



BAZE DE DATE

CURS 13

Limbajul de prelucrare a datelor

- *SQL* furnizează comenzi ce permit consultarea (*SELECT*) și actualizarea (*INSERT*, *UPDATE*, *DELETE*, *MERGE*) conținutului bazei de date.
- Aceste comenzi definesc limbajul de prelucrare a datelor (*LMD*).
- Comenzile limbajului *LMD* pot fi:
 - formulate direct;
 - utilizate în utilitare ale sistemului *Oracle*;
 - încapsulate într-un program *PL/SQL*;

Comanda *SELECT*

- Una dintre cele mai importante comenzi ale limbajului de prelucrare a datelor este *SELECT*;
- Cu ajutorul ei pot fi extrase submulțimi de valori atât pe verticală (coloane), cât și pe orizontală (linii) din unul sau mai multe tabele;
- Sintaxa comenzi este simplă, apropiată de limbajul natural;

Comanda **SELECT**

SELECT [ALL | DISTINCT] { * | listă de atribute selectate | expr AS alias }

FROM { [schema.]{tabel [PARTITION (partition_name)]} |
(subquery)} [alias_tabel] }

[**WHERE** condiție]

[**START WITH** condiție]

[**CONNECT BY** condiție]

[**GROUP BY** listă de expresii [**HAVING** condiție]]

[**ORDER BY** {expresie | poziție | c_alias} [**ASC** | **DESC**]] [**FOR UPDATE**
[**OF** [schema.]{table | view}.coloană] [**NOWAIT**]]

Comanda *SELECT*

- Clauzele *START WITH* și *CONNECT BY* sunt utile pentru a construi cereri ierarhizate.
 - *START WITH* -> înregistrarea rădăcină a arborelui
 - *CONNECT BY* -> relația dintre coloanele (*părinte* și *copil*)
 - Prin folosirea operatorului *PRIOR* se poate face referință la înregistrarea *părinte*.

Comanda *SELECT*

Operatorii utilizați sunt:

- operatori aritmetici (unari sau binari),
- operatorul de concatenare (`||`),
- operatorii de comparare (`=`, `!=`, `^=`, `<`, `>`, `>=`, `<`, `<=`, `IN` (echivalent cu `=ANY`, adică egal cu cel puțin una din valorile listei), `NOT IN` (echivalent cu `!=ALL`, adică diferit de toate elementele listei), `ALL`, `[NOT] BETWEEN x AND y`, `[NOT] EXISTS`, `[NOT] LIKE`, `IS [NOT] NULL`,
- operatori logici (`NOT`, `AND`, `OR`).

Comanda *SELECT*

- Limbajul permite prezența unor instrucțiuni *SELECT imbricate* în oricare din clauzele *SELECT*, *WHERE*, *HAVING* sau *FROM*.
- Instrucțiunile *SELECT* care apar în clauzele respective se numesc **subcereri**.
- În cazul folosirii subcererilor, pot fi utilizați:
 - operatorii *ALL*, *ANY*, *IN (=ANY)*, *EXIST*, *NOT IN (!=ANY)*, care sunt specifici cererilor ce returnează mai multe linii (*multiple-row subquery*)
 - operatorii de comparare *=*, *<*, *>*, *>=*, *<=*, *<>*, specifici cererilor care returnează o singură linie (*single-row subquery*).

Comanda *SELECT*

Executarea subcererilor se poate face:

- fie **cu sincronizare** (corelat -> evaluarea subcererii face referință la o coloană a cererii principale și cererea interioară se execută pentru fiecare linie a cererii principale care o conține);
- fie **fără sincronizare** (-> se execută mai întâi cererea interioară, iar rezultatul ei este transmis cererii de nivel imediat superior).

Comanda *INSERT*

INSERT INTO *nume_tabel | nume_view [(col1[, col2[,...]])]*

VALUES (*expresia1[, expresia2[,...]]*) | *subcerere*;

- *expresia1, expresia2* – reprezintă expresii a căror evaluare este atribuită coloanelor precizate (se inserează o linie);
- *subcerere* – reprezintă o interogare (se inserează una sau mai multe linii).

Comanda *INSERT*

- Dacă lipsește specificația coloanelor se consideră că sunt completeate toate câmpurile tabelului sau vizualizării.
- Dacă se introduc date doar în anumite coloane, atunci aceste coloane trebuie specificate. În restul coloanelor se introduce automat *null* (dacă nu există *DEFAULT*).
- Specificarea cererii din comanda *INSERT* determină copierea unor date dintr-un tabel în altul pe atâtea linii câte au rezultat din cerere.
- Dacă se introduc numai anumite câmpuri într-o înregistrare, atunci printre acestea trebuie să se găsească câmpurile cheii primare.
- Pentru a putea executa comanda *INSERT* este necesar ca utilizatorul care execută această instrucțiune să aibă privilegiul de a insera înregistrări în tabel sau în vizualizare.

Comanda *DELETE*

DELETE

[**FROM**] *tablename / viewname* [**AS** *alias*] [**WHERE** *condiție*]
[clauza_returning]

- Comanda *DELETE* nu sterge **structura** tabelului.
- Pentru a se putea executa instrucțiunea *DELETE*, utilizatorul care o lansează în execuție trebuie să aibă acest privilegiu.
- În clauza *WHERE* pot fi folosite și subcereri.
- Comanda nu poate fi folosită pentru stergerea valorilor unui câmp individual. Acest lucru se poate realiza cu ajutorul comenzi *UPDATE*.

Comanda *UPDATE*

UPDATE tablename | viewname

SET (column1[,column2[,...]]) = (subquery) | column = expr / (query)

[WHERE condition]

- Pentru a se putea executa instrucțiunea *UPDATE*, utilizatorul care o lansează în execuție trebuie să aibă acest privilegiu.
- Dacă nu este specificată clauza *WHERE* se vor modifica toate liniile.
- Cererea trebuie să furnizeze un număr de valori corespunzător numărului de coloane din paranteza care precede caracterul de egalitate.

LIMBAJUL PENTRU CONTROLUL DATELOR

Controlul unei baze de date cu ajutorul *SQL*-ului se refera la:

- asigurarea confidentialitatii si securitatii datelor;
- organizarea fizica a datelor;
- realizarea unor performante;
- reluarea unor actiuni in cazul unei defectiuni;
- garantarea coerentei datelor in cazul prelucrarii concurente;

LIMBAJUL PENTRU CONTROLUL DATELOR

Sistemul de gestiune trebuie:

- să pună la dispoziția unui număr mare de utilizatori o mulțime **coerentă** de date;
- să garanteze **coerența** datelor în cazul manipulării simultane de către diferiți utilizatori;

LIMBAJUL PENTRU CONTROLUL DATELOR

- Coerența este asigurată cu ajutorul conceptului de **tranzacție**.
- Tranzacția este unitatea logică de lucru constând din una sau mai multe instrucțiuni *SQL*

LIMBAJUL PENTRU CONTROLUL DATELOR

Proprietățile tranzacțiilor

- Atomicitate
- Consistență
- Izolare
- Durabilitate

->ACID

LIMBAJUL PENTRU CONTROLUL DATELOR

- Dacă toate operațiile ce constituie tranzacția sunt executate și devin efective, spunem că tranzacția este **validată**, iar modificările aduse de tranzacție devin **definitive**.
- Dacă dintr-un motiv sau altul (neverificarea condițiilor, accesul imposibil) o operație a tranzacției nu a fost executată spunem că tranzacția a fost **anulată**. Modificările aduse de toate operațiile tranzacției anulate sunt și ele **anulate** și se revine la **starea bazei de date de dinaintea tranzacției anulate**.

LIMBAJUL PENTRU CONTROLUL DATELOR

- Este posibil ca o tranzacție să fie descompusă în **subtranzacții**, astfel încât dacă este necesar să se anuleze doar parțial unele operații.
- Fiecare tranzacție se poate încheia:
 - “normal” (*commit*);
 - “anormal” (*rollback*).
- Controlul tranzacțiilor constă în:
 - definirea începutului și sfârșitului unei tranzacții;
 - validarea sau anularea acesteia;
 - eventuală descompunere în subtranzacții;

LIMBAJUL PENTRU CONTROLUL DATELOR

- Limbajul pentru controlul datelor (*LCD*) permite salvarea informației, realizarea fizică a modificărilor în baza de date, rezolvarea unor probleme de concurență.
- Limbajul conține următoarele instrucțiuni:
 - *COMMIT* - folosită pentru permanentizarea modificărilor executate asupra BD (modificările sunt înregistrate și sunt vizibile tuturor utilizatorilor);
 - *ROLLBACK* - folosită pentru refacerea stării anterioare a BD (sunt anulate toate reactualizările efectuate de la începutul tranzacției);
 - *SAVEPOINT* - folosită în conjuncție cu instrucțiunea *ROLLBACK*, pentru definirea unor puncte de salvare în fluxul programului;

LIMBAJUL PENTRU CONTROLUL DATELOR

O **tranzacție** constă:

- dintr-o singură instrucțiune *LDD*;
- dintr-o singură instrucțiune *LCD*;
- din instrucțiuni *LMD* care fac schimbări consistente în date;

LIMBAJUL PENTRU CONTROLUL DATELOR

- Tranzacția **începe**:
 - după o comandă *COMMIT*;
 - după o comandă *ROLLBACK*;
 - după conectarea inițială la Oracle;
 - când este executată prima instrucțiune *SQL*;
- Tranzacția se **termină**:
 - dacă sistemul nu mai funcționează;
 - dacă utilizatorul se deconectează;
 - dacă se dau comenzi *COMMIT* sau *ROLLBACK* ;
 - dacă se execută o comandă *LDD*;
- După ce se termină o tranzacție, prima instrucțiune *SQL* executabilă va genera automat începutul unei noi tranzacții.

LIMBAJUL PENTRU CONTROLUL DATELOR

- Din momentul în care s-a executat instrucțiunea **COMMIT**, BD s-a modificat (permanent) în conformitate cu instrucțiunile *SQL* executate în cadrul tranzacției care tocmai s-a terminat. Din acest punct începe o nouă tranzacție.
- Dacă se folosește utilitarul *SQL*Plus*, există posibilitatea ca după fiecare comandă *LMD* să aibă loc o permanentizare automată a datelor (un *COMMIT implicit*). Acest lucru se poate realiza folosind comanda:

SET AUTO[COMMIT] {ON | OFF}

- Comanda **ROLLBACK** permite restaurarea unei stări anterioare a BD.

ROLLBACK [TO [SAVEPOINT] savepoint];

LIMBAJUL PENTRU CONTROLUL DATELOR

- Dacă nu se specifică nici un *savepoint*, toate modificările făcute în tranzacția curentă sunt anulate;
- Dacă se specifică un anumit *savepoint*, atunci doar modificările de la acel *savepoint* până în momentul respectiv sunt anulate;
- Executarea unei instrucțiuni *ROLLBACK* presupune terminarea tranzacției curente și începerea unei noi tranzacții;

LIMBAJUL PENTRU CONTROLUL DATELOR

- Punctele de salvare pot fi considerate ca niște etichete care referă o submulțime a schimbărilor dintr-o tranzacție, marcând efectiv un punct de salvare pentru tranzacția curentă. Punctele de salvare **NU sunt obiecte ale schemei**. Prin urmare, nu sunt referite în DD.
- Server-ul *Oracle* implementează **un punct de salvare implicit** pe care îl mută automat după ultima comandă *LMD* executată.
- Dacă este creat un punct de salvare având același nume cu unul creat anterior, cel definit anterior este șters automat.

SAVEPOINT savepoint;

LIMBAJUL PENTRU CONTROLUL DATELOR

Starea datelor înainte de *COMMIT* sau *ROLLBACK* este următoarea:

- starea anterioară a datelor poate fi recuperată;
- utilizatorul curent poate vizualiza rezultatele operațiilor *LMD* prin interogări asupra tabelelor;
- alții utilizatori nu pot vizualiza rezultatele comenzielor *LMD* făcute de utilizatorul curent (***read consistency***);
- înregistrările (liniile) afectate sunt blocate și, prin urmare, alții utilizatori nu pot face schimbări în datele acestor înregistrări.

LIMBAJUL PENTRU CONTROLUL DATELOR

Execuția unei comenzi *COMMIT* implică anumite modificări.

- Toate schimbările (*INSERT*, *DELETE*, *UPDATE*) din baza de date făcute după anterioara comandă *COMMIT* sau *ROLLBACK* sunt **definitive**. Comanda se referă numai la schimbările făcute de utilizatorul care dă comanda *COMMIT*.
- Toate punctele de salvare vor fi sterse.
- Starea anterioară a datelor este pierdută definitiv.
- Toți utilizatorii pot vizualiza rezultatele.
- Blocările asupra liniilor afectate sunt eliberate; liniile pot fi folosite de alții utilizatori pentru a face schimbări în date.

LIMBAJUL PENTRU CONTROLUL DATELOR

Execuția unei comenzi *ROLLBACK* implică anumite modificări.

- **Anulează** tranzacția în curs și toate modificările de date făcute după ultima comandă *COMMIT*.
- Sunt **eliberate** blocările liniilor implicate.