## 数据库基础设计规范

1 所有数据库对象名称必须使用小写字母并用下划线分割（mysql对大小写敏感的）

2 所有数据库对象名称禁止使用mysql保留的关键字

3 数据库对象的命名要能做到见名识义，并且最好不要超过32个字符（减少不必要的网络开销）

4 临时库表必须以tmp为前缀并以日期为后缀

5 备份表，备份表必须以bak为前缀并以日期为后缀

6 所有存储相同数据的列名和列类型必须一致（如果作为关联列的数据类型不同的话在查询的时候就会进行数据类型转换，失去关联的作用）

7 5.6之后的所有表都必须用Innodb存储引擎，Innodb支持事务，行级锁，更好的恢复性，高并发下性能更好

8 数据库和表的字符集统一使用UTF-8（兼容性更好），UTF8字符集汉字占3个字节，ASCLL码占用1个字节

9 尽量控制单表数据量的大小，建议控制在500w以内，500w并不是mysql数据库的限制，只是会影响查询，备份，恢复

10 谨慎使用mysql分区表

分区表在物理上表现为多个文件，在逻辑上表现为一个表

谨慎选择分区建，跨分区查询效率可能更低

建议采用物理分表的方式管理大数据

11 尽量做到冷热数据分离，减小表的宽度

mysql限制是4096列主要是减少磁盘的IO，保证热数据的内存缓存命中率

更有效的利用缓存，避免读入无用的冷数据

经常一起使用的列放到一个表中

12 禁止在表中建立预留字段

预留字段的命名很难做到见名识义

预留字段无法确认存储的数据类型，所以无法选择合适的类型

13 禁止在数据库中存储图片，文件等二进制数据

禁止在线上做数据库压力测试

禁止从开发环境，测试环境直接连接生产环境数据库

## 数据库索引设计规范

1 限制每张表上的索引数量，建议单张表索引不超过5个

索引并不是越多越好！索引可以提高效率同样可以降低效率

禁止给表的每一列都建立单独的索引

2 每个innodb表必须有一个主键

不使用更新频繁的列左右主键，不使用多列主键

不使用UUID，MD5,HASH，字符串列作为主键

主键建议选择使用自增ID值

3常见的索引列建议

select update delete 语句的where从句中的列

包含在order by, group by,distinct中的字段

多表join的关联列

4 如何选择索引列的顺序（联合索引 索引列是从左到右的顺序来使用的）

区分度最高的列放在联合索引的最左侧

尽量把字段长度小的列放在联合索引的最左侧

使用最频繁的列放在索引的最左侧

5 避免建立冗余索引和重复索引

重复索引 primary key(id),index(id),unique index(id)

冗余索引 index(a,b,c) index(a,b) index(a)

6 对于频繁的查询优先考虑使用覆盖索引

覆盖索引：就是包含了所有查询字段的索引（指的是查询所用到的字段 where后的字段 ，orderby的字段，groupby的字段，select后面的字段）

避免innodb表进行索引的二次查找

可以把随机IO变为顺序IO加快查询效率

7 尽量避免使用外键约束

不建议使用外键约束，但一定在表与表之间的关联键上建立索引

外键可用于保证数据的参照完整性 但建议在业务端实现

外键会影响父表与字表的写操作从而降低性能

## 数据库字段设计规范

1 优先选择符合存储需要的最小的数据类型

将字符串转化为数字类型存储（减少字节数）

对于非负型的数据来说，要优先使用无符号整形来存储（无符号相对于有符号可以多出 一倍的存储空间）

varchar(n)中的n代表的是字符数，而不是字节数

过大的长度会消耗更多的内存

2 避免使用text，blob的数据类型

建议把blob或是text列分离到单独的扩展表中

text和blob类型只能使用前缀索引

3 避免使用 enum 数据类型

修改enum值需要使用alter语句（存在操作风险）

enum类型的orderby 操作效率低，需要额外的操作

禁止使用数值作为ENUM的枚举值

4 尽可能把所有列定义为NOT null

索引null列需要额外的空间来保存，所以要占用更多的空间

进行比较和计算时要对null值做特别的处理

5 字符串存储日期型的数据（不正确的做法）

缺点: 无法用日期函数进行计算和比较

用字符串存储日期要占用更多的空间

6 使用TIMESTAMP或DATETIME类型存储时间

TIMESTAMP 1970-01-01 00:00:01 ~2038-01-19 03:14:07

TIMESTAMP占用4字节和int相同，但比int可读性更高

超出的需要用DATETIME存储