Adatbázisok

5. labor: DML

SQL utasítások kategóriái

- DQL (Data Query Language)
  - SELECT
- DDL (Data Definition Language)
  - CREATE, ALTER, DROP, TRUNCATE, RENAME
- DML (Data Manipulation Language)
  - INSERT, UPDATE, DELETE
- DCL (Data Control Language)
  - GRANT, REVOKE
- TCL (Transaction Control Language)
  - SAVEPOINT, COMMIT, ROLLBACK

DML és DDL utasítások

- DML: Data Manipulation Language
  - **sorok** beszúrása, módosítása, törlése
- DDL: Data Definition Language
  - **táblák, nézetek** létrehozása, módosítása, törlése

SEQUENCE

- SEQUENCE: adatbázis objektum, növekvő egész számok generálásához, segítségével egyedi, elsődleges kulcs értéket lehet generálni
- Az első hivatkozás a SEQ_TAB.nextval értékére 1-et ad vissza, a második hivatkozás 2-t. Minden egyes későbbi hivatkozás értéke: az előző érték + 1.

```
Create sequence SEQ_TAB
Minvalue 1
Maxvalue 99999999
Start with 1
Increment by 1
nocache;

Select SEQ_TAB.nextval from DUAL;
```

SEQUENCE

- Elsődleges kulcs értékek generálására használható a szekvencia:

```
Create sequence SEQ_TAB
Minvalue 1
Maxvalue 99999999
Start with 1
Increment by 1
nocache;

Create table TAB(id number PRIMARY KEY, name
varchar2(16) not null);
Select SEQ_TAB.nextval from DUAL;
Insert into TAB (id, name)
VALUES (SEQ_TAB.NEXTVAL, 'CBA');
```

DML - Előkészületek

- Hozzunk létre egy **employees2** táblát a teljes **employees** tábla alapján.
- Hozzunk létre egy **departments2** táblát a teljes **departments** tábla alapján.
- A **departments2**-ben elsődleges kulcs a department_id.
- Az **employees2**-ben elsődleges kulcs az employee_id, idegen kulcs a department_id a **departments2**-re.
- Az last_name mező nem lehet üres, a fizetés legalább 700 dollár legyen.

DML - Előkészületek

- create table employees2 as select * from employees;
- create table departments2 as select * from departments;
- alter table departments2 add constraint dept2_pk primary key (department_id);
- alter table employees2 add constraint emp2_pk primary key (employee_id);
- alter table employees2 add constraint emp2_deptid_fk foreign key (department_id) references departments2;
- alter table employees2 add constraint lastname_ck check (last_name is not null and salary >= 700);

DML: beszúrás

- Helyezzünk el a **departments2** táblába egy újabb részleget.

```
INSERT INTO departments2 (department_id,
  department_name, location_id)
VALUES (42,'SAJTKÉSZÍTŐ', 1000);
```

- Ami érdekes:
  - sorrend
  - minden értéket megadunk?

DML: beszúrás

- Az alábbiak közül melyik működik? Miért?

```
INSERT INTO departments2
VALUES (55, 'NÉV', 200, 1700);

INSERT INTO departments2
VALUES (30, 'NÉV', 200, 1700);

INSERT INTO departments2
(department_name, location_id)
VALUES ('NÉV',1700);
```

DML: beszúrás

- Dolgozó is kell a sajtkészítők részlegébe: vegyünk fel az **employees2** táblába Accounting Manager munkakörbe egy 1111-es azonosítójú, Bőrönd Ödön nevű személyt. Email legyen ’bodon’ , belépés dátuma 2018.10.15!
  - TIPP: a dátum egyszerűbb beszúrásához is használhatjuk a TO_DATE függvényt.

```
INSERT INTO employees2
  (employee_id, first_name, last_name,
  email, hire_date, department_id, job_id)
VALUES (1111,'Ödön','Bőrönd','bodon',
to_date('2018.10.12','YYYY.MM.DD'),
42,'AC_MGR');
```

DML: beszúrás

- Vegyünk fel a 42-es részlegbe egy Huncut Izolda nevű dolgozót Sales Manager munkakörbe 1112-es azonosítóval, a belépési dátuma legyen 2018. március 1., email címe pedig ’hizolda’!
  - TIPP: a dátum egyszerűbb beszúrásához is használhatjuk a TO_DATE függvényt.

- INSERT INTO employees2 (employee_id, first_name, last_name, email, hire_date, department_id, job_id)
VALUES (1112, 'Izolda’, 'Huncut','hizolda’, TO_DATE('2018.03.01’, 'YYYY.MM.DD’), 42, 'SA_MAN');

DML: módosítás

- Ödön nevű dolgozónknak nem adtunk fizetést, főnöke sincs még. Adjunk neki!

```
UPDATE employees2
SET salary=8500, manager_id=100
WHERE employee_id=1111;
```

- Fontos a feltétel megadása, különben mindenhol átírja!!!

DML: módosítás

- Adjunk Ödönnek jutalékot is.
  (legyen 10 százalék)

```
UPDATE employees2 SET commission_pct=0.1
WHERE employee_id = 1111;
```

- Csökkentsük le Ödön fizetését 650 dollárra.
  - Mi történik és miért?

```
UPDATE employees2 SET salary=650
WHERE employee_id=1111;
```

DML példa 1

- Adjunk az **employees2** táblához egy *cardnumber* nevű mezőt, amely a dolgozó beléptető kártyájának 5 jegyű azonosítószámát tartalmazza. A kártyaszámot a dolgozói azonosítóból képezzük a következő módon:

  cardnumber = 10000 + employee_id

Megoldás

```
ALTER TABLE employees2
ADD cardnumber NUMBER(5);

UPDATE employees2
SET cardnumber=10000+employee_id;
```

DML példa 2

- Adjunk az **employees2** táblához egy *income* nevű szöveges mezőt, amely a dolgozó jövedelmének mértékét tárolja: akinek a jövedelme 2500 dollár feletti, legyen az income értéke "HIGH", a többieknek "LOW". A megfelelő megszorítással biztosítsuk, hogy mást ne is lehessen beírni ide.

Megoldás – DDL rész

```
ALTER TABLE employees2
ADD income VARCHAR2(4);

ALTER TABLE employees2
ADD CONSTRAINT employees2_ck
CHECK (income IN ('HIGH','LOW'));
```

Megoldás – DML rész

```
UPDATE employees2
SET income='HIGH'
WHERE
salary*(1+NVL(commission_pct,0))>100
00;

UPDATE employees2
SET income='LOW'
WHERE income IS NULL;
```

DML gyakorlás

- Kapjon 10% fizetésemelést minden olyan dolgozó, akinek a fizetése 10000 dollárnál kevesebb és nem kaphat jutalékot.

```
UPDATE employees2 SET salary = salary*1.1
WHERE salary<10000 and commission_pct IS
NULL;
```

- Emeljük meg 10 százalékkal minden 80-as részlegben dolgozó jutalékát.

```
UPDATE employees2 SET commission_pct =
commission_pct * 1.1 WHERE department_id =
80;
```

DML: törlés

- A cég felszámolta sajtkészítő üzletágát. Töröljük a részleget a **departments2** táblából.

```
DELETE FROM departments2
WHERE department_id=42;
```

- **Mi történt és miért?**

DML: törlés

- Szegény Ödönt elbocsátották. Töröljük az **employees2** táblából.

```
DELETE FROM employees2
WHERE employee_id=1111;
```

- Körültekintően határozzuk meg a feltételt!

DML: törlés

- Töröljünk minden 'Sales Manager' munkakörű dolgozót az **employees2** táblából.

- Töröljük a sajtkészítők részlegét a **departments2** táblából.
  - Most már lefut az utasítás. Miért?

```
DELETE FROM employees2
WHERE upper(job_id) = 'SA_MAN’;

DELETE FROM departments2
WHERE department_id = 42;
```

Nézetek

- View-k
- Logikailag: egy vagy több tábla adatainak egy részhalmaza.
- Gyakorlatilag: egy lekérdezést „mentünk és úgy használjuk, mintha tábla lenne”.

Nézet létrehozása

```
CREATE VIEW empv1 AS
  SELECT employee_id, last_name,
  job_id
  FROM employees
  WHERE department_id = 50;
```

Nézet módosítása

```
CREATE OR REPLACE VIEW empv2 AS
SELECT employee_id, first_name,
          last_name, job_id
FROM employees
WHERE department_id BETWEEN 10 AND
50;
```

- Megjegyzés: Az ALTER VIEW nem használható a nézet **definíciójának** módosítására (csak a kapcsolódó megszorítások módosítására).

Gyakorlás

- Hozzunk létre egy nézetet **dolgozók** néven, amely listázza a dolgozók nevét, jövedelmét (fizetés + jutalék), munkakörét, belépésük évét, valamint részlegük nevét.
- Módosítsuk a dolgozók nézetet úgy, hogy az előzőeken kívül még a dolgozók főnökének nevét is tartalmazza.

Gyakorlás

```
CREATE OR REPLACE VIEW dolgozók AS
SELECT first_name, last_name, salary*(1+NVL(commission_pct,0))
as jovedelem, job_title, hire_date, department_name
FROM employees inner join departments USING (department_id)
inner join jobs using (job_id);

select * from dolgozók;

CREATE OR REPLACE VIEW dolgozók AS
SELECT e.first_name, e.last_name, f.last_name as fonok,
e.salary*(1+NVL(e.commission_pct,0)) as jovedelem, job_title,
e.hire_date, department_name
FROM employees e inner join departments USING (department_id)
inner join jobs using (job_id) inner join employees f on
e.manager_id= f.employee_ID;
```

Nézet törlése

```
DROP  VIEW empv2;
DROP  VIEW dolgozók;
```

- Az adatok megmaradnak!

DML nézeten keresztül

- Bizonyos esetekben lehetséges nézeten keresztül beszúrni, módosítani és törölni
  - kivéve ha a nézetet WITH READ ONLY opcióval hoztuk létre
  - + sok egyéb szabály

Részletes leírás az érdeklődőknek:

https://docs.oracle.com/cd/B28359_01/server.111/b28310/views001.htm#ADMIN11774

DML nézeten keresztül

- Egyszerű nézet:

```
CREATE OR REPLACE VIEW nyolcvan AS
SELECT employee_id, last_name,
job_id, salary, department_id,
hire_date, email FROM employees2
WHERE department_id=80;

SELECT * FROM nyolcvan;
```

- ```
  INSERT INTO nyolcvan (employee_id,
  last_name, email, hire_date, job_id)
  values (999,'ODON', 'bodon',
  to_date('2018.10.12','YYYY.MM.DD'),
  'SA_MAN');
  ```

```
SELECT * FROM employees2;
```
```

DML nézeten keresztül

- Előző példa folytatása:

```
SELECT * FROM nyolcvan;
```

- **Hova lett Ödön???**
  - A WHERE feltétel miatt nem látszik a nézetben.
  - Ha nem szeretnénk ilyen módosítást engedélyezni, hozzuk létre a nézetet "WITH CHECK OPTION"-nel.
-
```
INSERT INTO nyolcvan (employee_id,
last_name, email, hire_date, job_id,
department_id) values (998, 'ALADÁR',
'baladar',
to_date('2018.10.12','YYYY.MM.DD'),
'SA_MAN',80);
```

DML nézeten keresztül

- Egyszerű nézet, módosítás és törlés:

```
UPDATE nyolcvan SET
salary=salary*1.1
WHERE salary<8000;

SELECT * FROM employees2;


DELETE FROM nyolcvan WHERE
salary<8000;

SELECT * FROM employees2;
```

DML nézeten keresztül

- WITH READ ONLY opció:

```
CREATE OR REPLACE VIEW nyolcvan AS SELECT
employee_id, last_name, job_id, salary,
department_id FROM employees2 WHERE
department_id=80
WITH READ ONLY;

INSERT INTO nyolcvan (employee_id,
last_name, email, hire_date, job_id,
department_id) values (997, 'ALADÁR',
'baladar',
to_date('2018.10.12','YYYY.MM.DD'),
'SA_MAN',80);
```

- **NEM működik!**

DML nézeten keresztül

Ha kitörölted, hozd létre ezeket ismét:

- create table employees2 as
  select * from employees;
- create table departments2 as
  select * from departments;
- alter table departments2 add
  constraint dept2_pk primary key
  department_id);
- alter table employees2 add constraint
  emp2_pk primary key (employee_id);
- alter table employees2 add constraint
  emp2_deptid_fk foreign key (department_id)
  references departments2;

DML nézeten keresztül

- Join nézet:

```
CREATE OR REPLACE VIEW ketto AS
SELECT employee_id, last_name, job_id,
salary, department_name
FROM employees2 NATURAL JOIN
departments2;

SELECT * FROM ketto;

UPDATE ketto SET salary=salary+1 WHERE
upper(department_name)='ACCOUNTING';

SELECT * FROM employees2;
```

DML nézeten keresztül

- Join nézet, departments módosítása:

```
UPDATE ketto SET
department_name='CSOROK'
WHERE salary<2000;
```

- **NEM működik!**
  - Csak azt a táblát lehet a join nézeten keresztül módosítani, amelynek kulcsa a nézet kulcsa is.