**分析平台[SpiritAnalysis]**

**数据库设计及备忘**

|  |  |  |
| --- | --- | --- |
| 文件状态：  [√] 草稿  [ ] 正式发布  [ ] 正在修改 | **文件标识：** | SPIRITANALYSIS-DATABASE(all) |
| **当前版本：** | 0.4 |
| **作者：** | 王晖 |
| **最后修改日期：** | 2014-11-20 |

**版本历史**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本/状态** | **作者** | **参与者** | **起止日期** | **备注** |
| 0.1发布 | 王晖 | 马海涛 | 2014-10-10 | 建立文档，对第一版本数据库存储进行设计 |
| 0.2发布 | 王晖 | 马海涛 | 2014-10-30 | 把字典内容加入 |
| 0.3发布 | 王晖 | 马海涛 | 2014-11-19 | 加入了文件数据管理的内容  修改了用户表，以便配合注册的功能 |
| 0.4发布 | 王晖 |  | 2014-11-20 | 对文件管理做了大幅度修改，把原来的导入文件管理与文件管理进行了合并 |
| 0.4.1发布 | 王晖 |  | 2014-11-24 | 文件管理做了修改：  1-反关系为inverse而不是anti  2-文件分类为categroy而不是原来的class |
| 0.5.0发布 | 王晖 |  | 2015-03-5 | 加入任务的设计 |
|  |  |  | 2015-04-03 | SA\_MD\_COLSEMANTEME表中：   1. 删除了语义权重列； 2. 语义类型长度由1改为4；   SA\_MD\_TABMODEL表中：  1、增加了titleName字段 |

**SQL文档**

参见”MySqlDDL.sql”

**目录**

[0. 文档介绍 4](#_Toc404613299)

[0.1. 文档目的 4](#_Toc404613300)

[0.2. 文档范围 4](#_Toc404613301)

[0.3. 读者对象 4](#_Toc404613302)

[0.4. 文档说明 4](#_Toc404613303)

[0.5. 关系说明 4](#_Toc404613304)

[1. 数据库环境说明 5](#_Toc404613305)

[2. 数据库规范 5](#_Toc404613306)

[2.1. 表的命名规则 5](#_Toc404613307)

[2.2. 布尔值取值 5](#_Toc404613308)

[2.3. 键 5](#_Toc404613309)

[2.4. 字段/列 6](#_Toc404613310)

[2.5. 数值类型 6](#_Toc404613311)

[3. 编程注意 7](#_Toc404613312)

[4. 设计过程说明 7](#_Toc404613313)

[4.1. 第一期内容 7](#_Toc404613314)

[5. 具体设计 7](#_Toc404613315)

[5.1. 表汇总 7](#_Toc404613316)

[5.2. 用户类{1} 8](#_Toc404613317)

[5.2.1. 用户[PLAT\_USER] 8](#_Toc404613318)

[5.3. 字典(枚举)类{2} 9](#_Toc404613319)

[5.3.1. 字典组[PLAT\_DICTM] 9](#_Toc404613320)

[5.3.2. 字典项[PLAT\_DICTD] 10](#_Toc404613321)

[5.4. 元数据信息{6} 12](#_Toc404613322)

[5.4.1. 元数据模式[SA\_MD\_TABMODEL] 12](#_Toc404613323)

[5.4.2. 元数据列描述[SA\_MD\_COLUMN] 13](#_Toc404613324)

[5.4.3. 元数据列语义[SA\_MD\_COLSEMANTEME] 13](#_Toc404613325)

[5.4.4. 元数据表对照[SA\_MD\_TABMAP\_REL] 14](#_Toc404613326)

[5.4.5. 实体表指标[SA\_MD\_TABQUOTA] 15](#_Toc404613327)

[5.4.6. 实体列指标[SA\_MD\_COLQUOTA] 15](#_Toc404613328)

[5.5. 数据导入类{1} 16](#_Toc404613329)

[~~5.5.1.~~ ~~数据导入日志[SA\_IMP\_LOG]~~ 16](#_Toc404613330)

[5.5.2. 数据文件/实体表对应[SA\_IMP\_TABMAP\_REL] 16](#_Toc404613331)

[5.6. 文件管理{4} 17](#_Toc404613332)

[5.6.1. 文件记录索引[SA\_FILE\_INDEX] 17](#_Toc404613333)

[5.6.2. 文件分类 [SA\_FILE\_CATEGORY] 18](#_Toc404613334)

[5.6.3. 文件关系[SA\_FILE\_REL] 18](#_Toc404613335)

[5.6.4. 反向文件关系[vSA\_FILE\_INVERSEREL] 19](#_Toc404613336)

[5.6.5. 数据导入日志[vSA\_IMP\_LOG] 20](#_Toc404613337)

[6. 数据逻辑视图 21](#_Toc404613338)

[7. 数据库管理与维护说明 22](#_Toc404613339)

1. 文档介绍
   1. 文档目的

描述分析平台数据对象实体，为平台建造提供核心信息架构。平台代码设计必须以数据库设计为依据。

* 1. 文档范围

包括底层平台的数据库及XML/json设计两类。

* 1. 读者对象

·平台开发组人员、测试人员、及产品相关人员；

·基于数据平台进行开发的人员；

* 1. 文档说明

·红色粗体字体部分为本此修改或需要注意的内容；

·蓝色粗体字体部分为需要完善和补充的内容；

·在表格中：黄色底的字段为主键；

·在表格中：淡蓝绿底的字段为外键或其值从其他表得到；

·在表格中：综色底的字段为冗余字段；

* 1. 关系说明

·本设计基于平台的数据库设计，参见(简单用户模块及权限设计(DB).docx)；

1. 数据库环境说明

希望满足跨数据库平台的要求，但目前只对MySql数据库。

若以Oracle为基准，表空间统一为platform，数据库用户统一为platform，口令platform，权限限制为资源访问（如：记录的增、删、改、查等）和应用级管理（建表、改表、删表、建索引等）。

**注意：各项目可以有自己的用户名，如海南项目——platform\_hn。**

1. 数据库规范
   1. 表的命名规则

·分析平台部分的表以“sa\_功能”命名，如sa\_mdTable(元数据-表)；

·表名最长为100个字符(由程序判断)，这受到PLAT\_SERIALNO表table\_name字段长度的限制。

·与权限相关的表用后缀“\_ORG”标明

·与关联关系相关的表用后缀“\_REL”标明

* 1. 布尔值取值

·用数值类型表示布尔值，数据库种类型为“NUBMER”，java中类型为int；

·0（空）、1（是）、2（否）；说明：**由于java中数值int的默认值为0，为便于编程采用本条规则；**

·布尔字段包括包括字段：

1. **isValidate**—是否有效：1（有效）、2（无效），与delTag意思是相近的。

**b)hasChild**—是否有子节点：1（是）、2（不是）；

* 1. 键

·主键用id命名，UUID或长整型，若为长整型，则从1开始编号，若为UUID则长度为VARCHAR2(32)。

·父结点外pId命名；

·主子关系，子表到主表的外键用mId；

·其他外键命名：用功能缩写Id，如uId或userId：用户ID，ddId：字典项ID；

**·实体表的ID用长整型，如用户表，情报表，公文表；内部关系表用UUID，如资产文件关联表，用户组用户关联表；**

* 1. 字段/列

1）全部小写：在Oracle中对象名称大小写不敏感，除非新建时用””框起来

2）每个英文单词或缩写与英文单词间用“\_”分开，如user\_name，un\_py，b\_code

3）是判断类型的用is+单词，并且单词首字母大写，如isValidate，isUnit

4）为了避免一些数据库系统的保留字，type，name，level均不作为表名或列明（包括其它一切用户对象的名称）

5）常用的命名及规范：

a)isValidateNUMBER //是否生效，1有效；2无效，默认值1

b)types NUMBER //类别，注意不用type，以免作为关键字；

c)descnVARCHAR2(2000) //说明字段

6）树结构通用规范，所有的数都要按这个建立：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段中文名** | **字段名** | **数据类型** | **空否** | **约束条件** | **说明** |
| 主键 | Id | NUMBER | 非空 | 主键PK | 从PLAT\_SERIALNO生成 |
| 父结点ID | pId | NUMBER | 非空 | 默认值为0 | 用户组表中的ID(外键)  若是第一级用户组，此值为0 |
| 是否有子结点 | hasChild | NUMBER | 非空 | 默认值2无子 | 1有结点,2无子结点 |
| 字典项层数 | Levels | NUMBER | 非空 | 默认值1无子 | 从1开始 |
| 排序 | Sort | NUMBER | 非空 | 默认值0 | 数值大者靠前，从0-99998 |
| 索引编码 | indexCode | VARCHAR2(1100) | 非空 |  | 计算得到，每级10位长，前五位为本级排序，后五位为主键，支持100级 |
| 是否生效 | isValidate | NUMBER | 非空 | 默认值1有效 | 1有效；2无效 |

* 1. 数值类型

所有用数值类型表示状态分类的字段，0都作为空值处理，这样避免程序中没有赋值的数值变量对数据库的影响。

1. 编程注意

·若主键采用长整型，并从PLAT\_SERIALNO生成，则SQLMap的namespace属性要和PLAT\_SERIALNO.table\_name字段的值项对应，否则不能争取获得序列号。**系统中强制规定：namespace、表明、PLAT\_SERIALNO.table\_name字段对应的值，三者一致。**

1. 设计过程说明
   1. 第一期内容

本期主要包括用户，元数据，业务数据，注册用户，业务数据等

**本期内容包括如下表：**

1. 具体设计
   1. 表汇总

下表按功能对表进行了划分，在具体介绍表的结构时也采用这一划分。

目前平台表分为如下：

|  |  |  |
| --- | --- | --- |
| **分类** | **表名** | **功能说明** |
| **1、用户类{1}** | PLAT\_USER（用户） | 用户表 |
| **2、字典类{2}** | PLAT\_DICTM（字典组） | 字典组表 |
| PLAT\_DICTD（字典项） | 字典项 |
| **3、元数据{6}** | SA\_MD\_TABMODULE（元数据模式） | 这两个表是元数据最基础的核心结构 |
| SA\_MD\_COLUMN（元数据列描述） |
| SA\_MD\_COLSEMANTEME（元数据语义） | 此表直接与列表相关 |
| SA\_MD\_TABQUOTA（实体表指标） | 这两个表用于对表数据信息的质量统计 |
| SA\_MD\_COLQUOTA（实体列指标） |
| SA\_MD\_TABMAP\_REL（模式实体表对应） | 此表是各类表的中心联系 |
| **4、数据导入类{1}** | ~~SA\_IMP\_LOG（导入日志）~~ | ~~主要记录文件信息~~ |
| SA\_IMP\_TABMAP\_REL（文件/实体表对应） | 文件中数据与实体表的对应关系 |
| **5、文件管理{5}**  **3+2** | SA\_FILE\_INDEX（文件记录索引）  原生态文件 | 主要记录日志和分析文件信息 |
| SA\_FILE\_CATEGORY（文件分类表） | 记录文件在系统的类别  一个文件可能在本系统中分属不同的类别 |
| SA\_FILE\_REL（文件关系对应） | 文件之间的关系：  可以是分类的关系，也可以是原生态文件的关系 |
| vSA\_FILE\_INVERSEEL  (文件反向对应关系) | 文件之间关系的反向 |
| vSA\_IMP\_LOG(导入数据文件的日志) | 导入数据文件的记录 |
| **6、报告管理{2}**  **1+1** | SA\_REPORT\_INFO（文件分类表） |  |
| vSA\_REPORT\_FILE(报告文件信息) | 报告文件信息 |
| **7、任务管理{4}**  **3+1** | SA\_TASK\_GROUP（任务组表） | 任务的分组信息记录。  有意义的工作是有一组任务来完成的，此表即维护这个分组信息 |
| SA\_TASK\_INFO（任务信息表） | 记录任务信息，包括执行类 |
| SA\_TASK\_REL（任务关系表） | 记录任务间的依赖关系是图的关系 |
| vSA\_TASK | 任务全信息视图 |

注意：本设计中的视图，在程序的业务处理中用处可能不大，主要在管理和查询中能用到。

* 1. 用户类{1}
     1. 用户[PLAT\_USER]

**说明：此表是在原框架中用户表基础上进行修改的，主要是增加了一些列，增加的列用红色表明。**

用户功能说明：

|  |
| --- |
| 这里简要说明一下用户的功能，用以备忘。类似互联网上的用户，通过注册方式录入，可以导入QQ/淘宝等用户。目前只设计最简单的用户功能。  用户相关功能包括：用户注册，用户修改。  *用户注册——*通过邮箱，用户名/密码/昵称，验证码；今后可能会有——用户联系信息/用户实名信息/用户个性化信息(头像/昵称/语录名等)。  *用户修改——*目前只能修改密码和名称 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **PLAT\_USER（用户）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束条件** | **说明** |
| 用户ID | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 登录名 | loginName | VARCHAR2(15) | 非空 | 唯一索引 | 可以自动生成 |
| 用户名 | userName | VARCHAR2(100) | 非空 |  |  |
| 密码 | Password | VARCHAR2(30) | 可空 |  |  |
| 邮箱 | mailAdress | VARCHAR2(100) | 非空 | 唯一索引 |  |
| 昵称 | nickName | VARCHAR2(100) | 可空 |  |  |
| 分类 | userType | NUMBER(1) | 非空 | 默认值为1 | 1自然人用户；2机构用户 |
| 备注 | Descn | VARCHAR2(2000) | 可空 |  |  |
| 用户状态 | UserState | INT(1) | 非空 |  | 0表示未激活，1表示已激活 |
| 验证序列 | validataSequence | VARCHAR2(32) | 可空 |  | 账号激活的验证码 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| 最后修改 | lmTime | TIMESTAMP(4) | 非空 |  | 每次更新的时间 |
| **补充说明** |  | | | | |
| **索引** | 1-id，主键  2-login\_name，IDX\_USER\_LNAME，保证登录名唯一  3-mailAddress，IDX\_USER\_MAIL，保证邮箱唯一 | | | | |
| **基本操作** |  | | | | |
| **程序使用** |  | | | | |

* 1. 字典(枚举)类{2}

字典结构由字典组合字典项构成。

字典组是一类字典项的统称，字典项采用排序树进行存储。

如“行政区划”，由行政单位组成的树，这棵树存储在字典项表中，而字典组表中存储“行政区划”这个统称及一些其他信息。

另外，用得很多的一级字典项，也用数来存储，如“性别”，其树就是一级树。另外，

* + 1. 字典组[PLAT\_DICTM]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **PLAT\_DICTM（字典组）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 字典组ID | id | VARCHAR2(32) | 非空 | 主键PK | 从PLAT\_SERIALNO生成 |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型1=用户；2=Session |
| 字典组名称 | dmName | VARCHAR2(200) | 非空 |  |  |
| 名称拼音 | nPy | VARCHAR2(800) | 可空 |  |  |
| 排序 | sort | NUMBER | 非空 | 默认值0 | 数值大者靠前，从0-99998 |
| 是否生效 | isValidate | NUMBER | 非空 | 默认值1有效 | 1有效；2无效 |
| 字典组类型 | mType | NUMBER | 非空 | 默认值3定义 | 1系统保留；2、系统；3定义； |
| 引用描述 | mRef | VARCHAR2(400) | 可空 |  | 可以是：  1-所引用的字典组：用“#+MID”标明的用“，”号分开的字典项ID，如“#3,#12”表明此字典向引用了id为3和12的字典组；  2-所参考的文件名：标准文件名称，如《国家XX标准 GB-23212》;  3-所参考的文件（存储在系统中）：用“$+FID”标明的用“，”号分开的文件ID  若包括多项以上内容，则各项间用“；”隔开。  如：#3,#45;《国家XX标准 GB-23212》,《国家XX标准GB-23》;#45;$2034,$2342 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  |  |
| 最后修改时间 | lmTime | TIMESTAMP(4) | 非空 |  |  |
| 说明 | descn | VARCHAR2(500) | 可空 |  |  |
| **补充说明** | 1-通过此表中的记录管理不同的字典项，资源与字典项的对应见表：PLAT\_RESOURCE\_SCHEMA(注意要进行必要的修改)  2-关于mType的说明：  mType=1，系统保留，只能修改名称和排序；不能向其内部加入新的字典项，其下属字典项的dType必须是1；（在本系统中，这应该是汇总得到的，属于data2）  mType=2，系统，只能修改名称和排序；可以向其下增加字典项，其下属字典项的dType是2或3；  mType=3，用户定义的字典组，可以修改和删除，并向其下加字典项，其下属字典项的dType必须3或4；(用户自己定义的)  3-修改了如下内容，要在本此平台整理中修改过来：  a)字典组名称，从name改为dm\_name  **本次实现与之前相比，变化如下：**  **1-加入了所有者信息** | | | | |
| **索引** | 1. id，主键 | | | | |

* + 1. 字典项[PLAT\_DICTD]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **PLAT\_DICTD（字典项）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束条件** | **说明** |
| 字典项ID | id | NUMBER | 非空 | 主键PK | 从PLAT\_SERIALNO生成 |
| 主表字典组ID | mId | NUMBER | 非空 | 默认值为0，无意义 | 字典组表中的ID(外键) |
| 父结点ID | pId | NUMBER | 非空 | 默认值为0，无意义 | 字典明细表的ID(外键)  若是明细表中的第一级字典项，此值为0 |
| 排序 | order | NUMBER | 非空 | 默认值0 | 数值大者靠前，从0-999999 |
| 是否生效 | isValidate | NUMBER | 非空 | 默认值1有效 | 1有效；2无效 |
| 字典项名称 | ddName | VARCHAR2(200) | 非空 |  |  |
| 名称拼音 | nPy | VARCHAR2(800) | 可空 |  |  |
| 别名或简称 | aliasName | VARCHAR2(200) | 可空 |  | 类似于省的简称，如川 |
| 别名拼音 | anPy | VARCHAR2(800) | 可空 |  |  |
| 业务编码 | bCode | VARCHAR2(50) | 非空 |  | 在新增时，若业务编码为空，则默认把业务编码设置为字典项的ID  在同一字典组内不能有相同的业务编码。  若是引用字典项，则其业务编码为“所引用字典组ID+’$’+所引用字典项的业务编码”，引用字典项此值不能改变 |
| 字典项类型 | dType | NUMBER | 非空 | 默认值3定义 | 1系统保留；2系统；3定义；4引用：其他字典项ID； |
| 引用描述 | dRef | NUMBER | 可空 | 默认值为0，无意义 | 若DTYPE=4，则此为引用的字典项的ID |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  |  |
| 说明 | descn | VARCHAR2(2000) | 可空 |  |  |
| **补充说明** | 1-外键不建立，只是说明关系  2-业务编码不能为空，在新增时，若业务编码为空，则默认把业务编码设置为字典项的ID  2-关于dType的说明：  dType=1，系统保留，只能修改名称和排序；也不能扩充其结构；  dType=2，导入系统，只能修改名称和排序；可以向其下增加字典项，其下属字典项的dType是2或3；  dType=3，用户定义的字典项，可以修改和删除，并向其下加字典项，其下属字典项的dType必须3；  dType=4，引用自其他字典项，其下属字典项的dType是3或4；  3-修改了如下内容，要在本此平台整理中修改过来：  a)字典项名称，从name改为dd\_name  **本次实现与之前相比，变化如下：**  **1-原来字典项表不存储字典组的根，现在存储根，并且，level从0开始**  **2-去掉了indexCode列**  **3-由于本次是导入字典项，所以可能会是这样的情况：**  **·只有1层结点**  **·isValidate总是生效；**  **·别名为空，没有实际意义**  **·字典性类型，都选取为2/3这个在上面已经解释；**  **·业务编码：这个需要复杂的分析才能导入，目前和主键相同** | | | | |
| **索引** | 1. id，主键   为便于统计分析，建立如下所引，这些索引可能要在之后进行调整   1. mId,bCode，此为隐含的唯一索引，但不建立(同一字典项下面不能有相同的业务编码)   由于字典信息进行缓存，因此除主键外，不建立其他索引 | | | | |

* 1. 元数据信息{6}

用来存储元数据的基本信息。这里又可以进行如下分类：

* **基础元数据：**记录元数据核心基础信息
* **元数据指标：**记录元数据的指标(统计)信息。
* **元数据语义：**记录元数据的语义信息。
  + 1. 元数据模式[SA\_MD\_TABMODEL]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_MD\_TABMODEL（元数据——模式）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 模式ID | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 或指向用户表 |
| 用户类型 | ownerType | NUMBER(1) | 非空 |  | 用户类型：1=用户；2=Session |
| 表名称 | tableName | VARCHAR2(40) | 非空 |  | 此名称与业务数据表名对应（注意是积累表） |
| 表描述名称 | titleName | VARCHAR2(100) | 可空 |  | 可能是页签（excel中的sheet）名称 |
| 模式说明 | descn | VARCHAR2(400) | 非空 |  | 可用此作为业务表的注解 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| **补充说明** | 1-这里的元数据需要明确与用户绑定，**与用户无关的元数据在D2中处理**  2-外键不建立，只是说明关系，弱外键；  3-对于某一个用户创建的元数据模式来说，有两类实例表：  A积累实例表：表名称就存储在本表的tableName中  B-临时实例表：每次导入数据时根据此元数据模式创建的临时表，在SA\_MD\_TABELMAP\_REL表中描述  4-这里的表名tableName是一个冗余字段，系统根据规则自动创建的。  5-积累表名可以从(**元数据实体表对应关系SA\_MD\_TABMAP\_REL**)表中计算得到  **Select tableName from sa\_md\_tabmap\_rel where tmId=this.id and tableType=1** | | | | |
| **索引** | 1-id，主键  2-onwerId/ownerTyp，是唯一索引  3-ownerId/ownerTyp /tableName，IDX\_TINFO\_oIDtNAME是唯一索引 | | | | |

* + 1. 元数据列描述[SA\_MD\_COLUMN]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_MD\_COLUMN（元数据——列描述）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 列ID | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 表模式ID | tmId | VARCHAR2(32) | 非空 |  | 表模式ID(外键) |
| 列名称 | columnName | VARCHAR2(10) | 非空 |  | 此名称作为表中列名 |
| 列意义名称  [中文名称] | titleName | VARCHAR2(100) | 可空 |  | 此名称根据导入数据来，若为excel则为表头名，若为DB，则比较复杂 |
| 列排序 | columnIndex | NUMBER (3) | 非空 | 默认值0 | 排序字段 |
| 列数据类型 | columnType | VARCHAR2(10) | 非空 | 默认String |  |
| 主键标识 | pkSign | NUMBER (1) | 非空 | 默认值0 | 0=不是主键；1=是确定主键；2=不确定主键 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| **补充说明** | 1-外键不建立，只是说明关系  2-这里的表名columnName是系统根据规则自动创建的  3-数据类型目前有如下：  A字符串(String)；B整型(Integer)；C浮点数(Double)；D时间(Time)；  4-一个表的键值列可以是多个； | | | | |
| **索引** | 1-id，主键；  2-tmId，IDX\_CINFO\_TID加快主子表关联；  3-tmId与columnName组成唯一索引、tmId与titleName组成唯一索引；这两个索引是逻辑所要求的 | | | | |

* + 1. 元数据列语义[SA\_MD\_COLSEMANTEME]

可能表也需要语义，这个以后再扩充。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_MD\_COLSEMANTEME（元数据列语义）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 列语义id | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 列描述Id | cId | VARCHAR2(32) | 非空 |  | 列描述ID(外键) |
| 元数据模式Id | tmId | VARCHAR2(10) | 非空 |  | 模式id，冗余外键 |
| 语义类型 | semantemeType | NUMBER (4) | 非空 |  | 1=身份证；2=字典项；… |
| 语义代码 | semantemeCode | VARCHAR2(32) | 非空 |  | 某列在确定语义类型后，对该类型的具体描述，目前只对字典项有意义 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| 最后修改 | lmTime | TIMESTAMP(4) | 非空 |  | 每次更新时修改 |
| **补充说明** | 1-此表中记录与SA\_MD\_COLUMN表中记录相对应，但不是必须的一一对应，如果表中某列能够分析出语义，则此表中就有该条记录。  2-列的语义有可能有多个，因此是一对多的关系。  3-其中元tmId可以从(**元数据列描述SA\_MD\_COLUMN**)表中计算得到  **Select tmId from sa\_md\_column where id =this.cId** | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 元数据表对照[SA\_MD\_TABMAP\_REL]

此表为元数据模式-实体表-实体表指标表的对照关系表，此表设计为了实现用户多次导入一个表的信息。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_MD\_TABMAP\_REL（元数据模式及实体表对应）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 对照表ID | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 元数据模式id | tmId | VARCHAR2(32) | 非空 |  | 表模式Id |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型1=用户；2=Session |
| 表名称 | tableName | VARCHAR2(40) | 非空 |  | 此名称与业务数据表名对应 |
| 表类型 | tableType | NUMBER (1) | 非空 | 默认1 | 1=积累表；2=临时表 |
| 表说明 | tableDescn | VARCHAR2(400) | 可空 |  | 表说明 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| **补充说明** | 1-对于用户创建的元数据模式来说，有两类实例表：  A积累实例表  B-临时实例表：每次导入数据时根据此元数据模式创建的临时表  实体表和元数据的对应关系都在此表中维护  2-这里的表名由系统根据规则自动创建的  3-ownerId由于SA\_MD\_TABLEMODEL中已经有这个信息了，因此是冗余信息，放在这里是便于查询，关系sql为  **Select ownerId from sa\_md\_tabmodel where id =this.tmId** | | | | |
| **索引** | 1-id，主键  2-ownerId/tableName，IDX\_TINFO\_oIDtNAME是唯一索引 | | | | |

* + 1. 实体表指标[SA\_MD\_TABQUOTA]

实体表指标——对表数据的统计信息

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_MD\_TABQUOTA（实体表指标）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 表指标ID | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 对照表Id | tmoId | VARCHAR2(32) | 非空 |  | 元数据对照表ID(外键) |
| 元数据模式Id | tmId | VARCHAR2(10) | 非空 |  | 模式id，冗余外键 |
| 表名称 | tableName | VARCHAR2(10) | 非空 |  | 表名称，冗余外键 |
| 表行数 | allCount | NUMBER (10)/long | 非空 | 默认值0 | 表中记录行数 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| 最后修改 | lmTime | TIMESTAMP(4) | 非空 |  | 每次更新时修改 |
| 最后访问修改 | laTime | TIMESTAMP(4) | 非空 |  | 最后访问时间 |
| **补充说明** | 1-目前表的指标项只有表行数有用，其他有用的指标信息可能今后要扩充  2-tableName是冗余信息，他与SA\_ MD\_TABMAP\_REL表中的tableName关联，有可能有重复的表名，因为此表中没有用户信息  3-tmId和tableName是冗余字段，其计算方法如下：  tmid= **select tmId from sa\_md\_tabmap\_rel where id=this.tomId**  tableName= **select tableName from sa\_md\_tabmap\_rel where id=this.tomId** | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 实体列指标[SA\_MD\_COLQUOTA]

实体列指标——对列数据的统计信息

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_MD\_COLQUOTA（实体表中列指标）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 列指标ID | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 列描述Id | cId | VARCHAR2(32) | 非空 |  | 列描述ID(外键) SA\_MD\_COLUMN |
| 表指标Id | tqId | VARCHAR2(32) | 非空 |  | 实体表指标ID(外键)  SA\_MD\_TABQUOTA |
| 最大值 | max | String | 非空 |  | 无论列是什么类型，在这里都要转换为字符串 |
| 最小值 | min | String | 非空 |  | 无论列是什么类型，在这里都要转换为字符串 |
| 空值数 | nullCount | NUMBER (10)/long | 非空 | 默认0 | 表中本列的空值数 |
| 单值数 | distinctCount | NUMBER (10)/long | 非空 | 默认0 | 表中本列的消重值个数 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 创建时的系统时间 |
| 最后修改 | lmTime | TIMESTAMP(4) | 非空 |  | 每次更新时修改，这个对于积累表有意义 |
| **补充说明** | 1-目前表的指标项只有表行数有用，其他有用的指标信息可能今后要扩充  2-tmId是冗余信息，他是元数据模式中的Id | | | | |
| **索引** | 1-id，主键； | | | | |

* 1. 数据导入类{1}
     1. ~~数据导入日志[SA\_IMP\_LOG]~~

~~文件上传日志，目前仅对excel有用~~

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **~~表名~~** | **~~SA\_IMP\_LOG（数据导入日志）~~** | | | | |
| **~~意义~~** | **~~字段名~~** | **~~数据类型~~** | **~~空否~~** | **~~约束~~** | **~~说明~~** |
| ~~日志id~~ | ~~id~~ | ~~VARCHAR2(32)~~ | ~~非空~~ | ~~主键PK~~ | ~~UUID~~ |
| ~~用户ID或SessionId~~ | ~~ownerId~~ | ~~VARCHAR2(32)~~ | ~~非空~~ |  | ~~可能指向用户表~~ |
| ~~用户类型~~ | ~~ownerType~~ | ~~NUMBER (1)~~ | ~~非空~~ |  | ~~user类型：1=用户2=Session~~ |
| ~~服务端文件名~~ | ~~sFileName~~ | ~~VARCHAR2(500)~~ | ~~非空~~ |  | ~~包括文件路径~~ |
| ~~客户端文件名~~ | ~~cFileName~~ | ~~NUMBER (1)~~ | ~~可空~~ |  | ~~包括文件路径~~ |
| ~~文件大小~~ | ~~fileSize~~ | ~~NUMBER(10)~~ | ~~非空~~ |  | ~~文件大小~~ |
| ~~创建时间~~ | ~~cTime~~ | ~~TIMESTAMP(4)~~ | ~~非空~~ |  | ~~也可作为上传时间~~ |
| **~~补充说明~~** |  | | | | |
| **~~索引~~** | ~~1-id，主键；~~ | | | | |

* + 1. 数据文件/实体表对应[SA\_IMP\_TABMAP\_REL]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_IMP\_TABMAP\_REL（数据导入日志）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件/实体表对应id | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 文件日志Id | fId | VARCHAR2(32) | 非空 |  | 文件表(外键) |
| 对照表Id | tmoId | VARCHAR2(32) | 非空 |  | 元数据实体表对照ID(外键)，只是积累表 |
| 元数据模式Id | tmId | VARCHAR2(32) | 非空 |  | 表模式ID(外键) |
| 页签名称 | sheetName | VARCHAR2(100) | 非空 |  | 页签名称 |
| 页签顺序 | sheetIndex | NUMBER(3) | 非空 |  | 页签排序 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 也可作为上传时间 |
| **补充说明** | 1-tmId是冗余字段，从对照表中也可得出此信息；  tmId= **select tmId from sa\_md\_tabmap\_rel where id=this.tomId**  **另：这里面会有积累表，若是积累表，则不表明是本次新加入的。** | | | | |
| **索引** | 1-id，主键； | | | | |

* 1. 文件管理{5=3+2}
     1. 文件记录索引[SA\_FILE\_INDEX]

文件记录索引表，用于记录系统生成的日志或分析结果信息。

一个文件只能在文件索引中出现一次。

**把文件管理类的功能扩充到数据导入，废弃掉原数据导入日志表。**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_FILE\_INDEX（文件记录索引）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件id | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表，也可能是sessionId，还可能是’Sys’表明是系统所有的 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型：  1=用户；2=Session；3=系统 |
| 文件访问类型 | accessType | NUMBER(1) | 非空 | 默认1 | 文件访问类型，如ftp,操作系统文件等，目前只支持1=操作系统文件 |
| 文件路径 | filePath | VARCHAR2(500) | 非空 |  | 文件路径，不包括文件名 |
| 文件名称 | fileName | VARCHAR2(100) | 非空 |  | 不包括路径，包括扩展名 |
| 文件扩展名 | fileExtName | VARCHAR2(30) | 非空 |  | 不包括路径 |
| 文件大小 | fileSize | NUMBER(10) | 非空 |  | 文件大小，字节数 |
| 说明 | descn | VARCHAR2(400) | 可空 |  | 说明 |
| 文件创建时间 | fcTime | TIMESTAMP(4) | 非空 |  | 文件创建时间 |
| 文件最后修改 | flmTime | TIMESTAMP(4) | 非空 |  | 文件最后修改时间 |
| 记录创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 记录创建时间 |
| **补充说明** | 1-一个文件只能在这张表中出现一次，可以有一下几个判断：  a)文件名+文件大小+修改时间——保证本系统文件唯一，此判断最精细，若能按字节比较更好；  b)访问类型+文件路径+“a)”——保证在文件的源存储系统唯一；  **由于目前这些文件都由系统生成，而且没有分布式存储的机制，以上约定搁置。**  **2-用户id和用户类型都加入了系统，这里有了些D2的概念。**  **3-**若ownerId不是userId或sessionId，可以表明这些数据是系统生成的，**这里的所有者信息表示文件的生成方，其他有关权限的数据再设计！！** | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 文件分类 [SA\_FILE\_CATEGORY]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_FILE\_CATEGORY（文件分类）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件关系id | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 文件id | fId | VARCHAR2(32) | 非空 |  | 主文件id，指向文件表 |
| 文件分类—大类 | type1 | VARCHAR2(10) | 非空 |  | 文件大类型：目前支持持三种,IMP、LOG和ANAL，即日志和分析，日志是文本文件，分析是jsonD格式 |
| 文件分类—中类 | type2 | VARCHAR2(60) | 非空 |  | 中类型，如分析中的元数据分析 |
| 文件分类—小类 | type3 | VARCHAR2(60) | 可空 |  | 小类型：如那类元数据分析 |
| 扩展信息 | extInfo | VARCHAR2(2000) | 可空 |  | 比如jsonD的一些说明，当为IMP时此信息为原文件名称，sFileName |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 关系创建时间 |
| **补充说明** | 1-一个文件在可以有多个分类属性，但至少有一个；  2-扩展信息的jsonD需要分别定义 | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 文件关系[SA\_FILE\_REL]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_FILE\_REL（文件关系）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件关系id | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 第一文件类型 | aType | NUMBER(1) | 非空 |  | 第一文件类型：=1是对原生态表的关联关系；=2是文件关联表 |
| 第一文件id | aId | VARCHAR2(32) | 非空 |  | 当aType=1指向文件表  当aType=2指向文件分类表 |
| 第二文件型 | bType0 | NUMBER(1) | 非空 |  | 第二文件类型：=1是对原生态表的关联关系；=2是文件关联表 |
| 第二文件id | bId | VARCHAR2(32) | 非空 |  | 当aType=1指向文件表  当aType=2指向文件分类表 |
| 关联类型1 | rType1 | NUMBER(1) | 非空 |  | 内部关联类型1：=1单向-说明rfid是fid的子；=0平等；=-1反向-说明fid是rfid的子(这个通过视图实现) |
| 关联类型2 | rType2 | VARCHAR2(200) | 非空 |  | 关联类型2 |
| 说明 | descn | VARCHAR2(400) | 可空 |  | 说明 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 关系创建时间 |
| **补充说明** | 1-文件关系可以建立原生态文件之间的关系，也可以建立文件分类之间的关系，所谓文件分类之间的关系是这样：可以把文件分类理解为对文件内容的一种语义概括，有些关系是当文件有了某种语义后才有的关系；  2-文件关联类型：若是单向，不能有循环，若是平等，可以有循环。如下图： | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 反向文件关系[vSA\_FILE\_INVERSEREL]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **视图名称** | **vSA\_FILE\_INVERSEREL（反向文件关系）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件关系id | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 第一文件类型 | aType | NUMBER(1) | 非空 |  | 第一文件类型：=1是对原生态表的关联关系；=2是文件关联表 |
| 第一文件id | aId | VARCHAR2(32) | 非空 |  | 当aType=1指向文件表  当aType=2指向文件分类表 |
| 第二文件型 | bType0 | NUMBER(1) | 非空 |  | 第二文件类型：=1是对原生态表的关联关系；=2是文件关联表 |
| 第二文件id | bId | VARCHAR2(32) | 非空 |  | 当aType=1指向文件表  当aType=2指向文件分类表 |
| 关联类型1 | rType1 | NUMBER(1) | 非空 |  | 内部关联类型1：=1单向-说明rfid是fid的子；=0平等；=-1反向-说明fid是rfid的子(这个通过视图实现) |
| 关联类型2 | rType2 | VARCHAR2(200) | 非空 |  | 关联类型2 |
| 说明 | descn | VARCHAR2(400) | 可空 |  | 说明 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 关系创建时间 |
| **补充说明** | 此视图是文件关系表的反向表，建立语句为：  **CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER**  **VIEW vsa\_file\_inverserel AS**  **select id, bId AS aId, bType AS aType, aId AS bId, aType AS bType,**  **(0-rType1) AS rType1, rType2, descn, cTime**  **from sa\_file\_rel;** | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 数据导入日志[vSA\_IMP\_LOG]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **视图名称** | **vSA\_IMP\_LOG（数据导入日志）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件id | id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型：  1=用户；2=Session；3=系统 |
| 文件访问类型 | accessType | NUMBER(1) | 非空 | 默认1 | 文件访问类型，如ftp,操作系统文件等，目前只支持1=操作系统文件 |
| 文件路径 | filePath | VARCHAR2(500) | 非空 |  | 包括文件路径 |
| 文件名称 | fileName | VARCHAR2(100) | 非空 |  | 不包括路径，包括扩展名 |
| 文件扩展名 | fileExtName | VARCHAR2(30) | 非空 |  | 不包括路径 |
| 文件大小 | fileSize | NUMBER(10) | 非空 |  | 文件大小，字节数 |
| 客户端文件名 | cFileName | NUMBER (1) | 可空 |  | 包括文件路径 |
| 表说明 | descn | VARCHAR2(400) | 可空 |  | 表说明 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 文件创建时间 |
| **补充说明** | 此试图是文件和文件分类的一个集合：  **CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER**  **VIEW vsa\_imp\_log AS**  **select id, bId AS aId, bType AS aType, aId AS bId, aType AS bType,**  **(0-rType1) AS rType1, rType2, descn, cTime**  **from sa\_file\_index a, sa\_file\_category b**  **where a.id=b.fid and b.type1='IMP';** | | | | |
| **索引** | 1-id，主键； | | | | |

**说明：**目前系统中对文件的管理有两部份：

1-用户上传的数据文件，在数据导入类中进行管理

2-系统生成的日志或分析结果的jsonD文件，在文件管理类中管理。

经过修改，系统中的文件进行了统一的管理。

* 1. 报告管理{2=1+1}
     1. 报告信息[SA\_REPORT\_INFO]

记录报告的信息，与vSA\_REPORT\_FILE很一致，是一对一的关系。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_REPORT\_INFO（报告信息）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 报告Id | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 任务组Id | taskGId | VARCHAR2(32) | 非空 |  | 任务组表(外键) |
| 报告文件Id | fId | VARCHAR2(32) | 非空 |  | 对应的报告文件id  文件表(外键)  文件是json格式 |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 或指向用户表 |
| 用户类型 | ownerType | NUMBER(1) | 非空 |  | 用户类型：1=用户；2=Session |
| 报告分类 | reportType | VARCHAR2(100) | 可空 |  | 报告的分类，目前没有明确的分类方法 |
| 报告名称 | reportName | VARCHAR2(100) | 非空 |  | 报告名称 |
| 报告说明 | desc | VARCHAR2(500) | 非空 |  | 报告说明 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 也可作为上传时间 |
| **补充说明** | 1-tasksId、fId是外键 | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 报告文件信息[vSA\_REPORT\_FILE]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **视图名称** | **vSA\_REPORT\_FILE（报告文件信息）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 文件id | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型：  1=用户；2=Session；3=系统 |
| 文件访问类型 | accessType | NUMBER(1) | 非空 | 默认1 | 文件访问类型，如ftp,操作系统文件等，目前只支持1=操作系统文件 |
| 文件路径 | filePath | VARCHAR2(500) | 非空 |  | 包括文件路径 |
| 文件名称 | fileName | VARCHAR2(100) | 非空 |  | 不包括路径，包括扩展名 |
| 文件扩展名 | fileExtName | VARCHAR2(30) | 非空 |  | 不包括路径 |
| 文件大小 | fileSize | NUMBER(10) | 非空 |  | 文件大小，字节数 |
| 报告ID | reportId  b.type2 | VARCHAR2(32) | 非空 |  | 指向报告表的id |
| 任务组ID | tasksId  b.type3 | VARCHAR2(32) | 非空 |  | 指向任务组的id |
| 说明 | descn  b.extInfo | VARCHAR2(400) | 可空 |  | 表说明 |
| 创建时间 | cTime | TIMESTAMP(4) | 非空 |  | 文件创建时间 |
| **补充说明** | 此试图是文件和文件分类的一个集合：  **CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER**  **VIEW vsa\_report\_file AS**  **select a.id, ownerId, ownerType, accessType, filePath, fileName, fileExtName, fileSize,**  **b.type2 reportId, b.type3 tasksId,b.extInfo descn, a.cTime**  **from sa\_file\_index a, sa\_file\_category b**  **where a.id=b.fid and b.type1='REPORT';**  **注意：这里的信息内容和report表基本一致** | | | | |
| **索引** | 1-id，主键； | | | | |

* 1. 任务管理{4=3+1}

任务相关信息的持久化记录。

任务是和具体工作相关的，在本模型中工作即任务组，一项工作通常是有多个有相互依赖关系的单个任务组成的。

任务：为了任务能够单独执行，记录任务执行的语言环境，任务执行的具体方法，任务执行所需的参数，在本框架中，任务执行所需的参数只有两类：数据表和jsonD存储的数据，另外任务执行的结果是存储在jsonD中的。

任务的依赖关系，只记录任务的前序任务，实际是一个图（数据结构）。

**另：**

1-所有任务/工作都是由系统处理的，这里的所有者信息为今后的计数管理打下基础。

2-任务组与任务之间是主子表的关系

* + 1. 任务组信息[SA\_TASK\_GROUP]

也可以理解为一个工作。是一些列相互依赖的独立任务的组合

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_TASK\_GROUP（任务组信息）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 任务组Id | Id | VARCHAR2(32) | 非空 | 主键PK | UUID |
| 报告Id | reportId | VARCHAR2(32) | 可空 |  | 报告信息表(外键) |
| 用户ID或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表，也可能是sessionId，还可能是’Sys’表明是系统所有的 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型：  1=用户；2=Session；3=系统 |
| 工作状态 | status | NUMBER (1) | 非空 |  | 1=准备执行；*2=正在执行；*  3=执行成功；4=执行失败；  5=任务失效； |
| 工作名称 | workName | VARCHAR2(100) | 可空 |  | 工作的名称 |
| 说明 | descn | VARCHAR2(400) | 可空 |  | 说明 |
| 任务组开启时间 | beginTime | TIMESTAMP(4) | 非空 |  | 实际就是记录的创建时间 |
| **补充说明** | 1-reportId任务不一定对应一个报告，因此报告id可为空，并不是严格意义的外键  2-关于时间：任务组的创建时间就是任务组开始执行的时间，任务组子任务的执行时间在任务表中记录，这里只是记录一个大概。  3-这里的所有者是为了今后发展计数系统做准备的  4-关于任务组状态：在数据表中的状态用斜体字表示，以和程序运行时的状态做区分（运行时状态不会写入数据库中，用天蓝字表示）  *1)准备执行：任务组信建立后就是这个状态；*  2)正在执行(运行时状态)：任务组信息放入任务框架缓存后，就视为任务组正在执行；  *3)执行成功：所有子任务都执行成功；*  *4)执行失败：有任何子任务执行失败，视为任务组执行失败；*  *5)任务失效：任务组由于一些原因已没有执行的意义时为此状态，如：过长时间没有执行，多次执行失败等；这样的任务组的所有子任务不会被执行；任何一个子任务失效，则任务组也失效* | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 任务信息[SA\_TASK\_INFO]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_TASK\_INFO（任务信息）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 任务Id | Id | VARCHAR2(32) | 非空 | 主键 | UUID |
| 任务组Id | taskGId | VARCHAR2(32) | 非空 |  | 任务组表(外键) |
| jsonD文件Id | resultFileId | VARCHAR2(32) | 可空 |  | 任务若有执行结果，则结果一定存储在jsonD文件中，此Id指向文件分类Id |
| 任务名称 | taskName | VARCHAR2(100) | 非空 |  | 任务名称 |
| 任务类型 | taskType | VARCHAR2(50) | 非空 |  | 任务类型，任务功能的分类信息 |
| 执行语言环境 | langType | VARCHAR2(100) | 非空 |  | 任务执行语言：目前只有Java |
| 执行方法 | executeFunc | VARCHAR2(200) | 非空 |  | 任务执行方法，要实现一个接口 |
| 任务执行参数 | Param | VARCHAR2(2000) | 非空 |  | 任务执行所需的参数，用json处理 |
| 任务状态 | Status | NUMBER (1) | 非空 |  | 1=准备执行；*2=正在执行；*  3=任务成功；4=任务失效；  5=执行失效；*6=等待执行；* |
| 任务执行次数 | executeCount | NUMBER(2) | 非空 | 0 | 任务执行次数 |
| 任务说明 | Descn | VARCHAR2(400) | 可空 |  | 说明 |
| 第一次执行时间 | firstTime | TIMESTAMP(4) | 非空 |  | 第一次放入执行队列的时间 |
| 本次执行开始时间 | beginTime | TIMESTAMP(4) | 非空 |  | 开始执行时间 |
| 本次执行结束时间 | endTime | TIMESTAMP(4) | 非空 |  | 结束执行时间 |
| **补充说明** | 1-关于执行结果result：若任务有执行结果，则这个字段有值，否则这个字段可以为空，为空时说明任务执行的无结果（如只是对原有数据的调整）或不生成新的数据。  2-关于时间：第一次放入执行对列的时间为firstTime，任务每次真正执行都会跟新beginTime，执行成功后会更新endTime  3-关于参数，参数支持多个输入，而且输入有不同的参数类型，目前参数有三类：  A)元数据——可能对元数据的统计信息进行处理  B)表——对表进行处理（可以指明是那一列）  C)具体的JsonD文件比如：   |  | | --- | | {  {“pType”:”metadata”, “mId”:”32425234”},  {“pType”:”table”, “tableName”:”plat\_user”},  {“pType”:”jsonD”, “jsonD”:”c:\aabc.jsond”}  } |   4-关于任务状态：在数据表中的状态用斜体字表示，以和程序运行时的状态做区分（运行时状态不会写入数据库中，用天蓝字表示）  *1)准备执行：任务信建立后就是这个状态；*  2)正在执行(运行时状态)：任务真正在线程中开始执行后，就视为任务正在执行；  *3)执行成功：任务执行成功；*  *4)执行失败：任务执行失败；*  *5)任务失效：任务由于一些原因已没有执行的意义时为此状态，如：过长时间没有执行，多次执行失败等；*  6)等待执行(运行时状态)：任务信息放入任务框架缓存后，就视为任务准备执行； | | | | |
| **索引** | 1-id，主键； | | | | |

* + 1. 任务关系[SA\_TASK\_REL]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **表名** | **SA\_TASK\_REL（任务关系[图关系，有向图]）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 关系Id | Id | VARCHAR2(32) | 非空 | 主键 | UUID |
| 任务Id | taskId | VARCHAR2(32) | 非空 | 外键 | 任务表(外键) |
| 前序任务Id | preTaskId | VARCHAR2(32) | 非空 | 外键 | 一个任务可以有多个前置任务，若一个任务没有前置任务，则不再这张表中存数据。 |
| 是否利用  前序数据 | usedPreData | NUMBER(1) | 非空 | 默认值2 | 是否利用前序任务的数据1-利用;2-不利用。  只有当preTaskId有值时此字段才有意义 |
| **补充说明** | 任务关系是一个有向图，如下：任务C，D的前置任务都是A和B | | | | |
| **索引** |  | | | | |

* + 1. 任务全信息[vSA\_TASKS]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **视图名称** | **vSA\_TASKS（任务全信息）** | | | | |
| **意义** | **字段名** | **数据类型** | **空否** | **约束** | **说明** |
| 主任务Id | MT\_Id | VARCHAR2(32) | 非空 | 主键 | UUID |
| 主任务  jsonD文件Id | MT\_RFile | VARCHAR2(32) | 可空 |  | 任务若有执行结果，执行结果文件，直接是访问的URI |
| 主任务  执行语言环境 | MT\_langType | VARCHAR2(100) | 非空 |  | 任务执行语言：  目前只有Java |
| 住任务执行方法 | MT\_excuteFunc | VARCHAR2(200) | 非空 |  | 任务执行方法，要实现一个接口 |
| 主任务  任务执行参数 | MT\_param | VARCHAR2(500) | 非空 |  | 任务执行所需的参数，用json处理 |
| 主任务状态 | MT\_status | NUMBER (1) | 非空 |  | 1=准备执行；*2=正在执行；*  3=任务成功；4=任务失效；  5=执行失效；*6=等待执行；* |
| 主任务名称 | MT\_tName | VARCHAR2(100) | 非空 |  | 任务名称 |
| 主任务类型 | MT\_tType | VARCHAR2(50) | 非空 |  | 任务类型 |
| 主任务说明 | MT\_descn | VARCHAR2(400) | 可空 |  | 说明 |
| 主任务时间 | MT\_firstTime | TIMESTAMP(4) | 非空 |  | 第一次放入执行队列的时间 |
| MT\_beginTime | TIMESTAMP(4) | 非空 |  | 开始执行时间 |
| MT\_endTime | TIMESTAMP(4) | 非空 |  | 结束执行时间 |
| 任务组Id | TG\_Id | VARCHAR2(32) | 非空 |  | UUID |
| 报告Id | reportId | VARCHAR2(32) | 可空 |  | 任务组表(外键) |
| 用户ID  或SessionId | ownerId | VARCHAR2(32) | 非空 |  | 可能指向用户表，也可能是sessionId，还可能是’Sys’表明是系统所有的 |
| 用户类型 | ownerType | NUMBER (1) | 非空 |  | 用户类型：  1=用户；2=Session；3=系统 |
| 工作状态 | TG\_status | NUMBER (1) | 非空 |  | 1=准备执行；2=正在执行；  3=执行成功；4=执行失败  5=任务失效； |
| 工作名称 | workName | VARCHAR2(100) | 非空 |  | 工作的名称 |
| 说明 | TG\_descn | VARCHAR2(400) | 可空 |  | 说明 |
| 任务开启时间 | TG\_beginTime | TIMESTAMP(4) | 非空 |  | 实际就是记录的创建时间 |
| 前置任务Id | PT\_Id | VARCHAR2(32) | 非空 | 主键 | UUID |
| 前置任务  jsonD文件Id | PT\_RFile | VARCHAR2(32) | 可空 |  | 任务若有执行结果，执行结果文件，直接是访问的URI |
| 前置任务  执行语言环境 | PT\_langType | VARCHAR2(100) | 非空 |  | 任务执行语言：  目前只有Java |
| 前置任务执行方法 | PT\_excuteFunc | VARCHAR2(200) | 非空 |  | 任务执行方法，要实现一个接口 |
| 前置任务  任务执行参数 | PT\_param | VARCHAR2(500) | 非空 |  | 任务执行所需的参数，用json处理 |
| 前置任务状态 | PT\_status | NUMBER (1) | 非空 |  | 1=准备执行；2=正在执行；  3=执行成功；4=执行失败  5=任务失效；6=等待执行 |
| 前置任务名称 | PT\_tName | VARCHAR2(100) | 非空 |  | 任务名称 |
| 前置任务类型 | PT\_tType | VARCHAR2(50) | 非空 |  | 任务类型 |
| 前置任务说明 | PT\_descn | VARCHAR2(400) | 可空 |  | 说明 |
| 前置任务时间 | PT\_firstTime | TIMESTAMP(4) | 非空 |  | 第一次放入执行队列的时间 |
| PT\_beginTime | TIMESTAMP(4) | 非空 |  | 开始执行时间 |
| PT\_endTime | TIMESTAMP(4) | 非空 |  | 结束执行时间 |
| **补充说明** | 此试图是有关任务的全信息：  **CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER**  **VIEW vsa\_tasks AS (**  **select t.\*, concat(sf.filePath, '/', sf.fileName) PT\_RFile from (**  **select**  **m.MT\_Id, m.MT\_RFile, m.MT\_langType, m.MT\_excuteFunc, m.MT\_param, m.MT\_status, m.MT\_tName, m.MT\_tType, m.MT\_descn, m.MT\_firstTime, m.MT\_beginTime, m.MT\_endTime,**  **m.TG\_Id, m.reportId, m.ownerId, m.ownerType, m.TG\_status, m.workName, m.TG\_descn, m.TG\_beginTime,**  **p.id PT\_id, p.resultFileid, p.langType PT\_langType, p.excuteFunc PT\_excuteFunc, p.param PT\_param, p.status PT\_status, p.taskName PT\_tName, p.taskType PT\_tType, p.descn PT\_descn, p.firstTime PT\_firstTime, p.beginTime PT\_beginTime, p.endTime PT\_endTime**  **from (**  **select**  **a.id MT\_Id, concat(d.filePath, '/', d.fileName) MT\_RFile,a.langType MT\_langType, a.excuteFunc MT\_excuteFunc, a.param MT\_param,**  **a.status MT\_status, a.taskName MT\_tName, a.taskType MT\_tType, a.descn MT\_descn, a.firstTime MT\_firstTime, a.beginTime MT\_beginTime, a.endTime MT\_endTime,**  **c.id TG\_Id, c.reportId, c.ownerId, c.ownerType, c.status TG\_status, c.workName, c.descn TG\_descn, c.beginTime TG\_beginTime,**  **b.preTaskId, b.usedPreData**  **from sa\_task\_info a**  **inner join sa\_task\_rel b on a.id=b.taskId**  **left join sa\_task\_group c on a.taskGId=c.id**  **left join sa\_file\_index d on a.resultFileid=d.id**  **) m**  **left join sa\_task\_info p on p.id=m.preTaskId**  **) t**  **left join sa\_file\_index sf on t.resultFileid=sf.id**  **);**  **注意：**  1-文件访问目前不考虑其他的属性，直接是URI  2-除了主任务，其他都是外连接，可以为空  3-由于mysql不支持子查询视图，因此以上的视图需要进行拆分创建。两个准备视图  vpSA\_TASKS1:   |  | | --- | | CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER  VIEW vpsa\_tasks1 AS ( select  a.id MT\_Id, concat(d.filePath, '/', d.fileName) MT\_RFile,a.langType MT\_langType, a.excuteFunc MT\_excuteFunc, a.param MT\_param,  a.status MT\_status, a.taskName MT\_tName, a.taskType MT\_tType,a.descn MT\_descn, a.firstTime MT\_firstTime, a.beginTime MT\_beginTime, a.endTime MT\_endTime,  c.id TG\_Id, c.reportId, c.ownerId, c.ownerType, c.status TG\_status, c.workName, c.descn TG\_descn, c.beginTime TG\_beginTime,  b.preTaskId, b.usedPreData  from sa\_task\_info a  inner join sa\_task\_rel b on a.id=b.taskId  left join sa\_task\_group c on a.taskGId=c.id  left join sa\_file\_index d on a.resultFileid=d.id  ); |   vpSA\_TASKS2:   |  | | --- | | CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER  VIEW vpsa\_tasks2 AS (  select  m.MT\_Id, m.MT\_RFile, m.MT\_langType, m.MT\_excuteFunc, m.MT\_param, m.MT\_status, m.MT\_tName, m.MT\_descn, m.MT\_firstTime, m.MT\_beginTime, m.MT\_endTime,  m.TG\_Id, m.reportId, m.ownerId, m.ownerType, m.TG\_status, m.workName, m.TG\_descn, m.TG\_beginTime,  p.id PT\_id, p.resultFileid, p.langType PT\_langType, p.excuteFunc PT\_excuteFunc, p.param PT\_param, p.status PT\_status, p.taskName PT\_tName, p.taskType PT\_tType, p.descn PT\_descn,  p.firstTime PT\_firstTime, p.beginTime PT\_beginTime, p.endTime PT\_endTime  from vpsa\_tasks1 m  left join sa\_task\_info p on p.id=m.preTaskId  ); |   **vSA\_TASKS: 正式视图**   |  | | --- | | CREATE OR REPLACE ALGORITHM=UNDEFINED SQL SECURITY DEFINER  VIEW vsa\_tasks AS (  select t.\*, concat(sf.filePath, '/', sf.fileName) PT\_RFile  from vpsa\_tasks2 t  left join sa\_file\_index sf on t.resultFileid=sf.id  ); | | | | | |
| **索引** |  | | | | |

1. 数据逻辑视图



说明：这里只是第一阶段功能主要部分，关于语义部分，应该还需要设计一些存储结构。

1. 数据库管理与维护说明