Adatbázisok II.

7

Jánosi-Rancz Katalin Tünde
tsuto@ms.sapientia.ro
327A

Oracle XML

Oracle XML DB

- Az XML adatok kezelésére az Oracle egy külön komponenst készített, az Oracle XML DB-t.
- Az XML adatok tárolására az XMLType típust fejlesztették ki.
- Az XMLType típus tulajdonképpen egy objektumtípus. Jellemzői:
  - tábla és tábla oszlopa is lehet XMLType típusú,
  - ugyanúgy használható, mint bármelyik másik típus, pl. szerepelhet PL/SQL eljárás paramétereként, függvény visszatérési értékeként stb.,
  - csak jól formázott (well-formed) XML dokumentumok lehetnek ilyen típusúak,
  - legfontosabb metódusai: extract(), extractValue(), existsNode(), xmlSequence(), updateXML(), ezek a függvények azonban önállóan is léteznek.
- Alapesetben az XML dokumentumok CLOB-ként (Character Large Object) tárolódnak.

Oracle XML

XML támogatás **Leírás**

XMLType Új rendszer definiálta adattípus, amely mind relációs táblák oszlopainak típusaként, mind PL/SQL argumentumok típusaként is megadható.

DBMS_XMLGEN PL/SQL package amely SQL lekérdezések eredményeit konvertálja kanonikus XML formára és vagy XMLType típusú, vagy CLOB típusú objektumként adja vissza. A DBMS_XMLGEN package C-ben készült, és része a kernelnek, funkcionalitásában hasonló a DBMS_XMLQuery csomaghoz.

SYS_XMLGEN Egy rendszer definiálta függvény, amely SQL lekérdezéseken belül generál XMLType típushoz tartozó adatot. DBMS_XMLGEN és más hasonló csomagok a lekérdezések szintjén operálnak és a lekérdezés teljes eredményéből állítanak elő egyesített XML adatot, SYS_XMLGEN pedig a lekérdezésen belül egy argumentumból generálnak XML-t.

SYS_XMLGEN skaláris értékekből, objektum típusból, vagy XMLType típusú adatokból állít elő XML dokumentumot. Eközben opcionálisan használ egy XMLGenFormatType objektumot, amely formattálási opciókat ír elő az eredményre.

SYS_XMLGEN XMLType típusú adatot hoz létre.

SYS_XMLAGG Rendszer által definiált „összegzési függvény”, amely XMLType típusú adathalmaz fölött operál: egyesíti (egymáshoz fűzi) az összes input XML dokumentumokat/dokumentum-részeket egyetlen XML dokumentummá egy gyökértag hozzáadásával.

XMLType függvények (oracle 9)

Függ-vény	Prototípus	Leírás
CreateXML()	STATIC FUNCTION createXML(xmlval IN varchar2) RETURN sys. XMLType	Statikus függvény, amely karakterláncot konvertál XML dokumentummá.A függvény ellenőrzi, hogy a bemenő karakterlánc jól formált XML dokumentum-e. Paramétere: xmlval (IN) - az XML dokumentumot tartalmazó karakterlánc (jól formált XML dokumentumnak kell lennie). RETURNS: egy sys.xmltype tipusú értéket. Determinisztikus függvény.
CreateXML()	STATIC FUNCTION CreateXML(xmlval IN clob) RETURN sys. XMLType	Statikus függvény, amely CLOB objektumot konvertál XML dokumentummá. A függvény ellenőrzi, hogy a bemenő objektum jól formált XML dokumentum-e. Paramétere: xmlval (IN) - az XML dokumentumot tartalmazó objektum (jól formált XML dokumentumnak kell lennie). RETURNS: egy sys.xmltype tipusú értéket. Determinisztikus függvény.
ExistsNode()	MEMBER FUNCTION ExistsNode(xpath IN varchar2) RETURN number	A megadott Xpath kifejezésről eldönti, hogy eredménye az adott dokumentum valamilyen érvényes csomópontje-e. PARAMETERS: xpath (IN) – a vizsgálandó Xpath kifejezés. RETURNS: 0, ha a vizsgált kifejezés nem jelöl ki az adott dokumentumban csomópontot, minden más esetben 1, hacsak a vizsgált Xpath karakterlánc nem null értékű, vagy az adott dokumentum nem üres. Az utóbbi két esetben a visszaadott érték 0. Determinisztikus függvény.
Extract()	MEMBER FUNCTION extract(xpath IN varchar2) RETURN sys. XMLType	A megadott Xpath kifejezést a dokumentumra alkalmazva visszaadja Az Xpath kifejezés értékét, mint sys.xmltype tipusú értéket, amely már nem dokumentumnak (csak részdokumentum) tekintendő akkor sem, ha a teljes dokumentumot tartalmazza. PARAMETERS: xpath (IN) – a kiértékelendő Xpath kifejezés. RETURNS: Egy sys.xmltype tipusú rész-dokumentum. Ha az Xpath kifejezés nem ad vissza semmilyen csomópontokat, a függvény null értékkel tér vissza. Determinisztikus függvény.

IsFragment()	MEMBER FUNCTION IsFragment() RETURN number	Ellenőrzi, hogy az adott sys.xmltype típusú érték valoban részdokumentum-e. A részdokumentum fogalma nincs pontosan definiálva (egy teljes dokumentum mindig részdokumeentuma saját magának). RETURNS: Egy numerikus 1 vagy 0 értéket aszerint, hogy a sys.xmltype típusú érték részdokumentum vagy jól formált teljes dokumentum. Nem determinisztikus függvény, vagyis a részdokumentum és a jól formált dokumentum mint sys.xmltype típushoz tartozó részhalmazok nem alkotják a sys.xmltype típus egy particióját.
GetClobVal()	MEMBER FUNCTION getClobVal() RETURN clob	A sys.xmltype típusú értékből egy CLOB objektumot állit elő. RETURNS: Egy CLOB objektum, amely tartalmazza a szerializált XML reprezentációt. Az érték felhasználása után a temporálisan létrehozott CLOB objektumot fölszabaditja. Determinisztikus függvény.
GetStringVal()	MEMBER FUNCTION GetStringVal() RETURN varchar2	A sys.xmltype típusú értékből stringet állit elő. RETURNS: A karakrakterlánc, amely tartalmazza a szerializált XML reprezentációt vagy szöveg csomópontok (text node) esetén magát a szöveget. Ha az XML dokumentum string reprezentációja hosszabb, mint a VARCHAR2 tipus maximális hossza (4000 byte), egy run-time hiba keletkezik. Determinisztikus függvény.
GetNumberVal()	MEMBER FUNCTION GetNumberVal() RETURN number	A numerikus karaktereket tartalmazó sys.xmltype típusú adat számértékét adaja vissza. RETURNS: A numerikus érétket tartalmazó, sys.xmltype típusú érvényes szöveg csomópont (text node) number típusú értéke. Determinisztikus függvény.

Példa XMLType oszlop létrehozására

```
CREATE TABLE raktar
(
  raktar_azon     NUMBER(3),
  raktar_spec     SYS.XMLTYPE,
  raktar_nev      VARCHAR2(35 BYTE),
  epulet_azon     NUMBER(4)
);
```

Beszúrás a táblába:

```
INSERT INTO raktar VALUES(1,
SYS.XMLType.CreateXML(
'<?xml version="1.0"?>
 <Raktar rNo="100">
  <Tulaj>Sajat</Tulaj>
  <Terulet>25000</Terulet>
  <Folyoviz>Igen</Folyoviz>
  <Vasut>Nem</Vasut>
  <Parkolas>Utca</Parkolas>
 </Raktar>'),
'raktar1', '1000');
```

```
INSERT INTO raktar VALUES(2,
 SYS.XMLType.CreateXML(
'<?xml version="1.0"?>
 <Raktar>
  <Tulaj>Berelt</Tulaj>
  <Terulet>20000</Terulet>
  <Folyoviz>Nem</Folyoviz>
  <Vasut>Nem</Vasut>
  <Parkolas>Garazs</Parkolas>
 </Raktar>'),
'raktar2', '2000');
```

Lekérdezés

EXTRACT(XMLType, Xpath) - Az XPath kifejezés által megjelölt elemeket (csomópontokat, részfákat) adja vissza dokumentumtöredékként (oracle10-tol).

Fontos a kisbetû/nagybetû megkülönböztetése. Ha kisbetûvel írnánk pl. a 'tulaj'-t, akkor nem adna vissza semmit.

```
SELECT r.raktar_azon, r.raktar_spec.EXTRACT('/Raktar/Tulaj')
Eredmeny
FROM raktar r;
```

```
RAKTAR_AZON          EREDMENY
----------- --------------------------------------------------
          1                <Tulaj>Sajat</Tulaj>
          2                <Tulaj>Berelt</Tulaj>
```

Egy XML dokumentumot karakter típusú adattá konvertálhatunk a getStringVal tagfüggvénnyel:(vagy CLOB-bá a getCLobVal függvénnyel)

```
SELECT r.raktar_azon, r.raktar_spec.EXTRACT('/Raktar/Terulet').getStringval()
"Eredmeny"
FROM raktar r WHERE r.raktar_azon=1
```

```
RAKTAR_AZON          Eredmeny
-----------          -----------------
      1          <Terulet>25000</Terulet>
```

Lekérdezés:

- Ha a nyitó és záró tag-ekre nincs szükségünk, magát a tartamát is megkaphatjuk a csomópontnak a text() függvény segítségével.

```
SELECT r.raktar_azon,
r.raktar_spec.EXTRACT('/Raktar/Tulaj/text()') "Eredmeny"
FROM raktar r;
```

```
RAKTAR_AZON      Eredmeny
---------- ------------------------------------------------
         1          Sajat
         2          Berelt
```

- ```
  SELECT r.raktar_azon,
  r.raktar_spec.extract('/Raktar/Tulaj/text()').getStringval() "Eredmeny",
         r.raktar_spec.extract('/Raktar/Terulet/text()').getNumberval()
  "Eredmeny2"
  FROM raktar r WHERE r.raktar_azon=1;

  RAKTAR_AZON      Eredmeny             Eredmeny2
  -----------      -------------        ----------
            1          Sajat                25000
  ```
  ```

Törlés, módosítás

```
UPDATE raktar SET raktar_azon =
          sys.XMLType.createXML(
                   '<Raktar rNo="100">
                      <Tulaj>Sajat</Tulaj>
                    </Raktar>'));
```

- ```
  DELETE FROM raktar r
  WHERE r.raktar_azon.extract('//Tulaj/text()').getStringVal()
          = 'Sajat';
  ```
  ```

XMLSequence()

- Az extract() függvény sok esetben dokumentum helyett dokumentum-töredékeket (document fragments) ad vissza, vagyis olyan XML elemeket, amelyeket „nem fog össze” egy közös gyökér, hanem függetlenek egymástól.
- Az XMLSequence() függvény minden egyes ilyen fragmentet XMLType típusú objektummá alakít, majd veszi ezek kollekcióját.
- A table() függvénnyel aztán a kollekció virtuális táblává alakítható.

SYS_XMLGEN(kif [,fmt]) függvény

- XML dokumentumot hoz létre a paraméterül kapott értékből.

```
SELECT SYS_XMLGEN(dnev) FROM dolgozo WHERE ROWNUM < 3;
```

```
<?xml version="1.0"?>
<DNEV>SMITH</DNEV>
<?xml version="1.0"?>
<DNEV>ALLEN</DNEV>
```

```
SELECT SYS_XMLGEN(EXTRACT(kolcs_spec, '//DVD')).getStringVal() FROM
kolcsonzes WHERE azon=1;

--------------------------------
<?xml version="1.0"?>
<ROW>
  <DVD ar="4000">Jegkorszak</DVD>
  <DVD ar="3500">Shrek</DVD>
  <DVD ar="2500">Uvegtigris</DVD>
</ROW>
```

XSYS_XMLAGG(kif [,fmt]) függvény

A kifejezés által meghatározott dokumentumokból vagy töredékekbõl (az összes sorból) egy XML dokumentumot hoz létre. Ennek is lehet formázó objektum paramétert megadni.

```
SELECT SYS_XMLAGG(SYS_XMLGEN(dnev)).getStringVal() FROM dolgozo
WHERE ROWNUM < 3;
---------------------
<?xml version="1.0"?>
<ROWSET>
  <DNEV>SMITH</DNEV>
  <DNEV>ALLEN</DNEV>
</ROWSET>
```

Vagy ugyanez formázás megadásával:

```
SELECT SYS_XMLAGG(SYS_XMLGEN(dnev),

SYS.XMLGenFormatType.CreateFormat('SOROK')).getStringVal() FROM
dolgozo WHERE ROWNUM < 3;
---------------------
<?xml version="1.0"?>
<SOROK>
  <DNEV>SMITH</DNEV>
  <DNEV>ALLEN</DNEV>
</SOROK>
```

XMLQUERY függvény

- **XMLQUERY**(*XQuery_string* [*XML_passing_clause*] **RETURNING CONTENT**)
  - *XQuery_string* egy teljes XQuery kifejezés
  - Az *XML_passing_clause* a **PASSING** kulcsszóból és az azt követő XMLType objektum előfordulásokat előállító SQL kifejezésekből áll. Legfeljebb egy kifejezést kivéve, minden kifejezést névvel kell ellátni az **AS** SQL záradékkal. Az AS-t követő névnek XQuery azonosítónak kell lennie. A kifejezések kiértékelésekor kapott értékek az AS záradékban megadott névhez kapcsolódnak. Az AS záradékkal el nem látott legfeljebb egyetlen kifejezés értéke ún. *környezet cikkely* (context item) szerepét tölti be az XQuery kifejezés kiértékelésekor.
  - RETURNING CONTENT azt jelzi, hogy az *XQuery_string* kiértékelésével kapott eredményt nem XQuery sorozatként, hanem részdokumentumként akarjuk megkapni. Használata kötelező. Az ORACLE nem támogatja a RETURNING SEQUENCE záradéknak megfelelő lehetőséget.

```
SELECT XMLQuery('(1, 2 + 3, "a", 100 to 102, <A>33</A>)'
       RETURNING CONTENT) AS output
FROM DUAL;

Eredmény:
OUTPUT
-------------------------
1 5 a 100 101 102<A>33</A>
```

Példa: XMLQuery

- ```
  SELECT XMLQuery ('for $i in (2, 3, 4), $j in ($i+5, 2)
       return ($i, $j)'
           returning content).getStringval()
    FROM dual;
  ```

- ```
  SELECT XMLQuery ('for $i in /Kolcsonzesek
                       let $j:=$i/Kolcsonzo/@nev
                       return count($j)'
           passing kolcs_spec returning content).getStringval()
    FROM kolcsonzes WHERE azon=1;
  ```

- (Adjuk meg a kölcsönzõk nevét és az általuk kölcsönzött CD-k árának összegét)

  ```
  SELECT XMLQuery('for $k in //Kolcsonzo let $ar :=
  sum($k//CD/Ar/text())
                     return ($k/@nev, $ar)'
           PASSING kolcs_spec RETURNING CONTENT) AS oszlop
    FROM kolcsonzes WHERE azon=1;
  ```

Oracle XQuery függvények

- Az *ora* névtérben érvényes függvény: az **ora:view**

```
ora:view ([db-schema STRING,] db-table STRING)     RETURNS document-
node(element())*
```

Az **ora:view** függvény segítségével lehetőség van létező adatbázis táblákra/nézetekre mint XML dokomentumokra vonatkozó, XQuery kifejezések belsejében levő lekérdezések használatára. Az **ora:view** a lekérdezés kiértékelése közben létrehozza a megadott relációs tábla XML nézetét.

db-schema – Egy string konstans, amely egy létező adatbázis séma neve.
db-table – Egy string konstans, amely egy *db-schema*-beli létező adatbázis tábla vagy nézet neve.

Az **ora:view** függvény visszatérési értéke az adatbázis tábla soraihoz előállított Element típusú dokumentum csúcsok egy rendezetlen, esetleg üres sorozata. A relációs tábla minden sorára a DBMS_XMLGEN csomaghoz hasonló XML nézetet generál (oszlopnevek elemtípus névként, a sorok a <ROW>…</ROW> elemtípus előfordulásként jelennek meg).

Példa: Ora:view

```
SELECT XMLQuery('for $i in ora:view("HR", "REGIONS"),
                        $j in  ora:view("HR", "COUNTRIES")
                           where $i/ROW/REGION_ID = $j/ROW/REGION_ID and
                                 $i/ROW/REGION_NAME = "Asia"
                                                return $j'
                  RETURNING CONTENT) AS asian_countries
        FROM DUAL;
```

Eredmény:

```
ASIAN_COUNTRIES
--------------------------
<ROW>
  <COUNTRY_ID>AU</COUNTRY_ID>
  <COUNTRY_NAME>Australia</COUNTRY_NAME>
  <REGION_ID>3</REGION_ID>
</ROW>
<ROW>
  <COUNTRY_ID>CN</COUNTRY_ID>
  <COUNTRY_NAME>China</COUNTRY_NAME>
  <REGION_ID>3</REGION_ID>
</ROW>
```

Feladatok 1

```
CREATE TABLE kolcsonzes(azon NUMBER(4), kolcs_spec SYS.XMLType);
```

```
INSERT INTO kolcsonzes VALUES(1,
SYS.XMLType.CreateXML(
'<Kolcsonzesek>
   <Kolcsonzo nev="Gipsz Jakab">
     <Konyvek>
       <Konyv cim="Momo" szerzo="Michael Ende">
         <Ar>2000</Ar>
       </Konyv>
       <Konyv cim="Tuskevar" szerzo="Fekete
             Istvan">
         <Ar>2500</Ar>
       </Konyv>
     </Konyvek>
     <CD-k>
       <CD eloado="Zoran">
         <Cim>Szep Holnap</Cim>
         <Ar>3000</Ar>
       </CD>
     </CD-k>
     <DVD-k>
       <DVD ar="4000">Jegkorszak</DVD>
     </DVD-k>
   </Kolcsonzo>
   <Kolcsonzo nev="Kis Virag">
        <CD-k>
          <CD eloado="ABBA">
            <Cim>Best Of</Cim>
            <Ar>2000</Ar>
          </CD>
          <CD eloado="Zoran">
            <Cim>Igy alakult</Cim>
            <Ar>3200</Ar>
          </CD>
        </CD-k>
        <DVD-k>
          <DVD ar="3500">Shrek</DVD>
        </DVD-k>
      </Kolcsonzo>
      <Kolcsonzo nev="Nagy Pal">
        <DVD-k>
          <DVD ar="2500">Uvegtigris</DVD>
        </DVD-k>
      </Kolcsonzo>
 </Kolcsonzesek>'));
```

Feladatok 1

- 1. Adjuk meg a teljes dokumentumot

  a. `SELECT EXTRACT(kolcs_spec, '/').getStringVal() FROM kolcsonzes;`

  b. `SELECT EXTRACT(kolcs_spec, '*').getStringVal() FROM kolcsonzes;`
  (A gyökérnek az egyetlen gyermek csomópontja maga a teljes dokumentum.)

  c. `SELECT EXTRACT(kolcs_spec, '/*').getStringVal() FROM kolcsonzes;`
  (A teljes dokumentum a gyökérnek gyermeke.)

  A fenti lekérdezések nem formázzák meg az eredményt, hanem string típusúként adják azt vissza. Ezért abból nem is látszik a fa szerkezete. Ha az EXTRACT tagfüggvényt (MEMBER) használjuk az alábbi módon, az megformázza az outputját.

FORMAZVA:

```
SELECT k.kolcs_spec.EXTRACT('/').getStringVal() FROM kolcsonzes k;
```

-- Adjuk meg az összes csúcsot (az összes létezõ részfát)

Feladatok 2

- **2.** Adjuk meg az elsõ kölcsönzõ nevét.

  Gipsz Jakab
- **3.** Adjuk meg azokat a csomópontokat, amelyeknek van 'ar' attribútuma
  ---
  <DVD ar="4000">Jegkorszak</DVD><DVD ar="3500">Shrek</DVD><DVD ar="2500">Uvegtigris</DVD>
- **4.** Adjuk meg a DVD csomópontok szöveges tartalmát nyitó és záró tagek nélkül. Ebben az esetben az EXTRACT a szöveges tartalmakat összevonja egyetlen szöveges dokumentumtöredékké.

  JegkorszakShrekUvegtigris
- **5.** Adjuk meg az összes elõadót

  ZoranABBAZoran
- **6.** Adjuk meg a 2600-nál olcsóbb vagy 3800-nál drágább DVD-ket
  ---
  <DVD ar="4000">Jegkorszak</DVD><DVD ar="2500">Uvegtigris</DVD>
- **7.** Adjuk meg azokat a csomópontokat, amelyeknek van 'eloado' vagy 'szerzo' attributuma
- **8.** Adjuk meg azokat a csomópontokat, amelyeknek van 'cim' és 'szerzo' attributuma
- **9.** Adjuk meg azokat a csomópontokat, amelyeknek nincs attribútuma
- **10**. Adjuk meg a Shrek DVD csomópontját

  <DVD-k><DVD ar="3500">Shrek</DVD></DVD-k>

Feladatok 1

- **11**. Adjuk meg a 'Szep Holnap' címû CD árát

  <Ar>3000</Ar>
- **12.** Adjuk meg azoknak a CD-knek a címeit, amelyeknek ára legalább 3000

  <Cim>Szep Holnap</Cim><Cim>Igy alakult</Cim>
- **13.** Adjuk meg a 'Shrek' DVD árát

  3500
- **14.** Adjuk meg a Gipsz Jakab által kölcsönzött konyvek szerzõit

  ---

  Michael EndeFekete Istvan
- **15.** Adjuk meg azoknak a CD-knek az elõadóit, amelyeket nem Gipsz Jakab kölcsönzött
- **16.** Adjuk meg azoknak a nevét, akik kölcsönöztek Zorán CD-t

  ---

  Gipsz JakabKis Virag
- **17.** Adjuk meg azok nevét, akik kölcsönöztek könyvet

  Gipsz Jakab
- **18.** Adjuk meg azok nevét, akik kölcsönöztek könyvet vagy CD-t

  ---

  Gipsz JakabKis Virag

Feladatok 2

- A levelezes tábla felett adjuk meg a következő lekérdezéseket.
  - Adjuk meg Melák kiknek küldött email-t.
  - Adjuk meg, hogy Luca összesen hány email-t küldött.
  - Adjuk meg ki írt olyan email-t, amelyben szerepel a Melak szó.
  - Adjuk meg Luca válaszként küldött email-jeinek (subject RE:-vel kezdődik) szövegét.
  - Adjuk meg azon email-ek szövegét, amelyekre választ is küldtek. A válaszok szövegét is adjuk meg.

Feladatok 2.

- 1.SELECT extractValue(value(d), '/email/cimzett'),
  extractValue(value(d), '/email/subject')
  FROM levelezes k,
  table (XMLSequence(extract(k.emailek, '/emailek/email'))) d
  WHERE felhasznalo='melak';
- 2. SELECT count(*)
  FROM levelezes k,
  table (XMLSequence(extract(k.emailek, '/emailek/email'))) d
  WHERE felhasznalo='luca';
- 3. SELECT DISTINCT felhasznalo
  FROM levelezes k,
  table (XMLSequence(extract(k.emailek, '/emailek/email'))) d
  WHERE extractValue(value(d), '/email/szoveg') like '%Melak%';
- 4. SELECT extractValue(value(d), '/email/szoveg')
  FROM levelezes k,
  table (XMLSequence(extract(k.emailek, '/emailek/email'))) d
  WHERE felhasznalo='luca' AND extractValue(value(d),
  '/email/subject') like 'RE:%';