7. Integritatea Bazelor de date. Constrangeri.

Securitatea si integritatea BD.

Securitatea BD – se refera la protectia datelor impotriva unor accese neautorizate.
Aceasta se realizeaza prin modalitatile prezentate in lucrarea 5:

- Definirea unor utilizatori autorizati prin **CREATE USER**
- Acordarea pentru utilizatori (**GRANT**) sau suprimarea (**REVOKE**) a anumitor **drepturi de acces sistem,** sau **pentru obiecte ale BD**
- Definirea unor grupuri de drepturi de acces (**ROLE)**
- Definirea unor vederi **(VIEW)** ale BD pentru care se acorda drepturi de acces unor grupuri de utilizatori **.** Vederile pot cuprinde numai anumite coloane sau randuri ale BD din una sau mai multe tabele

Integritatea BD – presupune protectia datelor impotriva unor distrugeri accidentale determinate de erori de program, de sistem, de echipament sau caderi de tensiune.

Starea consistenta (coerenta) a BD presupune ca toate constrangerile impuse la definirea logica a BD sunt respectate si nu exista date pierdute sau legaturi intre tabele incomplete sau incorecte (pointeri, sau referinte). Toate prelucrarile trebuie sa lase BD intr-o stare consistenta.

Secventele critice de comenzi pot apare in timpul prelucrarii si pot aduce temporar BD intr-o stare incosistenta.
Ex: - secventa de actualizare a pointerilor sau referintelor externe
- transferul unor sume dintr-un cont in altul
- secventa de prelucrare a statului de plata pentru un salariat
- secventa de marire a salariilor pentru toti angajatii

Daca apare un incident care opreste programul intr-o secventa critica, inainte de terminarea ei, BD poate ramane intr-o stare inconsistenta (pointeri sau legaturi neactualizate, mariri de salarii numai pentru o parte din angajati).
Secventele critice trebuie grupate in tranzactii marcate prin comanda **COMMIT.**

Tranzactia este o secventa de comenzi care pleaca dintr-o stare consistenta a BD si ajunge in alta stare consistenta. Intr-o tranzactie BD poate trece temporar prin stari inconsistente. Din acest motiv tranzactia

trebuie sa se faca ori complet ori deloc. Sfarsitul corect al unei tranzactii se marcheaza cu un COMMIT care face ca modificarile facute sa fie definitive. Toate valorile vechi ale inregistrarilor modificate in timpul tranzactiei se memoreaza intr-un fisier Rollback. Daca pe timpul tranzactiei apare un incident, starea BD se considera inconsistenta si reluarea prelucrarii se va face obligatoriu prin comanda ROLLBACK. Aceasta reface toate inregistrarile modificate de la inceputul tranzactiei folosind vechile valori citite din fisierul Rollback. BD revine astafel la starea consistenta de la inceputul tranzactiei. La sfarsitul unei tranzactii prin COMMIT se sterge fisierul Rollback si modificarile devin definitive.

Caracteristicile tranzactiei

Acronimul **ACID** (**A**tomicitate, **C**onsistenta, **I**zolare, **D**urabilitate) sintetizeaza conditiile care trebuie sa le indeplineasca o tranzactie.

Atomicitate – O tranzactie trebuie sa se faca ori complet terminata cu COMMIT) ori deloc cand se revine la starea de la inceputul tranzactiei prin executia ROLLBACK.

Consistenta – O tranzactie trebuie sa inceapa intr-o stare consistenta a BD si sa se termine lasand BD intr-o sare consistenta.

Izolare urmareste asigurarea consistentei citirii informatiilor din BD.
Pe timpul tranzactiei modificarile facute se vor vedea numai de user-ul care le-a executat. Inregistrarile modificate sunt blocate si nu pot fi modificate de alti useri, care ar putea determina conflicte intre modificarile facute de diferiti useri.
Modificarile facute in tranzactie vor fi vizibile pentru toti user-ii numai la terminarea tranzactiei cu COMMIT.
Pe timpul transactiei inregistrarile modificate pot fi citite de alti useri, dar valoarea lor este cea din momentul inceperii tranzactiei si este luata din fisierul Rollback.

Durabilitea – cere ca la sfarsitul tranzactiei, modificarile facute sa fie permanente (durabile), lucru care se realizeaza prin COMMIT care sterge fisierul Rollback si incepe o noua tranzactie.

Comenzi folosite pentru controlul tranzactiei.
COMMIT; - indica terminarea unei tranzactii si inceperea alteia
ROLLBACK; - determina refacerea modificarilor facute in timpul tranzactie folosind vechile valori din fisierul Rolback, aducand BD la starea consistenta de la inceputul transactiei.

SAVEPOINT nume_savepoint; - creaza un jalon in fisierul Rollback care poate fi referit in comanda ROLLBACK pentru a aduce BD in starea din momentul definirii si a nu pana la inceputul tranzactiei..

ROLLBACK TO nume_savepoint;

SET AUTOCOMMIT ON

Determina declansarea unui COMMIT dupa terminarea fiecarei comenzi. Nu se recomanda in programele profesionale ci numai pentru exercitii.
DELETE FROM stud; - sterge toate inregistrarile din tabela stud si le memoreaza in fisierul Rollback. La terminarea operatiei corect se sterge si fisierul Rollback incat comanda ROLLBACK este inefectiva.
Daca in timpul stergerii apare un incident si operatia nu s-a terminat corect, se poate da ROLLBACK si vor fi refacute inregistrarile sterse pana in momentul incidentului.

Atentie: Pot fi refacute numai modificarile realizate de comenzile DML (Delete, Insert, Update). Cele facute cu comenzile Drop sau Truncate sunt definitive fiindca informatiile nu se mai salveaza in fisierRollback.

Definirea constrangerilor in BD

Pentru a evita introducerea unor informatii eronate in BD, la definirea tabelelor se pot introduce constrangeri, care sun conditii care se verifica la orice comanda DML.
Constrangerile se pastreaza pe server in tabela ALL_CONSTRAINTS si se verifica automat, fara a fi necesare verificari in program. Se asigura astfel integritea BD.

Constrangerile pot fi:

- pe coloanele tabelei (NOT NULL, DEFAULT val)
- pe tabela (PRIMARY KEY, UNIQUE KEY, CHECK cond)
- intre tabele (FOREIGN KEY)

Constrangerile primesc nume si se pot definii la crearea tabelei sau ulterior prin ALTER TABLE.

Sintaxa comenzii CREATE TABLE cu constangeri devine:

```
CREATE TABLE owner.table
     (col1 tip1 DEFAULT expr1 constrangere1,
     col2 tip2 DEFALT expr2 constrangere2,
     …….
     CONSTRAINT nume_constr1        tip_constangere,
     …….                            );
```

Tipul constrangerii pe tabela poate fi:

PRIMARY KEY (coloana),

FOREIGN KEY (col) REFERENCES table(coloana) [CASCADE],

UNIQUE (coloana),

CHECK conditie

Ex:

```
CREATE TABLE stud
(Nume Varchar2(20) NOT NULL,
Bursa NUMBER(7) DEFAULT 0,
Cods CHAR(5) NOT NULL,
Sectie CHAR(3) NOT NULL,
CONSTRAINT c_cods UNIQUE (Cods),
CONSTRAINT c_sectie FOREIGN KEY (Sectie)
    REFERENCES Spec(Sect),
CONSTRAINT c_bursa CHECK (bursa BETWEEN 0 AND 1500),
CONSTRAINT C_nume PRIMARY KEY (Nume));
```

Modificarea constrangerilor sau invalidarea lor se poate face prin:

ALTER TABLE tab

ADD CONSTRAINT nume_constr tip_constr (coloana);

DROP

ENABLE

DISABLE

Afisati tabelele **USER_CONSTRAINTS**

USER_CONS_COLUMNS

Constrangerile se pastreaza in vederi sistem a caror structura si continut pot fi afisate.

```
DESCR USER_CONSTRAINTS
DESCR USER_CONS_COLUMNS
```

Crearea obiectelor secventa

Obiectul creat este un generator de secventa pe server accesibil tuturor useri-lor.

```
CREATE SEQUENCE Secv_cods INCREMENT BY n1 START
WITH n2 MAXVALUE n3 MINVALUE n4 CYCLE CACHE k
```

Functii: Secv.CURRVAL - valoarea curenta

Secv.NEXTVAL - valoarea urmatoare