**SQL语言编码规范**

**编制单位：西部大区一部**

**编 制 人：王瀚贤**

**审 核：**

**批 准：**

**用友汽车信息科技（上海）股份有限公司**

**Yonyou Auto Information Technology (Shanghai) Co., Ltd.‘**

**文档更新记录**

| **编号** | **章节** | **变更类型** | **修订内容简述** | **修订日期** | **修订前版本** | **修订后版本** | **修订人** |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| 1 | 所有章节 | C |  | 2017-06-19 | / | 1.0 | 王瀚贤 |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

\*变更类型：C——创建，A——增加，M——修改，D——删除

文档发送范围：

本部门员工。

目录

[1 模块命名 4](#_Toc444248282)

[2 数据库设计命名规则 4](#_Toc444248283)

[3 数据类型规范 5](#_Toc444248284)

[4 存储过程 6](#_Toc444248285)

[5 触发器 6](#_Toc444248286)

[6 视图 6](#_Toc444248287)

[7 索引 6](#_Toc444248288)

[8 表及外键 7](#_Toc444248289)

[9 SQL语句规范 7](#_Toc444248290)

[10 表别名规范 7](#_Toc444248291)

[11 函数规范 7](#_Toc444248292)

[12 建库规范 7](#_Toc444248293)

[13 SQL注释风格 8](#_Toc444248294)

[14 书写优化性能建议 8](#_Toc444248295)

[15 其他经验性规则 9](#_Toc444248296)

# 模块命名

规范模块命名，在后面的表命名等处需要用到。

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 模块名称 | 系统规则名 | | 模块名称 | | 系统规则名 | | | 模块名称 | | 系统规则名 | | |
| **基准前缀** | | | | | | | | | | | | |
| 平台基础数据 | TC | | 实体数据 | | TM | | | 业务数据 | | TT | | |
| 关系数据 | TR | | 工作流管理 | | TA | | | 临时表 | | TMP | | |
|  |  |  | |  | |  |  | |  | |  |  | |

# 数据库设计命名规则

* 所有表名、字段名统一小写，多个单词或前后缀之间使用‘\_’连接。
* 所有表、视图、触发器、索引、函数、约束、主外键必须指定名称，规则如下：

|  |  |  |  |
| --- | --- | --- | --- |
| **数据库对象** | **规则** | **最大长度** | **例子** |
| 表 | 基准前缀\_英文名称 | 20 | Tm\_naterial |
| 字段名 | 英文名称 | 20 | code, name,shortname |
| 视图 | v\_模块命名\_英文名称 | 20 | v\_cbo\_material |
| 触发器 | t\_表名\_名称 | 20 | t\_cbo\_material \_getmaterial |
| 函数 | f\_英文名称 | 20 | f\_fun() |
| Check | ck\_表名\_字段名 | 20 | ck\_cbo\_material\_code |
| 主键 | pk\_表名 | 20 | pk\_cbo\_material |
| 外键 | fk\_表名\_字段名 | 20 | fk\_material\_code |
| 索引 | i\_表名\_字段名 | 20 | i\_cbo\_material \_code |
| 表空间 | ts\_英文名称 | 20 | ts\_sys |
| 数据类型 | ty\_英文名称单数形式 | 20 | ty\_dollas |

（\*）注：由于数据库对象之间的命名规则与最大长度限制可能造成命名存在冲突。如存在，可以通过缩减表名或自编序号。

* 表空间名约定

|  |  |
| --- | --- |
| **表空间** | **使用范围** |
| ts\_sys | 系统表空间 |
| ts\_data | 标准操作表 |
| ts\_view | 标准视图操作 |
| ts\_index | 标准操作表索引 |
| ts\_rbs | 标准操作回滚段 |
| ts\_temp | 标准操作临时段 |
| ts\_trig | 标准触发器操作 |

* 所有的外键、约束、索引、函数、触发器、存储过程名不允许重复。
* 业务中主子表的命名规则：模块命名\_英文名称, 模块命名名\_英文名称line。
* 业务中主子子表的命名规则：模块命名\_英文名称, 模块命名\_英文名称line，模块命名\_英文名称subline。
* 字段的命名规则，英文名称建议使用以下字母后缀。

|  |  |
| --- | --- |
| **类型** | **规则(字母后缀)** |
| 编码类 | code |
| 名称类 | name |
| 简称 | shortname |
| 标志类 | flag |
| 级次类 | lev |
| 金额类 | mny |
| 系统标识类 | id |
| 自定义项类 | def |
| 单价类 | price |
| 税类 | tax |
| 数量类 | qty |
| 比率类 | rate |
| 日期类 | date |
| 枚举 | enum |

# 数据类型规范

* 可用数据类型

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **类型** | **范围** | **Sqlserver** | **Oracle** | **Db2** | **Sybase** |
| 字符型 | 长度固定 | Char | Varchar2 | Char | Char |
| 可变字符型 | 长度不固定 | Varchar | Varchar2 | Varchar | Varchar |
| 16位整型 | -32,768~32,767 | Smallint | Smallint | Smallint | Smallint |
| 32位整型 | -2,147,483,648 ~2,147,483,647 | Int | Int | integer | Int |
| 精确数值型 | pppppppppppppp.ss | Decimal(p,s) | Number（p,s） | Decimal(p,s) | Decimal(p,s) |
| 近似数值型 | -1.79E+308 ~1.79E+308 | Float | Float | Float | Float |
| 日期型 | yyyy-mm-dd | Char(10) | Varchar2(10) | Char(10) | Char(10) |
| 日期时间型 | yyyy-mm-dd hh\_mm\_ss | Char(19) | Varchar2(19) | Char(19) | Char(19) |
| 时间戳 | System date and time | date | datetime | date | datetime |
| 布尔型 | 0(false) or 1(true) | Smallint | Smallint | Smallint | Smallint |

* 禁止用SQL 保留字来命名表、视图、字段、索引。数据库的关键字。
* 不要设立诸如是否使用的标志字段，统一由数据库删除规则来确定。
* 标志字段要求提供缺省值。
* 数字类必须指定缺省值。
* 字符型字段的确定。长度固定用char，不固定用varchar。
* 系统序列号必须指定缺省值为0。
* 不要将Null 与 “空串”视为相同。在不同的数据库中对这两者的理解是不相同的。
* 在Oracle中空串与Null理解一样。如果碰到这种情况统一用Null。数据库中不允许出现空串。

# 存储过程

* 与数据库交互频繁的业务规则，可以考虑用存储过程实现。
* 每个存储过程代码不能超过200行。
* 对重要的计算应用注释说明其功能。

# 触发器

* 避免使用后触发。（Sqlserver、Sybase不支持前触发事件）。如必须使用，需项目经理审批。
* 系统序列号的存储统一通过触发器来控制。程序插入赋值时缺省为0。
* 不允许动态创建触发器。

# 视图

* 使用静态视图，不允许动态创建视图。
* 视图中尽量避免包含ORDER BY 语法，影响效率。

# 索引

* 为了保证索引具有提高效率的作用，估计记录数不大于50的表一律不允许建索引。
* 只用于DISTINCT或GROUP BY子句引用的列，不用建立索引。
* 对于数据值变化不大的列，比如只有yes和no，那么不要建索引，否则效率降低。
* 复合索引为多列，同复合主键一样将变化显著的列放到复合索引的首位。
* 一个复合索引可包含多达16个列(具体可能多数据库会不同)，索引太多则插入数据会变慢，当索引太多的时候，必须权衡查询和插入数据的响应时间是否满足需求。
* 不允许动态创建索引。

# 表及外键

* 不允许动态创建表、约束、外键。

# SQL语句规范

* SQL语句的所有表名、字段名统一小写。
* 系统保留字、内置函数名、SQL保留字大写。
* 连接符OR、IN、AND、运算符号、以及＝、<=、>=等前后加上一个空格；否则容易导致以下类似问题。在语句SELECT a–b FROM table 中， a，b均为变量。拼写该语句时，如果a=6， b= -3，则语句变为SELECT 6—3 FROM table。--变为Sql的注释，语句报错。
* 对SQL语句加上注释，说明算法、功能。
* 多表连接时，必须使用表的别名，并以此来引用列；别名必须是有意义的词。
* 供别的文件或函数调用的函数，绝不应使用全局变量交换数据。
* 禁止使用“SELECT \* FROM ”语法，必须标明字段名。
* 禁止使用“INSERT INTO table\_name VALUES(?,?,……)”语法，统一使用“INSERT INTO table\_name (col1,col2,……) VALUES(?,?,……)”。
* 如果在语句中有NOT IN（IN）操作，应考虑用NOT EXISTS（EXISTS）来重写。
* 避免显式或隐含的类型转换。例如在WHERE子句中numeric 型和int型的列的比较。
* 一行有多列，超过80个字符时，基于列对齐原则，采用下行缩进。
* where子句书写时，每个条件占一行，语句另起一行时，以保留字或者连接符开始。
* 每个事务处理必须有相应的COMMIT和ROLLBACK对应。

# 表别名规范

* 要求起别名用表名中有实际含义的单词的首字母。如果在同一个代码块中，单个字母有重复的，可采用首2个字母，以此类推。如：表名tb\_user，别名为：u。

# 函数规范

* 不允许动态创建函数。
* 只能使用通用SQL的函数。如要使用新的函数必须申报，审批后才能使用。

# 建库规范

* 建库脚本文件分类：
  + - 删除表类：包括模块所有表的DROP语句。
    - 表结构类：包括模块所有表的建表语句、主键定义以及CHECK约束。还包括数据类型以及规则定义。
    - 外键、视图、函数、索引类（简称外键类）：包括模块所有的外键、视图、索引、函数。包括DROP语句。
    - 触发器、存储过程类： 包括模块的触发器以及存储过程。包括DROP 语句。
    - 数据类：包括模块节点以及初始化数据；包括删除数据命令。包括DELETE语句。
* 书写格式：
  + - 注释用破折号“--”。
    - 遵循各个数据库SQL执行标准。例如，每条DDL以及DML语句结束符采用如下符号。

|  |  |  |  |
| --- | --- | --- | --- |
| **Db2** | **Oracle** | **Sqlserver** | **Sybase** |
| ; | DDL / DML; | go | go |

* + - Insert 语句使用单行插入模式。
    - 遵循模版格式。
    - 编写相应表格数据字典用于备份。

# SQL注释风格

* 注释单独成行、放在语句前面。
* 应对分支条件表达式加注释。
* 过长的函数实现，应将其语句按实现的功能分段加以概括性说明。
* 常量及变量注释时，应注释被保存值的含义（必须），合法取值的范围（可选）。

# 书写优化性能建议

* 避免嵌套连接。例如：A = B AND B = C AND C = D。
* WHERE条件中尽量减少使用常量比较，比如，不要用 1=1。
* 系统可能选择基于规则的优化器，所以将结果集返回数据量小的表作为驱动表（FROM后边最后一个表）。
* 大量的排序操作影响系统性能，所以尽量减少ORDER BY和GROUP BY排序操作，如必须使用排序操作，请遵循如下规则：
  + - 排序尽量建立在有索引的列上。
    - 如结果集不需唯一，使用UNION ALL代替UNION。
* 索引的使用。
  + - 尽量避免对索引列进行计算。如对索引列计算较多，请提请系统管理员建立函数索引。
    - 尽量注意比较值与索引列数据类型的一致性。
    - 对于复合索引，SQL语句必须使用主索引列。
    - 索引中，尽量避免使用NULL。
    - 对于索引的比较，尽量避免使用NOT=（!=）。
    - 查询列和排序列与索引列次序保持一致。
* 尽量避免相同语句由于书写格式的不同，而导致多次语法分析。
* 尽量使用共享的SQL语句。
* 查询的WHERE过滤原则，应使过滤记录数最多的条件放在最前面。
* 任何对列的操作都将导致表扫描，它包括数据库函数、计算表达式等等，查询时要尽可能将操作移至等号右边。
* IN、OR子句常会使用工作表，使索引失效；如果不产生大量重复值，可以考虑把子句拆开；拆开的子句中应该包含索引。

# 其他经验性规则

* 尽量少用嵌套查询。如必须，请用NOT EXIST代替NOT IN子句。
* 用多表连接代替EXISTS子句。
* 少用DISTINCT，用EXISTS代替 。
* 使用UNION ALL、MINUS、INTERSECT提高性能
* 使用ROWID提高检索速度。对SELECT得到的单行记录，需进行DELETE、UPDATE操作时，使用ROWID将会使效率大大提高。（需要数据库支持）。
* 使用优化线索机制进行访问路径控制。
* 不建议使用游标，禁止使用隐式游标。
* 允许使用临时表，但必须显式DROP。