MYSQL 设计规范

1. 基础规范

1.1 使用 innodb 存储引擎

1.2 表字符集使用 UTF8

1.3 所有表都需要添加注释

1.4 单表数据量建议控制在 5000W 以内

1.5 不在数据库中存储图片、文件等大数据

1.6 禁止从测试和开发环境直接连接数据库

2. 命名规范

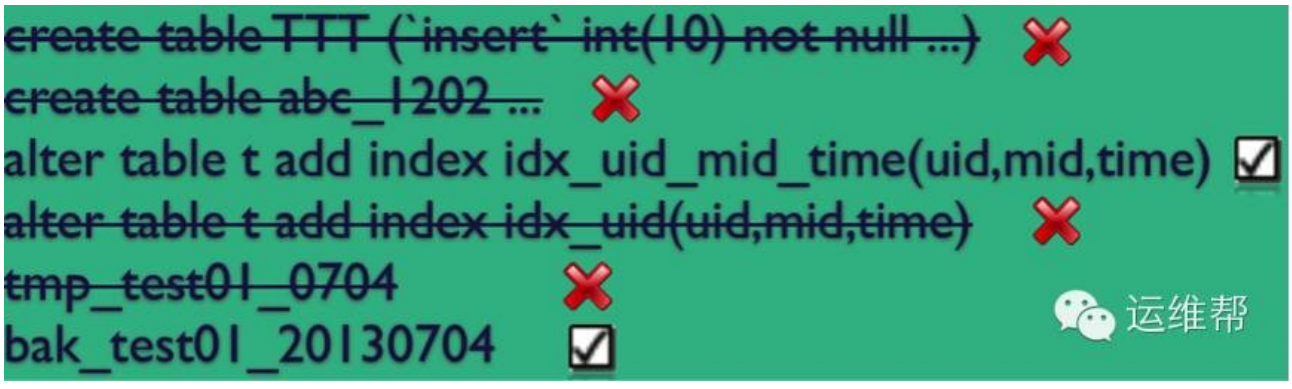
2.1 库名表名字段必须有固定的命名长度，12 个字符以内

2.2 库名、表名、字段名全部小写，多个字段用下划线连接，禁止使用大写

2.3 库名、表名、字段名禁止使用 MYSQL 保留字

2.4 临时库、临时表必须以 tmp 为前缀，日期做后缀

2.5 备份库、备份表必须以 bak 为前缀，并以日期为后缀



3. 库、表、字段开发设计规范

3.1 禁止使用分区表

3.2 拆分大字段和访问频次低的字段，分离冷热数据

3.3 用 HASH 进行散表，表名后缀使用十进制，下标从 0 开始

3.4 按日期时间分表需要符合 YYYY[MM][DD][HH]格式

3.5 选择合适的分库和分表策略。例如千库十表、十库百表

comment_20120815
comment_20120816
~~comment_120817~~
~~user_39~~
~~user_3A~~
~~user_3B~~
~~user_3C~~

```
CREATE TABLE `blog` (
  `blog_id` int(11) NOT NULL AUTO_INCREMENT,
  `author_id` int(11) NOT NULL,
  `create_date` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `rank` int(11) DEFAULT '0',
  `category` varchar(20) DEFAULT NULL,
  `abstract` varchar(100) DEFAULT NULL,
  `tags` varchar(100) NOT NULL DEFAULT '',
  `body` text,
  PRIMARY KEY (`blog_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```


3.6 尽量不使用 TEXT、BLOB 类型

3.7 用 DECIMAL 代替 FLOAT 和 DOUBLE 存储精确浮点是

3.8 越简单越好：将字符转化为数字、使用 TINYINT 来代替 ENUM 类似

3.9 所有字段均定义为 NOT NULL

3.10 使用 UNSIGNED 存储非负整数

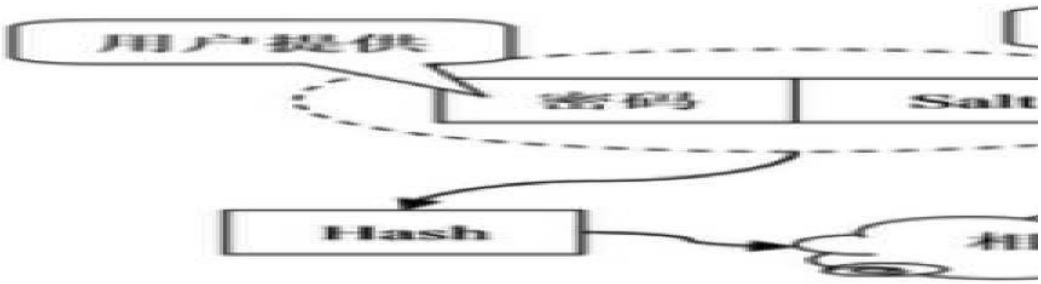
3.11 INT 类型固定占用 4 字节存储

3.12 使用 timestamp 存储时间（单位毫秒数）

3.13 使用 INT UNDIGNED 存储 IPV4

3.14 使用 VARBINARY 存储大小敏感的长字符串

3.15 禁止在数据库中存储明文密码，把密码加密后存储



3.16 用好数值类型字段

tinyint 1 字节

smallint 2 个字节

mediumint 3 个字节

int 4 个字节

bigint 8 个字节

3.17 存储 ip 最好用 int 存储而非 char(15)

3.18 不允许使用 enum

3.19 避免使用 NULL 字段

3.20 少用 text/blob, varchar 的性能会比 text 高很多，实在避免不了 blob,请拆表

索引规范

4.1 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，必要是可添加伪列并建立索引

4.2 主键准则

a、表必须有主键

b、不使用更新频繁的列作为主键

c、尽量不选择字符串列作为主键

d、不使用 UUID MD5 HASH 这些作为主键(数值太离散了)

e、默认使□非空的唯一键作为主键

f、建议选择自增或发号器

4.3 重要的 SQL 必须被索引

比如：

a、UPDATE、DELETE 语句的 WHERE 条件列

b、ORDER BY、GROUP BY、DISTINCT 的字段

4.4 多表 JOIN 的字段注意以下：

a、区分度最大的字段放在前面

b、核□SQL 优先考虑覆盖索引

c、避免冗余和重复索引

d、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

4.5 索引禁忌

a、不在低基数列上建立索引，例如“性别”

b、不在索引列进行数学运算和函数运算

4.6 尽量不使用外键

a、外键用来保护参照完整性，可在业务端实现

b、对父表和子表的操作会相互影响，降低可用性

4.7 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

4.8 新建的唯一索引必须不能和主键重复

4.9 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。

4.10 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

4.11 能使用唯一索引就要使用唯一索引，提高查询效率

4.12 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

5. SQL 规范

5.1 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)

5.2 事务要简单，整个事务的时间长度不要太长

5.3 避免使用触发器、函数、存储过程

5.4 降低业务耦合度，为 sacle out、sharding 留有余地

5.5 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)

5.6 不要用 select *，查询哪几个字段就 select 这几个字段

5.7 sql 中使用到 OR 的改写为用 IN() (or 的效率没有 in 的效率高)

5.8 in 里面数字的个数建议控制在 1000 以内

5.9 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 t limit 10000, 10; => select id from t where id > 10000 limit 10;

5.10 使用 union all 替代 union

5.11 避免使□大表的 JOIN

5.12 使用 group by 分组、自动排序

5.13 对数据的更新要打散后批量更新，不要一次更新太多数据

5.14 减少与数据库的交互次数

5.15 注意使用性能分析工具：Sql explain / showprofile / mysqlsla

5.16 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

5.17 SQL 语句不可以出现隐式转换，比如：select id from 表 where id='1'

5.18 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

5.19 能不用 NOT IN 就不用 NOTIN，坑太多了。。会把空和 NULL 给查出来

5.20 在 SQL 语句中，禁止使用前缀是%的 like

5.21 不使用负向查询，如 not in/like

5.22 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

5.23 禁止在数据库中跑大查询

5.24 使用预编译语句，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

5.25 禁止使用 order by rand()

5.26 禁止单条 SQL 语句同时更新多个表

6. 流程规范

6.1 所有的建表操作需要提前告知该表涉及的查询 sql；

6.2 所有的建表需要确定建立哪些索引后才可以建表上线；

6.3 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；

6.4 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

6.5 批量导入、导出数据必须提前通知 DBA 协助观察

6.6 禁□在线上从库执行后台管理和统计类查询

6.8 禁□有 super 权限的应用程序账号存在

6.9 推广活动或上线新功能必须提前通知 DBA 进□行流量评估

6.10 不在业务高峰期批量更新、查询数据库

//数据库
create database push_detail default character set=utf8;
//日志库
create database push_log default character set=utf8;

//建数据表
```
CREATE TABLE `house_refresh_log` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '自增 ID',
  `fangid` int(11) NOT NULL COMMENT '房贴子 ID',
  `refresh_time` int(11) NOT NULL COMMENT '刷新时间',
  PRIMARY KEY (`id`),
  UNIQUE KEY `fangid` (`fangid`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='房刷新记录表'


CREATE TABLE `wanted_post` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `puid` int(10) unsigned NOT NULL,
  `user_id` int(10) NOT NULL COMMENT '发贴用户的 id',
  `username` varchar(50) NOT NULL COMMENT '发贴用户的用户名',
  `city` smallint(4) NOT NULL COMMENT '所在城市',
  `ip` bigint(14) NOT NULL COMMENT '发帖人的 ip',
  `district_id` tinyint(2) NOT NULL COMMENT '所在区域的 id',
  `district_name` varchar(20) NOT NULL COMMENT '行政区名字',
  `street_id` tinyint(2) NOT NULL COMMENT '所在街道(地标)的 id',
  `street_name` varchar(20) NOT NULL COMMENT '小区名字',
  `title` varchar(255) NOT NULL COMMENT '帖子的标题',
  `description` text NOT NULL COMMENT '帖子详情描述',
  `post_at` int(11) NOT NULL COMMENT '用户发帖时间,数据创建的时间,使用整型存储',
  `refresh_at` int(11) NOT NULL COMMENT '帖子被修改的时间,整型存储',
  `show_time` int(11) NOT NULL COMMENT '帖子显示时间',
  `age_max` int(11) NOT NULL DEFAULT '0' COMMENT '招聘最小年龄',
  `age_min` int(11) NOT NULL DEFAULT '0' COMMENT '招聘最大年龄',
  `post_refresh_at` int(11) NOT NULL COMMENT '刷新时间',
  PRIMARY KEY (`id`),
  UNIQUE KEY `idx_puid` (`puid`),
  KEY `user_id_index` (`user_id`),
```

```
  KEY `post_at_index` (`post_at`),
  KEY `refresh_at_index` (`refresh_at`),
  KEY `show_time_index` (`show_time`)
) ENGINE=InnoDB AUTO_INCREMENT=55295 DEFAULT CHARSET=utf8 COMMENT='招聘
帖子表
```