目录

[R12WEBADI设置过程讲解-Integratorsetup 2](#_Toc355423492)

[R12WEBADI设置过程讲解-Interfacesetup 3](#_Toc355423493)

[R12WEBADI设置过程讲解-Contentsetup 6](#_Toc355423494)

[R12 webadi导入程序设置 9](#_Toc355423495)

[R12WEBADI删除API 10](#_Toc355423496)

[R12WEBADI的一个小技巧 11](#_Toc355423497)

[R12WEBADI建立不了的解决办法 11](#_Toc355423498)

[R12WEBADI基于TABLE的LOV详细研究 12](#_Toc355423499)

[R12WEBADI的参数设置研究之一:设置CreateDocumentParameters 14](#_Toc355423500)

[R12WEBADI中如何建立DATE字段的LOV验证 20](#_Toc355423501)

[R12WEBADI中建立基于KeyFlexfields的LOV验证 20](#_Toc355423502)

[R12WEBADI如何建立基于table的LOV验证 21](#_Toc355423503)

[R12WEBADI如何在template中自动生成sequence字段 24](#_Toc355423504)

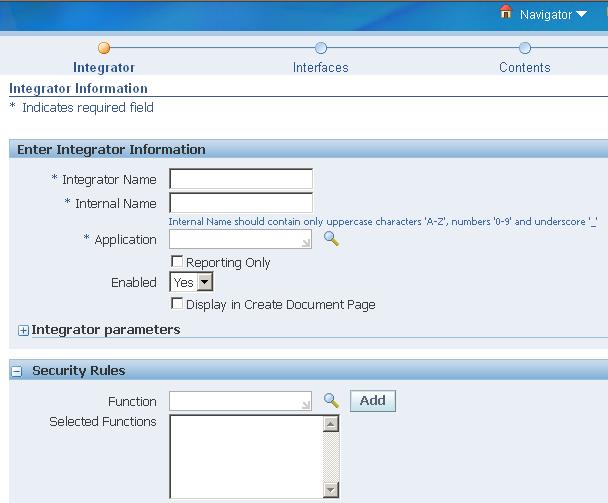
[R12WEBADI-关于ADI securityrules 25](#_Toc355423505)

[在WEBADI中实现组织屏蔽 25](#_Toc355423506)

[R12WEBADI表结构 27](#_Toc355423507)

## R12WEBADI设置过程讲解-Integratorsetup

这是建立ADI的第一步，需要设置的东西如下:

[](http://space.itpub.net/batch.download.php?aid=34551)

**Integrator Name:**随便写，如果有naming standard,要遵守

**Internal Name:**即Integrator code,这里不宜输入太长的code,建议10个以内，以保证后面生成其它对象的code不会超长，因为恶心的oracle会在自定义的对象code后面加东西，比如这个integrator code,如果你输入CAPINV,存到表里后是CAPINV\_XINTG,无故多出6个字符。

**Application:**一般选客制化专用的application

**Reporting Only:**仅用于下载数据的Integrator,如果是upload的，这项万万不能选

**Enabled:**Yes,必须的

**Display in Create Docuement Page:**因为Integrator要提供给user使用的话，有两种途径，一种是为这个Integrator单独建个form.function,然后加入到user的menu中；还有一种就是把标准的Create Document功能提供给user,而user能在里面找到这个Integrator就必须选上这项.一般来说我们会用第一种方式，所以这项一般不用选.

**Integrator Parameters:**目前我只测了Document Parameters,另两个不知干什么用的，可恶的oracle也没文档.

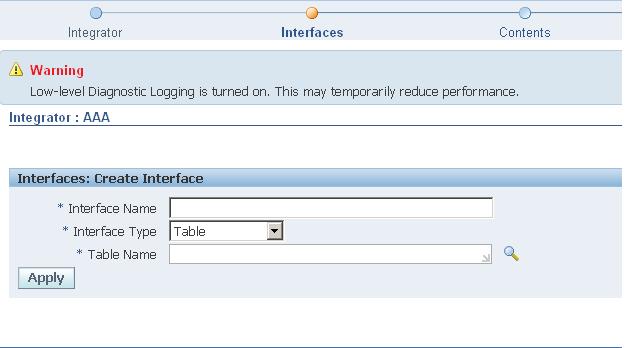
**Security Rules:**如果这个Profile：BNEAllowNoSecurityRule设成No的话，这项要设，可选个user肯定有的function或自已建个新的sub function，做为security function,不过如果是自已建的function的话，需要另外加入user的menu中，否则user无法使用这个Integrator.R12WEBADI这里支持多个security function,但目前我用的这个版本的ADI只能选一个，不知是bug还是其它原因。

这步设置对应的后台表为bne\_integrators\_b和bne\_integrators\_tl.

客制化的Integrator后台表bne\_integrators\_b中的source为'C',因此我们可以在manager职责中修改，如果不想让人改，把C删了；同样，如果想修改oracle标准的Integrator,可把source改为C,这样就可以修改了，不过不建议修改，因为升级不支持。不过改成C后，你可以直观的在manager职责中看到它的详细的setup,比在后台看方便一点.

## R12WEBADI设置过程讲解-Interface setup

这是建立一个新的Integrator的第二步，也是最关键的一步!



**Interface Name:**随便写，反正**[oracle](javascript:;" \t "_self)**会生成一个interface code,而这个才是我们需要的，而我们需要的，oracle偏偏不让我们看到，而且还加东西。。。生成的code应该是integrator code+INTF1之类的

**Interface Type:**主要是**[Table](javascript:;" \t "_self)**和API Procedure.那么什么时候用Table,什么时候用Procedure呢?下面是我个人的意见，仅供参考:

1.如果validation特别简单(都是单行单字段的校验),且template上的数据要进到某一张表里的话，用table.

2.如果validation比较复杂(可能包含了多行组合校验，比如订单行金额是否等于头金额；多字段组合校验，比如某个vendor,某个vendor site下不能有这种currency的invoice之类的；duplicate校验，比如某个asset,在base table或interface里已经有了，不能再用tempate导一遍之类的，因为这些校验要写**[java](javascript:;" \t "_self)** class,我不知怎么写(可恶的oracle也没文档).所以只能用Procedure去做.

3.如果template的数据仅仅用来传给标准API，用来做某种处理，不用进到某张表的话，当然不要用table,而是用Procedure.

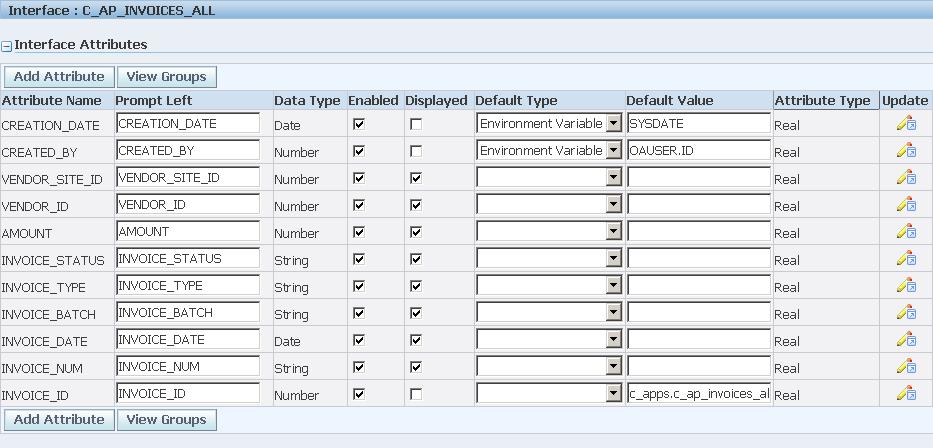
**Table Name:**表名，此表需要注册在database里，否则无法识别.

**Package Name:**不用说了，你写的upload package的名字

**Procedure Name:**Upload Procedure的名字

**API returns:**Procedure/Function传出的参数，一般选fnd message code,用来接收procedure中raise出来的error.

Apply一下:

[](http://space.itpub.net/batch.download.php?aid=34568)

**Attribute Name:**其实就是table的字段或Procedure的传入参数名.

**Prompt Left:**这个要改一下，一般格式是"Invoice Number"这样的，你可以只为要放到layout header上的字段改，因为行上的字段只有Prompt above,没有left.

**[Data](javascript:;" \t "_self)Type:**从table或procedure继承下来的

**Enabled:**此项如果不选的话，该字段即使在template可以显示，并且可以填东西，但数据不会参与upload.即此字段完全无用了(适合一些无需填值的字段).

**Displayed:**此项如果不选的话，在定义layout时，将无法看到此项.也就无法在template中看到。当然，就算你这选了，但layout中没选此项，也是无法看到的.

**Default Type/Default Value:**用来设定字段的默认值，在11i里我一般是在layout里定义的，但R12好象主要在这里定义了，layout中的default反而不用了，而且我在layout中定义好象也老出问题，不知是bug,还是其它原因.

**Environment Parameters:**OAUSER.ID/SYSDATE/DATABASE常用的，另外一些sob的，例如sob.chart of account sid,sob.period setname,sob.setofbooksid之类的，也不知还能不能用，还没测

**Constant:**这个最简单，直接写个东西进去就行了。

**[SQL](javascript:;" \t "_self):**这个也简单，写个SQL完事，结果多行的话，取第一行。sql里可包含$env$.userid,$env$.appid,$env$.respid之类的环境变量，还有$PROFILES$.default\_org\_id之类的取profileoptions的东东。

**Profile Option:**从字面上理解，是用来取配置文件中某个option的值，但我测了，不成功，不知是bug还是别的原因.

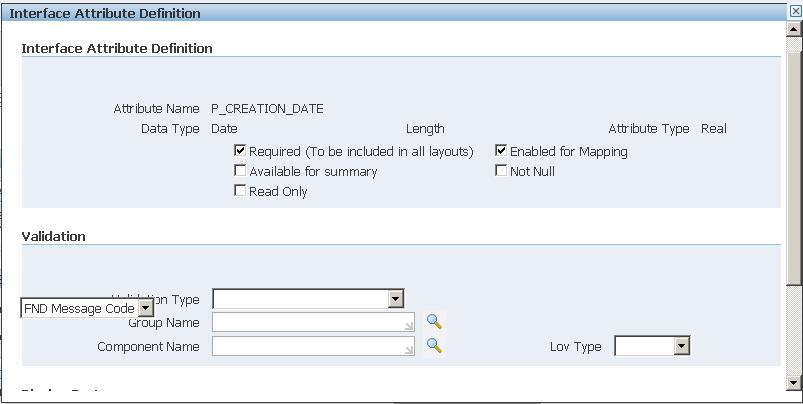
**Lookups:**不知干什么的，前两天无意看到有个标准的Integrator具然包含多个Interface,而Interface间的联接似乎就用这个。。。因为此项在后台叫TABLELOOKUP...多Interface的Integrator还不知怎么做，待研究.

**ExcelFormula:**这项不需要选default type,直接在default value中输入公式，格式如下：=R[]C[]\*R[]C[]之类的，比如=RC[-1]\*RC[-2],即表示此字段的值为左边两列的乘积.

**Sequence:**此项需后台修改表，default type为SEQUENCE\_EACH\_ROW,default value为sequence的名字，如果不在apps下的话，需要在apps下建同义词

**Attribute Type:**Real表明此字段为真实的表字段或过程参数，而Custom的是我们另外建的字段，一般用来做占位列（比如一个LOV显示的是DESC,存的是ID,这个DESC一般就放在这种列里）或只读显示项，比如显示当前的database信息.

通过点后面的update，可以修改字段的一些属性.

[](http://space.itpub.net/batch.download.php?aid=34596)

**DataType:**不需改，从table或procedure中继承下来的

**Length:**大家发现这项没有，因为这是基于procedure的interface,而对于procedure来说，所有的传入参数都是没有length的，而如果基于table的话，则会有length.而没有length的结果就是excel无法做字段宽度的校验，你就需要在procedure中写这样的语句:

If length(p\_vendor\_name)>50 then

raise\_application\_error(-20001,'Length exceeds max chars');

End if;

这样明显增加了开发的量，所以对于procedure的interface,我们可以通过直接修改后台表来设定字段的length.

**Attribute Type:**无需修改

**Required:**此项很重要，设定字段为必填字段，如果你在template上没输的话，excel校验会报错，这样的话，你就不用在procedure中写这样的语句了:

If p\_vendor\_name is null then

raise\_application\_error(-20001,'You have to enter a Vendor name');

endif;

**Enabled for Mapping:**设定该字段是否可建mapping,当你的Integrator中含有Content时，这项有用，一般来说，要在layout中显示的字段这项要选，其它可选可不选。

**Available for summary:**这项不知道，字面看是是否允许汇总。没测过

Not Null:这项似乎和required功能一样啊

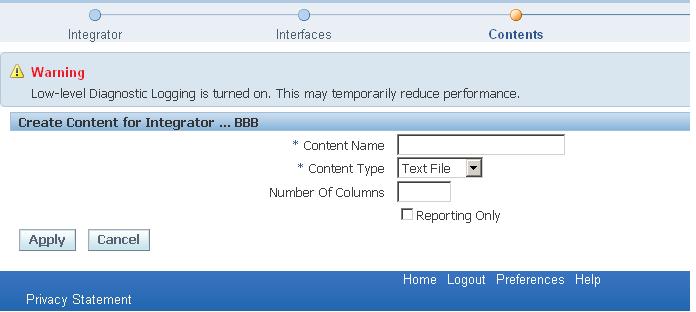
Read Only:设定某字段为只读字段，一般适用于Custom类型的显示字段.或不需用户修改的有默认值的字段.

下面的是建validation的部分，这里可建tableLOV,dateLOV,KFF/DFFLOV,及java类型的LOV.

最后一部分是设Promptabove,对于line上的字段，此项要写一下。UserHint最好也写一下，必填字段的一般为\*Text,\*Date,\*Number这样的，非必填的把\*去了。有LOV的一般为\*List-Text或List-Text之类的.

## R12WEBADI设置过程讲解-Contentsetup

这是建立一个新的Integrator的第三步，当然，不是必须要设置的一步。Content的作用是在user下载template的时候，可以把本机的准备好的**[data](javascript:;" \t "_self)**file传给这个template,这样的话，download下来的template就是一个有数据的template,也就不需要user输入或copy数据了.

[](http://space.itpub.net/batch.download.php?aid=34588)

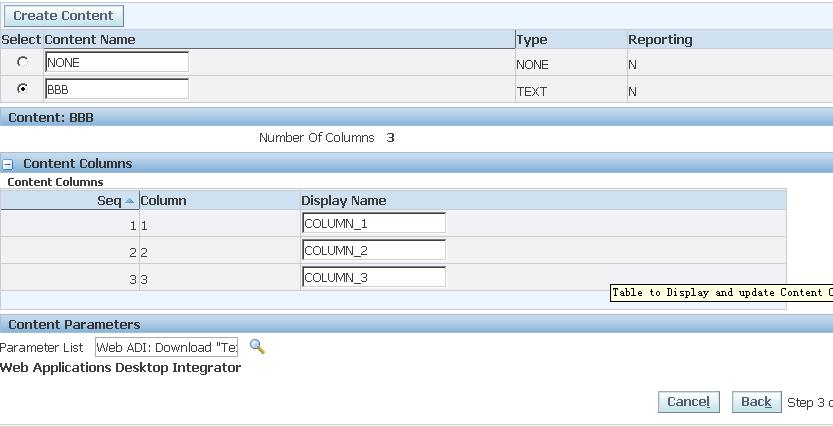
**ContentName:**随便写

**ContentType:**主要是TextFile和**[SQL](javascript:;" \t "_self)**Query,TextFile很明显是user需要提供一个datafile,而SQL则是直接从系统中取数据到template中。我们先看TextFile的。

**NumberOfColumns:**根据需要确定，一般来说，你的layout有多少列，这是就写多少，与layout一一对应。也可少于layout的列，仅针对部分列传入。但最好不要多于layout列。

**ReportingOnly:**万万不能选，此项仅用于下载数据.

Apply一下：

[](http://space.itpub.net/batch.download.php?aid=34589)

**ContentColumns**这是我选的是3列，DisplayName可改可不改，改了也就是为了后面建mapping的时候看的清楚点，没什么实际用途.

**ContentParameters:**此项很重要，一定要有个parameterlist,这里ADI已经预置了一个，我们直接拿来用即可：

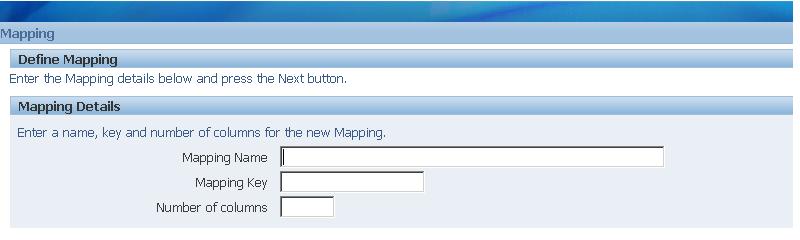
**Parameterlistname:**WebADI:Download"TextFile"Parameters

**Application:**WebApplicationDesktopIntegrator

Sumbit修改，然后开始建Mapping.Mapping的功能是建立Content字段与Interface字段的映射关系，也就是datafile的列与layout列的映射，告诉ADIdatafile里的每一列分别导到template里的哪一列上.Mapping的建立过程很简单，不过有一个要特别注意的，在第二步我们设定interface的时候，点update修改字段属性时，有一项叫做**EnabledforMapping**,这项如果不选的话，该字段就无法与Content建立映射。

DefineMapping

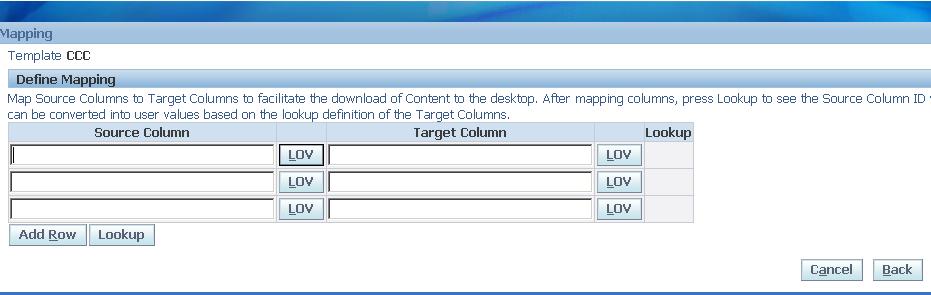
选择相应的Integrator后，系统会查询有没有建好的Mapping,如没有的话，点definemapping新建一个

[](http://space.itpub.net/batch.download.php?aid=34590)

**MappingName:**随便写

**MappingKey:**Unique的，而且要大写

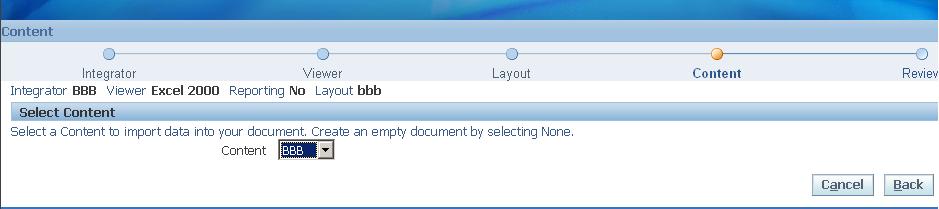
**NumberofColumns:**与Content的列数一致

[](http://space.itpub.net/batch.download.php?aid=34593)

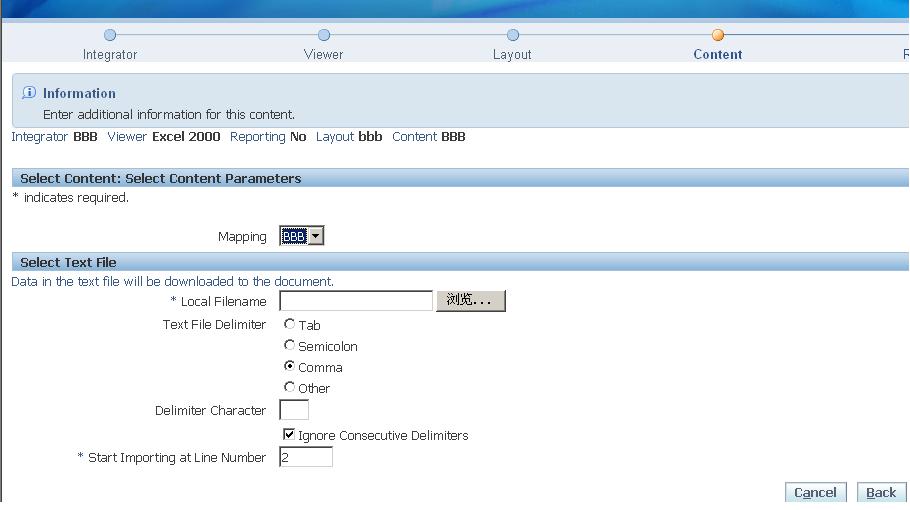
左边选Content列，右边选对应的Interface列。注意，一定要选，手工输会有问题。

Submit修改.

现在来Preview一下Integrator

[](http://space.itpub.net/batch.download.php?aid=34594)

Donwloadtemplate时，有一步会让我们选Content,选刚建好的这个.

[](http://space.itpub.net/batch.download.php?aid=34595)

出现ContentParameter的界面，

LocalFilename:选择本机中已经准备好的数据文件，如果多于3列的话，后面的列会忽略，ADI只读前3列.

TextFileDelimiter:列分隔符，按数据文件中的实际分隔符来选，一般为逗号或Tab.

DelimiterCharacter:上面如果选Other的话，需要在这手工输入分隔符

IgnoreConsecutiveDelimiters:连续分隔符是否忽略，比如文件中有这样的格式：

AAA,BBB,,,CCC,ADI会认为是3列，如不选的话，ADI会认为是5列.

StartImportingatLineNumber:如果数据文件有标题行的话，选2,否则输1.

Next后downloadtemplate,你会发现下载的template已经有数据了.

## R12 webadi导入程序设置

如果你导入的数据不是插入到系统标准的接口表，或者插入到接口表之后要手动提交一个请求，才能将接口表里的数据导入到ERP，则需要设置导入程序。

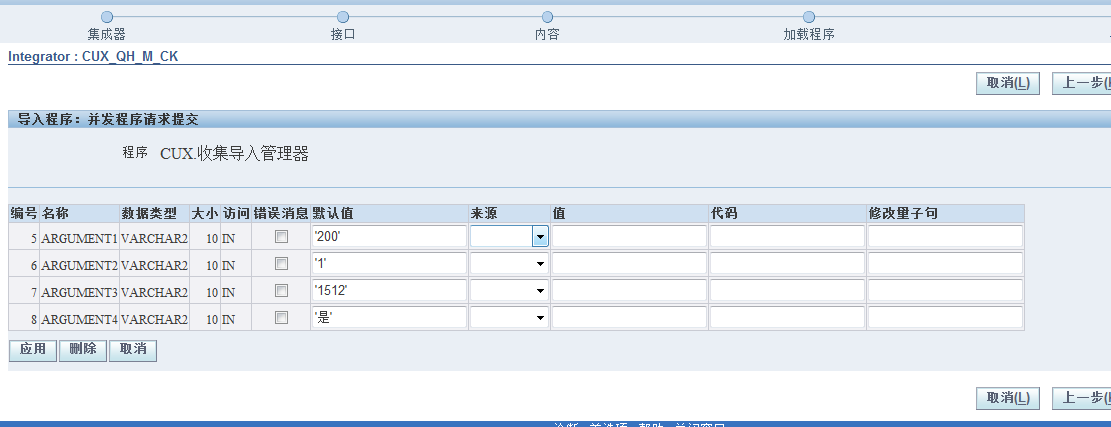
导入程序是指你一次性导入几十上百条数据之后，再一次性跑请求，将这些数据导入到ERP正式表，千万不要把跑请求的代码写到接口程序里，那样会导致导入非常慢。

1. 选择导入程序类型

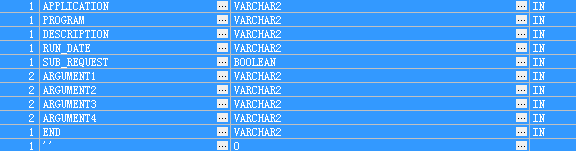


一般选择“异常并发请求”或者 PL/SQL API，个人觉得把跑请求的代码写到一个PL/SQL里，可能会方便，这里假定选择“异步并发请求”。

1. 设置并发程序



不知道是BUG还是什么原因，在这里输入之后，还要在bne\_attributes表中后台更新



1，2，3，4，5行数据是系统自动加进去的，实际就是调用的

fnd\_request.submit\_request(application =>,

program =>,

description =>,

start\_time =>,

sub\_request =>,

argument1 =>,

argument2 =>,

argument3 =>

.....)

，要特别留意的是，参数如果是文本，要给值加上引号，‘200’

1. 组定义：组定义的意思是分几次跑请求，一般在组织ID上定义一个组，这样相同的组织就会跑一次。

## R12WEBADI删除API

R12 WEBADI 删除一个WEBADI不要在界面上删除，这样会还有很多冗余数据还在后台表里，要通过bne\_integrator\_utils.DELETE\_INTEGRATOR 这个API去删，这样会把跟这样WEBADI有关的接口、内容、导入程序、加载程序、参数等全都删掉。

## R12WEBADI的一个小技巧

现遇到这么一个情况，如果多个用户同时往一张表里导数据，为了解决并发问题，有时我们需要为每个用户的template数据生成一个唯一的batchnumber，用来在表里区分各自的数据。有三种办法:

1.最简单的，用createdby来区分，但如果两个user用同一个account去导的话，这种区分会失败.

2.用一个autosequence来生成batchnumber,这个虽然是唯一的，也可以区分，但又有个问题，这个sequence一般我们会在downloadtemplate的时候生成在某个字段的默认值里，如果用户保存了这个template，隔段时间再打开，这个sequence仍是旧的值，这就又产生了risk,如果表里正好也有旧的batch的数据还没处理，那用户用这个旧的模块上传新的数据就会产生问题，两批数据会混在一起。当然，这是很极端的情况了.

3.用EXCEL公式来解决，这是目前我发现的最好的办法。公式如下:

=VALUE(TEXT(NOW(),"YYYYMMDDHHMMSS"))

可在template里建个number类型的字段，放在context中，readonly,把上述公式放在interface的defaultvalue中，这样的话，不管用户是下载，还是保存后重新打开，这个值取的都是当前的日期时间(已转成数字).而这个时间的重复可能性基本为0.

另，每次导入时取当前的SESSIONID做为BATCHNUMBER也不失为一个好办法！

## R12WEBADI建立不了的解决办法

如果在建WEBADI时，浏览器一直在运行，但是就是怎么都进不去，可能是某个人LOCK了这个表，可以通过后台查是不是BNE打头的表被锁定了，可以通过SQL强制解锁

|  |
| --- |
| 如果进不了WEBADI，首先通过后台查哪些表锁住了，再将锁死的表删掉  select sess.sid,  sess.serial#,  lo.oracle\_username,  lo.os\_user\_name,  ao.object\_name,  lo.locked\_mode  from v$locked\_object lo,  dba\_objects ao,  v$session sess  where ao.object\_id = lo.object\_id and lo.session\_id = sess.sid;  解锁  alter system kill session '73,31'; **--以下几个为相关表** SELECT \* FROM v$lock; SELECT \* FROM v$sqlarea; SELECT \* FROM v$session; SELECT \* FROM v$process ; SELECT \* FROM v$locked\_object; SELECT \* FROM all\_objects; SELECT \* FROM v$session\_wait; |

如果能进去，但是一建就报错，可能就是你在删除WEBADI时，有数据没删干净，特别要留意以下几个表中的数据

|  |
| --- |
| SELECT \* FROM BNE\_PARAM\_LISTS\_TL WHERE APPLICATION\_ID = 20003;  SELECT \* FROM BNE\_PARAM\_LISTS\_B WHERE APPLICATION\_ID = 20003;  SELECT \* FROM BNE\_PARAM\_LIST\_ITEMS WHERE APPLICATION\_ID = 20003;  SELECT \* FROM BNE\_ATTRIBUTES WHERE APPLICATION\_ID = 20003; |

## R12WEBADI基于TABLE的LOV详细研究

R12WEBADI提供了基于Table的LOV的功能，很方便，但由于LOV的显示或值的提取与component的定义有很大关系，所以特做了几个例子验证一下。

LOV的定义可以在interface的定义界面上做（但component需另外定义），也可以使用API去做，相比较而言，我较倾向于用API.另外，如用API定义时，不提供component的参数的话，WEBADI会依据LOV的相关信息为我们建立默认的component,但我发现，结果总是差强人意，因此建议，component还是显式定义为好，不要让WEBADI自动建.

有一点不好的是，用API创建LOV时，依据参数建立的component的默认源是ORACLE,这样的话，我们无法在managercomponents的WEB页面中修改这个component(除非你改componentcode,好象在前面加\_XX\_，但这样还要修改bne\_interface\_cols\_b中的componentcode,很麻烦，因此最好直接后台修改component的信息(在bne\_param\_list\_items\_b表中),不要动componentcode).

**假设**

Template上有三个字段，vendor\_id,vendor\_num,vendor\_name,要求LOV建在vendor\_num上，选择后，提取id,num,name分别到三个字段中。

**API代码:**

begin

bne\_integrator\_utils.CREATE\_TABLE\_LOV(P\_APPLICATION\_ID=>20003

,P\_INTERFACE\_CODE=>'A1\_XINTG\_INTF1'

,P\_INTERFACE\_COL\_NAME=>'VENDOR\_NUM'

--以下为LOV定义参数

,P\_ID\_COL=>'VENDOR\_ID'

,P\_MEAN\_COL=>'SEGMENT1'

,P\_DESC\_COL=>'VENDOR\_NAME'

,P\_TABLE=>'ap\_suppliers'

,P\_ADDL\_W\_C=>'ROWNUM<10'

,P\_WINDOW\_CAPTION=>'Vendor'

,P\_WINDOW\_WIDTH=>null

,P\_WINDOW\_HEIGHT=>null

,P\_TABLE\_BLOCK\_SIZE=>null

,P\_TABLE\_SORT\_ORDER=>null

,P\_USER\_ID=>10653

--以下为component的定义参数

,P\_TABLE\_COLUMNS=>'VENDOR\_ID,SEGMENT1,VENDOR\_NAME'--提取的表的实际字段,即LOV的显示字段

,P\_TABLE\_SELECT\_COLUMNS=>'VENDOR\_ID,VENDOR\_NUM,VENDOR\_NAME'--需要提取到template上的字段，需与interface的字段名一致

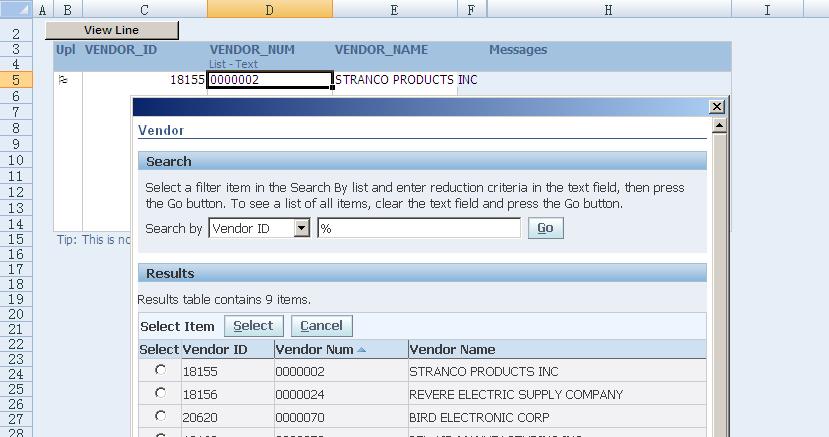
,P\_TABLE\_COLUMN\_ALIAS=>'VENDOR\_ID,VENDOR\_NUM,VENDOR\_NAME'--如果提取的LOV字段名与interface的字段名不一样，需要在此为LOV的字段定义别名，与interface字段名一致即可

,P\_TABLE\_HEADERS=>'VendorID,VendorNum,VendorName'

);

end;

**最终效果如下:**

[](http://space.itpub.net/batch.download.php?aid=34745)

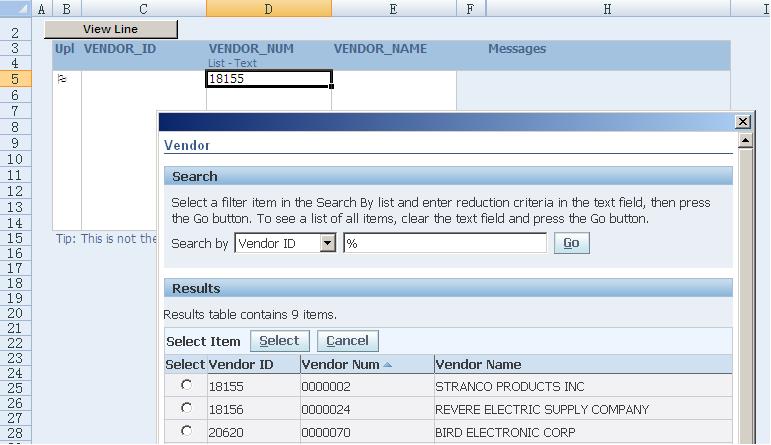
如果仅提取id,num,不提取name,则修改成:

**P\_TABLE\_SELECT\_COLUMNS=>'VENDOR\_ID,VENDOR\_NUM'**

但如果你仅想提取VendorNUM,如果你修改成这样的话:

**P\_TABLE\_SELECT\_COLUMNS=>'VENDOR\_NUM'**

你会发现结果变成了这样:

[](http://space.itpub.net/batch.download.php?aid=34746)

提取的仍是vendorid,并且放到了vendornum这个字段上去了...

因此，我们还需要修改这个:

**P\_TABLE\_COLUMNS=>'SEGMENT1,VENDOR\_ID,VENDOR\_NAME'**

**P\_TABLE\_COLUMN\_ALIAS=>'VENDOR\_NUM,VENDOR\_ID,VENDOR\_NAME'**

**P\_TABLE\_HEADERS=>'VendorNum,VendorID,VendorName'**

把segment1提到第一位去，这样的话，它将被第一个提取出来，放到template中的vendorNUM中.

从上可以看出，LOV定义的各字段都可以显示在LOV中，并且提取到template中，但如何提取，这就要看component的了...

## R12WEBADI的参数设置研究之一:设置CreateDocumentParameters

在WEBADI中，我们可以定义一些参数用来在整个WEBADI的导入过程中使用，一般来说，可分3个部分:

1.在生成template时所使用的参数，一般用来为layout中的字段设置默认值或初始化一些变量供LOV使用

2.在定义Uploader时所使用的参数，一般用来在Upload数据前提供给User一些option，可改变Upload的模式，例如要不要做validaiton,要不要import，要不要import一些非正常记录（如借贷不平的记录）。

3.在定义Importer时所使用的参数，一般用来在数据进入interface前和后所需要完成的一些事情，比如可在数据进入interface后，调用一个Concurrent（这个Concurrent可能需要参数）等等。

这里先说下第一种参数，在生成template的过程中使用的参数.

这种参数一般有两个用途：

1.在生成template的过程中，用户可选择某些参数值，这些值可用来做为template中的字段默认值或用于LOV的where条件

2.在生成template的过程中，参数可引用某些环境变量或profile值，然后用于生成template中的字段默认值或用于LOV.

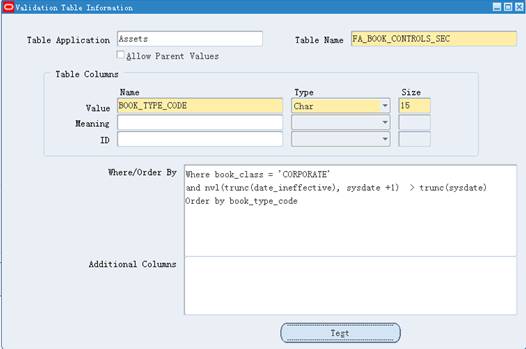
首先来看第一种，用户可在template生成时选择一些参数，来定义字段默认值.

假设我现在开发一个Integrator用来导入FAassets.是分book导入，要求template生成时booktypecode默认为用户在downloadtemplate时所选择的book,并且在template中无法修改。

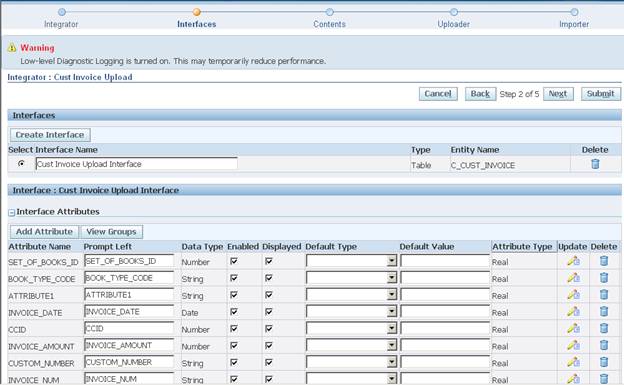
以下是开发步骤:

**1.建立一个valueset,此valueset用来显示所有的有效的Corpbook,供用户选择.**

[](http://space.itpub.net/batch.download.php?aid=33989)

[](http://space.itpub.net/batch.download.php?aid=33990)

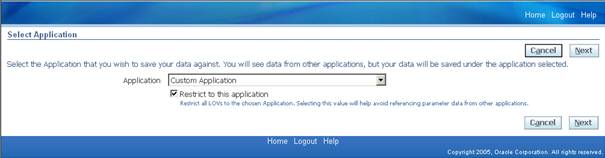
**2.建立integrator,生成interface(含有book\_type\_code字段),依次建立uploader,importer后保存.**

[](http://space.itpub.net/batch.download.php?aid=33992)

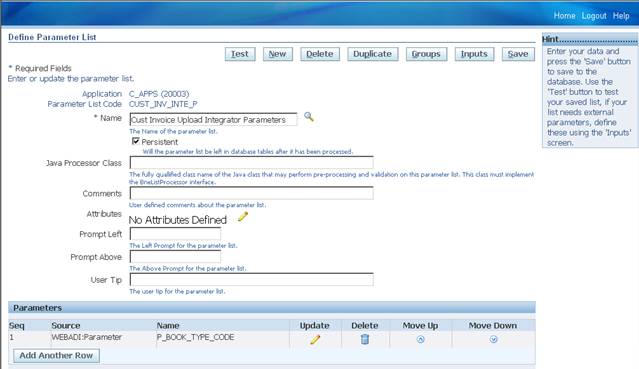
**3.建立Parameterlist,此list将包含一个参数，此参数将引用上述的valueset.**

Path:DesktopIntegrationManager/DefineParameter

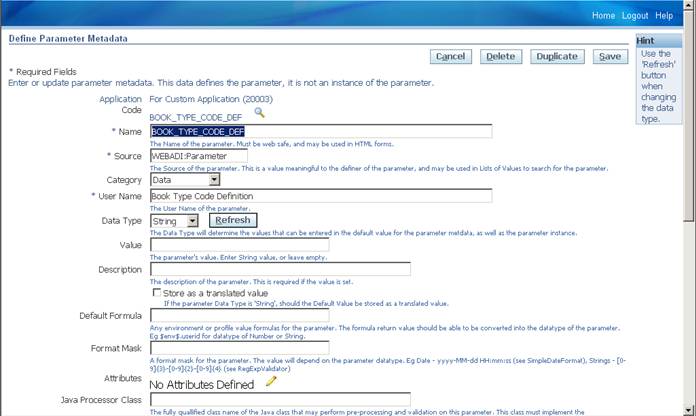
Application选择你所需的应用，应与integrator的应用一致.

[](http://space.itpub.net/batch.download.php?aid=33993)

建立一个新的Parameterlist,code为CUST\_INV\_INTE\_P,并新增一个参数为P\_BOOK\_TYPE\_CODE

[](http://space.itpub.net/batch.download.php?aid=33994)

新建此参数P\_BOOK\_TYPE\_CODE的Definition

[](http://space.itpub.net/batch.download.php?aid=34000)

[](http://space.itpub.net/batch.download.php?aid=33996)

注意:

1.Source为WEBADI:Parameter

2.Category为Data

3.Validationtype为valueset,Validationvalue即为上面建的valueset

4.Displaytype为Listbox，即为下拉列表

保存上面所有的定义：Parameter list/Parameter item/Parameter definition

**4.把上面定义的Parameterlist加入到Intergrator中,此步可以在定义integrator时定义integratorparameters时定义，也可直接update后台表，这里我直接Update.**

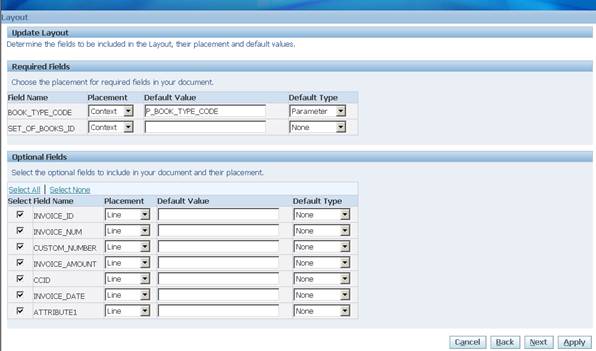
表名:bne\_integrators\_b

找到你定义的integrator记录，update下面两个字段:

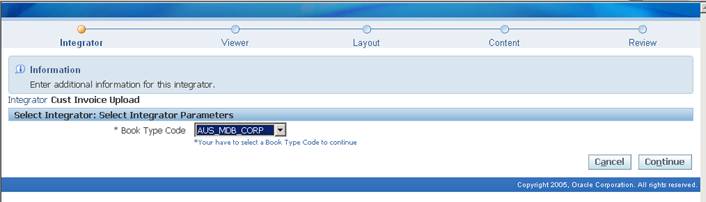
CREATE\_DOC\_LIST\_APP\_IDupdate为上面的Parameterlist的应用

CREATE\_DOC\_LIST\_CODEupdate为上面的Parameterlistcode

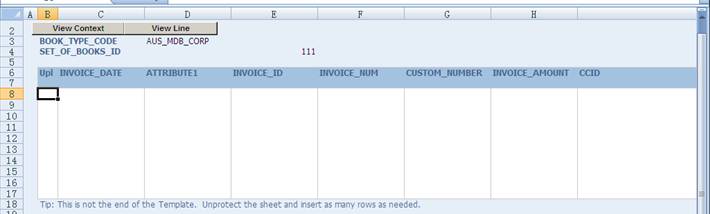
**5.DefineLayout,把booktypecode字段放到Context上,并定义Defaulttype为Parameter,Defaultvalue为上面定义的参数名：P\_BOOK\_TYPE\_CODE.**

[](http://space.itpub.net/batch.download.php?aid=33997)

**6.Downloadtemplate**

[](http://space.itpub.net/batch.download.php?aid=33998)

这是Download的最开始显示的，用户需选择一个BOOK.然后依次下去，最后Downlaod.

[](http://space.itpub.net/batch.download.php?aid=33999)

这是最终的template,我们发现用户选的BOOK已经出来，并且无法修改.

到此，完成了IntegratorParameter的设置.

## R12WEBADI中如何建立DATE字段的LOV验证

前提：

后台客制化表或标准Interface中含有日期型字段

建立步骤：

1.新建integrator,按照向导依次建立interface,uploader,importer,保存

2.通过如下API为日期型字段建立LOV

bne\_integrator\_utils.CREATE\_CALENDAR\_LOV(P\_APPLICATION\_ID=>20003,--Yourcustomapplication

P\_INTERFACE\_CODE=>'CUST\_INV\_XINTG\_INTF1',--Yourcustominterfacecode

P\_INTERFACE\_COL\_NAME=>'INVOICE\_DATE',--Yourdateitemininterface

P\_WINDOW\_CAPTION=>'SelectData',--WindowPrompt

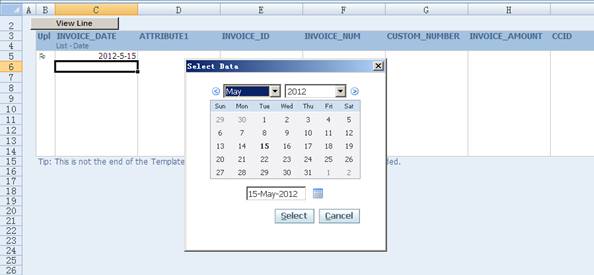
P\_WINDOW\_WIDTH=>null,--Usedefault

P\_WINDOW\_HEIGHT=>null,--Usedefault

P\_TABLE\_COLUMNS=>'INVOICE\_DATE',--Your**[table](javascript:;" \t "_self)**datefields

P\_USER\_ID=>5693);

Commit后下载template,发现已可以实现日期类型的LOV:

[](http://space.itpub.net/batch.download.php?aid=33901)

## R12WEBADI中建立基于KeyFlexfields的LOV验证

前提：

后台表中有存储CCID的字段（如果是标准interface的话，这个字段应该已经具备）

步骤：

1.新建Customintegrator

2.新建Interface,attribute中包括上述的CCID字段

3.依次建立uploader,importer后保存

4.使用下述的API建立KFF的LOV验证

bne\_integrator\_utils.CREATE\_CCID\_KFF(P\_APPLICATION\_ID=>20003,--Yourcustomapplicationid

P\_INTERFACE\_CODE=>'CUST\_INV\_XINTG\_INTF1',--Yourcustominterfacecode

P\_INTERFACE\_COL\_NAME=>'CCID',--YourCCIDitemininterface

P\_NUM\_FLEX\_SEGS=>30,--GLcodecombinationsegmentscount,最大30,实际显示的为启用的段

P\_GROUP\_NAME=>'Account',--AttributeGroupname,随便写

P\_FLEX\_APPLICATION\_ID=>101,--GLapplicationid

P\_FLEX\_CODE=>'GL#',--Accountflexcode

P\_FLEX\_NUM=>50328,--Chartsofaccountid,这里可用参数，比如“SOB.ChartOfAccountsID"

p\_vrule=>NULL,

p\_effective\_date\_col=>'date\_col02',--如interface中有effectivedate字段的话，会提供CCID的effectivedate到此字段（还没测）

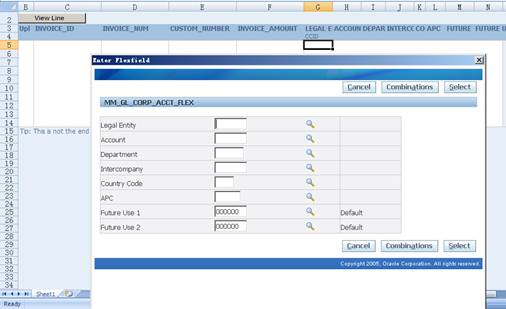
p\_prompt\_above=>'CCID',

p\_prompt\_left=>'CCID',

p\_user\_hint=>'CCID',

p\_user\_id=>5693);

Commit后,生成Template，发现已经生效:

[](http://space.itpub.net/batch.download.php?aid=33900)

**注意：在测试中，发现如果P\_NUM\_FLEX\_SEGS给的不够的话，文件生成时会报错。我一开始给的是7(因为只启用了7段），结果不行。后搞了半天，才发现这个问题，这里一律给最大的segment数量.**

## R12WEBADI如何建立基于table的LOV验证

假设interface中有一字段为customer number,需要提供LOV供用户选择，建立步骤如下:

1.建立Component

Component主要定义LOV弹出窗口的风格、布局及字段提取的规则

[](http://space.itpub.net/batch.download.php?aid=33886)

输入Displayname/Internalname和Application

ComponentType选LOV-TABLE

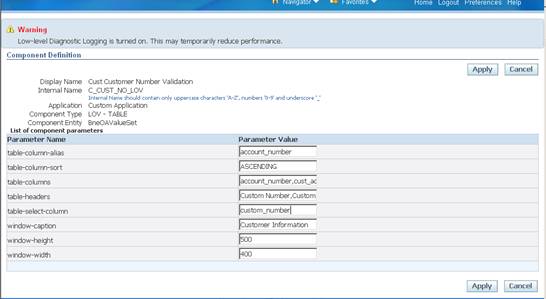
[](http://space.itpub.net/batch.download.php?aid=33884)

table-column-alias:输入提取的字段放在哪个interface column中（interface中的字段名)

table-column-sort:保持默认

table-columns:输入需要显示的表字段，按显示顺序输入，中间用逗号间隔

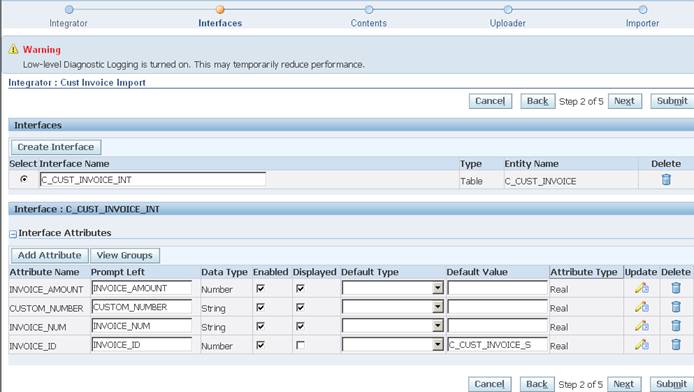
table-headers:输入各字段的标题

table-select-column:输入提取的字段放在哪个interface column中（interface中的字段名)

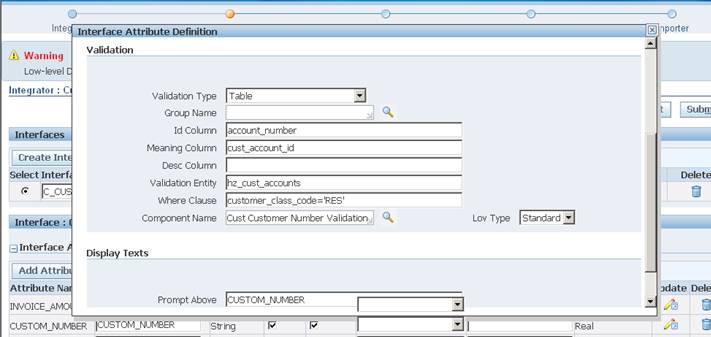
window-caption:根据需要输入提示信息

2.定义interface字段的属性

找出之前定义好的integrator,点update,next到interface页面后，显示如下:

[](http://space.itpub.net/batch.download.php?aid=33891)

update相应的字段，此处是CUSTOM\_NUMBER,显示如下:

[](http://space.itpub.net/batch.download.php?aid=33893)

ValidationType:**[Table](javascript:;" \t "_self)**

IdColumn:account\_number

MeaningColumn:cust\_account\_id

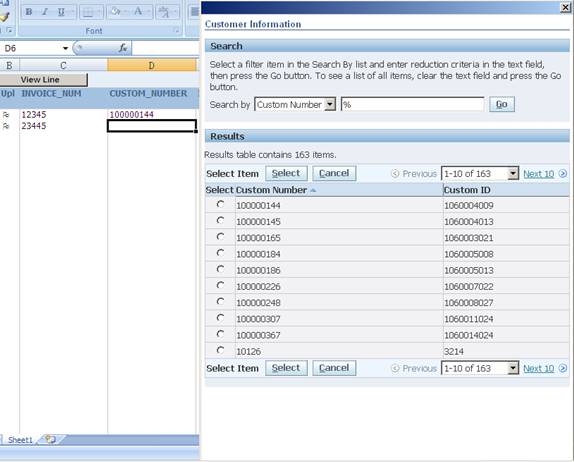
ValidationEntity:hz\_cust\_accounts(查询的表名，用逗号分隔多张表)

WhereClause:根据需要输入，不要加Where

ComponentName:输入上面定义的Componentname

此步定义与EBS中的基于table的valueset定义基本一样.

保存后，downloadtemplate,实际效果如下:

[](http://space.itpub.net/batch.download.php?aid=33894)

## R12WEBADI如何在template中自动生成sequence字段

最近要做大量的WEBADI开发(基于R1212.1.3),因此这几天在不断的做新功能的研究。其实一直有不少的问题，这次希望能一点点的研究出来。。。包括dependentLOV和attributegroupvalidation等等功能.

让exceltemplate自动生成sequence.

真没想到这个功能Oracle居然没提供...当然如果integrator是基于客制化的API(package)，那不存在这个问题，直接在package里取sequence就行了。但如果直接以Interfacetable做integratorinterface时，就而临这个问题了，你没地方写语句去取sequence...通过标准的integrator研究，发现实现方法如下:

a.createintegrator的第二步生成interface后，需要生成sequence的key字段此时已出现在interface里，但defaulttype里没有关于sequence的设置

b.查询bne\_interface\_cols\_b表，找到此key字段的记录，进行后台修改，在defaulttype里放入"SEQUENCE\_EACH\_ROW",defaultvalue里放入sequencename(注意，此sequence需要在apps下建同义词),submit修改。

c.再进入web页面查出interface，你会发现此key字段的defaulttype居然为空...进一步说明可恶的Oracle没有开放sequence的定义功能。

e.生成template后，你会发现，定义了sequence的字段不会在excel里显示，即使你在interface里把它选成display，系统也会去除勾选。这样的话，用户就不会错误的操作此字段了.

f.输入数据上传，查询后台表，sequence已正常显示在key字段里了.

## R12WEBADI-关于ADI securityrules

在11i里，我们一般是建立一个subfunction做为securityfunciton,然后与integrator建立关联，再把这个securityfunction及ADIfunction一起加入到usermenu中，user即可使用此ADI,当时就觉得挺麻烦。当然，你可以通过设置profile去禁用ADIsecurity.

在R12中，可以很方便的在定义integrator时定义securityfunction了，而且一个integrator可以定义多个securityfunction.但我查了一下诸多的标准integrator,它们用的securityfunction大多是系统标准的form.function和webfunciton,而不是subfunction.这也就意味着，只要usermenu中包含某个职责，这个职责下有这个form.function的话，即可使用这个ADI,这样的话，我们就不需要再建一个subfunction了，更方便了。

因此测试了一下，我们客制化的integrator也可以选择某一个系统标准的form.function做为securityfunction.比如我建了一个assetmassload的ADI,我希望只要能进行asset常规操作的user都能使用这个ADI.我选择了一个系统标准的form.function:FAXASSET(AssetWorkbench),做为我的ADI的securityfunciton.但布署时要注意:

你的ADI的function必须要和FAXASSET这个function在同一个menu里。即在同一个职责下。

一开始我把ADI的funciton单独放在一个职责里，然后把此职责和另一个包含assetworkbench的职责加给用户，但测试结果是此ADI无法使用。之后把ADI的funciton加入到assetworkbench的职责中，测试通过。

此项功能大大方便了ADI的布署，非常有用.

## 在WEBADI中实现组织屏蔽

这几天做了个WEBADI的任务，要求在模板里实现组织屏蔽。也就是有个业务实体列让用户选，但只能选当前职责能操作的业务实体。

最初用这样的方法来实现LOV提取可操作OU:

Select name

From hr\_operating\_unitshou

Where mo\_global.check\_access(hou.organization\_id)='Y'

但发现不行，原因可能是mo\_global需要进行初始化。

最后用了下面的笨方法:

思路是:检查当前职责，当前用户的配置文件，取MO业务实体，MO默认业务实体，MO安全性配置文件的信息，然后计算可操作的OU.

1.写个函数取当前模板的职责用户所用的安全性配置文件，默认只取用户层及职责层的，如果用户层有，则不取职责层.

FUNCTION get\_sec\_profile\_id(pi\_user\_idINNUMBER,pi\_resp\_idINNUMBER)

RETURNNUMBERIS

v\_profile\_idNUMBER;

BEGIN

BEGIN

SELECTto\_number(pov.profile\_option\_value)

INTOv\_profile\_id

FROMfnd\_profile\_optionspo

,fnd\_profile\_option\_valuespov

WHEREpo.profile\_option\_name='XLA\_MO\_SECURITY\_PROFILE\_LEVEL'

ANDpo.profile\_option\_id=pov.profile\_option\_id

ANDpov.level\_id=10004--用户层

ANDpov.level\_value=pi\_user\_id;

EXCEPTION

WHENno\_data\_foundTHEN

BEGIN

SELECTto\_number(pov.profile\_option\_value)

INTOv\_profile\_id

FROMfnd\_profile\_optionspo

,fnd\_profile\_option\_valuespov

WHEREpo.profile\_option\_name='XLA\_MO\_SECURITY\_PROFILE\_LEVEL'

ANDpo.profile\_option\_id=pov.profile\_option\_id

ANDpov.level\_id=10003--职责层

ANDpov.level\_value=pi\_resp\_id;

EXCEPTION

WHENno\_data\_foundTHEN

v\_profile\_id:=NULL;

END;

END;

RETURNv\_profile\_id;

END;

2.然后业务实体的LOV用如下**[SQL](javascript:;" \t "_self)**来构造:

SELECThou.name

,hou.organization\_id

FROMhr\_operating\_unitshou

WHEREhou.organization\_id=$profiles$.org\_id

ORhou.organization\_id=$profiles$.default\_org\_id

OREXISTS(SELECT1

FROMper\_security\_profilessp

WHEREsp.security\_profile\_id=

cux\_common\_utilities\_pkg.get\_sec\_profile\_id($env$.userid,

fnd\_profile.value('RESP\_ID'))

ANDsp.view\_all\_organizations\_flag='Y')

OREXISTS(SELECT1

FROMper\_security\_organizationsso

WHEREso.security\_profile\_id=

cux\_common\_utilities\_pkg.get\_sec\_profile\_id($env$.userid,

fnd\_profile.value('RESP\_ID'))

ANDso.organization\_id=hou.organization\_id)

## R12WEBADI表结构

1.bne\_integrators\_b：存贮定义的WebADI集成器

2.bne\_interfaces\_b：存贮集成器对应接口信息

3.bne\_interface\_cols\_b：存贮接口对应的列信息，包括列的各种验证信息和属性

4.bne\_components\_b：OA组件定义，如一个验证的LOV就是一个组件

5.bne\_param\_lists\_b：参数列表，可以理解为参数的分组，目的是为了实现参数的重用

6.bne\_param\_list\_items：存贮参数列表和参数的关联信息

7.bne\_param\_defns\_b：存贮参数定义，包括默认值、验证信息、显示信息等

--1、integrator table

--集成器表

--alter session set nls\_language='SIMPLIFIED CHINESE';

--alter session set nls\_language='AMERICAN';

SELECT bni.application\_id

,bni.integrator\_code --集成器code

,bni.user\_name --集成器名称

,bni.date\_format

FROM bne\_integrators\_vl bni

WHERE bni.user\_name LIKE 'CUX%';

--2.1集成器关联package

SELECT bni.application\_id

,bni.integrator\_code --集成器code

,bni.user\_name --集成器名称

,bni.date\_format

,inf.interface\_code

,inf.interface\_name --集成器关联的packege的proced

FROM bne\_integrators\_vl bni

,bne\_interfaces\_vl inf

WHERE bni.user\_name LIKE 'CUX%'

AND bni.application\_id = inf.integrator\_app\_id

AND bni.integrator\_code = inf.integrator\_code;

--2.2集成器对应的字段

SELECT intg.application\_id

,intg.integrator\_code --集成器code

,intg.user\_name --集成器

,intg.date\_format

,bic.interface\_code --package code

,bic.sequence\_num

,bic.display\_order

,bic.interface\_col\_name --字段名

,bic.data\_type

,bic.prompt\_left --字段显示名称

,bic.prompt\_above --可update此字段为中文

FROM bne\_interface\_cols\_vl bic

,bne\_interfaces\_vl intf

,bne\_integrators\_vl intg --集成器

WHERE bic.interface\_code = intf.interface\_code

AND intf.integrator\_code = intg.integrator\_code

AND intg.user\_name = 'CUX:WEBADI\_LOV';

--2.3.1 Excel列的题头和API的参数一样，需要加以修改，需直接更新表bne\_interface\_cols\_tl的

--alter session set nls\_language='SIMPLIFIED CHINESE';

--alter session set nls\_language='AMERICAN';

SELECT bic.interface\_code --package code

,bic.prompt\_left --头表题头说明

,bic.prompt\_above --行表题头说明

,bic.sequence\_num

FROM bne\_interface\_cols\_tl bic

WHERE EXISTS (SELECT NULL

FROM bne\_interfaces\_vl intf

,bne\_integrators\_vl intg --集成器

WHERE bic.interface\_code = intf.interface\_code

AND intf.integrator\_code = intg.integrator\_code

AND intg.user\_name LIKE 'CUX%')

-- FOR UPDATE

--2.3.2 Excel LOV实现，可以直接update基表bne\_interface\_cols\_b

SELECT bic.interface\_col\_name --字段名

,bic.val\_type --填 TABLE

,bic.val\_id\_col --传递给程序参数的字段，填 ORGANIZATION\_ID

,bic.val\_mean\_col --在Ecxel中显示的字段， 填ORANIZATION\_ID

,bic.val\_desc\_col --LOV中说明字段，填ORGANIZATION\_CODE,ORGANIZATION\_NAME

,bic.val\_obj\_name --列表所在的表明，填ORG\_ORGANIZATION\_DEFINITIONS

,bic.val\_addl\_w\_c --where语句,可不填

,bic.val\_component\_app\_id --组件应用id，填20003

,bic.val\_component\_code --组件代码，还未创建，先填一个，然后创建， 填CUX\_ORGANIZATION

,bic.lov\_type --填POPLIST或STANDARD,也可以填NONE

,bic.offline\_lov\_enabled\_flag --离线编辑EXCEL表格时是否可用，填Y

FROM bne\_interface\_cols\_b bic

WHERE EXISTS (SELECT NULL

FROM bne\_interfaces\_vl intf

,bne\_integrators\_vl intg --集成器

WHERE bic.interface\_code = intf.interface\_code

AND intf.integrator\_code = intg.integrator\_code

AND intg.user\_name = 'CUX\_WEBADI\_DEMO')

AND bic.interface\_col\_name = 'P\_ORGANIZATION\_ID'

--FOR UPDATE

;

--3、integrator and function associations

--集成器和功能关联

SELECT bni.application\_id

,bni.integrator\_code --集成器code

,bni.user\_name --集成器名称

,bsr.security\_value --功能code

FROM bne\_integrators\_vl bni

,bne\_security\_rules bsr

WHERE bni.user\_name LIKE 'CUX%'

AND bsr.application\_id = bni.application\_id

AND bsr.security\_code = bni.integrator\_code;

--4、集成器对应的布局

--alter session set nls\_language='SIMPLIFIED CHINESE';

--alter session set nls\_language='AMERICAN';

SELECT bni.application\_id

,bni.integrator\_code --集成器code

,bni.user\_name --集成器名称

,blv.layout\_code --布局代码

,blv.user\_name --布局用户名称

,blc.sequence\_num

,bic.\*

FROM bne\_integrators\_vl bni

,bne\_layouts\_vl blv

,bne\_layout\_cols blc

,bne\_interface\_cols\_vl bic

WHERE bni.user\_name LIKE 'CUX\_WEBADI\_DEMO'

AND blv.integrator\_app\_id = bni.application\_id

AND blv.integrator\_code = bni.integrator\_code

AND blv.layout\_code = blc.layout\_code

AND blc.interface\_app\_id = bic.application\_id

AND blc.interface\_code = bic.interface\_code

AND blc.interface\_seq\_num = bic.sequence\_num;

--5、集成器对应的映射

--alter session set nls\_language='SIMPLIFIED CHINESE';

--alter session set nls\_language='AMERICAN';

SELECT bni.application\_id

,bni.integrator\_code --集成器code

,bni.user\_name --集成器名称

,bmv.mapping\_code --映射code

FROM bne\_integrators\_vl bni

,bne\_mappings\_vl bmv

WHERE bni.user\_name LIKE 'CUX%'

AND bmv.integrator\_app\_id = bni.application\_id

AND bmv.integrator\_code = bni.integrator\_code;