

Adatbázisok II.

1

Jánosi-Rancz Katalin Tünde

tsuto@ms.sapientia.ro

327A

Ajánlott irodalom

- Ullman-Widom: Adatbázisrendszerek. Alapvetés (Második, átdolgozott kiadás), Panem, 2008. november (bővítés pl.UML, UDT, JDBC, PHP, XML)
- Kende-Nagy: Oracle példatár (Oracle 9i és 10g verziókhoz), Panem, 2005.
- Gábor A.-Juhász I.: PL/SQL-programozás ORACLE 10g-ben, Panem, 2006.
- Varga Ibolya: Adatbázisrendszerek, Egyetemi Kiadó, Kolozsvár, 2005
- Steven Feuerstein weboldala:

<http://www.toadworld.com/Knowledge/DatabaseKnowledge/StevenFeuersteinsPLSQLObsession/tabid/153/Default.aspx>

Websites for PL/SQL Developers



www.plsqlchallenge.com

Daily PL/SQL quiz with weekly and monthly prizes



www.plsqlchannel.com

27+ hours of detailed video training on Oracle PL/SQL



www.stevenfeuerstein.com

Monthly PL/SQL newsletter



Steven Feuerstein's
PL/SQL Obsession

www.toadworld.com/SF

Quest Software-sponsored portal for PL/SQL developers

Célkitűzések

- ▶ **Tárolt eljárások, függvények, csomagok bemutatása**
- ▶ **Tranzakciókezelés részletes bemutatása, Holtpont megelőzés és felfedezés. Semmiségi és helyrehozó naplózás.**
- ▶ **XML típus bemutatása, XQUERY**
- ▶ **Osztott adatbázis fogalmak, tervezés, tranzakciókezelés osztott adatbázisok esetén, osztott lekérdezések feldolgozása, megismerése.**
- ▶ **Adattárházak ismertetése**

- ▶ **Laboron bonyolult tárolt eljárások és tranzakciókezelés lesz, majd egy többfelhasználós konkurencia problémákat kezelő kollektív adatbázis projektet kell készíteni.**

Szerver oldali programozás

Tárolt eljárások és függvények
Oracle-ben

▶ PL/SQL

- ★ kiterjesztett SQL, pl. elágazások, ciklusok, stb.

▶ Tárolt eljárás

- ★ PL/SQL utasítások gyűjteménye, ami egy feladatot valósít meg, paraméterezhetőek, saját egyedi azonosító nevük van és az adatbázisban lefordított formában eltárolásra kerülnek

- ★ Több kimeneti értéke is lehet

▶ Függvény

- ★ egyetlen kimeneti érték
- ★ bárhol használható, ahol érték használható

Tárolt eljárások létrehozása (deklarációja)

```
CREATE OR REPLACE PROCEDURE eljárás_név
    [paraméter_lista] IS --IN, OUT, INOUT
BEGIN
    <az eljárás végrehajtandó része>
    [EXCEPTION
        <a kivétel-kezelő rész>]
END [eljárás_név];
```

Példa egy tárolt eljárásra

```
CREATE OR REPLACE PROCEDURE Hire_Emp(  
    name VARCHAR2,  
    job VARCHAR2,  
    mgr NUMBER,  
    hiredate DATE,  
    sal NUMBER,  
    comm NUMBER,  
    deptno NUMBER)  
IS  
BEGIN  
    INSERT INTO emp  
    VALUES(emp_sequence.nextval, name, job, mgr, hiredate, sal, comm, deptno);  
END;
```

meghívás:

```
EXECUTE Hire_Emp('Queen', 'SALESMAN', 7698, '11-Nov-2000',2005,NULL,30)
```

törlés:

```
DROP PROCEDURE Hire_Emp
```


Függvények létrehozása (deklarálása)

```
CREATE [OR REPLACE] FUNCTION függvény_név  
    [(paraméter1 [IN | OUT | IN OUT] típus,  
     [paraméter2 [IN | OUT | IN OUT] típus, ... )]  
RETURN <az_eredmény_típusa>  
AS  
<függvény_törzse>;    -- egy PL/SQL blokk.
```

Példa egy függvényre

A visszatérési értéket a RETURN utasítással határozzuk meg

```
CREATE FUNCTION atlag (  
    tip IN CHAR(20))  
RETURN NUMBER  
IS  
    ertek    NUMBER;  
BEGIN  
    SELECT AVG(ar)  
    INTO ertek  
    FROM autok  
    WHERE tipus LIKE tip;  
  
    RETURN (ertek);  
END;
```

Létező eljárások lekérdezése:

```
SELECT object_name  
FROM user_objects  
WHERE object_type LIKE 'PROCEDURE%';
```

Létező függvények lekérdezése:

```
SELECT object_name  
FROM user_objects  
WHERE object_type LIKE 'FUNCTION%';
```

A tárolt eljárások és függvények előnyei

- ▶ Biztonság
 - ★ Nem kell a táblákra jogosultságot adnunk, hanem elég az eljárásra
 - ★ Ha valaki meghívja a tárolt eljárást (ha joga van rá), akkor az eljárást létrehozó felhasználó kontextusában fut
- ▶ Teljesítmény
 - ★ Kevesebb információt kell átküldeni a hálózaton
 - ★ A lefordított változat tárolódik az adatbázisban
 - ★ Lehet, hogy már amúgy is benn van a memóriában
 - ★ Memória-foglalás
 - ❖ Több felhasználó is megosztozhat ugyanazon a betöltött kódon

A tárolt eljárások és függvények előnyei

★ Termelékenység

- ❖ Tárolt eljárások a szerveren – kliensek ezeket hívogatják
- ❖ Ha változik a logika, elég a szerveren változtatni

★ Adatintegritás

- ❖ Működési korlátozások biztosítása

★ Azaz:

- ❖ Megosztott alkalmazás-logika
- ❖ Elrejt az adatbázis szerkezetét
- ❖ Biztonsági mechanizmus
- ❖ Javítja a teljesítményt
- ❖ Csökkenti a hálózati forgalmat

Tárolt eljárások típusai

- ▶ Egyedülálló (standalone)
- ▶ Csomagban (package)
- ▶ Külső (External)
 - ★ C-ben íródik
 - ★ az Oracle-től elkülönített címtartományban fut

Tárolt eljárás létrehozása

- ▶ Lefordítja a tárolt eljárást
- ▶ Eltárolja a lefordított eljárást
 - ★ neve
 - ★ forráskód, és értelmezési fa
 - ★ pszeudo kód (P-kód)
 - ★ hibaüzenetek
 - ★ **EZEK AZ ADATSZÓTÁRBAN VANNAK!!!**
(SYSTEM)

Felhasználói jogosultságok

- ▶ EXECUTE jogosultság kell annak, aki végre akarja hajtani (a tulajdonosnál ez alapértelmezés szerint megvan)
- ▶ A tulajdonosnak megfelelő jogosultságokkal kell rendelkeznie, hogy elérje a megfelelő adatokat.

A tárolt eljárás érvényessége

- ▶ Érvénytelenné válik az eljárás, ha
 - ★ valamilyen hivatkozott objektumot módosítunk vagy törlünk
 - ★ olyan rendszer-jogosultságot vonunk meg a tulajdonostól, amit használ az eljárásban
 - ★ olyan objektum-jogosultságot vonunk meg a tulajdonostól, amit használ az eljárásban
- ▶ Ha érvénytelenné válik, akkor a következő futtatáskor automatikusan újrarendeltődik

Megoldott feladatok

1) **CREATE OR REPLACE PROCEDURE** kiir(szoveg in varchar2)
IS
BEGIN
dbms_output.put_line(szoveg);
END;

2) Készítsünk eljárást, amely új terméket visz fel úgy, hogy a termék adatait paraméterként kapja!

```
CREATE OR REPLACE PROCEDURE termekbeszur (  
    cikkszam in char, termeknev in varchar2, ar in number,  
    raktmenny in number, gyid in number)  
IS  
BEGIN  
    INSERT INTO termek (cikkszam, termeknev, ar, raktmenny, gyid)  
    VALUES (cikkszam, termeknev, ar, raktmenny, gyid);  
COMMIT;  
EXCEPTION  
    WHEN dup_val_on_index then  
        kiir('Mar van ilyen cikkszam');  
END;
```

3) Készítsünk egy eljárást, amely segítségével számlaszám és cikkszám alapján egy tételt tudunk törölni a számla_termék táblából!

```
CREATE OR REPLACE PROCEDURE torol
```

```
    (szlasz in number, csz in char)
```

```
IS
```

```
BEGIN
```

```
    DELETE FROM számla_termék
```

```
    WHERE szlaszam = szlasz AND cikkszam = csz;
```

```
    COMMIT;
```

```
END;
```

- 4) Írjunk eljárást, amely bekéri egy termék azonosítóját, és megnöveli az összes termék árát, amely termékek ára kisebb, mint a bekért termék ára! Az emelés mértékét szintén a felhasználó adja meg. Amennyiben nem létezik az adott cikkszámú termék, akkor erre figyelmeztesse a felhasználót!

```
CREATE OR REPLACE PROCEDURE modosit
(cszam in char, szazalek in number)
IS
    ara termék.ar%type;
BEGIN
SELECT ar INTO ara
FROM termék
WHERE cikkszam = cszam;
UPDATE termék
SET ar=ar*(1+szazalek/100)
WHERE ar<ara; COMMIT;
EXCEPTION
    WHEN no_data_found THEN
        kiir('Nincs ilyen termék');
END;
```