23

Use of XLink and XInclude with Oracle XML DB

You can use XLink and XInclude with resources in Oracle XML DB Repository. But the use of XLink is *deprecated*.

Note:

The Oracle XML DB Repository is deprecated with Oracle Database 23ai.

- Overview of XLink and XInclude
  A document-oriented, or **content-management**, application often tracks relationships, between documents, and those relationships are often represented and manipulated as links of various kinds. Such links can affect application behavior in various ways, including affecting the document content and the response to user operations such as mouse clicks.
- Link Types for XLink and XInclude
  XLink and XInclude link types are described, as well as the relation between these and Oracle XML DB Repository links. XLink links (*deprecated*) are more general than repository links. XLink links can be simple or extended. Oracle XML DB supports only simple, not extended, XLink links.
- XInclude: Compound Documents
  XInclude is the W3C recommendation for the syntax and processing model for merging the infosets of multiple XML documents into a single infoset. Element `xi:include` is used to include another document, specifying its URI as the value of an `href` attribute.
- Oracle XML DB Support for XLink
  You can configure Oracle XML DB Repository resources so that XLink links are ignored, or so that they are mapped to Oracle XML DB document links. However, the use of XLink with Oracle XML DB Repository is *deprecated*.
- Oracle XML DB Support for XInclude
  Oracle XML DB supports XInclude 1.0 as the standard mechanism for managing compound documents. It does not support attribute `xpointer` and the inclusion of document fragments, however. Only complete documents can be included (using attribute `href`).
- Use View DOCUMENT_LINKS to Examine XLink and XInclude Links
  You can query the read-only public view `DOCUMENT_LINKS` to obtain system information about document links derived from both XLink (*deprecated*) and XInclude links. The information in this view includes the following columns, for each link:
- Configuration of Repository Resources for XLink and XInclude
  The resource configuration file that you use as a resource to configure XLink (*deprecated*) and XInclude processing for other resources is described.
- Manage XLink and XInclude Links Using DBMS_XDB_REPOS.processLinks
  You can use PL/SQL procedure `DBMS_XDB_REPOS.processLinks` to manually process all XLink (*deprecated*) and XInclude links in a single document or in all documents of a folder.

Overview of XLink and XInclude

A document-oriented, or **content-management**, application often tracks relationships, between documents, and those relationships are often represented and manipulated as links of various kinds. Such links can affect application behavior in various ways, including affecting the document content and the response to user operations such as mouse clicks.

W3C has two recommendations that are pertinent in this context, for documents that are managed in XML repositories:

- **XLink –** Defines various types of links between resources. These links can model arbitrary relationships between documents. Those documents can reside inside or outside the repository.
- **XInclude –** Defines ways to include the content of multiple XML documents or fragments in a single infoset. This provides for compound documents, which model inclusion relationships. **Compound documents** are documents that contain other documents. More precisely, they are file resources that include documents or document fragments. The included objects can be file resources in the same repository or documents or fragments outside the repository.

Each of these standards is very general, and it is not limited to modeling relationships between XML documents. There is no requirement that the documents linked using XLink or included in an XML document using XInclude be XML documents.

Using XLink and XInclude to represent document relationships provides flexibility for applications, facilitates reuse of component documents, and enables their fine-grained manipulation (access control, versioning, metadata, and so on). Whereas using XML data structure (an ancestor–descendents hierarchy) to model relationships requires those relationships to be relatively fixed, using XLink and XInclude to model relationships can easily allow for change in those relationships.

Note:

For XML schema-based documents to be able to use XLink and XInclude attributes, the XML schema must either explicitly declare those attributes or allow any attributes.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

See Also:

- XML Linking Language (XLink) Version 1.0
- XML Inclusions (XInclude) Version 1.0 (Second Edition)

Link Types for XLink and XInclude

XLink and XInclude link types are described, as well as the relation between these and Oracle XML DB Repository links. XLink links (*deprecated*) are more general than repository links. XLink links can be simple or extended. Oracle XML DB supports only simple, not extended, XLink links.

- XLink and XInclude Links Model Document Relationships
  XLink (*deprecated*) and XInclude links model arbitrary relationships among documents. The meaning and behavior of a relationship are determined by the applications that use the link. They are not inherent in the link itself. XLink and XInclude links can be mapped to Oracle XML DB document links.
- XLink Link Types and XInclude Link Types
  XLink (*deprecated*) and XInclude can provide links to other documents. In the case of XInclude, attributes `href` and `xpointer` are used to specify the target document.

XLink and XInclude Links Model Document Relationships

XLink (*deprecated*) and XInclude links model arbitrary relationships among documents. The meaning and behavior of a relationship are determined by the applications that use the link. They are not inherent in the link itself. XLink and XInclude links can be mapped to Oracle XML DB document links.

When document links target Oracle XML DB Repository resources, they can (according to a configuration option) be hard or weak links. In this, they are similar to repository links in that context. Repository links can be navigated using file system-related protocols such as FTP and HTTP. Document links cannot, but they can be navigated using the XPath 2.0 function `fn:doc`.

Related Topics

- Hard Links and Weak Links
  Links that target repository resources can be hard or weak. Hard and weak links have different dependencies with respect to the resources that they target. Hard links cannot target ancestor folders; weak links can. You can query the repository path view, `PATH_VIEW`, to determine the type of a repository link.

XLink Link Types and XInclude Link Types

XLink (*deprecated*) and XInclude can provide links to other documents. In the case of XInclude, attributes `href` and `xpointer` are used to specify the target document.

Xlink links can be simple or extended. **Simple** links are unidirectional, from a source to a target. **Extended** links (sometimes called **complex**) can model relationships between multiple documents, with different directionalities. Both simple and extended links can include link metadata. XLink links are represented in XML data using various attributes of the namespace `http://www.w3.org/1999/xlink`, which has the predefined prefix `xlink`. Simple links are represented in XML data using attribute `type` with value `simple`, that is, `xlink:type = "simple"`. Extended Xlink links are represented using `xlink:type = "extended"`.

Third-party extended Xlink links are not contained in any of the documents whose relationships they model. Third-party links can thus be used to relate documents, such as binary files, that, themselves, have no way of representing a link.

The source end of a simple Xlink link (that is, the document containing the link) must be an XML document. The target end of a simple link can be any document. There are no such restrictions for extended links. Example 23-3 shows examples of simple links. The link targets are represented using attribute `xlink:href`.

> **Note:**
>
> The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

XInclude: Compound Documents

XInclude is the W3C recommendation for the syntax and processing model for merging the infosets of multiple XML documents into a single infoset. Element `xi:include` is used to include another document, specifying its URI as the value of an `href` attribute.

Element `xi:include` can be nested, so that an included document can itself include other documents.

(However, an inclusion cycle raises an error in Oracle XML DB. The resources are created, but an error is raised when the inclusions are expanded.)

XInclude thus provides for *compound* documents: repository file resources that include other XML documents or fragments. The included objects can be file resources in the same repository or documents or fragments outside the repository.

A book might be an example of a typical compound document, as managed by a content-management system. Each book includes chapter documents, which can each be managed as separate objects, with their own URLs. A chapter document can have its own metadata and access control, and it can be versioned. A book can include (reference) a specific version of a chapter document. The same chapter document can be included in multiple book documents, for reuse. Because inclusion is modeled using XInclude, content management is simplified. It is easy, for example, to replace one chapter in a book by another.

Example 23-1 illustrates an XML `Book` element that includes four documents. One of those documents, `part1.xml`, is also shown. Document `part1.xml` includes other documents, representing chapters.

These are some additional features of XInclude:

- Inclusion of plain text – You can include unparsed, non-XML text using attribute `parse` with a value of `text`: `parse = "text"`.
- Inclusion of XML fragments – You can use an `xpointer` attribute in an `xi:include` element to specify an XML fragment to include, instead of an entire document.
- Fallback processing – In case of error, such as inability to access the URI of an included document, an `xi:include` syntax error, or an `xpointer` reference that returns null, XInclude performs the treatment specified by element `xi:fallback`. This generally specifies an alternative element to be included. The alternative element can itself use `xi:include` to include other documents.

Example 23-1 XInclude Used in a Book Document to Include Parts and Chapters

The top-level document representing a book contains element `Book`.

```
<Book xmlns:xi="http://www.w3.org/2001/XInclude">
  <xi:include href=toc.xml"/>
  <xi:include href=part1.xml"/>
  <xi:include href=part2.xml"/>
  <xi:include href=index.xml"/>
</Book>
```

A major book part, file (resource) `part2.xml`, contains a `Part` element, which includes multiple chapter documents.

```
<?xml version="1.0"?>
<Part xmlns:xi="http://www.w3.org/2001/XInclude">
  <xi:include href="chapter5.xml"/>
  <xi:include href="chapter6.xml"/>
  <xi:include href="chapter8.xml"/>
  <xi:include href="chapter9.xml"/>
</Part>
```

Oracle XML DB Support for XLink

You can configure Oracle XML DB Repository resources so that XLink links are ignored, or so that they are mapped to Oracle XML DB document links. However, the use of XLink with Oracle XML DB Repository is *deprecated*.

Oracle XML DB supports only simple XLink links, not extended XLink links.

When an XML document containing XLink attributes is added to Oracle XML DB Repository, either as resource content or as user-defined resource metadata, special processing can occur, depending on how the repository or individual repository resources are configured. Element `XLinkConfig` of the resource configuration document, `XDBResConfig.xsd`, determines this behavior.

If you configure resources so that XLink links are mapped to Oracle XML DB document links, you can specify that the document links are to be hard or weak. Hard and weak document links have the same properties as hard and weak repository links.

The privileges needed to create or update document links are the same as those needed to create or update repository links. Even partially updating a document requires the same privileges needed to delete the entire document and reinsert it. In particular, even if you update just one document link you must have delete and insert privileges for *each* of the documents linked by the document containing the link.

If configuration maps XLink links to document links, then, whenever a document containing XLink links is added to the repository, the XLink information is extracted and stored in a system link table. Link target (destination) locations are replaced by direct paths that are based on the resource OIDs. Configuration can also specify whether OID paths are to be replaced by named paths (URLs) upon document retrieval. Using OID paths instead of named paths generally offers a performance advantage when links are processed, including when resource contents are retrieved.

You can use XLink within resource content, but not within resource metadata.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

Related Topics

- Use View DOCUMENT_LINKS to Examine XLink and XInclude Links
  You can query the read-only public view `DOCUMENT_LINKS` to obtain system information about document links derived from both XLink (*deprecated*) and XInclude links. The information in this view includes the following columns, for each link:
- User-Defined Repository Metadata
  You can create your own metadata to associate with XML data stored in Oracle XML DB Repository.
- Hard Links and Weak Links
  Links that target repository resources can be hard or weak. Hard and weak links have different dependencies with respect to the resources that they target. Hard links cannot target ancestor folders; weak links can. You can query the repository path view, `PATH_VIEW`, to determine the type of a repository link.
- Configuration of Repository Resources for XLink and XInclude
  The resource configuration file that you use as a resource to configure XLink (*deprecated*) and XInclude processing for other resources is described.
- XDBResConfig.xsd: XML Schema for Resource Configuration
  A full listing is presented of the Oracle XML DB-supplied XML schema used to configure repository resources. It is accessible in Oracle XML DB Repository at path `/sys/schemas/PUBLIC/xmlns.oracle.com/xdb/XDBResConfig.xsd`.

Oracle XML DB Support for XInclude

Oracle XML DB supports XInclude 1.0 as the standard mechanism for managing compound documents. It does not support attribute `xpointer` and the inclusion of document fragments, however. Only complete documents can be included (using attribute `href`).

You can use XInclude to create XML documents that include existing content. You can also configure the implicit decomposition of non-schema-based XML documents, creating a set of repository resources that contain XInclude inclusion references.

The content of included documents must be XML data or plain text (with attribute `parse = "text"`). You cannot include binary content directly using XInclude, but you can use XLink to link to binary content.

You can use XInclude within resource content, but not within resource metadata.

- Expanding Compound-Document Inclusions
  You can optionally expand `xi:include` elements with their targets when you retrieve a compound document from Oracle XML DB Repository.
- Validation of Compound Documents
  You validate a compound document the way you would any XML document. However, you can choose to validate it in either form: with `xi:include` elements as is or after replacing them with their targets.

- Update of a Compound Document
  You can update a compound document just as you would update any resource. This replaces the resource with a new value.
- Compound Document Versioning, Locking, and Access Control
  The components of a compound document are separate resources. They are versioned and locked independently, and their access is controlled independently.

Related Topics

- Use View DOCUMENT_LINKS to Examine XLink and XInclude Links
  You can query the read-only public view `DOCUMENT_LINKS` to obtain system information about document links derived from both XLink (*deprecated*) and XInclude links. The information in this view includes the following columns, for each link:

Expanding Compound-Document Inclusions

You can optionally expand `xi:include` elements with their targets when you retrieve a compound document from Oracle XML DB Repository.

When you retrieve a compound document you have a choice:

- Retrieve it as is, with the `xi:include` elements remaining as such. This is the default behavior.
- Retrieve it after replacing the `xi:include` elements with their targets, recursively, that is, after expansion of all inclusions. An error is raised if any `xi:include` element cannot be resolved.

To retrieve the document in expanded form, use PL/SQL constructor `XDBURIType`, passing a value of `'1'` or `'3'` as the second argument (flags). Example 23-2 illustrates this. These are the possible values for the second argument of constructor `XDBURIType`:

- `1` – Expand all XInclude inclusions before returning the result. If any such inclusion cannot be resolved according to the XInclude standard fallback semantics, then raise an error.
- `2` – Suppress all errors that might occur during document retrieval. This includes dangling `href` pointers.
- `3` – Same as `1` and `2` together.

Example 23-2 retrieves all documents that are under repository folder `public/bookdir`, expanding each inclusion:

(The result shown here corresponds to the resource `bookfile.xml` shown in Example 23-8, together with its included resources, `chap1.xml` and `chap2.xml`.)

See Also:

- Compound Document Versioning, Locking, and Access Control for information about access control during expansion
- *Oracle Database PL/SQL Packages and Types Reference* for more information about `XDBURIType`

Example 23-2   Expanding Document Inclusions Using XDBURIType

```
SELECT XDBURIType(ANY_PATH, '1').getXML() FROM RESOURCE_VIEW
  WHERE under_path(RES, '/public/bookdir') = 1;

XDBURITYPE(ANY_PATH,'1').GETXML()
---------------------------------
<Book>
  <Title>A book</Title>
  <Chapter id="1">
    <Title>Introduction</Title>
    <Body>
      <Para>blah blah</Para>
      <Para>foo bar</Para>
    </Body>
  </Chapter>
  <Chapter id="2">
    <Title>Conclusion</Title>
    <Body>
      <Para>xyz xyz</Para>
      <Para>abc abc</Para>
    </Body>
  </Chapter>
</Book>

<Chapter id="1">
  <Title>Introduction</Title>
  <Body>
    <Para>blah blah</Para>
    <Para>foo bar</Para>
  </Body>
</Chapter>

<Chapter id="2">
  <Title>Conclusion</Title>
  <Body>
    <Para>xyz xyz</Para>
    <Para>abc abc</Para>
  </Body>
</Chapter>

3 rows selected.
```

Validation of Compound Documents

You validate a compound document the way you would any XML document. However, you can choose to validate it in either form: with `xi:include` elements as is or after replacing them with their targets.

You can also choose to use one XML schema to validate the unexpanded form, and another to validate the expanded form. For example, you might use one XML schema to validate without first expanding, in order to set up storage structures, and then use another XML schema to validate the expanded document after it is stored.

Update of a Compound Document

You can update a compound document just as you would update any resource. This replaces the resource with a new value.

It thus corresponds to a resource deletion followed by a resource insertion. This means, in particular, that any `xi:include` elements in the original resource are deleted. Any `xi:include` elements in the replacement (inserted) document are processed as usual, according to the configuration defined at the time of insertion.

Compound Document Versioning, Locking, and Access Control

The components of a compound document are separate resources. They are versioned and locked independently, and their access is controlled independently.

- Document links to version-controlled resources (VCRs) always resolve to the latest version of the target resource, or the selected version within the current workspace. You can, however, explicitly refer to any specific version, by identifying the target resource by its OID-based path.
- Locking a document that contains `xi:include` elements does not also lock the included documents. Locking an included document does not also lock documents that include it.
- The access control list (ACL) on each referenced document is checked whenever you retrieve a compound document with expansion. This is done using the privileges of the current user (invoker's rights). If privileges are insufficient for any of the included documents, the expansion is canceled and an error is raised.

Related Topics

- Expanding Compound-Document Inclusions
  You can optionally expand `xi:include` elements with their targets when you retrieve a compound document from Oracle XML DB Repository.
- Resource Versions
  Oracle XML DB Repository resources can be versioned. A record is kept of all changes to a resource that is under version control.
- Repository Access Control
  Oracle Database provides classic database security such as row-level and column-level secure access by database users. It also provides fine-grained access control for resources in Oracle XML DB Repository. You can create, set, and modify access control lists (ACLs).

Use View DOCUMENT_LINKS to Examine XLink and XInclude Links

You can query the read-only public view `DOCUMENT_LINKS` to obtain system information about document links derived from both XLink (*deprecated*) and XInclude links. The information in this view includes the following columns, for each link:

- `SOURCE_ID` – The source resource OID. `RAW(16)`.
- `TARGET_ID` – The target resource OID. `RAW(16)`.
- `TARGET_PATH` – Always `NULL`. Reserved for future use. `VARCHAR2(4000)`.
- `LINK_TYPE` – The document link type: `Hard` or `Weak`. `VARCHAR2(8)`.
- `LINK_FORM` – Whether the original link was of form `XLink` or `XInclude`. `VARCHAR2(8)`.
- `SOURCE_TYPE` – Always `Resource Content`. `VARCHAR2(17)`.

You can obtain information about a resource from this view only if one of the following conditions holds:

- The resource is a link source, and you have the privilege `read-contents` or `read-properties` on it.
- The resource is a link target, and you have the privilege `read-properties` on it.

- Querying DOCUMENT_LINKS for XLink Information
  If the folder containing a given resource has been configured to map XLink links to document hard links then you can query public view `DOCUMENT_LINKS` to obtain system information about the document links.
- Querying DOCUMENT_LINKS for XInclude Information
  You can query view `DOCUMENT_LINKS` to show the document links that are mapped from XInclude links.

See Also:

Oracle Database Reference for more information on public view `DOCUMENT_LINKS`

Querying DOCUMENT_LINKS for XLink Information

If the folder containing a given resource has been configured to map XLink links to document hard links then you can query public view `DOCUMENT_LINKS` to obtain system information about the document links.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

Example 23-3 shows how XLink links are treated when resources are created, and how to obtain system information about document links from view `DOCUMENT_LINKS`. It assumes that the folder containing the resource has been configured to map XLink links to document hard links.

See Also:

Example 23-5 for an example of configuring a folder to map XLink links to hard links

Example 23-3 Querying Document Links Mapped From XLink Links

```
DECLARE
  b BOOLEAN;
BEGIN
  b := DBMS_XDB_REPOS.createResource(
         '/public/hardlinkdir/po101.xml',
         '<PurchaseOrder id="101" xmlns:xlink="http://www.w3.org/1999/xlink">
            <Company xlink:type="simple"
```

```
                             xlink:href="/public/hardlinkdir/oracle.xml">Oracle Corporation</Company>
                    <Approver xlink:type="simple"
                              xlink:href="/public/hardlinkdir/quine.xml">Willard Quine</Approver>
                  </PurchaseOrder>');

          b := DBMS_XDB_REPOS.createResource(
                 '/public/hardlinkdir/po102.xml',
                 '<PurchaseOrder id="102" xmlns:xlink="http://www.w3.org/1999/xlink">
                    <Company xlink:type="simple"
                             xlink:href="/public/hardlinkdir/oracle.xml">Oracle Corporation</Company>
                    <Approver xlink:type="simple"
                              xlink:href="/public/hardlinkdir/curry.xml">Haskell Curry</Approver>
                    <ReferencePO xlink:type="simple"
                                 xlink:href="/public/hardlinkdir/po101.xml"/>
                  </PurchaseOrder>');
        END;
        /
```

```
SELECT r1.ANY_PATH source, r2.ANY_PATH target, dl.LINK_TYPE, dl.LINK_FORM
  FROM DOCUMENT_LINKS dl, RESOURCE_VIEW r1, RESOURCE_VIEW r2
  WHERE dl.SOURCE_ID = r1.RESID and dl.TARGET_ID = r2.RESID;
```

```
SOURCE                          TARGET                          LINK_TYPE LINK_FORM
------------------------------ ------------------------------ --------- ---------
/public/hardlinkdir/po101.xml /public/hardlinkdir/oracle.xml Hard      XLink
/public/hardlinkdir/po101.xml /public/hardlinkdir/quine.xml  Hard      XLink
/public/hardlinkdir/po102.xml /public/hardlinkdir/oracle.xml Hard      XLink
/public/hardlinkdir/po102.xml /public/hardlinkdir/curry.xml  Hard      XLink
/public/hardlinkdir/po102.xml /public/hardlinkdir/po101.xml  Hard      XLink
```

Querying DOCUMENT_LINKS for XInclude Information

You can query view DOCUMENT_LINKS to show the document links that are mapped from XInclude links.

Example 23-4 queries view DOCUMENT_LINKS to show all document links.

Example 23-4   Querying Document Links Mapped From XInclude Links

```
DECLARE
  ret BOOLEAN;
BEGIN
  ret := DBMS_XDB_REPOS.createResource(
           '/public/hardlinkdir/book.xml',
           '<Book xmlns:xi="http://www.w3.org/2001/XInclude">
              <xi:include href="/public/hardlinkdir/toc.xml"/>
              <xi:include href="/public/hardlinkdir/part1.xml"/>
              <xi:include href="/public/hardlinkdir/part2.xml"/>
              <xi:include href="/public/hardlinkdir/index.xml"/>
            </Book>');
END;

SELECT r1.ANY_PATH source, r2.ANY_PATH target, dl.LINK_TYPE, dl.LINK_FORM
  FROM DOCUMENT_LINKS dl, RESOURCE_VIEW r1, RESOURCE_VIEW r2
  WHERE dl.SOURCE_ID = r1.RESID and dl.TARGET_ID = r2.RESID;

SOURCE                        TARGET                        LINK_TYPE LINK_FORM
------                        ------                        --------- ---------
/public/hardlinkdir/book.xml /public/hardlinkdir/toc.xml   Hard      XInclude
/public/hardlinkdir/book.xml /public/hardlinkdir/part1.xml Hard      XInclude
```

```
/public/hardlinkdir/book.xml /public/hardlinkdir/part2.xml Hard      XInclude
/public/hardlinkdir/book.xml /public/hardlinkdir/index.xml Hard      XInclude
```

Configuration of Repository Resources for XLink and XInclude

The resource configuration file that you use as a resource to configure XLink (*deprecated*) and XInclude processing for other resources is described.

You configure XLink and XInclude treatment for Oracle XML DB Repository resources as you would configure any other treatment of repository resources: using a resource configuration file. See Configuring a Resource for an example.

A resource configuration file is an XML file that conforms to the XML schema `XDBResConfig.xsd`, which is accessible in Oracle XML DB Repository at path `/sys/schemas/PUBLIC/xmlns.oracle.com/xdb/XDBResConfig.xsd`.

You use elements `XLinkConfig` and `XIncludeConfig`, children of element `ResConfig`, to configure XLink and XInclude treatment, respectively. If one of these elements is absent, then there is no treatment of the corresponding type of links.

Both `XLinkConfig` and `XIncludeConfig` can have attribute `UnresolvedLink` and child elements `LinkType` and `PathFormat`. Element `XIncludeConfig` can also have child element `ConflictRule`. If the `LinkType` element content is `None`, however, then there must be no `PathFormat` or `ConflictRule` element.

You cannot define any preconditions for `XLinkConfig` or `XIncludeConfig`. During repository resource creation, the `ResConfig` element of the parent folder determines the treatment of XLink and XInclude links for the new resource. If the parent folder has no `ResConfig` element, then the repository-wide configuration applies.

Any change to the resource configuration file applies only to documents that are created or updated after the configuration-file change. To process links in existing documents, use PL/SQL procedure `DBMS_XDB_REPOS.processLinks`, after specifying the appropriate resource configuration parameters.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

- Configure the Treatment of Unresolved Links: Attribute UnresolvedLink
  A `LinkConfig` element can have an `UnresolvedLink` attribute with a value of `Error` (default value) or `Skip`. This determines what happens if an XLink (*deprecated*) or XInclude link cannot be resolved at the time of document insertion into the repository (resource creation).
- Configure the Type of Document Links to Create: Element LinkType
  You use the `LinkType` element of a resource configuration file to specify the type of document link to be created whenever an XLink (*deprecated*) or XInclude link is encountered when a document is stored in Oracle XML DB Repository. The `LinkType` element has these possible values (element content):

- Configure the Path Format for Retrieval: Element PathFormat
  You use the `PathFormat` element of a resource configuration file to specify the path format to be used when retrieving documents with `xlink:href` or `xi:include:href` attributes.
- Configure Conflict-Resolution for XInclude: Element ConflictRule
  You use the `ConflictRule` element of a resource configuration file to specify the conflict-resolution rules to use if the path computed for a component document is already present in Oracle XML DB Repository. The `ConflictRule` element has these possible values (element content):
- Configure the Decomposition of Documents Using XInclude: Element SectionConfig
  You use element `SectionConfig` of a resource configuration file to specify how non-schema-based XML documents are to be decomposed when they are added to Oracle XML DB Repository to create a set of resources that contain XInclude inclusion references.
- XLink and XInclude Configuration Examples
  Examples show how XLink (*deprecated*) and XInclude are to be related to Oracle XML DB Repository resources and links.

Related Topics

- Manage XLink and XInclude Links Using DBMS_XDB_REPOS.processLinks
  You can use PL/SQL procedure `DBMS_XDB_REPOS.processLinks` to manually process all XLink (*deprecated*) and XInclude links in a single document or in all documents of a folder.
- Configuration of Oracle XML DB Repository
  Overall configuration of Oracle XML DB Repository applies to all repository resources. It does not include configuring parameters for handling events or managing XLink and XInclude processing. You use resource configuration files to configure resources.

Configure the Treatment of Unresolved Links: Attribute UnresolvedLink

A `LinkConfig` element can have an `UnresolvedLink` attribute with a value of `Error` (default value) or `Skip`. This determines what happens if an XLink (*deprecated*) or XInclude link cannot be resolved at the time of document insertion into the repository (resource creation).

`Error` means raise an error and roll back the current operation. `Skip` means skip any treatment of the XLink or XInclude link. Skipping treatment creates the resource with no corresponding document links, and sets the resource's `HasUnresolvedLinks` attribute to `true`, to indicate that the resource has unresolved links.

Using `Skip` as the value of attribute `UnresolvedLink` can be especially useful when you create a resource that contains a cycle of weak links, which would otherwise lead to unresolved-link errors during resource creation. After the resource and all of its linked resources have been created, you can use PL/SQL procedure `DBMS_XDB_REPOS.processLinks` to process the skipped links. If all XLink and XInclude links have been resolved by this procedure, then attribute `HasUnresolvedLinks` is set to `false`.

Resource attribute `HasUnresolvedLinks` is also set to `true` for a resource that has a weak link to a resource that has been deleted. Deleting a resource thus effectively also deletes any weak links pointing to that resource. In particular, whenever the last hard link to a resource is deleted, the resource is itself deleted, and all resources that point to the deleted resource with a weak link have attribute `HasUnresolvedLinks` set to `true`.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

Related Topics

- Hard Links and Weak Links
  Links that target repository resources can be hard or weak. Hard and weak links have different dependencies with respect to the resources that they target. Hard links cannot target ancestor folders; weak links can. You can query the repository path view, `PATH_VIEW`, to determine the type of a repository link.
- Manage XLink and XInclude Links Using DBMS_XDB_REPOS.processLinks
  You can use PL/SQL procedure `DBMS_XDB_REPOS.processLinks` to manually process all XLink (*deprecated*) and XInclude links in a single document or in all documents of a folder.

Configure the Type of Document Links to Create: Element LinkType

You use the `LinkType` element of a resource configuration file to specify the type of document link to be created whenever an XLink (*deprecated*) or XInclude link is encountered when a document is stored in Oracle XML DB Repository. The `LinkType` element has these possible values (element content):

- `None` (default) – Ignore XLink or XInclude links: create no corresponding document links.
- `Hard` – Map XLink or XInclude links to hard document links in repository documents.
- `Weak` – Map XLink or XInclude links to weak document links in repository documents.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

See Also:

- Example 23-5
- Example 23-6

Configure the Path Format for Retrieval: Element PathFormat

You use the `PathFormat` element of a resource configuration file to specify the path format to be used when retrieving documents with `xlink:href` or `xi:include:href` attributes.

The `PathFormat` element has these possible values (element content) for hard and weak document links:

- `OID` (default) – Map XLink or XInclude `href` paths to OID-based paths in repository documents — that is, use OIDs directly.
- `Named` – Map XLink or XInclude `href` paths to named paths (URLs) in repository documents. The path is computed from the internal OID when the document is retrieved, so retrieval can be slower than in the case of using OID paths directly.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

See Also:

- Example 23-5
- Example 23-6

Configure Conflict-Resolution for XInclude: Element ConflictRule

You use the `ConflictRule` element of a resource configuration file to specify the conflict-resolution rules to use if the path computed for a component document is already present in Oracle XML DB Repository. The `ConflictRule` element has these possible values (element content):

- `Error` (default) – Raise an error.
- `Overwrite` – Update the document targeted by the existing repository path, replacing it with the document to be included. If the existing document is a version-controlled resource, then it must already be checked out, unless it is autoversioned. Otherwise, an error is raised.
- `Syspath` – Change the path to the included document to a new, system-defined path.

Related Topics

- Resource Versions
  Oracle XML DB Repository resources can be versioned. A record is kept of all changes to a resource that is under version control.

Configure the Decomposition of Documents Using XInclude: Element SectionConfig

You use element `SectionConfig` of a resource configuration file to specify how non-schema-based XML documents are to be decomposed when they are added to Oracle XML DB Repository to create a set of resources that contain XInclude inclusion references.

You use simple XPath expressions in the resource configuration file to identify which parts of a document to map to separate resources, and which resources to map them to.

Element `SectionConfig` contains one or more `Section` elements, each of which contains the following child elements:

- `sectionPath` – Simple XPath 1.0 expression that identifies a section root. This must use only child and descendant axes, and it must not use wildcards.
- `documentPath` (optional) – Simple XPath 1.0 expression that is evaluated to identify the resources to be created from decomposing the document according to `sectionPath`. The XPath expression must use only child, descendant, and attribute axes.
- `namespace` (optional) – Namespace in effect for `sectionPath` and `documentPath`.

Element `Section` also has a `type` attribute that specifies the type of section to be created. Value `Document` means create a document. The default value, `None`, means do not create anything. Using `None` is equivalent to removing the `SectionConfig` element. You can thus set the `type` attribute to `None` to disable a `SectionConfig` element temporarily, without removing it, and then set it back to `Document` to enable it again.

If an element in the document being added to the repository matches more than one `sectionPath` value, only the first such expression (in document order) is used.

If no `documentPath` element is present, then the resource created has a system-defined name, and is put into the folder specified for the original document.

See Also:

- Example 23-7
- Example 23-8

XLink and XInclude Configuration Examples

Examples show how XLink (*deprecated*) and XInclude are to be related to Oracle XML DB Repository resources and links.

Example 23-5 shows a configuration-file section that configures XInclude treatment, mapping XInclude attributes to Oracle XML DB Repository hard document links. Repository paths in retrieved resources are configured to be based on resource OIDs.

Example 23-6 shows an `XLinkConfig` section that maps XLink links to weak document links in the repository. In this case, retrieval of a document uses named paths (URLs).

Example 23-7 shows a `SectionConfig` section that specifies that each `Chapter` element in an input document is to become a separate repository file, when the input document is added to Oracle XML DB Repository. The repository path for the resulting file is specified using configuration element `documentPath`, and this path is relative to the location of the resource configuration file of Example 23-6.

The XPath expression here uses XPath function `concat` to concatenate the following strings to produce the resulting repository path to use:

- `chap` – (prefix) `chap`.
- The value of attribute `id` of element `Chapter` in the input document.
- `.xml` as a file extension.

For example, a repository path of `chap27.xml` would result from an input document with a `Chapter` element that has an `id` attribute with value `27`:

```
<Chapter id="27"> ... </Chapter>
```

If the configuration document of Example 23-6 and the book document that contains the `XInclude` elements are in repository folder `/public/bookdir`, then the individual chapter files generated from `XInclude` decomposition are in files `/public/bookdir/chapN.xml`, where the values of *N* are the values of the `id` attributes of `Chapter` elements.

The book document that is added to the repository is derived from the input book document. The embedded `Chapter` elements in the input book document are replaced by `xi:include` elements that reference the generated chapter documents — Example 23-8 illustrates this.

Note:

The use of XLink with Oracle XML DB Repository is *deprecated*, starting with Oracle Database 12c Release 2 (12.2.0.1).

Example 23-5 Mapping XInclude Links to Hard Document Links, with OID Retrieval

```
<ResConfig>
  . . .
  <XIncludeConfig UnresolvedLink="Skip">
    <LinkType>Hard</LinkType>
    <PathFormat>OID</PathFormat>
  </XIncludeConfig>
  . . .
</ResConfig>
```

Example 23-6 Mapping XLInk Links to Weak Links, with Named-Path Retrieval

```
<ResConfig>
  . . .
  <XLinkConfig UnresolvedLink="Skip">
    <LinkType>Weak</LinkType>
    <PathFormat>Named</PathFormat>
  </XLinkConfig>
  . . .
</ResConfig>
```

Example 23-7 Configuring XInclude Document Decomposition

```
<ResConfig>
  . . .
  <SectionConfig>
    <Section type = "Document">
      <sectionPath>//Chapter</sectionPath>
      <documentPath>concat("chap", @id, ".xml")</documentPath>
    </Section>
  </SectionConfig>
```

```
                  . . .
              </ResConfig>
```

Example 23-8   Repository Document, Showing Generated xi:include Elements

```
SELECT XDBURIType('/public/bookdir/bookfile.xml').getclob() FROM DUAL;

XDBURITYPE('/PUBLIC/BOOKDIR/BOOKFILE.XML').GETCLOB()
--------------------------------------------------------------------------------
<Book>
  <Title>A book</Title>
  <xi:include xmlns:xi="http://www.w3.org/2001/XInclude" href="/public/bookdir/chap1.xml"/>
  <xi:include xmlns:xi="http://www.w3.org/2001/XInclude" href="/public/bookdir/chap2.xml"/>
</Book>
```

Related Topics

- Configuration of Oracle XML DB Repository
  Overall configuration of Oracle XML DB Repository applies to all repository resources. It does not include configuring parameters for handling events or managing XLink and XInclude processing. You use resource configuration files to configure resources.
- XDBResConfig.xsd: XML Schema for Resource Configuration
  A full listing is presented of the Oracle XML DB-supplied XML schema used to configure repository resources. It is accessible in Oracle XML DB Repository at path `/sys/schemas/PUBLIC/xmlns.oracle.com/xdb/XDBResConfig.xsd`.
- Configure the Decomposition of Documents Using XInclude: Element SectionConfig
  You use element `SectionConfig` of a resource configuration file to specify how non-schema-based XML documents are to be decomposed when they are added to Oracle XML DB Repository to create a set of resources that contain XInclude inclusion references.

Manage XLink and XInclude Links Using DBMS_XDB_REPOS.processLinks

You can use PL/SQL procedure `DBMS_XDB_REPOS.processLinks` to manually process all XLink (*deprecated*) and XInclude links in a single document or in all documents of a folder.

Pass `RECURSIVE` as the mode argument to this procedure, if you want to process all hard-linked subfolders recursively. All XLink and XInclude links are processed according to the corresponding configuration parameters. If any of the links within a resource cannot be resolved, the resource's `HasUnresolvedLinks` attribute is set to `true`, to indicate that the resource has unresolved links. The default value of attribute `HasUnresolvedLinks` is `false`.

Related Topics

- Configure the Treatment of Unresolved Links: Attribute UnresolvedLink
  A `LinkConfig` element can have an `UnresolvedLink` attribute with a value of `Error` (default value) or `Skip`. This determines what happens if an XLink (*deprecated*) or XInclude link cannot be resolved at the time of document insertion into the repository (resource creation).