**三度数据库开发规范**

Version:1.0

Author:liuzhao

目录

[**一、 需求概述** 3](#_Toc504553263)

[二、规范细则以DB对象为维度 3](#_Toc504553264)

[1. 逻辑库对象 3](#_Toc504553265)

[1.1 命名规范 3](#_Toc504553266)

[1.2 创建注意 3](#_Toc504553267)

[2. 表对象 4](#_Toc504553268)

[2.1 命名规范 4](#_Toc504553269)

[2.2 创建注意事项 4](#_Toc504553270)

[3. 视图对象 6](#_Toc504553271)

[3.1 命名规范 6](#_Toc504553272)

[3.2 创建注意 6](#_Toc504553273)

[4. 索引对象 7](#_Toc504553274)

[4.1 命名规范 7](#_Toc504553275)

[4.2 创建注意 7](#_Toc504553276)

[5．外键对象 8](#_Toc504553277)

[5.1命名规范 8](#_Toc504553278)

[6．存储过程 9](#_Toc504553279)

[6.1 命名规范 9](#_Toc504553280)

[6.2 创建注意 9](#_Toc504553281)

[7．函数对象 10](#_Toc504553282)

[7.1 命名规范 10](#_Toc504553283)

[7.2 创建注意 10](#_Toc504553284)

[8．触发器对象 10](#_Toc504553285)

[8.1 命名规范 10](#_Toc504553286)

[8.2 创建注意 11](#_Toc504553287)

[9．事件对象 11](#_Toc504553288)

[9.1 命名规范 11](#_Toc504553289)

[9.2 创建注意 11](#_Toc504553290)

[三、其它DB开发注意事项 12](#_Toc504553291)

[四、 DB开发硬性规定 12](#_Toc504553292)

1. **需求概述**

前期项目压力大，数据开发比较随意，自由度比较高。现在处在相对稳定的时间，对数据库开发做出一些规范约束。包含对对象级别的命名规范，书写规范，引导使用最合适的数据类型，更高效的设计方法，以及在数据库设计中应该避免的一些设计。

# 二、规范细则以DB对象为维度

## 1. 逻辑库对象

### 1.1 命名规范

逻辑库名—— 项目缩写

以语义清晰、简短为宜，小写，下划线分隔语义

### 1.2 创建注意

A.新项目比较独立或旧项目垂直拆分时，需创建逻辑库

B.不能指定存储引擎，只能使用默认引擎（开发人员请了解当前引擎）

C.库级字符集不显式指定，统一使用默认字符集（开发人员请了解当前字符集）

D.逻辑DB的创建需由DBA来完成

E.逻辑DB的帐号信息创建需由DBA来完成

F.逻辑DB的备份策略由DBA来完成

## 2. 表对象

### 2.1 命名规范

表名——项目缩写为前缀 + 表义

语义清晰、简短为宜，小写，下划线分隔语义

如,用户中心请以sys\_做为前缀

设计方案中的表请以design\_做为前缀

公共项目中的表请以base\_做为前缀

报表中的表请以rpt\_做为前缀

产品中的表请以product\_做为前缀等

列名——语义清晰、简短为宜，小写，下划线分隔语义，

务必保证多表中同一语义的字段名称是一样

尽量避免使用DB关键字做列名如DESC，KEY，VALUE，STATUS等

### 2.2 创建注意事项

1. 表要统一使用powerdesiner 来创建逻辑视图，做好物理视图的维护
2. 表要显式指定存储引擎，缺省为innodb
3. 表一定要有主键，可简单使用自增列来做为主键
4. 表要显式指定表级字符集及比对规则，指定为CHARSET=utf8 COLLATE utf8\_bin
5. 字段建表字段时应考虑原有表中该语义字段的类型、长度、字符集等信息一致，即不同表中相同语义的字段应保持完全一致
6. 字段使用int，long型时不要指定长度，除非有业务上的要求，因为并不能节省存储空间
7. 字段小范围数值型状态字段请使用tinyint,字符型状态请用char(1)
8. 字段不需要区分大小写的字段，请显式指定字符集排序规则COLLATE utf8\_general\_ci
9. 字段能用整型，禁止使用VARCHAR字符型，节省存储空间，提高效率。避免使用TEXT，禁止BLOB类型
10. 字段非空字段必须要有缺省值，字段最好都要有缺省值，统一规定成缺省成空字串’’，数值型缺省为0
11. 表中一定要指定CREATE\_TIME和UPDATE\_TIME时间字段,优先timestamp类型，update\_time要指定 ON UPDATE CURRENT\_TIMESTAMP
12. 若字段在逻辑上是唯一的，请加上唯一约束，防止重复数据的出现
13. 字段表注释，字段注释必须完整，状态字段的注释禁止随意改动，若有改动，请做好更新维护
14. 自增字段不做业务逻辑使用，禁止同时命名字段conf\_id，conf\_code;

建议conf\_id 改成id 做自增主键，不参与业务逻辑。业务中只使用conf\_code;

## 3. 视图对象

### 3.1 命名规范

视图名——v\_ + 视图义

### 3.2 创建注意

1. MYSQL 视图创建可显式指定MERGE算法和TEMPTABLE算法

TEMPTABLE 一定会产生严重的性能问题，和子查询导致性能差的原因一样，会先创建一个临时表，且对视图EXPLAIN 的时候亦会先产生临时结果表

MERGE 基本上不会有性能问题，但会有使用限制

只要视图定义中使用了以下SQL构造块就无法使用MERGE算法：

a. 聚集函数

b. DISTINCT

c. GROUP BY

d. HAVING

e. 集合操作（在MySQL中只有UNION, UNION ALL，没有EXCEPT和INTERSECT）

f. 子查询

定义SQL中若含有以上用法就不能使用，只适用简单的SQL定义

B. 创建视图时请显式指定WITH CASCADED CHECK OPTION参数，这种方式比较严格，能保证数据的安全性

C. MYSQL视图可实现视图更新，当然只有在视图和行列与基础表的行列之间存在一一对应关系时才能更新。禁止使用视图来更新数据！

D.开发时慎用视图定义，若已使用请关注是否有性能问题

E. VIEW的创建需要DBA审核

## 4. 索引对象

### 4.1 命名规范

索引名——idx\_ + 第一个索引字段

惟一性索引名——uqidx\_ + 第一个索引字段

语义清晰、简短为宜，小写，下划线分隔语义

Eg:Alter table tb\_name add key idx\_user\_id(user\_id，name);

Alter table tb\_name add unique key uqidx\_user\_id(user\_id，name);

### 4.2 创建注意

1. 项目开发过程中，根椐常见的SQL语句，在表上创建好索引，避免裸表的存在;
2. 索引有前导性，组合索引更划算，避免出现拥有同一前导字段的多个索引

EG：Alter table tb\_name add key idx\_user\_id(user\_id，name);

Alter table tb\_name add key idx\_user\_id2(user\_id);

索引idx\_user\_id2没必要再创建，若必须创建，请在索引名称后加上数据序号

C. 索引是把双刃剑，过多的索引会影响写效率，占用大量的存储空间，尤其是大表

D. 大表的索引创建会是长时间的，对大表的索引修改要慎重

E. 惟一索引不能保证NULL值是惟一的，如unidx\_id(id,type); 可存在多条1,null;请设定字段缺省值，来避免为NULL

F. 外键列一定要有索引，否则会容易引起死锁，禁止随意删除外键列上的索引

G. 索引都是有序的，可加以利用，进行性能优化

## 5．外键对象

### 5.1命名规范

外键名——fk\_ + 列名

ALTER TABLE tb\_a add constraint fk\_user\_id foreign key(user\_id) REFERENCE tb\_b(user\_id);

5.2 创建注意

1. 两表的外键引用字段，数据类型长度字符集规则必须一致，否则创建失败
2. 当创建外键时，DB会在外键列上自动加上索引，避免产生死锁，禁止随意修改外键列上的索引
3. 慎重使用MYSQL外键，由于表关系复杂，会导致外键的运维维护复杂化，可人为控制表数据的完整性，只在核心关系表中小范围使用,其它非核心关系表禁止使用外键，请自己来控制数据的完整性！

## 6．存储过程

### 6.1 命名规范

存储过程名——pro\_ + 语义

语义清晰、简短为宜，小写，下划线分隔语义

### 6.2 创建注意

1. 创建过程时请显式指明definer=root@127.0.0.1 开发人员可不写，DBA执行时请添加
2. 存储过程一定要有输出值，类型为varchar(255)，用来标记该过程是否成功执行
3. 存储过程中请使用好异常定义，如循环的跳出判断，SQL异常的回滚退出
4. **存储过程中避免使用游标，游标会引发性能问题与批量化处理相悖，请尝试使用临时表转化成批量化处理的方式**

**E. 存储过程可对少量结果集使用临时表，来提升开发效率和执行效率，临时表可添加索引，使用前和后请drop temporary table if exists tb\_tmp;来释放内存资源**

F. 存储过程只能降低网络IO开销，和把计算放在DB服务端进行，请不要滥用。建议在程序端不易实现，或者实时性要求不高，可书写存储过程

G. 存储过程中注意事务的封装，避免大事务，可拆成小事务组去执行，注意DDL语句会潜在的添加commit语句

H. 存储过程不易滥用

## 7．函数对象

### 7.1 命名规范

函数名——fn\_+语义

语义清晰、简短为宜，小写，下划线分隔语义

### 7.2 创建注意

1. 函数定义需尽量简单，功能尽量单一，复杂的函数定义会有效率隐患，间接导致从库的不稳定
2. 禁止使用功能定义复杂的函数，请使用过程替换！
3. 函数的创建需要DBA审核

## 8．触发器对象

### 8.1 命名规范

触发器名——tr\_ + 语义

语义清晰、简短为宜，小写，下划线分隔语义

### 8.2 创建注意

1. 可用来做数据的同步
2. 创建时候的一些限制

如果你在触发器里面对刚刚插入的数据进行了 insert/update, 会造成循环的调用.如：create trigger test before update on test for each row update test set NEW.updateTime = NOW() where id=NEW.ID; END

应该使用set： create trigger test before update on test for each row set NEW.updateTime = NOW(); END

C. 注意若相关表含有非事务性表，会影响触发器的使用效果

D. 不可过度依赖触发器，少用

F. 触发器的创建需要DBA审核

## 9．事件对象

### 9.1 命名规范

事件名——event\_ + 语义

语义清晰、简短为宜，小写，下划线分隔语义

### 9.2 创建注意

1. 业务中的一些定时任务，实时性要求不高的数据更新
2. 要将定时任务的执行日志，写入统一的事件日志表
3. 事件的创建需要DBA审核

# 三、其它DB开发注意事项

1. 以上DB开发细则，起指导规范作用，务必紧靠规则进行开发
2. DB访问帐号将会以项目为维度来创建，目前是以DB来维度，比较粗
3. 项目上测试生产环境必须提供项目的物理视图PDM文件和数据字典文件，将会统一更新至SVN处（地址待定）
4. 项目在测试阶段，会搜集该项目SQL语句，会初步筛选出慢查询SQL，进行统一优化后再上生产DB
5. 页面数据，只允许单表查询，避免用户使用大量的join操作，异步更新冗余
6. 禁止使用多表关联，大量子查询。
7. 禁止在项目中出现跨库操作的SQL！

方案一：若有业务需求，请联接多个数据源，切换到该数据源中进行查询，取出数据，再切回本库做进一步的查询

方案二：交由DBA做数据同步至同一个库中

推荐方案一，在不具备方案一的执行条下，再考虑方案二！

# DB开发硬性规定

**1.业务DB中禁 止使用联合查询。**

[核心表](../数据视图及字典/广联数据研发部门_MYSQL核心表数据字典.xlsx)上只允许出现单表查询的简单SQL

替代方案：

场景一：

1组入参——1条记录——1条记录

取表A所需展示字段，及表B入参

SELECT A\_NAME,B\_ID into @a\_name, @b\_id FROM TAB\_A WHERE A\_ID=? LIMIT 1;

传入表B入参取表B所需字段

SELECT B\_NAME FROM TAB\_B WHERE B\_ID=@b\_id;

自行组装展示

SELECT @a\_name，@a\_name;

场景二：

1 组入参——n条记录——n条记录

取表A所需展示字段，及表B入参

SELECT A\_NAME,B\_ID into @a\_name\_list,@b\_id\_list FROM TAB\_A WHERE A\_ID=?

传入表B入参取表B所需字段

SELECT B\_NAME FROM TAB\_B WHERE B\_ID in (@b\_id)

自行组装展示

SELECT @a\_name\_list，@a\_name\_list;

**以上伪SQL仅作为方法参考，请自行保存中间值，及自行组装展示结果**

**2.业务DB中禁止出现嵌套查询。**

单表查询中请使用 in关键字代替eg:in (1,2,3,4,…..)

**3.业务DB中禁 止使用存储过程，业务逻辑请上移至应用中处理或迁移至从库计算，减少DB压力**

**4.新项目请维护好PDM文件，并生成EXCEL数据字典，见**[**模版**](广联数据研发部门_MYSQL数据字典模版.xlsx)

**5.**[**核心表**](../数据视图及字典/广联数据研发部门_MYSQL核心表数据字典.xlsx)**结构的变更需评审通过，方能更改**

**6.新项目上线前，若发现无PDM，存在关联查询SQL，将会打回禁止上线！**

7.实行场景：新需求新项目中开始实行，会重点检查[核心表](../数据视图及字典/广联数据研发部门_MYSQL核心表数据字典.xlsx)的相关SQL

8. **业务表内禁止使用mysiam引擎，统一规范使用innodb引擎**

9. **临时表在创建删除时均要指定temporary关键字**