Creating an XML File with PeopleTools using the Rowset Method

PT 8.50+
Randall Groncki

Introduction

Several tools in PeopleSoft use and XML file as a data source. Prominent among these is the BI Publisher reporting tool. We can use PeopleTools to quickly generate the data, enrich and shape it before sending it on to the receiving utility.

This document will discuss the Rowset method of creating an XML File using PeopleTools and PeopleCode.

The process in brief:

1. Create and load a rowset
2. Enrich the data in that rowset
3. Hand the data in the rowset to a delivered PeopleTools Application Package which returns a formatted XML String
4. Write that string to a file

For demonstration purposes, our premise is creating an XML file for a BI Publisher report. However, there are other situations requiring XML Files. This tool can be used if the target schema and tag names are flexible, can be replicated in the target file or the developer applies a transformation which would convert the structure to the target schema.

XML File Creation

Since PeopleTools version 8.50, BI Publisher reports can only use XML Files, Queries and Connected Queries as data sources for reports. Prior versions of the tool also included options for XMLDoc & Rowsets Objects as data sources. Though no new data sources may be defined as type Rowset or XMLDoc, backward compatibility allows BI Publisher reports created in earlier tools versions using these objects to continue functioning.

Since all versions use XML Files as a data source, this demonstration will focus on generating XML Files using the Rowset Object

Premise for all XML File Creation Examples.

In order to demonstrate the differences multiple XML File creation techniques, all demonstrations will use the same report premise: A letter to all employees summarizing their current job and showing all training taken through the company. This is a complex data set with a parent/child relationship. Each employee will get a separate page in the report. For each employee, there will be zero or more training entries. See Appendix A for the class populating this data structure. The class returns this Rowset object:

Root

Employee Record: X_EE_RPT_LTR_VW (Parent Record)

- EMPLID
- NAME
- DEPTID
- LOCATION
- JOBCODE
- COMPRATE
- CHANGE_PCT
- LOCATION_DESCR
- JOB_DESCR
- COMPANY_DESCR
- X_EMPLOYEE_PHOTO

Training Record: X_TRAINING_VW (Child Record)

- EMPLID
- COURSE_START_DT
- COURSE
- SESSION_NBR
- COURSE_TITLE
- COURSE_END_DT
- ATTENDANCE
- TRAINING_REASON
- COURSE_GRADE
- PREREQ_MET
- DEPTID
- BUSINESS_UNIT

*Note: Per MOS Doc ID: Doc ID 962712.1:

> PSQuery and rowset data sources may not be the most efficient way to generate the XML file that is used as input to BI Publisher, (XML Publisher). Consider using SQR or other mechanisms to generate the XML file

Rowset method of creating an XML File

The Rowset Method uses delivered PeopleCode Application Package “PSXP_XMLGEN” to generate both the XML File and XML Schema file (XSD). In the example below, the data Rowset is passed to the class method “getXSDSchema” and “getXMLData”. The method result is a string containing a XML or XSD file. Write this string to the file object with one write statement.

It’s important to know that the XML File generated with this method has field tags with “fld_” appended to the front of the field names. Example: EMPLID becomes tag “fld_EMPLID” in the XML File. Row and Row Set objects also get identifiers appended to the tags. This has to be accounted for in your template mapping.

This example creates both the schema and XML file.

- See appendix B for the XML File generated
- See appendix C for the XSD file generated

```
import PSXP_XMLGEN:*;
import PSXP_RPTDEFNMANAGER:*;
import X_BI_PUB_PCODE:LoadTestData;

Local PSXP_XMLGEN:RowSetDS &oXML_GENERATOR;
Local File &XSD_File, &XML_File;
Local Rowset &RS_Employee;
Local string &XML_Filename_path, &XSD_Filename_path, &my_schema, &my_xml;
Local integer &i;

/* Load data into rowset */
Local X_BI_PUB_PCODE:LoadTestData &LoadTestData = create
X_BI_PUB_PCODE:LoadTestData();
&RS_Employee = &LoadTestData.LoadTestDataSet();

/* output files */
/* create xsd */
&oXML_GENERATOR = create psxp_xmlgen:RowSetDS();
&my_schema = &oXML_GENERATOR.getXSDSchema(&RS_Employee);
&XSD_File = GetFile("Rowset.xsd", "W", "UTF8");
&XSD_File.WriteLine(&my_schema);

/* Get filename and path for creating xml file */
&XSD_Filename_path = &XSD_File.Name;
&XSD_File.Close();

rem create sample xml file;
&my_xml = &oXML_GENERATOR.getXMLData(&RS_Employee, "");
&XML_File = GetFile("Rowset.xml", "W", "UTF8");
&XML_File.WriteLine(&my_xml);

/* save file name and path for publishing */
&XML_Filename_path = &XML_File.Name;
&XML_File.Close();
```

XML File Generation Considerations

Concurrency: Generating multiple versions of the same report simultaneously

PeopleSoft by its very nature may have multiple, different users attempting to create the same report at the same time. Ensure that your code accounts for this probability by making the file names unique per each instance of the report. One idea to accomplish this is to append the User or Employee ID to the file creating a unique file name.

Another method is to append the current date/time stamp to the filename to force uniqueness.

Schema Files

For PeopleTools version 8.48 & 8.49, BI Publisher requires XML Schema files for the correct mapping of tags to a PDF Template. You could include these for RTF (MSWord) templates, but they were not necessary.

The easiest way to generate correct schemas for the generated XML Files is to use an XML File editing utility such as XML Spy or XML Fox. XML Fox is a freeware utility that works very nicely. These utilities can generate schemas from your final XML Files.
As demonstrated earlier, the Rowset method has a utility to generate the schema file at the time the XML File is created.

Starting with PeopleTools 8.50, BI Publisher does not require a schema file for PDF Template mapping. Only a sample data file is required.

Sample Files for defining BI Publisher Data Sources.

You must provide a Sample XML Data file when creating a new BI Publisher report. A good idea is to execute a version of the XML File generation code using a sample or representative employee. Another good idea is to, for the sample run, modify your code to put the XML Tag names as values in the XML File. This will aid in mapping and formatting: do you have the right field in the right place on your report?

House Keeping

Depending on data policies of the implementations site, it may be a good idea to delete the BI Publisher Source files from the files directory after the report documents have been created. The easiest way to delete the file is to re-open the file after the BI Publisher code is complete, then use the “delete()” method instead of the “close()” method. This will remove the file from the servers.

Appendix A

Data set class for data source creation. This function populates and returns a complex RowSet object.

```
/*********************************************************/
/** GronkWare, Inc                                     **/
/** Randy Groncki   20168-04-16                          **/
/** BI Publisher                                       **/
/** XML File Generation Examples                       **/
/** Contact: Randy.Groncki@GronkWare.com               **/
/*********************************************************/

class LoadTestData
   method LoadTestData();
   method LoadTestDataSet() Returns Rowset;

private

   method LoadEmployeeImage(&Emplid As string) Returns string;

end-class;

method LoadTestData

end-method;

method LoadTestDataSet
   /+ Returns Rowset +/

   Local Rowset &RS_Training, &RS_Employees;
   Local Record &JOB_REC, &LOCATION_TBL_REC, &COMPANY_TBL_REC, &JOBCODE_TBL_REC;
   Local integer &i;

   /* create records */
   &JOB_REC = CreateRecord(Record.JOB);
   &LOCATION_TBL_REC = CreateRecord(Record.LOCATION_TBL);
   &COMPANY_TBL_REC = CreateRecord(Record.COMPANY_TBL);
   &JOBCODE_TBL_REC = CreateRecord(Record.JOBCODE_TBL);

   /* create rowsets */
   &RS_Training = CreateRowset(Record.X_TRAINING_VW); /* child rowset */
   &RS_Employees = CreateRowset(Record.X_EE_RPT_LTR_VW, &RS_Training); /* parent
rowset */

   /* Fill Parent */
   &RS_Employees.Fill("where emplid like 'KU00%' and exists (select 'x' from
ps_training t where t.emplid = fill.emplid)");

   /* Loop through parent rowset for processing on each row */
   For &i = 1 To &RS_Employees.ActiveRowCount

      /* Fill child rowset */
      &RS_Training = &RS_Employees(&i).GetRowset(Scroll.X_TRAINING_VW);
      &RS_Training.Fill("where emplid = :1",
&RS_Employees(&i).X_EE_RPT_LTR_VW.EMPLID.Value);

      /* Get job row for linking other data */
      &JOB_REC.EMPLID.Value = &RS_Employees(&i).X_EE_RPT_LTR_VW.EMPLID.Value;
      &JOB_REC.EMPL_RCD.Value = 0;
      /* get the current effdt & effseq for the EEs job row */
      SQLExec("select %dateout(j.effdt), j.effseq from ps_job j where j.emplid = :1
and j.empl_rcd = :2 and j.effdt = (select max(j2.effdt) from ps_job j2 where
j2.emplid = j.emplid and j2.empl_rcd = j.empl_rcd and j2.effdt <= %datein(:3)) and
j.effseq = (select max(j3.effseq) from ps_job j3 where j3.emplid = j.emplid and
j3.empl_rcd = j.empl_rcd and j3.effdt = j.effdt)", &JOB_REC.EMPLID.Value,
&JOB_REC.EMPL_RCD.Value, %Date, &JOB_REC.EFFDT.Value, &JOB_REC.EFFSEQ.Value);
      &JOB_REC.SelectByKey();
```

Data set function for data source creation (Continued)

```
      /* retrieve specific location data for description in report */
      &LOCATION_TBL_REC.SETID.Value = &JOB_REC.SETID_LOCATION.Value;
      &LOCATION_TBL_REC.LOCATION.Value = &JOB_REC.LOCATION.Value;
      &LOCATION_TBL_REC.SelectByKeyEffDt(%Date);
      &RS_Employees(&i).X_EE_RPT_LTR_VW.LOCATION_DESCR.Value =
&LOCATION_TBL_REC.DESCR.Value;

      /* retrieve specific company data for description in report */
      &COMPANY_TBL_REC.COMPANY.Value = &JOB_REC.COMPANY.Value;
      &COMPANY_TBL_REC.SelectByKeyEffDt(%Date);
      &RS_Employees(&i).X_EE_RPT_LTR_VW.COMPANY_DESCR.Value =
&COMPANY_TBL_REC.DESCR.Value;

      /* retrieve specific jobcode data for description in report */
      &JOBCODE_TBL_REC.SETID.Value = &JOB_REC.SETID_JOBCODE.Value;
      &JOBCODE_TBL_REC.JOBCODE.Value = &JOB_REC.JOBCODE.Value;
      &JOBCODE_TBL_REC.SelectByKeyEffDt(%Date);
      &RS_Employees(&i).X_EE_RPT_LTR_VW.JOB_DESCR.Value = &JOBCODE_TBL_REC.DESCR.Value;

      /* get employee image */
      &RS_Employees(&i).X_EE_RPT_LTR_VW.X_EMPLOYEE_PHOTO.Value =
%This.LoadEmployeeImage(&JOB_REC.EMPLID.Value);

   End-For;

   Return &RS_Employees;
end-method;

method LoadEmployeeImage
   /+ &Emplid as String +/
   /+ Returns String +/

   Local File &Image_File;
   Local string &Base64String, &NewFileName, &FQ_Filename_path;
   Local integer &retcode;

   &NewFileName = %UserId | %Datetime | ".jpg";
   &Image_File = GetFile(&NewFileName, "W");
   &FQ_Filename_path = &Image_File.Name;
   &Image_File.Close();

   &retcode = GetAttachment("record://X_EPHOTO_VW", &Emplid, &FQ_Filename_path);

   If &retcode < 2 Then
      &Image_File = GetFile(&FQ_Filename_path, "R", %FilePath_Absolute);
      &Base64String = &Image_File.GetBase64StringFromBinary();
      &Image_File.Close();
   End-If;

   /* delete file */
   &Image_File = GetFile(&FQ_Filename_path, "R", %FilePath_Absolute);
   &Image_File.Delete();

   Return &Base64String;

end-method;
```

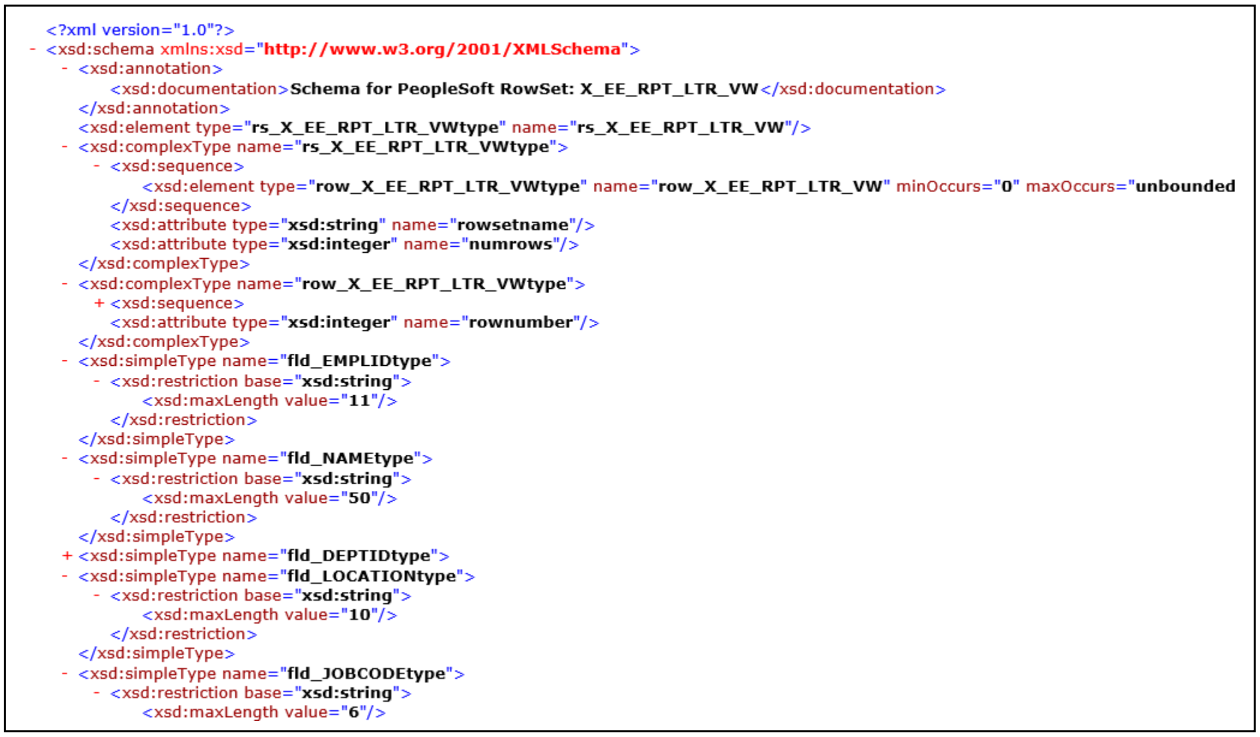
Appendix B

Rowset XML File Example

```
<?xml version="1.0"?>
<rs_X_EE_RPT_LTR_VW xsi:noNamespaceSchemaLocation=""
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" rowsetname="X_EE_RPT_LTR_VW"
numrows="5">
        <row_X_EE_RPT_LTR_VW rownumber="1">
                <fld_EMPLID>KU0001</fld_EMPLID>
                <fld_NAME>Douglas Lewis</fld_NAME>
                <fld_DEPTID>ADMIN</fld_DEPTID>
                <fld_LOCATION>KUNY00</fld_LOCATION>
                <fld_JOBCODE>700005</fld_JOBCODE>
                <fld_COMPRATE>21666.666667</fld_COMPRATE>
                <fld_CHANGE_PCT>0</fld_CHANGE_PCT>
                <fld_LOCATION_DESCR>Corporation Headquarters</fld_LOCATION_DESCR>
                <fld_JOB_DESCR>President & CEO</fld_JOB_DESCR>
                <fld_COMPANY_DESCR>Global Business Institute</fld_COMPANY_DESCR>
                <rs_X_TRAINING_VW rowsetname="X_TRAINING_VW" numrows="1">
                        <row_X_TRAINING_VW rownumber="1">
                                <fld_EMPLID>KU0001</fld_EMPLID>
                                <fld_COURSE_START_DT>2005-01-13</fld_COURSE_START_DT>
                                <fld_COURSE>M2005</fld_COURSE>
                                <fld_SESSION_NBR>0001</fld_SESSION_NBR>
                                <fld_COURSE_TITLE>course with 2000 hours</fld_COURSE_TITLE>
                                <fld_COURSE_END_DT/>
                                <fld_ATTENDANCE>E</fld_ATTENDANCE>
                                <fld_TRAINING_REASON/>
                                <fld_COURSE_GRADE/>
                                <fld_PREREQ_MET>N</fld_PREREQ_MET>
                                <fld_DEPTID>ADMIN</fld_DEPTID>
                                <fld_BUSINESS_UNIT>GBIBU</fld_BUSINESS_UNIT>
                        </row_X_TRAINING_VW>
                </rs_X_TRAINING_VW>
        </row_X_EE_RPT_LTR_VW>
</rs_X_EE_RPT_LTR_VW>
```

Appendix C

Rowset Schema (XSD) File Example



```
  <?xml version="1.0"?>
- <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  - <xsd:annotation>
      <xsd:documentation>Schema for PeopleSoft RowSet: X_EE_RPT_LTR_VW</xsd:documentation>
    </xsd:annotation>
    <xsd:element type="rs_X_EE_RPT_LTR_VWtype" name="rs_X_EE_RPT_LTR_VW"/>
  - <xsd:complexType name="rs_X_EE_RPT_LTR_VWtype">
    - <xsd:sequence>
        <xsd:element type="row_X_EE_RPT_LTR_VWtype" name="row_X_EE_RPT_LTR_VW" minOccurs="0" maxOccurs="unbounded
      </xsd:sequence>
      <xsd:attribute type="xsd:string" name="rowsetname"/>
      <xsd:attribute type="xsd:integer" name="numrows"/>
    </xsd:complexType>
  - <xsd:complexType name="row_X_EE_RPT_LTR_VWtype">
    + <xsd:sequence>
      <xsd:attribute type="xsd:integer" name="rownumber"/>
    </xsd:complexType>
  - <xsd:simpleType name="fld_EMPLIDtype">
    - <xsd:restriction base="xsd:string">
        <xsd:maxLength value="11"/>
      </xsd:restriction>
    </xsd:simpleType>
  - <xsd:simpleType name="fld_NAMEtype">
    - <xsd:restriction base="xsd:string">
        <xsd:maxLength value="50"/>
      </xsd:restriction>
    </xsd:simpleType>
  + <xsd:simpleType name="fld_DEPTIDtype">
  - <xsd:simpleType name="fld_LOCATIONtype">
    - <xsd:restriction base="xsd:string">
        <xsd:maxLength value="10"/>
      </xsd:restriction>
    </xsd:simpleType>
  - <xsd:simpleType name="fld_JOBCODEtype">
    - <xsd:restriction base="xsd:string">
        <xsd:maxLength value="6"/>
```