密集	内部公开

鸟域道场

产品研发 MySQL 数据库设计规范

（V1.0）

Nander 研发团队

2023 年 3 月

文档信息

文档编号				发布版本	
起草时间	2023-03-01	定稿时间		当前版本	V1.0
起草人	姓名	部门	电话	电子邮件	
	商鞅	鸟域行政司		26089183@qq.com	
审阅人			签发人		

文档修改记录

序号	修改时间	修改人	主要修改内容	存档版本
1				
2				
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				
15				

目录

1 目的 4
2 适用范围....4
3 数据库设计规范....4
3.1 基础规范.... 4
3.2 命名规范.... 4
3.3 表设计规范.... 4
3.4 字段设计规范.... 5
3.5 索引设计规范.... 6
4 SQL书写规范 7
5 工具 7
5.1 archery.... 7
6 附则 8
7 参考文献....8

1 目的

为了提升研发、测试、运维等技术人员基于MySQL做出适合线上业务的数据库设计，从而为业务系统稳定、健康的运行状态提供保障，需要在处理数据库变更、数据库表设计、SQL编写等方面予以约束，特制定本规范。

2 适用范围

本规范适用于组织内所有自研产品与交付类项目的基于 MySQL 的数据库设计。

3 数据库设计规范

3.1 基础规范

1、【强制】库、表、列所有字符集必须保持一致，使用utf8mb4。

- 多表 JOIN 时字符集不一致会造成索引失效。
- utf8mb4 是 utf8 的超集并完全兼容，同时支持表情等字符。

2、【强制】数据表、数据字段必须加入中文注释。

3、【强制】避免使用数据库存储大数据文件，比如图片、文档等可以改成保存URL。如有场景需要存储文件时需审批通过。

- 使用数据库存储大文件会浪费更多的磁盘和内存空间，非必要的大量的大字段查询会淘汰掉热数据导致内存命中率急剧降低，影响数据库性能。

4、【建议】避免使用存储过程、视图、触发器、Event，业务都交给程序处理。

- 对数据库性能影响较大，调试、排错、迁移都比较困难，扩展性较差。

3.2 命名规范

1、【强制】库名、表名、字段名和索引名命名规范：小写、下划线分隔、不超过32个字符，必须见名知意，避免拼音英文混用。

2、【强制】临时表：tmp_xxx，备份表：bak_xxx，数据表：td_xxx，关系表：tr_xxx，普通索引名：idx_xxx，唯一索引名：uniq_xxx。

3、【强制】禁用保留字，如desc、range、match、delayed等，请参考MySQL官方保留字。

3.3 表设计规范

1、【强制】创建表时必须显式指定表存储引擎类型，如无特殊需求一律为InnoDB，索引类型

必须为BTREE。

- InnoDB 支持事务、行级锁、并发性能更好、CPU 及内存缓存页优化使得资源利用率更高。

2、【强制】表名不使用复数名词。

- 说明：表名应该仅仅表示表里面的实体内容，不应该表示实体数量。

3、【强制】表必须有主键，例如自增主键，推荐使用BIGINT UNSIGNED为主键。

- 主键递增，写入数据行时可以提高插入性能，避免 page 分裂，减少表碎片。
- 主键要选择较短的数据类型；使用 InnoDB 引擎时普通索引都会保存主键的值，较短的数据类型可以有效地减少索引的磁盘空间，提高索引的缓存效率。
- 在 row 模式的主从架构下删除无主键表的数据时会导致备库夯住。

4、【强制】禁止使用外键，如果有外键完整性约束，需要应用程序控制。

- 外键会导致表与表之间耦合，UPDATE 与 DELETE 操作都会涉及相关联的表，十分影响 SQL 的性能，甚至会造成死锁。

5、【强制】每个表都定义以下三列：

- created_time（创建时间）、update_time（修改时间）、deleted（是否删除）

6、【建议】建议将大字段、访问频度低的字段拆分到单独的表中存储，分离冷热数据。

7、【建议】字段允许适当冗余以提高查询性能，但需考虑数据一致性。冗余字段应遵循：

- 不是频繁修改的字段。
- 不是 VARCHAR 超长字段，更不能是 TEXT 字段。

示例代码：

```
CREATE TABLE `td_user`  (
  `id` bigint(20) NOT NULL COMMENT '雪花 ID',
  `mobilephone` varchar(11) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL COMMENT '手机号',
  `password` varchar(32) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NULL DEFAULT NULL COMMENT '密码',
  `salt` varchar(4) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NULL DEFAULT NULL COMMENT '盐值',
  `name` varchar(11) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NOT NULL COMMENT '姓名',
  `avatar` varchar(100) CHARACTER SET utf8mb4 COLLATE utf8mb4_bin NULL DEFAULT NULL COMMENT '头像',
  `status` tinyint(4) NOT NULL DEFAULT 0 COMMENT '状态',
  `internal` bit(1) NOT NULL DEFAULT b'0' COMMENT '是否内部用户',
  `create_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP COMMENT '创建时间',
  `update_time` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP
  COMMENT '更新时间',
  `deleted` bit(1) NOT NULL DEFAULT b'0' COMMENT '是否删除',
  PRIMARY KEY (`id`) USING BTREE,
  UNIQUE INDEX `uniq_mobilephone_internal`(`mobilephone`, `internal`) USING BTREE
) ENGINE = InnoDB CHARACTER SET = utf8mb4 COLLATE = utf8mb4_bin COMMENT = '用户信息表' ROW_FORMAT
```

3.4 字段设计规范

1、【强制】必须把字段定义为NOT NULL。

- 使用 NULL 值会存在以下问题：每一行都会占用额外存储空间、数据迁移容易出错、聚合函数计算结果偏差。

2、【强制】避免使用TEXT、BLOB类型，如有场景使用时需审批通过。

- 会浪费更多的磁盘和内存空间，非必要的大量的大字段查询会淘汰掉热数据，导致内存命中率急剧降低，影响数据库性能。

3、【强制】禁止使用ENUM，可使用TINYINT代替。

- 可维护性差，增加新的 ENUM 值要做 DDL 操作。
- 数据迁移的时候，其他数据库可能不支持。

4、【强制】禁止单独设置列的字符集。

5、【建议】浮点类型字段建议使用DECIMAL，避免带来精度损失。

6、【建议】根据业务区分使用CHAR/VARCHAR；字段长度固定，或者长度近似的业务场景，适合使用CHAR，能够减少碎片，查询性能高；字段长度相差较大，或者更新较少的业务场景，适合使用VARCHAR，能够减少空间。

7、【建议】根据业务区分使用TINYINT/INT/BIGINT，分别会占用1/4/8字节。

8、【建议】根据业务区分使用DATETIME/TIMESTAMP；前者占用8个字节，后者占用4个字节，存储年使用YEAR，存储日期使用DATE，存储时间使用DATETIME(3)。

- timestamp 存储的时间与时区有关，变换时区数据会受影响；datetime 与时间无关；
- timestamp 存储时间范围小，1970-01-01 00:00:00 到 2038-01-19 03:14:07；datetime 存储时间范围大，1000-01-01 00:00:00 到 9999-12-31 23:59:59。

3.5 索引设计规范

1、【强制】单表索引控制在5个以内。

- 高并发业务使用太多索引会影响写性能。
- 生成执行计划时，如果索引太多会降低性能，并可能导致 MySQL 选择不到最优索引。

2、【强制】单索引字段数禁止超过5个。

- 字段超过 5 个时，实际已经起不到有效过滤数据的作用。

3、【强制】禁止在更新频繁、区分度不高的字段上建索引。

- 更新会变更 B+树，更新频繁的字段建立索引会大大降低数据库性能。
- “性别”这种区分度不大的属性不能有效过滤数据，性能与全表扫描类似。

4、【强制】建立组合索引，必须把区分度高的字段放前面。

- 能够更加有效地过滤数据。

5、【强制】非必要不要进行JOIN查询，如果要进行JOIN查询，被JOIN的字段必须类型相同并建立索引。

- JOIN 字段类型不一致会导致全表扫描。

6、【强制】业务上有唯一特性的字段，即使使用其创建组合索引，也必须建成唯一索引，这样既可以提高查询速度，也可以有效规避脏数据产生。

7、【建议】分页搜索避免左模糊或者全模糊，请使用搜索引擎。

- 禁止使用 like %str 和 like %str%，因为不走索引。
- 使用 like str%，走索引。

8、【建议】理解组合索引最左前缀原则，避免重复创建索引。

- 如果建立了(a,b,c)，相当于建立了(a)，(a,b)，(a,b,c)。

9、【建议】善于利用覆盖索引来进行查询操作，避免回表。

10、【建议】如果用到ORDER BY，注意利用上索引的有序性；ORDER BY 最后的字段是组合索引的一部分，并且放在索引组合顺序的最后，避免出现file_sort的情况，影响查询性能。

4 SQL 书写规范

1、【强制】禁止使用SELECT *，只获取必要字段，因为读取不需要的列会增加CPU、IO、网络消耗。

2、【强制】禁止使用INSERT INTO table VALUES(xxx_fields)，必须显式指定列名，避免列变动导致BUG。

3、【强制】语句中字段类型值要准确表达，不要使用属性隐式转换，以免索引失效。

- 比如：字段 name 为 VARCHAR 类型，查询时传递了整数类型的值会产生隐式转换，造成索引失效。

4、【强制】禁止跨db的JOIN语句。

- 可以减少模块间耦合，为数据库拆分奠定坚实基础。

5、【建议】避免在WHERE条件的字段上使用函数或者表达式，以免索引失效。

6、【建议】避免负向查询，以及%开头的模糊查询，避免全表扫描。

- 负向查询条件：NOT、!=、<>、!<、!>、NOT IN、NOT LIKE 等。

7、【建议】SQL优化目标：至少达到range级别，要求是ref级别，最好能到const级别。

5 工具

5.1 archery

archery是一站式的SQL审核查询平台，该平台集成了goInception工具可对上线的SQL语句规范化进行自动审核，特殊情景可通过人工审核确保语句合理性；同时提供了丰富的SQL优化工具，协助开发人员和DBA高效、快速地优化语句。

■ 官网：Archery

6 附则

- 本规范自发布之日起实施。
- 本规范由鸟域道场进行解释。
- 本文件著作权归北京鸟域花香科研室所有，用户除认真学习外不得以任何理由在未经同意的情况下擅自将文件用于其他用途。

7 参考文献

《阿里开发手册 mysql_MySQL 开发准则》