

Objektum-relációs adatbázisok – 3. rész

Felhasználói típusok (User-Defined Types)

Objektum ID-k

Beágyazott táblák (Nested Tables)

Oracle beágyazott táblák (*nested tables*)

- Megengedi, hogy a sorok egyes komponensei teljes relációk legyenek.
- Ha T egy UDT, létrehozhatunk egy S típust, amelynek az értékei relációk, amelyeknek a sortípusa viszont T :

```
CREATE TYPE S AS TABLE OF T;
```

Példa: beágyazott tábla típusok létrehozása

```
CREATE TYPE TeaTípus AS OBJECT (  
    név          CHAR(20) ,  
    fajta        CHAR(10) ,  
    szín         CHAR(10)  
);
```

```
CREATE TYPE TeaTáblaTípus AS  
    TABLE OF TeaTípus;
```

Példa -- folytatása

- **TeaTáblaTípus-t** használjuk **Gyártók** relációban, amelyik a teák gyártóit tárolja, mindegyik gyártó egy sorban.

```
CREATE TABLE Gyártók (  
    név          CHAR(30) ,  
    cím          CHAR(50) ,  
    teak         TeaTáblaTípus  
);
```



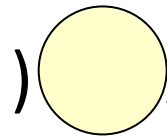
Ez így még nem lesz jó!
Ld. később a helyes szintaxist!

A beágyazott relációk eltárolása

- Oracle valójában nem tárolja el a beágyazott relációkat külön relációkként – még ha így is tűnik.
- Ehelyett, egy *R* reláció van, amelyben egy *A* attribútumra az összes beágyazott táblázatot és azok összes sorát eltárolja.
- Deklaráció a CREATE TABLE-ben:
NESTED TABLE *A* STORE AS *R*

Példa: Beágyazott táblák tárolása

```
CREATE TABLE Gyártók (  
    név CHAR(30),  
    cím CHAR(50),  
    teák TeaTáblaTípus
```



```
NESTED TABLE teák STORE AS TeaTábla;
```



A pontosvessző (;) vessző használatára
figyelni!

Beágyazott táblák lekérdezése

- Bármely beágyazott táblázat ugyanúgy jeleníthető meg, nyomtatható ki, mint bármilyen más érték.
- Azonban ennek az alábbi két értéknek van két típuskonstruktor:
 1. A tábláknak
 2. A soroknak a táblákban

Példa: Beágyazott táblák lekérdezése

- Lipton teáit keressük ki:

```
SELECT teák FROM Gyártók  
WHERE név = 'Lipton';
```

- Egy értéket eredményez:

```
TeaTáblaTípus(  
  TeaTípus('Brisk', 'iced', 'brown'),  
  TeaTípus('Rooibos', 'herbal', 'red'),...  
)
```


Beágyazott táblán belüli lekérdezés

- Egy beágyazott táblát hagyományos relációvá lehet konvertálni a TABLE() alkalmazásával
- Ezt a relációt, ugyanúgy mint bármely másikat, a FROM záradékban lehet alkalmazni.

Példa: TABLE() használata

- Keresd meg Lipton által gyártott „gyógyteák”-kat:

```
SELECT bb.név
```

```
FROM TABLE(
```

```
    SELECT teák  
    FROM Gyártók  
    WHERE név = 'Lipton'
```

```
) bb
```

```
WHERE bb.fajta = 'herbal';
```

Beágyazott tábla
Lipton
teákra

Alias a névnélküli
beágyazott táblára

Példa2: TABLE() használata

```
CREATE TYPE cim_t AS OBJECT (  
    utca      VARCHAR2(30),  
    varos     VARCHAR2(20),  
    ir_szam   CHAR(4) );  
  
CREATE TYPE cim_tab AS TABLE OF cim_t;  
  
CREATE TABLE vevok (  
    vevo_id   NUMBER,  
    cimek     cim_tab )  
NESTED TABLE cimek STORE AS vevo_cimek;
```

Példa2: TABLE() használata

```
INSERT INTO vevok VALUES (1,  
    cim_tab(  
        cim_t('Malom utca 2.', 'Varos', '9999'),  
        cim_t('Nagy utca 1.', 'Falu', '0000')  
    ) );  
  
INSERT INTO vevok VALUES (2,  
    cim_tab(  
        cim_t('Pajta utca 1.', 'Varos', '9999') ) );
```

Példa2: TABLE() használata

```
SELECT * FROM vevok;
```

```
vevo_id cimek
```

```
-----
```

```
1 CIM_TAB(CIM_T('Malom utca 2.', 'Varos', '9999'),  
          CIM_T('Nagy utca 1.', 'Falu', '0000'))  
2 CIM_TAB(CIM_T('Pajta utca 1.', 'Varos', '9999'))
```

```
SELECT v.vevo_id, u.* FROM vevok v, TABLE(v.cimek) u;
```

```
vevo_id utca          varos ir_szam
```

```
-----
```

```
1 Malom utca 2. Varos 9999  
1 Nagy utca 1.  Falu  0000  
2 Pajta utca 1. Varos 9999
```

Példa3: TABLE() használata

```
CREATE TYPE mobilszam_t AS OBJECT (  
    szam                CHAR(7),  
    szolgaltato         INT);
```

```
CREATE TYPE mobilszam_tab AS TABLE OF mobilszam_t;
```

```
CREATE TABLE vevok (  
    vevo_id  NUMBER,  
    cimek    cim_tab,  
    szamok   mobilszam_tab )  
NESTED TABLE cimek STORE AS vevo_cimek  
NESTED TABLE szamok STORE AS vevo_szamok;
```

Példa3: TABLE() használata

```
INSERT INTO vevok VALUES (1,  
    cim_tab(  
        cim_t('Malom utca 2.', 'Varos', '9999'),  
        cim_t('Nagy utca 1.', 'Falu', '0000')  
    ),  
    mobilszam_tab(  
        mobilszam_t('0123456', 30),  
        mobilszam_t('6543210', 20)  
    ) );
```

Példa3: TABLE() használata

```
SELECT v.vevo_id, u.*, t.*
```

```
FROM vevok v, TABLE(v.cimek) u, TABLE(v.szamok) t;
```

vevo_id	utca	varos	ir_szam	szam	szolgaltato
---------	------	-------	---------	------	-------------

-----	-----	-----	-----	-----	-----
-------	-------	-------	-------	-------	-------

1	Malom utca 2.	Varos	9999	0123456	30
---	---------------	-------	------	---------	----

1	Malom utca 2.	Varos	9999	6543210	20
---	---------------	-------	------	---------	----

1	Nagy utca 1.	Falu	0000	0123456	30
---	--------------	------	------	---------	----

1	Nagy utca 1.	Falu	0000	6543210	20
---	--------------	------	------	---------	----

Példa4: TABLE() használata

```
CREATE TYPE cim_t AS OBJECT (  
    utca      VARCHAR2(30),  
    varos     VARCHAR2(20),  
    ir_szam   CHAR(4) );
```

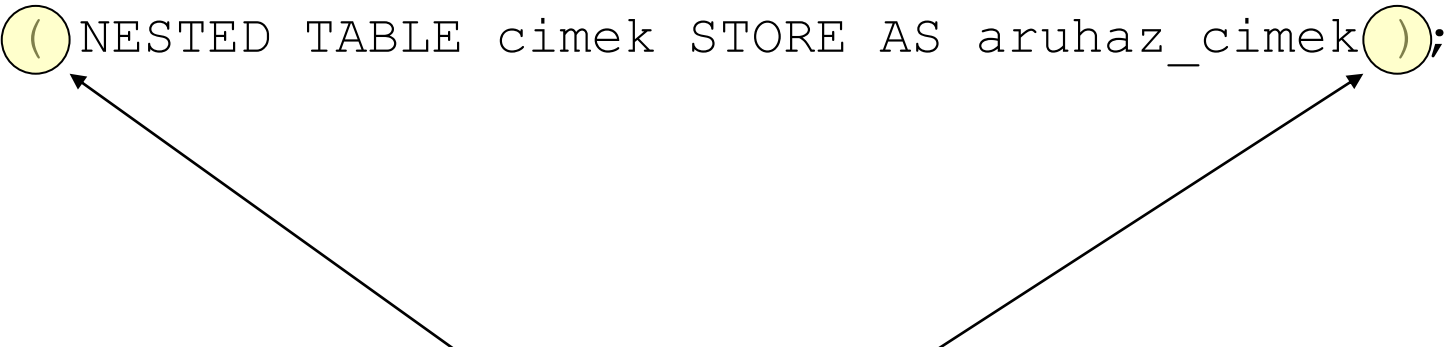
```
CREATE TYPE cim_tab AS TABLE OF cim_t;
```

```
CREATE TYPE vevo_t AS OBJECT (  
    vevo_id   NUMBER,  
    cimek     cim_tab );
```

```
CREATE TYPE vevo_tab AS TABLE OF vevo_t;
```

Példa4: TABLE() használata

```
CREATE TABLE aruhazak (  
    aruhaz_id      NUMBER,  
    nap            DATE,  
    vevok           vevo_tab )  
NESTED TABLE vevok STORE AS aruhaz_vevok  
( NESTED TABLE cimек STORE AS aruhaz_cimek );
```

The diagram consists of two yellow circles with black outlines. The first circle is positioned over the opening parenthesis '(' of the line '(NESTED TABLE cimек STORE AS aruhaz_cimek);'. The second circle is positioned over the closing parenthesis ')' of the same line. Two black arrows originate from these circles and point downwards towards the text 'A zárójelek használatára figyelni!'.

A zárójelek használatára figyelni!

Példa4: TABLE() használata

```
INSERT INTO aruhazak VALUES (1, DATE '2020-12-20',
    vevo_tab(
        vevo_t(1,
            cim_tab(
                cim_t('Malom utca 2.', 'Varos', '9999'),
                cim_t('Nagy utca 1.', 'Falu', '0000')
            )
        ),
        vevo_t(2,
            cim_tab(
                cim_t('Pajta utca 1.', 'Varos', '9999')
            )
        )
    ) );
```

Relációk beágyazott táblává alakítása

- Bármely reláció megfelelő számú attribútummal és azok illeszkedő adattípusaival egy beágyazott tábla értékei lehetnek.
- Használjuk a `CAST(MULTISET(...) AS <type>)` utasítást a reláción azért, hogy a helyes adattípussal rendelkező értékeivel egy beágyazott táblázattá alakítsuk.

Példa: CAST --- 1

- Tegyük fel, hogy a **Teák(tea, gyártó)** olyan reláció, hogy a **tea** egy **TeaTípus** típusú objektum és *gyártó* pedig egy sztring – a tea gyártója.
- Egy új sort akarunk beilleszteni a Gyártók –ba , „Pete’s” -t, mint (gyártó) nevet, és Pete által gyártott teák halmazát.

```
CREATE TYPE TeaTípus AS OBJECT (  
    név      CHAR(20),  
    fajta    CHAR(10),  
    szín     CHAR(10)  
);
```

```
CREATE TYPE TeaTáblaTípus AS TABLE OF TeaTípus;
```

```
CREATE TABLE Gyártók (  
    név      CHAR(30),  
    cím      CHAR(50),  
    teák     TeaTáblaTípus  
)  
NESTED TABLE teák STORE AS TeaTábla;
```

Példa: CAST --- 2

INSERT INTO Gyártók VALUES (

'Pete''s', 'Palo Alto',

CAST(

MULTISET(

SELECT bb.tea

FROM Teák bb

WHERE bb.gyártó = 'Pete''s'

) AS TeaTáblaTípus

)

);

Pete teái
TeaTípus objektumok
halmaza

Az objektumok halmazát
beágyazott relációvá alakítjuk