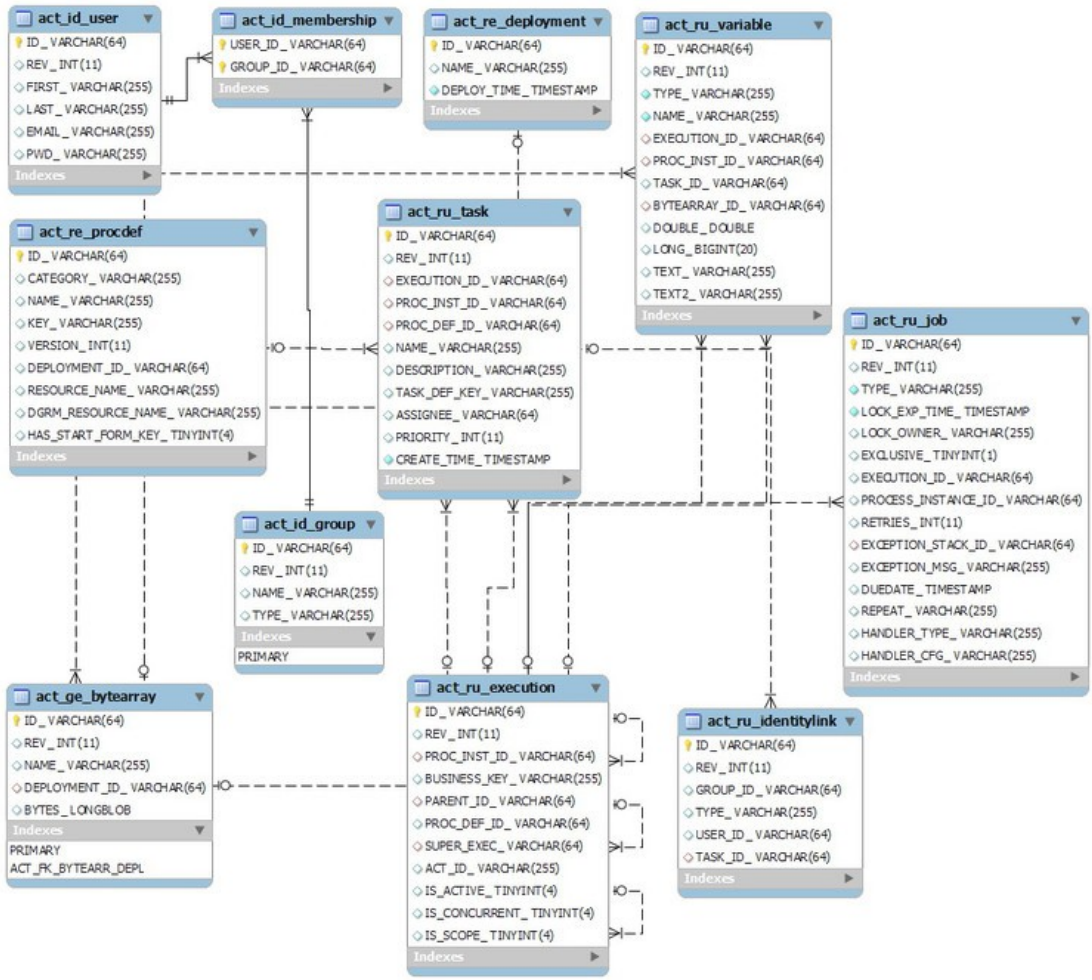
Activiti 工作流引擎数据库表结构

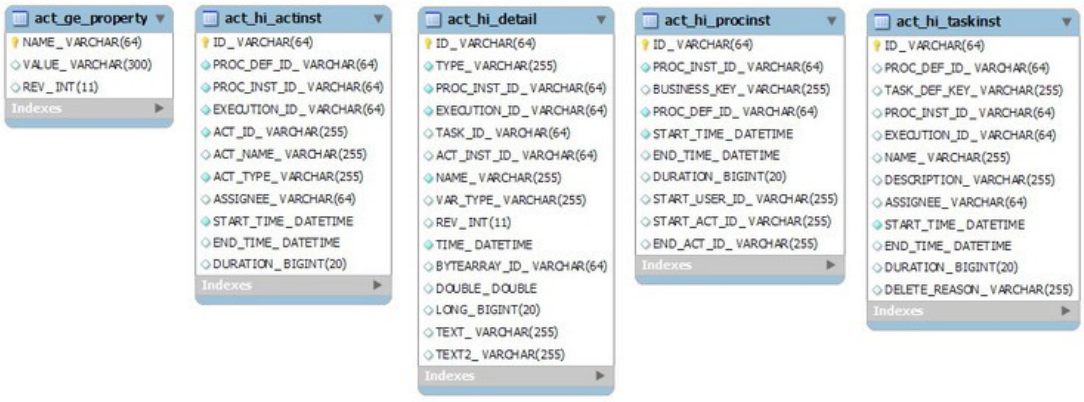
数据库表的命名

Acitiviti 数据库中表的命名都是以 ACT_开头的。第二部分是一个两个字符用例表的标识。此用例大体与服务 API 是匹配的。

- ACT_RE_*:’RE’表示 repository。带此前缀的表包含的是静态信息，如，流程定义，流程的资源（图片，规则等）。
- ACT_RU_*:’RU’表示 runtime。这是运行时的表存储着流程变量，用户任务，变量，职责（job）等运行时的数据。Activiti 只存储实例执行期间的运行时数据，当流程实例结束时，将删除这些记录。这就保证了这些运行时的表小且快。
- ACT_ID_*:’ID’表示 identity。这些表包含标识的信息，如用户，用户组，等等。
- ACT_HI_*:’HI’表示 history。就是这些表包含着历史的相关数据，如结束的流程实例，变量，任务，等等。
- ACT_GE_*:普通数据，各种情况都使用的数据。

数据库表结构图





数据库表结构说明

- ACT_GE_PROPERTY:属性数据表。存储这个流程引擎级别的数据。

1. NAME_:属性名称
2. VALUE_:属性值
3. REV_INT:版本号

- ACT_GE_BYTEARRAY:用来保存部署文件的大文本数据
  1. ID_:资源文件编号，自增长
  2. REV_INT:版本号
  3. NAME_:资源文件名称
  4. DEPLOYMENT_ID_:来自于父表 ACT_RE_DEPLOYMENT 的主键
  5. BYTES_:大文本类型，存储文本字节流
- ACT_RE_DEPLOYMENT:用来存储部署时需要持久化保存下来的信息
  1. ID_:部署编号，自增长
  2. NAME_:部署包的名称
  3. DEPLOY_TIME_:部署时间
- ACT_RE_PROCDEF:业务流程定义数据表
  1. ID_:流程 ID，由“流程编号：流程版本号：自增长 ID”组成
  2. CATEGORY_:流程命名空间（该编号就是流程文件 targetNamespace 的属性值）
  3. NAME_:流程名称（该编号就是流程文件 process 元素的 name 属性值）
  4. KEY_:流程编号（该编号就是流程文件 process 元素的 id 属性值）
  5. VERSION_:流程版本号（由程序控制，新增即为 1，修改后依次加 1 来完成的）

6. DEPLOYMENT_ID_:部署编号
7. RESOURCE_NAME_:资源文件名称
8. DGRM_RESOURCE_NAME_:图片资源文件名称
9. HAS_START_FROM_KEY_:是否有 Start From Key

注：此表和 ACT_RE_DEPLOYMENT 是多对一的关系，即，一个部署的 bar 包里可能包含多个流程定义文件，每个流程定义文件都会有一条记录在 ACT_REPROCDEF 表内，每个流程定义的数据，都会对于 ACT_GE_BYTEARRAY 表内的一个资源文件和 PNG 图片文件。和 ACT_GE_BYTEARRAY 的关联是通过程序用 ACT_GE_BYTEARRAY.NAME 与 ACT_RE_PROCDEF.NAME_完成的，在数据库表结构中没有体现。

- ACT_ID_GROUP:用来存储用户组信息。
  1. ID_：用户组名*
  2. REV_INT:版本号
  3. NAME_:用户组描述信息*
  4. TYPE_:用户组类型
- ACT_ID_MEMBERSHIP:用来保存用户的分组信息
  1. USER_ID_:用户名
  2. GROUP_ID_:用户组名
- ACT_ID_USER:
  1. ID_:用户名
  2. REV_INT:版本号

3. FIRST_:用户名称
4. LAST_:用户姓氏
5. EMAIL_:邮箱
6. PWD_:密码

- ACT_RU_EXECUTION：
  1. ID_：
  2. REV_：版本号
  3. PROC_INST_ID_：流程实例编号
  4. BUSINESS_KEY_：业务编号
  5. PARENT_ID_：找到该执行实例的父级，最终会找到整个流程的执行实例
  6. PROC_DEF_ID_：流程 ID
  7. SUPER_EXEC_： 引用的执行模板
  8. ACT_ID_： 节点 id
  9. IS_ACTIVE_： 是否访问
  10. IS_CONCURRENT_：
  11. IS_SCOPE_：
- ACT_RU_TASK：运行时任务数据表。
  1. ID_：
  2. REV_：
  3. EXECUTION_ID_： 执行实例的 id
  4. PROC_INST_ID_： 流程实例的 id

5. PROC_DEF_ID_： 流程定义的 id,对应 act_re_procdef 的 id_
6. NAME_： 任务名称，对应 ***task 的 name
7. PARENT_TASK_ID_：对应父任务
8. DESCRIPTION_：
9. TASK_DEF_KEY_： ***task 的 id
10. OWNER_ ：发起人
11. ASSIGNEE_： 分配到任务的人
12. DELEGATION_：委托人
13. PRIORITY_： 紧急程度
14. CREATE_TIME_： 发起时间
15. DUE_TIME_：审批时长

- ACT_RU_IDENTITYLINK：任务参与者数据表。主要存储当前节点参与者的信息。
  1. ID_： 标识
  2. REV_： 版本
  3. GROUP_ID_： 组织 id
  4. TYPE_： 类型
  5. USER_ID_： 用户 id
  6. TASK_ID_： 任务 id
- ACT_RU_VARIABLE：运行时流程变量数据表。
  1. ID_：标识
  2. REV_：版本号

3. TYPE_：数据类型
4. NAME_：变量名
5. EXECUTION_ID_： 执行实例 id
6. PROC_INST_ID_： 流程实例 id
7. TASK_ID_： 任务 id
8. BYTEARRAY_ID_：
9. DOUBLE_：若数据类型为 double ,保存数据在此列
10. LONG_： 若数据类型为 Long 保存数据到此列
11. TEXT_： string 保存到此列
12. TEXT2_：

- ACT_HI_PROCINST：
  1. ID_：唯一标识
  2. PROC_INST_ID_：流程ＩD
  3. BUSINESS_KEY_：业务编号
  4. PROC_DEF_ID_ ： 流程定义 id
  5. START_TIME_： 流程开始时间
  6. ENT__TIME： 结束时间
  7. DURATION_：流程经过时间
  8. START_USER_ID_：开启流程用户 id
  9. START_ACT_ID_：开始节点
  10. END_ACT_ID_： 结束节点
  11. SUPER_PROCESS_INSTANCE_ID_：父流程流程 id

12. DELETE_REASON_：从运行中任务表中删除原因

- ACT_HI_ACTINST：
    1. ID_：标识
    2. PROC_DEF_ID_:流程定义 id
    3. PROC_INST_ID_：流程实例 id
    4. EXECUTION_ID_：执行实例
    5. ACT_ID_：节点 id
    6. ACT_NAME_：节点名称
    7. ACT_TYPE_：节点类型
    8. ASSIGNEE_：节点任务分配人
    9. START_TIME_：开始时间
    10. END_TIME_：结束时间
    11. DURATION：经过时长
- ACT_HI_TASKINST：
    1. ID_ ： 标识
    2. PROC_DEF_ID_ ： 流程定义 id
    3. TASK_DEF_KEY_：任务定义 id
    4. PROC_INST_ID_： 流程实例ｉｄ
    5. EXECUTION_ID_：执行实例 id
    6. PARENT_TASK_ID_：父任务 id
    7. NAME_：任务名称
    8. DESCRIPTION_：说明

9. OWNER_： 拥有人（发起人）
10. ASSIGNEE_ : 分配到任务的人
11. START__TIME_ : 开始任务时间
12. END_TIME_ : 结束任务时间
13. DURATION_ : 时长
14. DELETE_REASON_ :从运行时任务表中删除的原因
15. PRIORITY_ : 紧急程度
16. DUE_DATE_ :

- ACT_HI_DETAIL：启动流程或者在任务 complete 之后,记录历史流程变量
  1. ID_ : 标识
  2. TYPE_ : variableUpdate 和 formProperty 两种值
  3. PROC_INST_ID_ : 对应流程实例 id
  4. EXECUTION_ID_ : 对应执行实例 id
  5. TASK_ID_ : 对应任务 id
  6. ACT_INST_ID : 对应节点 id
  7. NAME_ : 历史流程变量名称，或者表单属性的名称
  8. VAR_TYPE_ : 定义类型
  9. REV_ : 版本
  10. TIME_ : 导入时间
  11. BYTEARRAY_ID_
  12. DOUBLE_ : 如果定义的变量或者表单属性的类型为 double，他的值存在这里

13. LONG_：如果定义的变量或者表单属性的类型为 LONG ,他的值存在这里
14. TEXT_： 如果定义的变量或者表单属性的类型为 string，值存在这里
15. TEXT2_:

- ACT_HI_COMMENT 意见表
  1. ID_ :标识
  2. TYPE_ : 意见记录类型 为 comment 时 为处理意见
  3. TIME_ : 记录时间
  4. USER_ID_ :
  5. TASK_ID_ ： 对应任务的 id
  6. PROC_INST_ID_ : 对应的流程实例的 id
  7. ACTION_ ： 为 AddComment 时为处理意见
  8. MESSAGE_ : 处理意见
  9. FULL_MSG_ :

结论及总结

- 流程文件部署主要涉及到 3 个表，分别是：ACT_GE_BYTEARRAY、ACT_RE_DEPLOYMENT、ACT_RE_PROCDEF。主要完成“部署包”-->“流程定义文件”-->“所有包内文件”的解析部署关系。从表结构中可以看出，流程定义的元素需要每次从数据库加载并解析，因为流程定义的元素没有转化成数据库表来完成，当然流程元素解析后是放在缓存中的，具体的还需要后面详细研究。

- 流程定义中的 java 类文件不保存在数据库里 。
- 组织机构的管理相对较弱，如果要纳入单点登录体系内还需要改造完成，具体改造方法有待研究。
- 运行时对象的执行与数据库记录之间的关系需要继续研究
- 历史数据的保存及作用需要继续研究。

Activiti 使用 Mybatis3 做持久化工作，可以在配置中设置流程引擎启动时创建表。

Activiti 使用到的表都是 ACT_开头的。

ACT_RE_*:流程定义存储。

ACT_RU_*:流程执行记录，记录流程启动到结束的所有动作，流程结束后会清除相关记录。

ACT_ID_*:用户记录，流程中使用到的用户和组。

ACT_HI_*:流程执行的历史记录。

ACT_GE_*:通用数据及设置。

使用到的表：

ACT_GE_BYTEARRAY：流程部署的数据。

ACT_GE_PROPERTY：通用设置。

ACT_HI_ACTINST：流程活动的实例。

ACT_HI_ATTACHMENT：

ACT_HI_COMMENT：

ACT_HI_DETAIL：

ACT_HI_PROCINST：流程实例。

ACT_HI_TASKINST：任务实例。

ACT_ID_GROUP：用户组。

ACT_ID_INFO：

ACT_ID_MEMBERSHIP：

ACT_ID_USER：用户。

ACT_RE_DEPLOYMENT：部署记录。

ACT_RE_PROCDEF：流程定义。

ACT_RU_EXECUTION：流程执行记录。

ACT_RU_IDENTITYLINK：

ACT_RU_JOB：

ACT_RU_TASK：执行的任务记录。

ACT_RU_VARIABLE：执行中的变量记录。

activiti-administrator

自带的用户管理系统，维护用户和组，需要配置数据连接参数，在 activiti-administrator\WEB-INF\applicationContext.xml 中，并加入 JDBC 驱动包。

activiti-cycle

PVM 活动检测的，由 activiti-rest 提供服务，不需配置。

activiti-explorer

可以查看用户任务和启动流程，由 activiti-rest 提供服务，不需配置。

activiti-kickstart

简单的点对点流程定义维护工具，需要配置数据连接，把 activiti.cfg.xml 文件放在 classes 下，并加入驱动包。

activiti-modeler

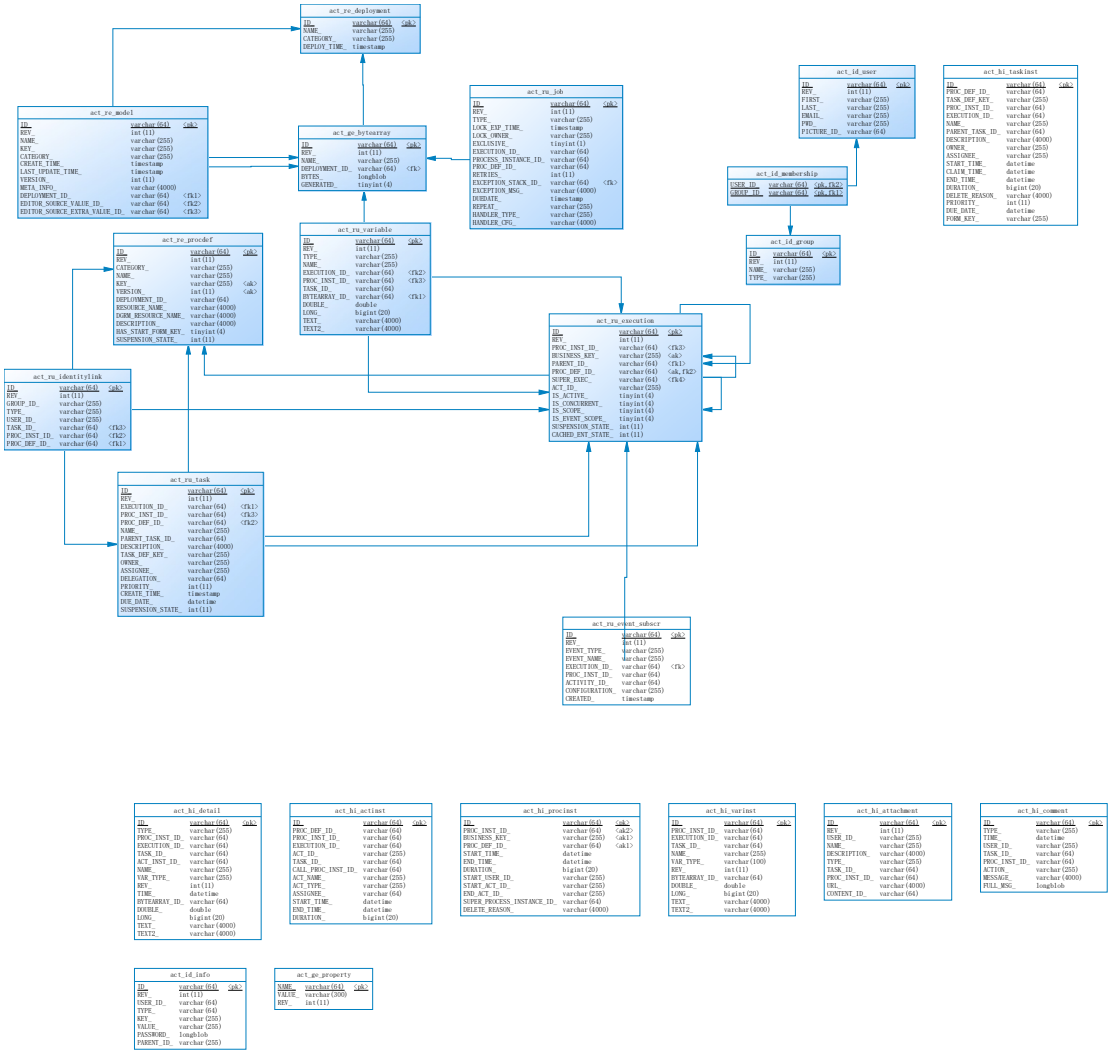
在线编辑和维护流程定义的工具，最后以文件夹方式部署，需要配置 activiti-modeler\WEB-INF\classes\configuration.properties 文件。

activiti-probe

PVM 的观测服务，由 activiti-rest 提供服务，不需配置，可以查看 deployment、processdefinition、processinstance、database。

activiti-rest

其他几个应用的服务提供者，需要配置数据连接，把 activiti.cfg.xml 文件放在 classes 下，并加入驱动包。


act_re_deployment
ID_ varchar(64) <pk>
NAME_ varchar(255)
CATEGORY_ varchar(255)
DEPLOY_TIME_ timestamp
act_re_model
ID_ varchar(64) <pk>
REV int(11)
NAME_ varchar(255)
KEY_ varchar(255)
CATEGORY_ varchar(255)
CREATE_TIME_ timestamp
LAST_UPDATE_TIME_ timestamp
VERSION_ int(11)
META_INFO_ varchar(4000)
DEPLOYMENT_ID_ varchar(64) <fk1>
EDITOR_SOURCE_VALUE_ID_ varchar(64) <fk2>
EDITOR_SOURCE_EXTRA_VALUE_ID_ varchar(64) <fk3>
act_ge_bytearray
ID_ varchar(64) <pk>
REV_ int(11)
NAME_ varchar(255)
DEPLOYMENT_ID_ varchar(64) <fk>
BYTES_ longblob
GENERATED_ tinyint(4)
act_ru_job
ID_ varchar(64) <pk>
REV_ int(11)
TYPE_ varchar(255)
LOCK_EXP_TIME_ timestamp
LOCK_OWNER_ varchar(255)
EXCLUSIVE_ tinyint(1)
EXECUTION_ID_ varchar(64)
PROCESS_INSTANCE_ID_ varchar(64)
PROC_DEF_ID_ varchar(64)
RETRIES_ int(11)
EXCEPTION_STACK_ID_ varchar(64) <fk>
EXCEPTION_MSG_ varchar(4000)
DUEDATE_ timestamp
REPEAT_ varchar(255)
HANDLER_TYPE_ varchar(255)
HANDLER_CFG_ varchar(4000)
act_id_user
ID_ varchar(64) <pk>
REV_ int(11)
FIRST_ varchar(255)
LAST_ varchar(255)
EMAIL_ varchar(255)
PWD_ varchar(255)
PICTURE_ID_ varchar(64)
act_hi_taskinst
ID_ varchar(64) <pk>
PROC_DEF_ID_ varchar(64)
TASK_DEF_KEY_ varchar(255)
PROC_INST_ID_ varchar(64)
EXECUTION_ID_ varchar(64)
NAME_ varchar(255)
PARENT_TASK_ID_ varchar(64)
DESCRIPTION_ varchar(4000)
OWNER_ varchar(255)
ASSIGNEE_ varchar(255)
START_TIME_ datetime
CLAIM_TIME_ datetime
END_TIME_ datetime
DURATION_ bigint(20)
DELETE_REASON_ varchar(4000)
PRIORITY_ int(11)
DUE_DATE_ datetime
FORM_KEY_ varchar(255)
act_id_membership
USER_ID_ varchar(64) <pk, fk2>
GROUP_ID_ varchar(64) <pk, fk1>
act_id_group
ID_ varchar(64) <pk>
REV_ int(11)
NAME_ varchar(255)
TYPE_ varchar(255)
act_re_procdef
ID_ varchar(64) <pk>
REV int(11)
CATEGORY_ varchar(255)
NAME_ varchar(255)
KEY_ varchar(255) <ak>
VERSION_ int(11) <ak>
DEPLOYMENT_ID_ varchar(64)
RESOURCE_NAME_ varchar(4000)
DGRM_RESOURCE_NAME_ varchar(4000)
DESCRIPTION_ varchar(4000)
HAS_START_FORM_KEY_ tinyint(4)
SUSPENSION_STATE_ int(11)
act_ru_variable
ID_ varchar(64) <pk>
REV_ int(11)
TYPE_ varchar(255)
NAME_ varchar(255)
EXECUTION_ID_ varchar(64) <fk2>
PROC_INST_ID_ varchar(64) <fk3>
TASK_ID_ varchar(64)
BYTEARRAY_ID_ varchar(64) <fk1>
DOUBLE_ double
LONG_ bigint(20)
TEXT_ varchar(4000)
TEXT2_ varchar(4000)
act_ru_execution
ID_ varchar(64) <pk>
REV_ int(11)
PROC_INST_ID_ varchar(64) <fk3>
BUSINESS_KEY_ varchar(255) <ak>
PARENT_ID_ varchar(64) <fk1>
PROC_DEF_ID_ varchar(64) <ak, fk2>
SUPER_EXEC_ varchar(64) <fk4>
ACT_ID_ varchar(255)
IS_ACTIVE_ tinyint(4)
IS_CONCURRENT_ tinyint(4)
IS_SCOPE_ tinyint(4)
IS_EVENT_SCOPE_ tinyint(4)
SUSPENSION_STATE_ int(11)
CACHED_ENT_STATE_ int(11)
act_ru_identitylink
ID_ varchar(64) <pk>
REV int(11)
GROUP_ID_ varchar(255)
TYPE_ varchar(255)
USER_ID_ varchar(255)
TASK_ID_ varchar(64) <fk3>
PROC_INST_ID_ varchar(64) <fk2>
PROC_DEF_ID_ varchar(64) <fk1>
act_ru_task
ID_ varchar(64) <pk>
REV_ int(11)
EXECUTION_ID_ varchar(64) <fk1>
PROC_INST_ID_ varchar(64) <fk3>
PROC_DEF_ID_ varchar(64) <fk2>
NAME_ varchar(255)
PARENT_TASK_ID_ varchar(64)
DESCRIPTION_ varchar(4000)
TASK_DEF_KEY_ varchar(255)
OWNER_ varchar(255)
ASSIGNEE_ varchar(255)
DELEGATION_ varchar(64)
PRIORITY_ int(11)
CREATE_TIME_ timestamp
DUE_DATE_ datetime
SUSPENSION_STATE_ int(11)
act_ru_event_subscr
ID_ varchar(64) <pk>
REV_ int(11)
EVENT_TYPE_ varchar(255)
EVENT_NAME_ varchar(255)
EXECUTION_ID_ varchar(64) <fk>
PROC_INST_ID_ varchar(64)
ACTIVITY_ID_ varchar(64)
CONFIGURATION_ varchar(255)
CREATED_ timestamp
act_hi_detail
ID_ varchar(64) <pk>
TYPE_ varchar(255)
PROC_INST_ID_ varchar(64)
EXECUTION_ID_ varchar(64)
TASK_ID_ varchar(64)
ACT_INST_ID_ varchar(64)
NAME_ varchar(255)
VAR_TYPE_ varchar(255)
REV_ int(11)
TIME_ datetime
BYTEARRAY_ID_ varchar(64)
DOUBLE_ double
LONG_ bigint(20)
TEXT_ varchar(4000)
TEXT2_ varchar(4000)
act_hi_actinst
ID_ varchar(64) <pk>
PROC_DEF_ID_ varchar(64)
PROC_INST_ID_ varchar(64)
EXECUTION_ID_ varchar(64)
ACT_ID_ varchar(255)
TASK_ID_ varchar(64)
CALL_PROC_INST_ID_ varchar(64)
ACT_NAME_ varchar(255)
ACT_TYPE_ varchar(255)
ASSIGNEE_ varchar(64)
START_TIME_ datetime
END_TIME_ datetime
DURATION_ bigint(20)
act_hi_procinst
ID_ varchar(64) <pk>
PROC_INST_ID_ varchar(64) <ak2>
BUSINESS_KEY_ varchar(255) <ak1>
PROC_DEF_ID_ varchar(64) <ak1>
START_TIME_ datetime
END_TIME_ datetime
DURATION_ bigint(20)
START_USER_ID_ varchar(255)
START_ACT_ID_ varchar(255)
END_ACT_ID_ varchar(255)
SUPER_PROCESS_INSTANCE_ID_ varchar(64)
DELETE_REASON_ varchar(4000)
act_hi_varinst
ID_ varchar(64) <pk>
PROC_INST_ID_ varchar(64)
EXECUTION_ID_ varchar(64)
TASK_ID_ varchar(64)
NAME_ varchar(255)
VAR_TYPE_ varchar(100)
REV_ int(11)
BYTEARRAY_ID_ varchar(64)
DOUBLE_ double
LONG_ bigint(20)
TEXT_ varchar(4000)
TEXT2_ varchar(4000)
act_hi_attachment
ID_ varchar(64) <pk>
REV_ int(11)
USER_ID_ varchar(255)
NAME_ varchar(255)
DESCRIPTION_ varchar(4000)
TYPE_ varchar(255)
TASK_ID_ varchar(64)
PROC_INST_ID_ varchar(64)
URL_ varchar(4000)
CONTENT_ID_ varchar(64)
act_hi_comment
ID_ varchar(64) <pk>
TYPE_ varchar(255)
TIME_ datetime
USER_ID_ varchar(255)
TASK_ID_ varchar(64)
PROC_INST_ID_ varchar(64)
ACTION_ varchar(255)
MESSAGE_ varchar(4000)
FULL_MSG_ longblob
act_id_info
ID_ varchar(64) <pk>
REV_ int(11)
USER_ID_ varchar(64)
TYPE_ varchar(64)
KEY_ varchar(255)
VALUE_ varchar(255)
PASSWORD_ longblob
PARENT_ID_ varchar(255)
act_ge_property
NAME_ varchar(64) <pk>
VALUE_ varchar(300)
REV_ int(11)