACHE和充分利用多核CPU)
(2) 事务要简单，整个事务的时间长度不要太长
(3) 避免使用触发器、函数、存储过程
(4) 降低业务耦合度，为sacle out、sharding留有余地
(5) 避免在数据库中进数学运算(MySQL不擅长数学运算和逻辑判断)
(6) 不要用select *，查询哪几个字段就select 这几个字段
(7) sql中使用到OR的改写为用 IN()   (or的效率没有in的效率高)
(8) in里面数字的个数建议控制在1000以内
(9) limit分页注意效率。Limit越大，效率越低。可以改写limit，比如例子改写：
     select id from tlimit 10000, 10;  =>  select id from t where id > 10000 limit10;
(10)  使用union all替代union
(11) 避免使大表的JOIN
(12) 使用group by 分组、自动排序
(13) 对数据的更新要打散后批量更新，不要一次更新太多数据
(14) 减少与数据库的交互次数
(15) 注意使用性能分析工具
     Sql explain  /  showprofile  /    mysqlsla
(16) SQL语句要求所有研发，SQL关键字全部是大写，每个词只允许有一个空格
(17) SQL语句不可以出现隐式转换，比如 select id from 表 where id='1'
(18) IN条件里面的数据数量要少，我记得应该是500个以内，要学会使用exist代替in，exist在一些场景查询会比in快
(19) 能不用NOT IN就不用NOTIN，坑太多了。。会把空和NULL给查出来
(20) 在SQL语句中，禁止使用前缀是%的like
(21) 不使用负向查询，如not in/like
(22) 关于分页查询：程序里建议合理使用分页来提高效率limit，offset较大要配合子查询使用
(23) 禁止在数据库中跑大查询
(24) 使预编译语句，只传参数，比传递SQL语句更高效；一次解析，多次使用；降低SQL注入概率
(25) 禁止使order by rand()
(26) 禁单条SQL语句同时更新多个表
```

六.流程规范

```
(1) 所有的建表操作需要提前告知该表涉及的查询sql；
(2) 所有的建表需要确定建立哪些索引后才可以建表上线；
(3) 所有的改表结构、加索引操作都需要将涉及到所改表的查询sql发出来告知DBA等相关人员；
(4) 在建新表加字段之前，要求研发至少要提前3天邮件出来，给dba们评估、优化和审核的时间
(5) 批量导入、导出数据必须提前通知DBA协助观察
(6) 禁在线上从库执行后台管理和统计类查询
(7) 禁有super权限的应用程序账号存在
(8) 推广活动或上线新功能必须提前通知DBA进行流量评估
(9) 不在业务高峰期批量更新、查询数据库
```

六.配置优化规范

(1) 开启慢查询，用于sql语句分析。

(2) 开启二进制日志，用于遇到mysql崩溃，数据恢复

(3) no-auto-rehash 确保这个服务启动得比较快。

(4) back_log = 600

```
在MYSQL暂时停止响应新请求之前，短时间内的多少个请求可以被存在堆栈中。如果系统在短时间内有很多连接，则需要增大该参数的值，该参数值指定到来的
```

(5) max_connections = 3000
#MySQL允许最大的进程连接数，如果经常出现Too Many Connections的错误提示，则需要增大此值。默认151

(6) max_connect_errors = 6000

#设置每个主机的连接请求异常中断的最大次数，当超过该次数，MYSQL服务器将禁止host的连接请求，直到mysql服务器重启或通过flush hosts命令清空此host的相关信息。默认100

(7) external-locking = FALSE
#使用–skip-external-locking MySQL选项以避免外部锁定。该选项默认开启

(8) max_allow ed_packet = 32M
#设置在网络传输中一次消息传输量的最大值。系统默认值 为4MB，最大值是1GB，必须设置1024的倍数。

(9) sort_buffer_size = 2M
#Sort_Buffer_Size 是一个connection级参数，在每个connection（session）第一次需要使用这个buffer的时候，一次性分配设置的内存。
#Sort_Buffer_Size 并不是越大越好，由于是connection级的参数，过大的设置+高并发可能会耗尽系统内存资源。例如：500个连接将会消耗 500*sort_buffer_size(8M)=4G内存
#Sort_Buffer_Size 超过2KB的时候，就会使用mmap() 而不是 malloc() 来进行内存分配，导致效率降低。系统默认2M，使用默认值即可

(10)join_buffer_size = 2M
#用于表间关联缓存的大小，和sort_buffer_size一样，该参数对应的分配内存也是每个连接独享。系统默认2M，使用默认值即可

(11)thread_cache_size = 300
#默认38 服务器线程缓存这个值表示可以重新利用保存在缓存中线程的数量,当断开连接时如果缓存中还有空间,那么客户端的线程将被放到缓存中,如果线程重新被请求，那么请求将从缓存中读取,如果缓存中是空的或者是新的请求，那么这个线程将被重新创建，如果有很多新的线程，增加这个值可以改善系统性能.通过比较 Connections 和 Threads_created 状态的变量，可以看到这个变量的作用。设置规则如下：1GB 内存配置为8，2GB配置为16，3GB配置为32，4GB或更高内存，可配置更大。

(12)thread_concurrency = 8
#系统默认为10，使用10先观察 ,设置thread_concurrency的值的正确与否，对mysql的性能影响很大，在多个cpu(或多核)的情况下，错误设置了thread_concurrency的值，会导致mysql不能充分利用多cpu(或多核)，出现同一时刻只能一个cpu(或核)在工作的情况。thread_concurrency应设为CPU核数的2倍. 比如有一个双核的CPU，那么thread_concurrency的应该为4; 2个双核的cpu，thread_concurrency的值应为8

(13)query_cache_size = 64M
#在MyISAM引擎优化中，这个参数也是一个重要的优化参数。但也爆露出来一些问题。机器的内存越来越大，习惯性把参数分配的值越来越大。这个参数加大后也引发了一系列问题。我们首先分析一下 query_cache_size的工作原理：一个SELECT查询在DB中工作后，DB会把该语句缓存下来，当同样的一个SQL再次来到DB里调用时，DB在该表没发生变化的情况下把结果从缓存中返回给Client。这里有一个关建点，就是DB在利用Query_cache工作时，要求该语句涉及的表在这段时间内没有发生变更。那如果该表在发生变更时，Query_cache里的数据又怎么处理呢？首先要把Query_cache和该表相关的语句全部置为失效，然后在写入更新。那么如果Query_cache非常大，该表的查询结构又比较多，查询语句失效也慢，一个更新或是Insert就会很慢，这样看到的就是Update或是Insert怎么这么慢了。所以在数据库写入量或是更新量也比较大的系统，该参数不适合分配过大。而且在高并发，写入量大的系统，建议把该功能禁掉。

(14)query_cache_limit = 4M
#指定单个查询能够使用的缓冲区大小，缺省为1M

(15)query_cache_min_res_unit = 2k
#默认是4KB，设置值大对大数据查询有好处，但如果你的查询都是小数据查询，就容易造成内存碎片和浪费
#查询缓存碎片率 = Qcache_free_blocks / Qcache_total_blocks * 100%
#如果查询缓存碎片率超过20%，可以用FLUSH QUERY CACHE整理缓存碎片，或者试试减小query_cache_min_res_unit，如果你的查询都是小数据量的话。
#查询缓存利用率 = (query_cache_size – Qcache_free_memory) / query_cache_size * 100%
#查询缓存利用率在25%以下的话说明query_cache_size设置的过大，可适当减小;查询缓存利用率在80%以上而且Qcache_low mem_prunes > 50的话说明query_cache_size可能有点小，要不就是碎片太多。
#查询缓存命中率 = (Qcache_hits – Qcache_inserts) / Qcache_hits * 100%

(16)thread_stack = 192K
#设置MYSQL每个线程的堆栈大小，默认值足够大，可满足普通操作。可设置范围为128K至4GB，默认为256KB，使用默认观察

(17)transaction_isolation = READ-COMMITTED
设定默认的事务隔离级别.可用的级别如下:READ UNCOMMITTED-读未提交 READ COMMITTE-读已提交 REPEATABLE READ -可重复读 SERIALIZABLE -串行

(18)tmp_table_size = 256M
tmp_table_size 的默认大小是 32M。如果一张临时表超出该大小，MySQL产生一个 The table tbl_name is full 形式的错误，如果你做很多高级 GROUP BY 查询，增加 tmp_table_size 值。如果超过该值，则会将临时表写入磁盘。

(18)max_heap_table_size = 256M

(19)expire_logs_days = 7

(20)key_buffer_size = 2048M
#批定用于索引的缓冲区大小，增加它可以得到更好的索引处理性能，对于内存在4GB左右的服务器来说，该参数可设置为256MB或384MB。

(21)read_buffer_size = 1M
#默认128K
MySql读入缓冲区大小。对表进行顺序扫描的请求将分配一个读入缓冲区，MySql会为它分配一段内存缓冲区。read_buffer_size变量控制这一缓冲区的大小。如果对表的顺序扫描请求非常频繁，并且你认为频繁扫描进行得太慢，可以通过增加该变量值以及内存缓冲区大小提高其性能。和sort_buffer_size一样，该参数对应的分配内存也是每个连接独享。

(22)read_rnd_buffer_size = 16M
MySql的随机读（查询操作）缓冲区大小。当按任意顺序读取行时(例如，按照排序顺序)，将分配一个随机读缓存区。进行排序查询时，MySql会首先扫描一遍该缓冲，以避免磁盘搜索，提高查询速度，如果需要排序大量数据，可适当调高该值。但MySql会为每个客户连接发放该缓冲空间，所以应尽量适当设置该值，以避免内存开销过大。

(23)bulk_insert_buffer_size = 64M
#批量插入数据缓存大小，可以有效提高插入效率，默认为8M

(24)myisam_sort_buffer_size = 128M
MyISAM表发生变化时重新排序所需的缓冲 默认8M

(25)myisam_max_sort_file_size = 10G
MySQL重建索引时所允许的最大临时文件的大小 (当 REPAIR, ALTER TABLE 或者 LOAD DATA INFILE).
如果文件大小比此值更大,索引会通过键值缓冲创建(更慢)

(26)myisam_max_extra_sort_file_size = 10G 5.6无此值设置

(27)myisam_repair_threads = 1 默认为1
如果一个表拥有超过一个索引, MyISAM 可以通过并行排序使用超过一个线程去修复他们.
这对于拥有多个CPU以及大量内存情况的用户,是一个很好的选择.

(28)innodb_additional_mem_pool_size = 16M
#这个参数用来设置 InnoDB 存储的数据目录信息和其它内部数据结构的内存池大小，类似于Oracle的library cache。这不是一个强制参数，可以被突破。

(29)innodb_buffer_pool_size = 2048M
这对Innodb表来说非常重要。Innodb相比MyISAM表对缓冲更为敏感。MyISAM可以在默认的 key_buffer_size 设置下运行的可以，然而Innodb在默认的 innodb_buffer_pool_size 设置下却跟蜗牛似的。由于Innodb把数据和索引都缓存起来，无需留给操作系统太多的内存，因此如果只需要用Innodb的话则可以设置它高达 70-80% 的可用内存。一些应用于 key_buffer 的规则有 — 如果你的数据量不大，并且不会暴增，那么无需把 innodb_buffer_pool_size 设置的太大了

(30)#innodb_data_file_path = ibdata1:1024M:autoextend 设置过大导致报错，默认12M观察

```
#表空间文件 重要数据
```

(31)#innodb_file_io_threads = 4 不明确，使用默认值

```
#文件IO的线程数，一般为 4，但是在 Windows 下，可以设置得较大。
```

(32)innodb_thread_concurrency = 8

```
#服务器有几个CPU就设置为几，建议用默认设置，一般为8.
```

(33)innodb_flush_log_at_trx_commit = 2

```
# 如果将此参数设置为1，将在每次提交事务后将日志写入磁盘。为提供性能，可以设置为0或2，但要承担在发生故障时丢失数据的风险。设置为0表示事务日
```

(34)#innodb_log_buffer_size = 16M 使用默认8M

```
#此参数确定些日志文件所用的内存大小，以M为单位。缓冲区更大能提高性能，但意外的故障将会丢失数据.MySQL开发人员建议设置为1－8M之间
```

(35)#innodb_log_file_size = 128M 使用默认48M

```
#此参数确定数据日志文件的大小，以M为单位，更大的设置可以提高性能，但也会增加恢复故障数据库所需的时间
```

(36)#innodb_log_files_in_group = 3 使用默认2

```
#为提高性能，MySQL可以以循环方式将日志文件写到多个文件。推荐设置为3M
```

(37)#innodb_max_dirty_pages_pct = 90 使用默认75观察

```
# Buffer_Pool中Dirty_Page所占的数量，直接影响InnoDB的关闭时间。参数innodb_max_dirty_pages_pct 可以直接控制了Dirty_Page在Buffer_P
```

(38)innodb_lock_w ait_timeout = 120

```
#默认为50秒 InnoDB 有其内置的死锁检测机制，能导致未完成的事务回滚。但是，如果结合InnoDB使用MyISAM的lock tables 语句或第三方事务引擎，
```

(39)innodb_file_per_table = 0
#默认为No #独享表空间（关闭）