有利网(FUSCENT)MYSQL 开发规范

V1.1

文档修订记录

日期	版本	说明	作者
2014-09-05	1.0	创建	尹延涛
2015-10-28	1.1	更新	袁建强

目录

1 命名规范

1.1 命名规范

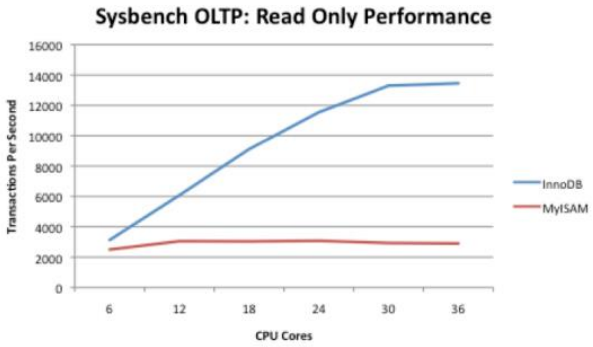
- 库名、表名、字段名必须使用小写字母，并采用下划线进行分割。
- 库名、表名、字段名禁止超过 32 个字符。须见名知意。
- 库名、表名、字段名禁止使用 MySQL 保留字。
- 临时库、表名必须以 tmp 为前缀，并以日期为后缀。
- 备份库、表必须以 bak 为前缀，并以日期为后缀。

- 索引命名以 idx_为前缀。唯一索引以 uniq_为前缀

2 基础规范

2.1 基础规范

- ✓ 使用 INNODB 存储引擎
- ✓ 表字符集使用 UTF8
- ✓ 所有表都需要添加注释
- ✓ 所有的列都需要添加注释
- ✓ 单表数据量建议控制在 5000W 以内
- ✓ 不在数据库中存储图片、文件等大数据
- ✓ 禁止在线上做数据库压力测试
- ✓ 禁止从测试、开发环境直连数据库



Feature	InnoDB	MyISAM
ACID Transactions	Yes	No
Configurable ACID Properties	Yes	No
Crash Safe	Yes	No
Foreign Key Support	Yes	No
Row-Level Locking Granularity	Yes	No (Table)
MVCC	Yes	No

3 库表设计

3.1 库表设计

- 禁止使用分区表
- 拆分大字段和访问频率低的字段，分离冷热数据
- 用 HASH 进行散表，表名后缀使用十进制数，下标从 0 开始
- 按日期时间分表需符合 YYYY[MM][DD][HH]格式
- 采用合适的分库分表策略。例如千库十表、十库百表等
- 建表时自增列的类型为 int 或者 bigint，指定初始值为 1
- 自增列必须是无符号型　unsigned
-

3.2 字段设计

- ✧ 尽可能不使用 TEXT、BLOB 类型
- ✧ 用 DECIMAL 代替 FLOAT 和 DOUBLE 存储精确浮点数
- ✧ 将字符转化为数字
- ✧ 使用 TINYINT 来代替 ENUM 类型
- ✧ 存储 “hello”时 VARCHAR(5) VSVARCHAR(200)

Value	CHAR(4)	Storage Required	VARCHAR(4)	Storage Required
''	' '	4 bytes	''	1 byte
'ab'	'ab '	4 bytes	'ab'	3 bytes
'abcd'	'abcd'	4 bytes	'abcd'	5 bytes
'abcdefgh'	'abcd'	4 bytes	'abcd'	5 bytes

The best strategy is to allocate only as much space as you really need.

- ✧ 所有字段均定义为 NOT NULL
- ✧ 使用 UNSIGNED 存储非负整数
- ✧ INT 类型固定占用 4 字节存储
- ✧ 使用 timestamp 存储时间
- ✧ 禁止在数据库中存储明文密码

3.3 索引设计

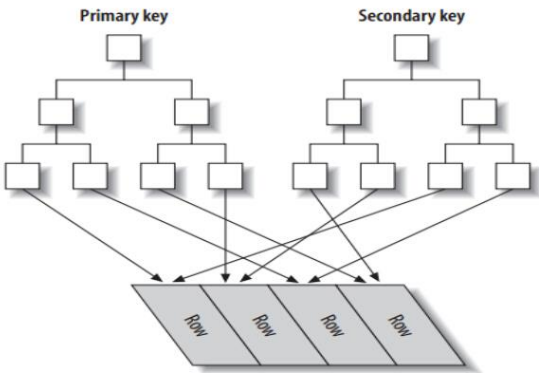
3.3.1 索引的用途

- ➢ 去重
- ➢ 加速定位
- ➢ 避免排序
- ➢ 覆盖索引

3.3.2 索引数量的控制

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

3.3.3 主键的准则

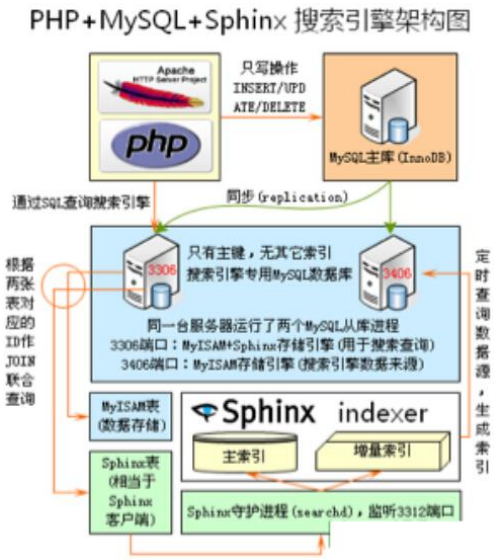


- 表必须有主键
- 不使用更新频繁的列
- 尽量不选择字符串列
- 不使用 UUID MD5 HASH
- 默认使用非空的唯一键
- 建议选择自增或发号器

- **重要的 SQL 必须被索引**
- ✓ UPDATE、DELETE 语句的 WHERE 条件列
- ✓ ORDER BY、GROUP BY、DISTINCT 的字段
- ✓ 多表 JOIN 的字段
- **区分度最大的字段放在前面**
- **核心 SQL 优先考虑覆盖索引**
- **避免冗余和重复索引**
- **索引不是越多越好**
- ✓ 综合评估数据密度和分布
- ✓ 考虑查询和更新比例

索引是一把双刃剑：降低插入和更新速度，占用磁盘空间

- **索引禁忌**
- ✓ 不在低基数列上建立索引，例如“性别”
- ✓ 不在索引列进行数学运算和函数运算
- **尽量不使用外键**
- ✓ 外键用来保护参照完整性，可在业务端实现
- ✓ 对父表和子表的操作会相互影响，降低可用性
- ✓ INNODB 本身对 onlineDDL 的限制
- 不使用%前导的查询，如 like “%ab”
- 不使用负向查询，如 not in/like
- ✓ 无法使用索引，导致全表扫描
- ✓ 全表扫描导致 bufer pool 利用率降低



3.4 SQL 的设计

3.4.1 SQL 设计

- DML 语句（insert，update，delete）
  - ✓ 禁止在 DML 语句中使用 LIMIT
  - ✓ 禁止在 DML 语句中使用 order by
  - ✓ DML 语句中必须包含 where 条件
- 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
- 避免隐式转换
  - ✓ 会导致索引失效
- 充分利用前缀索引
  - ✓ 必须是最左前缀
  - ✓ 不可能同时用到两个范围条件

idx_col1_col2_col3(col1,col2,col3)
select * from tbl_name where col1=val1 and clo2=val2;
~~select * from tbl_name where col2=val2 and col3=val3;~~
~~select * from tbl_name where col1=100 and col2 between 1000 and 2000 and col3 > 3000~~

- 避免使用存储过程，触发器，UDF，events 等
  - ✓ 让数据库做最擅长的事
  - ✓ 降低业务耦合度,为 sacle out、sharding 留有余地
  - ✓ 避开 BUG

- 避免使用大表的 JOIN
  - ✓ MySQL 最擅长的是单表的主键/二级索引查询
  - ✓ JOIN 消耗较多内存，产生临时表

- 避免在数据库中进行数学运算
  - ✓ MySQL 不擅长数学运算和逻辑判断
  - ✓ 无法使用索引

~~md5()/order by rand()~~
~~select ... where to_days(current_date) - to_days(date_col)<=10~~
select ... where date_col>=date_sub(current_date, interval 10 day)
select ... where date_col>=date_sub('2013-08-17', interval 10 day)
select ... where date_col>='2013-08-07'

数据库是有状态的服务，调整代码部署更灵活、简单、高效!

- 减少与数据库的交互次数
- ✓ INSERT ... ON DUPLICATE KEY UPDATE
- ✓ REPLACE INTO、INSERT IGNORE 、INSERT INTO VALUES(),(),()
- ✓ UPDATE … WHERE ID IN(10,20,50,…)

- 拒绝大 SQL，拆分成小 SQL
- ✓ 充分利用 QUERY CACHE
- ✓ 充分利用多核 CPU

```
select * from profiles where sex='M' order by rating limit 10;
select * from profiles where sex='M' order by rating limit 100000,10;
select * from profiles inner join (select <pk> from pfofiles where
x.sex='M' order by rating limit 100000,10) as x using (<pk>)
```

- 使用 in 代替 or, in 的值不超过 1000 个
- 禁止使用 order by rand()
- 使用 EXPLAIN 诊断，避免生成临时表
- 用 union all 而不是 union
- 程序应有捕获 SQL 异常的处理机制
- 禁止单条 SQL 语句同时更新多个表
- 不使用 select *
- ✓ 消耗 CPU 和 IO、消耗网络带宽
- ✓ 无法使用覆盖索引
- ✓ 减少表结构变更带来的影响
- ✓ 因为 select/join 可能生成临时表

~~select * from opp where phone='12345678' or phone='234234234'~~
select * from opp where phone in ('12345678', '234234234')
~~select * from app where phone='010-88886666' or cellphone='18618111111'~~
select * from opp where phone='010-88886666'
union all
select * from opp where cellphone=' 18618111111'

4 行为规范

4.1 行为规范

- ✧ 批量导入、导出数据必须提前通知 DBA 协助观察
- ✧ 禁止在从库上执行后台管理和统计类功能的查询
- ✧ 禁止有 super 权限的应用程序账号存在
- ✧ 产品出现非数据库导致的故障时及时通 DBA 协助排查
- ✧ 推广活动或上线新功能必须提前通知 DBA 进行流量评估
- ✧ 数据库数据丢失，及时联系 DBA 进行恢复
- ✧ 对单表的多次 alter 操作必须合并为一次操作
- ✧ 不在 MySQL 数据库中存放业务逻辑
- ✧ 重大项目的数据库方案选型和设计必须提前通知 DBA 参与
- ✧ 对特别重要的库表，提前与 DBA 沟通确定维护和备份优先级
- ✧ 不在业务高峰期批量更新、查询数据库
- ✧ 提交线上建表改表需求，必须详细注明所有相关 SQL 语句

良好的线上环境需要大家共同的努力!

PS: 猛击吧，请提出你的建议！请 mail:yantao.yin@yooli.com