239

DBMS_XMLSCHEMA_UTIL

The DBMS_XMLSCHEMA_UTIL package provides an interface for XML schema validation.

This chapter contains the following topics:

- DBMS_XMLSCHEMA_UTIL Security Model
- Summary of DBMS_XMLSCHEMA_UTIL Subprograms

See Also:

- *Oracle XML DB Developer's Guide*

DBMS_XMLSCHEMA_UTIL Security Model

Applications must have the EXECUTE privilege on the DBMS_XMLSCHEMA_UTIL package

Summary of DBMS_XMLSCHEMA_UTIL Subprograms

This topic lists the DBMS_XMLSCHEMA_UTIL subprograms in alphabetical order and briefly describes them.

Table 239-1 ***DBMS_XMLSCHEMA_UTIL Package Subprograms***

Subprogram	Description
CONFORMING Functions	Validates an XML document against a single schema or an array of schemas. This function is overloaded.
VALIDATE Procedures	Validates an XML document against a single schema or an array of schemas. This procudere is overloaded.

CONFORMING Functions

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

Syntax

This function is used to validate an XML document against a schema:

```
DBMS_XMLSCHEMA_UTIL.CONFORMING (
   doc                IN XMLTYPE,
   sch                IN XMLTYPE)
RETURN NUMBER;
```

Note:

The function takes an xml data instance and xml schema instance. The user does not have to register the schema.

Syntax

This function is used to validate an XML document against multiple XML schema documents, as an array of XML schemas, each identified by the matching schema URI in the array URIs :

```
DBMS_XMLSCHEMA_UTIL.CONFORMING (
   doc              IN XMLTYPE,
   uris             IN XDB.XDB$STRING_LIST_T,
   schemas          IN sys.XMLSequenceType)
RETURN NUMBER;
```

Parameters

Table 239-2 CONFORMING Function Parameters

Parameter	Description
`doc`	The XML instance
`sch`	The XML schema
`schemas`	The XML schema array
`uris`	The schema URIs

Function Exceptions

The exceptions of `DBMS_XMLSCHEMA_UTIL.CONFORMING` function are as follows:

- The function returns zero if the schema is legal and the document conforms to the schema; otherwise it returns an LSX error code from the schema validation.
- The function raises ORA-31093, if `doc` is `NULL`, URIs or schemas is null or empty, or URIs and schemas are of different size.

Example: The document conforms to the schema

```
SELECT DBMS_XMLSCHEMA_UTIL.CONFORMING(
     XMLType('<A/>'),
     XMLType(
         '<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
          <xs:element name="A" type="xs:string"/>
          </xs:schema>')) "LSX CODE"
    FROM DUAL;
```

Example: The document conforms to schemas, and the conforming function returns zero

```
SQL> select
  2    dbms_xmlschema_util.conforming(
  3      xmltype(
  4       '<Company xmlns="http://www.oracle.com/company.xsd"
```

```
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://www.oracle.com/company.xsd company">
        <Name>Oracle</Name>
        <Employee>
          <EmployeeId>1001</EmployeeId>
          <Name>John Doe</Name>
        </Employee>
      </Company>'),
    XDB$STRING_LIST_T(
      'employee',
      'company'),
    XMLSequenceType(
      xmltype(
       '<schema xmlns="http://www.w3.org/2001/XMLSchema"
                xmlns:my="http://www.oracle.com/company.xsd"
                targetNamespace="http://www.oracle.com/company.xsd"
                elementFormDefault="qualified">
          <complexType name = "EmployeeType">
           <sequence>
            <element name = "EmployeeId" type = "positiveInteger"/>
            <element name = "Name" type = "string"/>
           </sequence>
          </complexType>
        </schema>'),
      xmltype(
       '<schema xmlns="http://www.w3.org/2001/XMLSchema"
                xmlns:my="http://www.oracle.com/company.xsd"
                targetNamespace="http://www.oracle.com/company.xsd"
                elementFormDefault="qualified">
          <include schemaLocation="employee"/>
          <element name = "Company">
            <complexType>
              <sequence>
                <element name = "Name" type = "string"/>
                <element name = "Employee" type = "my:EmployeeType"
                    maxOccurs = "unbounded"/>
              </sequence>
            </complexType>
          </element>
        </schema>'))) "LSX CODE"
from dual;

  LSX CODE
----------
         0

1 row selected.
```

VALIDATE Procedures

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

Syntax

This procedure is used to validate an XML document against a schema:

```
DBMS_XMLSCHEMA_UTIL.VALIDATE (
   doc             IN XMLTYPE,
   sch             IN XMLTYPE);
```

Note:

The procedure takes an xml data instance and xml schema instance. The user does not have to register the schema.

Syntax

This procedure is used to validate an XML document against multiple XML schema documents, as an array of XML schemas, each identified by the matching schema URI in the array URIs:

```
DBMS_XMLSCHEMA_UTIL.VALIDATE (
   doc              IN XMLTYPE,
   uris             IN XDB.XDB$STRING_LIST_T,
   schemas          IN sys.XMLSequenceType);
```

Parameters

Table 239-3 VALIDATE Procedure Parameters

Parameter	Description
`doc`	The XML instance
`sch`	The XML schema
`schemas`	The XML schema array
`uris`	The schema URIs

Procedure Exceptions

The exceptions of `DBMS_XMLSCHEMA_UTIL.VALIDATE` procedure are as follows:

- The procedure raises ORA-31154 if either the schema is not legal, or the document does not conform to the schema.

Example: The document conforms to the schema

```
BEGIN
      DBMS_XMLSCHEMA_UTIL.VALIDATE(
         XMLType('<A/>'),
         XMLType(
                 '<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
                  <xs:element name="A" type="xs:string"/>
                  </xs:schema>'));
END;
/
```

Example: The document conforms to the schemas

```
SQL> begin
  2    dbms_xmlschema_util.validate(
  3      xmltype(
  4       '<Company xmlns="http://www.oracle.com/company.xsd"
  5                 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  6                 xsi:schemaLocation="http://www.oracle.com/company.xsd company">
```

```
        <Name>Oracle</Name>
        <Employee>
          <EmployeeId>1001</EmployeeId>
          <Name>John Doe</Name>
        </Employee>
      </Company>'),
    XDB$STRING_LIST_T(
      'employee',
      'company'),
    XMLSequenceType(
      xmltype(
       '<schema xmlns="http://www.w3.org/2001/XMLSchema"
                xmlns:my="http://www.oracle.com/company.xsd"
                targetNamespace="http://www.oracle.com/company.xsd"
                elementFormDefault="qualified">
          <complexType name = "EmployeeType">
           <sequence>
            <element name = "EmployeeId" type = "positiveInteger"/>
            <element name = "Name" type = "string"/>
           </sequence>
          </complexType>
        </schema>'),
      xmltype(
       '<schema xmlns="http://www.w3.org/2001/XMLSchema"
                xmlns:my="http://www.oracle.com/company.xsd"
                targetNamespace="http://www.oracle.com/company.xsd"
                elementFormDefault="qualified">
          <include schemaLocation="employee"/>
          <element name = "Company">
            <complexType>
              <sequence>
                <element name = "Name" type = "string"/>
                <element name = "Employee" type = "my:EmployeeType"
                    maxOccurs = "unbounded"/>
              </sequence>
            </complexType>
          </element>
        </schema>')));
end;
/

PL/SQL procedure successfully completed.
```