区域卫生研发部研发规范

ORACLE

2018年06月

修改历史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本号** | **修改人** | **修改日期** | **修改内容** | **审批人** |
| 0.1 | 潘豪格 | 2018/06/07 | 创建 |  |
| 0.2 | 魏星星 | 2018/06/14 | 添加各章节详细内容 |  |
|  |  |  |  |  |

**目 录**

[1. 设计及开发规范 2](#_Toc521575833)

[1.1. 表空间 2](#_Toc521575834)

[1.2. 用户 4](#_Toc521575835)

[1.3. 表 5](#_Toc521575836)

[1.4. 字段 7](#_Toc521575837)

[1.5. 主键 9](#_Toc521575838)

[1.6. 外键 10](#_Toc521575839)

[1.7. 索引 10](#_Toc521575840)

[1.8. 序列 11](#_Toc521575841)

[1.9. 视图 13](#_Toc521575842)

[1.10. 函数 14](#_Toc521575843)

[1.11. 触发器 14](#_Toc521575844)

[1.12. 存储过程 15](#_Toc521575845)

[1.13. 包 16](#_Toc521575846)

[2. 使用规范 16](#_Toc521575847)

[2.1. Select 16](#_Toc521575848)

[2.2. where条件 17](#_Toc521575849)

# 设计及开发规范

在对系统进行数据库结构设计时，对于表空间、用户、表、字段、主键、索引、序列等，建议使用专业工具Sybase Power Designer进行设计开发，并将pdm源文件提交至源代码服务器中。

## 表空间

* 整体规范

1. 【推荐】建议使用组合式命名：前缀TS\_系统标识简称/数据库用户名。

正例：TS\_YPJG、TS\_RADTS

1. 【推荐】建议为用户创建单独的索引表空间，即指定各表中各项索引的存储表空间。索引表空间命名：前缀TS\_系统标识简称/数据库用户名\_INX。

说明：分开存放表数据和索引的表空间，可以优化存储管理。若条件允许，可将数据表空间和索引表空间文件分开存放在不同的物理磁盘上，减少磁盘I/O竞争，进而提高性能。

正例：TS\_RADTS\_INX

1. 【强制】必须以字母开头，禁止出现数字开头，禁止两个下划线中间出现数字，禁止在名称中使用中文。
2. 【强制】新建的任何一个表空间，原则上仅服务于某一个用户，不允许出现多个用户使用同一个表空间的情形（ORACLE默认表空间除外）。
3. 【推荐】应设置表空间文件为自动扩展模式（AUTOEXTEND），且需根据数据增加量情况设置合适的每次扩展容量大小（NEXT）。

说明：若系统存在大数据量插入的情形，则NEXT的大小不应设置的过小，否则会出现频繁自动扩展导致性能较低。

1. 【推荐】在表空间自动扩展的情形下，应根据系统实际情况，在表空间文件创建时应设置MAXSIZE的大小，单个表空间文件的最大大小不应超过操作系统环境允许的文件大小要求。一般最大值在30GB以下。

说明：不同操作系统的文件系统，所允许的文件最大大小不同：

FAT16（Windows）：支持最大分区2GB，最大文件2GB

FAT32（Windows）：支持最大分区128GB，最大文件4GB

NTFS（Windows）：支持最大分区2TB，最大文件2TB

Ext2/Ext3（Linux）：最大文件1TB

1. 【推荐】建议设置表空间为段自动管理模式（SEGMENT SPACE MANAGEMENT AUTO），若数据表中存在较多的LOB字段，则应设置为手动管理模式更合适（SEGMENT SPACE MANAGEMENT MANUAL）。
2. 【推荐】表空间文件命名，一般建议使用组合：表空间名称\_文件序号，后缀为dbf。使用序号便于为表空间添加多个表空间文件。

说明：ORACLE的5种文件类型及其扩展名，如下：

1)参数文件——后缀为ora，这些文件告诉Oracle实例在哪里寻找控制文件。例如，某个内存结构有多大，等等。

2)数据文件——后缀为.dbf，用于（这些文件包含表空间、表、索引及其他的部分）。

3)日志文件——后缀为.log，重做日志文件(事务日志)、归档日志文件。

4)控制文件——后缀为.ctl，它告诉我们数据文件在什么地方，以及关于数据文件的状态信息。

5)临时文件——后缀为.tmp，用于磁盘上的分类和临时的存储。

1. 【强制】创建表空间时，需确保指定表空间文件所在目录已经存在，否则创建失败。且需注意不同操作系统的文件目录路径格式的区别。

说明：可以在sys用户下使用查询语句“SELECT \* FROM DBA\_DATA\_FILES”查看已存在各表空间的路径。

Windows路径：'D:\oradata\TS\_RADTS\_01.DBF'；

Linux路径：'/u02/oracle/data/TS\_RADTS\_01.DBF'

示例：

/\*==============================================================\*/

/\* 创建数据表空间及文件 \*/

/\*==============================================================\*/

create tablespace TS\_RADTS

datafile 'D:\oradata\RADTS\TS\_RADTS\_01.DBF'

size 50M

autoextend on

next 100M

maxsize 8G

segment space management auto;

/\*==============================================================\*/

/\* 创建索引表空间及文件 \*/

/\*==============================================================\*/

create tablespace TS\_RADTS\_INX

datafile 'D:\oradata\RADTS\TS\_RADTS\_INX\_01.DBF'

size 50M

autoextend on

next 100M

maxsize 16G

segment space management auto;

/\*==============================================================\*/

/\* 新增TS\_RADTS\_INX表空间文件 \*/

/\*==============================================================\*/

ALTER TABLESPACE "TS\_RADTS\_INX"

ADD

DATAFILE 'D:\oradata\RADTS\TS\_RADTS\_INX\_02.DBF' SIZE 8G REUSE

AUTOEXTEND

ON NEXT 500M MAXSIZE 25G;

## 用户

* 整体规范

1. 【推荐】用户名，一般使用系统中文名的声母简拼或是系统英文名简称。

正例：区域预约诊疗系统（Regional appointment diagnosis and treatment system）的用户名：RADTS；药品监管系统的用户名：YPJG

1. 【强制】必须以字母开头，禁止出现数字开头，禁止两个下划线中间出现数字，禁止在名称中使用中文。
2. 【推荐】用户名和密码，原则上不允许相同；
3. 【强制】创建用户时，必须指定用户所属的默认表空间，且一般情况下默认表空间为新创建的表空间，原则上不允许使用其他用户在使用的表空间，禁止使用系统自带的表空间（如system、temp、users等）作为该用户的默认表空间。
4. 【推荐】为保障安全，创建用户时给新用户授权，一般遵循“满足日常使用”的原则，即够用就行，不可过多开放不必要的权限。常用权限如下表所示：

|  |  |  |
| --- | --- | --- |
| **权限** | **中文解释** | **说明** |
| CONNECT | 链接 | 不给新建用户授予connect权限，新建用户无法通过SID或SERVICE\_NAME连接数据库实例，即无法登录数据库 |
| DEBUG CONNECT SESSION | 调试链接 | 同时授权，用于调试 |
| DEBUG ANY PROCEDURE | 调试过程 |
| RESOURCE | 资源 | 若不授予此权限，则无法创建表、索引 |
| UNLIMITED TABLESPACE | 无限使用表空间 | 若不授予此权限，则用户无法自动扩展表空间 |
| SELECT ANY TABLE | 查询所有表 | 授予此权限后，可访问同一实例下各普通用户的表 |

示例：

/\*==============================================================\*/

/\* User: RADTS \*/

/\* 请根据需要修改数据库用户的密码，建议设置为"RADTS\_"加上本地区编码\*/

/\*==============================================================\*/

create user "RADTS" PROFILE "DEFAULT" identified by "RADTS\_3301" DEFAULT TABLESPACE "TS\_RADTS" ACCOUNT UNLOCK;

/\*==============================================================\*/

/\* 给RADTS用户授权 \*/

/\*==============================================================\*/

grant CONNECT, DEBUG CONNECT SESSION, DEBUG ANY PROCEDURE, RESOURCE,SELECT ANY TABLE,UNLIMITED TABLESPACE to RADTS;

## 表

* 命名规范

1. 【推荐】建议使用组合式命名：子系统或业务模块的缩写（中文字声母简拼）\_表作用的中文拼音（前两个中文字全拼+后续中文拼音首字母），且同一类业务表的前缀一致；其中若子系统或业务模块有多层次，允许使用多个下划线分割（常规下两级模块即可，不建议有太深的层次）；原则上不允许使用中文拼音+英文混合的方式作为表名。

正例：高血压里的登记、随访，GXY\_DENGJI、GXY\_SUIFANG；系统公用部分的表，XT\_YONGHU、XT\_YONGHUGW；统计业务下的门诊就诊人次，TJ\_MZ\_JIUZHENRC

反例：JianDangSYZL1、JianDangSYZL\_1、XT\_QuanXianExtend

1. 【强制】必须以字母开头，禁止出现数字开头，禁止两个下划线中间出现数字，禁止在名称中使用中文。

正例：GXY\_DENGJI

反例：1JianDangSYZL、JianDangSYZL\_3\_MX、XT\_用户表

1. 【强制】建表时，禁止在表名上使用双引号。

说明：ORACLE的表名、字段名、触发器名等，不区分大小写，但可以使用双引号强制设定。一旦用了双引号限定，在后续使用时，也必须要使用双引号，双引号内的各字符的大小写也必须与创建时一致。

1. 【强制】禁止使用ORACLE保留字作为表的名字，请参考Oracle官方保留字。

说明：V$RESERVED\_WORDS视图中有Oracle数据库里面的保留字相关信息。

1. 【强制】名称长度不能超过30个字符（ORACLE限制）。
2. 【推荐】原则上，尽量避免在表名中使用无具体意义的数字，转而使用英文字符替代；部分特殊标识意义的除外，如年份、月份的标识。

正例：2018年住院人次统计表，TJ\_ZY\_JIUZHENRC\_2018；建档索引资料表的扩展表，JianDangSYZL\_KZ

反例：JianDangSYZL1、JianDangSYZL\_1

* 注释说明

1. 【强制】表的注释说明（Comments）不允许为空，且需相对明确的指明表所具有的作用。一般建议使用：子系统或业务模块名称+表作用说明，且同一类业务表的前缀一致；其中若子系统或业务模块有多层次，允许使用多个分割线分割。

正例： MH\_JIUZHENR表的说明为“门户-就诊人信息”、MH\_TZ\_DIAOYUE表的说明为“门户-通知-调阅信息”

* 表空间

1. 【强制】若建用户时未设置默认表空间，则建表时必须指定表其所属存储表空间，原则上仅可使用本用户所属的表空间，不能使用其他用户的表空间；禁止使用系统默认的表空间（如system、sysaux、users、temp、undotbs等）。
2. 【推荐】若单表的记录数超过千万、或单表容量超过2GB、有明显的查询范围或时间段的，建议考虑表分区，且分区字段须是在更新操作时不会变更的字段。一般建议使用业务时间作为分区字段。

说明：若分区字段的内容在更新操作时发生变更、且导致所在分区出现变动，则会出现更新失败。碰到此类情况，则需先对原纪录进行delete操作，再新插入纪录。

## 字段

* 命名规范

1. 【推荐】使用字段含义的中文拼音组合：前两个中文字全拼+后续中文拼音首字母。若同一张表的多个字段的拼音组合一样，则将第三个字段改为全拼，若第三个全拼仍无法区分，则第四个字段改为全拼，依次类推。

正例：姓名：XingMing，出生日期：ChuShengRQ，地址省、地址市：DiZhiSheng、DiZhiShi

1. 【强制】必须以字母开头，禁止出现数字开头，禁止两个下划线中间出现数字，禁止在名称中使用中文。

反例：1JiBingDM、JiBingDM\_1\_ZZD、JiBingDM\_中医

1. 【强制】字段新增时，禁止在字段名上使用双引号。
2. 【强制】禁止使用ORACLE保留字作为字段的名称，请参考Oracle官方保留字。

说明：V$RESERVED\_WORDS视图中有Oracle数据库里面的保留字相关信息。

1. 【强制】名称长度不能超过30个字符（ORACLE限制）。

* 类型长度

1. 【强制】除本身不具备长度控制的类型外（如Date、BLOB、CLOB等），其他任何类型的字段，在创建时均需有明确的长度定义。

正例：varchar2(100)、number(5)、number(5,2)

反例：number、long

1. 【推荐】对于可变长度的字符型字段，建议使用varchar2类型，以保证更好的兼容性。

说明：varchar、varchar2、nvarchar、nvarchar2的区别，请翻阅oracle官方文档。

1. 【推荐】varchar2最大长度为4000字节，若字段长度大于此值，则应定义为LOB类型，且建议独立出来一张表，用主键来作关联，避免影响其他字段索引效率。
2. 【推荐】在对字符型字段进行长度定义时，需考虑汉字在不同字符集编码时所占字节数的不同，进而确定如何定义最大长度。如下表所示：

|  |  |  |
| --- | --- | --- |
| **字符集** | **单个汉字占字节数** | **varchar2实际最大汉字数** |
| SIMPLIFIED CHINESE\_CHINA.ZHS16GBK | 2 字节 | 4000 / 2 = 2000汉字 |
| SIMPLIFIED CHINESE\_CHINA.AL32UTF8 | 3 字节 | 4000 / 3 =1333 汉字 |

说明：可使用相关语句查询当前Oracle字符集环境，以及所占长度

查询当前字符集：select userenv('language') from dual;

查询占用字符数：select length('中华12') from dual; -- 返回4，也就是占用4个字符；

查询占用字节数：select lengthb('中华12') from dual; -- 返回6，也就是占用6个字节，GBK字符集下，中华各占2个字节，12两个字符各占一个字节；

1. 【推荐】对于固定长度的字符型字段（如值域为枚举代码），则建议使用Char类型，以提升性能、减少存储。非固定长度的字段，禁止使用char类型。

说明：若非固定长度字段使用了char类型，实际入库时的字段值长度小于定义的长度，则会自动补足空格，不仅浪费存储空间，可能还会导致数据查询失败。

1. 【推荐】对于数值类型的字段，建议使用Number类型。对于小数类型，亦使用Number(p,s)类型进行定义，其中p为最大总长度，s为小数点后位数（整数部分的最大长度为p-s）。不建议使用float、double类型。

说明：oracle在语法上支持float、double等类型，是为了SQL的兼容性，其底层实际上都是Number类型。

1. 【推荐】若某字段的值为0-9的数值（多为枚举类型），当需要在此字段上建立索引时，应当首选使用number(1)类型，以提升索引效率；若不需要建立索引，建议定义数据类型为char(1)，以减少存储空间。
2. 【强制】需要两表或多表Join的字段，数据类型必须绝对一致。

* 默认值

1. 【推荐】对于值域为枚举值、且枚举值为互斥内容时，建议在建表时添加默认值。如1-是、0-否，1-有、2-无等。

说明：枚举值类型的默认值，一般使用代码值，不建议直接使用中文。

1. 【推荐】对已经存在较大数据量的表增加枚举值字段、且该字段需有默认值时，为避免执行时间较长、等待假死、锁表情况出现，一般建议在add column时不添加默认值，添加字段成功后通过update操作进行设置默认值，最后再对该字段进行变更操作增加默认值。

* 注释说明

1. 【强制】任何字段，均须有注释说明。
2. 【推荐】字段值为枚举，值域较少时需直接写出各枚举值及其含义，若值域较多，则需指明其值域所对应的字典代码的出处。
3. 【推荐】如果修改字段含义或对字段表示的状态追加（即枚举值域扩充）时，需要及时更新字段注释。

* 附加规范

1. 【强制】表必备的三个字段：ID、INSERT\_TIME、UPDATE\_TIME

说明：ID为主键字段，类型推荐为Number(20)，使用一对一匹配的序列进行自增长；类型亦可以使用Varchar2(36)，使用GUID生成唯一值。INSERT\_TIME为插入时间，DATE类型，创建时设置默认值为sysdate。UPDATE\_TIME为更新时间，DATE类型，创建时设置默认值为sysdate，更新时必须赋值数据库服务器的当前系统时间。

1. 【推荐】业务上不会出现为空情况的字段，需在定义时设置为非空。
2. 【推荐】字段允许适当冗余，以提高查询性能，但必须要考虑数据一致性。冗余字段应遵循：
3. 不是频繁修改的字段；
4. 不是varchar2超长字段，更不能是LOB字段；

## 主键

* 命名规范

1. 【强制】使用组合式命名：PK\_表名

正例：公用字典表的表名为GY\_JICHUZDMX，其对应的主键为PK\_GY\_JICHUZDMX

1. 【强制】总长度不能超过30个字符（ORACLE限制）。

* 设计规范

1. 【强制】任何表，必须建主键。
2. 【强制】主键的类型为Primary，不允许使用Unique、Foreign类型。

说明：在PLSQL中创建主键时，会自动作为一个Unique类型的索引；而使用Power Designer创建主键时，需勾选using index选项。

1. 【强制】主键列，必须使用单一的ID字段，不允许使用业务上的唯一约束（单个或多个）列作为主键列。

说明：ID字段的说明，参见字段的附加规范。

业务上的唯一约束列，应当为其建立Unique类型的索引。

1. 【推荐】若已为用户创建单独的索引表空间，则需为主键对应的索引分配存储表空间。

示例：

alter table GY\_JICHUZDMX

add constraint PK\_GY\_JICHUZDMX primary key (ID)

using index

tablespace TS\_RADTS\_INX;

## 外键

禁止使用。

不推荐理由如下：

1. 性能问题：若建立一个或多个外键，在数据处理（插入、删除、更新）时，必会对关联的表进行数据检查，降低性能，尤其是在频繁的进行数据插入、或是关联表为大数据量时，影响尤为突出；
2. 先后逻辑定死：外键定死了两个表之间的数据生成先后关系，但实际业务处理时可能会出现反向的先后顺序，导致无法处理；
3. 数据完整性：设计外键，主要是为了维护数据的完整性，但实际操作时，可以在程序中使用事务进行完整性的判断、处理，而非建外键；
4. 维护麻烦，且存在人为因素导致约束失效；
5. 会用的人不多，设计人员清楚，研发、实施人员不一定清楚。

## 索引

* 命名规范

1. 【强制】使用组合式命名：前缀IDX\_表名\_含义标识，含义标识可为首字母简拼，以便减少总长度。

正例：如项目类别编码字段上的索引 IDX\_GY\_JICHUZDMX\_LBDM

1. 【强制】总长度不能超过30个字符（ORACLE限制）。

* 设计规范

1. 【推荐】常规情况下，单张表中的索引数量，包括主键索引在内，不应超过7个。索引建的越多，对于数据入库的性能影响越大。
2. 【推荐】不建议在小表上建索引。
3. 【推荐】若经常在表中的某个字段上进行Order By操作，则建议对此字段建立索引。
4. 【强制】业务上具有唯一特性的字段，即使是多个字段的组合，亦需建成唯一索引。

说明：唯一索引会影响insert速度，但这个速度损耗可以忽略，其所带来的查询速度的提高是明显的；另外，即使在应用层做了非常完善的校验机制，没有唯一索引，根据墨菲定律，必然会有脏数据产生。

1. 【推荐】建立索引的字段，一般选择区分度（可选性 Selectivity）较高的；区分度较低，或是只有两三个值得字段，亦或是检索结果行数超过表总行数的15%以上时，不适合建索引。

正例：如公用字典明细表的项目类别编码字段，属于上级分类编码，区分度高，建立索引 INX\_GY\_JICHUZDMX\_LBDM

反例：是否注销SHIFOUZX、修改标志XIUGAIBZ，其值域只有两三种

1. 【推荐】建立索引的字段，应尽量避免出现大量为空值的情况；对于部分枚举值类型的字段，若必须在此列上建索引，则建议使用默认值代码代替空值。
2. 【推荐】建复合索引时，应尽最大程度考虑并满足以下几点情况：
3. 考虑前缀性，即在SQL语句中，只有将复合索引的第一个字段作为约束条件，该复合索引才会启用；
4. 高可选性，即区分度最高的字段放在左边；
5. 用的次数较多的放在左边；
6. 复合索引开销较但字段索引大，若某个字段为主键、唯一字段，或者是区分度非常高的字段，也不一定非要建成复合索引；
7. 【推荐】若已为用户创建单独的索引表空间，则需为主键对应的索引分配存储表空间。

## 序列

* 命名规范

1. 【强制】使用组合式命名：前缀SEQ\_表名\_对应字段名。

正例：

表EHR\_YILIAOHD的ID主键的序列：SEQ\_EHR\_YILIAOHD\_ID

表GY\_JICHUZD的排序字段的序列：SEQ\_GY\_JICHUZD\_PX

1. 【强制】总长度不能超过30个字符（ORACLE限制）。

* 设计规范

1. 【强制】若为递增（递减）序列，为序列设置最大值MAXVALUE（最小值MINVALUE）。若该序列值最终使用在某个字段中，则序列最大值长度，应与关联字段的长度一致。
2. 【强制】若序列数据作为数据入库，禁止使用CYCLE，防止出现重复数据导致的数据冲突。
3. 【推荐】若业务上并没有要求序列的值必须具有连贯性、不允许出现中断，则一般建议设置序列为CACHE，以提升性能。
4. 【推荐】存在高并发访问时，应设置序列为CACHE，且CACHE值可适当扩大。

说明：创建NOCACHE在高并发访问时，容易导致row cache lock等待事件，主要原因是每次获取nextval时都需要修改rowcache中的字典信息，且在commit修改信息时，亦可能导致log file sync等待。

1. 【推荐】建议使用ORDER模式。

语法及参数说明：

CREATE SEQUENCE 序列名 [相关参数];

参数说明

INCREMENT BY：序列变化的步进，负值表示递减。(默认1)

START WITH：序列的初始值 。(默认1)

MAXvalue：序列可生成的最大值。(默认不限制最大值，NOMAXVALUE)

MINVALUE：序列可生成的最小值。(默认不限制最小值，NOMINVALUE)

CYCLE：用于定义当序列产生的值达到限制值后是否循环(NOCYCLE:不循环，CYCLE:循环)。

CACHE：表示缓存序列的个数，数据库异常终止可能会导致序列中断不连续的情况，默认值为20，如果不使用缓存可设置NOCACHE

ORDER：按申请顺序产生序列。

-- 示例

create sequence SEQ\_GY\_JICHUZD\_PX

minvalue 1

maxvalue 999999999999

start with 1

increment by 1

cache 20

order;

## 视图

* 命名规范

1. 【强制】普通视图，使用组合式命名：前缀V\_子系统或业务模块的缩写\_视图作用拼音缩写，且同一类业务视图的子系统或业务模块的缩写应一致；其中若子系统或业务模块有多层次，允许使用多个下划线分割；原则上不允许使用中文拼音+英文混合的方式作为表名。

正例：V\_EHRVIEW\_CHUFANG、V\_CBB\_JCZD\_XZQH、V\_CBB\_JCZD\_ZLKS

反例：V\_JianDangSYZL\_1、V\_XT\_QuanXianExtend

1. 【强制】物化视图，使用组合命名：前缀MV\_子系统或业务模块的缩写\_视图作用拼音缩写。
2. 【强制】建视图时，禁止在视图名上使用双引号。
3. 【强制】名称长度不能超过30个字符（ORACLE限制）。
4. 【推荐】原则上，尽量避免在视图名中使用无具体意义的数字，转而使用英文字符替代；部分特殊标识意义的除外，如年份、月份的标识。

* 设计规范

1. 【强制】创建视图的Create语句上，建议加上 or replace关键字，便于视图的更新。

create or replace view 视图名 as 子查询

1. 【推荐】考虑到视图的SQL编译特性、性能上的问题，不建议视图中嵌套使用视图。
2. 【强制】编写视图内容时，禁止出现连续两个及以上的空行。若存在两个以上连续空行，脚本在Command命令下会执行失败。
3. 【强制】在视图编写结束后，以表注释的语法，为视图添加注释说明。

说明：视图又称虚表，Oracle中将其归类于表，只是普通表的TABLE\_TYPE为TABLE，视图的TABLE\_TYPE为VIEW，表的注释语法同样适用于视图

示例：

CREATE OR REPLACE VIEW V\_CBB\_JCZD\_XZQH AS

SELECT

A.HUJIBH,

A.HUJIMC,

A.SHANGCENGBH,

A.HUJIJB,

A.SHIFOUZX

FROM CBB\_SHENGSHIQX A

ORDER BY A.HUJIBH;

COMMENT ON TABLE V\_CBB\_JCZD\_XZQH IS '视图-基础字典-行政区划'; -- 使用表的注释语法，为视图添加说明

1. 【推荐】视图编写完毕后，建议使用PLSQL的代码美化器进行美化。

## 函数

尽量不要使用。

* 命名规范

1. 【强制】使用组合式命名：前缀FUN\_[子系统或业务模块的缩写\_]动名词+中文拼音（前两个中文字全拼+后续中文字声母简拼）；

正例：FUN\_GetPinYinM、FUN\_JD\_GetGeRenJWBS

1. 【推荐】入参的命名规范，参考字段的命名规范。

* 设计规范

1. 【强制】创建函数的Create语句上，建议加上 or replace关键字，便于函数的更新。
2. 【强制】自定义函数，只能作查询、计算使用，禁止在函数内部做insert、update、delete操作。
3. 【强制】必须至少有一个输入参数，并指定入参的数据类型。
4. 【推荐】函数通常使用在DML SQL语句中，考虑上下文环境，不建议使用out参数，而应直接使用返回值得到结果。
5. 【强制】函数内部，需要有异常处理块代码。
6. 【推荐】需对函数的用途、作者、出入参、返回值添加必要的注释说明。
7. 【推荐】在函数编写完毕后，使用PLSQL的代码美化器进行美化。
8. 【强制】若sql脚本文件包含内容不仅仅只有过程，且函数创建语句在前，其他语句在后，则必须在该函数脚本结束符后，另起一行添加“/”。否则sql脚本在Command命令中会执行失败。

说明：过程的sql脚本，可使用Export User Objects（导出用户对象）、或在PLSQL中使用Ctrl+S命令进行导出保存，结果文件中会自动添加“/”，不可使用Ctrl+C、Ctrl+V方式进行复制黏贴。

## 触发器

尽量不要使用。

* 命名规范

1. 【推荐】普通视图，使用组合式命名：前缀TRI\_表名缩写\_触发器作用拼音缩写。
2. 【强制】名称长度不能超过30个字符（ORACLE限制）。

* 设计规范

1. 【推荐】触发器在性能、维护便捷性上，都不具备使用价值；且因触发器是隐藏的，触发器的失效与否，对DML操作没有影响，存在漏洞，故，能不用就不用。
2. 【推荐】创建触发器的Create语句上，建议加上 or replace关键字，便于触发器的更新。
3. 【强制】在触发器的执行部分只能用DML语句（SELECT、INSERT、UPDATE、DELETE），不能使用DDL语句（CREATE、ALTER、DROP）。
4. 【强制】触发器中不能使用commit语句，触发器的操作与触发事件（INSERT,UPDATE,DELETE）一起进行COMMIT和ROLLBACK。
5. 【强制】禁止在触发器中嵌套使用触发器。
6. 【推荐】若数据表存在大批量数据DML，或是高并发情况，应避免使用触发器，改用程序逻辑实现或存储过程处理。
7. 【推荐】一个表上的触发器越多，对于表的DML操作性能影响越大，需尽量减少触发器的使用。对单表的多种触发性的同类操作，尽量在同一个触发器中完成，不允许创建多个触发器。

## 存储过程

老系统尽量不新增，新系统禁止使用。

* 命名规范

1. 【推荐】使用组合式命名：前缀USP\_[子系统或业务模块的缩写\_]动名词+中文拼音（前两个中文字全拼+后续中文字声母简拼）；

正例：USP\_CBB\_GetFuWUJG

1. 【推荐】入参的命名规范，参考字段的命名规范。

* 设计规范

1. 【推荐】考虑异构数据库的移植性，应尽量少使用存储过程。
2. 【推荐】创建存储过程的Create语句上，建议加上 or replace关键字，便于更新。
3. 【强制】必须为每一个参数指定数据类型、输入输出类型。
4. 【强制】编写过程时，禁止出现连续两个及以上的空行。若存在两个以上连续空行，脚本在Command命令下会执行失败。
5. 【推荐】在过程中使用FETCH语句，必须要使用%NOTFOUND，以便正常终止游标FETCH循环。
6. 【推荐】尽可能使用显示游标，避免使用隐式游标。且在使用游标前，应检查游标是否已打开，在结束后（无论是正常终止还是异常退出），都需要关闭所有已打开的游标。
7. 【强制】存储过程内部，需要有异常处理块代码，对错误进行捕捉并处理。
8. 【强制】需对过程的用途、作者、出入参、返回值、以及内部重要逻辑添加必要的注释说明。
9. 【推荐】在过程编写完毕后，使用PLSQL的代码美化器进行美化。
10. 【强制】若sql脚本文件包含内容不仅仅只有过程，且过程创建语句在前，其他语句在后，则必须在该过程脚本结束符后，另起一行添加“/”。否则sql脚本在Command命令中会执行失败。

说明：过程的sql脚本，可使用Export User Objects（导出用户对象）、或在PLSQL中使用Ctrl+S命令进行导出保存，结果文件中会自动添加“/”，不可使用Ctrl+C、Ctrl+V方式进行复制黏贴。

## 包

老系统尽量不新增，新系统禁止使用。

# 使用规范

## Select

1. 【推荐】建议在select语句中为表指定别名字符，并使用“别名字符.字段名”方式查询结果字段

正例：

select t.code,t.name,t.value,t.xiugaisj,t.memo,t.edit,t.paixu from gy\_system t;

1. 【推荐】select子句中避免使用（\*），需要哪些字段必须显式写明。

说明：Oracle在解析的过程中，会将“\*”依次转换成所有的列名，这个工作是通过查询数据字典完成的, 这意味着将耗费更多的时间。

正例：

select t.code,t.name,t.value,t.xiugaisj,t.memo,t.edit,t.paixu from gy\_system t;

select count(1) from gy\_system t

反例：

select \* from gy\_system t;

select count(\*) from gy\_system t;

1. 【推荐】

## where条件

1. 【推荐】条件顺序，Oracle 10g及以上，建议将能使结果最少的条件放在最右边，Oracle 10g以下，建议将能使结果最少的条件放在最左边。
2. Oracle在不同的版本中，解析和执行SQL语句的顺序不一样，10g以下从左至右，10g以上从右至至左。
3. Oracle在RBO优化器模式下（10g及以上确实为RBO模式），会对SQL执行顺序进行自动优化处理，首先会找出索引及表连接的条件，过滤数据集，条件顺序对性能几乎无影响。考虑实际环境的不确定性、以及性能优化的最大化，仍然建议在编写SQL语句时采用上述条件顺序规则，养成良好习惯。
4. 【推荐】Where子句中，尽量避免在列上使用函数，此操作会使索引失效。且一般字段列在等式左边，函数操作在等式右边。

正例：

-- yewurq 类型为varchar2(10)，格式yyyy-mm-dd

where yewurq = to\_char(sysdate,'yyyy-MM-dd')

反例：

where to\_date(yewurq,’yyyy-MM-dd’) = trunc(sysdate);

1. 【推荐】防止因字段类型不同造成的隐式转换，导致索引失效。

反例：

-- dangqianzt类型为char(1)

where dangqianzt = 1; -- 进行了类型的隐式转换（int->char），导致索引失效

1. 【推荐】in或not in操作能避免则避免，若不能避免，in或not in中的集合元素数量应小于1000（Oracle限制）。
2. 【推荐】在内表较大的情况下，需用exists或not exists代替in或not in。

in适合于外表大而内表小的情况；exists适合于外表小而内表大的情况。

1. 【推荐】