237
DBMS_XMLSCHEMA

`DBMS_XMLSCHEMA` package provides procedures to manage XML schemas.

It is created by script `dbmsxsch.sql` during Oracle database installation.

This chapter contains the following topics:

> **See Also:**
>
> *Oracle XML DB Developer's Guide*

DBMS_XMLSCHEMA Overview

The `DBMS_XMLSCHEMA` package uses subprograms to manage XML schemas.

These subprograms provide the following XML schema management::

- Register an XML schema
- Delete a previously registered XML schema
- Re-compile a previously registered XML schema
- Generate an XML schema
- Evolves an XML schema

DBMS_XMLSCHEMA Security Model

Owned by `XDB`, the `DBMS_XMLSCHEMA` package must be created by `SYS` or `XDB`. The `EXECUTE` privilege is granted to `PUBLIC`. Subprograms in this package are executed using the privileges of the current user.

DBMS_XMLSCHEMA Constants

The `DBMS_XMLSCHEMA` package defines several constants to use when specifying parameter values.

These constants are shown in following tables.

Table 237-1 *DBMS_XMLSCHEMA Constants* - Delete Option

Constant	Type	Value	Description
DELETE_RESTRICT	NUMBER	1	Deletion of an XML schema fails if there are any tables or XML schemas that depend on it
DELETE_INVALIDATE	NUMBER	2	Deletion of an XML schema does not fail if there are tables or XML schemas that depend on it. All dependent tables and schemas are invalidated.
DELETE_CASCADE	NUMBER	3	Deletion of an XML schema also drops all SQL types and default tables associated with it. SQL types are dropped only if gentypes argument was set to TRUE during registration of the XML schema. However, deletion of the XML schema fails if there are any instance documents conforming to the schema or any dependent XML schemas.
DELETE_CASCADE_FORCE	NUMBER	4	This option is similar to DELETE_CASCADE except that it does not check for any stored instance documents conforming to the schema or any dependent XML schemas. Also, it ignores any errors.

Table 237-2 *DBMS_XMLSCHEMA Constants* - Enable Hierarchy

Constant	Type	Value	Description
ENABLE_HIERARCHY_NONE	PLS_INTEGER	1	The ENABLE_HIERARCHY procedure of the DBMS_XDBZ package will not be called on any tables created while registering that schema
ENABLE_HIERARCHY_CONTENTS	PLS_INTEGER	2	The ENABLE_HIERARCHY procedure of the DBMS_XDBZ package will be called for all tables created during schema registration with hierarchy_type as DBMS_XDBZ.ENABLE_CONTENTS

Table 237-2 (Cont.) *DBMS_XMLSCHEMA Constants* - Enable Hierarchy

Constant	Type	Value	Description
ENABLE_HIERARCHY_RESMETADATA	PLS_INTEGER	3	The ENABLE_HIERARCHY procedure of the DBMS_XDBZ package will be called on all tables created during schema registration with hierarchy_type as DBMS_XDBZ.ENABLE_RESMETADATA. Users should pass in DBMS_XMLSCHEMA.ENABLE_RESMETADATA for schemas they intend to use as resource metadata tables.

Table 237-3 *DBMS_XMLSCHEMA Constants* - Register CSID

Constant	Type	Value	Description
REGISTER_NODOCID	NUMBER	1	If a schema is registered for metadata use (using the value ENABLE_HIER_RESMETADATA for parameter enablehierarchy during registration), a column named DOCID is added to all tables created during schema registration. This constant can be used in the options argument of REGISTERSCHEMA to prevent the creation of this column if the user wishes to optimize on storage
REGISTER_CSID_NULL	NUMBER	-1	If user wishes to not specify the character set of the input schema document when invoking REGISTERSCHEMA, this value can be used for the csid parameter

Views

This table lists the views used by the DBMS_XMLSCHEMA package

The columns of these views are described in detail in the *Oracle Database Reference*

Table 237-4 Summary of Views used by DBMS_XMLSCHEMA

Schema	Description
USER_XML_SCHEMAS	All registered XML Schemas owned by the user
ALL_XML_SCHEMAS	All registered XML Schemas usable by the current user
DBA_XML_SCHEMAS	All registered XML Schemas in the database
DBA_XML_TABLES	All XMLType tables in the system

Table 237-4 (Cont.) Summary of Views used by DBMS_XMLSCHEMA

Schema	Description
USER_XML_TABLES	All XMLType tables owned by the current user
ALL_XML_TABLES	All XMLType tables usable by the current user
DBA_XML_TAB_COLS	All XMLType table columns in the system
USER_XML_TAB_COLS	All XMLType table columns in tables owned by the current user
ALL_XML_TAB_COLS	All XMLType table columns in tables usable by the current user
DBA_XML_VIEWS	All XMLType views in the system
USER_XML_VIEWS	All XMlType views owned by the current user
ALL_XML_VIEWS	All XMLType views usable by the current user
DBA_XML_VIEW_COLS	All XMLType view columns in the system
USER_XML_VIEW_COLS	All XMLType view columns in views owned by the current user
ALL_XML_VIEW_COLS	All XMLType view columns in views usable by the current user

DBMS_XMLSCHEMA Operational Notes

There are guidelines for using in-place XML schema evolution.

Before you perform an in-place XML-schema evolution, you should follow these preparatory steps:

1. Back up all existing data (instance documents) for the XML schema that will be evolved.
2. Perform a dry run using trace only, that is, without actually evolving the XML schema or updating any instance documents, to produce a trace of the update operations that would be performed during evolution. To do this, set the flag parameter value to only INPLACE_TRACE. Do not also use INPLACE_EVOLVE. After performing the dry run, examine the trace file, verifying that the listed DDL operations are in fact those that you intend.

Summary of DBMS_XMLSCHEMA Subprograms

This table lists the DBMS_XMLSCHEMA subprograms and briefly describes them.

Table 237-5 DBMS_XMLSCHEMA Package Subprograms

Method	Description
COMPILESCHEMA Procedure	Used to re-compile an already registered XML schema. This is useful for bringing a schema in an invalid state to a valid state.
COPYEVOLVE Procedure	Evolves registered schemas so that existing XML instances remain valid
DELETESCHEMA Procedure	Removes the schema from the database
INPLACEEVOLVE Procedure	Evolves registered schemas by propagating schema changes to object types and tables
PURGESCHEMA Procedure	Removes the XML schema
REGISTERSCHEMA Procedures	Registers the specified schema for use by Oracle. This schema can then be used to store documents conforming to this.

Table 237-5 (Cont.) DBMS_XMLSCHEMA Package Subprograms

Method	Description
REGISTERURI Procedure	Registers an XML schema specified by a URI name

COMPILESCHEMA Procedure

This procedure can be used to re-compile an already registered XML schema. This is useful for bringing a schema in an invalid state to a valid state. Can result in a `ORA-31001` exception: invalid resource handle or path name.

Syntax

```
DBMS_XMLSCHEMA.COMPILESCHEMA(
   schemaurl IN VARCHAR2);
```

Parameters

Table 237-6 COMPILESCHEMA Procedure Parameters

Parameter	Description
`schemaurl`	URL identifying the schema

COPYEVOLVE Procedure

This procedure evolves registered schemas so that existing XML instances remain valid.

This procedure is accomplished in according to the following basic scenario (alternative actions are controlled by the procedure's parameters):

- copies data in schema based `XMLType` tables to temporary table storage
- drops old tables
- deletes old schemas
- registers new schemas
- creates new `XMLType` tables
- Populates new tables with data in temporary storage; auxiliary structures (constraints, triggers, indexes, and others) are not preserved
- drops temporary tables

> **See Also:**
>
> – "Schema Evolution" chapter of the *Oracle XML DB Developer's Guide* for examples on how to evolve existing schemas
>
> – *Oracle Database Error Messages* for information on exceptions specific to schema evolution, ORA-30142 through ORA-30946.

Syntax

```
DBMS_XMLSCHEMA.COPYEVOLVE(
   schemaurls       IN  XDB$STRUBG_LIST_T,
   newschemas       IN  XMLSequenceType,
   transforms       IN  XMLSequenceType :=NULL,
   preserveolddocs  IN  BOOLEAN :=FALSE,
   maptablename     IN  VARCHAR2 :=NULL,
   generatetables   IN  BOOLEAN :=TRUE,
   force            IN  BOOLEAN :=FALSE,
   schemaowners     IN  XDB$STRING_LIST_T :=NULL
   parallelDegree   IN  PLS_INTEGER := 0,
   options          IN  PLS_INTEGER := 0);
```

Parameters

Table 237-7 COPYEVOLVE Procedure Parameters

Parameter	Description
schemaurls	VARRAY of URLs of all schemas to be evolved. Should include the dependent schemas. Unless the FORCE parameter is TRUE, URLs should be in the order of dependency.
newschemas	VARRAY of new schema documents. Should be specified in same order as the corresponding URLs.
transforms	VARRAY of transforming XSL documents to be applied to schema-based documents. Should be specified in same order as the corresponding URLs. Optional if no transformations are required.
preserveolddocs	Default is FALSE, and temporary tables with old data are dropped. If TRUE, these table are still available after schema evolution is complete.
maptabname	Specifies the name of the table mapping permanent to temporary tables during the evolution process. Valid columns are: • SCHEMA_URL - VARCHAR2(700) - URL of schema to which this table conforms • SCHEMA_OWNER -VARCHAR2(30) - Owner of the schema • ELEMENT_NAME - VARCHAR2(256)- Element to which this table conforms • TAB_NAME - VARCHAR2(65) - Qualified table name: *<owner_name>.<table_name>* • COL_NAME - VARCHAR2(4000) - Name of the column (NULL for XMLType tables) • TEMP_TABNAME - VARCHAR2(30) - Name of temporary tables which holds data for this table.
generatetables	Default is TRUE, and new tables will be generated. If FALSE: • new tables will not be generated after registration of new schemas • preserveolddocs must be TRUE • maptablename must be non-NULL
force	Default is FALSE. If TRUE, ignores errors generated during schema evolution. Used when there are circular dependencies among schemas to ensure that all schemas are stored despite possible errors in registration.

Table 237-7 (Cont.) COPYEVOLVE Procedure Parameters

Parameter	Description
schemaowners	VARRAY of names of schema owners. Should be specified in same order as the corresponding URLs. Default is NULL, assuming that all schemas are owned by the current user.
paralleldegree	Specifies the degree of parallelism to be used in a PARALLEL hint during the data copy stage of the evolution. If this is 0 (default), the PARALLEL hint will not be given in the data copy statements.
options	Currently, the only supported option is COPYEVOLVE_BINARY_XML which lets you register the new schemas for binary XML and create the new tables/columns with binary XML as the storage type.

Usage Notes

You should back up all schemas and documents prior to invocation because COPYEVOLVE Procedure deletes all conforming documents prior to implementing the schema evolution.

DELETESCHEMA Procedure

This procedure deletes the XML Schema specified by the URL.

Syntax

```
DBMS_XMLSCHEMA.DELETESCHEMA(
   schemaurl      IN  VARCHAR2,
   delete_option  IN  PLS_INTEGER := DELETE_RESTRICT);
```

> **See Also:**
>
> "XMLSCHEMA Storage and Query: Basic" chapter of the *Oracle XML DB Developer's Guide*

Parameters

Table 237-8 DELETESCHEMA Procedure Parameters

Parameter	Description
schemaurl	URL identifying the schema to be deleted

Table 237-8 (Cont.) DELETESCHEMA Procedure Parameters

Parameter	Description
delete_option	Delete options: • DELETE_RESTRICT - Schema deletion fails if there are any tables or schemas that depend on this schema • DELETE_INVALIDATE - Schema deletion does not fail if there are any dependencies. Instead, it simply invalidates all dependent objects. • DELETE_CASCADE - Schema deletion will also drop all default SQL types and default tables. However the deletion fails if there are any stored instances conforming to this schema. • DELETE_CASCADE_FORCE - Similar to DELETE_CASCADE except that it does not check for any stored instances conforming to this schema. Also, it ignores any errors.

Exceptions

Table 237-9 DELETESCHEMA Procedure Exceptions

Exception	Description
ORA-31001	Invalid resource handle or path name

INPLACEEVOLVE Procedure

This procedure evolves registered schemas by propagating schema changes to object types and tables.

Syntax

```
DBMS_XMLSCHEMA.INPLACEEVOLVE(
   schemaURL     IN   VARCHAR2,
   diffXML       IN   XMLType,
   flags         IN   NUMBER);
```

Parameters

Table 237-10 INPLACEEVOLVE Procedure Parameters

Parameter	Description
schemaurl	URL of the schema to evolve
diffXML	Changes to be applied to the schema. This is an XML document conforming to the XDIFF schema and specifies what changes need to be applied and the locations in the schema document where the changes are to be applied.

Table 237-10 (Cont.) INPLACEEVOLVE Procedure Parameters

Parameter	Description
`flags`	The following bits may be set in this parameter to control the behavior of this procedure: • `INPLACE_EVOLVE` (value 1, meaning that bit 1 is on) – Perform in-place XML schema evolution: construct a new XML schema and validate it (against the XML schema for XML schemas); construct the DDL statements needed to evolve the instance-document disk structures, execute the DDL statements, and replace the old XML schema with the new. • `INPLACE_TRACE` (value 2, meaning that bit 2 is on) – Perform all steps necessary for in-place evolution, except executing the DDL statements and overwriting the old XML schema with the new, then write both the DDL statements and the new XML schema to a trace file. That is, each of the bits constructs the new XML schema, validates it, and determines the steps needed to evolve the disk structures underlying the instance documents. In addition: • Bit `INPLACE_EVOLVE` carries out those evolution steps and replaces the old XML schema with the new. • Bit `INPLACE_TRACE` saves the evolution steps and the new XML schema in a trace file (it does not carry out the evolution steps)

Exceptions

The procedure raises exceptions in the following cases:

- An error will be raised for invalid `XPATH` expressions and for `XDIFF` documents that do not conform to the xdiff schema.
- Path expressions that are syntactically correct but result in an invalid node in the schema document will result in an error.
- If the schema change makes the schema an ill-formed XML document or an invalid XML schema, this will raise an error.
- Any errors resulting from `CREATE TYPE`, `ALTER TYPE` and like commands will generate error messages.

Usage Notes

- Users are required to backup all their data before attempting in-place evolution, as there is no rollback with this operation.
- A user must register their new XML schema with the database using the REGISTERSCHEMA Procedures and the REGISTERURI Procedure at a schema URL that is different from that of the one to be evolved. If the new schema registers successfully and is usable, only then should the user attempt to evolve the existing schema to the new schema by means of this subprogram. If the registration of the new schema is successful, then the user must delete this schema (and all its dependent objects) before attempting to evolve the schema at the old schema URL.

PURGESCHEMA Procedure

This procedure removes the XML schema.

> **See Also:**
>
> "XMLSCHEMA Storage and Query: Advanced" chapter of the *Oracle XML DB Developer's Guide*

Syntax

```
DBMS_XMLSCHEMA.PURGESCHEMA(
   schemaid   IN  RAW);
```

Parameters

Table 237-11 PURGESCHEMA Procedure Parameters

Parameter	Description
`schemaid`	ID of the schema to be purged

Usage Notes

- The schema should have been originally registered for binary encoding and should have been deleted in the `HIDE` mode.
- Once a schema has been deleted in `HIDE` mode, it continues to exist in the XML DB dictionary and is used for decoding already encoded documents. The user invokes this interface when there are no stored instances encoded with this schema.
- Once the schema is purged, any space used by that schema will be reclaimed and documents encoded using the schema will raise an error if an attempt is made to decode them.
- The Schema ID can be obtained from the catalog views.

REGISTERSCHEMA Procedures

This procedure registers the specified schema for use by the database.

The procedure is overloaded. The different functionality of each form of syntax is presented along with the definition.

> **Note:**
>
> As of Oracle Database 11*g* Release 2 (11.2) the `genbean` parameter is deprecated. Oracle recommends that you do not use this parameter in new applications. Support for this feature is for backward compatibility only.

> **See Also:**
>
> "XMLSCHEMA Storage and Query: Basic" chapter of the *Oracle XML DB Developer's Guide*

Syntax

Registers a schema specified as a `VARCHAR2`:

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
    schemaurl        IN  VARCHAR2,
    schemadoc        IN  VARCHAR2,
    local            IN  BOOLEAN := TRUE,
    gentypes         IN  BOOLEAN := TRUE,
    genbean          IN  BOOLEAN := FALSE,
    gentables        IN  BOOLEAN := TRUE,
    force            IN  BOOLEAN := FALSE,
    owner            IN  VARCHAR2 := NULL,
    enablehierarchy  IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
    options          IN  PLS_INTEGER := 0);
```

Registers the schema specified as a `BFILE`. The contents of the schema document must be in the database character set:

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl        IN  VARCHAR2,
   schemadoc        IN  BFILE,
   local            IN  BOOLEAN := TRUE,
   gentypes         IN  BOOLEAN := TRUE,
   genbean          IN  BOOLEAN := FALSE,
   force            IN  BOOLEAN := FALSE,
   owner            IN  VARCHAR2 := NULL,
   enablehierarchy  IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
   options          IN  PLS_INTEGER := 0);
```

Registers the schema specified as a `BFILE` and identifies the character set id of the schema document:

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl        IN  VARCHAR2,
   schemadoc        IN  BFILE,
   local            IN  BOOLEAN := TRUE,
   gentypes         IN  BOOLEAN := TRUE,
   genbean          IN  BOOLEAN := TRUE,
   gentables        IN  BOOLEAN := TRUE,
   force            IN  BOOLEAN := TRUE,
   owner            IN  VARCHAR2 := '',
   csid             IN  NUMBER,
   enablehierarchy  IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
   options          IN  PLS_INTEGER := 0);
```

Registers the schema specified as a `BLOB`. The contents of the schema document must be in the database character set:

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl        IN  VARCHAR2,
```

```
   schemadoc         IN  BLOB,
   local             IN  BOOLEAN := TRUE,
   genTypes          IN  BOOLEAN := TRUE,
   genBean           IN  BOOLEAN := FASLE,
   force             IN  BOOLEAN := FALSE,
   owner             IN  VARCHAR2 := NULL,
   enablehierarchy   IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
   options           IN  PLS_INTEGER := 0);
```

Registers the schema specified as a `BLOB` and identifies the character set id of the schema document:

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl         IN  VARCHAR2,
   schemadoc         IN  BLOB,
   local             IN  BOOLEAN := TRUE,
   gentypes          IN  BOOLEAN := TRUE,
   genbean           IN  BOOLEAN := TRUE,
   gentables         IN  BOOLEAN := TRUE,
   force             IN  BOOLEAN := TRUE,
   owner             IN  VARCHAR2 := '',
   csid              IN  NUMBER,
   enablehierarchy   IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
   options           IN  PLS_INTEGER := 0);
```

Registers the schema specified as a `CLOB`

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl         IN  VARCHAR2,
   schemadoc         IN  CLOB,
   local             IN  BOOLEAN := TRUE,
   gentypes          IN  BOOLEAN := TRUE,
   genbean           IN  BOOLEAN := FALSE,
   force             IN  BOOLEAN := FALSE,
   owner             IN  VARCHAR2 := NULL,
   options           IN  PLS_INTEGER := 0);
```

Registers the schema specified as an `XMLTYPE`.

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl         IN  VARCHAR2,
   schemadoc         IN  SYS.XMLTYPE,
   local             IN  BOOLEAN := TRUE,
   gentypes          IN  BOOLEAN := TRUE,
   genbean           IN  BOOLEAN := FALSE,
   force             IN  BOOLEAN := FALSE,
   owner             IN  VARCHAR2 := NULL,
   enablehierarchy   IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
   options           IN  PLS_INTEGER := 0);
```

Registers the schema specified as a `BLOB`. The contents of the schema document must be in the database character set:

```
DBMS_XMLSCHEMA.REGISTERSCHEMA(
   schemaurl         IN  VARCHAR2,
   schemadoc         IN  SYS.URIType,
   local             IN  BOOLEAN := TRUE,
   gentypes          IN  BOOLEAN := TRUE,
```

```
  genbean          IN  BOOLEAN := FALSE,
  force            IN  BOOLEAN := FALSE,
  owner            IN  VARCHAR2 := NULL,
  enablehierarchy  IN  PLS_INTEGER := DBMS_XMLSCHEMA.ENABLE_CONTENTS,
  options          IN  PLS_INTEGER := 0);
```

Parameters

Table 237-12 REGSITERSCHEMA Procedure Parameters

Parameter	Description
schemaurl	URL that uniquely identifies the schema document. This value is used to derive the path name of the schema document within the database hierarchy. Can be used inside `schemalocation` attribute of XML Schema import element.
schemadoc	A valid XML schema document
local	Is this a local or global schema? • By default, all schemas are registered as local schemas, under `/sys/schemas/<username>/...` • If a schema is registered as global, it is added under `/sys/schemas/PUBLIC/...` You need write privileges on the directory to be able to register a schema as global.
gentypes	Determines whether the schema compiler generates object types. By default, `TRUE`. If you use binary XML, you must be set `gentypes` to `FALSE`.
genbean	Determines whether the schema compiler generates Java beans. By default, `FALSE`. Oracle recommends that this parameter always be set to `FALSE`.
gentables	Determines whether the schema compiler generates default tables. By default, `TRUE`
force	If this parameter is set to `TRUE`, the schema registration will not raise errors. Instead, it creates an invalid XML schema object in case of any errors. By default, the value of this parameter is `FALSE`.
owner	This parameter specifies the name of the database user owning the XML schema object. By default, the user registering the schema owns the XML schema object. This parameter can be used to register a XML schema to be owned by a different database user.
csid	Identifies the character set of the input schema document. If this value is `0`, the schema document's encoding is determined by the current rule for "text/xml" MIME type.
enablehierarchy	• `ENABLE_HIERARCHY_NONE` - enable hierarchy will not be called on any tables created while registering that schema • `ENABLE_HIERARCHY_CONTENTS` - enable hierarchy will be called for all tables created during schema registration with `hierarchy_type` as `DBMS_XDBZ.ENABLE_CONTENTS`. This is the default. • `ENABLE_HIERARCHY_RESMETADATA` - enable hierarchy will be called on all tables created during schema registration with `hierarchy_type` as `DBMS_XDBZ.ENABLE_RESMETADATA`. Users should pass in `DBMS_XMLSCHEMA.ENABLE_RESMETADATA` for schemas they intend to use as resource metadata tables.

Table 237-12 (Cont.) REGSITERSCHEMA Procedure Parameters

Parameter	Description
options	Additional options to specify how the schema should be registered. The various options are represented as bits of an integer and the options parameter should be constructed by doing a BITOR of the desired bits. Possible bits: • REGISTER_NODOCID - this will suppress the creation of the DOCID column for out of line tables. This is a storage optimization which might be desirable when we do not need to join back to the document table (for example if we do not care about rewriting certain queries that could be rewritten by making use of the DOCID column) • REGISTER_BINARYXML - Register the schema for Binary XML • REGISTER_NT_AS_IOT - Store nested tables created during schema registration as index organized tables. The default is to store nested tables as heap tables

REGISTERURI Procedure

This procedure registers an XML Schema specified by a URI name.

> **Note:**
>
> As of Oracle Database 11g Release 2 (11.2) the genbean parameter is deprecated. Oracle recommends that you do not use this parameter in new applications. Support for this feature is for backward compatibility only.

Syntax

```
DBMS_XMLSCHEMA.REGISTERURI(
   schemaurl      IN  VARCHAR2,
   schemadocuri   IN  VARCHAR2,
   local          IN  BOOLEAN := TRUE,
   gentypes       IN  BOOLEAN := TRUE,
   genbean        IN  BOOLEAN := FALSE,
   gentables      IN  BOOLEAN := TRUE,
   force          IN  BOOLEAN := FALSE,
   owner          IN  VARCHAR2 := NULL,
   options          IN  PLS_INTEGER := 0);
```

Parameters

Table 237-13 REGISTERURI Procedure Parameters

Parameter	Description
schemaurl	Uniquely identifies the schema document. Can be used inside schemaLocation attribute of XML Schema import element.

Table 237-13 (Cont.) REGISTERURI Procedure Parameters

Parameter	Description
schemadocuri	Pathname (URI) corresponding to the physical location of the schema document. The URI path could be based on HTTP, FTP, DB or Oracle XML DB protocols. This function constructs a URIType instance using the urifactory, and invokes the REGISTERSCHEMA Procedures.
local	Determines whether this is a local or global schema. By default, all schemas are registered as local schemas, under /sys/schemas/<username>/... If a schema is registered as global, it is added under /sys/schemas/PUBLIC/... The user needs write privileges on the directory to register a global schema.
gentypes	Determines whether the compiler generate object types. By default, TRUE.
genbean	Determines whether the compiler generate Java beans. By default, FALSE.
gentables	Determines whether the compiler generate default tables. TRUE by default.
force	TRUE: schema registration will not raise errors. Instead, it creates an invalid XML schema object in case of any errors. By default, the value of this parameter is FALSE.
owner	This parameter specifies the name of the database user owning the XML schema object. By default, the user registering the schema owns the XML schema object. This parameter can be used to register a XML schema to be owned by a different database user.
options	Additional options to specify how the schema should be registered. The various options are represented as bits of an integer and the options parameter should be constructed by doing a BITOR of the desired bits. Possible bits: • REGISTER_NODOCID - this will suppress the creation of the DOCID column for out of line tables. This is a storage optimization which might be desirable when we do not need to join back to the document table (for example if we do not care about rewriting certain queries that could be rewritten by making use of the DOCID column)