White Paper

Exposing pl/sql as REST Web Services using Integrated SOA Gateway

Author: Dipti Kaul
Sep-2015
Fujitsu Consulting India Private Limited

Table of Contents

Overview

This document describes the process of exposing Standard and Custom API as Rest Web service using Integrated SOA Gateway. This document will help developers in understanding the ISG Customization, Rest Web Services and its development. This document is complete guide to ISG Customization with screenshots .It covers following case studies:

1. Exposing **Standard API** as REST Web Service and invoking it

2. Exposing **Pl/Sql Function** in Package as REST Web Service returning **single value a**nd invoking it.

3. Exposing **Pl/Sql Procedure** in Package as REST Web Service **with Table Type as OUT Variable** and invoking it.

REST Web Services are available in Oracle E-Business Suite Release 12.2.3 and onwards only. REST based web services are deployed on Oracle E-Business Suite's application server. They do not dependent on Oracle SOA Suite and Oracle Applications Adapter.

Integrated SOA Gateway

Oracle E-Business Suite Integrated SOA Gateway (ISG) is a complete set of service infrastructure to provide, consume, and administer Oracle E-Business Suite Web services.
You can use this tool to easily discover and search on interfaces, regardless of custom or Oracle seeded ones.

Major Features of ISG

Oracle E-Business Suite Integrated SOA Gateway can do the following:

- Display all Oracle E-Business Suite integration interface definitions through Oracle Integration Repository
- Support custom integration interfaces from Oracle Integration Repository
- Provide service enablement capability (SOAP and REST services) for seeded and custom integration interfaces within Oracle E-Business Suite
- Use the Integration Repository user interface to perform design-time activities such as generate and deploy Oracle E-Business Suite Web services
- Support synchronous interaction pattern for REST-based Web services
- Support multiple authentication types for inbound service requests in securing Web service content
- Enforce function security and role-based access control security to allow only authorized users to execute administrative functions
- Provide centralized, user-friendly logging configuration for Web services generated through Oracle E-Business Suite Integrated SOA Gateway's service provider
- Audit and monitor Oracle E-Business Suite inbound service operations from Service Monitor
- Leverage Oracle Workflow Business Event System to enable Web service invocation from Oracle E-Business Suite

REST Web Services

The acronym REST stands for Representational State Transfer, this basically means that each unique URL is a representation of some object. **REST** describes a set of architectural principles by which data can be transmitted over a standardized interface (such as HTTP). You can get the contents of that object using an HTTP GET, to delete it, you then might use a POST, PUT, or DELETE to modify the object (in practice most of the services use a POST for this).

Key features of EBS REST Services are as follows:

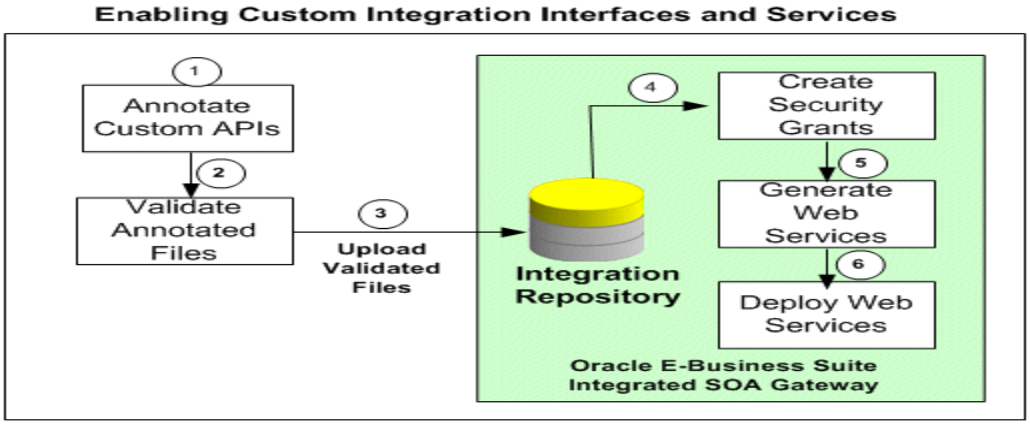
- Supports POST HTTP Verb
- Supports XML and JSON message format
- Described in WADL, XSDs
- No expensive tools require to interact with the Web service
- Smaller learning curve
- Efficient (SOAP uses XML for all messages, REST can use smaller message formats)
- Fast (no extensive processing required)
- REST-based implementation is simple compared to SOAP.

Integration Repository Customization

The term Customization of Integration Repository means option to add custom interfaces into Integration Repository. The term interface stands for all types of programs and technologies in Integration Repository, eg. plsql, java, business events, xml gateway etc.

Integration developers create and annotate custom integration interfaces based on the Integration Repository annotation standards. Integration administrators use a standalone design-time tool to validate these annotated source files against the annotation standards. After validation, a loader file is generated and then uploaded to the Integration Repository through backend processing. These custom interfaces are displayed based on the interface types to which they belong and displayed together with Oracle seeded ones from the Integration Repository user interface.

Process Flow



Annotation Standards

Please refer the below link for details on Annotating Pl/Sql code.

http://docs.oracle.com/cd/E18727_01/doc.121/e12065/T511473T545912.htm

Setting Up Oracle E-Business Suite Integrated SOA Gateway

Please refer to below link for details:

http://docs.oracle.com/cd/E18727_01/doc.121/e12169/T511175T517162.htm

Also, please refer to the Doc ID 1311068.1 for ISG installation for REST Services.

Case Studies

Exposing Standard API as REST Web Service and Invoking it.

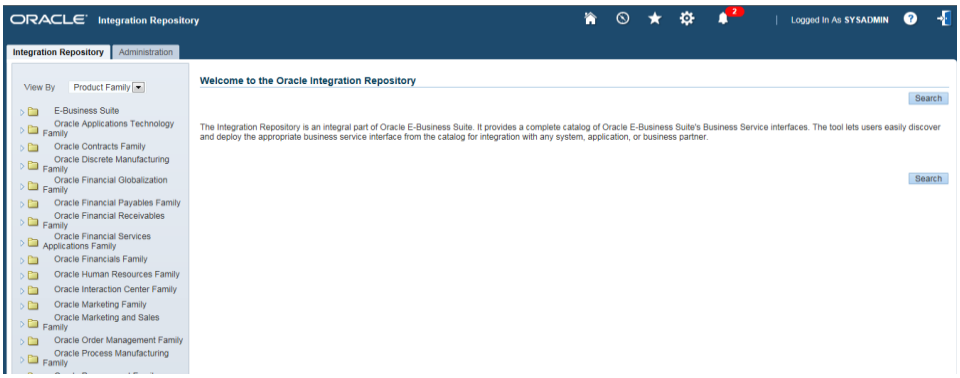
Perform following steps to expose Pl/Sql as a web service

1. Search API in Integration Repository
2. Deploy API as REST Service
3. Create Grants for REST Service
4. Invoke EBS REST Service (in JSON message format)
5. Verify the results

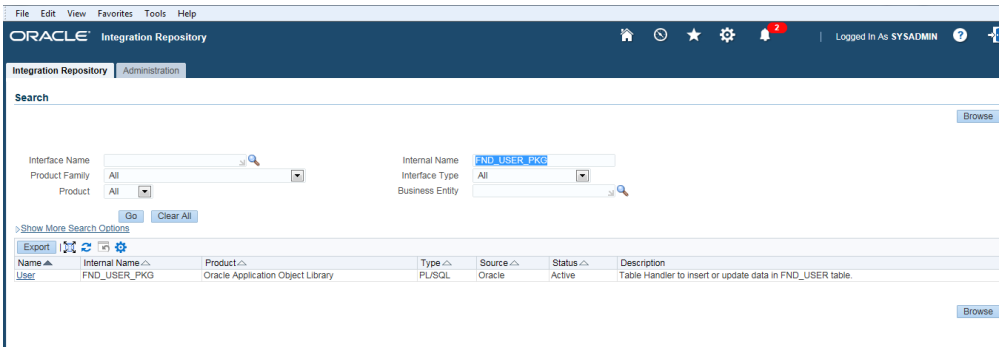
Here we will take an example of creating a user by exposing FND_USER_PKG Interface API and then invoking it to create user.

Following steps will explain the process of exposing the standard API and then invoking it.

Step 1: Navigate to Integration Repository Responsibility, Click on Search.

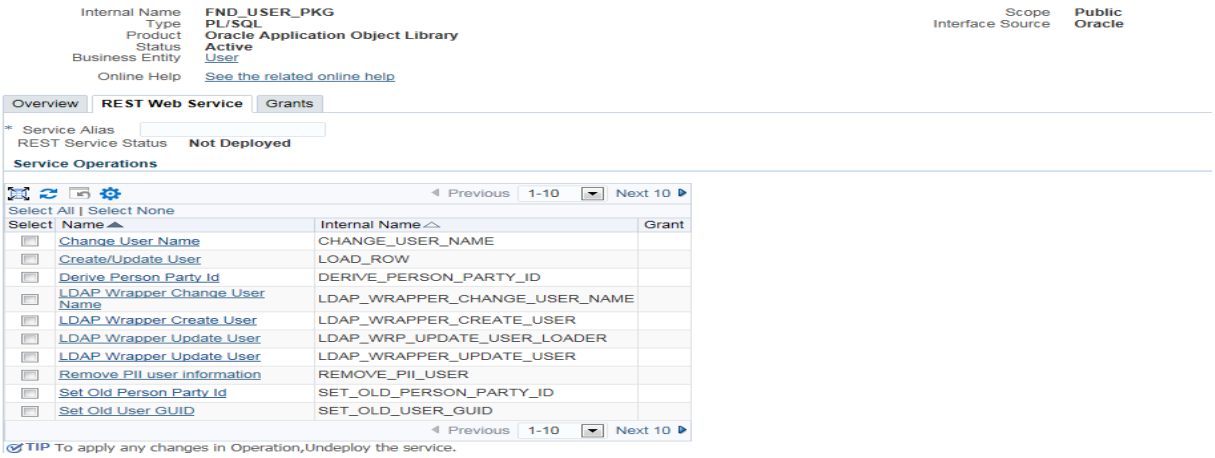


Then on search Page enter "FND_USER_PKG" in Internal Name and click on Go



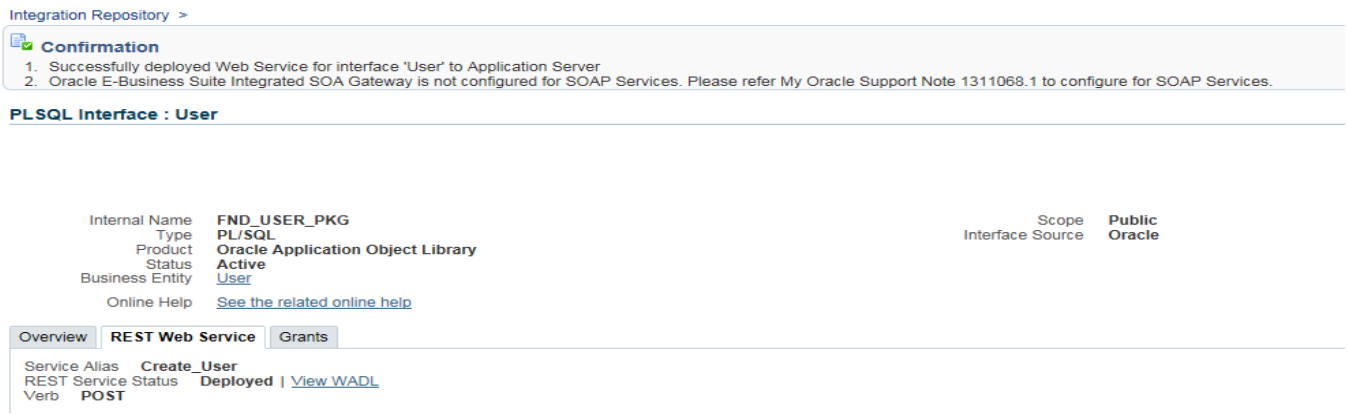
Step 2: Click on the search result, it will display List of methods available in the Interface Package.

Click on **REST Web Service** Tab



This page will display List of methods that can be deployed.

Step 3: Now select Create/Update User (we will create a user by invoking this Service once Deployed.) and enter Service Alias (This is mandatory) and then Click on Deploy button at bottom.
You will receive message that service is successfully deployed and the status will be deployed.

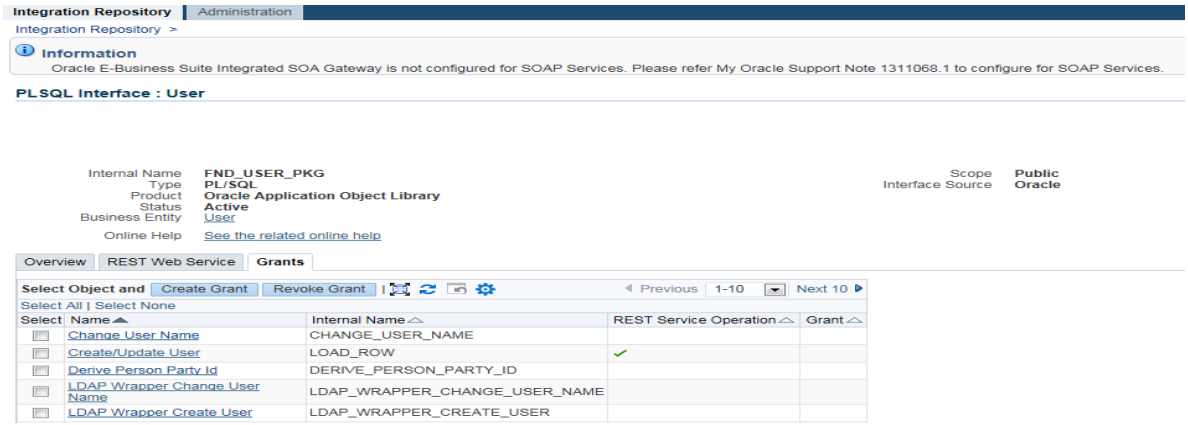


Step 4: Now you will see the link to WADL file, Click on the link. It shows the physical location of the service endpoint where the service is hosted.

```
<?xml version="1.0" encoding="UTF-8"?>
<application
xmlns:tns1="http://xmlns.oracle.com/apps/fnd/rest/create_user/load_row/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://wadl.dev.java.net/2009/02"
xmlns:tns="http://xmlns.oracle.com/apps/fnd/soaprovider/plsql/rest/fnd_user_pkg/"
targetNamespace="http://xmlns.oracle.com/apps/fnd/soaprovider/plsql/rest/fnd_user_pkg/"                                    name="FND_USER_PKG"><grammars><include
xmlns="http://www.w3.org/2001/XMLSchema"

href="http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Create_User/?XSD=LOAD_ROW_SYNCH_TYPEDEF.xsd"/>

</grammars>
<resources                                                                  base="http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Create_User/"><resource path="load_row/">
<method            name="POST"            id="LOAD_ROW"><request><representation
type="tns1:InputParameters" mediaType="application/xml"/>
<representation type="tns1:InputParameters" mediaType="application/json"/>
</request>
</method>
</resource>
</resources>
</application>
```

Now open the link highlighted in Yellow in new Browser window, it will show the **XSD file** with input parameters that will be used later for invoking the Web Service.

Step 5.Now Click on the Grants Tab, select Create/Update User and then click on Create Grant button



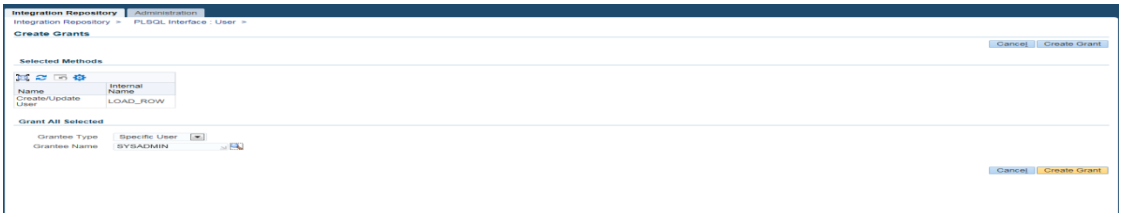
Step 6: Select a grantee type.

Grantee Type can be one of the following values:

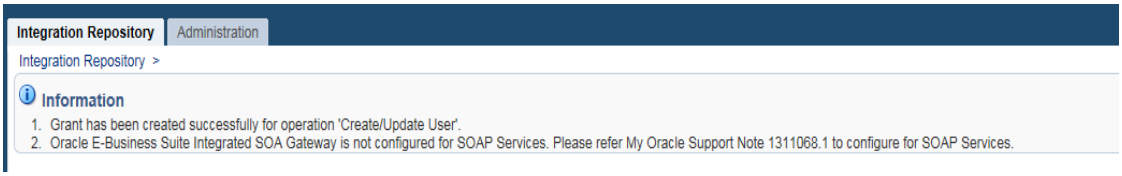
- USER - The grantee is an individual user who was selected directly.
- GROUP - The grantee is a group of users or a member of a group of users.
- GLOBAL - The grant was issued to all users.

If you selected Specific User or Group of Users, specify the user or group for which to create the grants in the Grantee Name field.

In our Case Study we will enter SYSADMIN and then click on Create Grant Button.



It will display confirmation message:

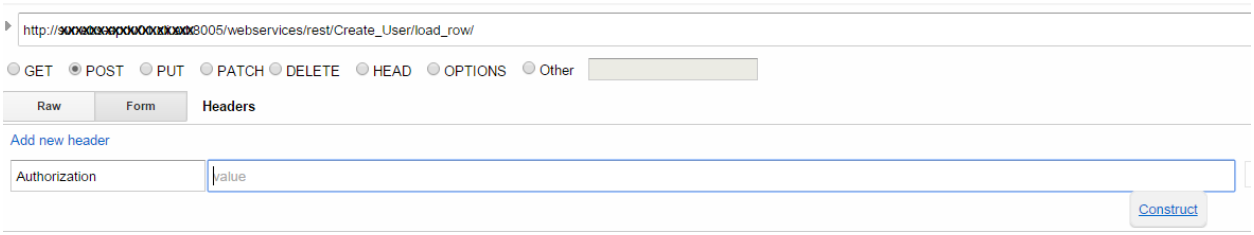


To **Revoke Grants** click on Revoke Grants button under Grants Tab ,it will display List of users with access for revoking the Grants.
To **Undeploy** the Web services select the method and click on undeploy under the REST Web Services Tab

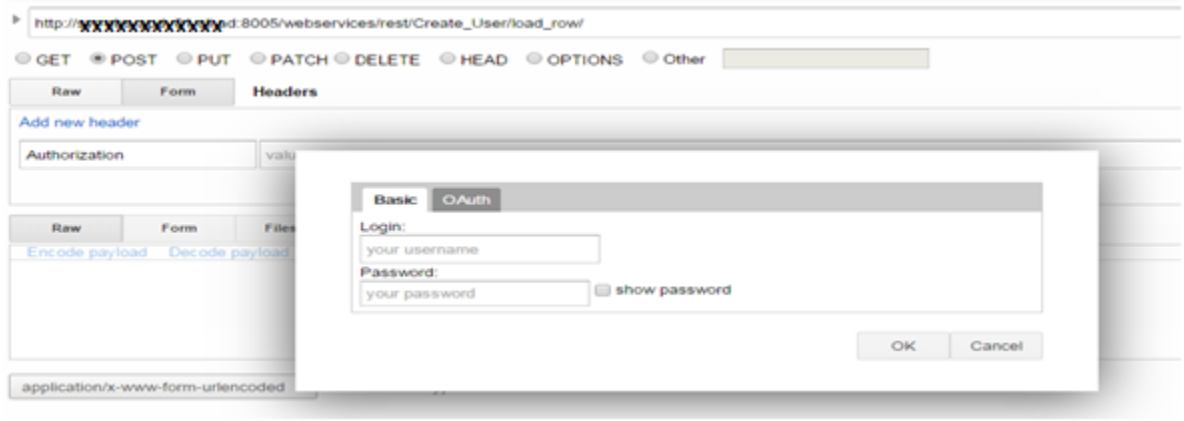
Step 7: Now we will invoke this web service using Advanced REST Client apps available in Google Chrome.
If not available download it from following link:

https://chrome.google.com/webstore/detail/advanced-rest-client/hgmloofddffdnphfgcellkdfbfbjeloo

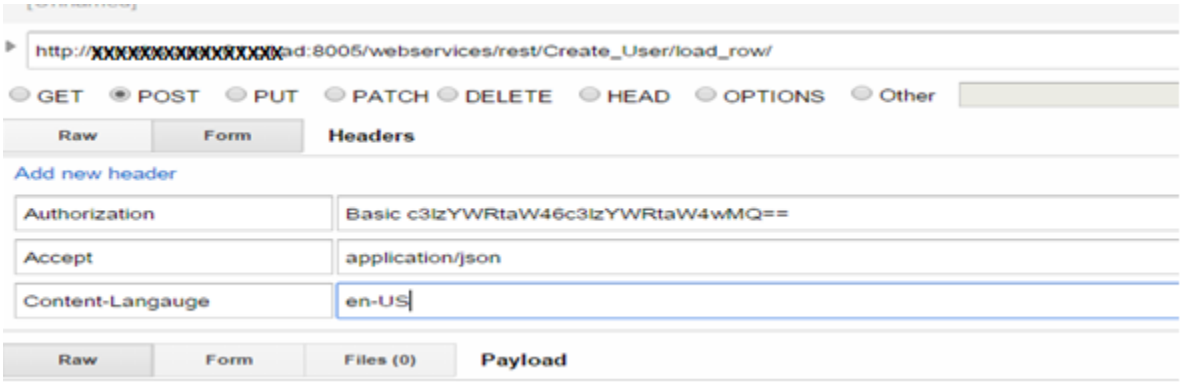
Now, after installing the REST client we will enter following details to invoke the web service:



Step 8: Enter the following header details:
Authorization by clicking on Add new header link then enters the application username/password by clicking on construct.



Then enter Accept, Content language



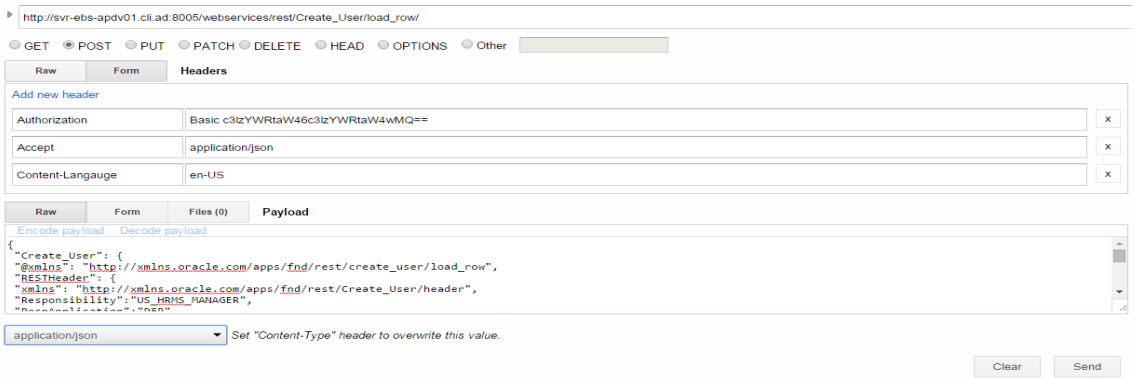
Select application/json in the Content Type below.

Step 9: Enter the following Link and payload information and click on send. The payload is constructed using the xsd file generated at Step 4.

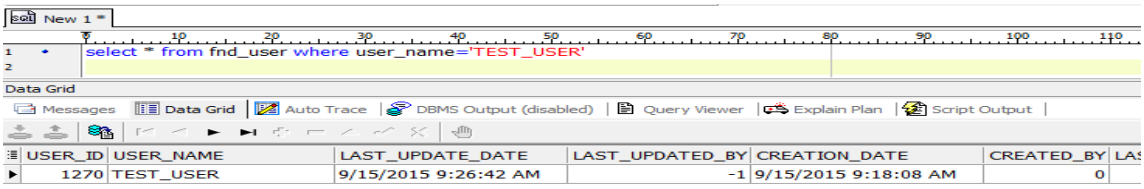
Link : http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Create_User/load_row/

Payload:

```
{
 "Create_User": {
"@xmlns": "http://xmlns.oracle.com/apps/fnd/rest/create_user/load_row",
"RESTHeader": {
"xmlns": "http://xmlns.oracle.com/apps/fnd/rest/Create_User/header",
"Responsibility":"US_HRMS_MANAGER",
"RespApplication":"PER",
"SecurityGroup":"STANDARD",
"NLSLanguage":"AMERICAN",
"Org_Id" :"201"
},
"InputParameters": {
"X_USER_NAME": "TEST_USER",
"X_ENCRYPTED_USER_PASSWORD": "Welcome123",
"X_START_DATE": "14/09/2015",
"X_EMAIL_ADDRESS": "test_user@test.com"
}
}
}
```



Step 10: Now to verify if user is created query in FND_USER table to check the results.



Exposing Pl/Sql as REST Web Service (Function Returning single value)

Perform following steps to expose Pl/Sql as a web service

1. Create a Custom Package ,annotate it
2. Create iLDT File

3. Upload the iLDT File to Integration Repository.
4. Search API in Integration Repository
5. Deploy API as REST Service
6. Create Grants for REST Service
7. Invoke EBS REST Service (in JSON message format)
8. Verify the results

Here we will take an example of Getting Employee Number based on Employee details provided when invoking the Web service.

Following steps will explain the process of exposing the Custom API that returns varchar variable as output when invoked.

Step 1: First Create a custom package xxtrc_get_emp_numer_pkg with .pls extension and annotate it Only annotate the package specification. Below code was used for this Case study.

Package Specification:

```
CREATE OR REPLACE PACKAGE xxtrc_get_emp_number_pkg AS

/* $Header: $ */

/*#

* This interface returns the Employee Number.

* @rep:scope public

* @rep:product PER

* @rep:displayname xxtrc_get_emp_number_pkg

* @rep:lifecycle active

* @rep:compatibility S

* @rep:category BUSINESS_ENTITY PER_EMPLOYEE

*/



/*#

* Returns Employee Number

* @param P_First_name VARCHAR2 First Name

* @param P_Last_name VARCHAR2 Last Name
```

* @param P_Email_address VARCHAR2 Email Address

* @return Employee_Number

* @rep:scope public

* @rep:lifecycle active

* @rep:displayname Return Employee Number

*/

FUNCTION get_emp_number (P_First_name IN VARCHAR2,

P_Last_name IN VARCHAR2,

P_Email_address IN VARCHAR2

) RETURN VARCHAR2;

END xxtrc_get_emp_number_pkg;

Package Body:

```
CREATE OR REPLACE PACKAGE BODY xxtrc_get_emp_number_pkg
AS
   FUNCTION get_emp_number (P_First_name      IN VARCHAR2,
                            P_Last_name       IN VARCHAR2,
                            P_Email_address   IN VARCHAR2)
      RETURN VARCHAR2
   AS
      l_emp_num   VARCHAR2 (30);
   BEGIN
      SELECT employee_number
        INTO l_emp_num
        FROM per_all_people_f
       WHERE NVL (UPPER (first_name), 'X') = NVL (UPPER
(P_First_name), 'X')
             AND UPPER (last_name) = UPPER (last_name)
             AND NVL (email_address, 'X') = NVL (P_Email_address,
'X');

      RETURN l_emp_num;
   EXCEPTION
      WHEN NO_DATA_FOUND
      THEN
```

```
         RETURN 'No Data Found';
      WHEN OTHERS
      THEN
         RETURN 'Invalid Data';
   END get_emp_number;
END xxtrc_get_emp_number_pkg;
```

Step 2: Upload the .pls file to Oracle server.For our case study file was uploaded to custom_top Bin folder

Step 3: Create iLDT file by using following command and current Directory is where the .pls file was uploaded :

$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -g -v -username=sysadmin PER:patch/115/sql:xxtrc_get_emp_number_pkg.pls:12.0=xxtrc_get_emp_number_pkg.pls

Following message will appear after executing the command and it will generate ilDT file.

```
[applmgr@svr-ebs-apdv01 ~]$ cd $XXTRC_TOP
[applmgr@svr-ebs-apdv01 12.0.0]$ cd bin
[applmgr@svr-ebs-apdv01 bin]$ $IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -g -v -username=sysadmin PER:patch/115/sql:xxtrc_get_emp_number_pkg.pls:12.0=xxtrc_get_emp_number_pkg.pls
# Interface Repository Annotation Processor, 12.0.0

#
# Generating annotation output.
# Processing file 'xxtrc_get_emp_number_pkg.pls'.
# Using YAPP-based parser.
#  Found a package-level annotation for 'XXTRC_GET_EMP_NUMBER_PKG'.
#  Found a detail-level annotation...
# Found a function named 'GET_EMP_NUMBER'.
# Done all files.
[applmgr@svr-ebs-apdv01 bin]$
```

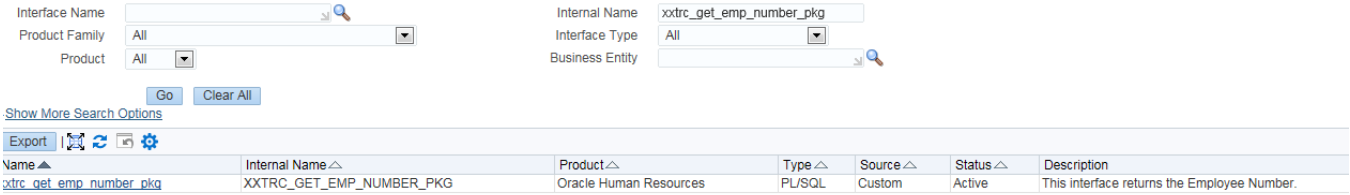
Step 4: Upload the generated iLDT file to Integration repository by using following command:

$FND_TOP/bin/FNDLOAD apps/apps 0 Y UPLOAD $FND_TOP/patch/115/import/wfirep.lct xxtrc_get_emp_number_pkg_pls.ildt

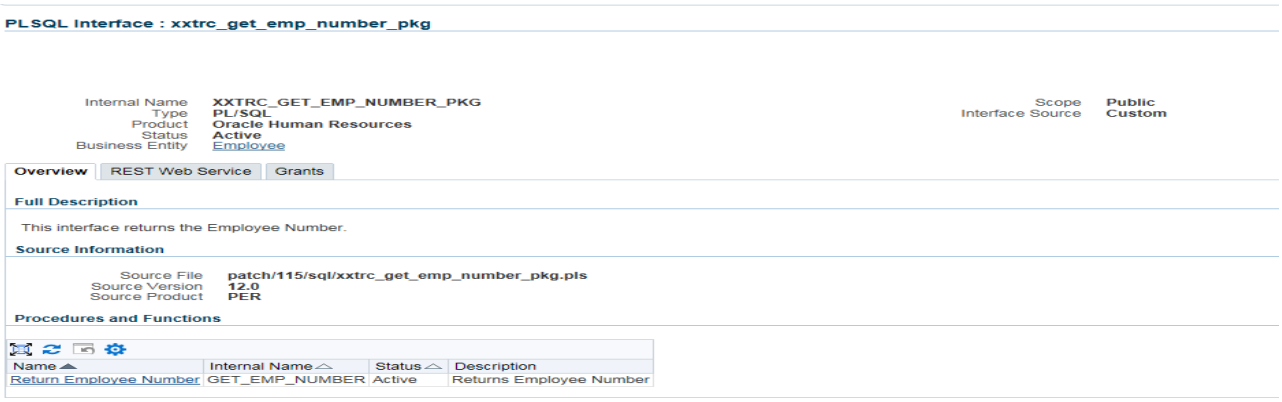
```
[applmgr@svr-ebs-apdv01 bin]$ $FND_TOP/bin/FNDLOAD apps/apps 0 Y UPLOAD $FND_TOP/patch/115/import/wfirep.lct xxtrc_get_emp_number_pkg_pls.ildt
Log filename : L551303.log


Report filename : O551303.out
[applmgr@svr-ebs-apdv01 bin]$
```

Step 5: Navigate to Integration Repository Responsibility, Click on Search.

Then on search Page enter “xxtrc_get_emp_number_pkg” in Internal Name and click on Go



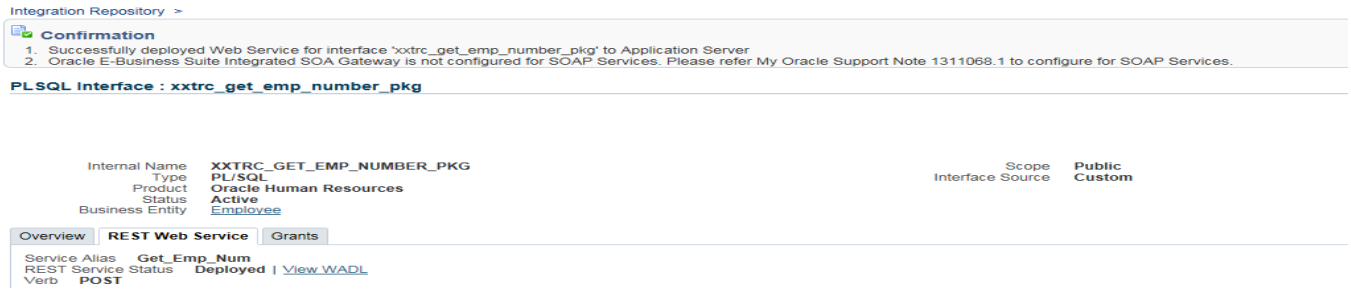
Step 6: Click on the search result Link, it will display List of method available in the Interface Package.



Click on **REST Web Service** Tab

This page will display the Return Employee Number method.

Step 7: Now select Return Employee Number method (we will create a user by invoking this Service once Deployed.) and enter Service Alias (This is mandatory) and then Click on Deploy button at bottom. You will receive message that service is successfully deployed and the status will be deployed.

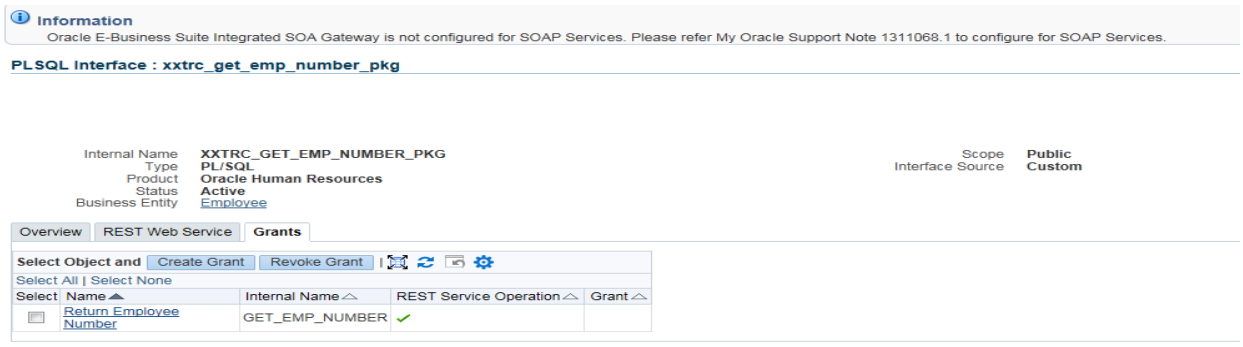


Step 8: Now you will see the link to WADL file , Click on the link . It shows the physical location of the service endpoint where the service is hosted.

```
<?xml version="1.0" encoding="UTF-8"?>
<application
xmlns:tns1="http://xmlns.oracle.com/apps/per/rest/get_emp_num/get_emp_number/
"                                    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://wadl.dev.java.net/2009/02"
xmlns:tns="http://xmlns.oracle.com/apps/per/soaprovider/plsql/rest/xxtrc_get
_emp_number_pkg/"
targetNamespace="http://xmlns.oracle.com/apps/per/soaprovider/plsql/rest/xxtr
c_get_emp_number_pkg/" name="XXTRC_GET_EMP_NUMBER_PKG"><grammars>
<include xmlns="http://www.w3.org/2001/XMLSchema"
href="http://xxx-xxx-
apdv01.cli.ad:8005/webservices/rest/Get_Emp_Num/?XSD=GET_EMP_NUMBER_SYNCH_TYP
EDEF.xsd"/></grammars>
<resources                                    base="http://xxx-xxx-
apdv01.cli.ad:8005/webservices/rest/Get_Emp_Num/"><resource
path="get_emp_number/"><method                                    name="POST"
id="GET_EMP_NUMBER"><request><representation      type="tns1:InputParameters"
mediaType="application/xml"/><representation      type="tns1:InputParameters"
mediaType="application/json"/></request><response><representation
type="tns1:OutputParameters"      mediaType="application/xml"/><representation
type="tns1:OutputParameters"
mediaType="application/json"/></response></method></resource></resources></ap
plication>
```

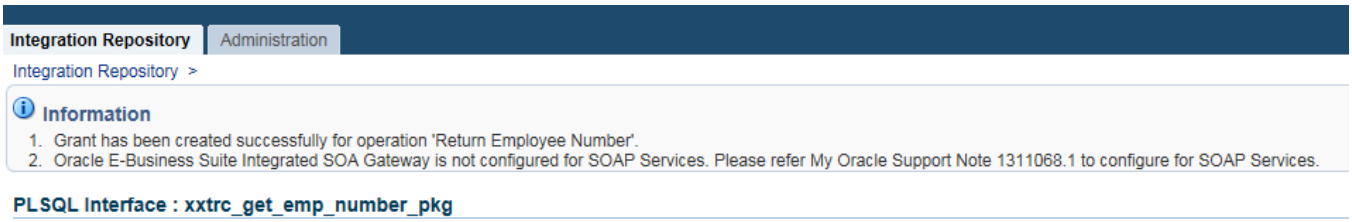
Now open the the link highlighted in Yellow in new Browser window, it will show the input parameters that will be used later for invoking the Web Service.

Step 9.Now Click on the Grants Tab, select return Employee Number and then click on Create Grant button



Step 10: Select a grantee type. In our Case Study we will enter SYSADMIN and then click on Create Grant Button.

It will display confirmation message:



To **Revoke Grants** click on Revoke Grants button under Grants Tab, it will display List of users with access for revoking the Grants.
To **Undeploy** the Web services select the method and click on undeploy under the REST Web Services Tab

Step 11: Now we will invoke this web service using Advanced REST Client apps available in Google Chrome.
Please refer to the steps from **case study 1** to invoke the web service. For this case study following details were used:

Link:- http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Get_Emp_Num/get_emp_number/

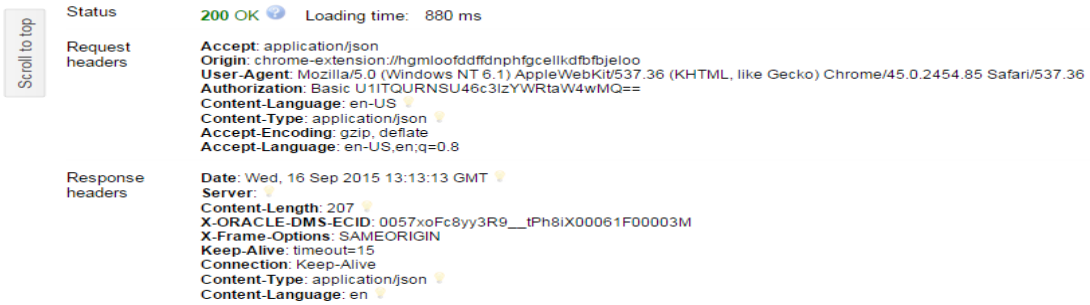
Payload:

```
{
 "Get_Employee_Number": {
 "@xmlns": "http://xmlns.oracle.com/apps/per/rest/Get_Emp_Num/get_emp_number",
"RESTHeader": {
 "xmlns": "http://xmlns.oracle.com/apps/per/rest/Get_Emp_Num/header",
 "Responsibility":"US_HRMS_MANAGER",
 "RespApplication":"PER",
 "SecurityGroup":"STANDARD",
 "NLSLanguage":"AMERICAN",
 "Org_Id" :"201"
 },
 "InputParameters": {
 "P_FIRST_NAME": "Kaul",
 "P_LAST_NAME": "Dipti"
 }
 }
 }
```

Step 12: Enter the payload information and click on send. You will receive attached response.

```
{
  "OutputParameters" : {
    "@xmlns:xsi" : "http://www.w3.org/2001/XMLSchema-instance",
```

```
    "@xmlns"                                                                    :
"http://xmlns.oracle.com/apps/per/rest/Get_Emp_Num/get_emp_number/",
    "GET_EMP_NUMBER" : "1"
  }
}
```



Exposing Pl/Sql as REST Web Service (Procedure Returning Table Type value)

Perform following steps to expose Pl/Sql as a web service

1. Create a Custom Package ,annotate it
2. Create iLDT File
3. Upload the iLDT File to Integration Repository.
4. Search API in Integration Repository
5. Deploy API as REST Service
6. Create Grants for REST Service
7. Invoke EBS REST Service (in JSON message format)
8. Verify the results

Here we will take an example of Getting Location details based on Location ID provided when invoking the Web service.

Following steps will explain the process of exposing the Custom API that returns **Table type** variable as output when invoked. The steps will remain same as used in **Case Study 2**. The package used for this Case Study is below:

Package Specification:

```
CREATE OR REPLACE PACKAGE xxtrc_hr_get_location_pkg

AS

  /* $Header: $ */

  /*#
```

```
  * This is Public Interface for Location and gives location information on the basis of Location ID.

  * @rep:scope public

  * @rep:product PER

  * @rep:displayname xxtrc_hr_get_location

  * @rep:lifecycle active

  * @rep:compatibility S

  * @rep:category BUSINESS_ENTITY Location

  */

 TYPE LOCATION_TBL IS TABLE OF HR_LOCATIONS%ROWTYPE

                  INDEX BY BINARY_INTEGER;


  /*#

  * Returns Location Information

  * @param P_LOCATION_ID NUMBER LOCATION_ID

  * @param P_LOCATION_TBL LOCATION_TBL LOCATION_TBL

  * @rep:scope public

  * @rep:lifecycle active

  * @rep:displayname Returns Location Details

  */


   PROCEDURE get_location_details (P_LOCATION_ID IN NUMBER , P_LOCATION_TBL OUT
LOCATION_TBL);

END xxtrc_hr_get_location_pkg;
```

Package Body:

```
CREATE OR REPLACE PACKAGE BODY xxtrc_hr_get_location_pkg
AS
```

```
   PROCEDURE get_location_details (P_LOCATION_ID IN NUMBER
,P_LOCATION_TBL OUT LOCATION_TBL)
   AS


      l_return_tbl        LOCATION_TBL;
      l_return_tbl_null   LOCATION_TBL;
      l_count                  NUMBER;

      CURSOR my_cur (P_LOCATION_ID IN NUMBER)
      IS
         SELECT *
           FROM HR_LOCATIONS
          WHERE LOCATION_ID = P_LOCATION_ID;
   BEGIN
      FOR rec IN my_cur (P_LOCATION_ID)
      LOOP
         l_count := 1;
         l_return_tbl (l_count).LOCATION_ID := rec.LOCATION_ID;
         l_return_tbl (l_count).LOCATION_CODE := rec.LOCATION_CODE;
         l_return_tbl (l_count).LOCATION_USE := rec.LOCATION_USE;
         l_return_tbl (l_count).BUSINESS_GROUP_ID :=
rec.BUSINESS_GROUP_ID;
         l_return_tbl (l_count).DESCRIPTION := rec.DESCRIPTION;
         l_return_tbl (l_count).SHIP_TO_LOCATION_ID :=
rec.SHIP_TO_LOCATION_ID;
         l_return_tbl (l_count).SHIP_TO_SITE_FLAG :=
rec.SHIP_TO_SITE_FLAG;
         l_return_tbl (l_count).RECEIVING_SITE_FLAG :=
rec.RECEIVING_SITE_FLAG;
         l_return_tbl (l_count).BILL_TO_SITE_FLAG :=
rec.BILL_TO_SITE_FLAG;
         l_return_tbl (l_count).IN_ORGANIZATION_FLAG :=
rec.IN_ORGANIZATION_FLAG;
         l_return_tbl (l_count).OFFICE_SITE_FLAG :=
rec.OFFICE_SITE_FLAG;
         l_return_tbl (l_count).DESIGNATED_RECEIVER_ID :=
rec.DESIGNATED_RECEIVER_ID;
         l_return_tbl (l_count).INVENTORY_ORGANIZATION_ID :=
            rec.INVENTORY_ORGANIZATION_ID;
         l_return_tbl (l_count).TAX_NAME := rec.TAX_NAME;
         l_return_tbl (l_count).INACTIVE_DATE := rec.INACTIVE_DATE;
         l_return_tbl (l_count).STYLE := rec.STYLE;
         l_return_tbl (l_count).ADDRESS_LINE_1 := rec.ADDRESS_LINE_1;
         l_return_tbl (l_count).ADDRESS_LINE_2 := rec.ADDRESS_LINE_2;
         l_return_tbl (l_count).ADDRESS_LINE_3 := rec.ADDRESS_LINE_3;
         l_return_tbl (l_count).TOWN_OR_CITY := rec.TOWN_OR_CITY;
         l_return_tbl (l_count).COUNTRY := rec.COUNTRY;
         l_return_tbl (l_count).POSTAL_CODE := rec.POSTAL_CODE;
         l_return_tbl (l_count).REGION_1 := rec.REGION_1;
         l_return_tbl (l_count).REGION_2 := rec.REGION_2;
         l_return_tbl (l_count).REGION_3 := rec.REGION_3;
```

```
         l_return_tbl (l_count).TELEPHONE_NUMBER_1 :=
rec.TELEPHONE_NUMBER_1;
         l_return_tbl (l_count).TELEPHONE_NUMBER_2 :=
rec.TELEPHONE_NUMBER_2;
         l_return_tbl (l_count).TELEPHONE_NUMBER_3 :=
rec.TELEPHONE_NUMBER_3;
         l_return_tbl (l_count).LOC_INFORMATION13 :=
rec.LOC_INFORMATION13;
         l_return_tbl (l_count).LOC_INFORMATION14 :=
rec.LOC_INFORMATION14;
         l_return_tbl (l_count).LOC_INFORMATION15 :=
rec.LOC_INFORMATION15;
         l_return_tbl (l_count).LOC_INFORMATION16 :=
rec.LOC_INFORMATION16;
         l_return_tbl (l_count).LOC_INFORMATION17 :=
rec.LOC_INFORMATION17;
         l_return_tbl (l_count).LOC_INFORMATION18 :=
rec.LOC_INFORMATION18;
         l_return_tbl (l_count).LOC_INFORMATION19 :=
rec.LOC_INFORMATION19;
         l_return_tbl (l_count).LOC_INFORMATION20 :=
rec.LOC_INFORMATION20;
         l_return_tbl (l_count).ATTRIBUTE_CATEGORY :=
rec.ATTRIBUTE_CATEGORY;
         l_return_tbl (l_count).ATTRIBUTE1 := rec.ATTRIBUTE1;
         l_return_tbl (l_count).ATTRIBUTE2 := rec.ATTRIBUTE2;
         l_return_tbl (l_count).ATTRIBUTE3 := rec.ATTRIBUTE3;
         l_return_tbl (l_count).ATTRIBUTE4 := rec.ATTRIBUTE4;
         l_return_tbl (l_count).ATTRIBUTE5 := rec.ATTRIBUTE5;
         l_return_tbl (l_count).ATTRIBUTE6 := rec.ATTRIBUTE6;
         l_return_tbl (l_count).ATTRIBUTE7 := rec.ATTRIBUTE7;
         l_return_tbl (l_count).ATTRIBUTE8 := rec.ATTRIBUTE8;
         l_return_tbl (l_count).ATTRIBUTE9 := rec.ATTRIBUTE9;
         l_return_tbl (l_count).ATTRIBUTE10 := rec.ATTRIBUTE10;
         l_return_tbl (l_count).ATTRIBUTE11 := rec.ATTRIBUTE11;
         l_return_tbl (l_count).ATTRIBUTE12 := rec.ATTRIBUTE12;
         l_return_tbl (l_count).ATTRIBUTE13 := rec.ATTRIBUTE13;
         l_return_tbl (l_count).ATTRIBUTE14 := rec.ATTRIBUTE14;
         l_return_tbl (l_count).ATTRIBUTE15 := rec.ATTRIBUTE15;
         l_return_tbl (l_count).ATTRIBUTE16 := rec.ATTRIBUTE16;
         l_return_tbl (l_count).ATTRIBUTE17 := rec.ATTRIBUTE17;
         l_return_tbl (l_count).ATTRIBUTE18 := rec.ATTRIBUTE18;
         l_return_tbl (l_count).ATTRIBUTE19 := rec.ATTRIBUTE19;
         l_return_tbl (l_count).ATTRIBUTE20 := rec.ATTRIBUTE20;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE_CATEGORY :=
            rec.GLOBAL_ATTRIBUTE_CATEGORY;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE1 :=
rec.GLOBAL_ATTRIBUTE1;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE2 :=
rec.GLOBAL_ATTRIBUTE2;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE3 :=
rec.GLOBAL_ATTRIBUTE3;
```

```
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE4 :=
rec.GLOBAL_ATTRIBUTE4;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE5 :=
rec.GLOBAL_ATTRIBUTE5;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE6 :=
rec.GLOBAL_ATTRIBUTE6;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE7 :=
rec.GLOBAL_ATTRIBUTE7;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE8 :=
rec.GLOBAL_ATTRIBUTE8;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE9 :=
rec.GLOBAL_ATTRIBUTE9;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE10 :=
rec.GLOBAL_ATTRIBUTE10;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE11 :=
rec.GLOBAL_ATTRIBUTE11;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE12 :=
rec.GLOBAL_ATTRIBUTE12;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE13 :=
rec.GLOBAL_ATTRIBUTE13;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE14 :=
rec.GLOBAL_ATTRIBUTE14;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE15 :=
rec.GLOBAL_ATTRIBUTE15;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE16 :=
rec.GLOBAL_ATTRIBUTE16;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE17 :=
rec.GLOBAL_ATTRIBUTE17;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE18 :=
rec.GLOBAL_ATTRIBUTE18;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE19 :=
rec.GLOBAL_ATTRIBUTE19;
         l_return_tbl (l_count).GLOBAL_ATTRIBUTE20 :=
rec.GLOBAL_ATTRIBUTE20;
         l_return_tbl (l_count).LAST_UPDATE_DATE :=
rec.LAST_UPDATE_DATE;
         l_return_tbl (l_count).LAST_UPDATED_BY :=
rec.LAST_UPDATED_BY;
         l_return_tbl (l_count).LAST_UPDATE_LOGIN :=
rec.LAST_UPDATE_LOGIN;
         l_return_tbl (l_count).CREATED_BY := rec.CREATED_BY;
         l_return_tbl (l_count).CREATION_DATE := rec.CREATION_DATE;
         l_return_tbl (l_count).ENTERED_BY := rec.ENTERED_BY;
         l_return_tbl (l_count).TP_HEADER_ID := rec.TP_HEADER_ID;
         l_return_tbl (l_count).ECE_TP_LOCATION_CODE :=
rec.ECE_TP_LOCATION_CODE;
         l_return_tbl (l_count).OBJECT_VERSION_NUMBER :=
rec.OBJECT_VERSION_NUMBER;
         l_return_tbl (l_count).LEGAL_ADDRESS_FLAG :=
rec.LEGAL_ADDRESS_FLAG;
         l_return_tbl (l_count).TIMEZONE_CODE := rec.TIMEZONE_CODE;
```

```
         l_count := l_count + 1;
      END LOOP;

      P_LOCATION_TBL:= l_return_tbl;
   EXCEPTION
      WHEN NO_DATA_FOUND
      THEN
         P_LOCATION_TBL:= l_return_tbl_null;
      WHEN TOO_MANY_ROWS
      THEN
         P_LOCATION_TBL:=l_return_tbl_null;
   END get_location_details;
END xxtrc_hr_get_location_pkg;
```

Step 1: First create a custom package xxtrc_hr_get_location_pkg with help of attachment below with .pls extension and annotate it.

Step 2: Upload the .pls file to Oracle server. For our case study file was uploaded to custom_top Bin folder

Step 3: Create iLDT file by using following command and current Directory is where the .pls file was uploaded:

$IAS_ORACLE_HOME/perl/bin/perl $FND_TOP/bin/irep_parser.pl -g -v -username=sysadmin PER:patch/115/sql:xxtrc_hr_get_location_pkg.pls:12.0=xxtrc_hr_get_location_pkg.pls

Step 4: Upload the generated iLDT file to Integration repository by using following command:

$FND_TOP/bin/FNDLOAD apps/apps 0 Y UPLOAD $FND_TOP/patch/115/import/wfirep.lct xxtrc_hr_get_location_pkg_pls.ildt

Step 5: Navigate to Integration Repository Responsibility, Click on Search. Then on search Page enter "xxtrc_hr_get_location_pkg" in Internal Name and click on Go

Step 6: Click on the search result Link, it will display List of method available in the Interface Package. Click on **REST Web Service** Tab.

Step 7: Now select returns Location method (we will create a user by invoking this Service once Deployed.) and enter Service Alias (This is mandatory) and then Click on Deploy button at bottom. You will receive message that service is successfully deployed and the status will be deployed.

Step 8: Now you will see the link to WADL file, Click on the link. It shows the physical location of the service endpoint where the service is hosted.

```
<?xml version="1.0" encoding="UTF-8"?>
<application
xmlns:tns1="http://xmlns.oracle.com/apps/per/rest/get_location_details/get_location_details/"                    xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

xmlns="http://wadl.dev.java.net/2009/02"
xmlns:tns="http://xmlns.oracle.com/apps/per/soaprovider/plsql/rest/xxtrc_hr_get_location_pkg/"
targetNamespace="http://xmlns.oracle.com/apps/per/soaprovider/plsql/rest/xxtrc_hr_get_location_pkg/" name="XXTRC_HR_GET_LOCATION_PKG"><grammars><include
xmlns="http://www.w3.org/2001/XMLSchema"
href="http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Get_Location_Details/?XSD=GET_LOCATION_DETAILS_SYNCH_TYPEDEF.xsd"/>
</grammars><resources base="http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Get_Location_Details/"><resource
path="get_location_details/"><method name="POST"
id="GET_LOCATION_DETAILS"><request><representation
type="tns1:InputParameters" mediaType="application/xml"/><representation
type="tns1:InputParameters"
mediaType="application/json"/></request><response><representation
type="tns1:OutputParameters" mediaType="application/xml"/><representation
type="tns1:OutputParameters"
mediaType="application/json"/></response></method></resource></resources></application> Now open the the link highlighted in Yellow in new Browser window, it will show the input parameters that will be used later for invoking the Web Service.

Step 9.Now Click on the Grants Tab, select return Employee Number and then click on Create Grant button

Step 10: Select a grantee type. In our Case Study we will enter SYSADMIN and then click on Create Grant Button.

Step 11: Now we will invoke this web service using Advanced REST Client apps available in Google Chrome.
Please refer to the steps from case study 1 to invoke the web service. For this case study following details was used:

Link:- http://xxx-xxx-apdv01.cli.ad:8005/webservices/rest/Get_Location_Details/get_location_details/

Payload:

```
{
 "Get_Location_Details": {
 "@xmlns": "http://xmlns.oracle.com/apps/per/rest/Get_Location_Details/get_location_details",
 "RESTHeader": {
 "xmlns": "http://xmlns.oracle.com/apps/per/rest/Get_Location_Details/header",
 "Responsibility":"US_HRMS_MANAGER",
 "RespApplication":"PER",
 "SecurityGroup":"STANDARD",
 "NLSLanguage":"AMERICAN",
 "Org_Id" :"201"
 },
 "InputParameters": {
 "P_LOCATION_ID": "421" } } }
```

Step 12: Enter the payload information and click on send. You will receive status and response.

Conclusion

This document gives you an overview of the Integrated SOA Gateway and REST Web Services .It also gives you step by step instructions on exposing Pl/Sql as REST Web from scratch. You should now be able to create REST Web Service either using Standard API or Custom API.

References

1. Oracle E-Business Suite Integrated SOA Gateway Implementation Guide

2. https://blogs.oracle.com/stevenChan/entry/how_to_publish_pl_sql