https://lh4.googleusercontent.com/UYur34TGchvgOG99OH-GSJ_mA9xgj8kUv-sTJYu73kvMNED8SlDmfIBoqLMNLTer_jU1QbRtgByL25csWPeQ8xsXgL65ihPF-HuDk-hs-J0KrNBycbTkAdD0Zq0ver2cbdmq9L6R

**Blog Template:**

Your name:Shrikant M Ghuge

Your email address: ghuge.shrikant@gmail.com

Title:

|  |
| --- |
| Creating directory with SFTP/FTP connector |

Keywords / Tags (separate by commas):

|  |
| --- |
| SFTP, FTP, Directory with SFTP, Folder with SFTP, Folder with FTP |

Body: (Please make sure your blog post is at least 3 paragraphs (500-1,000 words) long and sufficiently covers the subject. If you submit code snippets, please make sure they are clearly labeled with the order that they should appear in the post.)

|  |
| --- |
| While working with Mule ESB, I came across situation where I have to send and receive User defined type with the help of stored procedure, The problem statement and solution is given below :  Problem statement:  call my\_pkg.get\_few\_files(:1,:2,:3)  Where,  1 : IN (Integer)  2 : IN (DBUSR.MY\_OWN\_TYPE)  3 : OUT (DBUSR.MY\_OWN\_TYPE)  You cannot send user defined type in mule Database connector. As shown in fig 1.1 there is no provision for user defined types.  Solution :  We have mule expression language, with the help of which we can call the procedure and get the desired response.  Steps are as :  1. Create method in mule expression global function.  2. Call the method from any variable  2. Send data and application registry details to the defined method.  3. Make connection using application registry from method and call the procedure and return the result to the variable from which the method was called and do further manipulation on resultant data.  **Mule example is :**  <?xmlversion=*"1.0"*encoding=*"UTF-8"*?>  <mulexmlns:db=*"http://www.mulesoft.org/schema/mule/db"*  xmlns:http=*"http://www.mulesoft.org/schema/mule/http"*xmlns=*"http://www.mulesoft.org/schema/mule/core"*xmlns:doc=*"http://www.mulesoft.org/schema/mule/documentation"*  xmlns:spring=*"http://www.springframework.org/schema/beans"*  xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*  xsi:schemaLocation=*"http://www.mulesoft.org/schema/mule/db http://www.mulesoft.org/schema/mule/db/current/mule-db.xsd*  *http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-current.xsd*  *http://www.mulesoft.org/schema/mule/core http://www.mulesoft.org/schema/mule/core/current/mule.xsd*  *http://www.mulesoft.org/schema/mule/http http://www.mulesoft.org/schema/mule/http/current/mule-http.xsd"*>  <http:listener-configname=*"HTTP\_Listener\_Configuration"*host=*"0.0.0.0"*port=*"8081"*basePath=*"dbproc"*doc:name=*"HTTP Listener Configuration"*/>  <db:oracle-configname=*"Oracle\_Configuration"*  doc:name=*"Oracle Configuration"*dataSource-ref=*"dataSource"*>  </db:oracle-config>  <configurationdoc:name=*"Configuration"*>  <expression-languageautoResolveVariables=*"false"*>  <global-functions>  deffetchDBData(requestTypes,reg) {  javax.sql.DataSourceds = (javax.sql.DataSource)reg.get("dataSource");  oracle.jdbc.OracleConnectioncon = (oracle.jdbc.OracleConnection)ds.getConnection();  oracle.sql.ArrayDescriptorresourceArrayList = oracle.sql.ArrayDescriptor.createDescriptor("ALL\_FILELIST", con); <!-- define user defined in parameters -->  oracle.sql.ARRAYresourceArray = new oracle.sql.ARRAY(resourceArrayList, con , requestTypes);  java.sql.CallableStatementstmt = con.prepareCall("begin file\_pkg.get\_all\_file\_details(:1,:2,:3); end;");  stmt.setObject(1, resourceArray);  stmt.registerOutParameter(2, 2003, "ADM.GET\_FILEDETAILS\_COLL" ); <!-- Register user defined out parameters -->    stmt.execute();  oracle.sql.ARRAYouterArray = null;  Object[] outerObjList = null;  if(stmt.getArray(2)!=null){  outerArray = (oracle.sql.ARRAY)stmt.getArray(2);  outerObjList = (Object[]) outerArray.getArray();  oracle.sql.STRUCTfirstResultStruct= (oracle.sql.STRUCT)outerObjList[0];  Object[] firstResultObject = firstResultStruct.getAttributes();  }  return {outerObjList,stmt.getString(3)};  }    </global-functions>  </expression-language>  </configuration>  <spring:beans>  <!-- Spring bean for DB connectivity -->  <spring:beanid=*"dataSource"*name=*"dataSource"*class=*"oracle.jdbc.pool.OracleDataSource"*>  <spring:propertyname=*"password"*value=*"XXXXXX"*/>  <spring:propertyname=*"URL"*value=*"jdbc:oracle:thin:@//testxxxx:1111/DBDETAI.domain.com"*/>  <spring:propertyname=*"user"*value=*"anyone"*/>  <spring:propertyname=*"dataSourceName"*value=*"ds"*/>  </spring:bean>  </spring:beans>  <flowname=*"dbproccallFlow"*>  <http:listenerconfig-ref=*"HTTP\_Listener\_Configuration"*path=*"call"*allowedMethods=*"GET"*doc:name=*"HTTP"*/>  <set-variablevariableName=*"callDBProc"*value=*"#[fetchDBData(flowVars.requestTypes,app.registry)]"*doc:name=*"callDBProc"*/>  <set-payloadvalue=*"#[flowVars.callDBProc[1]]"*doc:name=*"Set Payload"*/>  </flow>  </mule>  NOTE : Please make ojdbc driver available in classpath |
|  |

Image (minimum 1 image required):

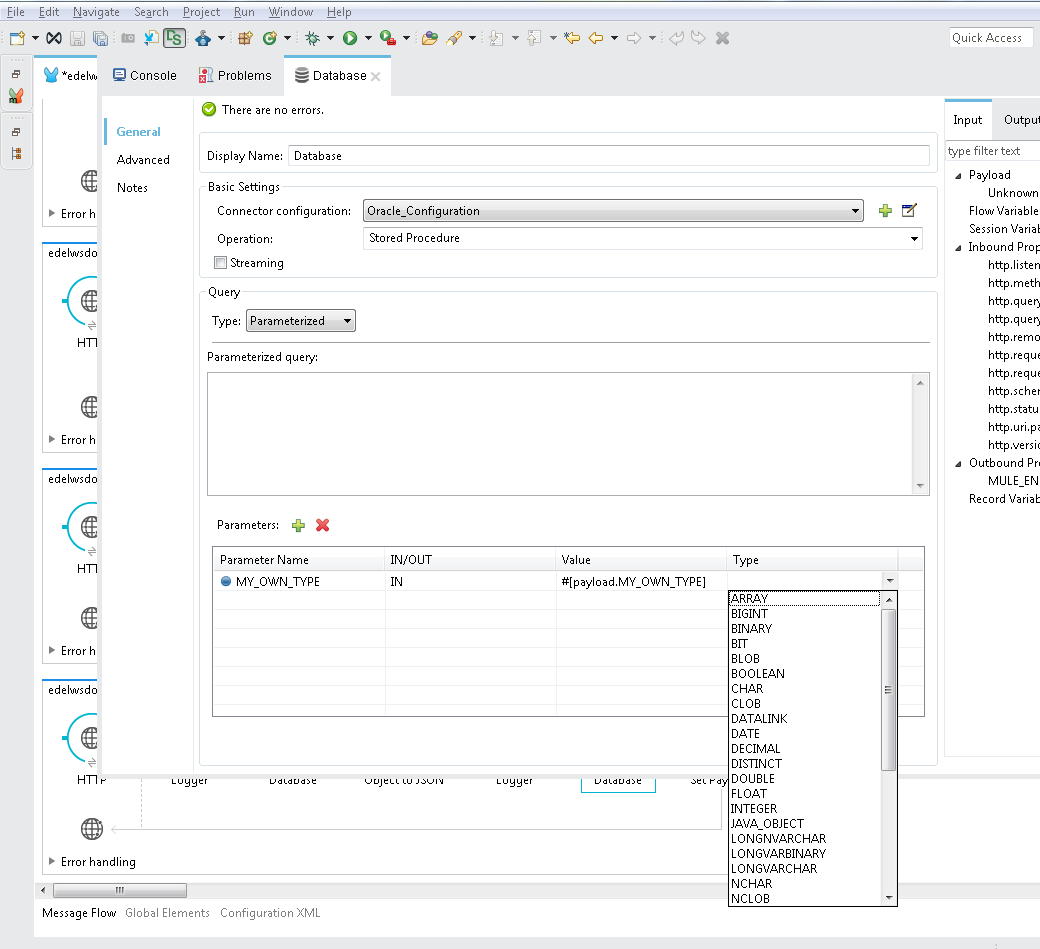


Fig 1.1

Categories (check all that apply):

|  |  |
| --- | --- |
| * Anypoint Platform * **Anypoint Studio** * API * CloudHub * Connectivity * Connectors and Templates * Design * Events * Industries * Life at MuleSoft | * Management * Mule * **MuleSoft** * News * **Resources** * SaaS * SOA * Tech Ramblings * Training and How to * Trends |

Thank you for being part of the Champions Program!