**MySQL开发规范**

**修订历史记录**

|  |  |  |  |
| --- | --- | --- | --- |
| **日期** | **版本** | **说明** | **修订** |
|  |  |  |  |
| 2016.06.26 | 1.0 | MySQL 开发规范 v1.0 | 辛鹏 |
| 2016.08.18 | 1.0 | 格式内容稍作调整 | 辛鹏 |
| 2016.12.16 | 1.0 | 添加常用计量单位说明，DML语句增加内容 | 辛鹏 |
| 2017.07.07 | 1.0 | 增加DDL语句规范（修改字段名） | 辛鹏 |
| 2017.8.14 | 2.0 | 增加【强制】选项及新增规范 | 李友伟 |
|  |  |  |  |
|  |  |  |  |

目录

[1 编写目的与适用范围 5](#_Toc494120123)

[1.1 编写目的 5](#_Toc494120124)

[1.2 适用范围 5](#_Toc494120125)

[2 数据库对象命名规范 5](#_Toc494120126)

[2.1 原则 5](#_Toc494120127)

[2.2 命名规范 5](#_Toc494120128)

[2.2.1 数据库命名规范 5](#_Toc494120129)

[2.2.2 表命名规范 6](#_Toc494120130)

[2.2.3 字段命名规范 6](#_Toc494120131)

[2.2.4 索引命名规范 6](#_Toc494120132)

[2.2.5 约束规范 6](#_Toc494120133)

[2.2.6 视图命名规范 7](#_Toc494120134)

[2.2.7 存储过程命名规范 7](#_Toc494120135)

[2.2.8 函数命名规范 7](#_Toc494120136)

[2.2.9 触发器命名规范 7](#_Toc494120137)

[3 数据库对象设计规范 7](#_Toc494120138)

[3.1 存储引擎的选择 7](#_Toc494120139)

[3.2 字符集的选择 8](#_Toc494120140)

[3.3 数据库设计规范 8](#_Toc494120141)

[3.4 表设计规范 8](#_Toc494120142)

[3.4.1 基本原则 8](#_Toc494120143)

[3.4.2 约束字段规范 9](#_Toc494120144)

[3.4.3 适当纵向拆分 10](#_Toc494120145)

[3.4.4 适度冗余 10](#_Toc494120146)

[3.4.5 大表水平拆分 10](#_Toc494120147)

[3.5 字段数据类型选择规范 10](#_Toc494120148)

[3.5.1 数据类型选择原则 10](#_Toc494120149)

[3.5.2 常用计量单位说明 10](#_Toc494120150)

[3.5.3 数字类型 11](#_Toc494120151)

[3.5.4 字符类型 11](#_Toc494120152)

[3.5.5 时间类型 11](#_Toc494120153)

[3.5.6 LOB类型 12](#_Toc494120154)

[3.5.7 数据类型转换函数 12](#_Toc494120155)

[3.5.8 常用数据类型 12](#_Toc494120156)

[3.6 索引设计规范 13](#_Toc494120157)

[3.6.1 索引的优点： 13](#_Toc494120158)

[3.6.2 索引的缺点： 13](#_Toc494120159)

[3.6.3 设计索引原则 13](#_Toc494120160)

[3.7 禁止使用存储过程、视图、触发器、Event 13](#_Toc494120161)

[4 SQL编写规范 14](#_Toc494120162)

[4.1 语句书写规范 14](#_Toc494120163)

[4.1.1 大小写规则 14](#_Toc494120164)

[4.1.2 使用缩进来表达语句的逻辑层次 14](#_Toc494120165)

[4.1.3 语句整洁规则 14](#_Toc494120166)

[4.2 数据类型转换规范 14](#_Toc494120167)

[4.3 函数使用规范 15](#_Toc494120168)

[4.4 SELECT \* 使用规范 15](#_Toc494120169)

[4.5 WHERE 条件规范 15](#_Toc494120170)

[4.6 IN条件规范 16](#_Toc494120171)

[4.7 OR操作规范 16](#_Toc494120172)

[4.8 UNION语句规范 16](#_Toc494120173)

[4.9 GROUP BY, ORDER BY, DISTINCT语句 16](#_Toc494120174)

[4.10 COUNT(\*)操作 16](#_Toc494120175)

[4.11 分页查询规范 16](#_Toc494120176)

[4.12 随机取数使用规范 17](#_Toc494120177)

[4.13 关联查询规范 17](#_Toc494120178)

[4.14 子查询规范 17](#_Toc494120179)

[4.15 事务使用规范 17](#_Toc494120180)

[4.16 DML语句其他规范 18](#_Toc494120181)

[4.17 DDL语句规范 18](#_Toc494120182)

[4.18 其他规范 18](#_Toc494120183)

[5 SQL语句注意事项 18](#_Toc494120184)

[6 常用函数 19](#_Toc494120185)

[6.1 字符串函数 19](#_Toc494120187)

[6.2 日期函数 19](#_Toc494120188)

[6.2.1 获取当前时间： 19](#_Toc494120189)

[6.2.2 日期数值的加减函数： 19](#_Toc494120190)

[6.2.3 返回日期某部分信息的函数： 20](#_Toc494120191)

[6.3 类型转换函数 20](#_Toc494120192)

[6.3.1 字符串转换成日期方式： 20](#_Toc494120193)

[6.3.2 常用的类型转换函数： 20](#_Toc494120194)

[7 附录1-MySQL的缺陷 20](#_Toc494120195)

[8 附录2-数据库特征 20](#_Toc494120196)

[9 附录3-字符集 21](#_Toc494120197)

[10 附录4-MySQL保留字 21](#_Toc494120198)

# 编写目的与适用范围

## 编写目的

* 实现数据库标准化、统一化的开发、管理与维护工作
* 保障数据库安全、稳定、高效的运行
* 保障数据完整性、数据一致性、数据有效性、数据结构可扩展性、数据易操作性
* 提升工作效率、释放工作压力（省去不必要的沟通，省去反复修改带来的繁琐，避免脚本版本冲突，避免上线后不必要的问题处理）

## 适用范围

第一车贷所有与MySQL相关的开发人员、数据库管理人员及运维人员。

以下所有规范会按照【强制】、【建议】级别进行标注，遵守优先级从高到低。

对于不满足【强制】级别的设计，DBA会强制打回要求修改。

# 数据库对象命名规范

命名规范是指数据库对象如数据库（SCHEMA）、表（TABLE）、索引（INDEX）、约束（CONSTRAINTS）等的命名约定。

## 原则

* 【建议】 命名使用具有实际意义的英文词汇，或有意义的词汇缩写，词汇之间使用下划线分隔
* 【强制】 命名只能使用小写英文字母、数字、下划线，且必须使用英文字母开头，下划线只能用作词汇间分割符
* 【强制】 避免使用MySQL的保留字如：call、group等，具体参考《附录保留字》
* 【强制】 数据库对象名不能超过32个字符
* 【建议】 数据库对象名尽可能简短（打开表操作：将会把表的元数据加载到内存中，并缓存到内存中）

## 命名规范

### 数据库命名规范

* 【强制】数据库命名必须为项目英文名称或有实际业务含义的英文简写
* 【强制】单库数据库名以“\_db”为后缀，分库命名以”\_db\_分库编号”为后缀
* 【强制】备份数据库名在原数据库名的基础上加“\_bak”后缀
* 【强制】 创建数据库时必须显式指定字符集，并且字符集只能是utf8或者utf8mb4

创建数据库SQL举例：create database xxx\_db default character set utf8;

### 表命名规范

* 【强制】同一个功能模块的表使用相同的前缀（有意义的英文简写）
* 【建议】表名称尽可能表达出该表的含义
* 【强制】普通表名命名规则为：功能模块前缀\_+tablename（login\_users）
* 【强制】临时表（运营、开发或数据库人员临时用作数据采集用的中间表）命名规则：tmp前缀+tablename+8位时间后缀（tmp\_login\_users\_20170501）
* 【强制】备份表（运营、开发或数据库人员备份用作保存历史数据的中间表）命名规则：bak前缀+tablename+8位时间后缀（bak\_login\_users\_20170501）；存放在备份数据库中，超出恢复时间范围的备份表需及时清理
* 【强制】 归档表命名规则：arch前缀+tablename+归档规则 (arch\_login\_users\_2013）

### 字段命名规范

* 【建议】命名尽可能表达出该字段的含义
* 【强制】各表之间相同意义的字段必须同名，数据类型、长度、单位必须相同

**常用约定：**

* 【建议】序号列字段：以id为后缀，如：user\_id表示用户编号
* 【建议】编码字段：以code为后缀，如：cust\_code表示客户编码
* 【建议】布尔值字段：以“is\_”前缀+字段描述+形容词。如member表上表示为enabled的会员的列命名为is\_member\_enabled
* 【强制】状态字段：以“\_status”为后缀，前面加业务逻辑名。如：用户状态可命名为user\_status

### 索引命名规范

* 【强制】二级（辅助）索引以idx\_开头，唯一索引以uq\_idx\_开头，后面跟索引所在字段名
* 【强制】多单词组成的列名，取尽可能代表意义的缩写，如t\_user\_contacts表member\_id和friend\_id上的组合索引：idx \_mid\_fid
* 【强制】组合索引命名应注意字段顺序。如在字段member\_id和字段user\_id上创建组合索引，则可以命名为idx \_uid\_mid(userid, member\_id)

### 约束规范

* 【强制】唯一约束: uk\_字段名
* 【强制】非空约束：如无特殊需要，建议所有字段默认非空，不同数据类型必须给出相应类型的默认值，字符串可以设置为 NOT NULL DEFAULT ‘’ （因为使用NULL值会存在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等问题）
* 【强制】外键约束：以fk\_为前缀，后面紧跟该外键所在的表名、字段名和对应的主表名和字段名
* 【强制】出于性能考虑，如无特殊需要，不建议使用外键，数据完整性由应用程序控制

### 视图命名规范

* 【强制】视图名以“v\_模块名”为前缀，表示view，以实现的功能为后缀。如：表示活跃用户的视图v\_login\_users\_active
* 【建议】如无特殊需要，严禁使用创建视图（性能、维护）

### 存储过程命名规范

* 【强制】存储过程名以“proc\_模块名”为前缀，表示procedure。存储过程名称中应体现其功能
* 【建议】存储过程中的输入参数以“i\_”为前缀，输出参数以“o\_”为前缀
* 【建议】如无特殊需要，严禁使用存储过程（性能、维护）

### 函数命名规范

* 【强制】函数名以“func\_模块名”为前缀，表示function。函数命名中应体现其功能
* 【建议】函数中输入参数以“i\_”为前缀，输出参数以“o\_”为前缀
* 【建议】如无特殊需要，严禁使用函数

### 触发器命名规范

* 【强制】触发器以“tri\_模块名”为前缀，紧跟相关表名，表示trigger。以（\_i，\_u，\_d）为后缀，表示触发条件的触发方式（insert,update或delete）
* 【建议】如无特殊需要，严禁使用触发器。

# 数据库对象设计规范

## 存储引擎的选择

如无特殊要求，默认使用innodb存储引擎，该引擎为5.5.5版本以后的默认存储引擎。

|  |  |
| --- | --- |
| **MySQL引擎** | **说明** |
| InnoDB | 索引和数据都可以缓存到内存中； |
| 支持事务； |
| 支持行级锁，可实现更高的并发度； |
| 支持故障恢复（crash recovery） |
| 支持MVCC特性 |
| 支持4种不同的事务隔离级别 |

## 字符集的选择

* 【强制】自开发系统的数据库字符集默认选择utf8mb4（utf8）
* 【建议】外购系统的字符集按照厂商要求选择，需提前说明潜在的风险

## 数据库设计规范

* 【强制】根据业务模块进行分库设计（对业务进行解耦后将相应业务模块的表对应到相关数据库）
* 【建议】不同业务模块所对应的数据库表之间的关联应尽可能少，需要关联的表的数据完整性由应用程序控制
* 【建议】控制单库表个数，建议单实例不超过500个表（特殊情况再说）
* 【建议】创建数据库的语句必须包含字符集子句和校对规则子句（由DBA创建）。如：CREATE DATABASE ... DEFAULT CHARACTER SET UTF8。

## 表设计规范

### 基本原则

* 【建议】表设计应尽量满足第三范式，尽量减少数据冗余，特殊场景允许反范式化设计
* 【强制】创建表的语句必须包含引擎子句和字符集子句，默认使用innodb和utf8mb4（utf8）

create table t (id int) engine=innodb default charset = utf8

* 【建议】单表字段数不要太多，尽可能的精简表结构；建议最多不要大于50个，且尽可能的少用字符型数据类型（BI系统另说）
* 【强制】表必须有主键，如果使用auto\_increment作为自增主键，注意将起始值设置为1.因为如果设为主键且主键值为随机插入，则会导致innodb内部page分裂和大量随机I/O，性能下降
* 【强制】表和列定义的时候必须加上comment，并能精确描述表和列的含义。类型、状态等字段必须明确给出各个值代表的含义；金钱等计量字段必须给出精确的计量单位；外键字段必须明确给出关联的表和字段
* 【强制】每个表都必须包含两个保留字段：create\_time（创建日期），update\_time（最后修改日期），设置为非空字段属性。这两个字段不应该包含额外的业务逻辑。默认值分别使用：CURRENT\_TIMESTAMP 和 CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP
* 【强制】每个表必须设置is\_del（0为未删除，1为删除）标记位字段，设置为非空，默认为0的字段属性，生产环境不允许物理删除
* 【建议】每个表必须包含version 版本字段，用来标记表的版本变更，版本规则应与应用程序相对应（暂时不使用）
* 【强制】 如无特殊需要，所有字段必须使用NOT NULL属性，业务可以根据需要定义DEFAULT值（MySQL NULL类型和Oracle的NULL有差异，会进入索引中。此外，NULL在索引中的处理也是特殊的，也会占用额外的存放空间；在表中存储也会占用空间标识位。另外容易引发数据迁移出错、聚合函数计算结果偏差等问题）
  + where name!=’xinpeng’，如果存在name为null值的记录，查询结果就不会包含name为null值的记录
  + count(col)，不统计NULL值行
  + null的列使索引/索引统计/值比较都更加复杂，对MySQL来说更难优化
  + null 类型MySQL内部需要进行特殊处理，增加数据库处理记录的复杂性；同等条件下，表中有较多空字段的时候，数据库的处理性能会降低很多（myisam\_stats\_method）
  + null值需要额外的存储空间，无论是表还是索引中每行中的null的列都需要额外的空间来标识
  + 对null 的处理时候，只能采用is null或is not null，而不能采用=、in、<、<>、!=、not in这些操作符号
* 【建议】表中尽可能避免有冗余字段，比如：年月日用不同的字段设计是不推荐的（特殊情况另说）
* 【强制】需要JOIN的字段（连接键），字段名称、数据类型、长度和单位必须保持绝对一致，避免隐式转换
* 【强制】无特殊需求，严禁使用分区表，使用自定义分表规则控制
* 【强制】无特殊需求禁止使用TEXT、BLOB类型（大文本、大文件、大照片存放在文件系统）
* 【建议】不推荐使用enum，set。因为它们浪费空间，且枚举值写死了，变更不方便。推荐使用tinyint或smallint
* 【建议】业务中IP地址字段推荐使用int类型，不推荐用char(15)

因为int只占4字节，可以用如下函数相互转换，而char(15)占用至少15字节。一旦表数据行数到了1亿，那么要多用1.1G存储空间！

SQL：select inet\_aton('192.168.2.12'); select inet\_ntoa(3232236044);

### 约束字段规范

* 【强制】表中必须包含PK（尽可能简短），且为无符号整型类型，PK应该是有序整型，由开发人员自定义生成规则（尽可能简短）或使用自增字段
  + 主键递增，数据行写入可以提高插入性能，可以避免页分裂，减少表碎片提升空间和内存的使用
  + 主键选择较短的数据类型,Innodb引擎普通索引都会保存主键的值，较短的数据类型可以有效的减少索引的磁盘空间，提高索引的缓存效率
  + 无主键的表数据变更，在row模式的主从架构，会导致备库延时
* 【强制】PK字段不允许update操作
* 【建议】表中除PK以外，还存在唯一性约束的，可以在数据库中创建以“uq\_idx\_xxx”作为前缀的唯一约束索引，并设置为 NOT NULL
* 【强制】禁止创建外键约束，外键约束由应用程序控制，外键字段必须设置为NOT NULL，并且赋予相应的默认值（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，影响sql 的性能，甚至会造成死锁。高并发情况下容易造成数据库性能问题，大数据高并发业务场景数据库使用以性能优先）

### 适当纵向拆分

当表中存在类似于 TEXT 或者是很大的 VARCHAR类型的大字段的时候，如果大部分访问这张表的时候都不需要这个字段，考虑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，也能提高内存中的缓存命中率。

其他不常用到的字段也可以考虑将其拆分到另外独立的表中。

### 适度冗余

被频繁引用且只能通过 JOIN 2张（或者更多）大表的方式才能得到的独立小字段，考虑设置冗余。由于每次JOIN仅仅只是为了取得某个小字段的值，JOIN到的记录又大，会造成大量不必要的 IO，可以通过空间换取时间的方式来优化。

需要注意，冗余的同时需要确保数据的一致性不会遭到破坏，确保更新的同时冗余字段也被更新。

### 大表水平拆分

* + 【建议】MySQL在处理大表时，性能会明显降低（尤其做DDL等维护操作时），所以建议核心业务表单表物理大小限制在8GB，表中数据控制在2000W内
  + 【建议】如果数据量或数据增长在前期规划时就较大，那么在设计时就应加入分表策略（如：日志表、流水表）
  + 【建议】拆分维度的选择需尽可能满足业务查询特性的需求

## 字段数据类型选择规范

### 数据类型选择原则

* 【强制】数据类型，必须严格按照实际业务数据的需求选择合适的数据类型（根据业务数据类型的实际情况选择对应的字段类型，避免产生脏数据）
* 【建议】满足业务需求的情况下只分配满足需求的最小字符数（跟需求方确认最大字符限度）
* 【建议】尽量选择使用数字类型，数字类型占用空间小，访问速度更快
* 【强制】禁止使用复杂数据类型（ENUM、SET等），使用TINYINT来代替
* 【强制】如无特殊需要，严禁使用TEXT、BLOB类型（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）

**注：更小的字段类型更小的字符数占用更少的内存，占用更少的磁盘空间，占用更少的磁盘IO，占用更少的带宽。**

### 常用计量单位说明

* 时间：DATETIME|DATE|YEAR
* 金钱：DECIMAL(M,2)，单位（元）
* 数量：INT|BIGINT，单位（个数等）
* 里程：INT|DECIMAL(M,2)，单位（公里）
* 利率：INT|DECIMAL(M,5), 无单位，存放利率值
* 面积：INT|DECIMAL(M,2)，单位（平米）
* IP：varchar(15)
* 手机号码：varchar(11)
* 地理位置信息（坐标）：DECIMAL(M,N)，精确到米（8,5）

### 数字类型

* + 【建议】如无特殊需要，存放整型数字使用UNSIGNED INT型。在数据量较大的情况下，建议区分开 TINYINT / INT / BIGINT 的选择
  + 【建议】对于浮点型数据存储，使用DECIMAL，且明确标识为无符号型（UNSIGNED），除非确实会出现负数；严禁使用FLOAT和DOUBLE。对于金钱类等计量字段必须统一计量单位
  + 【建议】精度要求不是很高的地理位置信息可以使用DECIMAL来存放（-180, 180），如：DECIMAL (10,7)
  + 【建议】固定精度的小数，建议乘以固定倍数转换成整数存储，可以节省存储空间、提高性能，且不会带来任何附加维护成本
  + 【建议】IP地址可以考虑转换为INT类型存储
  + 【建议】MySQL中没有BOOLEAN数据类型，使用TINYINT代替

### 字符类型

* 【建议】字符串类型一律使用VARCHAR类型，且只分配满足需求的最大长度，类似于状态等有限类别的字段，也可以使用VARCHAR类型，可以比较明确地表示出实际意义的字符串
* 【建议】如无特殊需要，原则上单个VARCHAR型字段不允许超过256个字符
* 【强制】仅当字符数量可能超过4000个的时候，才使用TEXT类型来存放字符类数据，所有使用TEXT类型的字段必须和原表进行分拆，与原表主键单独组成一个子表进行存放。如无特殊需要，严禁使用MEDIUMTEXT、TEXT、LONGTEXT类型
* 【强制】VARCHAR(N)，N表示的是字符数而不是字节数。比如VARCHAR(256)，最大可存储256个字符（字符包括英文字母，汉字，特殊字符等）。但N应尽可能小，因为MySQL server层限制一条记录的最大长度是65535个字节，且存储字符个数由所选字符集决定。如UTF8存储一个字符最大要3个字节，那么VARCHAR在存放占用3个字节长度的汉字时不应超过21845个汉字。同时，在进行排序和创建临时表一类的内存操作时，会使用N的长度申请内存（索引长度：innodb：767字节，myisam：1000字节）

### 时间类型

* 【强制】需要精确到天的字段全部使用DATE类型
* 【强制】需要精确到年的字段全部使用YEAR类型
* 【强制】需要精确到时间（时分秒）的字段使用DATETIME，不建议使用TIMESTAMP类型
* 【建议】不推荐使用INT来存储时间类型，最大值为2038-01-19 11:14:07（int11: 2147483647）
* 【建议】可以使用BIGINT来存储时间戳，并由应用程序生成时间戳（MySQL的时间戳生成函数返回的是INT型，只能到2038）

### LOB类型

* 【强制】严禁在数据库中存放 LOB 类型数据

### 数据类型转换函数

* 【建议】使用UNSIGNED INT存储IP：使用int类型替换char(15)来存储IP，可以使用MySQL提供的INET\_ATON()，INET\_NTOA()函数来进行转换

### 常用数据类型

|  |  |  |
| --- | --- | --- |
| **数据类型** | **取值范围** | **存储需求** |
| CHAR(n) | 0-255个字符 | 跟使用的字符集有关 |
| VARCHAR(n) | 0-65535个字符（汉字为21845） | 跟使用的字符集有关 |
| TINYTEXT | 0-255个字符 | 跟使用的字符集有关 |
| TEXT | 0-65535个字符 | 跟使用的字符集有关 |
| MEDIUMTEXT | 0-(2的24次方-1个字符) | 跟使用的字符集有关 |
| LONGTEXT | 0-(2的32次方-1个字符) | 跟使用的字符集有关 |
| TINYING | (-128-127) 或 (0-255) | 1字节 |
| SMALLINT | (-32768-32767)或（0-65535） | 2字节 |
| MEDIUMINT | (-8388608-8388607)或(0-16777215) | 3字节 |
| INT | (-2147483648-2147483647)或(0-4294967295) | 4字节 |
| BIGINT | (-9223372036854775808-9223372036854775807)或(0-18446744073709551615) | 8字节 |
| DATE | YYYY-MM-DD(1001年到9999年的范围) | 3字节 |
| TIME | HH:MM:SS | 3字节 |
| DATETIME | YYYY-MM-DD HH:MM:SS(1001年到9999年的范围) | 8字节 |
| TIMESTAMP | YYYY-MM-DD HH:MM:SS(1970年到2038年的范围) | 4字节 |
| YEAR | YYYY | 1字节 |
| DECIMAL(M, D) | 整数最大位数（M）为65，小数位数最大（D）为30 | 变长,每4字节存储9个数字 |

## 索引设计规范

### 索引的优点：

* 通过索引扫描数据可以避免全表扫描操作，降低IO操作
* 通过索引扫描可以避免排序操作
* 通过索引扫描可以避免回表扫描（覆盖索引）

### 索引的缺点：

* 占用磁盘空间
* 降低了数据插入速度

### 设计索引原则

* 在选择性高的列上创建索引（下列使用原则都结合该条原则进行判断），能不添加的索引尽量不加
* 作为查询条件使用频繁的字段列上考虑建立索引（特别是核心业务大表）
* 经常需要排序、分组和关联操作的字段考虑建立索引，或将这些字段添加到普通索引后面，形成覆盖索引
* 尽量在字符数少的列上创建索引，对于大字段考虑建立前缀索引。（当page\_size=8k时，默认MySQL InnoDB表的的单个索引长度限制为768个字节，UTF8是三字节，即：768/3<255个字符，当开启innodb\_large\_prefix 上限可以达到1536bytes）
* 索引长度原则上不允许超过32字节，如在超过32字节的VARCHAR列加索引需要DBA评估（如果需要对超过32字节的VARCHAR列整列进行完全匹配，可以新增一个字段，该字段是VARCHAR列的md5值，使用md5来进行完全匹配；这种情况下，无法进行范围查询）
* 选择性差的字段不适合单独创建索引，即使频繁作为查询条件
* 更新非常频繁的字段不适合创建索引
* 组合索引原则：必须使用到该索引中的第一个字段作为条件时才能保证系统使用该索引，否则该索引将不会被使用，并且应尽可能的让条件字段顺序与索引顺序相一致
* 对于确定需要组成组合索引的多个字段，建议尽可能将选择性高并且使用频繁的字段作为组合索引的首个字段
* 组合索引中的字段数建议不超过5个
* 核心SQL优先考虑使用组合索引
* 尽量避免使用外键，让应用维护数据的完整性，避免高并发环境产生锁或死锁
* 单张表的索引数量理论上应控制在5个以内
* 经常有大批量插入、更新操作的表应尽量少建索引

## 禁止使用存储过程、视图、触发器、Event

高并发大数据的互联网业务，架构设计思路是“解放数据库CPU，将计算转移到服务层”，并发量大的情况下，这些功能很可能将数据库拖死，业务逻辑放到服务层具备更好的扩展性，能够轻易实现“增机器就加性能”。数据库擅长存储与索引，CPU计算还是上移吧。

# SQL编写规范

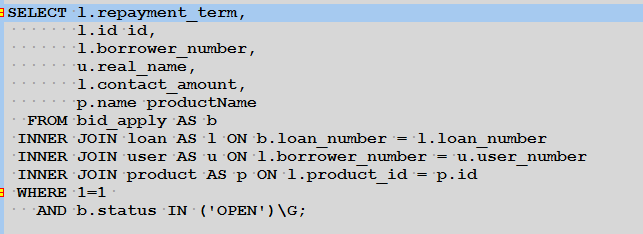
## 语句书写规范

### 大小写规则

在SQL语句中所有的关键字、内置函数名都使用大写，表名、列名使用小写。如下：

SELECT \* FROM tablename;

### 使用缩进来表达语句的逻辑层次



### 语句整洁规则

* 去掉不必要的空格和括号，使代码保持整洁
* 连接符OR、IN、AND、以及＝、<=、>=等符号前后加（且只加）一个空格
* 去掉1=1等无实际意义的条件
* 关键字使用大写，其余一律小写

## 数据类型转换规范

* 所有SQL的WHERE条件中必须使用和过滤字段完全一致的数据类型，杜绝任何隐式类型转换，避免造成因为数据类型不匹配而导致SQL执行计划的出错，造成性能问题。
* 所有WHERE条件的字段上不允许使用函数做类型转换，如有需要转换类型，只能转换变量代入值，而不是转换字段。
* 最为常见的隐式类型转换常见于整型类型与字符串类型之间，如下:

SELECT \* FROM dms\_payment WHERE payment\_number = 201506160001; # 其中payment\_number 为字符串类型

* 表连接SQL中，如果连接条件两端的数据类型不一致，必须保证将驱动表的连接条件数据类型转换为与被驱动表一致的数据类型

SELECT \* FROM test\_info a INNER JOIN test\_customer b ON a.order\_id =CAST(b.order\_id AS UNSIGNED INT) WHERE a.pay\_type= 'cash';

## 函数使用规范

* 插入、更新操作严禁使用NOW()，RAND()，SYSDATE()，CURRENT\_USER()等不确定结果的函数（语句级（STATEMENT）复制场景下、使用正反表设计的环境下、分布式数据库环境下，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QUERY CACHE（鸡肋））
* WHERE条件中的过滤条件字段上避免使用任何函数，包括数据类型转换函数

## SELECT \* 使用规范

* 严禁使用SELECT \* 进行查询，应明确给出要查询的字段，避免不必要的IO、网络、CPU开销

**原因说明：**

1. 增加不必要的IO、网络、CPU开销
2. MySQL数据库进行ORDER BY操作（ORDER BY 列无索引）有两种算法：
   * 第一种是直接取出所有需要返回字段(SELECT后面的字段)，存入内存中，然后排序(仅有需要排序的字段需要参与)。
   * 第二种是先取出需要排序的字段和相应的主键，然后排序，再回表中取出其他的字段，就相当于所有数据都有两次磁盘IO。表现在MYSQL执行计划中为FILESORT。
   * 如果select后面的字段长度总和超过1024字节（即参数max\_length\_for\_sort\_data的默认值）或者字段中包括BLOB、TEXT字段，都将会使用第二种算法。
3. 关联查询使用SELECT \* 可能导致只需要访问索引即可完成的查询需要回表扫描
4. 关联查询、子查询等操作中可以减小临时表的大小
5. MySQL中的TEXT类型字段和Oracle的CLOB一样，存储的时候不是和由其他普通字段类型的字段组成的记录存放在一起，而且读取效率本身也不如普通字段快。如果不需要取回TEXT字段，又使用了SELECT \* ，会让完成相同功能的SQL所消耗的IO量大很多，而且增加部分的IO效率也更低
6. 降低数据库与应用程序的耦合度，在做表结构变更时减小对应用造成的影响

## WHERE 条件规范

* 所有DML语句（除INSERT语句），都必须使用WHERE条件进行条件判断，如有特殊需求，另做说明
* WHERE条件中严禁在字段上进行数学运算或函数运算
* WHERE条件尽可能避免非等值条件（IN,BETWEEN,<,<=,>,>=,<>）
* WHERE条件中禁止单独使用负向查询NOT、!=、<>、!<、!>、NOT IN、NOT LIKE等，会导致全表扫描
* WHERE条件中严禁使用%前缀进行模糊前缀查询，可以使用%模糊后缀查询（Sphinx）

## IN条件规范

* 控制IN的个数应小于200，如果太大考虑拆分SQL
* 当IN的个数越多时，效率越低（特别是没有索引的列），如有可能使用>=、<=操作代替
* IN 条件中使用子查询时尽可能改写SQL为关联查询

## OR操作规范

* 用IN()或者UNION替换OR
* 同一字段的OR操作改写为IN()操作
* 不同字段的OR操作改写为UNION操作（MySQL Merge index特性并不完善）

## UNION语句规范

* 尽可能使用UNION ALL 代替UNION操作（UNION会对结果集进行排序操作）
* UNION ALL不需要对结果集进行排序去重，所以最好在程序中实现UNION操作

## GROUP BY, ORDER BY, DISTINCT语句

* 建立合适的索引：对于这类型语句的优化，通常的办法就是在相关的排序列上创建合适的索引（WHERE key1 ORDER BY key1）
* 尽可能在GROUP BY 和 ORDER BY 子句中的列只涉及一个表中的列，这种情况MySQL才有可能使用索引做排序优化
* 对于GROUP BY语句，可以添加ORDER BY NULL语句避免排序

## COUNT(\*)操作

* 尽可能少的使用COUNT(\*)操作
* 非实时的统计可以考虑中间表定时统计结果
* 如果必须使用，不要使用COUNT(col)，直接使用COUNT(\*)；COUNT(col) 不统计NULL值
* 使用缓存（REDIS、MEMCACHE）做计数器

注：COUNT(\*) MySQL会自动选择最优的方案去执行，优先选择非空字段占用空间最小的索引，最后选择主键。

## 分页查询规范

* 使用BETWEEN AND 来进行范围扫描，在分页语句中提前计算出相关的值，然后进行范围扫描。如：SELECT \* FROM tbl\_name WHERE postion BETWEEN 1000 AND 1010;
* 使用范围条件后再进行LIMIT。如在按价格进行分页的情况下，语句可以调整为：SELECT \* FROM product WHERE price > 1000 ORDER BY price DESC LIMIT 20;
* 考虑延迟回表扫描技巧。一般情况我们都是使用ORDER BY后的数据做分页数据检索，考虑使用索引检索需要的id后再进行关联操作。例：SELECT \* FROM t ORDER BY id LIMIT 10000, 10010;语句可以改写为：SELECT \* FROM t t1 INNER JOIN (SELECT id FROM t ORDER BY id LIMIT 10000, 10010) t2 ON t1.id=t2.id;

## 随机取数使用规范

* 严禁在MYSQL数据库中使用RAND()函数生成随机数
* 严禁直接使用ORDER BY RAND()取随机数，而应在应用中使用其他方法替换
* ORDER BY rand()不会利用索引排序

## 关联查询规范

* 所有连接的SQL必须使用INNER/LEFT JOIN ... ON ...方式进行连接，不允许直接通过WHERE条件关联方式
* 外连接的SQL语句，使用LEFT JOIN ... ON的Join方式，且所有外连接一律写成LEFT JOIN，不使用RIGHT JOIN
* 尽量使用结果集小的表做驱动表
* 关联查询中尽可能避免子查询（衍生表没有索引）
* 在关联字段上创建合适的索引
* 拆分关联查询：考虑将业务中涉及到的复杂SQL拆分成多个简单SQL由应用进行拼接整理
* 原则上不允许出现三个大表以上的关联语句（在高并发系统中只允许单表操作）
* 如无特殊需要，严禁使用STRAIGHT\_JOIN

**注：MySQL 中没有HASH JOIN算法，只有 NESTED LOOP 算法**

## 子查询规范

* 使用WHERE条件限制内外查询的结果集
* 当内查询和外查询返回的结果集较小时可以直接使用子查询
* 当内查询和外查询返回的结果集较大时，使用关联查询替代
* 5.5版本中MySQL对子查询采用EXISTS Strategy优化算法：将IN查询先改写为exists查询（先遍历outer table，对于其返回的每一条记录都执行一次内查询），并且对于子查询内的NULL值需要做特殊处理
* 5.6版本中MySQL增加了semi-join、Materialization+auto hash key+no duplicates方式来优化子查询，并有很多限制条件（如：子查询中存在GROUP BY等），总体而言没有关联查询快

## 事务使用规范

* 事务必须由应用程序控制，显示提交
* MySQL中SELECT语句在事务中时会阻塞DDL语句，单独的SELECT语句最好不要放到事务中执行。
* 必须保证一个事务只能包括单个数据库的相关对象（多线程复制时从库中会有问题）

**注：分库分表策略要考虑到业务实际情况进行解耦**

## DML语句其他规范

* 【强制】不能使用 DELETE 语句，使用逻辑删除替代
* UPDATE 语句 WHERE 条件必须使用主键字段（防止恶意破坏数据库、误操作、提升语句性能）
* UPDATE语句不允许使用LIMIT，否则可能会导致主从数据不一致，同时错误信息会记录到错误日志，占用大量空间
* 关联、子查询语句，必须使用外键做关联条件，不能使用冗余字段做关联（payment\_id，payment\_number）
* 查询语句需要分页的操作必须在SQL语句级进行分页
* INSERT 语句必须指明字段名称，避免后期因表结构变更，影响原有应用
* INSERT语句使用BULK提交时，VALUES的个数不应该过多。BULK提交可以提高写效率，但个数过多，数据恢复需要时间过长

## DDL语句规范

* 对同一表的多个ALTER操作必须合并为一次操作，如：ALTER TABLE t ADD b VARCHAR(10),ADD INDEX idx\_aa(aa);
* 不允许删除表、修改表名、删除字段、修改字段名、缩减字段长度，只允许在原有基础上扩展
* 添加非空字段必须有默认值

## 其他规范

* 进行大批量数据变更等操作时必须分批提交，且避开高峰操作（业务高峰期限制为100条每次）
* 拆分复杂的SQL为多个小SQL，避免大事务（减少锁表时间（特别是MYISAM）、可以使用多核CPU）
* 使用prepared statement 避免SQL注入
* LAST\_INSERT\_ID()函数只能返回当前session最近一次INSERT操作之后所使用到的AUTO\_INCREMENT类型字段的值，使用时只写"SELECT LAST\_INSERT\_ID ()"，不要再跟"FROM table\_name"。应尽量避免使用LAST\_INSERT\_ID ()函数，使用自增长列代替。
* 严禁使用LOCK TABLE语句人为加锁；仅允许使用SELECT \* .. FOR UPDATE语句

# SQL语句注意事项

* 尽可能简单应用MySQL，保持SQL语句尽可能简单（不做，少做，快速做）
* 尽量不在数据库做运算，复杂运算在程序端做
* 避免3B语句（BIG SQL，BIG Transaction，Big Batch），拆分语句，使SQL语句尽可能简单
* 尽量使事务短小
* 严禁使用存储过程、触发器、自定义函数，改由应用程序负责业务逻辑
* 避免不必要的排序，分组等操作
* 避免不必要的COUNT(\*)操作
* 避免不必要的SELECT \* 操作
* 避免使用OR操作，改写查询
* 使用UNION ALL 代替 UNION操作
* GROUP BY 语句尽量避免排序操作
* 谨慎对待子查询操作，如有必要，考虑改写查询
* 避免对大表不必要的全表扫描
* 频繁使用的WHERE 及 ORDER BY操作的列上（如果选择性较高）建立合适的索引
* 避免使用可能引发全表扫描的任何语句，如SELECT \* FROM table\_name等操作，除非必须要这么做。通常来说业务上不会要求查看全部数据，用where条件做限制
* 避免在 WHERE 条件中的字段进行函数、算术运算或其他表达式运算操作
* 避免负向查询和%前缀模糊查询：!=, <>, !<, !>, NOT IN, NOT LIKE, NOT EXISTS等操作
* 条件比较时注意条件两边使用相同的数据类型

# 常用函数



## 字符串函数

* 字符串连接方法，使用CONCAT()或CONCAT\_WS()函数，语法如下：

CONCAT(string1,string2,...)

CONCAT\_WS(separator,string1,string2,..)

* 字符串长度统计：

LENGTH(string)        #返回string所占的字节数

CHAR\_LENGTH(string)      #返回string中的字符个数

## 日期函数



### 获取当前时间：

* NOW()函数精确到秒，格式：YYYY-MM-DD HH:MM:SS
* CURDATE函数精确到天，格式：YYYY-MM-DD
* CURTIME函数精确到秒，格式：HH:MM:SS

### 日期数值的加减函数：

* DATE\_ADD(date,INTERVAL expr type)
* DATE\_SUB(date,INTERVAL expr type)
* DATEDIFF(expr1,expr2)，返回开始日期expr1与结束日期expr2之间，相差的天数 ，返回值为正数或负数
* TIMESTAMPDIFF(type,datetime\_expr1,datetime\_expr2)，返回相差的年月日数

常用的几种type类型：YEAR、MONTH、DAY、HOUR、MINUTE，其中expr可以为正数或负数，我们在开过程中，一般使用DATE\_ADD()函数，若要作日期减去一个数字的方式，就使用负数。

### 返回日期某部分信息的函数：

* YEAR(expr1) 返回日期expr1部分的年份
* MONTH(expr1) 返回日期expr1部分的月份
* DAY(expr1) 返回expr1部分的天数
* WEEKDAY(expr1) 返回expr1对应的星期数字

## 类型转换函数



### 字符串转换成日期方式：

* DATE\_FORMAT(expr1,format)
* STR\_TO\_DATE(expr1, format)

常用的日期格式YYYY-MM-DD HH:MM:SS对应的format为%Y-%m-%d %H:%i:%S

### 常用的类型转换函数：

* CAST(expr AS type)
* CONVERT(expr,type)
* CONVERT(expr USING transcoding\_name)

# 附录1-MySQL的缺陷

* MySQL 每个查询只能用到一个CPU（新版本5.7中，引擎层可以利用多CPU）
* MySQL 没有执行计划缓存（存储过程等并不会提高性能）
* MySQL 默认情况下，随着连接数的升高，性能会急剧下降（连接数的压力测试，thread pool可解决）
* MySQL join 算法只有nested loop 算法，没有hash join（MariDB 目前支持 hash join）
* MySQL 优化器采用RBO（5.7以后逐渐引进CBO）
* MySQL使用VIEW时，采用中间表的方式执行，会屏蔽原表中本身的索引

# 附录2-数据库特征

* 资源型应用（锁表现）
* 不易扩展型应用（最终数据一致性）
* IO密集型应用（数据存取）
* 资源消耗型应用（IO、MEM、CPU 、NET、DISK）
* 主要用来实现数据持久化存储（不适合做运算）

# 附录3-字符集

|  |  |  |  |
| --- | --- | --- | --- |
| **字符集名称** | **字节占用** | **字符集兼容性** | **Unicode字符支持** |
| UCS2 | 每字符2字节 |  | 所有Unicode 3.0字符 |
| UTF16 | 每字符2字节，或4字节。 | 与UCS2兼容 | 所有Unicode 5.0和Unicode 6.0字符，包括扩展字符。 |
| UTF16LE | 与UTF16相同，只是字节顺序相反。 |  | 所有Unicode 5.0和Unicode 6.0字符，包括扩展字符。 |
| UTF8 | 每字符1到3字节。 |  | 所有Unicode 3.0字符 |
| UTF8MB4 | 每字符1到4字节。 | 与UTF8兼容 | 所有Unicode 5.0和Unicode 6.0字符，包括扩展字符。 |

# 附录4-MySQL保留字

|  |  |  |
| --- | --- | --- |
| **ACCESSIBLE** (R) | **ACTION** | **ADD** (R) |
| **AFTER** | **AGAINST** | **AGGREGATE** |
| **ALGORITHM** | **ALL** (R) | **ALTER** (R) |
| **ANALYSE**[[a]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573266064) | **ANALYZE** (R) | **AND** (R) |
| **ANY** | **AS** (R) | **ASC** (R) |
| **ASCII** | **ASENSITIVE** (R) | **AT** |
| **AUTHORS**[[b]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573251904) | **AUTOEXTEND\_SIZE** | **AUTO\_INCREMENT** |
| **AVG** | **AVG\_ROW\_LENGTH** | **BACKUP** |
| **BEFORE** (R) | **BEGIN** | **BETWEEN** (R) |
| **BIGINT** (R) | **BINARY** (R) | **BINLOG** |
| **BIT** | **BLOB** (R) | **BLOCK** |
| **BOOL** | **BOOLEAN** | **BOTH** (R) |
| **BTREE** | **BY** (R) | **BYTE** |
| **CACHE** | **CALL** (R) | **CASCADE** (R) |
| **CASCADED** | **CASE** (R) | **CATALOG\_NAME** |
| **CHAIN** | **CHANGE** (R) | **CHANGED** |
| **CHAR** (R) | **CHARACTER** (R) | **CHARSET** |
| **CHECK** (R) | **CHECKSUM** | **CIPHER** |
| **CLASS\_ORIGIN** | **CLIENT** | **CLOSE** |
| **COALESCE** | **CODE** | **COLLATE** (R) |
| **COLLATION** | **COLUMN** (R) | **COLUMNS** |
| **COLUMN\_FORMAT**[[c]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573186208) | **COLUMN\_NAME** | **COMMENT** |
| **COMMIT** | **COMMITTED** | **COMPACT** |
| **COMPLETION** | **COMPRESSED** | **CONCURRENT** |
| **CONDITION** (R) | **CONNECTION** | **CONSISTENT** |
| **CONSTRAINT** (R) | **CONSTRAINT\_CATALOG** | **CONSTRAINT\_NAME** |
| **CONSTRAINT\_SCHEMA** | **CONTAINS** | **CONTEXT** |
| **CONTINUE** (R) | **CONTRIBUTORS**[[d]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573158576) | **CONVERT** (R) |
| **CPU** | **CREATE** (R) | **CROSS** (R) |
| **CUBE** | **CURRENT**[[e]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573148864) | **CURRENT\_DATE** (R) |
| **CURRENT\_TIME** (R) | **CURRENT\_TIMESTAMP** (R) | **CURRENT\_USER** (R) |
| **CURSOR** (R) | **CURSOR\_NAME** | **DATA** |
| **DATABASE** (R) | **DATABASES** (R) | **DATAFILE** |
| **DATE** | **DATETIME** | **DAY** |
| **DAY\_HOUR** (R) | **DAY\_MICROSECOND** (R) | **DAY\_MINUTE** (R) |
| **DAY\_SECOND** (R) | **DEALLOCATE** | **DEC** (R) |
| **DECIMAL** (R) | **DECLARE** (R) | **DEFAULT** (R) |
| **DEFAULT\_AUTH**[[f]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573113632) | **DEFINER** | **DELAYED** (R) |
| **DELAY\_KEY\_WRITE** | **DELETE** (R) | **DESC** (R) |
| **DESCRIBE** (R) | **DES\_KEY\_FILE** | **DETERMINISTIC** (R) |
| **DIAGNOSTICS**[[g]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573099408) | **DIRECTORY** | **DISABLE** |
| **DISCARD** | **DISK** | **DISTINCT** (R) |
| **DISTINCTROW** (R) | **DIV** (R) | **DO** |
| **DOUBLE** (R) | **DROP** (R) | **DUAL** (R) |
| **DUMPFILE** | **DUPLICATE** | **DYNAMIC** |
| **EACH** (R) | **ELSE** (R) | **ELSEIF** (R) |
| **ENABLE** | **ENCLOSED** (R) | **END** |
| **ENDS** | **ENGINE** | **ENGINES** |
| **ENUM** | **ERROR** | **ERRORS** |
| **ESCAPE** | **ESCAPED** (R) | **EVENT** |
| **EVENTS** | **EVERY** | **EXCHANGE** |
| **EXECUTE** | **EXISTS** (R) | **EXIT** (R) |
| **EXPANSION** | **EXPIRE**[[h]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573045360) | **EXPLAIN** (R) |
| **EXPORT**[[i]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573041360) | **EXTENDED** | **EXTENT\_SIZE** |
| **FALSE** (R) | **FAST** | **FAULTS** |
| **FETCH** (R) | **FIELDS** | **FILE** |
| **FIRST** | **FIXED** | **FLOAT** (R) |
| **FLOAT4** (R) | **FLOAT8** (R) | **FLUSH** |
| **FOR** (R) | **FORCE** (R) | **FOREIGN** (R) |
| **FORMAT**[[j]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573014256) | **FOUND** | **FROM** (R) |
| **FULL** | **FULLTEXT** (R) | **FUNCTION** |
| **GENERAL**[[k]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722573004656) | **GEOMETRY** | **GEOMETRYCOLLECTION** |
| **GET**[[l]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572999552) (R) | **GET\_FORMAT** | **GLOBAL** |
| **GRANT** (R) | **GRANTS** | **GROUP** (R) |
| **HANDLER** | **HASH** | **HAVING** (R) |
| **HELP** | **HIGH\_PRIORITY** (R) | **HOST** |
| **HOSTS** | **HOUR** | **HOUR\_MICROSECOND** (R) |
| **HOUR\_MINUTE** (R) | **HOUR\_SECOND** (R) | **IDENTIFIED** |
| **IF** (R) | **IGNORE** (R) | **IGNORE\_SERVER\_IDS**[[m]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572969568) |
| **IMPORT** | **IN** (R) | **INDEX** (R) |
| **INDEXES** | **INFILE** (R) | **INITIAL\_SIZE** |
| **INNER** (R) | **INOUT** (R) | **INSENSITIVE** (R) |
| **INSERT** (R) | **INSERT\_METHOD** | **INSTALL** |
| **INT** (R) | **INT1** (R) | **INT2** (R) |
| **INT3** (R) | **INT4** (R) | **INT8** (R) |
| **INTEGER** (R) | **INTERVAL** (R) | **INTO** (R) |
| **INVOKER** | **IO** | **IO\_AFTER\_GTIDS**[[n]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572932912) (R) |
| **IO\_BEFORE\_GTIDS**[[o]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572930208) (R) | **IO\_THREAD** | **IPC** |
| **IS** (R) | **ISOLATION** | **ISSUER** |
| **ITERATE** (R) | **JOIN** (R) | **KEY** (R) |
| **KEYS** (R) | **KEY\_BLOCK\_SIZE** | **KILL** (R) |
| **LANGUAGE** | **LAST** | **LEADING** (R) |
| **LEAVE** (R) | **LEAVES** | **LEFT** (R) |
| **LESS** | **LEVEL** | **LIKE** (R) |
| **LIMIT** (R) | **LINEAR** (R) | **LINES** (R) |
| **LINESTRING** | **LIST** | **LOAD** (R) |
| **LOCAL** | **LOCALTIME** (R) | **LOCALTIMESTAMP** (R) |
| **LOCK** (R) | **LOCKS** | **LOGFILE** |
| **LOGS** | **LONG** (R) | **LONGBLOB** (R) |
| **LONGTEXT** (R) | **LOOP** (R) | **LOW\_PRIORITY** (R) |
| **MASTER** | **MASTER\_AUTO\_POSITION**[[p]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572870368) | **MASTER\_BIND**[[q]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572868080) (R) |
| **MASTER\_CONNECT\_RETRY** | **MASTER\_DELAY** | **MASTER\_HEARTBEAT\_PERIOD**[[r]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572862784) |
| **MASTER\_HOST** | **MASTER\_LOG\_FILE** | **MASTER\_LOG\_POS** |
| **MASTER\_PASSWORD** | **MASTER\_PORT** | **MASTER\_RETRY\_COUNT**[[s]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572853504) |
| **MASTER\_SERVER\_ID** | **MASTER\_SSL** | **MASTER\_SSL\_CA** |
| **MASTER\_SSL\_CAPATH** | **MASTER\_SSL\_CERT** | **MASTER\_SSL\_CIPHER** |
| **MASTER\_SSL\_CRL**[[t]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572842672) | **MASTER\_SSL\_CRLPATH**[[u]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572840384) | **MASTER\_SSL\_KEY** |
| **MASTER\_SSL\_VERIFY\_SERVER\_CERT**(R) | **MASTER\_USER** | **MATCH** (R) |
| **MAXVALUE** (R) | **MAX\_CONNECTIONS\_PER\_HOUR** | **MAX\_QUERIES\_PER\_HOUR** |
| **MAX\_ROWS** | **MAX\_SIZE** | **MAX\_UPDATES\_PER\_HOUR** |
| **MAX\_USER\_CONNECTIONS** | **MEDIUM** | **MEDIUMBLOB** (R) |
| **MEDIUMINT** (R) | **MEDIUMTEXT** (R) | **MEMORY** |
| **MERGE** | **MESSAGE\_TEXT** | **MICROSECOND** |
| **MIDDLEINT** (R) | **MIGRATE** | **MINUTE** |
| **MINUTE\_MICROSECOND** (R) | **MINUTE\_SECOND** (R) | **MIN\_ROWS** |
| **MOD** (R) | **MODE** | **MODIFIES** (R) |
| **MODIFY** | **MONTH** | **MULTILINESTRING** |
| **MULTIPOINT** | **MULTIPOLYGON** | **MUTEX** |
| **MYSQL\_ERRNO** | **NAME** | **NAMES** |
| **NATIONAL** | **NATURAL** (R) | **NCHAR** |
| **NDB** | **NDBCLUSTER** | **NEW** |
| **NEXT** | **NO** | **NODEGROUP** |
| **NONE** | **NOT** (R) | **NO\_WAIT** |
| **NO\_WRITE\_TO\_BINLOG** (R) | **NULL** (R) | **NUMBER**[[v]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572765376) |
| **NUMERIC** (R) | **NVARCHAR** | **OFFSET** |
| **OLD\_PASSWORD** | **ON** (R) | **ONE** |
| **ONE\_SHOT**[[w]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572754240) | **ONLY**[[x]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572751936) | **OPEN** |
| **OPTIMIZE** (R) | **OPTION** (R) | **OPTIONALLY** (R) |
| **OPTIONS** | **OR** (R) | **ORDER** (R) |
| **OUT** (R) | **OUTER** (R) | **OUTFILE** (R) |
| **OWNER** | **PACK\_KEYS** | **PAGE** |
| **PARSER** | **PARTIAL** | **PARTITION**[[y]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572727712) (R) |
| **PARTITIONING** | **PARTITIONS** | **PASSWORD** |
| **PHASE** | **PLUGIN** | **PLUGINS** |
| **PLUGIN\_DIR**[[z]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572716720) | **POINT** | **POLYGON** |
| **PORT** | **PRECISION** (R) | **PREPARE** |
| **PRESERVE** | **PREV** | **PRIMARY** (R) |
| **PRIVILEGES** | **PROCEDURE** (R) | **PROCESSLIST** |
| **PROFILE** | **PROFILES** | **PROXY**[[aa]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572696080) |
| **PURGE** (R) | **QUARTER** | **QUERY** |
| **QUICK** | **RANGE** (R) | **READ** (R) |
| **READS** (R) | **READ\_ONLY** | **READ\_WRITE** (R) |
| **REAL** (R) | **REBUILD** | **RECOVER** |
| **REDOFILE** | **REDO\_BUFFER\_SIZE** | **REDUNDANT** |
| **REFERENCES** (R) | **REGEXP** (R) | **RELAY** |
| **RELAYLOG** | **RELAY\_LOG\_FILE** | **RELAY\_LOG\_POS** |
| **RELAY\_THREAD** | **RELEASE** (R) | **RELOAD** |
| **REMOVE** | **RENAME** (R) | **REORGANIZE** |
| **REPAIR** | **REPEAT** (R) | **REPEATABLE** |
| **REPLACE** (R) | **REPLICATION** | **REQUIRE** (R) |
| **RESET** | **RESIGNAL** (R) | **RESTORE** |
| **RESTRICT** (R) | **RESUME** | **RETURN** (R) |
| **RETURNED\_SQLSTATE**[[ab]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572637104) | **RETURNS** | **REVERSE** |
| **REVOKE** (R) | **RIGHT** (R) | **RLIKE** (R) |
| **ROLLBACK** | **ROLLUP** | **ROUTINE** |
| **ROW** | **ROWS** | **ROW\_COUNT**[[ac]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572620640) |
| **ROW\_FORMAT** | **RTREE** | **SAVEPOINT** |
| **SCHEDULE** | **SCHEMA** (R) | **SCHEMAS** (R) |
| **SCHEMA\_NAME** | **SECOND** | **SECOND\_MICROSECOND** (R) |
| **SECURITY** | **SELECT** (R) | **SENSITIVE** (R) |
| **SEPARATOR** (R) | **SERIAL** | **SERIALIZABLE** |
| **SERVER** | **SESSION** | **SET** (R) |
| **SHARE** | **SHOW** (R) | **SHUTDOWN** |
| **SIGNAL** (R) | **SIGNED** | **SIMPLE** |
| **SLAVE** | **SLOW**[[ad]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572582240) | **SMALLINT** (R) |
| **SNAPSHOT** | **SOCKET** | **SOME** |
| **SONAME** | **SOUNDS** | **SOURCE** |
| **SPATIAL** (R) | **SPECIFIC** (R) | **SQL** (R) |
| **SQLEXCEPTION** (R) | **SQLSTATE** (R) | **SQLWARNING** (R) |
| **SQL\_AFTER\_GTIDS**[[ae]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572560736) | **SQL\_AFTER\_MTS\_GAPS**[[af]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572558416) | **SQL\_BEFORE\_GTIDS**[[ag]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572556128) |
| **SQL\_BIG\_RESULT** (R) | **SQL\_BUFFER\_RESULT** | **SQL\_CACHE** |
| **SQL\_CALC\_FOUND\_ROWS** (R) | **SQL\_NO\_CACHE** | **SQL\_SMALL\_RESULT** (R) |
| **SQL\_THREAD** | **SQL\_TSI\_DAY** | **SQL\_TSI\_HOUR** |
| **SQL\_TSI\_MINUTE** | **SQL\_TSI\_MONTH** | **SQL\_TSI\_QUARTER** |
| **SQL\_TSI\_SECOND** | **SQL\_TSI\_WEEK** | **SQL\_TSI\_YEAR** |
| **SSL** (R) | **START** | **STARTING** (R) |
| **STARTS** | **STATS\_AUTO\_RECALC**[[ah]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572526624) | **STATS\_PERSISTENT**[[ai]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572524336) |
| **STATS\_SAMPLE\_PAGES**[[aj]](file:///D:\\%E5%AE%98%E6%96%B9%E6%96%87%E6%A1%A3\\refman-5.6-en.html-chapter\\language-structure.html" \l "ftn.idm139722572521792) | **STATUS** | **STOP** |
| **STORAGE** | **STRAIGHT\_JOIN** (R) | **STRING** |
| **SUBCLASS\_ORIGIN** | **SUBJECT** | **SUBPARTITION** |
| **SUBPARTITIONS** | **SUPER** | **SUSPEND** |
| **SWAPS** | **SWITCHES** | **TABLE** (R) |
| **TABLES** | **TABLESPACE** | **TABLE\_CHECKSUM** |
| **TABLE\_NAME** | **TEMPORARY** | **TEMPTABLE** |
| **TERMINATED** (R) | **TEXT** | **THAN** |
| **THEN** (R) | **TIME** | **TIMESTAMP** |
| **TIMESTAMPADD** | **TIMESTAMPDIFF** | **TINYBLOB** (R) |
| **TINYINT** (R) | **TINYTEXT** (R) | **TO** (R) |
| **TRAILING** (R) | **TRANSACTION** | **TRIGGER** (R) |
| **TRIGGERS** | **TRUE** (R) | **TRUNCATE** |
| **TYPE** | **TYPES** | **UNCOMMITTED** |
| **UNDEFINED** | **UNDO** (R) | **UNDOFILE** |
| **UNDO\_BUFFER\_SIZE** | **UNICODE** | **UNINSTALL** |
| **UNION** (R) | **UNIQUE** (R) | **UNKNOWN** |
| **UNLOCK** (R) | **UNSIGNED** (R) | **UNTIL** |
| **UPDATE** (R) | **UPGRADE** | **USAGE** (R) |
| **USE** (R) | **USER** | **USER\_RESOURCES** |
| **USE\_FRM** | **USING** (R) | **UTC\_DATE** (R) |
| **UTC\_TIME** (R) | **UTC\_TIMESTAMP** (R) | **VALUE** |
| **VALUES** (R) | **VARBINARY** (R) | **VARCHAR** (R) |
| **VARCHARACTER** (R) | **VARIABLES** | **VARYING** (R) |
| **VIEW** | **WAIT** | **WARNINGS** |
| **WEEK** | **WEIGHT\_STRING** | **WHEN** (R) |
| **WHERE** (R) | **WHILE** (R) | **WITH** (R) |
| **WORK** | **WRAPPER** | **WRITE** (R) |
| **X509** | **XA** | **XML** |
| **XOR** (R) | **YEAR** | **YEAR\_MONTH** (R) |
| **ZEROFILL** (R) |  |  |