ORACLE
PEOPLESOFT ENTERPRISE

Enterprise PeopleTools 8.51 PeopleBook: XML Publisher for PeopleSoft Enterprise

October 2011

ORACLE

Enterprise PeopleTools 8.51 PeopleBook: XML Publisher for PeopleSoft Enterprise
SKU pt8.51txml-b1011

Trademark Notice

License Restrictions Warranty/Consequential Damages Disclaimer

Warranty Disclaimer

Restricted Rights Notice

Hazardous Applications Notice

Third Party Content, Products, and Services Disclaimer

Contents

Chapter 6

Appendix A

Appendix B

XML Publisher for PeopleSoft Enterprise Preface

This preface discusses XML Publisher for PeopleSoft Enterprise.

XML Publisher for PeopleSoft Enterprise

XML Publisher for PeopleSoft Enterprise is a template-based reporting solution that separates the data extraction process from the report layout and allows the reuse of extracted application data into multiple report layouts. XML Publisher uses select features from Oracle Business Intelligence Publisher (BI Publisher) that have been integrated into PeopleTools.

PeopleBooks and the PeopleSoft Online Library

A companion PeopleBook called *PeopleBooks and the PeopleSoft Online Library* contains general information, including:

- Understanding the PeopleSoft online library and related documentation.
- How to send PeopleSoft documentation comments and suggestions to Oracle.
- How to access hosted PeopleBooks, downloadable HTML PeopleBooks, and downloadable PDF PeopleBooks as well as documentation updates.
- Understanding PeopleBook structure.
- Typographical conventions and visual cues used in PeopleBooks.
- ISO country codes and currency codes.
- PeopleBooks that are common across multiple applications.
- Common elements used in PeopleBooks.
- Navigating the PeopleBooks interface and searching the PeopleSoft online library.
- Displaying and printing screen shots and graphics in PeopleBooks.
- How to manage the locally installed PeopleSoft online library, including web site folders.
- Understanding documentation integration and how to integrate customized documentation into the library.
- Application abbreviations found in application fields.

You can find *PeopleBooks and the PeopleSoft Online Library* in the online PeopleBooks Library for your PeopleTools release.

Chapter 1

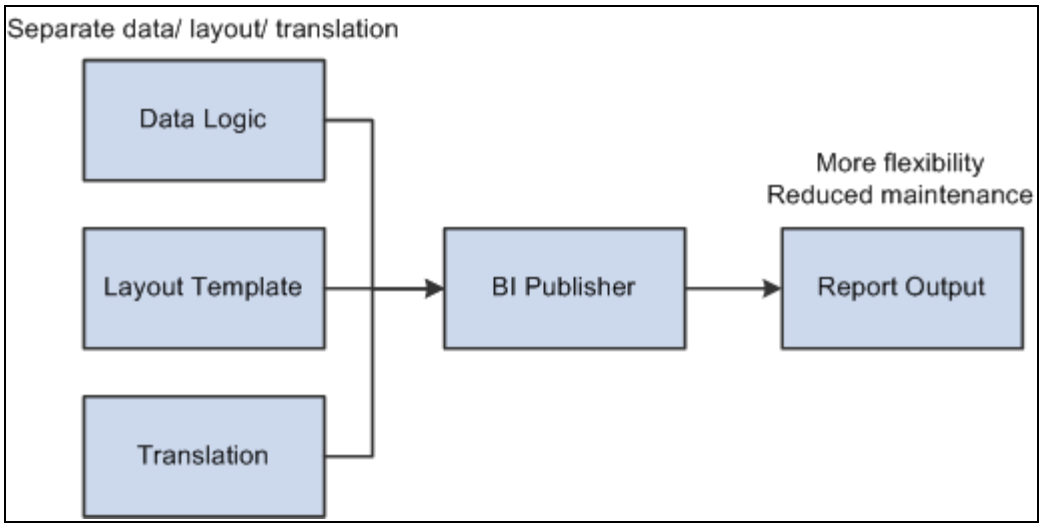
Getting Started with XML Publisher

This chapter provides an overview of XML Publisher and discusses XML Publisher phases.

XML Publisher Overview

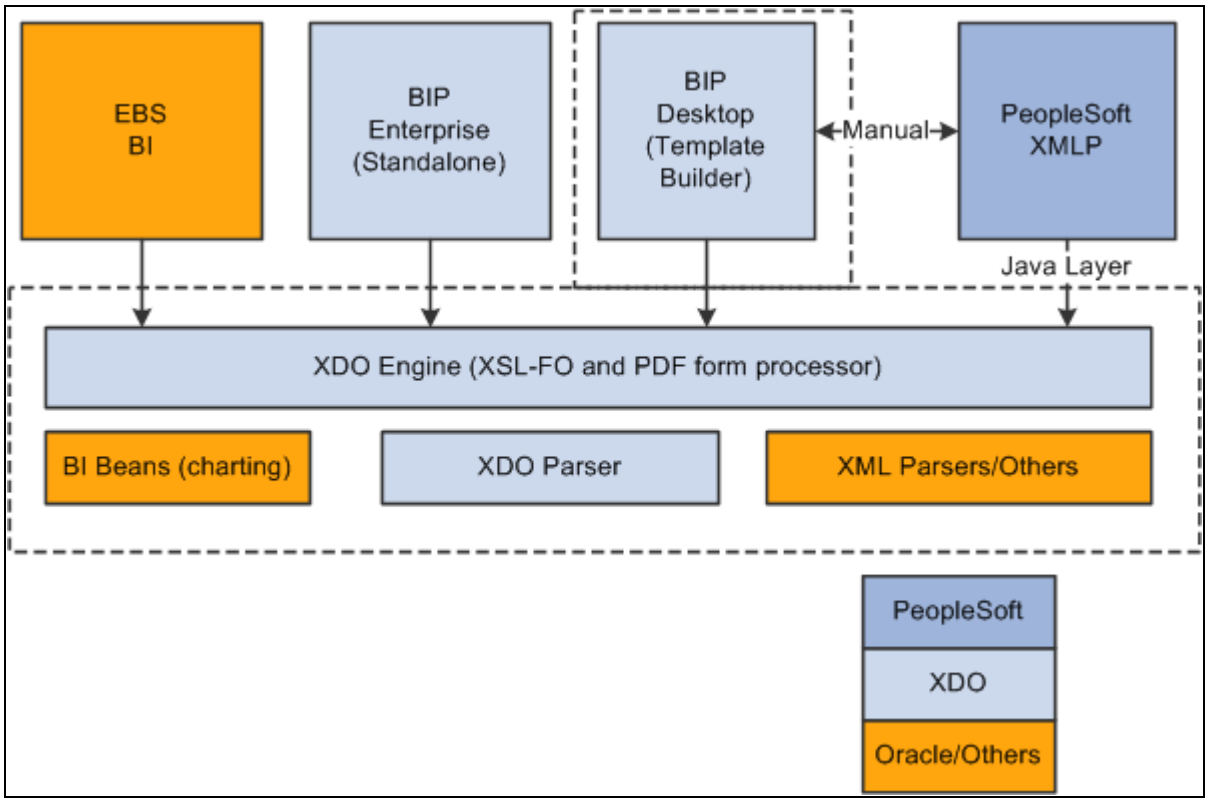
Oracle Business Intelligence Publisher (BI Publisher, formerly XML Publisher) is an enterprise reporting solution that streamlines report and form generation. A primary feature of Oracle's BI Publisher product is the separation of the data extraction process from the report layout. BI Publisher enables you to design and create report layout templates with the more common desktop applications of Microsoft Word and Adobe Acrobat, and renders XML data based on those templates. With a single template, it can generate reports in many formats (PDF, RTF, Excel, HTML, and so on) in many languages. This approach to reporting can dramatically reduce report maintenance, enabling power business users to adjust report templates without involvement of IT resources.

The following diagram illustrates the concept of BI Publisher.



BI Publisher concept

Select features of Oracle's BI Publisher product have been integrated into and enhanced for use with PeopleTools. Within PeopleSoft applications, this is referred to as XML Publisher (XMLP). This diagram illustrates PeopleSoft applications integration with BI Publisher:



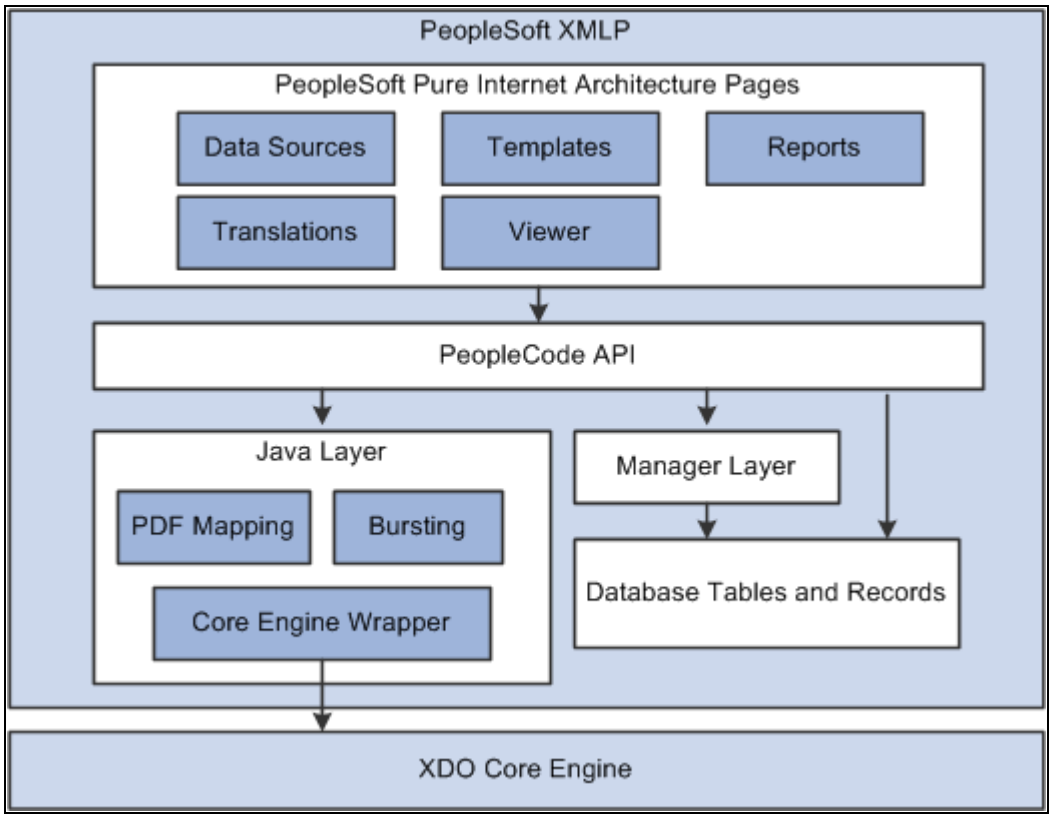
BI Publisher integration with PeopleSoft applications

PeopleSoft XML Publisher has a direct Java integration to the XDO Engine and XDO Parser. The BI Publisher Desktop requires installation and can be downloaded from a PeopleSoft Pure Internet Architecture page.

Note. Not all BI Publisher features are available through the PeopleSoft implementation.

PeopleSoft Query as well as any PeopleTools based applications providing XML data are available to BI Publisher as a data source. XML Publisher for PeopleSoft Enterprise provides an environment for managing templates, data sources, reports, translations, and content components. It also offers an electronic bursting capability to produce reports according to a user-defined criteria and secure the reports using an application's security join table. A set of PeopleCode XML Publisher classes for runtime report generation is also provided.

This diagram illustrates the XMLP components within the PeopleSoft system.



PeopleSoft XMLP components

Oracle provides a set of PeopleSoft Pure Internet Architecture pages for defining data sources, report definitions, templates, and translations and for running and viewing reports. Oracle also provides a set of PeopleCode application program interfaces (APIs) that wrap the Oracle XDO engine APIs. These APIs are used by the BI Publisher PeopleSoft Pure Internet Architecture pages and are available for advanced report developers to use for custom applications and batch processes.

XML Publisher Phases

XML Publisher implementation includes the following phases:

- Set up XML Publisher.
- Create and register data sources.
- Create and upload report templates.
- Define XML Publisher reports.
- Run, locate, and view XML Publisher reports.

Set Up XML Publisher

To prepare your system for using XML Publisher, perform the following steps:

Step	Reference
1. Define XML Publisher settings.	See Chapter 2, "Setting Up XML Publisher," page 7.
2. Set up XML Publisher permission list security.	See Chapter 2, "Setting Up XML Publisher," Assigning XMLP Permissions to Users, page 15.
3. Set up Report Manager.	See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Report Manager."
4. Define report categories, including Report Definition Editor security.	See Chapter 2, "Setting Up XML Publisher," Setting Up Report Categories, page 11.
5. Download design plug-ins to facilitate the offline template design activities.	See Chapter 2, "Setting Up XML Publisher," Working with Template Design Helpers, page 15.

Create and Register Data Sources

To create and register data sources, perform the following steps:

Step	Reference
1. Identify or create the source of your report data. Data sources can be PS Query, Connected Queries, or XML files. **Note.** Rowset and XML Doc object data sources have been deprecated. Rowset and XML Doc object data sources created in previous releases will continue to be supported. To generate XML files from XML Doc or rowsets, refer to PeopleCode API documentation. See *PeopleTools 8.51 PeopleBook: PeopleCode API Reference*, "XML Publisher Classes."	See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Query*, "Creating and Running Simple Queries" and Chapter 3, "Creating and Registering Data Sources," Creating Data Sources, page 17.
2. Register schema and sample data files for XML Publisher data sources. For PS Query, you can automatically generate schema file and sample data. **Note.** Schema is no longer used for bursting starting in PeopleTools 8.50. It is still available for backwards compatibility.	See Chapter 3, "Creating and Registering Data Sources," Registering Data Sources, page 21.

Create and Upload Report Templates

To create and upload templates, perform the following steps:

Step	Reference
1. Create and upload schema and sample data.	See Chapter 3, "Creating and Registering Data Sources," Creating Data Sources, page 17.
2. Download sample data from the appropriate data source to facilitate template design.	See Chapter 5, "Defining Report Definitions," Defining Reports, page 40.
3. Use either Microsoft Word or Adobe Acrobat to develop and maintain custom report formats.	See Chapter 4, "Creating Report Templates," Understanding Report Template Types, page 25.
4. (Optional) Create and maintain reusable sub-template definitions.	See Chapter 5, "Defining Report Definitions," Maintaining Sub-Templates, page 60.
5. (Optional) Register translation XLIFF files for report templates and Content Library sub-templates.	See Chapter 5, "Defining Report Definitions," Maintaining Template Translations, page 63.

Define XML Publisher Reports

To create and maintain report definitions, perform the following step:

Step	Reference
1. Define reports by associating data sources with layout template files.	See Chapter 5, "Defining Report Definitions," Defining Reports, page 40.

Run, Locate, and View XML Publisher Reports

You can run XML Publisher reports online or in batch through the Process Scheduler. For query-based reports, pages are available for running the reports both online and in batch. To run XML Publisher reports, perform the following steps:

Step	Reference
1. Schedule Query-based XML Publisher reports.	See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Scheduling Reports in Query Report Scheduler, page 71.
2. Schedule other XML Publisher reports. **Note.** You will need to create an application engine program using XMLP PeopleCode APIs.	See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Running Reports Using PeopleCode, page 75.
3. View Query-based XML Publisher reports online in real time.	See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Running Reports in Query Report Viewer, page 70.
4. View other XML Publisher reports online in real time.	See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Running Reports Using PeopleCode, page 75.

Step	*Reference*
5. Locate XML Publisher reports using enhanced search criteria.	See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Searching the XML Publisher Report Repository, page 79 and *PeopleTools 8.51 PeopleBook: PeopleCode API Reference*, "XML Publisher Classes," Search Operator Values.
6. View XML Publisher reports in the Report Manager.	See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Report Manager," Viewing Reports.

Chapter 2

Setting Up XML Publisher

This chapter provides an overview of XML Publisher Set Up and discusses how to:

- Define system properties and fonts.
- Set up XML Publisher.
- Assign XMLP permissions to users.

Understanding XML Publisher Set Up

Before using XML Publisher, there are some set up tasks necessary to set up the environment and facilitate template design. This table lists the categories for the set up tasks:

Set Up	***Description***
Properties	Properties for XML Publisher can be set at four levels. System and global properties should be configured as part of the initial set up for XML Publisher.
Security	Security is defined for creating and editing report definitions.
Template Design	Template Builder is an extension to Microsoft Word that simplifies the development of RTF templates. Template Builder can be downloaded from PeopleSoft Pure Internet Architecture page or Oracle Technical Network (OTN).

XML Publisher Properties

There are two types of properties used in XML Publisher:

System Properties System level properties are set in the xdo.cfg file. System level properties include:

- xslt-parser
- xslt-scalable
- system-cachepage-size
- system-temp-dir
- fonts

Non-system Properties Non-system or functional properties are set in PeopleSoft Pure Internet Architecture.

Property Definition Levels

There are four levels where properties are defined, this table lists the levels:

Level	*Description*	*Location*
System properties and fonts	System properties and fonts.	xdo.cfg file
Global properties	Global properties are shared by all reports and override the xdo engine default property values.	Reporting Tools, XML Publisher, Setup, Global Properties
Report properties	Properties are defined in the report definition and override global properties for a single report.	Reporting Tools, XML Publisher, Report Definition, Properties See Chapter 5, "Defining Report Definitions," Setting Report Properties, page 52.
Runtime properties	Override report properties.	Runtime properties are set at runtime through ReportDefn.SetRuntimeProperties PeopleCode API.

Defining System Properties and Fonts

This section provides an overview of the xdo.cfg file and discusses how to:

- Set system temp directory.
- Set Application Server or Process Scheduler domain-specific xdo.cfg file.
- Copy fonts.

Understanding xdo.cfg File

XML Publisher system properties settings are defined in the xdo.cfg file. The default xdo.cfg file is located in the $PSHOME/appserver directory, which is shared by all application server and process scheduler domains by default.

Note. In PeopleTools 8.4x, the xdo.cfg file is used to define all types of properties (system and non-system). In PeopleTools 8.50 the file should be used for system properties and fonts only. The result is unpredictable if the same property is defined in xdo.cfg and other levels.

This is an example of the xdo.cfg file:

```
<config version="1.0.0" xmlns="http://xmlns.oracle.com/oxp/config/">

    <properties>
        <!-- System level properties -->
        <property name="xslt-xdoparser">true</property>
        <property name="xslt-scalable">true</property>
        <property name="system-cachepage-size">50</property>
        <property name="system-temp-dir"></property>
    </properties>


    <!--<fonts>-->
        <!--<font family="3 of 9 Barcode" style="normal" weight="normal">-->
            <!--<truetype path="C:\WINNT\Fonts\3of9.ttf" />-->
        <!--</font>-->
    <!--</fonts>-->

</config>
```

See *Oracle XML Publisher Core Components Guide, Setting Runtime Properties.*

Setting System Temp Directory

By default, the `system-temp-dir` property is not set. This property must be set to point to a temp folder on the server. Note that temporary files created in that directory could grow very large in size depending on the size of your reports, so you need to choose your `system-temp-dir` for optimum system performance.

Setting Application Server or Process Scheduler Domain-Specific xdo.cfg File

You can also specify an application server or process scheduler domain-specific xdo.cfg file. To do this, you need to change the application server or process scheduler configuration file to update the`JavaVM Options -Dxdo.ConfigFile` setting. For example, to specify a separate xdo.cfg file for the application server domain P8488041, change the [PS_HOME]/appserv/P8488041/psappsrv.cfg file as indicated in the following code samples and put the new xdo.cfg into the [PS_HOME]/appserv/P8488041directory.

Original line in psappsrv.cfg:

```
JavaVM Options=-Xrs -Dxdo.ConfigFile=%PS_HOME%/appserv/xdo.cfg
```

New line in psappsrv.cfg:

```
JavaVM Options=-Xrs -Dxdo.ConfigFile==%PS_HOME%/appserv/P8488041/xdo.cfg
```

In the preceding code sample, P8488041 is the Application Server domain name.

If you change the content of xdo.cfg, you don't need to restart the application server or the process scheduler domain that uses it. It refreshes automatically the next time you run it. But if you change the application server or process scheduler configuration file, you need to restart the affected domain.

Copying Fonts

XMLP Core engine uses default fonts when the corresponding fonts are not available on the system for a particular report template. The engine looks for these fonts under PS_HOME\JRE\lib\fonts folder.

The default fonts are included in BI Publisher Desktop and need to be copied to the JRE directory. The fonts are located under the Desktop Publisher installation folder, the default is:

```
C:\Program Files\Oracle\BI Publisher\BI Publisher Desktop\Template Builder for⇒
 Word\fonts
```

The following files need to be copied to the PS_HOME/JRE/LIB folder:

- ALBANWTJ.ttf
- ALBANWTK.ttf
- ALBANWTS.ttf
- ALBANWTT.ttf
- ALBANYWT.ttf

Setting Up XML Publisher

This section discusses how to:

- Set up report categories.
- Define global properties.
- Work with template design helpers.

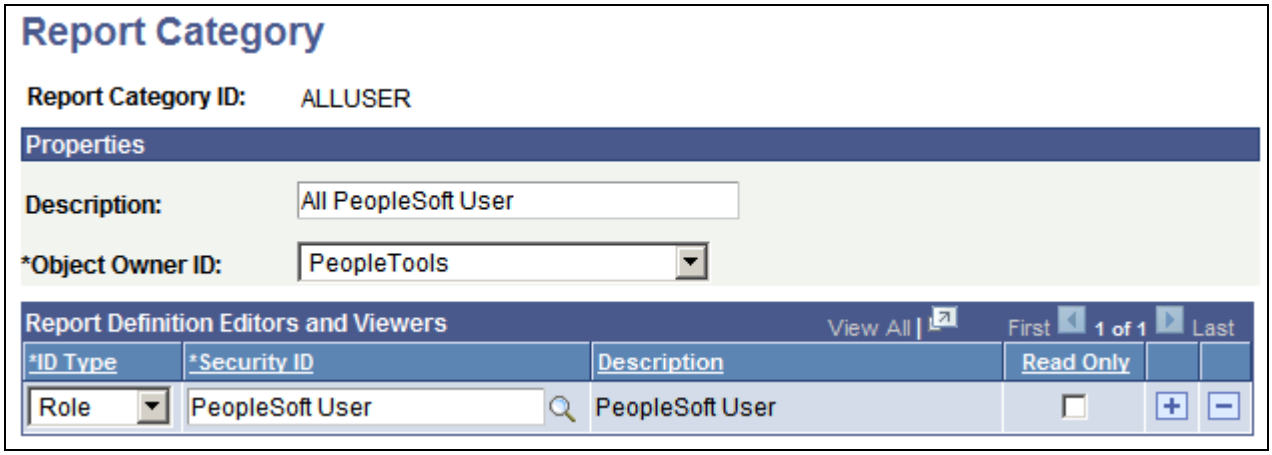
Pages Used to Set Up XML Publisher

Page Name	*Definition Name*	*Navigation*	*Usage*
Report Category	PSXPSETUPRPTCAT	Reporting Tools, XML Publisher, Setup, Report Category	Set up report categories.
Global Properties	PSXPGLBPROP	Reporting Tools, XML Publisher, Setup, Global Properties	Define global properties.

Page Name	Definition Name	Navigation	Usage
Design Helper	PSXPSETUPDWNLD	Reporting Tools, XML Publisher, Setup, Design Helper	Download plug-ins to facilitate offline template design.

Setting Up Report Categories

Access the Report Category page (Select Reporting Tools, XML Publisher, Setup, Report Category.)



Report Category page

Report Category is a required attribute on all report definitions and Content Library sub-templates. By assigning a report category, you are actually applying row level security to the data on those components.

Report Category ID Enter a report category ID to define a grouping that enables users to control who can access and edit report definitions and Content Library sub-templates.

Description (Optional) Enter descriptive text that provides more detail about the report category.

Object Owner ID Indicate which product, feature, or application owns this report category.

ID Type Select an ID type of either *Role* or *User ID* to grant authorization to.

Security ID Select the authorized editor's security ID based on the ID type.

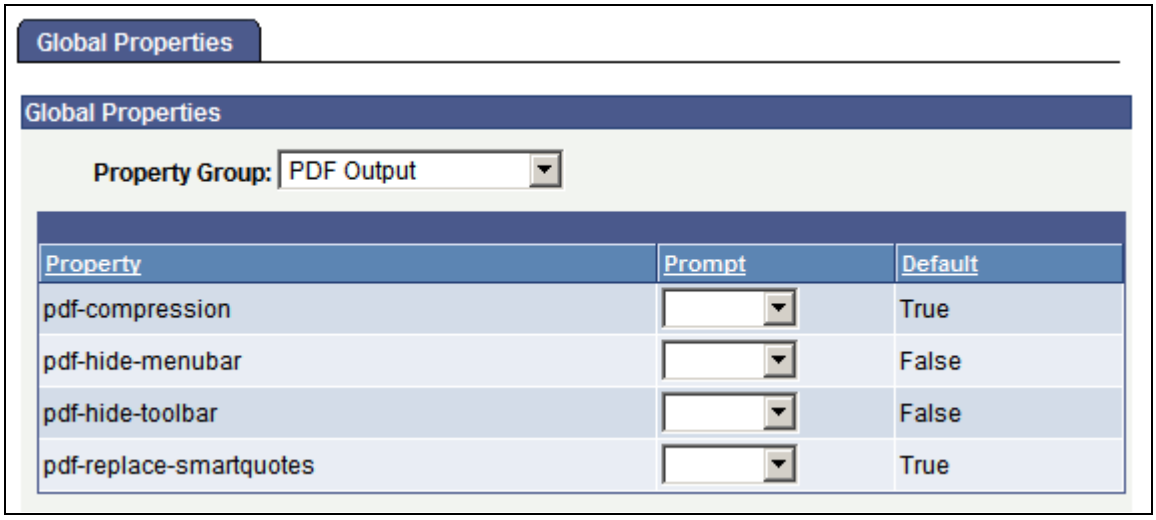
Description A read-only field that indicates the related display value for the security ID.

Read Only (Optional) Select to indicate that the designated ID is only able to view the report definitions under this category and not update them.

Note. The PeopleCode XML Publisher classes also respect report category settings and read-only access rights.

Defining Global Properties

Access the Global Properties page (Select Reporting Tools, XML Publisher, Setup, Global Properties.)



Global Properties page

Property Group Select the property group.

Property All properties available for the property group selected appear.

Prompt Select the value for the property.

Default Displays the default value for the property.

Property Groups

The following property groups are available:

- PDF Output
- FO Processing
- HTML Output
- PDF Security
- PDF Template
- PeopleTools Settings
- RTF Output
- RTF Template

See *Oracle Business Intelligence Publisher User's Guide, Setting Runtime Properties* for details on the properties.

Note. The Oracle Business Intelligence Publisher User's Guide (XDOUserGuide.pdf) is downloaded with Template Builder for Word and available in the directory selected for download. For example, C:\Program Files\Oracle\BI Publisher\BI Publisher Desktop\Template Builder for Word\doc\XDOUserGuide.pdf.

PeopleTools Settings

The properties in PeopleTools Settings control report attributes that are specific to PeopleSoft implementation of BI Publisher.

psxp_pdf_optimized

This property controls whether or not the core engine uses the "optimized" PDF Form Processing feature. This increases the efficiency and performance of PDF-template based reports, while disabling certain features. Valid values are:

- True.

  Enables core engine optimization for PDF-based reports. The optimized engine will provide better performance, while disabling certain PDF-template specific features such as repeated fields and editable fields.

  Note. Full path mapping can be used.

- False

  Uses the unoptimized engine (same as BIP server), which will enable repeated fields and editable fields in a PDF template.

  Note. Full path mapping is not supported.

  ---

psxp_debug

This property controls whether or not to leave temporary files on the application server or the process scheduler server for debugging purpose. It is recommended to set this property at the report definition level to debug a specific report. Valid values are:

- True

  Temporary files will not be deleted from application server or process scheduler server for debugging purpose.

- False

  Temporary files are deleted from application server or process scheduler server.

Note. If this property is set to true, remember to change it back to False when debugging is completed.

---

psxp_usedefaultoutdestination

This property is used to indicate that default processing directory is exposed to the OutDestination property even if this value has not been previously set. The default value of is *False.*

- True

  A basic tools directory is exposed to the user, without showing an additional RptInst directory. This is the behavior in pre-8.50 XML Publisher.

  If this property is set to True and the user does not set value for OutDestination at runtime, then Tools will create an output file <Domain>\files\XMLP\123456789\RptInst\MyReport.HTM where 123456789 is for a directory name being generated with a random name. In this example the OutDestination property will return the value: <Domain>\files\XMLP\123456789.

  Some directories will not be cleaned up after processing is done and the report is delivered into Report Manager. Any empty directories that are left after the XMLP reports are delivered to the Report Manager will be cleaned up when the regularly scheduled Application Engine process PRCSYSPURGE runs. You can also run the Application Engine program PSXP_DIRCLN to clean up the directories.

- False

  This is the default value. Querying the OutDestination property without previously setting it at runtime, will cause it to return blank. After the reports have been delivered to the report repository, the temporary files and directories used for processing will be deleted.

Editing PDF Output

In previous releases the ability to edit PDF output was defined on the Report Definition Output page using the *PDF report output may be edited* check box. Starting with PeopleTools 8.50, the set properties *pdf-no-changing-the-docment* and *pdf-security* are set either on the Global Properties page for all reports or on the Report Properties page for a specific report.

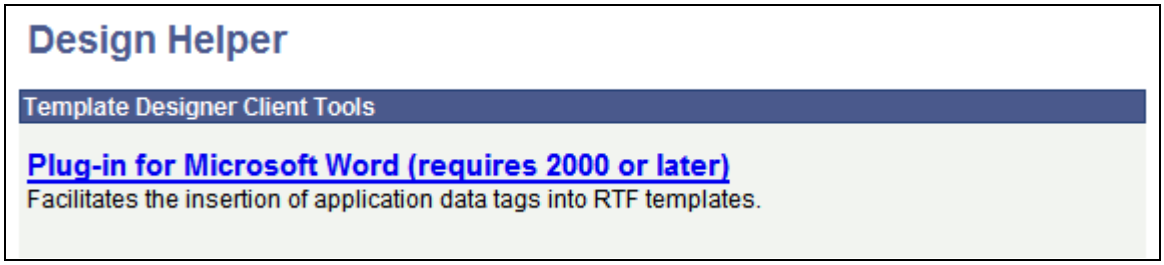
To allow editing of PDF reports, the properties for the property group PDF Security should be set as follows:

Property	***Value***
pdf-security	True
pdf-encryption-level	0
pdf-no-changing-the-docment	False

Note. It is recommended to set these properties at the report definition level.

Working with Template Design Helpers

Access the Design Helper page (Select Reporting Tools, XML Publisher, Setup, Design Helper.)



Design Helper page

During template creation, a design helper facilitates the insertion of application data tag placeholders into your templates. A schema or sample data file is required for using a design helper. If you use a sample data file, you can preview your template offline during the design process.

Two template design helpers are available: one for designing RTF reports (MS Word Template Builder) and one for the PDF template that gets embedded as a JavaScript inside the PDF template itself when you upload the template to PeopleTools.

This page enables users to download a plug-in for Microsoft Word to facilitate offline RTF template design. Select the link to download the tool.

The file xmlp_desktop.zip is downloaded. Unzip the file, which will contain a folder XMLP_DESKTOP. In this folder, select BIPublisherDesktop.exe.

Included in the BI Publisher desktop plug-in setup, multiple directories are created that contain documentation, samples, tutorial, and demos. The program is also added to the Start menu.

See Chapter 4, "Creating Report Templates," Using PDF Templates, page 33.

Assigning XMLP Permissions to Users

XML Publisher menu access is permission-list driven and depends on permission list and role assignment. PeopleTools delivers permission-list security and roles for XML Publisher report developers and XML Publisher power users.

Permission list PTPT2600 is intended for report developers. Users assigned a role with this permission list have access to all XML Publisher components, including setup capability on the advanced feature Report Definition Bursting page.

Permission list PTPT2500 is intended for power users and provides access to Query data sources for ad hoc reporting through Query Report Viewer and Query Report Scheduler. This permission list also provides access to report definitions and to the Content Library, though access to the report definition bursting information is view-only.

Users assigned to other permission lists and roles, such as permission list PTPT1000, can have access only to the XML Publisher Report Repository.

This table illustrates delivered permission-list security for XML Publisher:

Component	*XMLP Report Developer (PTPT2600)*	*XMLP Power User (PTPT2500)*	*PeopleSoft User (PTPT1000)*
Report Category	Yes	No	No
Design Helper	Yes	Yes	No
Global Properties	Yes	No	No
Data Source	Yes	No	No
Report Definition	Yes	Yes **Note.** Display-only access for bursting. **Note.** Report properties page is not available.	No
Content Library	Yes	Yes	No
Template Translations	Yes	No	No
Query Report Viewer	Yes	Yes	No
Query Report Scheduler	Yes	Yes	No
Report Repository	Yes	Yes	Yes

Chapter 3

Creating and Registering Data Sources

This chapter discusses how to:

- Create data sources.
- Register data sources.

Creating Data Sources

This section provides an overview of data generation and discusses how to create schema and sample data.

Understanding Data Generation

In XML Publisher, the data extraction is separate for the data presentation. Sample data can be used to design your RTF template and map your PDF templates. Data schema was used in previous releases for the bursting feature and is still available for backwards compatibility and bursting. If PeopleSoft queries are used for data extraction, the system will generate the schema; for all other data sources, you must create the data schema using tools outside of the PeopleSoft system.

XML Publisher can register PS/Query, Connected Query, and XML files as a data source, but you can generate XML data using any means including PS/Query, SQR, Application Engine, PeopleCode, File Layout, and so forth.

For RTF template-based reports, design your data source XML structure using groupings that resemble the groupings needed for the output report. This improves runtime performance by preventing unnecessary grouping by the formatting engine.

Creating Schema and Sample Data

Use sample data source information for developing your RTF report templates, defining bursting, and mapping your PDF templates. Schema files are no longer necessary as of PeopleTools 8.50, but can still be defined for backwards compatibility with previous PeopleTools releases.

Note. New PDF mapping are not supported if a schema file is used.

Storing the sample data file in PeopleTools provides a means to:

- Insert form field tags in RTF templates
- Conduct PDF mapping.

- Choose the bursting field during design time.
- Preview the template.

Sample Data File

Requirements for the structure of XML sample data file include:

- Must consist of a root node with one repeating group.

  Textual elements in this repeating group are candidates for bursting.

- Elements should have textual content.

  Element should not be empty.

- All expected elements must be included.

  All text elements should contain default values. All defined elements can be used for mapping.

This is an example of a sample XML file used as a data source:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
   <taxroot>
      <Box_Year>2005</Box_Year>
      <Box_Form>W2</Box_Form>
      <EE_SSN>111-11-1111</EE_SSN>
      <ER_EIN>ER_111111111</ER_EIN>
      <Employee>
          <EE_FirstName>Incheol</EE_FirstName>
          <EE_LastName>Kang</EE_LastName>
          <EE_Address1>500 Oracle Parkway</EE_Address1>
          <EE_Address2>Redwood Shores</EE_Address2>
          <EE_Address3>CA 94065</EE_Address3>
      </Employee>
      <Employer>
          <ER_Name>Oracle USA</ER_Name>
          <ER_Address1>500 Oracle Parkway</ER_Address1>
          <ER_Address2>Redwood Shores</ER_Address2>
          <ER_Address3>CA 94065</ER_Address3>
      </Employer>
      <Tax>
          <Fed_Wages_COR>20000</Fed_Wages_COR>
          <Fed_Tax_COR>20000</Fed_Tax_COR>
         <RETIRE_EE_PRV>Y</RETIRE_EE_PRV>
      </Tax>
   </taxroot>
   </root>
```

Note. In this example, the elements *Box_Form,Box_Year,EE_SSN* and *ER_EIN* are available as burst candidates. All of the elements will be available for mapping.

The actual data file may contain repeated instances of the high level repeating group, as shown in this example:

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
   <taxroot>
      <Box_Year>2005</Box_Year>
      <Box_Form>W2</Box_Form>
      <EE_SSN>111-11-1111</EE_SSN>
      <ER_EIN>ER_111111111</ER_EIN>
      <Employee>
          <EE_FirstName>Incheol</EE_FirstName>
          <EE_LastName>Kang</EE_LastName>
          <EE_Address1>500 Oracle Parkway</EE_Address1>
          <EE_Address2>Redwood Shores</EE_Address2>
          <EE_Address3>CA 94065</EE_Address3>
      </Employee>
      <Employer>
          <ER_Name>Oracle USA</ER_Name>
          <ER_Address1>500 Oracle Parkway</ER_Address1>
          <ER_Address2>Redwood Shores</ER_Address2>
          <ER_Address3>CA 94065</ER_Address3>
      </Employer>
      <Tax>
          <Fed_Wages_COR>20000</Fed_Wages_COR>
          <Fed_Tax_COR>20000</Fed_Tax_COR>
         <RETIRE_EE_PRV>Y</RETIRE_EE_PRV>
      </Tax>
   </taxroot>
   <taxroot>
       <Box_Year>2005</Box_Year>
       <Box_Form>W2</Box_Form>
       <EE_SSN>2222-22-2222</EE_SSN>
       <ER_EIN>ER_222222222</ER_EIN>
       <Employee>
           <EE_FirstName>Chang</EE_FirstName>
           <EE_LastName>Yu</EE_LastName>
           <EE_Address1>500 Oracle Parkway</EE_Address1>
           <EE_Address2>Redwood Shores</EE_Address2>
           <EE_Address3>CA 94065</EE_Address3>
       </Employee>
       <Employer>
            <ER_Name>Oracle USA</ER_Name>
            <ER_Address1>500 Oracle Parkway</ER_Address1>
            <ER_Address2>Redwood Shores</ER_Address2>
            <ER_Address3>CA 94065</ER_Address3>
       </Employer>
       <Tax>
            <Fed_Wages_COR>10000</Fed_Wages_COR>
            <Fed_Tax_COR>10000</Fed_Tax_COR>
            <RETIRE_EE_PRV>Y</RETIRE_EE_PRV>
       </Tax>
     </taxroot>
   </root>
```

Schema File

This is the sample schema for the XML file shown previously:

```
<?xml version="1.0"?>
<xs:schema id="root" targetNamespace="http://tempuri.org/example_xml.xsd" xmlns:⇒
mstns="http://tempuri.org/example_xml.xsd" xmlns="http://tempuri.org/example_⇒
xml.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-⇒
microsoft-com:xml-msdata" attributeFormDefault="qualified" elementFormDefault=⇒
"qualified">
  <xs:element name="root" msdata:IsDataSet="true" msdata:EnforceConstraints=⇒
"False">
    <xs:complexType>
      <xs:choice maxOccurs="unbounded">
        <xs:element name="taxroot">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Box_Year" type="xs:string" minOccurs="0" />
              <xs:element name="Box_Form" type="xs:string" minOccurs="0" />
              <xs:element name="EE_SSN" type="xs:string" minOccurs="0" />
              <xs:element name="ER_EIN" type="xs:string" minOccurs="0" />
              <xs:element name="Employee" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="EE_FirstName" type="xs:string" minOccurs="0"⇒
 />
                    <xs:element name="EE_LastName" type="xs:string" minOccurs="0" ⇒
/>
                    <xs:element name="EE_Address1" type="xs:string" minOccurs="0" ⇒
/>
                    <xs:element name="EE_Address2" type="xs:string" minOccurs="0" ⇒
/>
                    <xs:element name="EE_Address3" type="xs:string" minOccurs="0" ⇒
/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="Employer" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="ER_Name" type="xs:string" minOccurs="0" />
                    <xs:element name="ER_Address1" type="xs:string" minOccurs="0" ⇒
/>
                    <xs:element name="ER_Address2" type="xs:string" minOccurs="0" ⇒
/>
                    <xs:element name="ER_Address3" type="xs:string" minOccurs="0" ⇒
/>
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
              <xs:element name="Tax" minOccurs="0" maxOccurs="unbounded">
                <xs:complexType>
                  <xs:sequence>
                    <xs:element name="Fed_Wages_COR" type="xs:string" minOccurs=⇒
"0" />
                    <xs:element name="Fed_Tax_COR" type="xs:string" minOccurs="0" ⇒
/>
                    <xs:element name="RETIRE_EE_PRV" type="xs:string" minOccurs=⇒
"0" />
                  </xs:sequence>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
          </xs:complexType>
```

```
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Note. Schema and sample data are generated for the PeopleSoft Query data source.

Registering Data Sources

This section provides an overview of data source registration and discusses how to register data sources.

Understanding Data Source Registration

A data source registers the schema and sample data design files. The extracted application fields from the data source files are placed into the template files to create the final report.

The data source can be PS Query, Connected Query, or XML files.

Note. PeopleSoft queries with in tree prompts are not allowed as a data source.

Benefits of data source registration include the ability to:

1. Reuse previously registered data sources with multiple report definitions.
2. Take advantage of built-in bursting features.

   See Chapter 5, "Defining Report Definitions," Setting Bursting Options, page 53.

Note. When creating a report definition, you can select a PS Query data source that has not yet been registered and that data source is registered automatically when you save the report definition. However, all other types of data sources must be registered before they can be associated with a report definition.

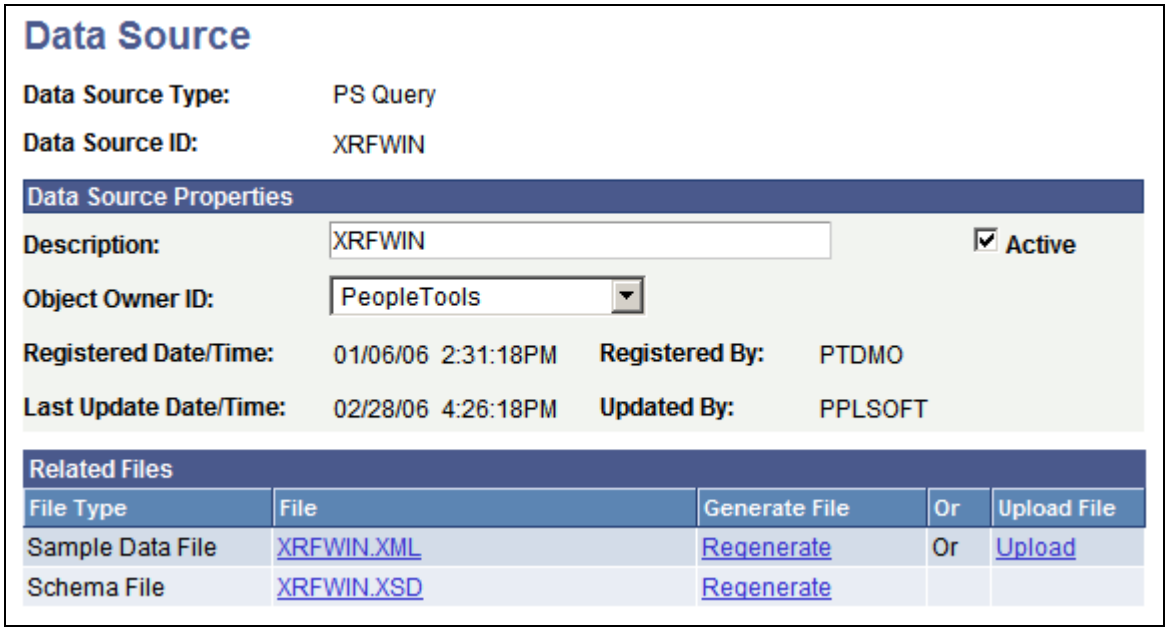
Note. You can check the format of an XML output file by opening it using Microsoft Internet Explorer (IE). IE opens the file and alerts you to any problems, such as unclosed tags.

Page Used to Register Data Sources

Page Name	*Definition Name*	*Navigation*	*Usage*
Data Source	PSXPDATASRC	Reporting Tools, XML Publisher, Data Source	Register existing processes that supply XML data for reports. Register optional schema and sample data files for XML Publisher data sources.

Registering Data Sources

Access the Data Source page (Select Reporting Tools, XML Publisher, Data Source.)



Data Source page

Data Source Type Select *PS Query*,*Connected Query*, or *XML File.*

Data Source ID Select or enter the data source ID.

When you are adding a new data source, for PS Query data source types, the corresponding data source ID listing is pulled from Query Manager.

For other data source types, this field accepts free-form text entry. Enter an ID that indicates what the data is, because you want to easily identify your data sources when defining reports.

Description (Optional) Enter descriptive text that provides more detail about the data source.

The description is automatically supplied by default to the data source ID.

Object Owner ID (Optional) Indicate which product, feature, or application owns this data source.

This field is used to extract and package production data source and report registrations and their supporting files.

Registered Date/Time This is a read-only field maintained by the system that indicates the date that the initial data source registration was made.

Last Update Date/Time This is a read-only field maintained by the system that indicates the date that the last update to the data source was made.

Active Select to indicate that this is an active data source.

Only active data sources can be chosen when creating a new report definition. Only reports with active data sources can be processed.

Registered By This is a read-only field maintained by the system that indicates the user ID of the operator who initially registered the data source.

Updated By This is a read-only field maintained by the system that indicates the user ID of the operator who last updated the data source.

Related Files

The sample data file is an XML file with sample data that is used for data mapping, template preview, and determining burst fields. Preview action is available within a desktop template designer or from within the report definition page. For PS Query and Connected Query data source types the sample data file can be system-generated or uploaded. For XML file data source type, the sample data file must be uploaded,

---

Note. Prior to PeopleTools 8.50, the sample data field was only used for data mapping and preview.

In PeopleTools 8.50, the schema file is deprecated. In prior releases, the schema file which is an XML Schema Definition (XSD) file that defines the structure and elements that is in the extracted XML data, was used required for bursting. In PeopleTools 8.50, the sample data file is used for bursting.

File (Optional) Click the file name links to view the XML and XSD files after you have generated, regenerated, or uploaded them.

Last Update Date/Time (Optional) This is a read-only field maintained by the system that indicates the date that the last update to the related file was made.

Generate/Regenerate (Optional) Click the Generate link for PS Query or Connected Query data sources to generate the related sample data .

When the related files have been initially generated for PS Query or Connected Query data sources, click the Regenerate link to regenerate them in case the underlying query has changed.

Upload (Optional) Click the Upload link for XML file data sources to bring the related sample data and schema files into the database.

You can also upload a sample data file for PS Query or Connected Query if you would prefer to use a sample data file with more realistic data.

Note. A validation is run against the schema XSD file that is uploaded to a data source, alerting the developer if problems occurred while the system was using their schema.

Chapter 4

Creating Report Templates

This chapter provides an overview of report template types and discusses how to:

- Use RTF templates.
- Use PDF templates.

See Also

Oracle BI Publisher Core Components Guide.

PeopleTools 8.51 PeopleBook: PeopleCode API Reference, "XML Publisher Classes"

Understanding Report Template Types

Template design involves the construction of a report layout in a template file and is dependent upon what the core Oracle BI Publisher engines accept for processing.

The nature of the data plays a role in the selection of a template.

Note. Internet Explorer does not have native support for svg graphics; an SVGViewer must be installed for you to see charts in Internet Explorer. SVGViewer is available for free download on the Adobe website at http://www.adobe.com/svg/viewer/install/.

The following table lists and describes supported template types and provides guidelines for you to consider:

Template Type	*Description*
PDF Template	Reports are prerendered PDF forms that are populated with data at runtime. Starting in PeopleTools 8.50 nested structures are supported. Nested structures should not be used for any reports that need to be backwards compatible. This template type is suitable when you have existing PDF forms that you need to use to generate reports, such as government forms. Generally, using this template type is faster than using RTF templates because no runtime rendering is involved. Use PDF templates when: • You already have PDF templates that you must use (for example, government forms). • You have simple form-based reporting requirements with no complex formatting, that is, no charting, dynamic tables, dynamic repeated fields, and so forth.
RTF Template	Reports are full rendered, which means that the actual output is generated at runtime using XSLFO technology. Report designers have full control of output formatting and can incorporate charts, dynamic tables, conditional formatting, and so forth. Reports generation is generally slower than PDF-based reports because they involve real-time output rendering.
eText	eText templates are RTF-based templates that are used to generate flat-file text output that can be transmitted to a bank or other customer for Electronic Funds Transfer (EFT) or Electronic Data Interchange (EDI). Because the output is intended for electronic communication, these templates must follow specific format instructions for data placement. **Note.** XML file is the recommended data source for eText templates because the requirements for eText templates are very specific. XML produced by PS Query data sources lacks the required structure for eText templates and is therefore not available. See *Oracle BI Publisher Core Components Guide, eText Templates.*
XSL Templates	For more complex design requirements, a number of XSL and XSL-FO elements are supported for use with your XSL templates. See *Oracle BI Publisher Core Components Guide, XSL, SQL, and XSL-FO Support.*

Note. Sample report templates are bundled with the BI Publisher Desktop, and available in <Installation Directory>\BI Publisher Desktop\samples.

Using RTF Templates

RTF templates support most XSL functionality and can be designed with robust processing and formatting code.

This section discusses how to:

- Create RTF templates.
- Incorporate sub-templates.
- Include images.
- Change default template font.
- Use drilling URL in RTF template.

Creating RTF Templates

To create an RTF template using Microsoft Word:

1. Download the delivered BI Publisher Template Builder plug-in for offline template design on the Reporting Tools, XML Publisher, Setup, Design Helper page to facilitate the insertion of application data tags into your RTF templates.

   The BI Publisher Template Builder is an extension to Microsoft Word that simplifies the development of RTF templates. While the Template Builder is not required to create RTF templates, it provides many automated functions that may increase your productivity.

   Note. You can choose to automatically view the Word Template Builder Tutorial File, Template Builder for Word Tutorial.doc, upon installing the plug-in. This document offers a quick and informative tutorial of the Template Builder.

   The Template Builder for Word Tutorial.doc is located in the \Template Builder for Word\doc directory of the folder where Oracle BI Publisher Desktop, BI Publisher Template Builder plug-in, was installed.

   Sample report templates are available in <Installation Directory>\BI Publisher Desktop\samples.

   ---

2. Download the XML sample data file by clicking the Sample Data link on the Reporting Tools, XML Publisher, Report Definition page for a specified query.
3. Load the sample data into the document by selecting Data, Load XML Data from the Microsoft Word Template Builder tool bar menu.
4. Design your template in the RTF document.

   By using the downloaded XML sample data, you can insert the data field tags into your template rather than manually typing XSL-formatted tags.

   You can preview the template output with the sample XML data from the Oracle BI Publisher menu using Preview Template or select Preview from the Template Builder toolbar.

5. Upload the completed template into the report definition by clicking the Upload button on the Reporting Tools, XML Publisher, Report Definition, Template page.

Note. Your data source XML structure should be designed to be as close as possible to the groupings used for in the actual report template structure; this improves runtime performance by preventing unnecessary XSL transformation. This is particularly applicable for reports with complex data structures and very large file sizes.

See *Oracle BI Publisher Core Components Guide, Creating an RTF Template.*

See "\\Word Template Builder\doc\Word Template Builder Tutorial.doc."

See Chapter 2, "Setting Up XML Publisher," Working with Template Design Helpers, page 15 and Chapter 5, "Defining Report Definitions," Creating Report Definitions, page 39.

Incorporating Sub-Templates

When designing a template, you can incorporate one or more sub-templates into your primary template.

You must use specific syntax to:

- Create sub-templates.
- Import sub-templates.
- Call sub-templates.

See Chapter 5, "Defining Report Definitions," Maintaining Sub-Templates, page 60.

Creating Sub-Templates

Within a single sub-template file, multiple sub-template components can be available. Start and end template indicators must exist to distinguish these various components.

```
<?template:component_name?>
<?end template?>
```

For example, syntax of a sub-template file containing two components could be:

```
<?template:peoplesoft?>
Pleasanton Campus
500 Oracle Lane
Pleasanton, CA  94488
<?end template?>

<?template:logo2x.5?>
 Oracle_Logo
<?end template?>
```

where `<?template:peoplesoft?>` is the start template indicator of the component *peoplesoft* and `<?template:logo2x.5?>` is the start template indicator of the component *logo2x.5*. Each `<?end template?>` tag indicates the end of its respective component.

Importing Sub-Templates

To import a sub-template file that is stored in the Content Library, place the following syntax at the top of the primary template file:

```
<?import:psxmlp://sub-template_NAME?>
```

where `sub-template_NAME` is the registered sub-template ID in the Content Library, for example: `<?import:psxmlp://STDHEADER?>`. This syntax must be in `Normal` text.

Note. The sub-template reference is defined only in the RTF template. The sub-template must be defined in Content Library; however, the relationship to templates using the sub-template is not defined in the database. Developers must be aware of the sub-template relationships when modifying the RTF sub-template.

See Chapter 5, "Defining Report Definitions," Maintaining Sub-Templates, page 61.

Calling Sub-Templates

Place the following syntax in the primary template file in the location where the desired text or XSL instructions from the sub-template file should appear:

```
<?call-template:peoplesoft?>
```

In the preceding sample code `peoplesoft` is the name of the component that you want to use in the sub-template file.

Note. Primary templates calling nonexistent or inactive sub-templates cause an error message to be issued indicating the reason for the problem. This error information is incorporated into Process Scheduler error handling as well as into online viewing or previewing of the report.

See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Running XML Publisher PeopleSoft Query Reports, page 69.

Testing a Sub-Template in Microsoft Word

You should test your template and sub-template using Template Builder before uploading to PeopleTools to make your sub-template is accessible to your template on the file system.

Use the following syntax when importing:

```
<?import:file:C:///Template_Directory/subtemplate_file.rtf?>
```

Notice the triple slashes and the use of the actual file name instead of template ID.

When your design is complete, you can change the import statement back to make the sub-template available to the main template in PeopleTools environment:

Including Images

BI Publisher supports a number of methods for including images in your reports:

- Inserting images.

- Importing images.

Inserting Images

To directly insert a .jpg, .gif, or .png image file into a template:

1. Select Insert, Picture, From File while the template is open in Microsoft Word.
2. Select the desired .jpg, .gif, or .png file to insert into the template.
3. Save the template.

Note. Oracle recommends that you use the Microsoft Word *Insert* menu option to insert the image, because the additional properties that you need to set for the RTF template to correctly generate reports with those images are automatically set by means of this method. Additionally, dragging and dropping an image onto a template creates a link to the local machine being used and may cause problems when the report is generated.

Importing Images

To import an image from a sub-template file:

1. Embed the .jpg, .gif, or .png into the sub-template file.

   For example,

   ```
   <?template:logo2x.5?>
    Oracle_Logo
   <?end template?>
   ```

   where `Oracle_Logo` is the actual .jpg, .gif, or .png.

2. Import the sub-template file that includes the image by including the following syntax at the top of the primary template file:

   ```
   <?import:psxmlp://sub-template_NAME?>
   ```

   In this code sample, `sub-template_NAME` is the registered sub-template ID in the Content Library.

3. Add the calling details in the primary template at the appropriate location using the following syntax:

   ```
   <?call-template:logo2x.5?>
   ```

   In this code sample, `logo2x.5` is the name of the component that contains the image in the sub-template file.

See Chapter 4, "Creating Report Templates," Incorporating Sub-Templates, page 28.

Changing Default Template Font

The output report from RTF template uses template-level default fonts for empty report spaces and empty table cells. If the default font size does not match the font height used in a template, a final report could look different from user expectations. In this case, the user can change the template default font either in design time or runtime:

- Design time

  Set the xdo.cfg for the font. For example, set the default font for a specific report to be Helvetica, size 8:
  ```
  <property name="rtf-output-default-font">Helvetica:8</property>
  ```

- Runtime

  Use PeopleCode to set the font. For example, set the default font for a specific report to be Times New Roman with height 10:

  ```
  &asPropName = CreateArrayRept("", 0);
  &asPropValue = CreateArrayRept("", 0);
  &asPropName.Push(""rtf-output-default-font");
  &asPropValue.Push(""Times New Roman:10");
  &oRptDefn.SetRuntimeProperties(&asPropName, &asPropValue);
  ```

Using Drilling URL in RTF Template

Drilling URLs are supported in XML Publisher reports with a data source of PS Query or Connected Query.

Note. Drilling URLs are supported only in RTF templates.

To use a drilling URL in a XML Publisher report:

1. Create the query with the drilling URL defined as a field.

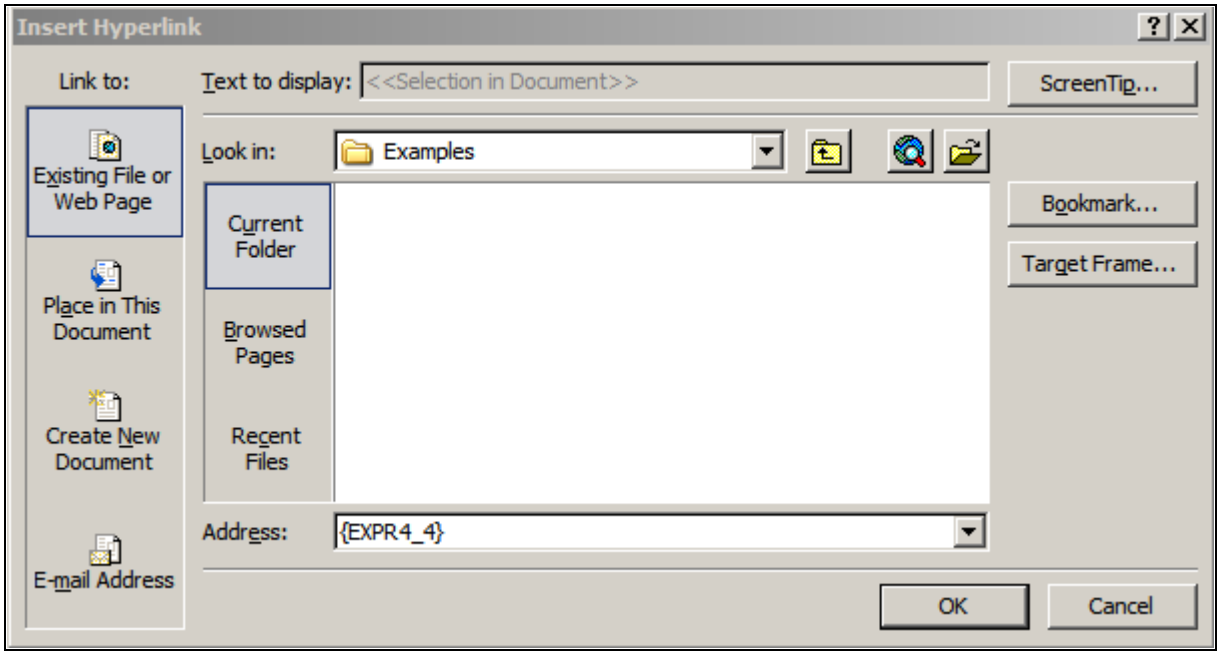
   See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Query*, "Defining Selection Criteria," Drilling URL in Oracle PeopleSoft Query.

2. Create an RTF template.

3. In the RTF template map one or more fields to the fields that contain drilling URL.

   - Highlight the field where you want to place the drilling URL.
   - Select Insert (from the Word menu), Hyperlink or use Ctrl+K.
   - In the Insert Hyperlink dialog box, enter the URL link in Address field.

     Each URL link should be defined as {URL_FIELD}, where URL_FIELD is the unique field name for the expression that contains a specific drilling URL.



Insert Hyperlink dialog box

   - Use the Target Frame push button to select how this URL link will be opened, either in the same window or in the new window

---

Note. The URL value does not need to be added to the report, as long as the unique field name (data file that contains the URL) is mapped to a field in the report.

---

4. If the XML Publisher report is run in Process Scheduler using an application engine program, you will need to add additional code to identify the process instance for the application engine program before processing the report. The process instance can be retrieved from the state record PSXPQRYRPT_AET. This call is needed to set a drilling URL during Query or Connected Query execution.

```
&ProcessInstance=PSXPQRYRPT_AET.PROCESS_INSTANCE;
&oRptDefn.ProcessInstance = &ProcessInstance;
&oRptDefn.ProcessReport("", "", %Date, "");
```

5. When you click the drilling URL in the report depending on the drilling URL type, one of the following occurs:

    - For Query URL, the Query results are displayed.
    - For Component URL, the appropriate PeopleSoft page is opened.
    - For External URL, the external page is opened.

See Also

Oracle Business Intelligence Publisher User's Guide, Creating an RTF template, Template Features, Hyperlinks

Using PDF Templates

This section discusses how to:

- Work with PDF templates.
- Create PDF templates.
- Map data tags.

See Chapter 5, "Defining Report Definitions," Associating Templates, page 43.

Working with PDF Templates

PDF templates do not require an external plug-in for offline template design. A mapping feature for XML data element tags is enabled when the PDF template file is uploaded to the Reporting Tools, XML Publisher, Report Definition, Template page. The XML Publisher PDF mapping functionality enables you to match existing form fields in a PDF template with sample data field tags.

You only need to do mapping, if the form field names in the PDF template do not match the tag names in the XML data. This is usually the case when you are using a third-party PDF template (such as government form) and when it is not easy to customize the tag names in XML data to match the PDF form fields.

Observe the following guidelines when working with PDF templates regardless of whether you are mapping PDF template fields or tags:

- The PDF document must allow editing.

  Check the Security setting in the File, Document Properties, Summary page.

- Files must be Adobe Acrobat 5.0-compatible.

  If you are using a later version of Adobe Acrobat, select File (or Document — depending on the version of Adobe), Reduce File Size and select the *Acrobat 5.0 and later* value in the Make Compatible with: option.

- Files must have form fields for the placement of application data, with each field tag being unique.

  If no form fields exist, use the Adobe Professional version to add field tags. If duplicate tags or nonunique tags are in forms obtained from third parties, use Adobe Professional to update the tags.

- Files should not have embedded javascript.

  XML Publisher removes it during the course of the `Map Enablement` function when the Generate button on the Reporting Tools, XML Publisher, Report Definition, Template page is selected.

Processing PDF Templates

The Oracle BI Publisher Core Engine adheres to the following rules when processing PDF templates:

- The search for the tag that matches the field name starts at the end of the XML file.
- The first match of the tag is used for the field value.
- If the tag is not found, the Oracle BI Publisher Core Engine looks at the map file (if provided).

  This means that even if the form is mapped, when a tag is in the XML data that matches the PDF template form field tag, it has priority over the map for placing the data.

Using Full Path Mapping

The PeopleSoft implementation for PDF mapping supports full path mapping. Full path mapping is not supported in the BIP Server. Full path mapping should only be used when you have no control on the structure and names of your XML data tags. For you to use full path mapping, the data source definition must include sample XML with no schema.

Note. Full path mapping is available only for PeopleTools 8.50 and later. It is not backward compatible.

Creating PDF Templates

To create a PDF template without mapping tags using Adobe Acrobat:

1. Design your template in the PDF document as documented in the *Oracle BI Publisher Core Components Guide.*

   Be sure that the PDF template field names match the XML data tags.

2. Upload the completed template into the Report Definition by clicking the Upload button on the Reporting Tools, XML Publisher, Report Definition, Template page.

Note. BI Publisher supports Adobe Acrobat 5.0 (PDF specification version 1.4). If you are using a later version of Adobe Acrobat, use the File, Reduce File Size option to save your file as Adobe Acrobat 5.0-compatible.

See *Oracle BI Publisher Core Components Guide, Creating a PDF Template.*

Mapping Data Tags

Third parties most often supply PDF templates in which the form fields already exist inside the form template. For the XML data element tags to know where they should print within the PDF template, a mapping is required between the field elements from the data source and the form field elements in the PDF template. Once a PDF form with editable form fields is mapped to the XML sample data fields, the template is ready for use by XML Publisher.

Prior to being able to perform this mapping, some XML Publisher-specific pre-processing of the file is required. This processing requires the existence of an open sample data and the report's data source. Adobe Standard or above and version 6 or above is required for the template mapping. In the event that the PDF form does not have form fields, the form field and tags can be inserted using the Designer or Professional versions of Adobe. The form field tags can then be mapped to the sample data tags.

To create a PDF template by mapping data element tags using Adobe Acrobat:

1. Upload the PDF template file to be mapped by clicking the template file Upload button on the Reporting Tools, XML Publisher, Report Definition, Template page.

2. If you are using full path mapping, select the Full Path Mapping check box.

3. Generate the file to be mapped by clicking the map file Generate button.

   Generate creates a map-enabled PDF, with the following naming convention. The plug-in enables you to access the data tags by embedding a JavaScript plug-in inside the PDF template.

   - A dash and the letter *m* added at the end of the file. For example, if the original file is *template.pdf*, the mapped file will be *template-m.pdf* if path mapping is not selected.

   - A dash and the letter *mfp* added at the end of the file. For example, if the original file is *template.pdf*, the mapped file will be *template-mfp.pdf* if path mapping is not selected.

4. Visually map the data tags to the form's field tags.

   The mapping exercise is performed offline within the Adobe Acrobat application.

5. Save the file.

   The generated file name indicates the type of mapping, as previously defined in step 2.

6. Upload the mapped PDF file on the Reporting Tools, XML Publisher, Report Definition, Template page by selecting the map file Upload button.

   When uploaded to the server, the mapping information is stored in the database along with the PDF form template.

Note. If the PDF template's field names are the same as the data source's data tag names, then no mapping or uploading of a map file is required.

Note. PDF file security has to allow editing and saving for the mapping to be completed. The ability to perform these functions depends on the Adobe version that you are working with.

Note. If no map file exists for your PDF file, selecting the Preview button on the Reporting Tools, XML Publisher, Report Definition, Template page will not show any data because the form fields names do not match XML data tag names.

XML Publisher and Adobe

XML Publisher provides the following features within Adobe:

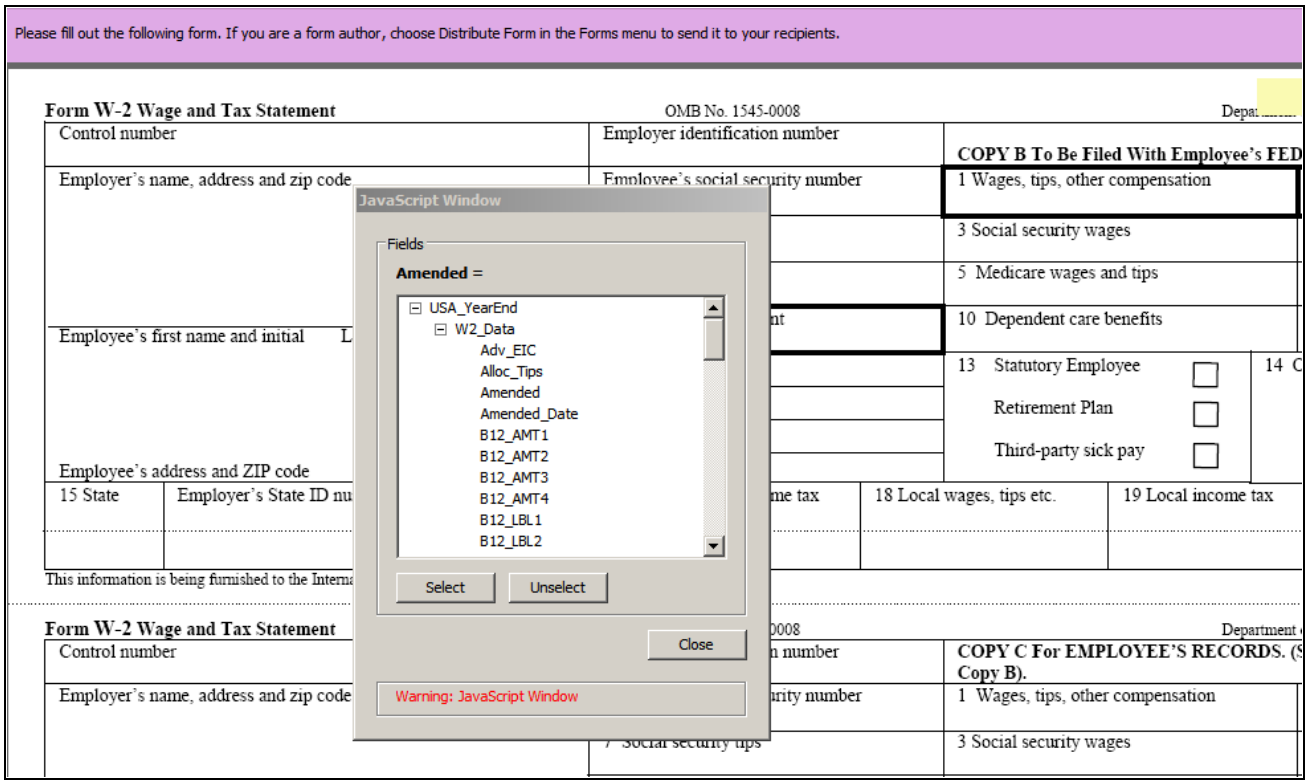
- A visual indication of the PDF form fields that have been mapped.

  A dark blue outline appears around the mapped form fields.

- Display of the mapped field tag name when the cursor hovers over the PDF form field.

- A pop-up dialog box containing an XML tag list that you can select from to insert the field tag when you click the PDF form field.

- Preparation of the PDF form for uploading to the report definition when you save the file locally by doing a File, Save.



Sample PDF file mapping

Chapter 5

Defining Report Definitions

This chapter discusses how to:

- Create report definitions.
- Assign report viewers at runtime.
- Maintain sub-templates.
- Maintain template translations.

Creating Report Definitions

This section provides an overview of report definitions and discusses how to:

- Define reports.
- Associate templates.
- Set output options.
- Set report properties.
- Set security options.
- Set bursting options.

Understanding Report Definitions

Report definitions associate a data source with template files. A data source registers the schema and sample data design files. The extracted application fields from the data source files are placed into the template files to create the final report.

A report can include multiple templates. A template is used to associate different layout formats as required by different countries and regions or as required by different channels (web posting, printer, fax, and so on).

The defined output options from the report definition are reflected on the output type and format prompts on the Process Scheduler request page when the application process that runs the report is assigned the process type of XML Publisher. Security settings for a report definition determine who can view the report when it has been run.

Report properties can be set to control formatting of the report.

With the advanced bursting feature, report generation results in separate output files when bursted reports are run through Process Scheduler.

Report definition access is based on user permission list security and roles. For example, bursting is read-only for XML Publisher power users, because only developers can set up bursting, and the page only appears when settings exist.

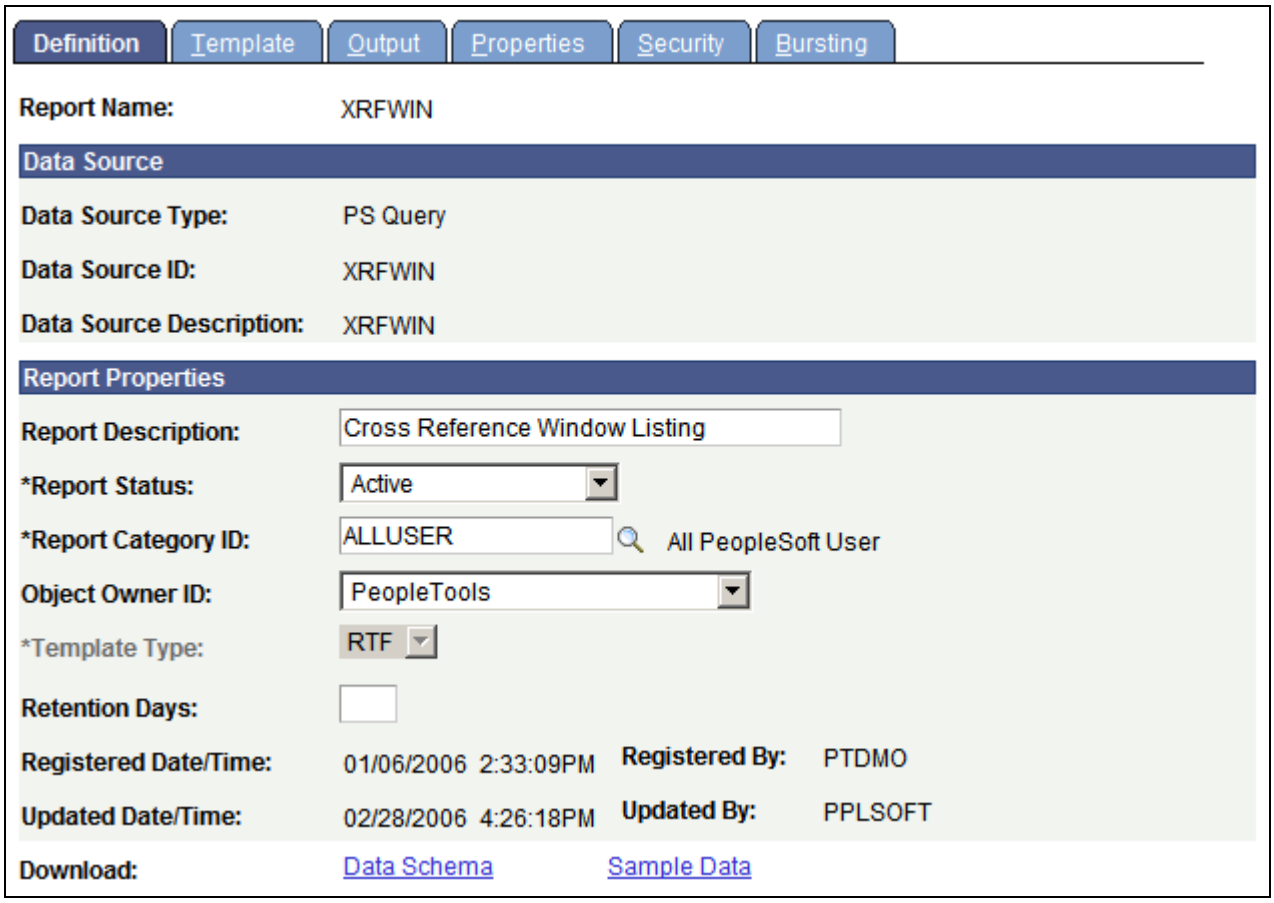
XML Publisher power users can start to define a report to download the sample data files to create their templates.

Pages Used to Create Report Definitions

Page Name	*Definition Name*	*Navigation*	*Usage*
Definition	PSXPRPTDEFN	Reporting Tools, XML Publisher, Report Definition, Definition	Define reports.
Template	PSXPRPTTMPL	Reporting Tools, XML Publisher, Report Definition, Template	Associate templates.
Output	PSXPRPTOUT	Reporting Tools, XML Publisher, Report Definition, Output	Set output options.
Properties	PSXPRPTPROP	Reporting Tools, XML Publisher, Report Definition, Properties	Set report properties to override global properties.
Security	PSXPRPTSEC	Reporting Tools, XML Publisher, Report Definition, Security	Set security options.
Bursting	PSXPRPTBURST	Reporting Tools, XML Publisher, Report Definition, Bursting	Set bursting options.

Defining Reports

Access the Definition page (Reporting Tools, XML Publisher, Report Definition, Definition.)



Report Definition-Definition page

Report Name

Enter a report name.

The report name must be unique, and it must not contain any special characters. If you enter spaces in the report name, the system replaces them with underscores.

Data Source Type

Select *Connected Query*,*PS Query*,*Rowset*,*XML Doc*, or *XML File*.

Note. For XML Publisher power users, the data source type is *PS Query* only and the drop-down list box is disabled.

Rowset and XMLDoc are deprecated in PeopleTools 8.50. If the data source was defined in a previous release, it will be available. You can not create a new data source for rowset or XmlDoc.

See Chapter 3, "Creating and Registering Data Sources," Registering Data Sources, page 21.

Data Source ID

Select the data source ID.

You can choose from data source IDs that are based on previously registered data sources. You can select queries regardless of whether they have been previously registered as data sources. For queries, the lookup table respects the public, private, and query access group security for the current user ID.

When you save a report definition with an unregistered query data source, the query is systematically registered as a data source. The query has no object owner ID, but that value can be entered manually on the Data Source page, if required.

Data Source Description

This is a read-only field that reflects the value that was entered when the data source was registered.

For unregistered query data sources, this field reflects the query description.

Report Description

(Optional) Enter descriptive text that provides more detail about the report.

If this field is left blank, the report name appears by default.

Report Status

Select *Active*,*In Progress,* or *Inactive.*

Setting the report status allows work in progress as well as retirement of report definitions. Active reports must have at least one active template. Only active reports can be selected at runtime and run to success.

Report Category ID

Select a report category ID.

This is a grouping mechanism for reports that provides row-level security for editing report definitions per the rights defined on the report category setup table.

See Chapter 2, "Setting Up XML Publisher," Setting Up Report Categories, page 11.

Object Owner ID

(Optional) Indicate which product, feature, or application owns this report.

Note. The default value that appears here is based on the Object Owner ID setting in the Report Category component (PSXPSETUPRPTCAT).

Template Type

Select *PDF*,*RTF*,*ETX,* or *XSL.*

Note. ETX is only available if the data source is XML file.

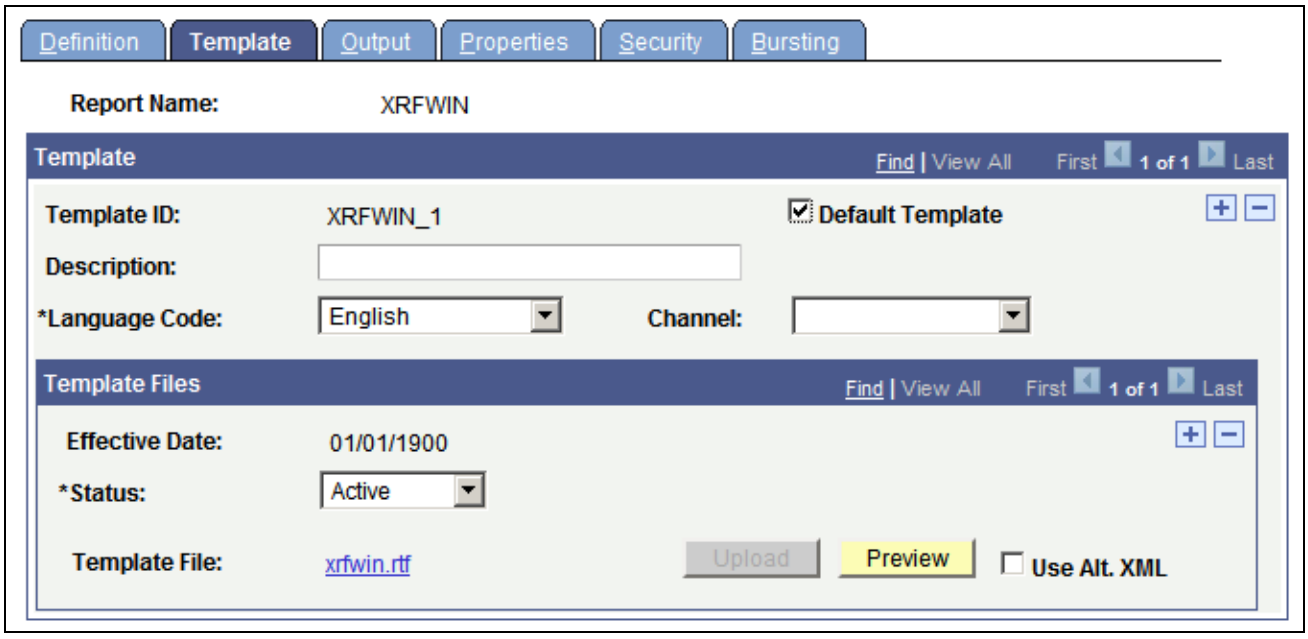
Only one template type is allowed per report.

The template file extension that you can upload on the Template page is controlled by this value. This value also controls which report templates appear on the Translation component (PSXPTMPLTRNS), because only RTF templates are translatable.

Retention Days	(Optional) Enter a value to set the option to purge the reports from the Report Repository and archive the data to the Report Archive table. The value that you enter overrides the system setting for retaining reports. The maximum value that you can enter is *999* days. If you don't select a value, the value from the PeopleTools, Process Scheduler, System Settings page applies. Only XML Publisher report developers or power users with permission list PTPT2600 or PTPT2500 can set this value. See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Report Manager," Maintaining Reports.
Registered Date/Time	This is a read-only field maintained by the system that indicates the date that the initial report definition was registered.
Updated Date/Time	This is a read-only field maintained by the system that indicates the date that the last update to the report definition was made.
Registered By	This is a read-only field maintained by the system that indicates the user ID of the operator who initially registered the report definition.
Updated By	This is a read-only field maintained by the system that indicates the user ID of the operator who last updated the report definition.
Download	Click Data Schema to detach the schema file or Sample Data to detach the data file. Detaching the files enables the user to view the data elements prior to finalizing the report definition. These links appear if the related files exist on the registered data source. For PS Query data sources, both links always appear regardless of whether the data source is registered because these files are system-generated. See Chapter 3, "Creating and Registering Data Sources," Registering Data Sources, page 21.

Associating Templates

Access the Template page (Reporting Tools, XML Publisher, Report Definition, Template.)



Report Definition-Template page (RTF template)

The Template group box on the Template page refers to a particular template layout, because one report definition can associate multiple template layouts differentiated by language code or channel.

Template ID

Enter a template ID that uniquely identifies this template.

The default template ID is a system-generated ID based on the report name. You can edit this ID when you first add a template to the report definition, but it must be unique across all templates in the system, not just within the current report definition.

Description

(Optional) Enter descriptive text that provides more detail about the template and identifies its use.

Entering a meaningful description helps the user select the proper template at runtime. For example, indicate a unique layout or channel.

Language Code

Select a language code for the template.

The default value reflects the default template language.

Default Template

Indicate whether this is the default template.

You can select only one template as the default template. The first template that you add to the report definition is automatically selected as the default. You can change this selection as necessary.

Default templates are automatically used at runtime if no other value is supplied.

Channel (Optional) Select the distribution channel for the template.

The Channel attribute supports the need to identify different layout formats as required by the various distribution mechanisms. For example, a printout may require a different template layout than an email or a web posting. Leaving the channel blank would indicate that this particular template does not have a format that is specifically suited to just one channel.

These values are for information only and do not trigger a particular Process Scheduler distribution mechanism. Developers can drive a template choice based on channel through the PeopleCode XML Publisher classes.

Adding Template Files

Within each template layout defined previously is one or more effective-dated versions of the template. For example, you can have a new government form for each year. In the Template Files group box, you attach effective-dated files that are the actual report templates.

Effective Date Select an effective date for the template file in order to maintain new versions or versions specific to a particular time period. For example, a new file could be uploaded to reflect a new format, effective for reports as of the new date.

The default date for a newly added template file is the current system date. The user can change the data per effective-dating logic with Update, Update/Display, and Correction modes.

See *PeopleTools 8.51 PeopleBook: PeopleSoft Applications User's Guide*, "Using PeopleSoft Application Pages," Using Effective Dates.

Status Select a status of *In Progress,Active,* or *Inactive* for the template file.

This field indicates the usability of the template file. Runtime selection logic for a template file uses this field in conjunction with the Effective Date field to determine which template file to use. At least one file must be active to save a report definition.

Template File When you upload the template, the template name appears as a link. Click this link to download the template file to your local computer for updating the field or tag assignments.

Upload Click to attach a template file to the template.

The file extension is checked against the template type value on the Definition page and a warning is issued if no match is found.

When you save the report definition, this button becomes disabled. To reupload a new version of the template, you must either delete and add it again in correction mode or add a new effective-dated row.

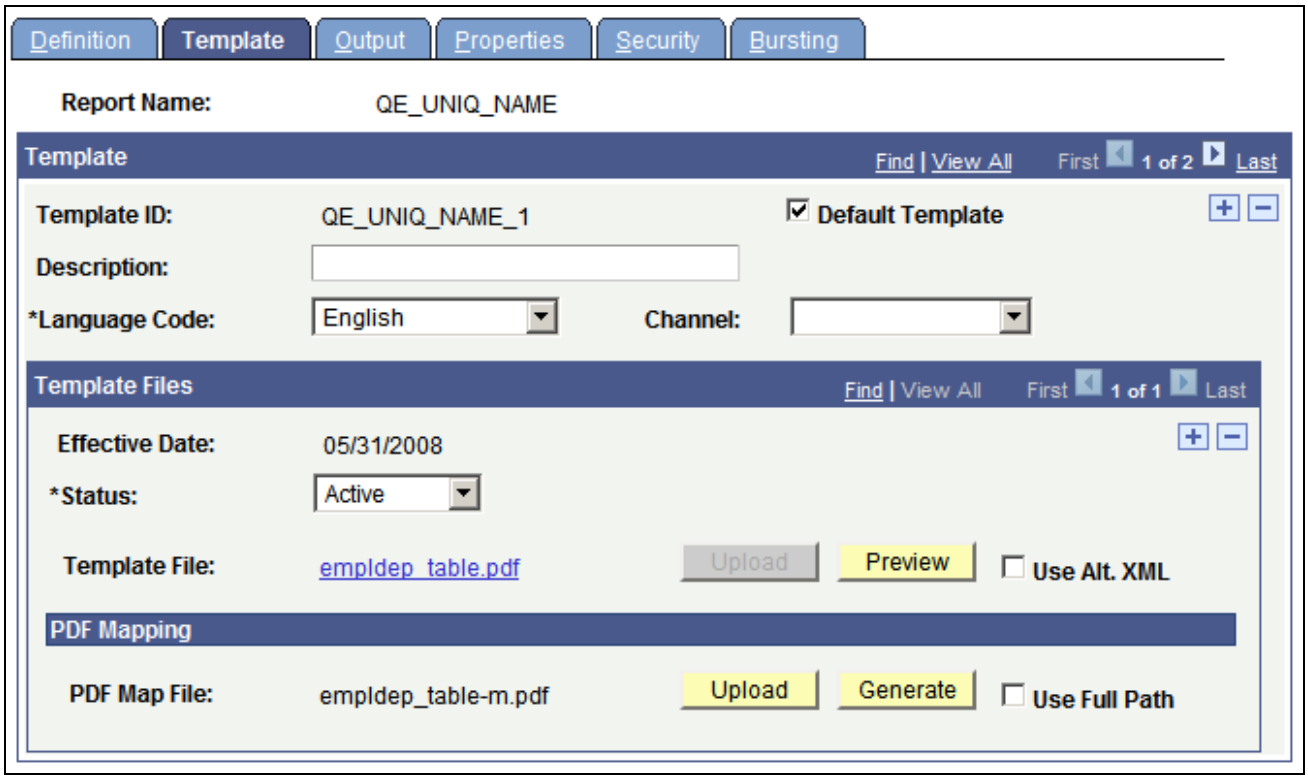
Preview	Click to preview the report using the current template file based upon the sample data file that was registered with the data source. The Preview button is not enabled when no sample data file is registered with the data source. The preview tab title depends on the default output type as follows: • PDF output uses the template name with a system-generated number. • Html output uses the title property from the word template. To change the title property in MS Word, select File, Properties, Summary.
Use Alt. XML (Use alternate XML)	Select to use an alternate XML file for previewing. When you click the Preview button, a dialog box appears, where you can select the file.

Note. The preview button uses the sample XML data file to generate report output. Sometimes, if the sample data does not match the real data, you may find discrepancies between preview and real report outputs. This is specifically true when the report template uses sample data in variables and conditional formatting. Creating your own sample file with real data makes the report look more realistic. This sample file can also be used to preview reports using template builder.

See Chapter 4, "Creating Report Templates," Mapping Data Tags, page 35.

Mapping PDF Template Files

For PDF files, a mapping is sometimes required between the field elements from the data source and the form field elements on the PDF template in order for the XML data element tags to print in the correct place within the PDF template. This is often true for third-party PDF templates, for which the form fields already exist inside the form template. However, if you create PDF form fields and XML tag names that are the same, no mapping is necessary.



Template page

The following fields appear on the Template page for PDF templates files:

Map File

When you upload the mapped PDF file, the file name appears as a link. Click this link to open or download the file to your local computer.

If changes are required in the map file, you can make the changes and upload the revised file without creating a new effective-dated row.

Generate

Click to generate the PDF map file.

The system uses the uploaded PDF template file and the sample XML data associated with the data source definition to generate a PDF template embedded with a Visual JavaScript plug-in used for mapping.

Any changes made to XML tag names and structure after the template is defined or mapped, require you to redefine or remap the template.

Note. PDF file security must allow altering and saving for the mapping to be completed. This depends on the version of Adobe with which you are working.

When working with PDF map files, some indication of mapping file should be included in the file name to distinguish the mapping file from the unmapped template file. By default, the generated mapping file name is the name of the template file followed by a dash and either an *m* for map file or *mfp* for full path mapping.

Upload

Click to upload the PDF map file when the tags have been mapped.

Full Path Mapping Select this check box if your XML data has elements with the same name at different levels. For instance, ADDRESS is used at the company level and also at the employee level.

This is an example of XML file that requires full path mapping:

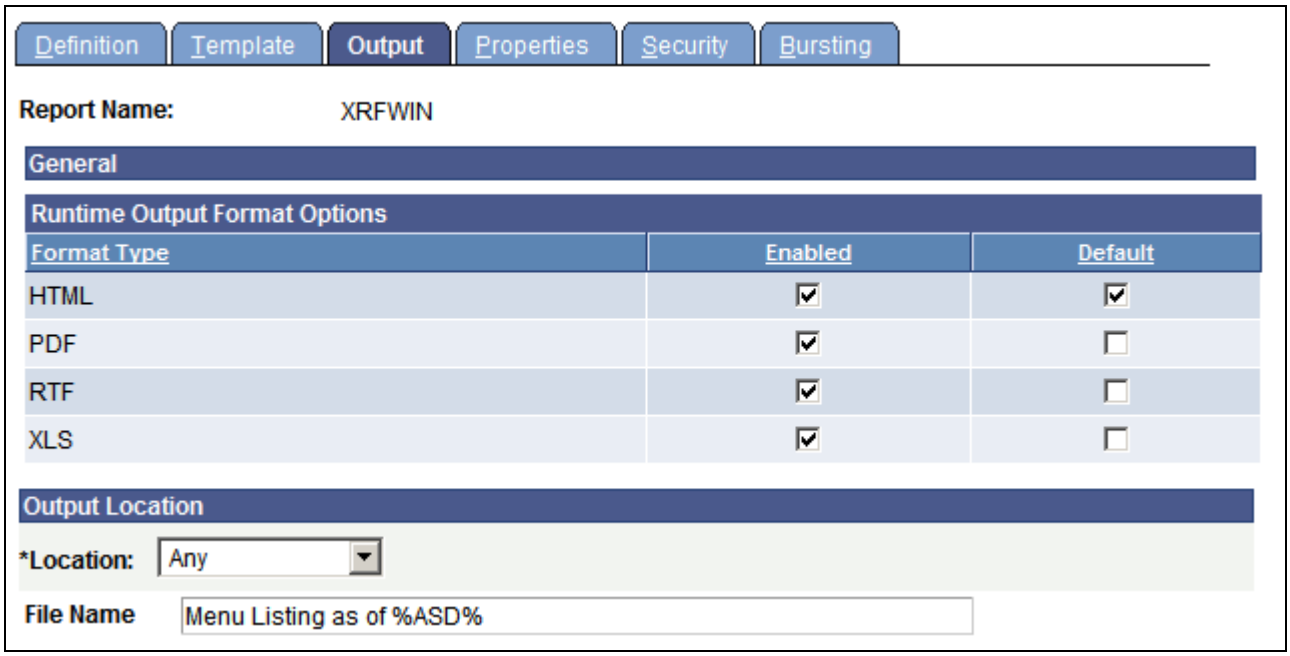
```
<PayChecks>
  <PayCheck>
    <EmpNo>00001</EmpNo>
    <CompanyInfo>
      <Address>1 Company st. CA 00001</Address>
      <Description>Company Info</Description>
    </CompanyInfo>
    <EmployeeInfo>
      <Address>1 Employee st. CA 00001</Address>
      <Description>Employee Info</Description>
      <Salary>50000</Salary>
      <Vacation>12</Vacation>
      ......
    </EmployeeInfo>
  </PayCheck>
  <PayCheck>
    ......
  </PayCheck>
  <PayCheck>
    ......
  </PayCheck>
</PayChecks>
```

The JavaScript plug-in will use the full path for address data elements instead of the element name. So it will use *PayCheck.Employee.Address* to map to the employee' address form field, and use *PayCheck.Company.Address* to map to the company's address field.

See Chapter 4, "Creating Report Templates," Mapping Data Tags, page 35.

Setting Output Options

Access the Output page (Select Reporting Tools, XML Publisher, Report Definition, Output.)



Report Definition-Output page (RTF template)

Format Type Dynamically lists the available output formats based on the template type.

Enabled Select specific values to limit the output choices for the user at runtime.

Default Select a default format type.

This value appears at runtime on the prompt or run control page. It specifies the output format that the system uses if no other value is fed into the XML Publisher engine.

Location

Select one of the following locations:

- *Any* indicates that the user can select the output location at runtime.
- *Email* indicates that the output goes to email.

Note. The users defined in the distribution list must have a valid email address defined in the user profile. If Allow viewer assignment at report runtime is selected, you can enter additional email addresses at runtime.

- *File* writes the output to the file that you indicate in the Output Destination field.
- *Printer* indicates that the output goes directly to a printer.

  Specify the printer destination for the output in the Printer field. This field is available only when the output location that you select is *Printer*.

  Printer is a valid selection only when PDF output format is enabled.
- *Web* indicates that the output goes to a web report repository that is accessible by the Report Manager.

  Select the folder for the output from the Report Manager Folder Name lookup. This field is available only when the output location that you select is *Web*.

  This is the default location used at runtime if no location is selected.
- *Window* indicates the output will be posted, like Web output, to the report repository and then streamed to the browser window, the same way scheduled query runs to Window.

Note. Window output is supported for scheduled and non-bursted reports only. Users building a custom process request page should check for the bursting field name (BurstFieldName) in the ReportDefn class before issuing a process request.

File Name

Specify a file name template that gets translated at runtime to a physical file name. This field accepts a combination of output variables and plain text.

Output variables are enclosed within percent signs (%) and used as part of the descriptive report name on report search page. The following variables are supported.

- *%ASD%* inserts the as of date.
- *%RID%* inserts the report ID.
- *%BTV%* inserts the burst field value.
- *%field%* where field is the name of a field from the XML data that lies below the first repeating field. For example, if you want the employee ID value to appear in the file name, you would use %EMPLID%.

Note. All burst key candidates on the bursting page are eligible.

For example, if you have a report CERTIFICATE that is burst by STUDENT_ID, you can use the file name to provide more details:

- If no file name is specified, the report description will use the report name, such as CERTIFICATE[2916]-CERTIFICATE.HTM.
- If a file name of *LOCATION %TRAINING_LOC% %END_DT%* is specified, the report description will include the variables, such as CERTIFICATE[2916]-LOCATION BOSTON 2009–03–13.HTM.
- If a file name of*%STUDENT_NAME%* is specified, the report description will include the variables, such as CERTIFICATE[2916]-LEE,JAMES.HTM.

Note. If you leave the File Name field blank, the system uses the report ID as the file name. For bursted report, burst value can be used as file name if set programmatically through the ReportDefn class property UseBurstValueAsOutputFileName. The Userfilename can also be set programmatically as a property of the ReportDefn class. If a Userfilename is set either in PeopleCode or on the page, it overrides the UseBurstValueAsOutputFileName property.

Note. The XML Publisher report definition output options are reflected in the output type and output format prompts on the Process Scheduler Request page only when the application process that runs the report is assigned the process type of XML Publisher.

Output Format Options

The output options are based on the template type as shown in this table:

Template Type	*Output Options*
RTF	.pdf, .html, .rtf, .xls (html)

Template Type	*Output Options*
PDF	.pdf
E-Text	.txt
XSL	.pdf, .html, .rtf, .xls (html)

Printing XML Publisher Report Output

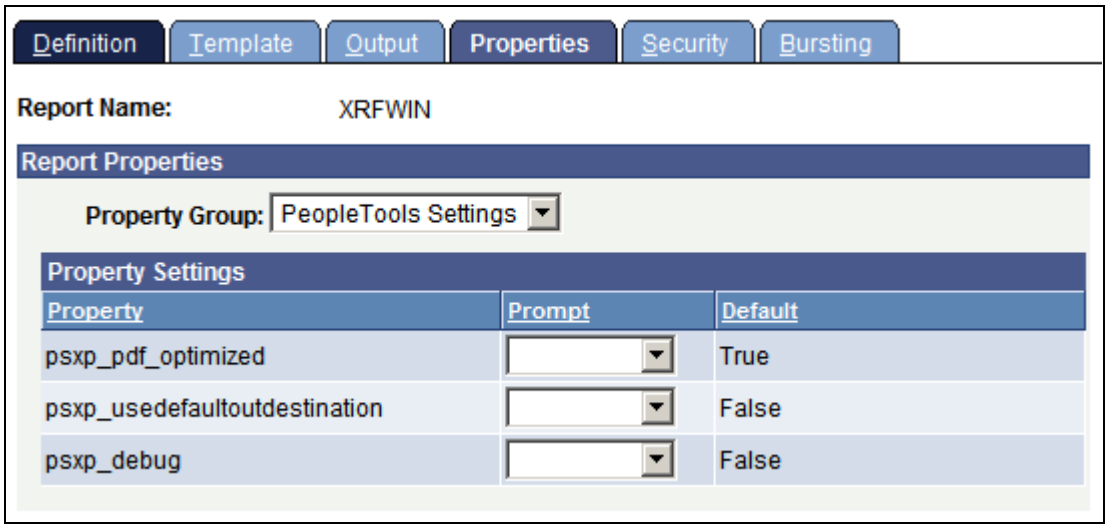
PeopleSoft applications support batch printing XML Publisher reports directly from a server using PDF output format. When you select *Printer* as the output location, PDF is the only output format displayed in the Process Scheduler Process Request Dialog page. When PDF format is not supported for a report definition, printing is not supported for that report. If you are not printing directly upon posting the report, you must open and print the report from Adobe Acrobat. All bursted output reports are sent to a single printer, but as multiple print jobs.

You can also convert the generated PDF files to other conventional printer output formats with an external software program. PeopleSoft applications provide PeopleCode support for inserting conversion logic from PDF to different printer formats.

See *PeopleTools 8.51 PeopleBook: PeopleCode API Reference*, "XML Publisher Classes"; *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Submitting and Scheduling Process Requests," Scheduling Process Requests and Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Customizing Printed Report Output, page 77.

Setting Report Properties

Access the Properties page (Select Reporting Tools, XML Publisher, Report Definition, Properties.)



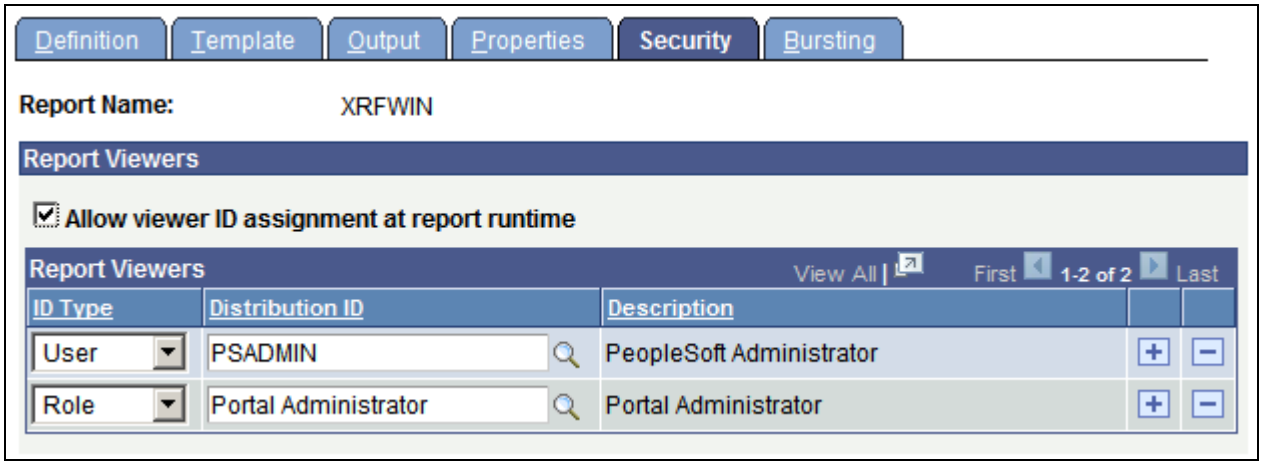
Properties page

Properties defined in the report definition will override the global properties for this report.

See Chapter 2, "Setting Up XML Publisher," Defining Global Properties, page 12.

Setting Security Options

Access the Security page (Select Reporting Tools, XML Publisher, Report Definition, Security.)



Report Definition-Security page

The Security page captures attributes regarding who can view web-posted output in the Report Manager repository and through the XML Publisher Report Repository Search page.

Allow viewer ID assignment at report runtime Select to indicate that the report requestor can add to the standard Distribute To values on the Process Scheduler Request, Distribution Detail page.

Note. If you are using security join tables to limit report distribution, leave this check box cleared. If you add a user or role at runtime, the associated users will be able to view all bursted reports for that report instance.

ID Type Select an ID type of either *Role* or *User ID.*

Distribution ID Select a corresponding distribution ID based on the ID type.

Description Displays the related description of the distribution ID.

Note. The users and roles defined on this page can view all bursted reports. If you are using security join tables to limit report distribution, do not enter any roles or users on this page.

Setting Bursting Options

Bursting is an optional advanced feature that is available only when reports are run through Process Scheduler. It is not intended for real-time online viewing. It is typically used when you are repeating the generation of a templated report layout many times for multiple like sets of data, for example, generating a batch run on vendor purchase orders or customer invoices. With bursting, you can generate individual report files resulting in separate secured output, for example, generating a file for each vendor, customer or employee.

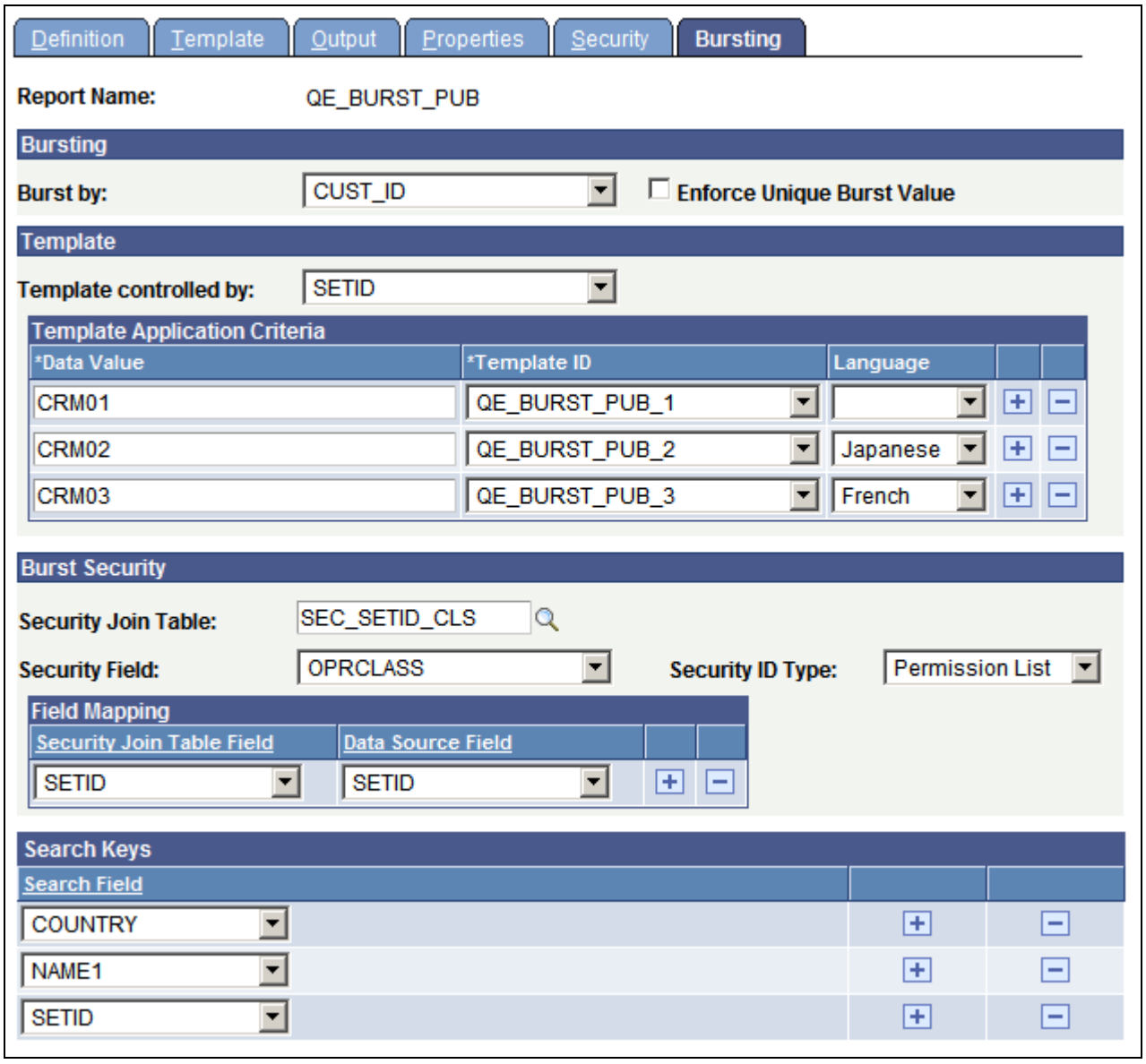
Setting up bursting requires thorough knowledge and understanding of data values and schema structures. You could possibly make entries on the Bursting page that would cause the report to fail at runtime. When you generate a bursted report, the system creates separate document files for each unique data value for a specified field tag.

Note. This Burst by field tag must be from the highest level repeating group (node) in the XML data. For bursting to work, only one high-level repeating group should be in the XML source.

Because bursting is an advanced feature, PeopleTools delivers permission list security that is intended for XML Publisher report developers (PTPT2600). When users are assigned a role with this permission list, they have access to setup entries on the Bursting page. A view-only permission list (PTPT2500) option also exists for XML Publisher power users that provides view-only access to the bursting information. The Bursting page appears for the power user only when bursting instructions exist for the report.

Note. In previous versions of XML Publisher, schemas were necessary for bursting. For backwards compatibility, you can still register and use schemas to define bursting. Sample files are the recommended approach since the existence of schema file in the data source definition indicates that backward compatibility is necessary and therefore schema will be used.

Access the Bursting page (Select Reporting Tools, XML Publisher, Report Definition, Bursting.)



Report Definition-Bursting page

Burst by

Select a burst by field to enable report bursting.

All subsequent bursting features are disabled until you select this value The values in the drop-down list box are the children from the highest-repeating level (group node) in the XML schema associated with the data source that is assigned to the report definition.

When you select a burst field, the report generates multiple files at runtime with a separate report instance file generated each time a unique value appears for the Burst by data tag. For example, this could be one report file for each employee when you are bursting by EmplID or one report for each department (that includes multiple occurrences of the report, one for each employee) when you are bursting by DeptID.

Use Unique Burst Value Select this check box to indicate that the Burst by field contains unique values. If a non unique value is found, the report will not be published and an error will be logged. It is recommended to use unique bursting values.

If this check box is cleared, bursted files with the same Burst by field will be combined in one report.

Note. Prior to 8.5x, unique burst value was not enforced. Non unique burst value will produce unpredictable results including incorrect search.

Template Assignment for Bursting (Optional)

This feature dynamically drives the template assignment at runtime based upon the data value of a designated schema tag. You can assign a language code to apply a specific template translation as well. This means that the various bursted report occurrences in one batch run can each have an appropriately assigned template and translation. For example, you can print Canadian paychecks in English or French depending upon the employee's preference.

You should select a template ID for each data value that requires a special template.

At runtime, the process looks for the specified template and language. If the language does not exist, then it applies the base untranslated template. If the process encounters a data value that is not assigned on the report definition, then it assigns the template ID that is entered on the run control. If the system captures no template ID selection at runtime, then it applies the default template of the report definition.

Template controlled by Select the schema tag value from the first child level to indicate the field with the template translation preference.

Data Value Enter a row for each data value that requires a specific template or template translation.

Template ID Select the template ID to apply when the data value specified previously is found in the XML data.

These drop-down list box values are dynamically determined by those already defined on the report.

Language (Optional) Select a language code for the desired translation of the template when the specified data value is found in the XML data.

The language choices in the drop-down list box reflect the complete list of available languages and are not limited by the existing registered Translation XLIFF files.

See Chapter 5, "Defining Report Definitions," Maintaining Template Translations, page 63.

Security for Bursting (Optional)

When a report is set up to be bursted, the report designer can also designate how the generated documents are secured when they are posted to the Report Manager. At runtime, the system uses this information to determine who can view each bursted report instance. You can use bursting security to supplement or replace the basic report viewer security by role or user ID. Otherwise, the system limits access to each report instance based on preexisting system security definitions.

The system automatically limits access to each report instance based on the Burst by field. For example, if the report is burst by employee ID, only the users designated with access to each employee ID can view the output file.

The report designer must provide the record name of the security join table and designate the common fields to join with the bursting field. The system performs the join and determines who can view the report instances. This matching allows the Report Manager's posting process to dynamically identify the user IDs or roles that are assigned viewing rights for each report instance.

Note. If a user has the role ReportDistAdministrator, that user can view all bursted reports, regardless of security join table.

Security Join Table	Select the record name for the table that stores either a user ID or a permission list assigned to a data value in the XML data.
Security Field	Select the field from the Security Join table that stores the user ID or permission list to secure on.
Security ID Type	Select either *User ID* or *Permission List* to indicate what type value is in the Security Field.
Security Join Table Field	Select the field from the Security Join table that joins with the schema data tag to identify the proper row from which to find the value in the in the Security Field that is used to secure the bursted file.
Data Source Field	Select the schema tag that stores the values that determines the security assignment. This may require more than one tag, because they must be first-child level tags. For example, they could be employee, customer, department ID, or a set ID/vendor ID combination, and so on.

Search Keys (Optional)

When report results are burst into separate files, you should be able to locate the desired individual report from the Report Manager repository. Delivered search keys include Burst By, Report Definition Name, and Generated On Date. You can define additional search keys to provide even more specific granularity.

At report runtime, the report posting program uses this information to store the key names defined here along with the specific data values for each burst report. From the XML Publisher Report Search page, users can use these configurable search fields to locate a specific report occurrence. For example, if the pay advice report runs regularly and posts numerous report files for self-service access, and as an employee you want to locate a particular dated advise, you would not want to browse through all the advise files to locate the one you want to see. By assigning the pay period as a Search Field in the report definition, the user can enter a date to search for the correct advise.

Search Field

Select an additional field to search on from within the XML Publisher Report Search page.

The drop-down list box values are taken from the children from the highest repeating level (group node) in the XML schema. Make sure that these values are unique per burst value.

At design time, you can select as many search fields as are required. However, at search time, the XML Publisher Report search page allows only two search criteria in addition to the Burst by value.

An API is provided to facilitate finding bursted XML Publisher reports in the Report Manager repository. When reports are burst into multiple separate files and posted in the Report Manager, the configurable search keys with their values are available as search keys in addition to Report Name, Burst By, Date, and Process Instance ID.

Note. The search feature uses Integration Broker functionality. The service operation PSXP_RATTR is used to insert XMLP report metadata for searching. This service operation must be active with a local-to-local routing.

See *PeopleTools 8.51 PeopleBook: PeopleCode API Reference*, "XML Publisher Classes," SearchAttribute Class.

Assigning Report Viewers at Runtime

There are three settings in the report definition that determine how web reports are distributed at runtime:

1. Report Viewer List on the report definition security page.

   Assign users and roles allowed to view the reports regardless of whether the report is bursted or not.

2. Security Join Table on the report definition bursting page.

   Assign users that can view individual bursted report files based on security join tables. These users are combined with the users and roles defined on the security page.

   Note. When security join tables are used, and the Allow viewer ID assignment at report runtime check box is selected, any users, roles or email addresses added at runtime will see all bursted reports. If roles or users are defined on the security page or at runtime, they can view all bursted reports ignoring the security join table.

3. Allow viewer ID assignment at report runtime check box on the report definition page.

   Allows the users running the report the ability to modify (add or remove) additional roles, users or email addresses on the runtime report distribution page.

This table describes how viewers are selected for non-bursted reports based on the report definition security settings.

Report viewers assigned	***Allow viewer ID assignment at report runtime***	***Viewers***
Yes	No	Reports are distributed to all roles and users defined on the security page. Runtime overrides are not allowed.
Yes	Yes	Reports distributed to all users and roles defined on the security page. Runtime overrides are allowed.
No	Yes	Distribution list is assigned at runtime on the Process Scheduler distribution detail page. By default the requester is added.
No	No	No reports posted to Report Repository. Runtime overrides are not allowed.

This table describes how viewers are selected for bursted reports based on the combination report definition settings.

Report viewers assigned	***Security join table implemented***	***Allow viewer ID assignment at report runtime***	***Viewers***
Yes	No	No	All bursted reports are distributed to all roles and users defined on the security page. Runtime overrides are not allowed.
Yes	Yes	No	All users and roles defined on the security page will see all bursted reports. Users defined from the security join table will see only the bursted reports based on their join criteria. Runtime overrides are not allowed.
Yes	Yes	Yes	All users and roles identified at runtime will see all bursted reports. Users defined from the security join table will see only the bursted reports based on their join criteria. Runtime overrides are allowed. Any users, roles or email addresses entered on the Process Scheduler distribution detail page will see all bursted reports.
Yes	No	Yes	All bursted reports are distributed to all roles and users defined on the security page. All users, roles or email addresses identified at runtime will see all bursted reports. Runtime overrides are allowed

Report viewers assigned	*Security join table implemented*	*Allow viewer ID assignment at report runtime*	*Viewers*
No	Yes	No	Users defined from the security join table will see only the bursted reports based on their join criteria. Runtime overrides are not allowed.
No	No	Yes	Assign distribution at runtime. By default requester is added. Any users, roles or email addresses entered on the Process Scheduler distribution detail page will see all bursted reports.
No	Yes	Yes	All users and roles identified at runtime will see all bursted reports. Runtime overrides are allowed. Users defined from the security join table will see only the bursted reports based on their join criteria.
No	No	No	No reports posted to Report Repository. Runtime overrides are not allowed.

Maintaining Sub-Templates

This section provides an overview of sub-templates and discusses how to maintain sub-templates.

Understanding Sub-Templates

You may have text, images, or logic in your templates that you want to reuse across many report templates. Examples include company headquarter address information or standard legal language. Rather than replicate this text, code in every template, or both, you can store sub-template files that include the reusable content. These sub-template files are referenced with standard XSL commands in the primary template file. Sub-template functionality is available for use only with primary RTF and XSL templates.

Sub-templates are secondary RTF or XSL templates that are imported by primary RTF or XSL report templates. The primary template accesses the sub-template through the XSL import style sheet feature. You can import any XSL style sheets or other RTF or XSL templates using standard XSL import and call functions. PeopleTools simplified sub-template syntax is also supported.

Primary templates calling nonexistent or inactive sub-templates causes an error message to be issued indicating the reason for the problem. This error information is incorporated into Process Scheduler error handling as well as into online viewing or previewing of the report.

The sub-template files are independently stored and are not registered in association with a data source or primary template. This being the case, if any form fields exist inside the sub-template, the report in which the sub-template is placed must have a related data source that supplies those fields, or the data must be passed in as runtime parameters.

The Content Library is a component provided for the registration of reusable sub-template files. The metadata is similar to that of primary template files and includes the sub-template ID, sub-template description, language, object owner ID, report category, effective date, and status. As with Report Definition security, sub-template editor registration security is applied through report categories. Because Report Category secures the data in the component, you can assign select users read-only access for a report category. These users can browse, view, and download sub-template files but not add them. This facilitates the offline design of primary templates for users who can access the library of existing sub-templates but who can't alter them.

Sub-template names are not exposed to the end user at either report design time or runtime. The complete template (primary and sub-templates) is systematically assembled by the XML Publisher engine during report generation. The same occurs during online previewing as long as the sub-template file exists.

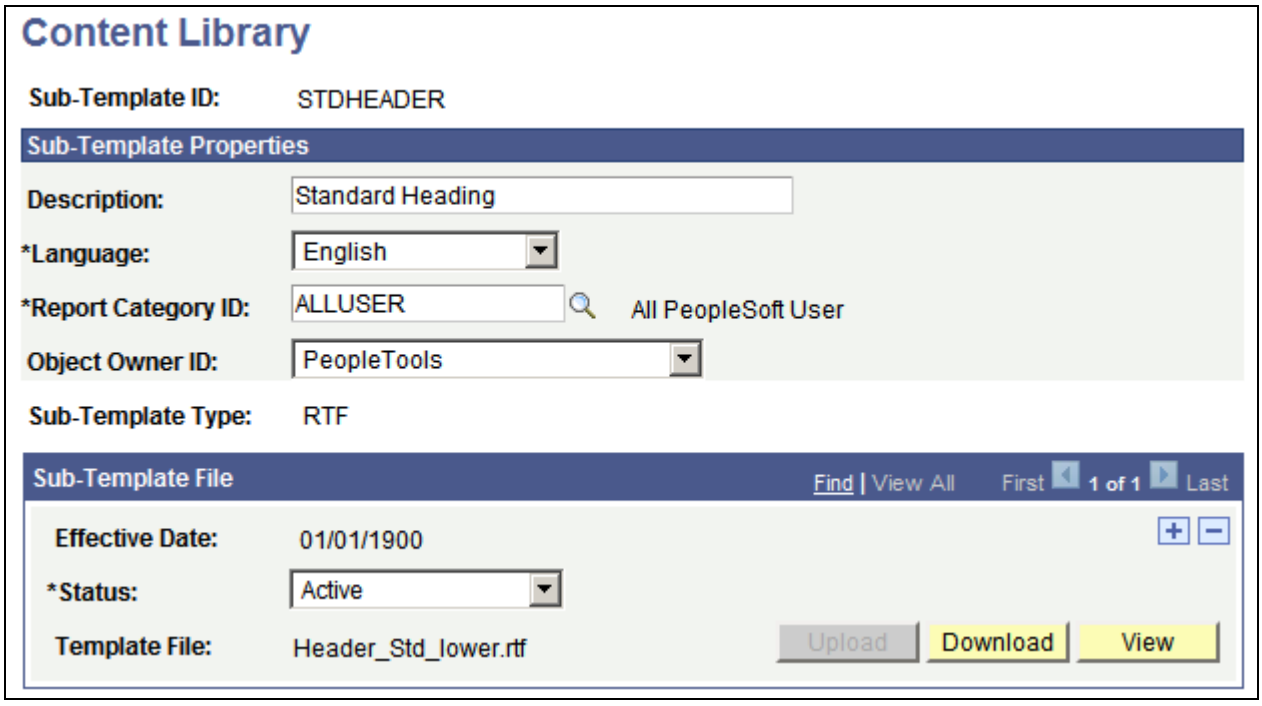
Note. No method is available for viewing which report templates include which sub-templates. This means that users must be careful about changing, deleting, or inactivating sub-templates.

Page Used to Maintain Sub-Templates

Page Name	*Definition Name*	*Navigation*	*Usage*
Content Library	PSXPSUBTMPLDEFN	Reporting Tools, XML Publisher, Content Library	Maintain sub-templates.

Maintaining Sub-Templates

Access the Content Library page (Select Reporting Tools, XML Publisher, Content Library.)



Content Library page

Sub-Template ID Enter a unique sub-template ID.

Description (Optional) Enter descriptive text that provides more detail about the sub-template and identifies its use.

Language Select a language code for the sub-template.

The default value reflects the users base language.

Report Category ID Select a report category ID.

This is a grouping mechanism that provides row-level security for editing sub-templates per the rights defined on the report category setup table.

See Chapter 2, "Setting Up XML Publisher," Setting Up Report Categories, page 11.

Object Owner ID (Optional) Indicate which product, feature, or application owns this sub-template.

Use this field to extract and package production data source and report registrations and their supporting files.

Sub-Template Type Select *RTF* or *XSL*.

Effective Date Select an effective date for the sub-template file in order to maintain new versions or versions that are specific to a particular time period. For example, a new file could be uploaded to reflect a new format or new legal language for reports, and the new sub-template is automatically used as of the new effective date.

The default date for a newly added sub-template file is the current system date. This effective date has no correlation with the effective date of the primary template. The as of date on the Query Report Viewer, Query Report Scheduler, or Run Control page determines which effective-dated templates and sub-templates are run.

Status Select a status of *In Progress*,*Active,* or *Inactive* for the sub-template file.

This field indicates the usability of the sub-template file. Runtime selection logic for a sub-template file uses this field in conjunction with the Effective Date field to determine which sub-template file to use at runtime. At least one file must be active to save a sub-template in the Content Library.

Template File Displays the name of the sub-template file.

Upload Click to attach an actual effective-dated sub-template file.

When you save the sub-template, this button becomes disabled. To reupload a new version of the sub-template, you must delete and add it again.

Download Click to download the sub-template to your local computer for updating.

View Click to view the contents of the sub-template.

Maintaining Template Translations

This section provides an overview of template translations and discusses how to:

- Search template translations.
- Maintain template translations.

Understanding Template Translations

The Template Translation component interacts with both report definition templates and Content Library sub-templates. Template translation files can be created only when a report's template type is RTF. Template Translation is a separate component with no row-level security, because the target user is different from the report developer, requestors, or viewers.

The Template Translation feature is based upon standard Localization Interchange File Format (XLIFF) .xlf file processing. Each report template or sub-template file can have related translation XLIFF files. These XLIFF files include translation units for each content element to be translated. The translatable units include all the fixed verbiage of the template excluding any values supplied by the data source. The Template Translations page includes an action button that generates a translatable file that must then be manually edited with the appropriately translated values. When the translation exercise is complete, the XLIFF file is uploaded and integrated into the XML Publisher translation system.

The Template Translation Search page provides advanced search capabilities to facilitate the location and management of template translations. Using this search page, you can determine whether a particular translation exists. The search can be focused by template or report, thus handling both Report Definition templates and Content Library sub-templates. You can also search based on target language.

Note. A template must exist before it can be translated.

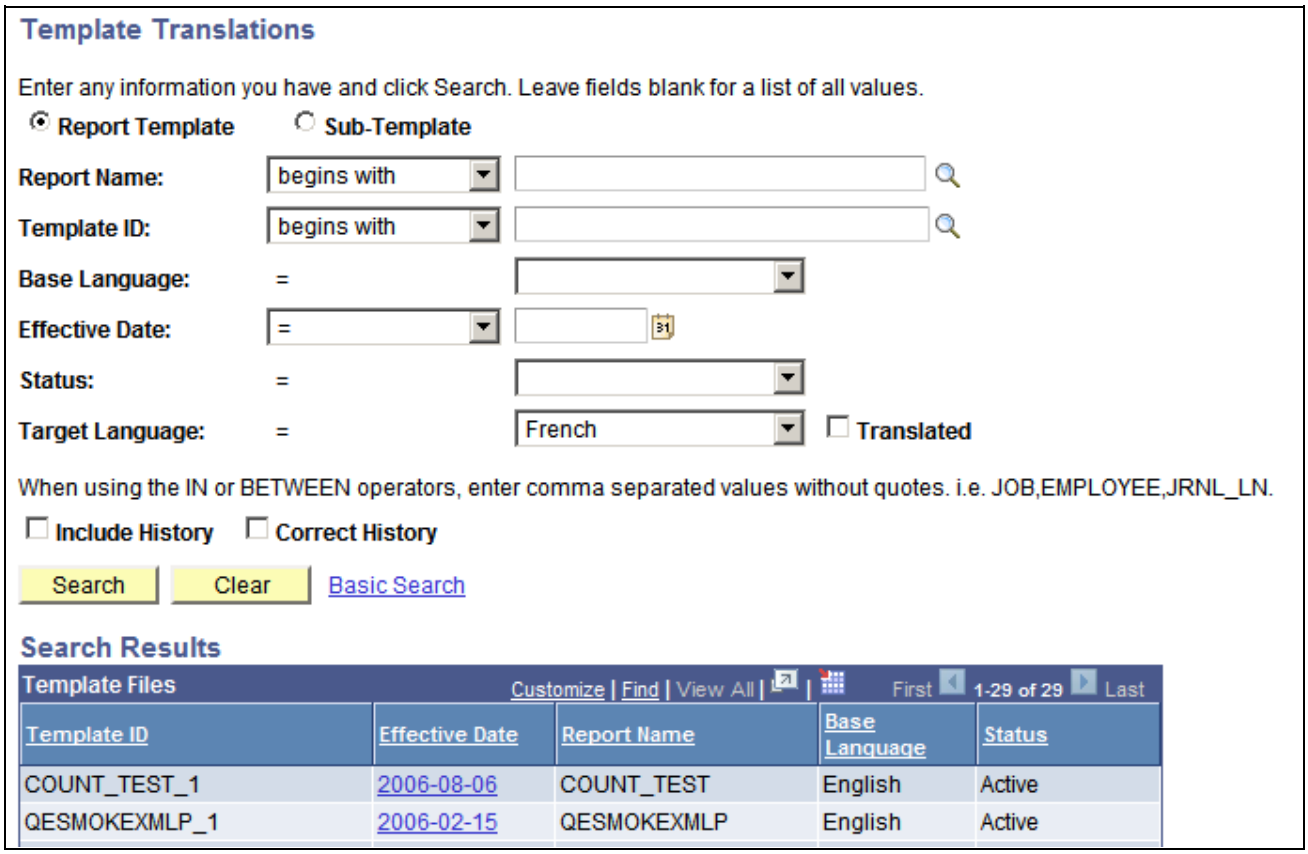
Template translations are not available for template types other than RTF. For a PDF report, multiple PDF templates must be registered to the report, one for each locale or language as required.

Pages Used to Maintain Template Translations

Page Name	*Definition Name*	*Navigation*	*Usage*
Template Translations Search	PSXPTMPLTRNSRCH	Reporting Tools, XML Publisher, Translations	Search template translations.
Template Translations	PSXPTMPLTRNS	Reporting Tools, XML Publisher, Translations Select the effective date of the template or sub-template for which you want to maintain translations.	Maintain template translations.

Searching Template Translations

Access the Template Translations Search page (Select Reporting Tools, XML Publisher, Translations.)



Template Translations Search page

To search for a template translation:

1. Select either the Report Template or Sub-template option, depending on whether you want to search the Report Definition templates or the Content Library sub-templates.

   The subsequent search prompts vary depending upon this choice. For example, the Report Name drop-down list box appears only if Report Template is selected.

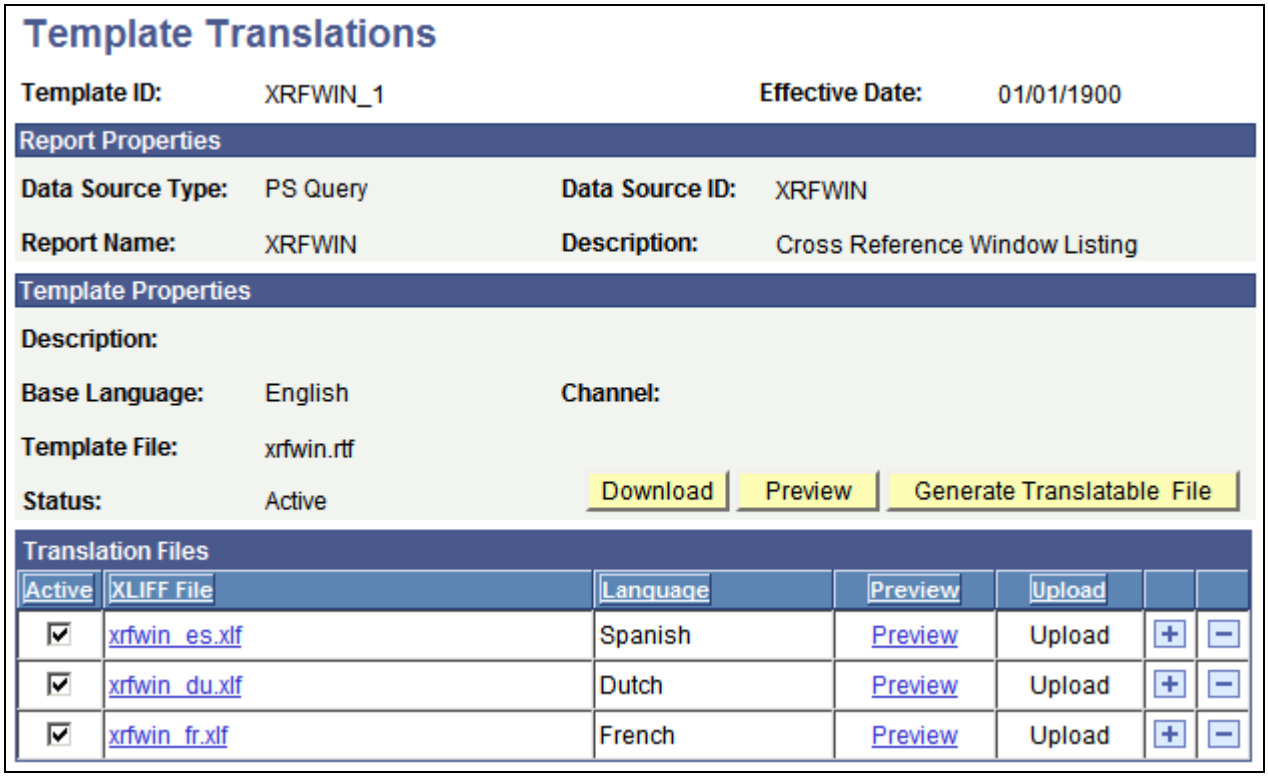
2. Select your search criteria and click the Search button.

   The Translated check box appears only when you have selected a value in the Target Language field. When selected, this check box enables you to search for templates that have already been translated into the selected target language. If this check box is cleared, you are searching for templates that have not yet been translated into the target language.

3. When your search results appear, select the effective date of the template for which you want to maintain translations.

Maintaining Template Translations

Access the Template Translations page (Select Reporting Tools, XML Publisher, Translations.)



Template Translations page

Template ID/Sub-Template ID Displays the unique template ID or sub-template ID.

Effective Date Displays the effective date as registered for the template under the Report Definition component or for the sub-template under the Content Library component.

Note. The translation inherits the same date and cannot be changed.

Report Properties

When the file to be translated is a report template, basic metadata about the report appears. This information does not appear when the file selected is a Content Library sub-template.

Data Source Type Displays the report's corresponding data source type of *PS Query*,*Rowset*, *XML Doc*, or *XML File*.

Data Source ID Displays the report's data source ID.

Report Name	Displays the report's name.
Description	Displays the report's description.

Template Properties/Sub-Template Properties

The Template Properties/Sub-Template Properties group box displays basic metadata about the base-language template file that has been selected for translation.

Description	Displays the template's description.
Base Language	Displays the base language of the template.
Channel	Displays the distribution channel for the template.
Template File	Displays the name of the template file.
Status	Displays a status of *In Progress,Active,* or *Inactive* for the template file.
Download	Click to open or save the base template file.
Preview	For report templates, click to preview the report template with sample data from the sample data file that was registered with the data source. The Preview button is not enabled when no sample data file is registered with the data source. For sub-templates, click the View button to view the sub-template file.
Generate Translatable File	Click to generate an .xlf file, which includes all translatable units extracted from the fixed text of the selected template or sub-template file. This file must be saved locally and then manually translated.

Translatable Files

The generated translatable XLIFF file includes the template's static headings and body text that require translation into another language. At the top of the file, the `<source-language>` tag indicates the base language value. You must update the `<target-language>` tag to the language that you are translating into. Initially the `<source-language>` and `<target-language>` values are the same. Prior to uploading the translated file into the database, you must edit the `<target-language>` tag to the translated language code. The value must be the two-character ISO language code.

For example, `fr` equals French, `jp` equals Japanese, and so on. The file won't load if the file type isn't .xlf or if the `<source-language>` equals the `<target-language>` and an error message appears.

In the `<body>` section of the file, each `<trans-unit id>` tag contains both a `<source>` tag and a `<target>` tag. The `<source>` tag contains the text in the base language. The corresponding `<target>` tag contains the translate fixed text.

No naming restriction is placed on XLIFF files; however, you should keep them close to the template file name and include the language. For example, for a French translation of the XRFWIN template, you could use XRFWIN_FR.xlf.

This code is an example of a translated XLIFF file:

```
  <?xml version="1.0" encoding="utf-8" ?>
- <xliff version="1.0">
 - <file source-language="en-US" target-language="fr-FR" datatype="XDO"
     original="orphen.rtf" product-version="orphen.xlf" product name="">
     <header />
   - <body>
     - <trans-unit id="" maxbytes="4000" maxwidth="15"
         size-unit="char" translate="yes">
         <source>Total</source>
         <target>Totale</target>
         <note>Text located: body/table</note>
       </trans-unit>
     - <trans-unit id="" maxbytes="4000" maxwidth="22"
         size-unit="char" translate="yes">
         <source>Seq Name/</source>
         <target>Nom de Seq/</target>
         <note>Text located: body/table/table header</note>
       </trans-unit>
```

Translation Files

You maintain the translated XLIFF files for your templates in the Translation Files grid .

Active

When it is uploaded, the translated template must be *Active* to make that language translation available at runtime.

The file is *Active* by default.

XLIFF File

Click the name of the uploaded translation file to open or save the file.

This action opens a new window that displays the file per the user's browser and OS settings and allows for updating and reloading the file.

Language

Displays the language into which the file was translated.

During the upload of the translated file, the system determines the language from the `<target-language>` tag and automatically updates the template translation metadata.

Preview

Select to display a translated version of the report in a new window.

This link is active only if the report's data source has a sample data file. No link is available for sub-templates, because no report context is available to preview.

Upload

Select to browse and upload the translation file.

Chapter 6

Running, Locating, and Viewing XML Publisher Reports

You can view and schedule query-based reports with XML Publisher. You can run custom reports as well as query-based reports batch through the Process Scheduler or online using PeopleCode APIs.

This chapter discusses how to:

- Run XML Publisher PeopleSoft Query reports.
- Run reports in Process Scheduler.
- Run reports using PeopleCode.
- Use Time zones in XML Publisher reports.
- Locate and view XML Publisher reports.

Running XML Publisher PeopleSoft Query Reports

You can view and schedule query-based reports with XML Publisher-delivered PeopleSoft Internet Architecture pages.

This section discusses how to:

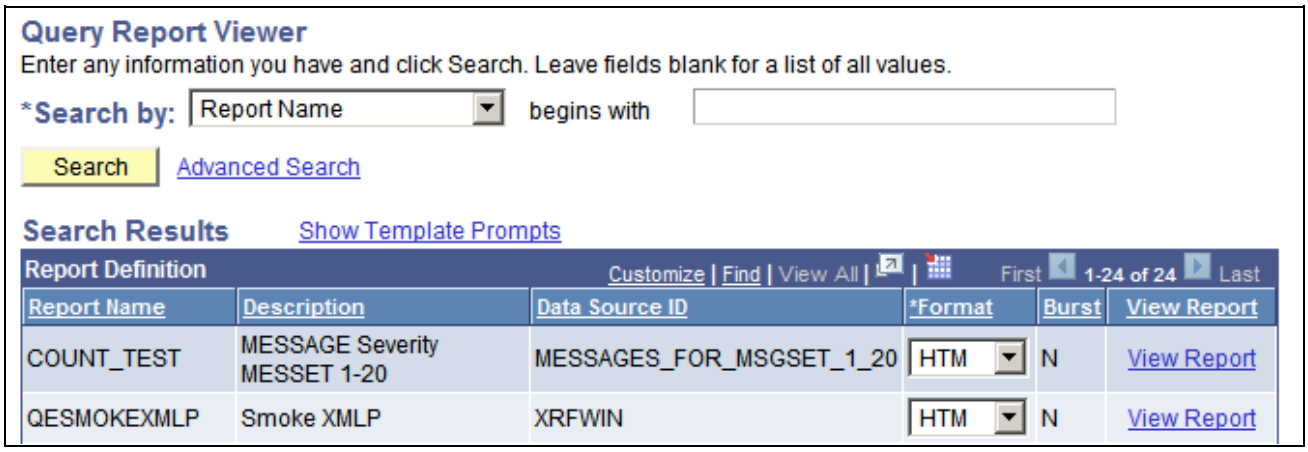
- Run reports in Query Report Viewer.
- Schedule reports in Query Report Scheduler.

Pages Used to Run XML Publisher PeopleSoft Query Reports

Page Name	*Definition Name*	*Navigation*	*Usage*
Query Report Viewer	PSXPQRYVIEWER	Reporting Tools, XML Publisher, Query Report Viewer	Run reports in Query Report Viewer.
Query Report Scheduler	PSXPQRYRUNCNTRL	Reporting Tools, XML Publisher, Query Report Scheduler	Select a run control ID and submit a process request to schedule query-based reports.

Running Reports in Query Report Viewer

Access the Query Report Viewer page (Select Reporting Tools, XML Publisher, Query Report Viewer.)



Query Report Viewer page

The Query Report Viewer allows selection and online viewing of those reports that have a data source type of PeopleSoft Query. Existing Query security applies so that each user has access to run only the reports to which he or she has qualified Query access to the data source.

Show Template Prompts Click to expand the Report Definition Search Results grid to include the template ID and as of date template prompts.

Report Name Displays the name of the report.

Description Displays the report's description.

Data Source ID Displays the report's data source ID.

Template ID Select from the templates associated with the report definition.

As Of Date Select the as of date for the template version that you want to view.

Format Select from the output format choices associated with the report definition.

Burst Indicates whether the report definition includes bursting instructions.

Note. Bursted reports are listed, but you can't run them from the Query Report Viewer component. Bursted reports must be run from the Query Report Scheduler component.

View Report

Click to view the report online. When Query runtime parameters exist, the parameters are displayed.

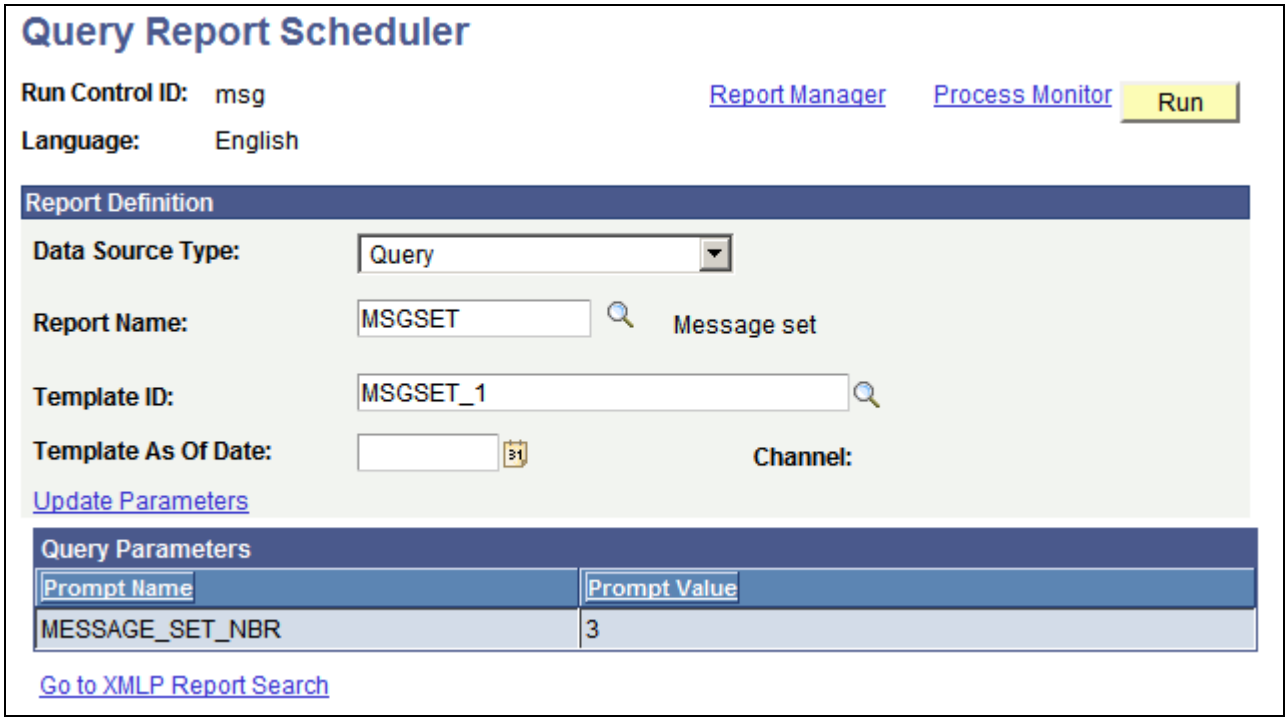
A new window opens displaying the report results according to the runtime inputs. You can save the report results locally by using the browser's Save functionality.

Note. This link is disabled for bursted reports.

Note. The data and template translation language choice is automatically supplied to the user's session language.

Scheduling Reports in Query Report Scheduler

Access the Query Report Scheduler page (Select Reporting Tools, XML Publisher, Query Report Scheduler.)



Query Report Scheduler page

Query Report Scheduler uses the existing Process Scheduler functionality to:

- Select runtime parameters for query-based and connected query-based reports.
- Monitor the report process request.
- Post and secure the results to either the Report Manager, a printer, or the Process Scheduler file directory.

Note. If a query is run through Reporting Tools, Query, Schedule Query, the XML Publisher-related prompts do not appear. Only the basic table-formatted query results are generated.

Run Control ID	Enter a run control ID.
Language	Indicates the language of the run control. The report selects data and template translations based upon the language code of the run control. The user sets this value on the My System Profile, General Profile Information page in the My Preferred Language for Reports and Email field. The language appears in the Query Report Scheduler Search Results so that you are informed of the language selection criteria.
Data Source Type	Select either *Query* or *Connected Query*.
Report Name	Select the name of the Query or Connected Query-based report that you want to schedule. The drop-down list box values are based on previously registered report definitions. Existing Query security applies so that each user has access to run only reports to which they have Query access.
Burst Field Name	Displays the value set in the Burst by field of the Report Definition, Bursting page. This field appears for bursted reports only.
Dynamic Template	Displays either *Active* or *Inactive*, depending on whether criteria exists to dynamically select the template, language, or both based upon a data value that is set in the Template group box of the Report Definition, Bursting page. This field appears for bursted reports only.
Bursting Security	Displays either *Active* or *Inactive*, depending on whether criteria exists to assign unique bursting security that is set in the Burst Security group box of the Report Definition, Bursting page. This field appears for bursted reports only.
Template ID	Select from the templates associated with the report definition.
Template As Of Date	(Optional) Select the as of date for the template version to use for the report. The system looks at the report definition for templates that are active as of this date.
Channel	Indicates the distribution channel of the template.
Update Parameters	Click to update the Query or Connected Query runtime prompt values. After runtime values are entered, they are saved with the run control ID.

Report Manager	Click to go to the Report Manager to check the progress of your process request and to view the report content immediately after the output file is posted. The final output file is posted to the Report Manager repository for web access by authorized users. See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Report Manager," Viewing Reports.
Process Monitor	Click to go to the Process Monitor to check the progress of your request. See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Process Monitor."
Run	Click to access the Process Scheduler Request page. The Process Scheduler Request page enables you to specify variables, such as where a process runs and in what format the process output is generated. The values for output type and output format appear by default per the report definition and can be changed if the report definition allows it. Distribution options are also active, allowing updates to viewers, only as allowed in the report definition.
Go to XMLP Report Search	Click to access the XMLP Report Search page. See Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Searching the XML Publisher Report Repository, page 79.

See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Submitting and Scheduling Process Requests."

Running Reports in Process Scheduler

This section discusses how to:

- Use the Process Scheduler Request page.
- Create the run control page.
- Create a process definition.
- Monitor requests.

Using the Process Scheduler Request Page

At runtime, the Process Schedule Request page appears after you click the Run button on the run control page. This page includes operator-selectable choices of output type and output format. Output type choices reflect the location values from the report definition. Output format choices reflect the output format values from the report definition.

Because values for output location, output format, and viewer security are associated with each report definition, these values should be passed to the Process Schedule Request page. These values are passed automatically only when the process definition type is *XML Publisher*.

Note. In order to execute XML Publisher reports with DB2 z/OS databases you will need to configure a Process Scheduler on a PeopleTools certified Windows or other UNIX batch server. The PeopleTools Process Scheduler on z/OS cannot execute XML Publisher reports.

Creating the Run Control Page

You need to create a custom run page that contains the prompts required by your report definition.

Your run control page should be a combination of the PeopleSoft PeopleTools-supplied run control subpage and the application-specific section for runtime parameters for the data extraction program. It should include report, template name, language, and as of date. Depending upon your application design, these values could be systematically deduced from user preferences, come from program defaults, or come from the operator input selection.

See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Submitting and Scheduling Process Requests," Running Processes from PeopleSoft Applications.

Creating a Process Definition

A process type of *XML Publisher* is delivered as system data and is available on the PeopleTools, Process Scheduler, Process Definition page. When application processes are defined and assigned the *XML Publisher* process type, entries for output type and format on the runtime Process Scheduler Request page reflect the definitional metadata under the XML Publisher report definition.

The XML Publisher report name, Process Scheduler process name, and the Application Engine process name are the same.

See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Defining PeopleSoft Process Scheduler Support Information," Defining Process Definitions.

Monitoring Requests

The Process Scheduler processes XML Publisher-based reports. You can define multiple related activities as separate processes. For example, generation of the XML data, the XML Publisher merging of that data with the template and creating the final output, and the subsequent postprocessing to send related emails. Each process appears separately in the Process Monitor. Error messages indicate whether the problem is on the data extraction or the XML Publisher portion of the report request.

Running Reports Using PeopleCode

This section provides an overview of PeopleCode XML Publisher classes and discusses how to:

- Run reports using PeopleCode.

- Choose a template.
- Pass parameters.
- Use time zones in XML Publisher reports.
- Burst reports.
- Customize printed report output.
- Distribute reports.
- Search for reports.

Understanding PeopleCode XML Publisher Classes

All report runtime functionality is available using the PeopleCode XML Publisher classes. The classes and methods that you use to define custom reports respect the report category security assigned to the report definition. Users with read-only access to report definitions cannot edit them.

Runtime classes are available to call and pass in XML data and a choice of report template to the XML Publisher core engine to generate the output in a desired format. For online viewing, a function is available to pass the output back to the browser. Processing a report through the Process Scheduler posts XML Publisher output entries to the web, the Report Manager, or both according to the existing processes. When processes are categorized under the XML Publisher process type, the capability to establish output destination, format, and authorized viewer choices from the related report definition is enhanced. A search method is also available for accessing reports in the report repository.

You can call the PeopleCode from a page for online processing, or you can create an application engine program to run the report in batch.

See Chapter 5, "Defining Report Definitions," Creating Report Definitions, page 39.

See *Enterprise PeopleTools 8.51 PeopleBook: Application Engine*, "Creating Application Engine Programs."

See *PeopleTools 8.51 PeopleBook: PeopleCode API Reference*, "XML Publisher Classes."

Running Reports Using PeopleCode

The XML Publisher classes enable you to access the runtime portions of the XML publishing process programmatically, that is, after the templates and reports have been created. The XML Publisher classes are not built-in classes, like rowset, field, record, and so on. They are application classes. Before you can use these classes in your PeopleCode program, you must import them to your program.

Your import statements should look like this:

```
import PSXP_RPTDEFNMANAGER:*;
```

See *PeopleTools 8.51 PeopleBook: PeopleCode API Reference*, "XML Publisher Classes," ReportDefn Class Constructor.

Example: Publish a Report Based on PS Query

This is a code snippet example for publishing a report based on PS Query:

```
import PSXP_RPTDEFNMANAGER:*;

/* get report definition object */
&oRptDefn = create PSXP_RPTDEFNMANAGER:ReportDefn (&sRptDefn);
&oRptDefn.Get();
/* fill query runtime prompt record */
&rcdQryPrompts = &oRptDefn.GetPSQueryPromptRecord();
If Not &rcdQryPrompts = Null Then
  &oRptDefn.SetPSQueryPromptRecord(&rcdQryPrompts);
End-If;
/*generate report*/
&oRptDefn.ProcessReport (&sTmpltID, &sLangCd, &AsOfDate, &sOutFormat);
/*publish report */
&oRptDefn.Publish(&sPrcsServerName,"",&sFolder, &prcsInstId);
```

Example: Print a Report Based on XMLFile

This is a code snippet example for printing a report based on XMLFile:

```
import PSXP_RPTDEFNMANAGER:*;

/* get report definition object */
&oRptDefn = create PSXP_RPTDEFNMANAGER:ReportDefn (&sRptDefn);
&oRptDefn.Get();
/* pass XMLFile to the report definition */
&oRptDefn.SetRuntimeDataXMLFile(&sXMLFilePath);
/*generate report*/
&oRptDefn.ProcessReport (&sTmpltID, &sLangCd, &AsOfDate, &sOutFormat);
/*print report */
&oRptDefn.PrintOutput(&sDestinationPath);
```

Choosing a Template

Because report definition information is available from the PeopleCode XML Publisher classes, you can incorporate prompts on runtime pages to select reports and templates. You must pass in XML data and a choice of report template to the XML Publisher core engine to generate the output in a desired format.

You can retrieve a particular template file or expose a choice of templates at runtime. Only active reports and templates are retrieved. An as of date is also required to coordinate with the template file's effective date. If not supplied, the as of date is assumed to be the system date. At runtime, the template as of date is used to select the appropriate active effective-dated template and sub-template that is current as of that date.

A PeopleCode class is available to retrieve a report's template IDs based on a channel value, although it is not exposed on a PeopleSoft PeopleTools-delivered Pure Internet Architecture page. You can also incorporate template administration functionality directly into your application pages. This functionality includes creating the definitions and storing the related files, as well as querying to find the templates associated with a report definition.

Passing Parameters

The system may need to pass runtime parameters into the XML Publisher core engine. Numbers and text are sent as strings with single quotes. By default, PeopleTools(through the RepoftDefn class ProcessReport method) always passes the following parameters/tags:


```
<?$ReportID?>
<?$ReportTitle?>
<?$RunDate?>
<?$RunTime?>
```

These tags can be included in the template layout wherever they are needed, they are especially useful for report headers. Before inserting these parameters into the template (or sub-template), the following declarations must be entered under a form field at the top of the report's primary template; one for each parameter called:

```
<xsl:param name="ReportID" xdofo:ctx="begin"/>
<xsl:param name="ReportTitle" xdofo:ctx="begin"/>
<xsl:param name="RunDate" xdofo:ctx="begin"/>
<xsl:param name="RunTime" xdofo:ctx="begin"/>
```

These tags can be included in the template layout wherever they are needed. These parameters are especially useful for report headers.

The PeopleSoft-delivered XML Publisher report XRFWIN demonstrates the usage of these values in a report calling a sub-template for a header.

For the standard parameter passage of report ID and report description, the translation of report descriptions may become important for report headers. XML Publisher includes PeopleSoft-related language tables for the data source, and report and template tables that support the report's data language values for the description fields.

Bursting Reports

The ReportDefn class ProcessReport method has code built in to process a single report request to create multiple output files per the bursting instructions defined on the report definition. Bursting always occurs at runtime if a burst value is stored in the report definition's burst field value.

See Chapter 5, "Defining Report Definitions," Setting Bursting Options, page 53.

Customizing Printed Report Output

The PeopleSoft application supports batch printing XML Publisher reports directly from a server using PDF output format. Printers with Postscript level 3 interpreter natively support printing PDF format. You can also convert the generated PDF report files to conventional formats supported by other printers, such as Postscript or PCL, by using an external software program. The PeopleSoft application provides PeopleCode support for sending PDF files directly to a specified printer, and it also provides customization capability for inserting conversion logic from PDF to different printer formats.

To send an XML Publisher report to a printer, use the PrintOutput method after the ProcessReport method as shown in this example:

```
&MyReportDefn.ProcessReport("myTemplate", "", %Date, "PDF");
&MyReportDefn.PrintOutput(&PrinterPath);
```

If you want to insert conversion logic from PDF to a different printer format before an output file is sent to a printer, create a batch file named `psxprint.bat` on Microsoft Windows or `psxprint.sh` on Unix under the Process Scheduler server home directory %PS_HOME%\appserv\prcs\%domain_name% and write a call to an external conversion program in this batch file.

In the batch file, you can use the following variables, which the `ReportDefn.PrintOutput()` method replaces with actual data at report runtime:

Variable	*Description*
`%RPTOUTDIR%`	Full path to the report output directory.
`%REPORTFILE%`	Full path to the report output file.
`%DESTPRINTER%`	Full path to the destination printer.

See Chapter 5, "Defining Report Definitions," Setting Output Options, page 48.

Distributing Reports

PeopleCode options are available for posting your generated report to a file server, printing it, or publishing it to the Report Manager with appropriate security.

For online viewing, a method is available for passing the output back to the browser. No report results are persisted, but the user viewing the results can use the browser's Save As feature to retain the report file locally.

When the output type is *Printer*, the output format is limited to PDF. A printer location must be specified, and the printer must be capable of printing PDF output. If the output file is large, adequate memory must be available on the print server.

Distribute To IDs are those defined in the Report Definition, Security page. Distribution functionality within the Process Scheduler is enhanced to assign values systematically per the XML Publisher report definition. The Report Definition, Security page provides choices for selecting a Report Manager folder as well as the ability to assign viewing rights to additional roles or user IDs at runtime if allowed by the report definition.

When the report definition has the Allow viewer ID assignment at report runtime check box selected, the report requestor can add or delete IDs. If no viewers are assigned, by default the requestor's ID is added systematically.

Searching for Reports

A search method is available for accessing reports in the Report Manager repository. The PeopleCode uses additional search keys based on the report definition's additional metadata.

See Chapter 5, "Defining Report Definitions," Setting Bursting Options, page 53 and Chapter 6, "Running, Locating, and Viewing XML Publisher Reports," Searching the XML Publisher Report Repository, page 79.

Using Time Zones in XML Publisher Reports

When displaying datetime values, XML Publisher takes into consideration the time zone of the user running the report. The time zone is retrieved from the user's Personalization settings (My Personalizations, Regional Settings, Local Time Zone). Personalized time zone display is dependent on the following conditions:

- The report template must be either *RTF* or *XSL*.
- The datetime element in the XML file must include the UTC offset, for example, 2008-07-28T09:00:00-0700.

  Note. A Query data source includes the offset for datetime fields.

- The time zone must have a valid 3 character time zone code, for example *PST* or *EST*. If custom time zones have been implemented in your environment, XML Publisher does not recognize these custom time zones and will display the time in UTC.
- The datetime field in the template should be formatted using an Oracle abstract format mask that displays the time zone. For example:

  ```
  <?format-date:STARTDATETIME;'SHORT_TIME_TZ'?>
  ```

  Note. Oracle abstract format masks are listed in *Oracle Business Intelligence Publisher Users Guide*, "Creating an RTF Template," Using Oracle Abstract Format Masks.

Note. XML Publisher uses Java Time Zones based on the JRE running under the PeopleSoft application server. In some releases of JRE, a known issue exists in that Daylight Savings Time is calculated incorrectly for EST, MST, and HST timezones. This problem has been documented by Sun and a solution is available on their website: http://java.sun.com/developer/technicalArticles/Intl/alert.html

Locating and Viewing XML Publisher Reports

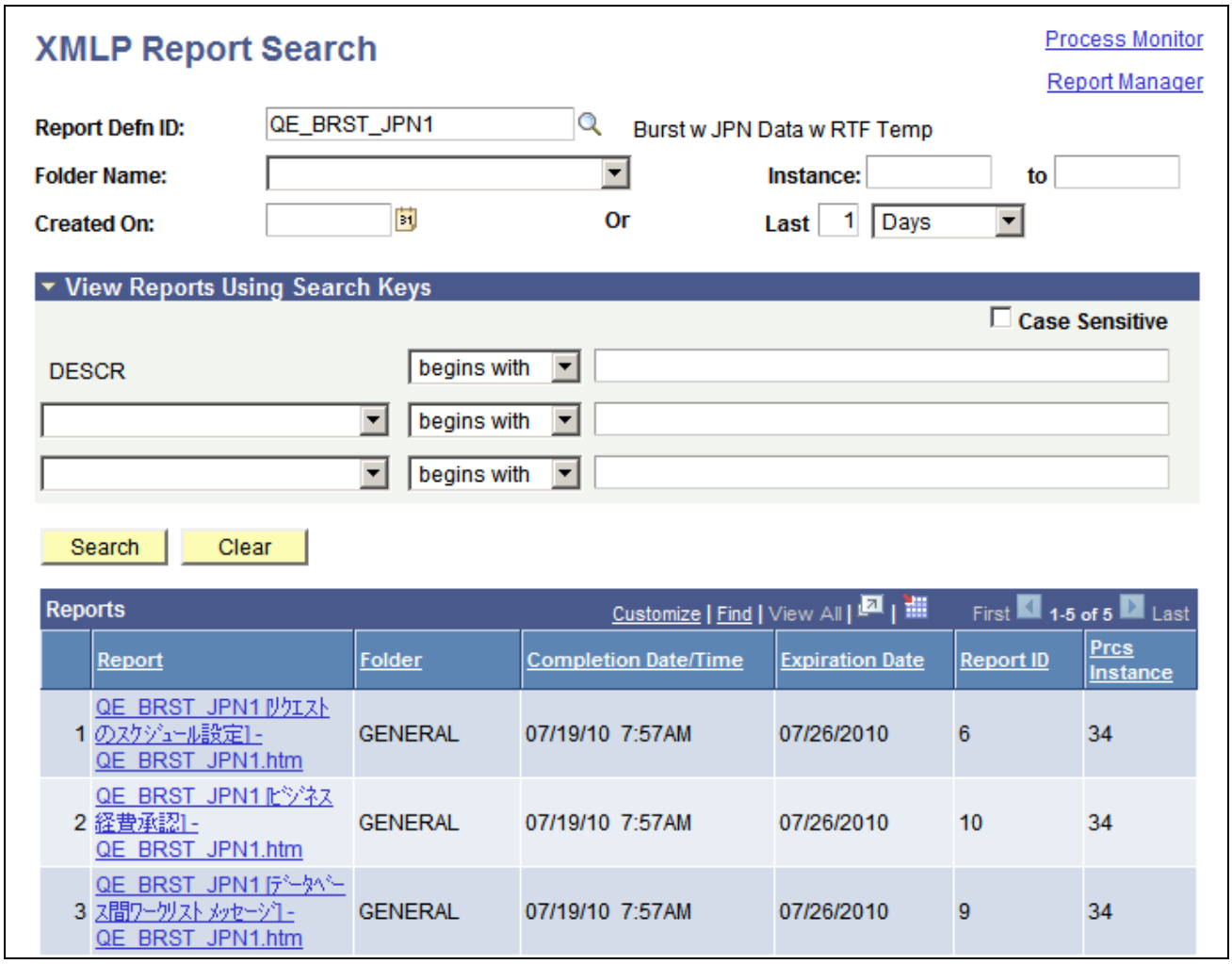
This section discusses how to search the XML Publisher report repository.

Pages Used to Locate and View XML Publisher Reports

Page Name	*Definition Name*	*Navigation*	*Usage*
XMLP Report Search	PSXPRPTMGR	Reporting Tools, XML Publisher, XMLP Report Search	Search the XML Publisher Report Repository.

Searching the XML Publisher Report Repository

Access the XML Report Search page (Select Reporting Tools, XML Publisher, XMLP Report Search.)



XMLP Report Search page

Enter criteria to filter the reports to list. XML Publisher Report Search ignores criteria for fields that are blank.

Report Definition ID (Optional) Select the name of the report definition to search on.

Folder Name (Optional) Select a specific folder to list only the reports that are contained in that folder.

Created On (Optional) Use the calendar, or enter a specific date to list only reports that are created on that date.

Instance and **to** (Optional) Enter a range of process instances. Leave the to field blank to list all instances after the number that you enter in the Instance field.

Last (Optional) Use to display only those reports that were created in the last number of days, hours, or minutes. For example, to list only those reports that were created within the last two hours, enter *2* and select *Hours*.

Report Manager	Click to go to the Report Manager. See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Report Manager."
Process Monitor	Click to go to the Process Monitor. See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Process Monitor."

See *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Using Report Manager," Viewing Reports.

Viewing Reports Using Additional Search Keys

Users can also search by the following criteria for bursted reports:

- A specific value in the Burst By field.

  This is a read-only field that appears automatically when a value is set in the Burst by field of the Report Definition, Bursting page.

- Up to two additional values in the predefined bursting Search Key fields.

  These drop-down list boxes display the values set in the Search Keys region of the Report Definition, Bursting page.

To view bursted reports using the additional search keys:

1. Select the Case Sensitive check box to perform a case-sensitive search.
2. For the Burst By field, select a search operator.
3. Enter a value to search on.
4. For the additional Search Key fields, select the search field name, search operator and search value.

See Chapter 5, "Defining Report Definitions," Setting Bursting Options, page 53.

Appendix A

Securing XML Publisher

This appendix discusses XML Publisher security.

XML Publisher Security

XML Publisher security can be separated into three categories:

- Defining reports.
- Running reports.
- Viewing reports.

When you are defining Query-based reports, Query security determines which queries you can access and select from to create your XML Publisher report definitions. Security for editing and viewing report definitions is controlled by the Report Category ID attribute, which is set on the Reporting Tools, XML Publisher, Setup, Report Category page.

Security for running and viewing XML Publisher reports is controlled by setting options in a number of places. This table illustrates where security can be set:

Activity	***Security Settings***	***Query-based reports (Non-Bursted)***	***Query-based reports (Bursted)***	***Non-Query-based reports (Non-Bursted)***	***Non-Query-based reports (Bursted)***
Running Reports	Query Security	X	X	NA	NA
Running Reports	Application Security	X	X	X	X
Running Reports	Process Scheduler Security	X	X	X	X
Viewing Report Definitions	Report Definition, Security page	X	X	X	X
Viewing Report Definitions	Report Definition, Bursting page	NA	X	NA	X

Application security and Process Scheduler security determine who can run reports. XML Publisher does not provide additional security beyond what Oracle currently provides. That means that the component security of the data extraction program drives access control to the associated reports. For processes, process security prevails and for queries, query security prevails. When you are running a Query-based report, the requester's row-level security to the underlying data source always applies.

Query-based reports viewed online in real time from the Query Report Viewer respect query access groups for the user's primary permission list. For non-Query-based reports viewed online in real time, security is controlled by the application.

When you are viewing a report that was run through either the Query Report Scheduler or the Process Scheduler, security is controlled by both the Distribution ID field on the Report Definition, Security page and, when the Allow viewer ID assignment at report runtime check box is selected, by those IDs selected at runtime on the Process Scheduler Request, Distribution Detail page. Additional viewing security can also be defined for bursted reports on the Report Definition, Bursting page.

If no viewers are designated on the Report Definition, Security page and the Allow viewer ID assignment at report runtime is selected, then the report requestor's ID is applied as a viewer by default at runtime. This applies to bursted reports as well..

See Chapter 2, "Setting Up XML Publisher," Setting Up Report Categories, page 11; Chapter 5, "Defining Report Definitions," Creating Report Definitions, page 39; *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Submitting and Scheduling Process Requests," Scheduling Process Requests; *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Process Scheduler*, "Setting Up PeopleSoft Process Scheduler Security" and *Enterprise PeopleTools 8.51 PeopleBook: PeopleSoft Query*, "PeopleSoft Query Security."

Appendix B

Migrating XMLP Definitions

This appendix provides an overview of XMLP definitions and discusses how to:

- Migrate XMLP definitions
- Migrate XML Publisher translated languages
- Clean Up XML Publisher Metadata

XMLP Definitions Overview

To facilitate the movement of reports and templates from development to test and then to production, XML Publisher (XMLP) objects are available as managed objects that can be placed into projects for migration from database to database. To facilitate the location of report-related objects, these items can be identified based on object owner ID.

Migrating XMLP Definitions

The following definition types can be added to projects in Application Designer:

- XMLP Data Src Defn.
- XMLP File Defn.
- XMLP Report Defn.
- XMLP Template Defn.

If the data source for the XMLP report is PS Query or Connected Query, then the query or connected query definition should also be included in the project.

Note. Because XML Publisher is based on managed objects, all updates to your data need to be performed using the PeopleSoft XML Publisher Pure Internet Architecture pages, PeopleSoft Application Designer, or XML Publisher PeopleCode APIs.

Note. As with other PeopleTools-delivered features, XML Publisher for PeopleSoft Enterprise uses managed object functionality, and you cannot use the switching feature of multilanguage data entry. You can populate Related Language tables by signing in and establishing a different session language. Then you can populate the Related Language table for that session.

See *PeopleTools 8.51 PeopleBook: Global Technology*, "Working With PeopleSoft Applications in Multiple Languages" and *PeopleTools 8.51 PeopleBook: PeopleTools Portal Technologies*, "Understanding PeopleSoft Pure Internet Architecture."

Migrating XML Publisher-Translated Languages

XMLP template translation uses related XLIFF files (one for each language) that contain not only specific translation pairs but the whole template definition. This is a standard for using XLIFF translation methodology.

Because the translation is tied to the template definition, the translation file (XMLP Template Defn) as well as the specific XLIFF files (XMLP File Defn) should be included in the project.

Note. If the template file is not copied with the language files, the correct translation file cannot be used when you run the report. Run a SYSAUDIT for Audit XML Publisher Integrity and delete any orphaned definitions.

See *PeopleTools 8.51 PeopleBook: Data Management*, "Ensuring Data Integrity," XML Publisher for PeopleSoft Integrity.

Cleaning Up XML Publisher Metadata

To ensure the integrity of the XML Publisher files, run the application engine program PSXPCLEAN.

This application engine program finds:

- Unreferenced objects in PSFILEDEFN.
- Template definitions and template translations for which file objects are missing.
- Inconsistencies between PSFILEDEFN and PSFILEDATA.

This application engine program is delivered in *Report and Delete* mode. To run the program in *Report Only* mode, open the application engine program PSXPCLEAN in Application Designer and remove the comment in the following statement in PSXPCLEAN:Main:Start PeopleCode action:

```
rem PSXPFILECLN_AET.REPORT_ONLY_FLAG = "Y"
```

You can schedule and run PSXPCLEAN using the PeopleTools, Process Scheduler, System Process Request page. You should run this program on a regular basis to keep template metadata consistent.

Index

S

T

X