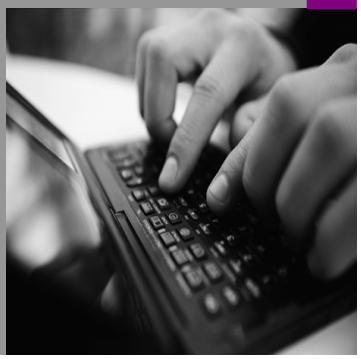
How-to Guide
SAP xApps



# How To… Integrate xMII with the Business Information Warehouse

Version 2.10 – March 2006

Applicable Releases:
SAP xApps
SAP Netweaver '04
SAP Netweaver '04s

Author: Salvatore Castro

SAP Library document classification: PUBLIC

# 1 Business Scenario

Integrating SAP xApp Manufacturing Integration and Intelligence (SAP xMII) and the Business Information Warehouse (BW or BI).

# 2 Introduction

This type of integration will allow xMII to access data stored in the Business Warehouse via OLAP queries through the XMLA interface. This document will outline the proper steps required to create a connection to the Warehouse and also ideas and tips on how to use the data being returned.

The xml format of the data being returned from the warehouse contains additional summation fields that must be accounted for to properly display the data. As a result of this, knowledge of Business Logic Services and XPath should be well known.

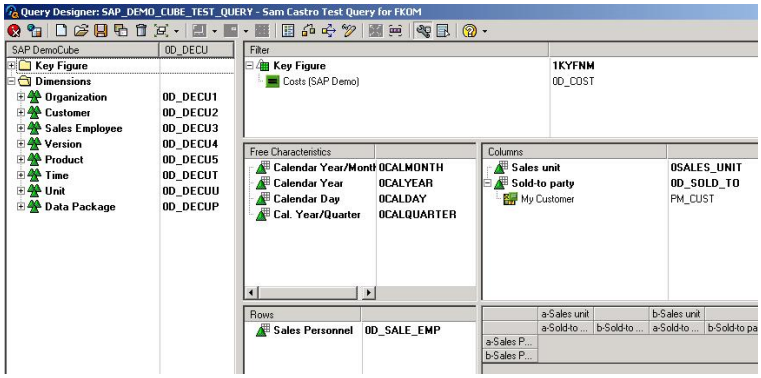
Additionally, a Cube should already be selected to create your queries against and previous working knowledge of BEx or Query Designer.

# 3 The Step By Step Solution Using OLAP (MDX Query)

The steps to setup this connection start off with creating a remote enabled query in BI. Then using the xmla interface we can use MDX to query data from this remote enable query.

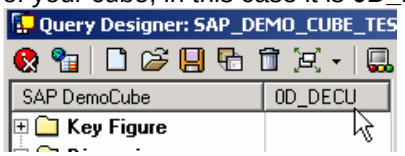
## 3.1 Remote Enable Query

1. Create a Query via the 'Query Designer'. Please take note of the PM_CUST value under the Columns definition, as this will be referenced later.

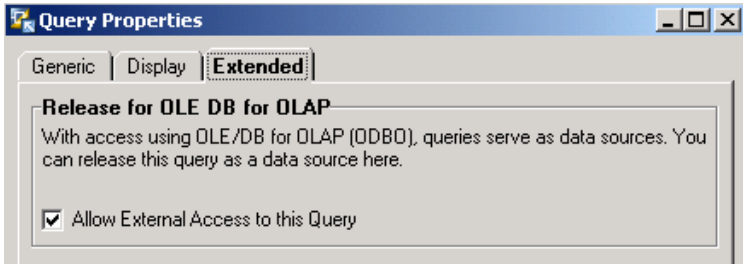



2. Press the Show Technical Names Icon,  to display internal names.

3. Take note of the internal system name of your cube, in this case it is **0D_DECU**



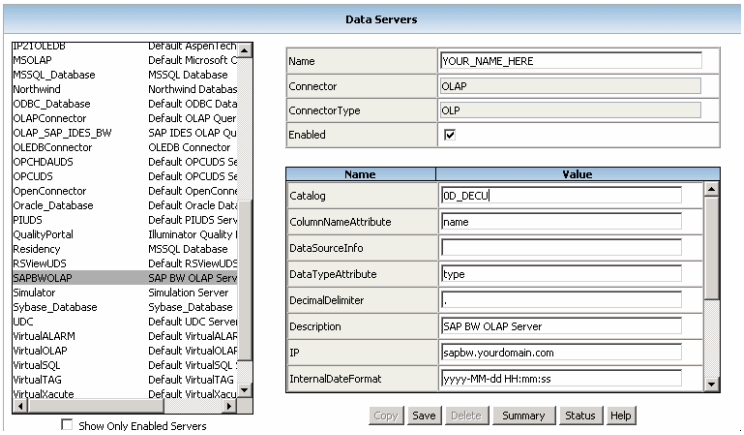

4. This example uses the Sales Demo Cube in the IDES longterm landscape. Now that your query is created. To remote Enable your query. This menu is found by clicking on the 'Query Properties' icon,  , and selecting the Extended Tab.




5. Save your Query 

## 3.2 Setup a New 'SAPBWOLAP' Connection in xMII

6. Login to the xMII Server. Navigate to Data Services->Data Servers. Deselect the 'Show Only Enabled Servers' checkbox and scroll to find the Name = **SAPBWOLAP**. Press the copy button on the right and Name and Enable your connection.

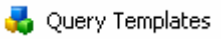



7. Proceed to fill in the remaining values of your connection so that it fits your particular BW system. On the right is an example setup for the 0D_DECU cube.

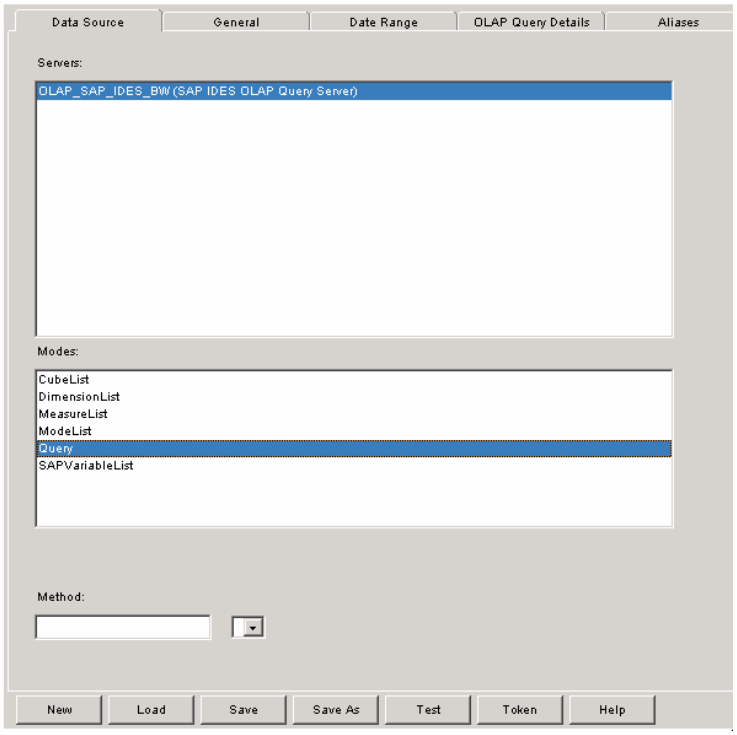
8. Press the save button to store your OLAP connection. This connection will not show up in the Status window as it is not a persistent connection to BI but rather an on demand connection.

| Name | Value |
|---|---|
| Catalog | 0D_DECU |
| ColumnNameAttribute | name |
| Connector | OLAP |
| ConnectorType | OLP |
| DataSourceInfo | |
| DataTypeAttribute | type |
| DecimalDelimiter | . |
| Description | SAP BW OLAP Query Server |
| Enabled | true |
| IP | scphl304.phl.sap.corp |
| InternalDateFormat | yyyy-MM-dd HH:mm:ss |
| Name | OLAP_BW_Test |
| Port | 54080 |
| ResultInNamespace | true |
| ServerPackage | com.lighthammer.Illuminator.connectors.OLAP |
| UserName | sapxmii |
| WebService | /sap/bw/xml/soap/xmla |

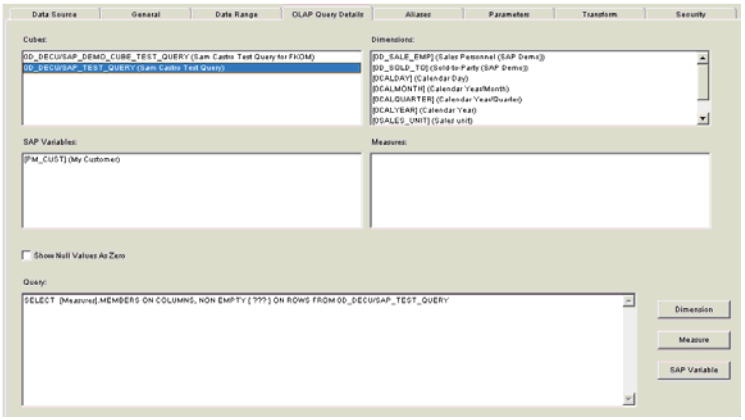
## 3.3 Building a Query

9. Under the Data Services tab select the Query Templates icon 

. To open the Query Template Editor and Select new OLAP Query. Select the OLAP Server that was created in step 8 and select Query mode. The press the OLAP Query Details tab to create your Query.




10. Notice in the SAP Variables window the value, “PM_CUST” and its description. Along with all of the dimensions from Query Designer. In order to ensure that your MDX statements are properly formed use the ‘mdxtest’ transaction in the BI SAP GUI to validate and then copy the MDX into the Query.
11. The alias tab can be used to change the column headers of the return values from ‘C000001’ to ‘Aliased Name’.



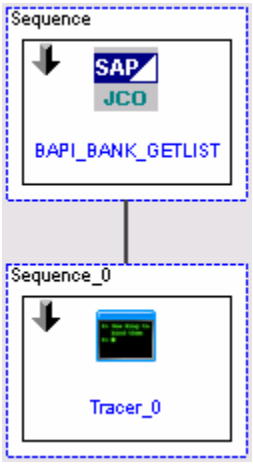
# 4 The Step By Step Solution Using a Business Logic (JCO BAPI Call)

The steps to setup this connection start off with knowing which BAPI call you would like to access in BI.

## 4.1 Setup the JCO Block

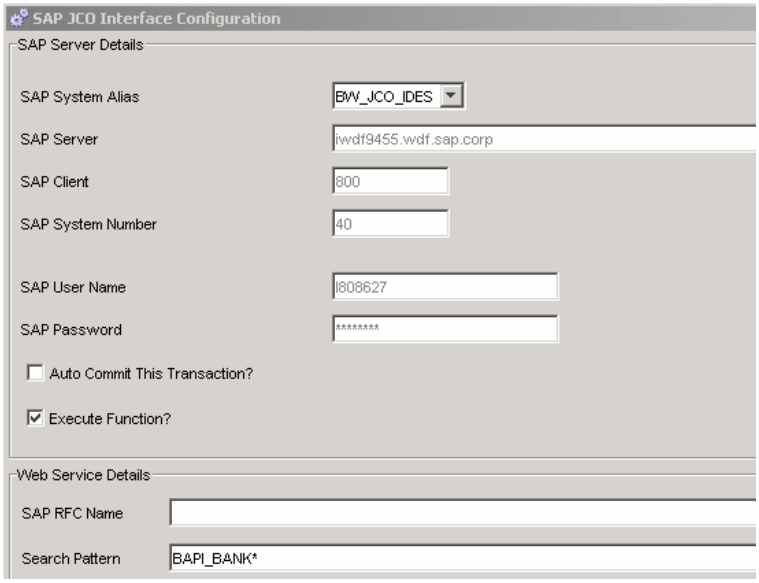
1. Create a new Business Logic Transaction by Opening up the Logic Editor:


2. This example uses the BAPI call BAPI_BANK_GETLIST to return the list of banks currently defined in the Warehouse. Create the following Transaction by loading a SAP JCO block and tracer block. A good standard it to name the Action block after the BAPI call that it is making.

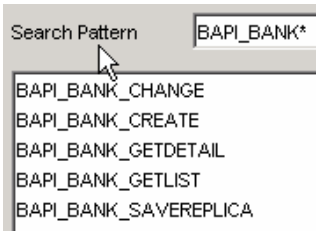


## 4.2 Configuring the JCO Block

3. Use a configuration similar to the one shown on the right, as it applies to your BI Server.



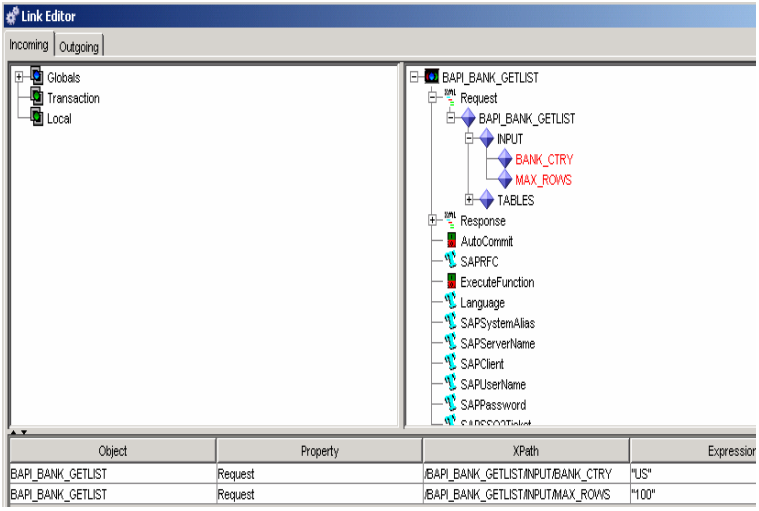

4. As a test, type in BAPI_BANK* into the Search Pattern textbox and press the Get List button. This should bring back a list of available BAPI calls:




5. Press the OK button once you have selected the GETLIST BAPI call.

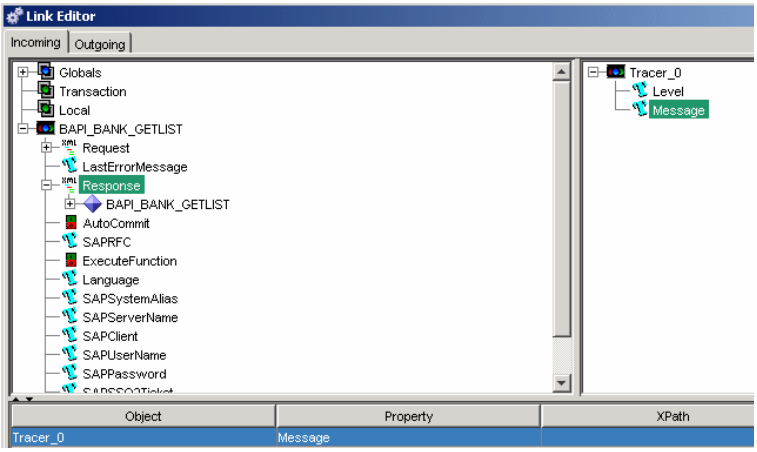
6. In the Link Editor for the JCO block, assign the value “US” to the “BANK_CTRY” node and the value “100” to the “MAX_ROWS” node. This will bring back the first 100 banks located in any country using the 2 letter country code.

7. Once this is completed close the link editor.

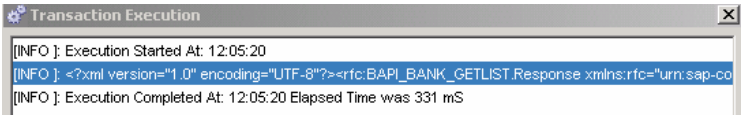


## 4.3 Viewing the Results of Your BI BAPI Call

8. Open up the link editor for the tracer action block and assign the BAPI_BANK_GETLIST.Response XML to the Message string of the tracer. Once this is done close the link editor.




9. Now it is time to test your transaction. Execute the transaction and double click on the xml in the transaction execution window. This will copy the entire string, including [INFO]:, into your windows clipboard. Then paste this into your favorite XML editor and remove the [INFO]: from the beginning. A sample of possible responses is shown below in section 5.1.



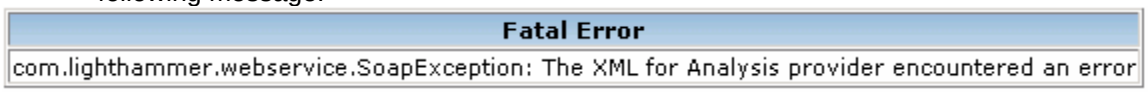
# 5 Appendix (Important to Note)

## 5.1 BAPI Call XML Result Sample

```
<?xml version="1.0" encoding="utf-8"?>
<BAPI_BANK_GETLIST>
   <INPUT>
      <BANK_CTRY>US</BANK_CTRY>
      <MAX_ROWS>100</MAX_ROWS>
   </INPUT>
   <OUTPUT>
      <RETURN>
         <TYPE>E</TYPE>
         <ID>BF00</ID>
         <NUMBER>007</NUMBER>
         <MESSAGE>No banks exist for country</MESSAGE>
         <LOG_NO />
         <LOG_MSG_NO>000000</LOG_MSG_NO>
         <MESSAGE_V1 />
         <MESSAGE_V2 />
         <MESSAGE_V3 />
         <MESSAGE_V4 />
         <PARAMETER />
         <ROW>0</ROW>
         <FIELD />
         <SYSTEM>B3TCLNT800</SYSTEM>
      </RETURN>
   </OUTPUT>
   <TABLES>
      <BANK_LIST />
   </TABLES>
</BAPI_BANK_GETLIST>
```

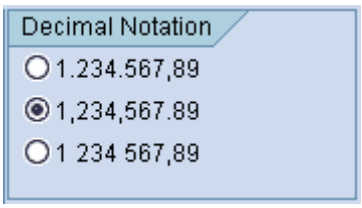
## 5.2 Trouble Shooting Tips

1. An important point to note is that if the MDX statement is malformed you will not receive the error message back from BI indicating what the problem is. Instead you will receive the following message:




   This message is NOT an internal xMII error but rather due to the fact that the message returned from the BI SOAP interface is in a non-xmla standard format. To get the correct error message copy and paste your MDX statement into the **mdxtest** transaction window and execute the query.

2. If the numerical values that you are receiving back from xMII appear to me truncating values and moving the decimal place to the left. Then the issue is probably with the profile setup for the user that the xMII OLAP connection is using not be compatible with the Stylesheets. The xMII Stylesheets are using the following number format 1,234,567.89 and not 1.234.567,00 and will cause numerical mismatches with the data in BW and the data in the queries. The profile for the user can be changed by logging into BW SAP GUI and going to **System** -> **User Profile** -> **Own Data** and clicking on the **Defaults** tab and then selecting the new format:




3. To troubleshoot XMLA connection issues please refer to the troubleshooting guide in OSS note 929015. This note will be kept as up to date as possible to accommodate any questions or issues that may arise when attempting to connect.

www.sap.com/xapps