

Actualizarea datelor (SQL)

INSERT

UPDATE

DELETE

1. Insert

```
INSERT INTO nume_tabela [(Cădă campuri)]
VALUES (Cădă val)
```

- de e să deț cămp tabelii poate fi și
val unbrad asoc unreg urmăred un ordinea pe
a campurilor

INSERT INTO Sucursala

```
VALUES ('x --- x', 22, 10)
```

13 char

- de se specifică numai anum cămp cel vor avea și
null (NULL)

INSERT INTO angajat (cnp, Nume, Prez, Salariu)

```
VALUES (11614010400243, 'Popescu', 'Ion', 370)
```

INSERT INTO angajat (Nume, Prez)

```
VALUES ('Popescu', 'Ion')
```

- pt cămp unique e oblig să se preciseze val
- val true sau false! = de cele 3

NULL

DEFAULT

INSERT INTO Stud (Nume, Prez, CNP, Anstol, Spec)

```
VALUES ('Popescu', 'Ion', 1611010400243, DEFAULT)
```

- un insert poate fi utilizat un cursor cu o liniere, ce parcurge tabela, rezultatul inserat linie cu linie un tabel specificat

```
CREATE TABLE DeptInfo (Name NUMBER(5),
Salmin NUMBER(8), Mediesal NUMBER(8), DeptNo
NUMBER(3))
```

```
INSERT INTO DeptInfo
VALUES (SELECT COUNT(*) MIN(Salary) , AVG(Sal
DeptNo FROM Angajat
GROUP BY DeptNo)
```

Ex.

- de regulă un insert este o oprire append (noile valori sunt adăugate la sfârșitul tabelei)
- avem medii în care și în INSERT și APPEND, cu structuri similare

```
APPEND INTO Tabelă [(Câte comune)]
VALUES (căstig)
```

- Ca un insert se adaugă în urmărirea curentă
- append hărțea curentă va adăuga înreg ca sf. tabeli
- de fapt doar un insert
 - unul mediu perm un insert & un reg crt
 - un reg crt = ultima
 - un insert = append

2. Update

- modif camp specificat în comanda


```
UPDATE nume-table
SET (nume-camp=Val; -- , nume-camp=i=
[WHERE <condiție>]
```

se mod val din camp SET ca urmă conditie

UPDATE Angajat

SET (Salarie = Salarie * 1.1)
WHERE Dnr IN (2, 4)

SELECT Salarie * 1.1

FROM Angajat
WHERE Dnr IN (2, 4)

- se updat fol succesi

ex: creșt val 10% pt ang cu \sum are Campl > 30

UPDATE Angajat

SET (Salarie = Salarie * 1.1)

WHERE (SELECT ^{exp IN} campl
FROM Succesori
GROUP BY campl
HAVING sum(are) > 30) } setea col

- ERRE - modif primary key și ac devene a val ce există deja
- de cănd WHERE se modif întreaga tabelă

UPDATE Angajat

SET (Salarie = Salarie * val)

MERGE - Cine fie ca insert, fie ca update în
faza de output

MERGE INTO Tab-dest

USING Tab-sursa ON (conditie JOIN)

WHEN TRUE

UPDATE SET (Cstacamps = valori)

WHEN NOT TRUE

INSERT VALUES (Cstă valori)

```

MERGE INTO Angajat
USING Departament
ON Department.D_nr = D_nr
WHEN TRUE
UPDATE (Salariu = Salariu * 1.2)
WHEN NOT TRUE
INSERT VALUES ('x--x', "Popescu", "Varia" -- D_nr)
13char CNP
dintreb
Departament

```

3. Delete

DELETE FROM numeTab
[WHERE conditie]

DELETE FROM Angajat ← tab goală
identic cu TRUNCATE

WHERE D_nr = 3 //

- să se steargă angajatul de departament care manag
este Popescu

DELETE FROM Angajat
WHERE emp IN (SELECT emp
FROM Angajat, Departament
WHERE D_nr = D_nr and
Manager=emp and Nume=)

Transacție

(SQL-DDL) comenzi ce nu pot fi revozate
succes de comenzi ce au ca efect modif de
conturi a B.D ce pot fi reflectate in B.D
doar sa urmare a unei comenzi de tip
COMMIT sau comenzi pot fi revozate prin
ROLLBACK

urmă UI la momentul t1 - SELECT (read)
 t1/t2 .. UPDATE Dep-unfa SET Nr-ang=0
 WHERE D-nr=3
 t2/t3 .. SELECT Nr-ang
 FROM Dep-unfa
 WHERE D-nr=3
 t2/t4 .. UPDATE Dep-unfa SET Nr-ang=Nr-a
 WHERE D-nr IN (2,3,5)
 t2/t5 .. SELECT * FROM Dep-unfa
 t1/t6 .. COUNT
 t2/t4 .. SELECT * FROM Dep-unfa

deatre UI

t2: blocare record (cel ce care trebuie să scrie
în bază - aici D-nr = 3)

t3: U2 vede seol modificarea (antieala)
cererea cui U1 nu a fost emisă, se oferă
deasupra un buffer

t4: U2 ~~verifica~~-ang+1 pt t câmp

t5: emuliază rez ↑ ← val urmt +1

t6: de la ac moment U1 a facut commit (se scrie în baza) → urmăz coresp D-nr = 3 a fost deblocat

t7: U2 read

de U1/t6 lipsește și un lucru ca U2/t6 commit, ⇒
se modif câmp pt D-nr 2,5 - si se arată pt 3
să fie deblocat

- | | | |
|----|-----------------------------|---|
| E1 | U1: UPDATE | de dormea ceste operații constituie
o transacție |
| E2 | U1: INSERT | |
| E3 | U1: UPDATE | |
| E4 | U1: DELETE | |
| E5 | U1: SELECT // vede modif | |
| E6 | U2: SELECT // vede val urmt | |

SET AUTOCOMMIT OFF ← var default
SET — OR ← Cat ap corect se face
autom dp formularia cerere

Safe Point

SAVE POINT nume

tx1: U1 UPDATE --

SAVE POINT pt1

tx2: U1 INSERT

— — — pt2

tx3: U1 UPDATE

— — — pt3

tx4: U2 SELECT

tx5: U1 ROLLBACK TO SAVE POINT pt2

tx6: U2 SELECT

WITH CHECK OPTION

insert
update
delete

- pt fi un record ca modif date din tabel
- un set un record table edificare

INSERT INTO (SELECT *

FROM Departament
WHERE Dnr=20)

VALUES(Dnr=22, * —)

*WITH CHECK OPTION

L do nr > 20 % op de insert va esua

SQL - VDL

vedere = def ce se creaza asupra BD

↳ result exec este o tabela, dar VDL nu e o tabela ci doar def acelora

VDL are autocommit

`CREATE TABLE Dep-umfă`

AS

(SELECT Dnr, COUNT(x), MAX(Salariu)
MAX(Salariu), AVG(Salariu)
FROM Angajat
GROUP BY Dnr)

`CREATE TABLE Dep-umfa(--)`

`INSERT INTO`

`COMMIT`

- alternativ \Rightarrow VIEW - cererea se executa si se vede unde este ca acel moment

`CREATE [OR REPLACE] VIEW nume_vedere`

def ca o tabela cu CREATE TABLE

`t([def col sau]) AS (subcerere)`

`[FORCE | NO FORCE]`

`[WITH CHECK OPTION [CONSTRAINT nume]`

`[WITH READ OPTION [CONSTRAINT nume]]]`

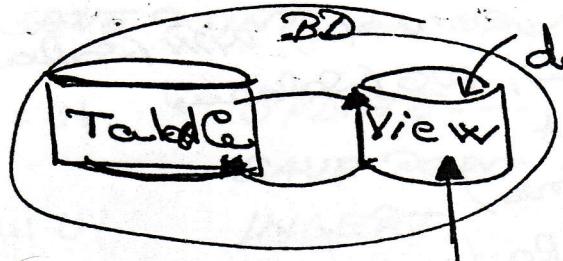
- un view nu e tab, nu se poate modifica cu ALTER TABLE, ci doar cu REPLACE

* de regulă fară const., se fol de că tabel utilizat în subcerere

`FORCE` \Rightarrow view e creat ca def chiar de tabel în loc

`NO FORCE` \Rightarrow verific subcererea si se vede se poate crea ca sit actualizat

- CHECK OPTION are ac interpretare ca in Ca update, delte, insert verifică dacă valoarea / + aici e rea că o constraintă care îl se asociază cu aceeași fi activă / dezactivată
- WITH READ ONLY < nu pot fi făcute apă de update



Vizualizare structura BD
- utilizând un plan de securitate

CREATE OR REPLACE View Depunșe

```
AS (SELECT Dnum, COUNT(*) AS numarSalarii
    MAX(Salariu), AVG(Salariu) Dnum
    FROM Angajat, Depart WHERE Dnum = Dcode
    GROUP BY Dnum)
```

WITH READ ONLY CONSTRAINT read

SELECT *

FROM Depunșe

• se apăsa pe tab în real adăuga

UPDATE Depunșe

SET (Dnum = "cerere")

WHERE Dnum = 4

modificare

• un BD Dnum
dacă cerere

• Nu se poate face UPDATE de
1-de update și ref Ca camp este prim
apămat / apămat de char

2-mu apăsat fi unei componente ale de date

GROUP BY

3- modifică camp ale securității

• face ca să se urmărească clauze de

tipă Ravning

INDEXES

- index nu se specifica tipul de index ce se doreste
- Ca un index BD of c.m. buna conditie
- de la a tab se - cu def inic spum / cod urm BD face autom index SGBD din numar definit
- index dp alt camp de la e utile/recom de - tabela are un nr max intreg
 - " " un camp de indexare nu are valoare reala
 - de camp dp care se face index sau des utiles un clause WHERE ZE nr max tab < 4%

SEVENTA

↳ Struct SQL ORACLE

```

CREATE Sequence SeqName
[ step val pas ] // def = 10
{ init val val min } // 0, def = 0
[ max val valmax ]
[ CYCLE ]
[ CACHE nr-val ] NO CACHE
de se manup spum struct
Seq-name, NEXTVAL
CURVAL
  
```