

1. Introduction à XMLType

Objectif : pouvoir stocker une donnée XML (arborescente) dans une BDD relationnelle en profitant de la puissance des 2 modèles (XML/XPATH/XSD/XSL et SQL)

Documentation : http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14259/partpg2.htm

2. Utilisation

2.1 Déclaration

- déclarer un champs contenant un donnée XML : type XMLTYPE

```
CREATE TABLE ma_table (
    champs_xml XMLTYPE
);
```

2.2 Insertion

- insérer une donnée de type XML : constructeur XMLTYPE(s) avec s de type VARCHAR2 ou CLOB

- o donnée courte

```
INSERT INTO ma_table (champs_xml) VALUES (XMLTYPE (
    '<?xml version="1.0" encoding="UTF-8" ?>
    <racine>...</racine>'));
```

- o donnée longue

```
DECLARE
    xml_string CLOB ;
BEGIN
    xml_string := '<?xml version="1.0" encoding="UTF-8" ?>
    <racine>...</racine>';
    Insert into maTable values(xmltype(xml_string));
END;
/
```

2.3 Sélection

- accès au xml complet : select classique

NB : pour sqlplus régler le nombre de caractères à visualiser pour un type CLOB/XMLTYPE

```
SET LONG 2000
```

- convertisseurs : méthodes "objets"

- o maDonneeXML.getClobVal() : XML extrait en tant que CLOB
- o maDonneeXML.getStringVal() : XML extrait en tant que VARCHAR2 (taille limite)
- o maDonneeXML.getNumberVal() : XML extrait en tant que nombre (élément simple ou attribut)

- fonctions d'accès

- o existsnode(maDonneeXML, XPATHExpression) : prédicat à utiliser dans un WHERE uniquement, renvoie 1 si vrai, 0 sinon

- `extract (maDonneeXML, XPATHExpression)` : à utiliser dans les champs projetés (SELECT) ou dans un prédicat (WHERE) ; renvoie le nœud de type XMLTYPE correspondant à l'expression XPATH
- `extractvalue (maDonneeXML, XPATHExpression)` équivaut à `extract (maDonneeXML, XPATHExpression).getStringVal()`
Pour un élément simple `<elt>un texte</elt>`, il n'est pas nécessaire de préciser le nœud fils, de type texte, dans l'expression XPATH (cf XSL et value-of) : `elt/text()` → `elt` suffit pour extraire 'un texte'

2.4 Modification

- fonctions de modification partielles (sinon update classique) à combiner avec un UPDATE pour modification dans une table ou sans pour obtenir simplement un donnée modifiée :
 - `updateXML(donneeXML, XPATHExpr, value)` : renvoie `donneeXML` modifiée (si `value=NULL` => suppression du nœud)
 - `insertChildXML(donneeXML, XPATHExpr, child_name, child_value)`
 - pour un élément : `child_name` est le nom de l'élément, `child_value` l'élément complet
 - pour un attribut : `child_name` est le nom de l'attribut, `child_value` sa valeur
 - `insertXMLbefore(donneeXML, XPATHExpr, value)` : pour les éléments uniquement
 - `appendChildXML(donneeXML, XPATHExpr, value)`
 - `deleteXML(donneeXML, XPATHExpr)`

2.5 Transformation

transformation d'une donnée XML : fonction SQL

- fonction SQL : `XMLTransform(donneeXML, feuilleXSL)` : renvoie la donnée transformée
- méthode objet : `donneeXML.transform(feuilleXSL)`
- Il peut être intéressant de stocker les feuilles de style dans une table

2.6 Validation par schema

1. DBMS_XMLSCHEMA. deleteSchema
2. DBMS_XMLSCHEMA. registerSchema
3. associer un schema à une colonne d'une table
 - vérification auto à chaque modif (insert/update/delete)
4. PL/SQL Functions
 - `XMLIsValid(donnéeXML [,schemaURL [, élément]])`
 - `donnéeXML.isSchemaValidated()` : number (1 = true)
 - `donnéeXML.isSchemaValid([schemaURL],[élément])` : NUMBER (1 = true)
5. PL/SQL procedure
 - `donnéeXML.schemaValidate`
 - `donnéeXML.setSchemaValidated(flag)` (1 = true)

2.7 Indexation

Il est possible de définir un index sur nœud d'un document XML

Rédigé par : Matthias Colin et Hervé de Milleville

Ref : *ING2-GSI-TRA-DON-XMLTYPE*

A l'intention de : Etudiants des ING2-GSI

Créé le : 07/02/2012