Adatbázisok MSc

6. téma

Az SQL objektum-orientált elemei

SQL OO alapok

Az SQL OO elemek bizonyos pontokban eltérést mutatnak az OOP-től

OOP	SQL
osztály	UDT
objektum	objektum
-	objektum halmaz
kollekciók	kollekciók
metódusok	metódusok
öröklés	öröklés
elrejtés	- [DBSM DAC]
polimorfizmus	polimorfizmus
konstruktor/..	konstruktor/observer/mutator/..

SQL OO alapok

Az UDT fogalma

UDT: User Defined Type

Az UDT egységbe foglalja az adattagokat és a kapcsolódó metódusokat

Előnyei:

- egység, keret
- újra felhasználhatóság
- szemantikai tartalom
- tömörség

Felhasználható:

- mező típusa
- változó típusa
- tábla típusa
- paraméter, visszatérési érték

```
UDT dolgozó: {
        név char(20);
        fizetés number(6);

        fizetés_emeles(nov number);
        kilépés();
}
```

SQL OO alapok

Az UDT fogalma

Az UDT típusai: eltérnek funkcionalitásban és felhasználásban

Az UDT használata

UDT mint megkülönböztető elemi típus

Létező gyári elemi típushoz kapcsolódik
Elemi értéket tárol
Nincs struktúra, metódus
Célja a szemantikai különbözőség jelzése

```
CREATE TYPE tnev AS gytipus FINAL;
```

```
CREATE TYPE eletkor_tipus AS INTEGER FINAL;
CREATE TABLE dolgozo (nev CHAR(25), kor eletkor_tipus, fizetes INT);
```

Nem keveredhet a műveletekben más típusokkal

hibás	helyes
SELECT … WHERE fizetes < kor	WHERE fizetes < CAST (kor TO INT)
SELECT … WHERE kor < 17	WHERE CAST (kor TO INT) < 17
UPDATE … SET kor = 19	SET kor = CAST(19 TO eletkor_tipus)

Az UDT használata

UDT mint megkülönböztető elemi típus

Az elemi UDT támogatása még igen heterogén (2008)

ORACLE: nem támogatja

SQLServer: részben támogatja

```
create type kor_tip from integer;
create table dolgozo (nev char(20), kor kor_tip, fiz int);
insert into dolgozo values ('peter', 23,233);
select * from dolgozo where kor > 21;
select * from dolgozo where cast(kor as int) > 21;
```

DB2: nagyrészben támogatja

```
create distinct type kor_tip as number(6);
```

Az UDT használata

UDT mint összetett típus

Több létező típusból áll össze
Összetett értéket tárol
Van struktúra és metódus
Célja az egységbe zárás és öröklés jelzése

```
CREATE TYPE tnev UNDER ostipus AS
 (
          szerkezet
 )
[NOT] FINAL
[NOT] INSTANTIABLE
opciók
metódus-interfészek
```

```
CREATE TYPE dolgozo_tipus AS (
          nev CHAR(25),
          kor eletkor_tipus
)   NOT FINAL;
```

Az UDT használata

UDT mint összetett típus

A modellben csak egyszeres öröklés támogatott
- egyértelmű a leszármazott struktúrája

A struktúra felépítése hasonlít a tábla sémára, de elvi eltérések vannak:

- UDT táblán kívüli is lehet
- UDT-ben nem lehet integritási feltételeket megadni

```
CREATE TYPE dolgozo_tipus AS (
        nev CHAR(25),        <-- nem lehet PRIMARY KEY
        kor eletkor_tipus    <-- nem lehet NOT NULL
)   NOT FINAL;
```

- a DEAFULT opció viszont megengedett

Az öröklésnek összhangban kell lennie a FINAL opció beállításokkal

Az UDT használata

UDT mint összetett típus

Beépített gyári kollekció típusok:

struktúra: ROW (szerkezet)
tömb: tipus ARRAY[méret]

```
CREATE TYPE dolgozo AS (
        nev CHAR (25),
        lakcim ROW (
                utca CHAR(25),
                hszam INT
        )
        telefon CHAR (15) ARRAY[5],
        kor eletkor_tipus
);
```

A különböző típusok egymásba ágyazhatók

Az UDT használata

UDT mint tábla típus

Adott típusú objektumok halmaza
A tárolt objektumoknak egyediségük van (OID)
Relációs jellegű műveletek értelmezettek rajta
Támogatja az öröklési kapcsolatot

```
CREATE TABLE tnev OF tipus UNDER ostabla
  tábla opciók;
```

```
CREATE TYPE diak AS (
        nev CHAR(25),
        atlag FLOAT);

CREATE TABLE diakok OF diak;
```

Speciális elemek: → integritási feltételek
Speciális elemek: → öröklési hierarchia

Az UDT használata

UDT mint tábla típus

Az integritási feltételek megadása:

```
CREATE TABLE tnev OF tipus [UNDER ostipus]
 (
        mezo WITH OPTION megszorítás
 )
```

megszorítások:

```
[CONSTRAINT mnev] NOT NULL
[CONSTRAINT mnev] CHECK feltétel
[CONSTRAINT mnev] UNIQUE
[CONSTRAINT mnev] PRIMARY KEY
[CONSTRAINT mnev] REFERENCES m
```

A PRIMARY KEY feltételt csak a gyökérős táblára lehet alkalmazni

A táblák hierarchiájának meg kell egyezni a típus hierarchiával

Az UDT használata

UDT mint tábla típus

A táblában tárolt objektumok azonosítása:

- normál mező nem megfelelő
- mesterséges azonosító (OID)

Az OID is tábla hatáskörű

Self-referencing column: az OID-t tartalamzó mező

```
CREATE TABLE tnev OF tipus [UNDER ostipus]
  (
        REF IS mező oid_tipus
  )
```

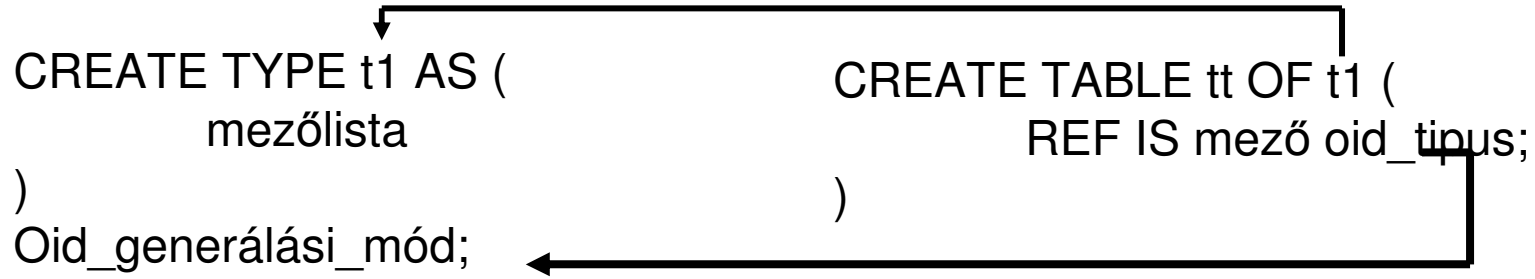
OID érték származtatási módjai (tipus):

- SYSTEM GENERATED
- USER GENERATED
- DERIVED

Az UDT használata

UDT mint tábla típus

Az OID megadásának módja redundánsan történik



oid_generálási_mód:
- REF USING tipus
- REF FROM (mezolista)
- REF IS SYSTEM GENERATED

oid_tipus:
- USER GENERATED
- DERIVED
- SYSTEM GENERATED

Az UDT használata

UDT mint tábla típus

Mivel az UDT objektumok egyedi OID-tel rendelkeznek, egyértelműen kijelölhetők

A kijelölés a referencia típussal történik

TYPE: mező REF (tipus) [REFERENCES ARE CHECKED] TABLE: mező WITH OPTIONS SCOPE tábla

```
CREATE TYPE csapat AS (
    nev CHAR(25),
    helyezes INT
)
INSTANTIABLE
NOT FINAL
REF IS SYSTEM GENERATED
```

```
CREATE TABLE csapatok OF csapat
(REF IS csid SYSTEM GENERATED)
```

```
CREATE TYPE jatekos AS (
        nev CHAR (25),
        csapat REF (csapat)
)
INSTANTIABLE
NOT FINAL
REF IS SYSTEM GENERATED
```

```
CREATE TABLE jatekosok OF jatekos
(REF IS jid SYSTEM GENERATED,
 csapat WITH OPTIONS SCOPE jatekosok)
```

Az UDT használata

UDT típusú mezők adatinak kezelése

```
CREATE TYPE csapat_tip AS (
    nev CHAR(25),
    letszam INT
)

CREATE TABLE bajnoksag (kod INT, csapat csapat_tip, pont INT);
```

Új rekord felvitele:

```
INSERT INTO bajnoksag ( 2, NEW csapat_tip('futoklub',6), 23);
```

Rekord módosítás:

```
UPDATE bajnoksag SET csapat = NEW csapat_tip('futoklub',6)
WHERE kod = 23;
```

```
UPDATE bajnoksag SET csapat = csapat.letszam(7)
WHERE kod = 23;
```

Az UDT használata

UDT típusú mezők adatinak lekérdezése

Összetett mezők esetén a lekérdezésben a táblanévhez aliast (iterátváltozót) kell rendelni

```
SELECT alias.mezo.tag….FROM tabla alias WHERE alias.mezo.tag…
```

Elemi hivatkozásnál nem szükséges az alias használata

```
SELECT mezo….FROM tabla  WHERE mezo
```

A mező tagokat összekötő operátorok:

. : objektum adattagja, metódusa

-> : hivatkozott objektum adattagja, metódusa

REF(o) : az objektum hivatkozása

DEREF(o) : a hivatkozott objektum

Tábla leszármazottak elrejtése:

ONLY (tábla)

Az UDT használata

UDT típusú mezők adatainak lekérdezése

```
CREATE TYPE jatekos AS (
        nev CHAR (25),
        csapat REF (csapat)
)
CREATE TABLE jatekosok OF jatekos
(REF IS jid SYSTEM GENERATED,
 csapat WITH OPTIONS SCOPE jatekosok)
```

```
SELECT nev FROM jatekosok

SELECT DEREF(csapat) FROM jatekosok

SELECT a.nev, a.csapat->nev FROM jatekosok a

SELECT a.nev FROM jatekosok a WHERE a.csapat->helyezes < 3

SELECT * FROM ONLY(jatekosok)
```

Az UDT használata

UDT típusú mezők adatainak lekérdezése

```
CREATE TABLA dolgozo (
   kod INTEGER,
   nyelvi NYELVISMERET )
```

```
INSERT INTO dolgozo VALUES (
  3, NEW NYELVISMERET('ANGOL',3));
UPDATE dolgozo
   SET nyelv = NEW NYELVISMERET('ANGOL',3);
UPDATE dolgozo
   SET nyelv = NEW NYELVISMERET(nyelv.nyelvi,3);
UPDATE dolgozo
  SET nyelv =  nyelv.szint(3);
UPDATE dolgozo
  SET nyelv.szint = 3;
SELECT kod, CAST (nyelvi AS CHAR(20)) FROM dolgozo;
```

Az UDT használata

UDT típusú nézeti táblák

Object VIEW: A nézeti táblák is objektum halmazok

```
CREATE VIEW vnev OF tipus UNDER ostipus
 (mezőlista)
AS lekérdezés WITH CHECK OPTION
```

Itt is meg kell adni az OID generálási módot

CHECK OPTION: a VIEW módosítására ad megkötést

```
CREATE VIEW sz_jatekosok OF jatekos
(REF IS SYSTEM GENERATED)
AS SELECT  a.nev, a.csapat
FROM jatekosok a WHERE a.csapat->helyezes < 4
```

Oracle – ORDMS lehetőségek

UDT: - objektum típusok
- kollekció típusok

```
SQL> CREATE TYPE SZEMELY AS OBJECT (
  2    NEV VARCHAR2(20),
  3    TEL VARCHAR2(14));
```

```
SQL> CREATE TABLE MUNKA (LEIRAS CHAR(20),
        FELELOS SZEMELY);
SQL> INSERT INTO MUNKA VALUES ('PROBA MUNKA',
        SZEMELY('PETER','123'));
SQL> SELECT * FROM MUNKA
LEIRAS                                   FELELOS(NEV, TEL)
----------------------------------------------------------------------
PROBA MUNKA                              SZEMELY('PETER', '123')
```

Alias név használata kötelező objektum attribútum hivatkozásoknál

```
SQL> SELECT * FROM MUNKA M WHERE M.FELELOS.NEV
        LIKE 'PETER%';
SQL> SELECT M.FELELOS.NEV FROM MUNKA M;
FELELOS.NEV
--------------------
PETER
```

SELECT FELELOS FROM MUNKA;	ok
SELECT FELELOS.NEV FROM MUNKA;	nem
SELECT MUNKA.FELELOS.NEV FROM MUNKA;	nem
SELECT M.FELELOS.NEV FROM MUNKA M;	ok

Objektum tábla

```
SQL> CREATE TABLE SZEMELYEK OF SZEMELY;
```

```
SQL> INSERT INTO SZEMELYEK VALUES ('ANNA','3424');
SQL> SELECT NEV FROM SZEMELYEK WHERE TEL
        LIKE '3424%';
NEV
------------------
ANNA
```

```
INSERT INTO SZEMELYEK VALUES(NULL)                nem
INSERT INTO SZEMELYEK
    VALUES(SZEMELY(NULL,NULL))                    ok
```

Objektum view

```
SQL> CREATE TYPE NEZET AS OBJECT (
  2    NEV CHAR(30),
  3    FIZ NUMBER(3));
SQL>  CREATE VIEW V   OF NEZET WITH OBJECT
     IDENTIFIER (NEV) AS SELECT M.FELELOS.NEV NEV,
     M.FIZETES FROM MUNKA M;
SQL> SELECT * FROM V;
     NEV                  FIZ
     -------------------- ----------
     PETER                 200
SQL> CREATE VIEW V2 (F1,F2) AS SELECT M.LEIRAS,
      NEZET(M.FELELOS.NEV, M.FIZETES) FROM MUNKA M;
SQL> SELECT V.F1, AVG(V.F2.FIZ) FROM V2 V
     GROUP BY V.F1;
```

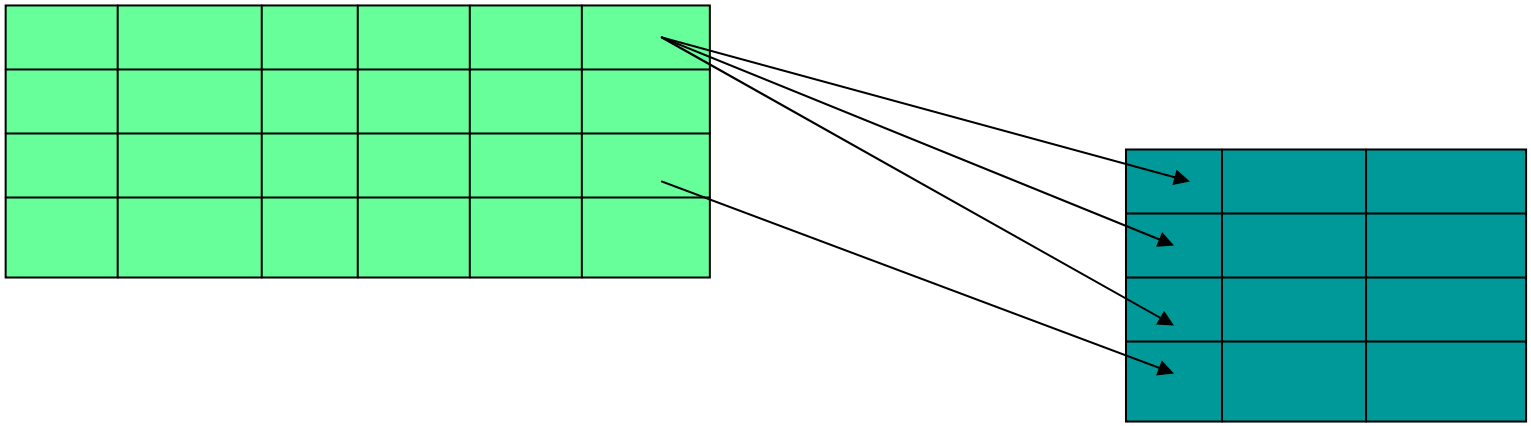
SELECT kifejezések szabadabb használata

```
SQL> SELECT LEIRAS, (SELECT MAX(FIZETES)
       FROM MUNKA) FROM MUNKA;                          ok
SQL> SELECT MAX(SELECT FIZETES FROM MUNKA)
       FROM DUAL;                                       hiba
SQL> SELECT MAX(AFIZ) FROM (SELECT LEIRAS,
       AVG(FIZETES) AFIZ FROM MUNKA
       GROUP BY LEIRAS);                                ok
SQL> INSERT INTO MUNKA VALUES ('PROBA',NULL,
       (SELECT MAX(FIZETES) + 1 FROM MUNKA));
1 sor létrejött.                                        ok
SQL> CREATE ASSERTION A1 CHECK (SELECT
       MAX(FIZETES) FROM MUNKA) > 100);                 hiba
SQL> UPDATE MUNKA SET FIZETES = (SELECT
       MIN(FIZETES) FROM MUNKA)
       WHERE FIZETES < 150;                             ok
```

Tábla típus

```
SQL> CREATE TYPE NYELV AS OBJECT (
        NYNEV CHAR(20),
        SZINT NUMBER(1));
SQL> CREATE TYPE NYELVEK AS TABLE OF NYELV;

SQL> ALTER TABLE MUNKA ADD (NYSZINT NYELVEK)
        NESTED TABLE NYSZINT STORE AS NYTABLA;
```



Tábla típus

```
SQL> INSERT INTO MUNKA VALUES ('UJABB',
        SZEMELY('KATI','3462'),301,  NYELVEK(
        NYELV('ANGOL',1),NYELV('NEMET',2)));
SQL> SELECT * FROM TABLE (SELECT NYSZINT FROM
        MUNKA WHERE LEIRAS='UJABB');
NYNEV                SZINT
-------------------- ----------
ANGOL                     1
NEMET                     2

SQL> INSERT INTO TABLE(SELECT NYSZINT FROM
        MUNKA WHERE LEIRAS='UJABB') VALUES ('FINN',3);
SQL> UPDATE TABLE(SELECT NYSZINT FROM MUNKA
       WHERE LEIRAS='UJABB') SET SZINT = 4
       WHERE NYNEV = 'FINN';
```

Objektum azonosítás, hivatkozás

objektum azonosítás (OID) : - rendszer által generált
- kulcsból képzett

indexelt

hivatkozás : REF()

```
SQL> CREATE TYPE AUTO AS OBJECT (
        RSZ CHAR(6),
        TULAJ REF SZEMELY);
SQL> CREATE TABLE AUTOK OF AUTO;
SQL> SELECT P.NEV, REF(P) FROM SZEMELYEK P;
        NEV     REF(P)
        ----------------------------------------------------------
        ANNA  0000280209447BAB1EDDE24A5886E9
       C64B6BC741586786A61E89134C158795B6AE6
       A5152000040C5820000
```

Objektum hivatkozás

```
SQL>  INSERT INTO AUTOK VALUES('R11', (SELECT
        REF(P) FROM SZEMELYEK P WHERE P.NEV='ZOLI'));

SQL> SELECT * FROM AUTOK;
        RSZ     TULAJ
        ------------------------------------------------------------
        R11      000022020884E1C92BF87047A48E8D41C….
SQL> SELECT A.RSZ, A.TULAJ.NEV FROM AUTOK A;
        RSZ    TULAJ.NEV
        ------ --------------------
        R11    ZOLI

SQL> CREATE TYPE CSOPREF AS TABLE OF REF SZEMELY;
SQL> ALTER TYPE AUTO ADD ATTRIBUTE UTASOK
        CSOPREF CASCADE;
```

Objektum hivatkozás

```
SQL> UPDATE AUTOK SET UTASOK = CSOPREF
        ((SELECT REF(S) FROM SZEMELYEK S WHERE
        S.NEV = 'ANNA'),  (SELECT REF(S) FROM SZEMELYEK
        S WHERE S.NEV = 'PETER'));

SQL> SELECT * FROM AUTOK;
        RSZ   TULAJ            UTASOK
        ------------------------------------------------------
        R11   000022020        CSOPREF(0000220208447B..
SQL> SELECT A.RSZ, A.TULAJ.NEV, A.UTASOK.NEV FROM
        AUTOK A;                                              hiba
SQL> SELECT P.UTASOK FROM AUTOK P;
        UTASOK
        ------------------------------------------------------
        CSOPREF(00002202084…, 0000220208A49BC… )
```

Objektum hivatkozás

```
SQL> SELECT * FROM TABLE (SELECT UTASOK
        FROM AUTOK);
        COLUMN_VALUE
        ------------------------------------------------------------
        0000220208447BAB1EDDE24A5886E9C64B6BC7415
        0000220208A49BCCF9F8874A1DB7F287F8D315B57
SQL>  SELECT P.COLUMN_VALUE.NEV FROM
        TABLE(SELECT UTASOK FROM AUTOK) P;
        COLUMN_VALUE.NEV
        ----------------
        ANNA
        PETER
```

ADT öröklés

```
SQL> CREATE TYPE EMBER AS OBJECT (
          NEV VARCHAR2(20)
         ) NOT FINAL;
SQL> CREATE TABLE T1 OF EMBER;
SQL> INSERT INTO T1 VALUES('PETER');
SQL> CREATE TYPE DIAK UNDER EMBER (
          ATLAG NUMBER(4,2));
SQL> CREATE TABLE T2 OF DIAK;
SQL> INSERT INTO T2 VALUES ('ZOLI',2.3);
```

```
SQL> SELECT * FROM T2;
          NEV                  ATLAG
          ZOLI                   2.
SQL> SELECT * FROM T1;
          NEV
          PETER
```