34 Administration of Oracle XML DB

Administration of Oracle XML DB includes installing, upgrading, and configuring it.

- Upgrade or Downgrade of an Existing Oracle XML DB Installation
  Various considerations apply to all upgrades of Oracle Database from a release prior to Oracle Database 12c Release 1 (12.1.0.1).
- Administration of Oracle XML DB Using Oracle Enterprise Manager
  You can use Oracle Enterprise Manager to easily perform Oracle XML DB administration tasks. This includes configuring Oracle XML DB; creating, searching, editing, and deleting XML schemas, `XMLType` tables and views, and Oracle XML DB Repository *resources* and their *access control lists* (ACLs); registering XML schemas; and creating function-based indexes.
- Configuration of Oracle XML DB Using xdbconfig.xml
  Oracle XML DB is managed internally through a configuration file, `xdbconfig.xml`, which is stored as a resource in Oracle XML DB Repository. As an alternative to using Oracle Enterprise Manager to configure Oracle XML DB, you can configure it directly using the Oracle XML DB configuration file.
- Oracle XML DB and Database Consolidation
  Each pluggable database has its own Oracle XML DB Repository, and its own Oracle XML DB configuration file, `xdbconfig.xml`.
- Package DBMS_XDB_ADMIN
  PL/SQL package `DBMS_XDB_ADMIN` has subprograms for managing and configuring Oracle XML DB and Oracle XML DB Repository.

Related Topics

- 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.

Upgrade or Downgrade of an Existing Oracle XML DB Installation

Various considerations apply to all upgrades of Oracle Database from a release prior to Oracle Database 12c Release 1 (12.1.0.1).

- Run script `catproc.sql`, as always.
- Replication of hierarchy-enabled tables is not supported for any replication method, including rolling upgrade.
- If supplemental logging is turned on then these operations are not supported:
  - Use of the `APPEND` hint for `INSERT`
  - SQL*Loader direct-path insertion of `XMLType` data
- If you use rolling upgrade and any of the following operations are invoked on the primary database, then an *unsupported operation* error is raised on the standby database:

  - `DBMS_XDB_ADMIN` — all operations
  - `DBMS_XMLSCHEMA.copyEvolve`
  - `DBMS_XMLSCHEMA.compileSchema`
  - `DBMS_XMLINDEX.dropparameter`
  - `DBMS_XMLINDEX.modifyparameter`
  - `DBMS_XMLINDEX.registerparameter`
- ACL security: In releases prior to Oracle Database 11g Release 1, conflicts among ACEs for the same principal and same privilege were resolved by giving priority to any ACE that had child `deny`, whether or not preceding ACEs had child `grant`. That is, ACE order did not matter. In Oracle Database 11g and later this `deny-trumps-grant` behavior is still available, but it is not the default behavior.
- Prior to Oracle Database 12c Release 1 (12.1.0.1), basic access authentication was the only available HTTP authentication mechanism. Starting with 12c Release 1, digest access authentication is available.

- Authentication Considerations for Database Installation, Upgrade and Downgrade
  Various authentication considerations apply to database installation, upgrades and downgrades.
- Automatic Installation of Oracle XML DB
  Oracle XML DB is automatically installed during an upgrade. Ensure that the compatibility level is at least 12.1.0.1 if you intend to use Oracle XML DB.
- Validation of ACL Documents and Configuration File
  During upgrade, existing ACL documents and the Oracle XML DB configuration file are validated against their XML schemas. If validation fails then you need to take certain measures.

> **See Also:**
>
> - *Oracle Data Guard Concepts and Administration* for information about performing a rolling upgrade
> - ACL and ACE Evaluation for information about conflicts among ACEs

Authentication Considerations for Database Installation, Upgrade and Downgrade

Various authentication considerations apply to database installation, upgrades and downgrades.

- Authentication Considerations for a Database Installation
  In a default database installation, digest authentication is enabled, and basic authentication is disallowed. Digest verifiers are automatically generated for *all* users.
- Authentication Considerations for a Database Upgrade
  After an upgrade from a release prior to Oracle Database 12c Release 1 (12.1.0.1), digest authentication is appended to the list of allowed authentication mechanisms. But basic

authentication remains the current authentication method if it was enabled before the upgrade. This is for backward compatibility *only*.

- Authentication Considerations for a Database Downgrade
  If you downgrade to a release where digest authentication was not supported, digest authentication is disabled and made unavailable as an authentication choice.

Related Topics

- Configuration and Management of Authentication Mechanisms for HTTP
  You configure the authentication mechanisms to allow for HTTP access to Oracle XML DB Repository by setting element `authentication`, a child of element `httpconfig`, in configuration file `xdbconfig.xml`.

Authentication Considerations for a Database Installation

In a default database installation, digest authentication is enabled, and basic authentication is disallowed. Digest verifiers are automatically generated for *all* users.

Authentication Considerations for a Database Upgrade

After an upgrade from a release prior to Oracle Database 12c Release 1 (12.1.0.1), digest authentication is appended to the list of allowed authentication mechanisms. But basic authentication remains the current authentication method if it was enabled before the upgrade. This is for backward compatibility *only*.

Oracle recommends that your database administrator *disable basic authentication as soon as possible* after upgrading. The reason that basic authentication remains in effect after such an upgrade is to allow users to change their passwords using a Web browser that does not support digest authentication.

For such an upgrade, digest verifiers are computed for all *new* users and for all previously existing *users whose passwords changed* during the upgrade. Other users do *not* have digest verifiers.

After an upgrade, a DBA can use database view `DBA_DIGEST_VERIFIERS` to check which users have digest verifiers and take appropriate action, as follows:

1. Configure *basic* authentication as the *first* allowed authentication mechanism in the Oracle XML DB configuration file, `xdbconfig.xml`. This ensures that basic authentication can be used for HTTP access.
2. Expire all passwords for those users who do not have digest verifiers. This query returns those users:

   ```
   SELECT USERNAME FROM DBA_DIGEST_VERIFIERS
     WHERE HAS_DIGEST_VERIFIERS = 'NO' AND DIGEST_TYPE is NULL;
   ```

3. After the passwords for all such users have been changed, configure `xdbconfig.xml` to re-enable digest as the first or (preferably) the only allowed authentication mechanism.

Authentication Considerations for a Database Downgrade

If you downgrade to a release where digest authentication was not supported, digest authentication is disabled and made unavailable as an authentication choice.

All digest verifiers are *erased* during a downgrade. This means, in particular, that if a downgrade is followed by an upgrade then users who were able to authenticate prior to the downgrade are denied digest authentication after the downgrade and the subsequent upgrade.

Automatic Installation of Oracle XML DB

Oracle XML DB is automatically installed during an upgrade. Ensure that the compatibility level is at least 12.1.0.1 if you intend to use Oracle XML DB.

If Oracle XML DB is not already installed in your database prior to an upgrade to Oracle Database 12c Release 1 (12.1.0.1) or later, then it is automatically installed in tablespace `SYSAUX` during the upgrade.

If Oracle XML DB has thus been automatically installed, and if you want to use Oracle XML DB, then, after the upgrade operation, you must set the database compatibility to at least 12.1.0.1. If the compatibility is less than 12.1.0.1 then an error is raised when you try to use Oracle XML DB.

If Oracle XML DB was automatically installed during an upgrade and the current compatibility level is less than 12.1.0.1, then Oracle DB is automatically uninstalled during any downgrade to a prior release.

Validation of ACL Documents and Configuration File

During upgrade, existing ACL documents and the Oracle XML DB configuration file are validated against their XML schemas. If validation fails then you need to take certain measures.

Access control list (ACL) documents are stored in table `XDB$ACL`. The Oracle XML DB configuration file, `xdbconfig.xml`, is stored in table `XDB$CONFIG`. Starting with Oracle Database 12c Release 1 (12.1.0.1), these tables use the post-parse (binary XML) storage model. This implies that ACL documents and the configuration file are fully validated against their respective XML schemas. Validation takes place during upgrade, using your existing ACL documents and configuration file and the corresponding existing XML schemas.

If an ACL document fails to validate during upgrade, then the document is moved to table `XDB$INVALID_ACL`.

If validation of configuration file `xdbconfig.xml` fails during upgrade, then the file is saved in table `XDB$INVALID_CONFIG`, the default configuration file replaces it in table `XDB$CONFIG`, and the `XDB` component of the database is marked invalid. You must then start the database in normal mode and fix the `XDB` component, before trying to use the database.

To fix the `XDB` component, you can fix the invalid files to make them valid, and then call PL/SQL procedure `RecoverUpgrade`. After validating, this procedure moves the fixed files to tables `XDB$ACL` and `XDB$CONFIG`, and marks the `XDB` component valid.

As an option, you can call procedure `RecoverUpgrade` with parameter `use_default` set to `TRUE`, to abandon any invalid files. In this case, any valid files are moved to tables `XDB$ACL` and `XDB$CONFIG`, and any remaining invalid files are deleted. Default files are used in place of any invalid files. For ACLs, the default ACL document is used. For the configuration file, the default `xdbconfig.xml` is used (in which ACE order matters).

> **Caution:**
>
> Use a `TRUE` value for parameter `use_default` *only* if you are certain that you no longer need the old ACL files or configuration file that are invalid. These files are *deleted*.

Administration of Oracle XML DB Using Oracle Enterprise Manager

You can use Oracle Enterprise Manager to easily perform Oracle XML DB administration tasks. This includes configuring Oracle XML DB; creating, searching, editing, and deleting XML schemas, `XMLType` tables and views, and Oracle XML DB Repository *resources* and their *access control lists* (ACLs); registering XML schemas; and creating function-based indexes.

See the online help available with Oracle Enterprise Manager for information about using Enterprise Manager to perform the following tasks:

- Configure Oracle XML DB. View or edit parameters for the Oracle XML DB configuration file, `xdbconfig.xml`.
- Search, create, edit, undelete Oracle XML DB Repository *resources* and their associated *access control lists* (ACLs).
- Search, create, edit, and delete `XMLType` *tables* and *views*.
- Search, create, register, and delete *XML schemas*.

> **See Also:**
>
> - Configuration of Oracle XML DB Using xdbconfig.xml for information about configuring Oracle XML DB without using Oracle Enterprise Manager
> - Oracle XML DB Repository for information about creating and managing resources without using Oracle Enterprise Manager
> - XML Schema Storage and Query: Basic for information about manipulating XML schemas without using Oracle Enterprise Manager

Configuration of Oracle XML DB Using xdbconfig.xml

Oracle XML DB is managed internally through a configuration file, `xdbconfig.xml`, which is stored as a resource in Oracle XML DB Repository. As an alternative to using Oracle Enterprise Manager to configure Oracle XML DB, you can configure it directly using the Oracle XML DB configuration file.

The configuration file can be modified at run time. Updating the configuration file creates a new version of this repository resource. At the start of each session, the current version of the configuration file is bound to that session. The session uses this configuration-file version for its duration, unless you make an explicit call to refresh the session to the latest version.

- Oracle XML DB Configuration File, xdbconfig.xml
  The configuration of Oracle XML DB is defined and stored in an Oracle XML DB Repository resource, `xdbconfig.xml`, which conforms to the Oracle XML DB configuration XML schema, `http://xmlns.oracle.com/xdb/xdbconfig.xsd`. To configure or reconfigure Oracle XML DB, update the configuration file, `xdbconfig.xml`. You need administrator privileges to access file `xdbconfig.xml`.

Oracle XML DB Configuration File, xdbconfig.xml

The configuration of Oracle XML DB is defined and stored in an Oracle XML DB Repository resource, `xdbconfig.xml`, which conforms to the Oracle XML DB configuration XML schema, `http://xmlns.oracle.com/xdb/xdbconfig.xsd`. To configure or reconfigure Oracle XML DB, update the configuration file, `xdbconfig.xml`. You need administrator privileges to access file `xdbconfig.xml`.

The structure of the configuration file is described in the following sections.

- Element xdbconfig (Top-Level)
  Element `xdbconfig` is the top-level element of the Oracle XML DB configuration file.
- Element sysconfig (Child of xdbconfig)
  Element `sysconfig` is a child of `xdbconfig`.
- Element userconfig (Child of xdbconfig)
  Element `userconfig` is a child of `xdbconfig`. It contains any parameters that you may want to add.
- Element protocolconfig (Child of sysconfig)
  Element `protocolconfig` is a child of `sysconfig`.
- Element httpconfig (Child of protocolconfig)
  Element `httpconfig`, which is a child of `protocolconfig`, is described.
- Element servlet (Descendant of httpconfig)
  Element `servlet` is a descendent of element `httpconfig`. It is used to configure servlets, including Java servlets and embedded PL/SQL gateway servlets.
- Oracle XML DB Configuration File Example
  A sample Oracle XML DB configuration file is presented.
- Oracle XML DB Configuration API
  You can access the Oracle XML DB configuration file, `xdbconfig.xml`, the same way you access any other XML schema-based resource. You can use FTP, HTTP(S), WebDAV, Oracle Enterprise Manager, or any of the resource and Document Object Model (DOM) APIs for Java, PL/SQL, or C (OCI).
- Configuration of Mappings from Default Namespace to Schema Location
  Oracle XML DB provides a mechanism to configure default XML schema location mappings.
- Configuration of XML File Extensions
  Oracle XML DB Repository treats certain files as XML documents, based on their file extensions. When such files are inserted into the repository, Oracle XML DB pre-parses them to identify the XML schema location (or uses the default mapping if present) and inserts the document into the appropriate default table.

See Also:

xdbconfig.xsd: XML Schema for Configuring Oracle XML DB for a complete listing of the Oracle XML DB configuration XML schema

Element xdbconfig (Top-Level)

Element `xdbconfig` is the top-level element of the Oracle XML DB configuration file.

Its structure is as follows:

```
<xdbconfig>
    <sysconfig> ... </sysconfig>
    <userconfig> ... </userconfig>
</xdbconfig>
```

Element `sysconfig` defines system-specific, built-in parameters. Element `userconfig` lets you store new custom parameters.

Element sysconfig (Child of xdbconfig)

Element `sysconfig` is a child of `xdbconfig`.

Its structure is as follows:

```
<sysconfig>
    general parameters
    <protocolconfig> ... </protocolconfig>
</sysconfig>
```

Element `sysconfig` includes as content several general parameters that apply to all of Oracle XML DB, such as the maximum age of an access control list (ACL). Child element `protocolconfig` contains protocol-specific parameters.

Note:

Element `case-sensitive`, child of element `sysconfig`, has no effect on the case-sensitivity of XQuery or full-text search. Otherwise, it affects the behavior of all of Oracle XML DB.

Element userconfig (Child of xdbconfig)

Element `userconfig` is a child of `xdbconfig`. It contains any parameters that you may want to add.

Element protocolconfig (Child of sysconfig)

Element `protocolconfig` is a child of `sysconfig`.

Its structure is as follows:

```
<protocolconfig>
  <common> ... </common>
  <ftpconfig> ... </ftpconfig>
  <httpconfig> ... </httpconfig> </protocolconfig>
```

Under element `common`, Oracle Database stores parameters that apply to all protocols, such as MIME-type information. Parameters that are specific to protocols FTP and HTTP(S) are in elements `ftpconfig` and `httpconfig`, respectively.

Related Topics

- Repository Access Using Protocols
  You can access Oracle XML DB Repository data using protocols FTP and HTTP(S)/WebDAV.

> **See Also:**
>
> Table 28-1, Table 28-2, and Table 28-3, for a list of protocol configuration parameters

Element httpconfig (Child of protocolconfig)

Element `httpconfig`, which is a child of `protocolconfig`, is described.

This is the structure of `httpconfig`:

```
<httpconfig>
  ...
  <webappconfig>
    ...
    <servletconfig>
      ...
      <servlet-list>
        <servlet> ... </servlet>
        ...
      </servlet-list>
    </servletconfig>
  </webappconfig>
  ...
  <plsql> ... </plsql>
</httpconfig>
```

Element `httpconfig` has the following child elements, in addition to others:

- `webappconfig` – used to configure Web-based applications. This includes Web application-specific parameters, such as icon name, display name for the application, and a list of servlets.

Element `servletconfig` is a child of `webappconfig` that is used to define servlets. It has child element `servlet-list`, which has child element `servlet` (see Element servlet (Descendant of httpconfig)).

- `plsql` – used to define global configuration parameters when configuring the *embedded PL/SQL gateway*. Each global parameter is defined with a child element of `plsql`. The element name is the same as the global parameter name. The element content is the same as the parameter value.

  The recommended way to configure the embedded PL/SQL gateway is to use the procedures in PL/SQL package `DBMS_EPG`, *not* to edit file `xdbconfig.xml`.

Related Topics

- Repository Access Using Protocols
  You can access Oracle XML DB Repository data using protocols FTP and HTTP(S)/WebDAV.

See Also:

- Table 28-1, Table 28-2, and Table 28-3, for a list of protocol configuration parameters
- *Oracle Database Development Guide*, for complete information about configuring and using the embedded PL/SQL gateway
- *Oracle Database PL/SQL Packages and Types Reference*, for information about package `DBMS_EPG`

Element servlet (Descendant of httpconfig)

Element `servlet` is a descendent of element `httpconfig`. It is used to configure servlets, including Java servlets and embedded PL/SQL gateway servlets.

See Element httpconfig (Child of protocolconfig).

Note:

The following servlets are preconfigured in file `xdbconfig.xml`. Do *not* delete them.

- `ORSServlet`
- `EMExpressServlet`

An optional element `plsql`, child of `servlet`[1], configures the embedded PL/SQL gateway servlet. However, the *recommended* way to configure the embedded gateway is to use the procedures in PL/SQL package `DBMS_EPG`, *not* to edit file `xdbconfig.xml`.

[1] There are two different `plsql` elements that are used to configure the embedded PL/SQL gateway. One, a child of `httpconfig`, defines *global parameters*. The other, a child of `servlet`, defines *DAD attributes*.

Element `plsql` has a child element for each embedded PL/SQL DAD attribute[2] that is needed to configure the embedded gateway. All such children are optional. The element name is the same as the DAD attribute name. The element content is the same as the DAD-attribute value.

> **See Also:**
>
> - Guidelines for Oracle XML DB Applications in Java for information about configuring Java servlets
> - *Oracle Database Development Guide*, for complete information about configuring and using the embedded PL/SQL gateway
> - *Oracle APEX App Builder User's Guide*, for information about Oracle Application Express
> - *Oracle Database PL/SQL Packages and Types Reference*, for information about package `DBMS_EPG`

Oracle XML DB Configuration File Example

A sample Oracle XML DB configuration file is presented.

Example 34-1 Oracle XML DB Configuration File

```
<xdbconfig xmlns="http://xmlns.oracle.com/xdb/xdbconfig.xsd"
                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                 xsi:schemaLocation="http://xmlns.oracle.com/xdb/xdbconfig.xsd
                                    http://xmlns.oracle.com/xdb/xdbconfig.xsd">
   <sysconfig>
      <acl-max-age>900</acl-max-age>
      <acl-cache-size>32</acl-cache-size>
      <invalid-pathname-chars>,</invalid-pathname-chars>
      <case-sensitive>true</case-sensitive>
      <call-timeout>300</call-timeout>
      <max-link-queue>65536</max-link-queue>
      <max-session-use>100</max-session-use>
      <persistent-sessions>false</persistent-sessions>
      <default-lock-timeout>3600</default-lock-timeout>
      <xdbcore-logfile-path>/sys/log/xdblog.xml</xdbcore-logfile-path>
      <xdbcore-log-level>0</xdbcore-log-level>
      <resource-view-cache-size>1048576</resource-view-cache-size>

      <protocolconfig>
          <common>
             <extension-mappings>
                <mime-mappings>
                  <mime-mapping>
                    <extension>au</extension>
                    <mime-type>audio/basic</mime-type>
                  </mime-mapping>
                  <mime-mapping>
                    <extension>avi</extension>
                    <mime-type>video/x-msvideo</mime-type>
                  </mime-mapping>
                  <mime-mapping>
                    <extension>bin</extension>
                    <mime-type>application/octet-stream</mime-type>
                  </mime-mapping>
                </mime-mappings>
```

[2] DAD is an abbreviation for Database Access Descriptor. DAD attributes are parameters that define such a descriptor.

```
          <lang-mappings>
            <lang-mapping>
              <extension>en</extension>
              <lang>english</lang>
            </lang-mapping>
          </lang-mappings>

          <charset-mappings>
          </charset-mappings>

          <encoding-mappings>
            <encoding-mapping>
              <extension>gzip</extension>
              <encoding>zip file</encoding>
            </encoding-mapping>
            <encoding-mapping>
              <extension>tar</extension>
              <encoding>tar file</encoding>
            </encoding-mapping>
          </encoding-mappings>
       </extension-mappings>

       <session-pool-size>50</session-pool-size>
       <session-timeout>6000</session-timeout>
    </common>

    <ftpconfig>
      <ftp-port>2100</ftp-port>
      <ftp-listener>local_listener</ftp-listener>
      <ftp-protocol>tcp</ftp-protocol>
      <logfile-path>/sys/log/ftplog.xml</logfile-path>
      <log-level>0</log-level>
      <session-timeout>6000</session-timeout>
      <buffer-size>8192</buffer-size>
    </ftpconfig>

    <httpconfig>
      <http-port>8080</http-port>
      <http-listener>local_listener</http-listener>
      <http-protocol>tcp</http-protocol>
      <max-http-headers>64</max-http-headers>
      <session-timeout>6000</session-timeout>
      <server-name>XDB HTTP Server</server-name>
      <max-header-size>16384</max-header-size>
      <max-request-body>2000000000</max-request-body>
      <logfile-path>/sys/log/httplog.xml</logfile-path>
      <log-level>0</log-level>
      <servlet-realm>Basic realm="XDB"</servlet-realm>
      <webappconfig>
        <welcome-file-list>
          <welcome-file>index.html</welcome-file>
          <welcome-file>index.htm</welcome-file>
        </welcome-file-list>
        <error-pages>
        </error-pages>
        <servletconfig>
          <servlet-mappings>
            <servlet-mapping>
              <servlet-pattern>/oradb/*</servlet-pattern>
              <servlet-name>DBURIServlet</servlet-name>
            </servlet-mapping>
          </servlet-mappings>

          <servlet-list>
            <servlet>
              <servlet-name>DBURIServlet</servlet-name>
              <display-name>DBURI</display-name>
              <servlet-language>C</servlet-language>
              <description>Servlet for accessing DBURIs</description>
```

```
                    <security-role-ref>
                      <role-name>authenticatedUser</role-name>
                      <role-link>authenticatedUser</role-link>
                    </security-role-ref>
                  </servlet>
                </servlet-list>
              </servletconfig>
            </webappconfig>
          </httpconfig>
      </protocolconfig>
    <xdbcore-xobmem-bound>1024</xdbcore-xobmem-bound>
    <xdbcore-loadableunit-size>16</xdbcore-loadableunit-size>
   </sysconfig>
</xdbconfig>
```

Oracle XML DB Configuration API

You can access the Oracle XML DB configuration file, xdbconfig.xml, the same way you access any other XML schema-based resource. You can use FTP, HTTP(S), WebDAV, Oracle Enterprise Manager, or any of the resource and Document Object Model (DOM) APIs for Java, PL/SQL, or C (OCI).

For convenience, you can use PL/SQL package DBMS_XDB_CONFIG package for configuration access. It exposes the following functions and procedures:

- cfg_get – Returns the configuration information for the current session.
- cfg_refresh – Refreshes the session configuration information using the current configuration file. Typical uses of cfg_refresh include the following:
  - You have modified the configuration and now want the session to pick up the latest version of the configuration information.
  - It has been a long running session, the configuration has been modified by a concurrent session, and you want the current session to pick up the latest version of the configuration information.
- cfg_update – Updates the configuration information, writing the configuration file. A COMMIT is performed.

Example 34-2 updates parameters ftp-port and http-port in the configuration file.

If you have many parameters to update, then it can be easier to use FTP, HTTP(S), or Oracle Enterprise Manager to update the configuration.

> **See Also:**
>
> *Oracle Database PL/SQL Packages and Types Reference*

Example 34-2 Updating the Configuration File Using CFG_UPDATE and CFG_GET

```
DECLARE
  v_cfg XMLType;
BEGIN
  SELECT XMLQuery('declare default element namespace
                   "http://xmlns.oracle.com/xdb/xdbconfig.xsd";
                   copy $i := $cfg
                   modify for $j in $i/xdbconfig
                   return (replace value of node $j/descendant::ftp-port with
                             "2121",
                           replace value of node $j/descendant::http-port with
                             "19090")
                   return $i'
                  PASSING DBMS_XDB_CONFIG.cfg_get() AS "cfg"
                  RETURNING CONTENT)
    INTO v_cfg FROM DUAL;
  DBMS_XDB_CONFIG.cfg_update(v_cfg);
END;
/
```

Related Topics

- DBMS_XDB_CONFIG: Configuration Management
  PL/SQL package `DBMS_XDB_CONFIG` provides Oracle XML DB configuration management functions and procedures.

Configuration of Mappings from Default Namespace to Schema Location

Oracle XML DB provides a mechanism to configure default XML schema location mappings.

Oracle XML DB identifies XML Schema-based `XMLType` instances by pre-parsing input XML documents. If the appropriate `xsi:schemaLocation` or `xsi:noNamespaceSchemaLocation` attribute is found, then the specified schema location URL is used to consult the registered schema.

If the appropriate `xsi:` attribute is not found, the XML document is considered to be non-schema-based. Oracle XML DB provides a mechanism to configure default schema location mappings. If the appropriate `xsi:` attribute is not specified in the XML document, the default schema location mappings is used.

Element `schemaLocation-mappings` of the Oracle XML DB configuration XML schema, `xdbconfig.xsd`, can be used to specify the mapping between *`(namespace, element)`* pairs and the default schema location. If the *`element`* value is empty, the mapping applies to all global elements in the specified namespace. If the *`namespace`* value is empty, it corresponds to the null namespace.

The definition of the `schemaLocation-mappings` element is as follows:

```
<element name="schemaLocation-mappings"
         type="xdbc:schemaLocation-mapping-type" minOccurs="0"/>
<complexType name="schemaLocation-mapping-type"><sequence>
     <element name="schemaLocation-mapping"
              minOccurs="0" maxOccurs="unbounded">
     <complexType><sequence>
       <element name="namespace" type="string"/>
       <element name="element" type="string"/>
       <element name="schemaURL" type="string"/>
     </sequence></complexType>
     </element></sequence>
</complexType>
```

The schema location used depends on mappings in the Oracle XML DB configuration file for the namespace used and the root document element. For example, assume that the document does not have the appropriate `xsi:` attribute to indicate the schema location. Consider a document root element *R* in namespace *N*. The algorithm for identifying the default schema location is as follows:

1. If the Oracle XML DB configuration file has a mapping for *N* and *R*, the corresponding schema location is used.
2. If the configuration file has a mapping for *N*, but not *R*, the schema location for *N* is used.
3. If the document root *R* does not have any namespace, the schema location for *R* is used.

For example, suppose that your Oracle XML DB configuration file includes the following mapping:

```
<schemaLocation-mappings>
   <schemaLocation-mapping>
      <namespace>http://www.oracle.com/example</namespace>
      <element>root</element>
      <schemaURL>http://www.oracle.com/example/sch.xsd</schemaURL>
   </schemaLocation-mapping>
   <schemaLocation-mapping>
      <namespace>http://www.oracle.com/example2</namespace>
      <element></element>
      <schemaURL>http://www.oracle.com/example2/sch.xsd</schemaURL>
   </schemaLocation-mapping>
   <schemaLocation-mapping>
     <namespace></namespace>
     <element>specialRoot</element>
     <schemaURL>http://www.oracle.com/example3/sch.xsd</schemaURL>
   </schemaLocation-mapping>
</schemaLocation-mappings>
```

The following schema locations are used:

- Root element = `root`
  - Namespace = `http://www.oracle.com/example`
  - Schema URL = `http://www.oracle.com/example/sch.xsd`

  This mapping is used when the instance document specifies:

  `<root xmlns="http://www.oracle.com/example">`
- Root element = `null` (any global element in the namespace)
  - Namespace = `http://www.oracle.com/example2`
  - Schema URL = `http://www.oracle.com/example2/sch.xsd`

  This mapping is used when the instance document specifies:

  `<root xmlns="http://www.oracle.example2">`
- Root element = `specialRoot`
  - Namespace = `null` (i.e null namespace)
  - Schema URL = `http://www.oracle.com/example3/sch.xsd`

  This mapping is used when the instance document specifies:

  `<specialRoot>`

> **Note:**
>
> This functionality is available only on the server side, that is, when XML is parsed on the server. If XML is parsed on the client side, the appropriate `xsi:` attribute is still required.

Configuration of XML File Extensions

Oracle XML DB Repository treats certain files as XML documents, based on their file extensions. When such files are inserted into the repository, Oracle XML DB pre-parses them to identify the XML schema location (or uses the default mapping if present) and inserts the document into the appropriate default table.

By default, the following extensions are considered as XML file extensions: **`xml`**, **`xsd`**, **`xsl`**, **`xlt`**. In addition, Oracle XML DB provides a mechanism for applications to specify other file extensions as XML file extensions. The **`xml-extensions`** element is defined in the configuration schema, `http://xmlns.oracle.com/xdb/xdbconfig.xsd`, as follows:

```
<element name="xml-extensions"
         type="xdbc:xml-extension-type" minOccurs="0"/>

<complexType name="xml-extension-type"><sequence>
     <element name="extension" type="xdbc:exttype"
              minOccurs="0" maxOccurs="unbounded">
     </element></sequence>
 </complexType>
```

For example, the following fragment from the Oracle XML DB configuration file, `xdbconfig.xml`, specifies that files with extensions `vsd`, `vml`, and `svgl` should be treated as XML files:

```
<xml-extensions>
  <extension>vsd</extension>
  <extension>vml</extension>
  <extension>svgl</extension>
</xml-extensions>
```

Oracle XML DB and Database Consolidation

Each pluggable database has its own Oracle XML DB Repository, and its own Oracle XML DB configuration file, `xdbconfig.xml`.

A multitenant container database (CDB) consists of zero or more pluggable databases (PDBs), a root, and a seed PDB (a template for creating PDBs). A given PDB can be associated with only one CDB at a time. A PDB appears to users and applications as a separate database. Your applications always interact with a single PDB at a time. Queries and dictionary views are local to a PDB. Each PDB has its own Oracle XML DB Repository, and its own Oracle XML DB configuration file, `xdbconfig.xml`.

The root of a CDB (`CDB$ROOT`) contains no user data. It does, however, have its own configuration file, `xdbconfig.xml`.

The root configuration file for a CDB has only certain parameters, and those parameters are used only from the root configuration file. If any of those parameters are also present in a configuration file of a PDB that is part of a CDB, they are ignored in favor of the corresponding parameters in the root configuration file.

These are the configuration parameters that are used from the root (and ignored from any PDBs):

- `acl-cache-size`
- `acl-max-age`
- `resource-view-cache-size`
- `xdbcore-loadableunit-size`
- `xdbcore-xobmem-bound`

Parameters `xdbcore-loadableunit-size` and `xdbcore-xobmem-bound` are process-specific. The others are SGA configuration parameters.

Database schema (user account) `XDB` is a common user, which means that it can connect to, and perform operations within, both the root and any PDBs.

Note:

Oracle recommends that you never unlock database schema XDB, under any circumstances.

See Also:

- *Oracle Multitenant Administrator's Guide* and *Oracle Multitenant Administrator's Guide* for information about database consolidation and PDBs
- Performance Guidelines for Oracle XML DB Repository Operations for information about configuration parameters `resource-view-cache-size`, `xdbcore-loadableunit-size`, and `xdbcore-xobmem-bound`
- Considerations for Loading and Retrieving Large Documents with Collections for information about configuration parameters `xdbcore-loadableunit-size` and `xdbcore-xobmem-bound`
- ACL Caching for information about configuration parameter `acl-max-age`

Package DBMS_XDB_ADMIN

PL/SQL package `DBMS_XDB_ADMIN` has subprograms for managing and configuring Oracle XML DB and Oracle XML DB Repository.

Table 34-1 describes these subprograms.

Table 34-1   DBMS_XDB_ADMIN Management Procedures

Function/Procedure	Description
`moveXDB_tablespace`	Move database schema (user account) `XDB` to the specified tablespace.
`rebuildHierarchicalIndex`	Rebuild the hierarchical repository index. This can be needed from time to time, in particular after invoking `moveXDB_tablespace`.

> **Note:**
>
> Prior to Oracle Database 11g Release 2 (11.2.0.3), these procedures belonged to PL/SQL package `DBMS_XDB`. These two procedures in package `DBMS_XDB` are *deprecated* as of release 11.2.0.3.

> **Note:**
>
> The tablespace containing Oracle XML DB Repository *must not* be read-only, because that would adversely affect XML operations.
>
> By default, Oracle XML DB Repository resides in tablespace `SYSAUX`, which is used for other things as well. Oracle recommends instead that you create a dedicated tablespace for use only by the repository.
>
> Use procedure `DBMS_XDB_ADMIN.moveXDB_tablespace` to move database schema `XDB` to that tablespace dedicated to the repository. Ensure that the tablespace is not read-only.

> **See Also:**
>
> *Oracle Database PL/SQL Packages and Types Reference*