**Mysql开发规范**

|  |  |  |
| --- | --- | --- |
| 文件状态  [ ] 草稿  [ ] 正式发布  [ ] 修改稿 | 文件编号 |  |
| 保密等级 | ■内部公开 □秘密 □机密 □绝密 |
| 当前版本 | V1.0 |
| 编制人 | 刘波 |
| 审核人 |  |
| 编制日期 | 2015-8-18 |

**文档修订记录**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 日期 | 修订内容 | 修改类型 | 修改人 | 审核人 | 版本号 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

修改类型分为 **A** - ADDED **M** - MODIFIED **D** – DELETE

## 禁止项：

## 禁止使用select \* 等查询

当查询所有字段(select \*)会导致下列问题

1. 增加网络带宽消耗
2. Select \*必然会导致回表查询/返回数据，使覆盖所以失效
3. 如果表结构有修改比如增加多列，返回多余数据是比较危险的

## 禁止库名、表名、字段名使用MySQL保留字

当库名、表名、字段名等属性含有保留字时,SQL语句必须用反引号引用属性名称,这将使得SQL语句书写、SHELL脚本中变量的转义等变得非常复杂。

## 禁止使用分区表

Mysql分区表技术还是不是很成熟，而且对分区键有严格要求，分区表变大后对于表备份恢复都有很大困难，建议在业务端使用sharding技术。

## 禁止在数据库中存储明文密码

如果需要存储MySQL密码可以用MySQL内置函数password()对明文密码进行MD5进行加密。

## SQL中禁止出现now()、rand()、sysdate()、current\_user()等不确定结果的函数。

建议不确定的时间在程序层取出时间，语句级复制场景下,引起主从数据不一致; 不确定值的函数,产生的SQL语句无法利用。

## 禁止使用VARBINARY、BLOB存储图片、文件等，使用VARCHAR(N)，N尽量可能小

## 禁止在列上进行运算

在列上运算将导致Mysql索引失效而进行全表扫描。

## 规范项：

## 1、建表字符集使用UTF8

UTF8统一而且通用,不会出现转码出现乱码风险。

## 2、表必须有主键,推荐使用UNSIGNED自增列作为主键

表没有主键,INNODB会默认设置隐藏的主键列;没有主键的表在定位数据行时效率会非常低而且降低基于行复制的效率。在建表时务必定义一个自增列做主键（与业务逻辑无关，而应用程序的数据如果有唯一的候选列可以做成唯一键），再次重申INNODB存储引擎中每张表一定要有一个于业务无关的自增列做主键。

## 建议项：

## 1、建议慎重使用前缀匹配的模糊查询

前缀匹配会导致直接全表扫描或全索引扫描，性能最差，无任何扩展，基本不可接受。

## 2、建议所有字段均定义为NOT NULL，设置default值。

定义为Not Null原因如下：

1. MySQL数据库中每个为NULL的列都需要额外的1个字节进行存储，浪费空间资源。

2. B树索引时不会存储NULL值,如果索引字段可以为NULL,索引效率会下降。

建议用0、特殊值或空串代替NULL值。

## 3、建议查询中避免隐式转换

MySQL中如果查询字段与表定义字段不同则会发生隐式转换，从而无法用到索引导致查询效率低下。

## 4、建议不要在MySQL数据库中存放业务逻辑。

数据库是有状态的服务,变更复杂而且速度慢,如果把业务逻辑放到数据库中,将会限制业务的快速发展。建议把业务逻辑提前,放到前端或中间逻辑层,而把数据库作为存储层,实现逻辑与存储的分离。

## 5、建议不要使用子查询

对于子查询，mysql会对子查询结果返回给外部表，并对外部表进行全表扫描

## 6、建议将大字段、访问频率低的字段拆分到单独的表中存储,分离冷热数据

当我们的表中存在类似于 TEXT 或者是大的 VARCHAR类型的大字段的时候，如果我们大部分访问这张表的时候都不需要这个字段，我们就该将其拆分到另外的独立表中，以减少常用数据所占用的存储空间。这样做的一个明显好处就是每个数据块中可以存储的数据条数可以大大增加，既减少物理 IO 次数，也能大大提高内存中的缓存命中率。

## 7、建议用in() /union替换or，并注意in的个数（个数多少依照具体情况而定）

## 8、建议尽量不使用mysql存储过程、触发器、函数等（依照具体情况而定）

容易将业务逻辑和DB耦合在一起，并且对于目前数据量存储过程、触发器、函数等没有任何优势（存储过程、函数对大数据量的处理和复杂业务逻辑很有优势），而且mysql存储过程还有一定BUG。