1. Ce probleme ridică programarea în contextul bazelor de date (două)?

Programarea în contextul BD este centrată în jurul unor limbaje de programare limitate. Își găsesc sens doar în aria unde se pot utiliza limbaje „non-Turing-complete”. Conduce la o programare foarte succintă.

Problemele care apar:

1. Controlul concurenței – Multe activități (tranzacții) cu baza de date se desfășoară la același moment de timp și acționează asupra acelorași bucăți de informație.

2. Optimizarea interogărilor – Volum mare de date.

2. Explicați funcționalitatea de transformare a datelor pentru un Sistem de Gestiune de Baze de Date.

Datele introduse de utilizator nu au întotdeauna structura identică cu cea definită în baza de date

3. Nivelul conceptual de abstractizare al unui Sistem de Gestiune de Baze de Date.

Există 3 nivele de abstractizare în SGBD:

1. nivel extern (Utilizator 1, 2…n)
2. nivel conceptual (BD logică – schema BD)
3. nivel intern (BD fizică)

Schema conceptuală a BD sau **schema logică**, descrie datele stocate în BD în termeni ai modelului de date al SGBD

- Pentru un SGBD relațional, schema conceptuală descrie toate tabelele (relațiile) stocate în BD.
- Limbajul folosit de numește Limbaj de Descriere a Datelor.

4. Descrieti pe scurt modelul de aplicatie trei-straturi

Arhitectura client-server pe „trei straturi“ presupune faptul că fiecare strat este rulat pe o platformă diferită.

• stratul **Client (logica de prezentare)** este format din interfaţa cu utilizatorul, care este rulat pe calculatorul utilizatorului final (Internet Explorer, Mozilla Firefox, Google Chrome)

• stratul **Server de aplicaţii (Web) (Logica aplicației),** ce manevrează logica aplicaţiilor şi prelucrării datelor, şi care poate servi mai mulţi clienţi (conectare la celelalte două straturi se face prin reţele locale LAN sau de mare suprafaţă WAN);

• stratul **Server-ul de baze de date(SGBD),** care se ocupă cu validarea datelor şi accesarea bazei de date (stochează date necesare stratului din mijloc).

Arhitectura se potriveşte natural mediului Web. Un browser Web acţionând drept client şi un server Web fiind server de aplicaţie. Middleware este un strat, evident software, între aplicaţia postului client şi server-ul de baze de date, constituit dintr-o interfaţă de programare a aplicaţiilor (API - Application Programming Interface) şi un protocol de reţea. API descrie tipul de interacţiune dintre o aplicaţie client şi un server la distanţă, via un protocol de comunicaţie şi de formatare a datelor. Scopul existenţei interfeţei de programare a aplicaţiilor este de a oferi o interfaţă unică mai multor server-e de baze de date. Este convenabil ca sistemele de baze de date să fie considerate ca fiind formate dintr-un server (sistemul SGBD însăşi) şi un set de clienţi (aplicaţiile). Frecvent, clienţii şi server-ul pot fi rulate pe calculatoare diferite, realizându-se un tip simplu de procesare distribuită. În general, fiecare server poate deservi mai multi clienţi, iar fiecare client poate accesa mai multe server-e. Dacă sistemul oferă transparenţă totală (fiecare client se poate comporta ca şi cum ar comunica cu un singur server, de pe un singur calculator) atunci este vorba despre un sistem de baze de date distribuite.

5. Care este diferenta intre „row-level triggers” și „statement-level triggers?

- Trigerele sunt “row-level” sau “statementlevel”.
- FOR EACH ROW indică “row-level”; dacă lipseşte indică “statement-level”.

➢ **Row level triggers :** se execută o singură dată pentru fiecare tuplă modificată.

➢ **Statement-level triggers :** se execută o singură dată pentru o instrucţiune SQL, indiferent de numărul tuplelor modificate.

Instrucţiunile INSERT implică o tuplă nouă (pentru “row-level”) sau o tabelă nouă (pentru “statement-level”). “tabela nouă” este mulţimea tuplelor inserate

6. Explicați CLI.

În principiu, un preprocesor translatează instrucțiunile SQL în apeluri de procedură ce se încadrează în codul limbajului gazdă. O altă abordare pentru conectarea la BD a limbajelor convenţionale este apelul prin intermediul bibliotecilor.

CLI, alături de **JDBC, PHP/DB,** reprezintă **unelte de conectare** pentru a permite unui limbaj convențional să acceseze o bază de date.

În loc să se folosească un preprocesor (embedded SQL), se poate folosi o bibliotecă cu funcții. Biblioteca pentru C este numită SQL/**CLI („Call-Level Interface”).** Preprocesorul pentru embedded SQL translatează instrucțiunile EXEC SQL … în **CLI** sau apeluri similare.

7. Explicați R semi-join S folosind un exemplu.

Semi-join se utilizează de către SGBD-uri pentru baze de date distribuite și se notează **R ⋉ S.** Se presupune că cele două relaţii A şi B ce trebuie cuplate se găsesc la locaţii diferite.

Pentru optimizarea traficului în reţea este mai convenabil să se transporte doar mulţimea de valori corepunzătoare atributului de join de la un site la altul, se efectuează join şi la urmă probabil la o a treia locaţie se transportă fracţiunile din cele două relaţii pentru a obține rezultatul final.

După notaţia **R ⋉ S** se transportă mai întâi de la locaţia de reşedinţă a locaţiei **S** la locaţia de reşedinţă a relaţiei **R.** De exemplu, **R** este **Student (nr_matr, gen, coda)** şi B este **Nota (nr_matr, code, nota).** Se transportă doar nr_matr din Nota la locaţia unde se găseşte student. Interogările SQL se utilizează cu operatorii **EXISTS sau IN.**

De exemplu, dacă dorim să aflăm studenţii care au cel puţin o notă:

```
SELECT   R.nume
FROM student R
WHERE EXISTS
        (SELECT 1
        FROM   nota S
        WHERE  R.nr_matr = S.nr_matr)
ORDER BY R.nr_matr;
```

8. Explicați tranzacțiile Repeatable-Read

În cadrul unei tranzacții, se poate preciza nivelul de izolare (serializable, repeatable read, read commited, read uncommited).

La citirea unei resurse, **tranzacția solicită o blocare S pe resursa respectivă, dar blocarea S se eliberează doar la finalul tranzacției** ⇒ o altă tranzacție nu mai poate modifica datele citite de tranzacția curentă, înainte de finalizarea acesteia.

O altă tranzacție ar putea insera un rând în intervalul de tupluri citite de tranzacția curentă, înainte de finalizarea acesteia.

Presupunem că Sally execută cu REPEATABLE READ, şi ordinea de execuţie este **(max)(del)(ins)(min).**

(max) vede preţurile 2.50 şi 3.00.

(min) poate vedea 3.50, dar de asemenea vede **2.50** şi **3.00**, deoarece aceste preţuri au fost văzute la prima citire de (max).

9. Ce indică simbolurile *, +, ? la descrierea elementelor unui document XML cu DTD?

Un tag poate fi urmat de un simbol ce indica multiplicitatea.

*** = zero sau mai multe.**

+ = unu sau mai multe.

? = zero sau unu.

10. Care sunt caracteristicile pentru alegerea modelului NOSQL?

Înainte de alegerea modelului NoSQL(Not Only SQL), e nevoie să se desfășoare următoarele analize:

Analiza datelor care urmează să fie stocate.

Dacă sunt date de tip „high volume, low value”, NoSQL este o alegere mai bună.

Analiza schemei de aplicație.

Dacă este dinamică, atunci NoSQL este o alegere mai bună.

În NoSQL

- se operează cu date simple de tip „low-value, low-density”,
- relațiile sunt simple,
- se evită JOIN-urile,

- datele nu au o schemă, sunt nestructurate sau semi-structurate
- stocarea și procesarea sunt distribuite
- standardele nu sunt încă evoluate
- centrare pe aplicație și pe developeri (application-centric și developer-centric)

11. Care sunt operatorii primitivi din algebra relațională? Argumentați

Reuniunea, Diferența, Produsul cartezian, Selecția și Proiecția

Selecția:

- **R1 := σ_C (R2)**
- C este o condiţie (asemănător cu instrucţiunea “if”) ce face referire la atributele din R2.
- R1 conţine acele tuple din R2 ce satisfac C

Proiecția:

- **R1 := π_L (R2)**
- L este o listă de atribute din schema relaţiei R2.
- R1 este construită în felul următor:
  • Mai întâi pentru fiecare tuplă din R2 se extrag atributele din lista L, în ordinea specificată.
  • Se elimină tuplele duplicat, dacă există.

Produs Cartezian:

- **R3 := R1 X R2**
- Face pereche între fiecare tuplă t1 din R1 şi fiecare tuplă t2 din R2. tuplă din R3 se obţine prin **concatenarea t1t2.**
- Schema relaţiei R3 este constituită din atributele din R1 şi apoi din R2, în ordine.
- **Atenţie** la **atributele cu acelaşi nume** în R1 şi R2, de exemplu A : se foloseşte exprimarea **R1.A şi R2.A.**

12. Care este diferența între Forma Normală 3 și FNBC?

- În FNBC, **pentru orice relație A->B, A trebuie să fie o supercheie a relației**. În FN3 nu ar trebui să existe nicio dependență tranzitivă, adică niciun atribut neprimar nu ar trebui să depindă în mod tranzitoriu de cheia candidată.
- FNBC este mai puternică decât FN3
- În FNBC, dependențele funcționale sunt deja în FN1, FN2, și FN3. În FN3, dependențele funcționale se regăsesc deja în FN1 și FN2.
- **Redundanța** în FN3 este semnificativ mai ridicată în comparație cu FNBC.
- Este mult **mai dificil să se ajungă la FNBC** prin comparație cu FN3.
- **Descompunerea fără pierderi este mult mai greu de realizat în pentru FNBC** în comparație cu FN3

13. Explicaţi ID, IDREF si IDREFS în contextul unui document XML.

Atributele pot fi pointeri de la un obiect la altul. Se permite structurii unui document XML sa fie un graf, în loc de arbore.

Presupunem un element E cu un **atribut A de tip ID**. Atunci când se foloseste tag-ul <E > într-un document XML, atributul A primeste o valoare unica.

Exemplu: <E A = ”xyz”>

Pentru a permite elementelor de tip F sa faca referire la un alt element cu un atribut ID, F primeste un atribut de tip **IDREF**.

Atributul are tipul **IDREFS,** astfel încât elementul F poate face referire la oricâte alte elemente.

14. Ce proprietăţi (restricţii) trebuie să întrunească o formulă pentru a fi „sigură” în calculul relaţional?

Independenţa de domeniu a unei formule F este o problemă indecidabilă.

Formule sigure ⊂= Formule independente de

domeniu.

Proprietăţi:

- Orice formulă sigură este formulă independentă de domeniu.
- Fiind dată o formulă F, se poate spune dacă este sau nu sigură.
- Orice interogare din algebra relaţională poate fi exprimată cu ajutorul formulelor sigure.

Restrictii:

- formula NU contine ∀ (transformarea de echivalenta (∀X)F≡¬(∃X)¬F
- F1∨F2 => F1(X1,X2,...,Xn) ∧ F2(X1,X2,...,Xn) unde (X1, X2, ..., Xn) sunt variabile libere
- orice variabila libera din F1 ∧ F2 ∧ ...∧ Fm trebuie sa fie variabila limitata:
    - variabila libera în cel putin o F ce nu este comparatie aritmetica sau negatie
    - F este de forma X = a sau a = X , unde a este o constanta
    - F este de forma X = Y sau Y = X , Y este limitata
- în F1 ∧F2 ∧ ...∧ Fm cel putin o Fi trebuie sa fie pozitiva

15. Descrieţi proprietăţile unei descompuneri.

1. Cuplarea fără pierdere de informaţii : relaţiile originale trebuie să poată fi obţinute din schema descompusă, adică să se reconstruiască originalul.

2. Conservarea DF-le : relaţiile obţinute prin descompunere trebuie să satisfacă toate DF-le iniţiale.

Se poate obţine (1) la o descompunere FNBC.

Se pot obţine ambele (1) şi (2) la o descompunere FN3.

Nu totdeauna este posibil să se obţină ambele (1) şi (2) la o descomunere FNBC.

16. Care este diferenţa între Schema XML si DTD?

- Schema XML ofera o cale mai puternica pentru a descrie structura documentelor XML.
- Declaratiile Schema-XML sunt ele însele documente XML . Ele descriu “elemente” iar descrierea se face de asemenea cu “elemente”.
- Un DTD specifica ce elemente pot aparea si cum elementele se pot imbrica intr-un document XML care este conform cu acest DTD.
- Dacă se declară standalone = „yes”, înseamnă că documentul XML este fără DTD, dacă se declară standalone = „no”, înseamnă că se folosește DTD.
- Tag-urile XML pot avea atribute.

Într-un DTD, <!ATTLIST E . . . > declară atributele elementului E, împreună cu tipurile de date.

17. Care este diferenţa între R natural-join S si R theta-joinC S, ce are condiţia C: R.A=S.A pentru fiecare atribut A ce apare în ambele scheme R si S?

Pentru theta join se face produsul R1xR2 dupa care se aplica selectia σC rezultatului care afisaza doar rezultatele care indeplinesc conditia C. La Theta join nu trebuie sa existe atribute cu acelasi nume. La natural join se egalizeaza atributele cu acelasi nume dupa care se face proiectia unei singure copii a fiecarui atribut pereche.

18. Definiţi dependenţa funcţională. Daţi exemple.

X ->Y este o declaratie pentru relatia R astfel încât oricând doua tuple din R au aceleasi valori pentru toate atributele ce formeaza X, atunci valorile din cele doua tuple pentru toate atributele din setul Y trebuie sa coincida.

Exemplu: Drinkers(name, addr, beersLiked, manf, favBeer)

Se poate gândi ca:

1. name -> addr favBeer
2. beersLiked -> manf

19. Ce este o tranzacţie ACID?

Tranzacţiile ACID sunt tranzacții cu următoarele proprietăți:

Atomicitate : “Totul sau nimic”.

Consistenţă : Constrângerile BD să fie respectate.

Izolare : Utilizatorul vede ca şi cum la un moment dat de timp se execută un singur proces.

Durabilitate : Efectele unui proces “supravieţuiesc” unei căderi a sistemului.

Opţional: deseori sunt susţinute forme mai slabe de tranzacţii.

20. Descrieți funcționalitatea de control a accesului concurent la date pentru un Sistem de Gestiune de Baze de Date.

Fiecare utilizator are impresia că lucrează de unul singur (serializarea operaţiilor)

21. Câte nivele de independență a datelor cunoașteți pentru un Sistem de Gestiune de Baze de Date?

Aplicaţiile sunt izolate faţă de modificările la nivel conceptual sau la nivel fizic prin cele trei nivele de abstractizare

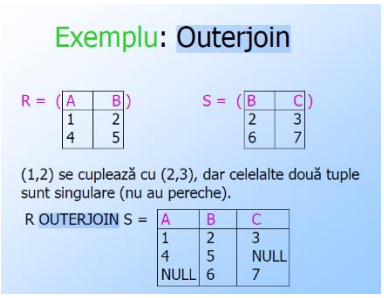
Independenţa logică a datelor

Vederile (view în modelul relaţional, tabelă virtuală, schema externă) asigură posibilitatea modificării structurii datelor (schema conceptuală), acest lucru fiind ascuns aplicaţiilor

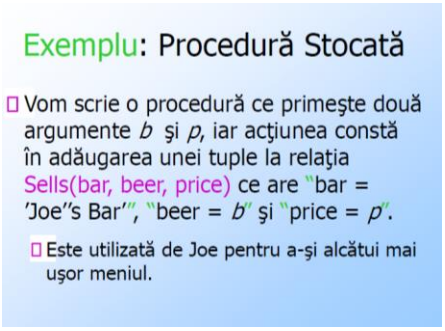
Independenţa fizică a datelor

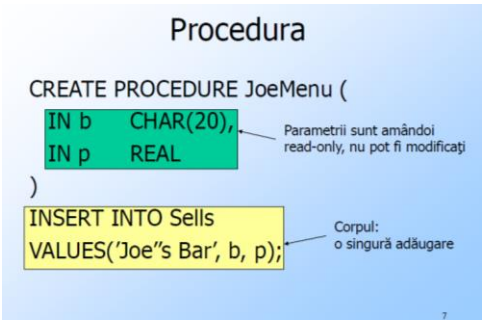
Schema conceptuală asigură posibilitatea modificării aranjării datelor pe suport secundar sau a indecşilor, acest lucru fiind de asemenea ascuns aplicaţiilor

22. Dați un exemplu de OUTER-JOIN la SQL.



23. Dați un exemplu de procedură PSM (Persistent Stored Modules)

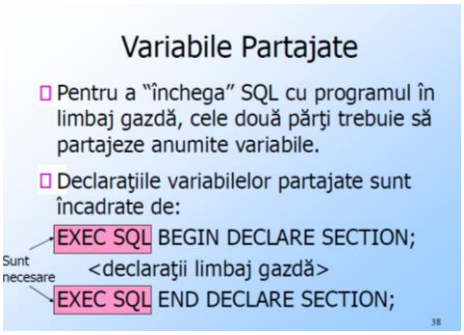


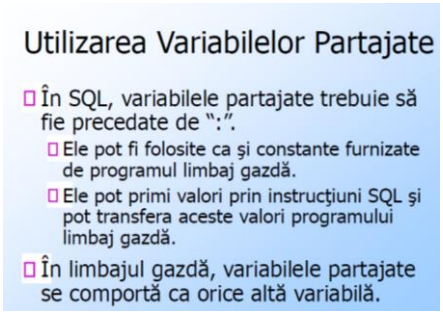


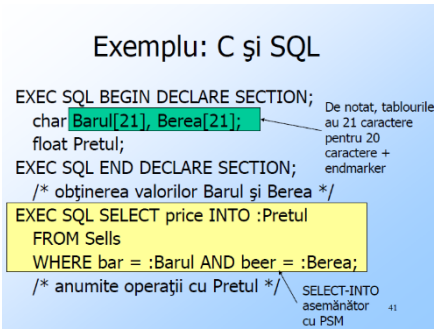
24. Care este mecanismul embedded SQL?

Ideea de bază: Un preprocesor translatează instrucţiunile SQL în apeluri de procedură ce se încadrează în codul limbajului gazdă.

Toate instrucţiunile "embedded SQL" încep cu EXEC SQL, în aşa fel încât preprocesorul să le descopere cu uşurinţă.

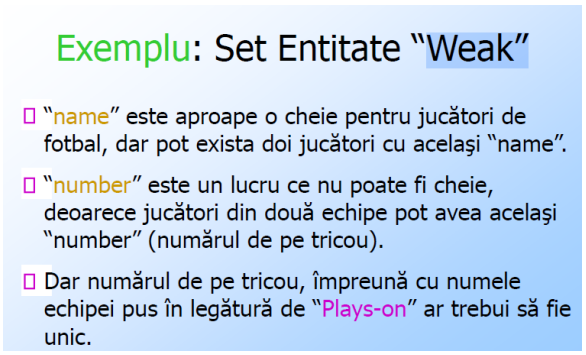


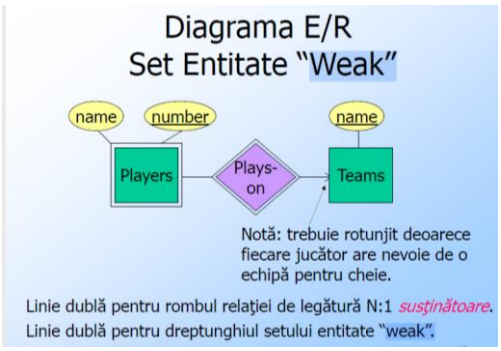




25. Definiți entitatea „weak" în contextul diagramei ER.

Setul entitate E se spune că este "weak" dacă pentru a **identifica unic entităţile din E,** este nevoie să se urmărească **relaţii de legătură "many-one"** plecând de la E şi **să se includă cheia entităţilor din seturile entitate conectate**.

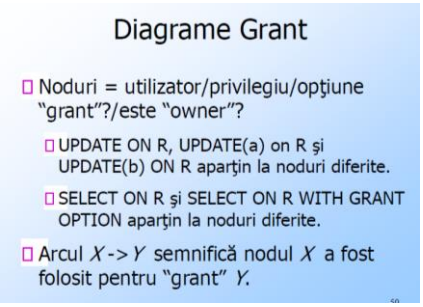


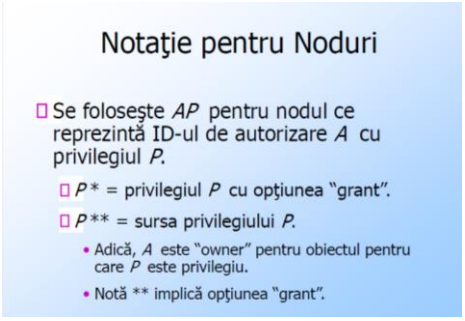


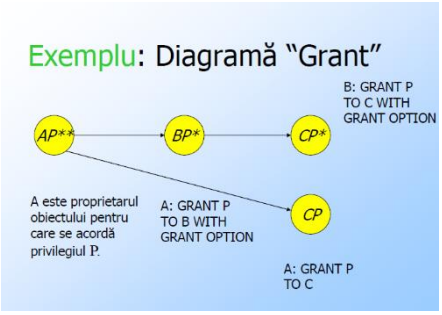
26. Ce este o diagramă „GRANT”? Dați un exemplu.

Acordarea Privilegiilor: Pentru obiectele (de exemplu relaţiile) create de un utilizator, acesta are toate privilegiile posibile.

- Se pot acorda privilegii altor utilizatori (ID-uri de autorizare), inclusiv PUBLIC.
- Se pot acorda privilegii cu clauza **WITH GRANT OPTION**, ce permite celui autorizat să acorde mai departe privilegii.

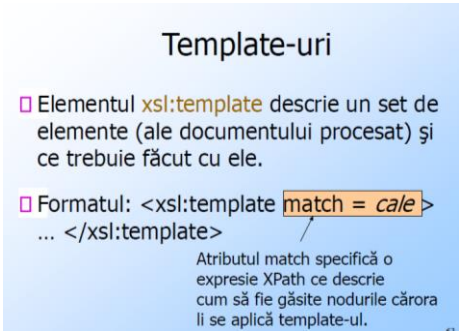






27. Descrieți XSLT.

- **XSLT (extensible stylesheet language – transforms)** este un alt limbaj de procesare a documentelor XML.
- La origine era intenţionat ca un limbaj pentru prezentare: să transforme XML într-o pagină HTML care să fie afişată.
- Dar poate de asemenea să transforme XML -> XML, astfel servind ca un limbaj de interogare.
- Ca şi Schema XML, un program XSLT este el însuşi **un document XML.**
- XSLT are un namespace special de taguri, de obicei indicat prin xsl:.



28. Descrieți funcționalitatea de asigurare a integrității datelor pentru un Sistem de Gestiune de Baze de Date.

Restricţii de integritate, de ex. SGBD-ul poate asigura că vârsta unei persoane la introducerea în BD este cuprinsă între 18 şi 40 ani

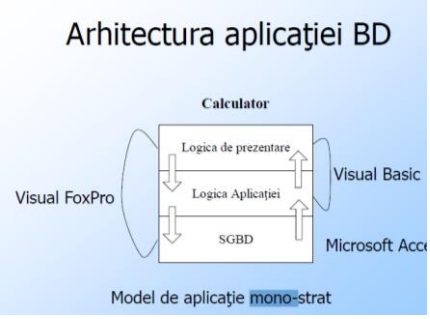
29. Limbajul de manipulare a datelor.

Un limbaj de manipulare a datelor (LMD) este un limbaj de programare utilizat pentru **adăugarea (inserarea), ștergerea și modificarea (actualizarea)** datelor într-o bază de date. Un DML este adesea un sublimbaj al unui limbaj de baze de date mai larg, cum ar fi SQL, DML-ul cuprinzând o parte din operatorii limbajului.

30. Descrieți pe scurt modelul de aplicație mono-strat.

Arhitectura pe un singur nivel presupune plasarea tuturor componentelor necesare pentru o aplicație software pe un singur server sau platformă. Acest tip de arhitectură este adesea în contrast cu arhitectura pe mai multe niveluri sau cu arhitectura pe trei niveluri, utilizată pentru unele aplicații Web, în care diferitele niveluri de prezentare, de afaceri și de acces la date sunt găzduite separat.

Practic, o arhitectură pe un singur nivel păstrează toate elementele unei aplicații, inclusiv interfața, middleware-ul și datele din back-end, într-un singur loc. Dezvoltatorii consideră că aceste tipuri de sisteme sunt cele mai simple și mai directe. Unii experți le descriu ca fiind aplicații care ar putea fi instalate și rulate pe un singur calculator. Nevoia de modele distribuite pentru aplicațiile Web și soluțiile de găzduire în cloud a creat multe situații în care arhitecturile cu un singur nivel nu sunt suficiente. Acest lucru a făcut ca arhitecturile pe trei niveluri sau pe mai multe niveluri să devină mai populare. Beneficiile unei soluții multi-tier sunt adesea evidente. Acestea pot oferi o mai bună securitate, o performanță mai bună și o mai mare scalabilitate, precum și medii individuale pentru centrele de date și aplicațiile front-end. Cu toate acestea, atractivitatea unei arhitecturi pe un singur nivel poate fi legată de costurile implicate, în cazul în care ar putea fi mai logic să păstrezi aplicațiile mai simple cuprinse într-o singură platformă ușoară.



31. Explicați cum funcționează o interogare SQL pe două relații.

Interogările combină adesea datele din mai multe relaţii.

Se pot menţiona mai multe relaţii într-o interogare prin introducerea lor în clauza FROM.

Se face distincţie între atributele cu acelaşi nume prin prefixare cu numele relaţiei:

"<relaţie>.<atribut>"

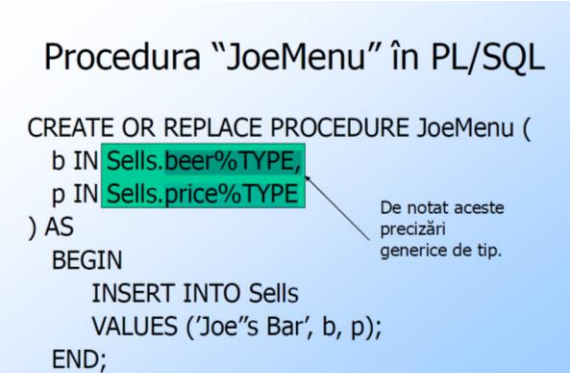
Semanticile Formale pentru o interogare cu mai multe relaţii sunt aproximativ aceleaşi ca şi în cazul interogării cu o singură relaţie:

1. **Se începe cu produsul relaţiilor** din clauza **FROM**.
2. **Se aplică selecţia** impusă de condiţia din clauza **WHERE.**
3. **Se aplică proiecţia extinsă** pe lista de atribute şi expresii din **clauza SELECT.**

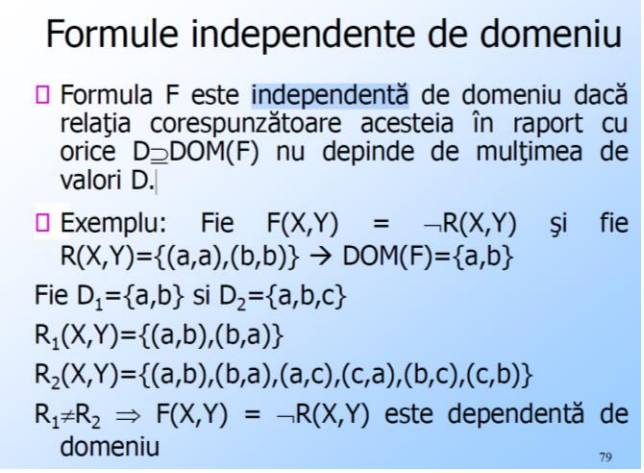
32. Poti fi actualizate datele din baza de date prin intermediul unei vederi? Explicați răspunsul dat.

Da, însă actualizările prin intermediul vederilor pot fi efectuate prin definiţia de trigere de tip **INSTEAD OF** ce **translatează actualizările în operaţii asupra tabelelor de bază.**

33. Dați un exemplu de precizare generică de tip în PL/SQL. La ce este utilă o astfel de precizare?



34. Definiți formula independentă de domeniu în calculul relațional.



35. Descrieți cum se obține schema relațională din diagrama ER pentru subclase de entități. (???)

Set entitate -> relatie.

-Atribute -> atribute.

Relatii de legatura -> relatii cu urmatoarele atribut:

-Cheile seturilor entitate conectate.

-Atributele relatiei de legatura (daca exista).

36. Explicați tranzacțiile Read-Commited.

- Nivelul de izolare al setului de tranzacții de tip Read Commited este **un nivel de izolare implicit** în SQL SERVER.
- La citirea unei resurse, **tranzacția solicită o blocare S pe resursa respectivă**, blocare care este eliberată la finalul operației.
- Astfel, **tranzacția nu mai poate vedea modificări not commited efectuate de altă tranzacție.** O altă tranzacție poate să modifice datele citite de tip tranzacția curentă, înainte de finalizarea acesteia.
- Dacă Sally = (max)(min) şi Joe = (del)(ins) sunt fiecare tranzacţii, iar Sally execută cu nivelul de izolare READ COMMITTED, atunci ea vede doar datele “committed”, dar nu neapărat aceleşi date de fiecare dată. Exemplu: Cu READ COMMITTED, intercalarea (max)(del)(ins)(min) este permisă atât timp cât Joe face “commit”. Sally vede MAX < MIN.

37. Enumerați funcțiile unui Sistem de Gestiune de Baze de date.

Gestiunea dicţionarului de date

Programele aplicative accesează datele prin intermediul SGBD, care caută în dicţionarul de date structura datelor şi a legăturilor necesare rezolvării problemei utilizatorului

Gestiunea fişierelor de date

Colecţia de date (BD) se materializează printr-un fişier sau colecţie de fişiere de date, fişiere index, etc.

Transformarea datelor

Datele introduse de utilizator nu au întotdeauna structura identică cu cea definită în baza de date

Gestiunea aplicaţiilor

Limbaj de descriere a datelor

Limbaj de manipulare a datelor

Limbaj pentru afişarea datelor (pe ecran sau la imprimantă)

Importul şi exportul datelor

Conversia datelor pentru prelucrarea cu alt SGBD sau cu aplicaţii terţe (de ex. Excel)

Controlul securităţii datelor

Utilizatorii ce au acces la date

La ce date are acces fiecare utilizator

Ce operaţii se pot efectua de fiecare utilizator cu datele la care are acces

Asigurarea integrităţii datelor

Restricţii de integritate, de ex. SGBD-ul poate asigura că vârsta unei persoane la introducerea în BD este cuprinsă între 18 şi 40 ani

Controlul accesului concurent la date

Fiecare utilizator are impresia că lucrează de unul singur (serializarea operaţiilor)

Gestiunea copiilor de siguranţă şi a recuperării datelor în caz de dezastre

38. Limbajul de descriere a datelor.

Limbajul de descriere a datelor (LDD) este partea din SQL care se ocupă de modul în care datele ar trebui să se afle în baza de date la nivel logic. Fiecare bază de date are propriul set de tipuri de obiecte pe care le permite. Cele mai multe includ tabele, indici, vizualizări, proceduri de stocare, funcții, sinonime și trigere. Fiecare bază de date are o sintaxă proprie pentru declarațiile LLD și pentru clauzele care pot fi incluse. Există câteva cuvinte cheie de bază pe care le vom găsi în aproape fiecare RDBMS. – CREATE, ALTER, DROP, TRUNCATE

39. Descrieți pe scurt categoriile de utilizatori pentru un Sistem de Gestiune de Baze de date.

1. Implementatori BD

- Scriu software SGBD (soft de bază)

2. Programatori de Aplicaţii BD

- Dezvoltă pachete de programe ce facilitează accesul la date al utilizatorilor finali.
- Folosesc “host languages” sau “data languages “ şi unelte software
- La modul ideal aplicaţiile BD lucrează prin schema externă, dar este posibil să acceseze datele şi la nivelele de mai jos când este posibil să fie compromisă independenţa datelor

3. Utilizatori finali

- Înregistrează şi interoghează BD, de obicei prin intermediul aplicaţiilor
- În marea majoritate a cazurilor nu sunt specialişti în domeniul calculatoarelor (de exemplu un agent comercial)

4. Administratorul BD

- Al patrulea și cel mai important utilizator : pentru BD de tip “corporate” sau “enterprise-wide” este activul cel mai important al companiei

  Principalele sarcini:

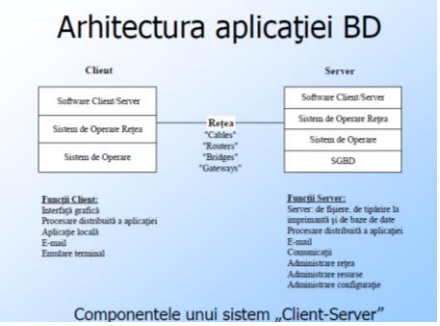
  Să proiecteze schema conceptuală şi schema fizică

  Să implementeze politici de securitate şi autorizare

  Să asigure disponibilitatea datelor şi să recupereze datele în cazul apariţiei de incidente

  Să efectueze “database tuning” (reglarea parametrilor pentru performanţă

40. Modelul de aplicație client-server.



41. Dați exemple de utilizare a operatorilor SQL „set"(pe mulțimi)

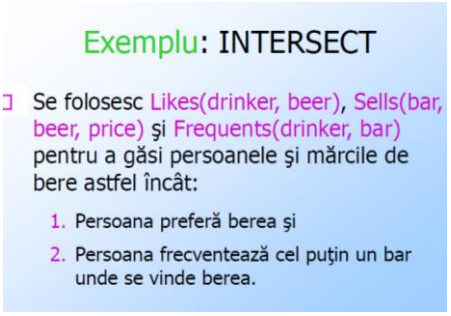
Operatori „set" (UNION/INTERSECT/EXCEPT)

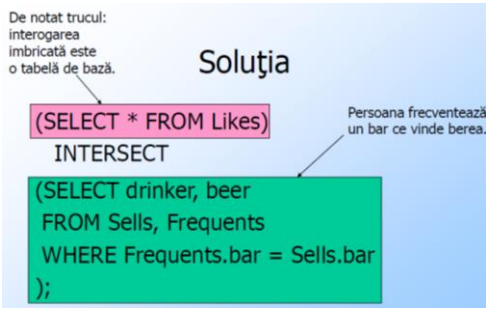
Uniunea, intersecţia şi diferenţa relaţiilor sunt exprimate prin formele următoare, fiecare implică interogări imbricate:

(<subquery>) UNION (<subquery>)

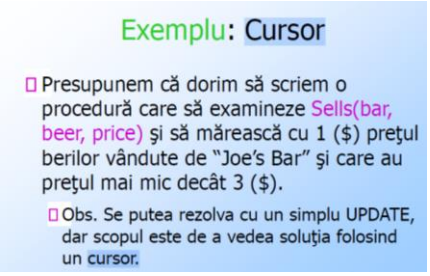
(<subquery>) INTERSECT (<subquery>)

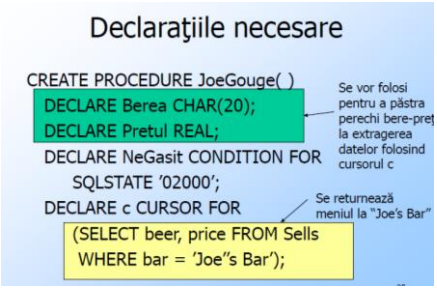
(<subquery>) EXCEPT (<subquery>)

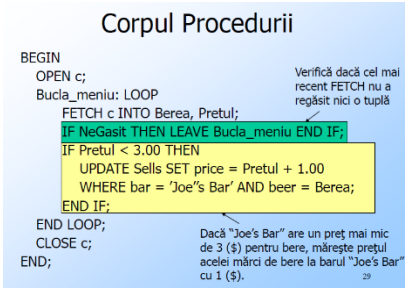




42. Dați un exemplu de utilizare cursor în PSM.







43. Explicați pe scurt utilizarea JDBC.

În principiu, un preprocesor translatează instrucțiunile SQL în apeluri de procedură ce se încadrează în codul limbajului gazdă. O altă abordare pentru conectarea la BD a limbajelor convenţionale este apelul prin intermediul bibliotecilor.

JDBC, alături de **CLI, PHP/DB,** reprezintă **unelte de conectare** pentru a permite unui limbaj convențional să acceseze o bază de date.

În loc să se folosească un preprocesor (embedded SQL), se poate folosi o bibliotecă cu funcții.

Biblioteca pentru JAVA este numită **JDBC („Java Database Connectivity").**

JDBC oferă două clase:

1. **Statement** = un obiect ce poate accepta un string ce este o instrucţiune SQL şi poate executa un astfel de string.
2. **PreparedStatement** = un obiect ce are o instrucţiune SQL asociată, ce este gata de execuţie.

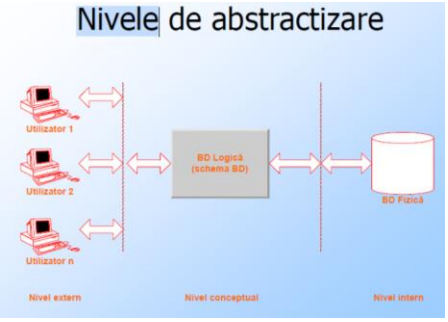
JDBC face **distincţie între interogări şi actualizări**, pe care le numeşte "updates."

"Statement" şi "PreparedStatement" au fiecare metodele **executeQuery** şi **executeUpdate.**

Pentru **"Statements":** un argument – interogarea sau actualizarea ce trebuie executată.

Pentru **"PreparedStatements":** nici un argument.

44. Câte nivele de abstractizare există la un sistem de Gestiune de Baze de Date?



45. Explicați funcționalitatea privind controlul securității datelor pentru un SGBD.

Controlul securităţii datelor

Utilizatorii ce au acces la date

La ce date are acces fiecare utilizator

Ce operaţii se pot efectua de fiecare utilizator cu datele la care are acces

46.Explicați diferența între ghilimele și apostrof în PHP.

Atât prin utilizarea ghilimelelor, cât și prin apostrof, codul PHP este forțat să evalueze întregul șir de caractere. Principala diferență între ghilimelele duble și ghilimelele simple este că, prin utilizarea ghilimelelor, putem include variabile direct în interiorul șirului. Se interpretează secvențele Escape

47. Inner Join, Natural Join, Theta Join

Inner Join unește două tabele pe baza coloanei care este specificată în mod explicit în clauza ON. Tabelul rezultat va conține toate atributele din ambele tabele, inclusiv coloana comună.

```
SELECT *
FROM student S INNER JOIN Marks M ON S.Roll_No = M.Roll_No;
```

Natural Join unește două tabele pe baza aceluiași nume de atribut și a acelorași tipuri de date. Tabelul rezultat va conține toate atributele ambelor tabele, dar va păstra doar o singură copie a fiecărei coloane comune.

```
SELECT *
FROM Student NATURAL JOIN Marks;
```

Theta Join unește două tabele pe baza condiției C. Îmbinările Theta funcționează pentru toți operatorii de comparație. Este desemnată prin simbolul θ. Cazul general al operațiunii JOIN se numește Theta join.

$R3 := R1 (\bowtie_C) R2$

48. Explicați care sunt violările posibile după definirea unei chei străine.

Dacă există o constrângere cheie străină de la relaţia R la relaţia S, două violări sunt posibile:

1. La o adăugare (insert) sau modificare (update) a unei tuple în R se introduc valori ce nu se regăsesc în S.
2. După o ştergere (delete) sau modificare (update) a unei tuple în S rezultă anumite tuple inconsecvente în R (nu au pereche în S).

49. Descrieți pe scurt relația dintre PSM și SQL.

- Codul scris într-un limbaj specializat este stocat în BD (de exemplu PSM)
- PSM, sau “persistent stored modules,” permit stocare de proceduri ca elemente ale schemei BD.
- PSM = un amalgam de instrucţiuni convenţionale (if, while, etc.) şi SQL.
- Oferă posibilităţi altfel inexistente în SQL.

50. Explicați cum se utilizează tablourile asociative în PHP.

Elementele unui tablou asociativ $a sunt perechi x => y, unde x este un şir de caractere cheie şi y este orice valoare.

Dacă x => y este un element al lui $a, atunci $a[x] este y.

Un mediu poate fi exprimat ca un tablou asociativ:

$med_Meu = array(

”phptype” => ”mySql”,

”hostspec” => ”localhost”,

”database” => ”scoala”,

”username” => ”student”,

”password” => ”nuSeStie”);

Se importă biblioteca DB şi se foloseşte tabloul $med_Meu:

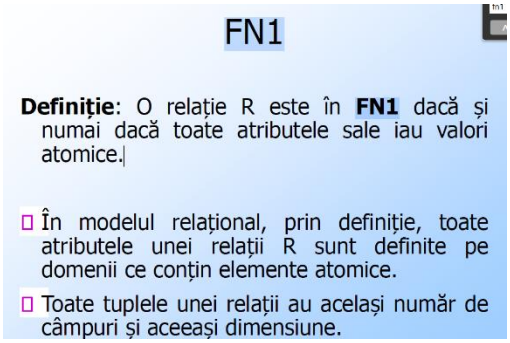
$con_Mea = DB::connect($med_Meu);

51. Care sunt caracteristicile unui SGBD total relațional?

- Principiul integrităţii domeniului
- Principiul integrităţii relaţiei
- Principiul integrităţii referinţei
- LMD cel puţin echivalent cu algebra relaţională

52. Dați un exemplu de normalizare pentru o relație inițial în FN1.

- Normalizarea - procesul prin care se “sparge” schema unei relatii în doua sau mai multe scheme.



Astfel, relatia Drinkers poate fi normalizata prin spargerea ei în Relația Drinkers și Adrese.

53. Rolul indecșilor în BD.

Indecşii sunt structuri de date care permit localizarea unor anumite înregistrări din baza de date într-un timp mult mai scurt, şi ca urmare folosirea indecşilor duce la micşorarea considerabilă a timpului de răspuns la interogări. Folosirea indecşilor este opţională, totuşi recomandată, deoarece ea duce la o îmbunătăţire vizibilă a performanţelor sistemului.

54. Definiți ce este un model de date.

Un model de date reprezintă o colecţie integrată de concepte necesare descrierii datelor, relaţiilor dintre ele, precum şi a constrângerilor asupra datelor (Connolly ş.a. 1998). Modelul de date este utilizat la descrierea schemei bazei de date, definind structura datelor, legăturile dintre acestea, semantica lor, precum şi constrângerile impuse, deşi nu este obligatoriu ca întotdeauna acestea să fie regăsite în orice model de date. Pe scurt, un model de date este utilizat pentru a reprezenta date despre date. Modelele de date oferă înţelegerea descriptivă necesară definirii schemelor logice şi externe şi sunt utile descrierii formale a schemei bazei de date. Schema logică a unei baze de date reprezintă o descriere abstractă a unei porţiuni din realitatea modelată împreună cu instanţele bazei de date.

Un model de date este alcătuit din trei elemente de bază: **entități, atribute și relații**

1. O reprezentare matematică a datelor.

   Exemple:

   - **modelul relaţional** = tabele;
   - **modelul semistructurat** = arbori/grafuri.

2. Operaţii cu datele.
3. Constrângeri.

55. Explicați funcționalitatea de import și export a datelor pentru un Sistem de Gestiune de Baze de Date.

Importul şi exportul datelor

Conversia datelor pentru prelucrarea cu alt SGBD sau cu aplicaţii terţe (de ex. Excel)

56. Nivelul intern de abstractizare al unui Sistem de Gestiune de Baze de Date.

Nivelul intern al SGBD reprezintă schema fizică a BD care specifică detalii suplimentare legate de stocarea datelor față de schema conceptuală (sau schema logică). Schema fizică menţionează modul în care tabelele (la modelul relaţional) descrise prin schema conceptuală sunt stocate pe dispozitive suport secundar, discuri sau benzi magnetice. De asemenea, descrie tipul fişierelor pentru stocare pe suport secundar (de ex. la MySQL: MyISAM sau InnoDB) şi crearea unor structuri auxiliare de date numite indecşi în scopul regăsirii mai rapide a datelor

57. Clauzele Group By, Having.

Într-o instrucţiune SELECT-FROM-WHERE se poate folosi clauza **GROUP BY** cu o listă de atribute (la sfârşit).

Tuplele din relaţia rezultat pentru blocul SELECT-FROM-WHERE **sunt grupate conform valorilor atributelor prezente în clauza GROUP BY** şi se poate aplica orice agregare pe fiecare grup în parte (şi numai pe grupurile formate).

```
SELECT beer, AVG(price)
FROM Sells
GROUP BY beer;
```

HAVING <condiţie> poate fi utilizată **după clauza GROUP BY.**

Dacă apare această clauză, condiţia se aplică fiecărui grup şi grupurile ce nu respectă condiţia sunt eliminate din rezultat.

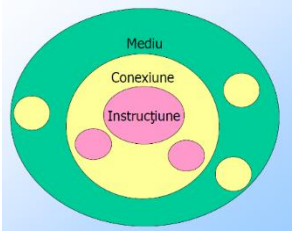
```
SELECT beer, AVG(price)
FROM Sells
GROUP BY beer
HAVING COUNT(bar) >= 3 OR
beer IN (SELECT name
FROM Beers
WHERE manf = 'Pete''s');
```

(Se foloseşte Sells(bar, beer, price) şi Beers(name, manf) pentru a găsi preţul mediu al acelor beri ce fie sunt servite în cel puţin trei baruri, fie sunt fabricate de "Pete's".)

58. La ce operații sunt testate constrângerile la nivel atribut/tuplă?

INSERT, UPDATE

59. Medii, conexiuni, interogări.



BD este, pentru majoritatea limbajelor de acces la baze de date, **un mediu** ("environment").

Serverele BD întreţin un număr de conexiuni, astfel încât serverele de aplicaţii pot lansa interogări sau pot efectua actualizări.

Serverul de aplicaţie emite de obicei instrucţiuni : **interogări şi actualizări**.

60. Sub ce formă poate fi folosită o interogare SELECT-FROM-WHERE în PSM?

În general interogările SELECT-FROMWHERE **NU** sunt permise în PSM.

Există trei moduri pentru a obţine efectul unei interogări:

1. **Interogările ce produc o singură valoare** pot fi utilizate ca **expresie într-o atribuire**.

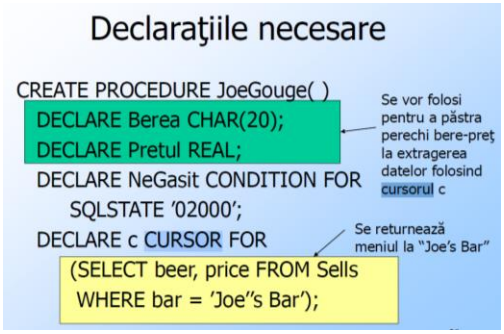
   Se foloseşte variabila locală p şi Sells(bar, beer, price), pentru a obţine preţul la care "Joe" vinde "Bud":

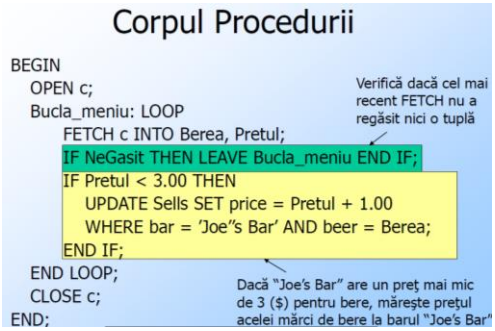
   SET p = (SELECT price FROM Sells WHERE bar = 'Joe''s Bar' AND beer = 'Bud');

2. **SELECT . . . INTO** având rezultat **1 tuplă.**

   SELECT price INTO p FROM Sells WHERE bar = 'Joe''s Bar' AND beer = 'Bud';

3. Cursoare





61. Definiți vederea SQL (view)

O vedere (view) este o relaţie definită în termeni de **tabele stocate** (numite tabele de bază) şi **alte vederi**.

Există două categorii de vederi:

1. Virtuale = datele afişate nu sunt stocate în baza de date; reprezintă doar o interogare pentru construirea relaţiei.

2. Materializate = sunt stocate

62. Transaction Manager, Lock Manager, Recovery Manager

Straturile “Transaction Manager ”, “Lock Manager ” şi “Recovery Manager” asigură **accesul concurent şi recuperarea datelor** în caz de incidente prin implementarea unor **protocoale de blocare (“lock”), prin planificarea cu atenţie a cererilor utilizatorilor şi prin păstrarea într-un jurnal a tuturor modificărilor asupra BD**

63. La ce este utilizat trigerul „INSTEAD OF”?

Triggerul **INSTEAD OF** este un triger definit pentru vederi și **permite actualizarea datelor din baza de date prin intermediul unei vederi**. De asemenea, trigger-ul permite interpretarea actualizărilor produse vederii astfel încât actualizările să aibă sens.

64. Interogare imbricată corelare între outer query și inner-query.

Să se găsească studenții care au nota 10

```
SELECT nume FROM student
WHERE nr_matr IN
    (SELECT nr_matr
     FROM Examen
    WHERE nota = 10);
```

65. Disk space manager.

Stratul “disk space manager” este cel mai de jos strat al unui SGBD şi **se ocupă de administrarea spaţiului pe suport extern, unde sunt stocate datele.**

Straturile superioare alocă, dealocă, citesc şi scriu pagini

66. Apelul procedurilor și al funcțiilor PSM.

Se foloseşte instrucţiunea EXECUTE (CALL) urmată de numele procedurii şi de argumente.

Exemplu: **EXECUTE JoeMenu(’Moosedrool’,5.00);**

67. Abordări pentru conectarea la BD a limbajelor convenționale.

Embedded SQL

Ideea de bază: **Un preprocesor translatează instrucţiunile SQL în apeluri de procedură ce se încadrează în codul limbajului gazdă**. Toate instrucţiunile "embedded SQL" încep cu **EXEC SQL**, în aşa fel încât preprocesorul să le descopere cu uşurinţă.

SQL Dinamic

SQL Dinamic se utilizează deoarece majoritatea aplicaţiilor folosesc interogări specifice şi instrucţiuni de actualizare ce interacţionează cu BD. SGBD-ul compilează instrucţiuni EXEC SQL … în apeluri procedură specifice şi produce un program obişnuit în limbaj gazdă ce foloseşte o bibliotecă. **Sqlplus** nu ştie ceea ce are nevoie să facă până la execuţie. Pregătirea ("prepare") unei interogări:

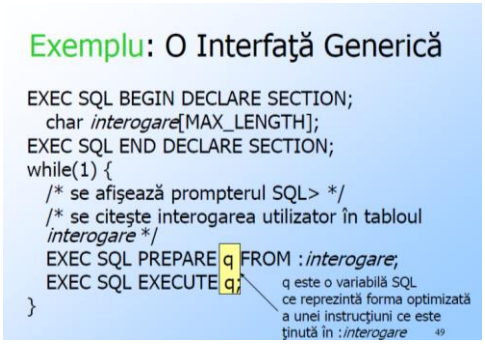
EXEC SQL PREPARE <nume-interogare>
FROM <textul interogării>;

Execuţia unei interogări:

EXEC SQL EXECUTE <nume-interogare>;

"Prepare" = optimizare interogare.

Se pregăteşte o dată, se execută de mai multe ori.



Limbaj Gazdă/ Interfeţe SQLVia Biblioteci

A treia abordare pentru conectarea la BD a limbajelor convenţionale este **apelul prin intermediul bibliotecilor. CLI,** alături de **JDBC, PHP/DB,** reprezintă **unelte de conectare** pentru a permite unui limbaj convențional să acceseze o bază de date.

În loc să se folosească un preprocesor (embedded SQL), se poate folosi o bibliotecă cu funcții. Biblioteca pentru C este numită SQL/**CLI („Call-Level Interface").** Preprocesorul pentru embedded SQL translatează instrucțiunile EXEC SQL … în **CLI** sau apeluri similare.

68. Descrieți pe scurt tipurile de constrângeri în contextul bazelor de date.

O constrângere este **o relaţie de legătură între elemente de date**, pe care SGBD-ul este obligat să le forţeze.

Tipuri de constrângeri:

- **Chei.**
  - Chei de tip **Atribut- Singular** și chei compuse din mai multe atribute (**Multi-Atribut**)
  - **Primary Key** sau **Unique Key** – Aceste chei evidențiază faptul că nu pot exista două tuple ale relaţiei care să corespundă întru-totul faţă de lista de atribute.
- **Cheie-străină,** sau integritate-referenţială - Valorile ce apar în atributele unei relaţii trebuie să apară împreună în anumite atribute din altă relaţie.
- **Constrângeri la nivel Valoare. -** constrâng valorile unui atribut particular. Se adaugă CHECK(<condiţie>) la declaraţia pentru un atribut. Condiţia poate folosi numele atributului, dar orice altă relaţie sau atribut ce apar trebuie folosite într-o interogare imbricată (subquery).
- **Constrângeri la nivel Tuplă** - Relaţie de legătură între componente.
- **Constrângeri Declarative ("Assertions"):** orice expresie logică SQL.

69. Tipuri de indecși

Index primar – index ce contine un set de campuri intre care este inclusa si cheia primara

Indexul secundar - cheia de cautare nu contine cheia primara (poate contine duplicate)

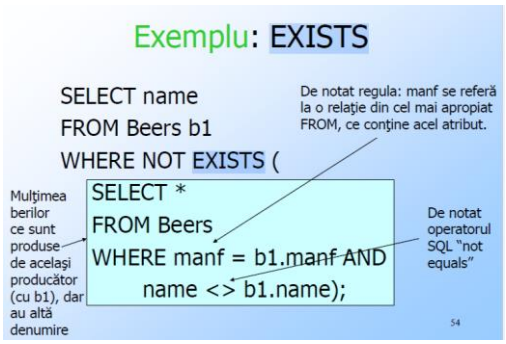
Index unic – cheia de cautare contine valori astfel incat nu exista tuple duplicate (cheia de cautare contine o cheie candidata)

O tabelă poate avea cel mult un index primar şi eventual mai mulţi indecşi secundari. Folosirea indecşilor sporeşte mult performanţele sistemului pentru operaţiile de căutare a datelor, totuşi trebuie avut grijă ca încărcarea sistemului să nu fie prea mare în cazul operaţiilor de scriere.

70. Operatorii EXISTS, ANY, ALL

EXISTS(<subquery>) are valoarea logică True dacă şi numai dacă rezultatul interogării imbricate este nevid.

Exemplu: **Se foloseşte Beers(name, manf) , pentru a găsi acele beri ce reprezintă unica bere a producătorului ei.**



Operatorul "ANY"

x = ANY(<subquery>) este o condiţie booleană ce **are valoarea logică True dacă x este *"egal"* cu cel puţin una** din tuplele rezultatului interogării imbricate.

"=" poate fi oricare din operatorii de comparaţie.

Exemplu: x >= ANY(<subquery>) semnifică **x nu este singura tuplă cea mai mică produsă de interogarea imbricată.**

De notat că tuplele trebuie să aibă doar o componentă.

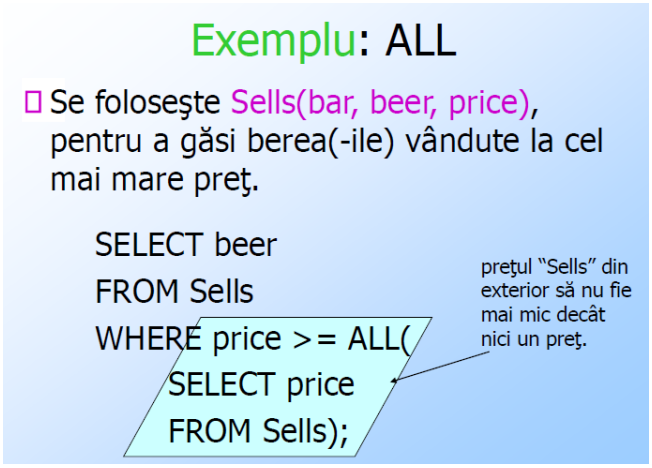
Operatorul "ALL"

x <> ALL(<subquery**>) are valoarea logică True dacă pentru fiecare tuplă t din relaţia "subquery", x este diferit de t.**

Cu alte cuvinte, x nu se regăseşte în rezultatul interogării imbricate.

"<>" poate fi orice operator de comparaţie.

Exemplu: x >= ALL(<subquery>) semnifică nu există tuplă mai mare ca x în rezultatul interogării imbricate.



71. Care este rolul funcției de gestiune a aplicațiilor ale unui SGBD?

Gestiunea aplicaţiilor

Limbaj de descriere a datelor

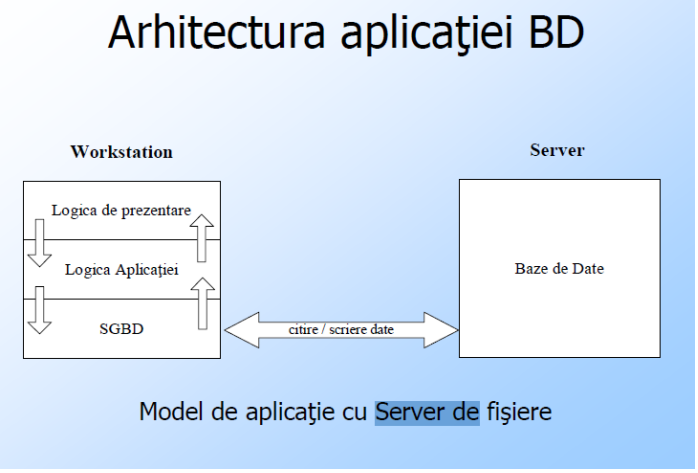
Limbaj de manipulare a datelor

Limbaj pentru afişarea datelor (pe ecran sau la imprimantă)

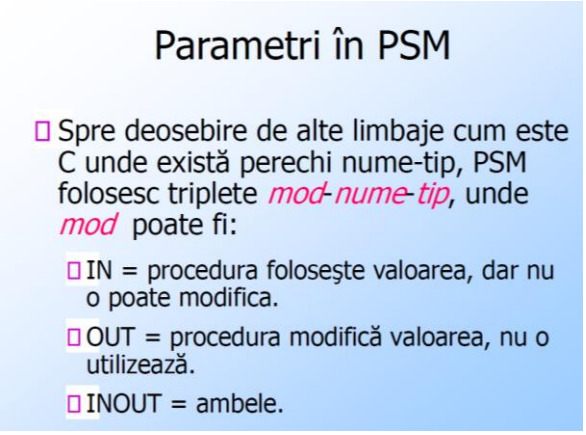
72. Buffer manager

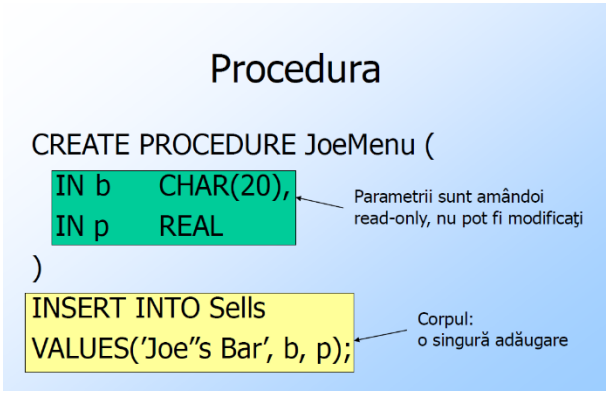
Stratul "buffer manager" este responsabil cu aducerea paginilor de pe disc în memoria internă

73. Modelul de aplicație cu server de fișiere



74. Transmiterea parametrilor în PSM





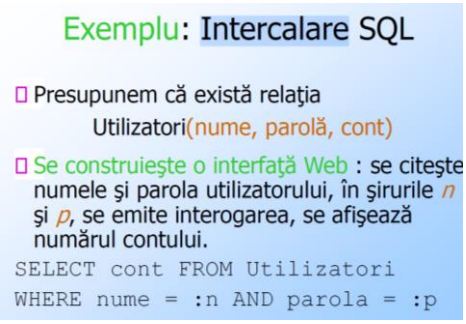
75. Intercalare SQL (SQL injection)

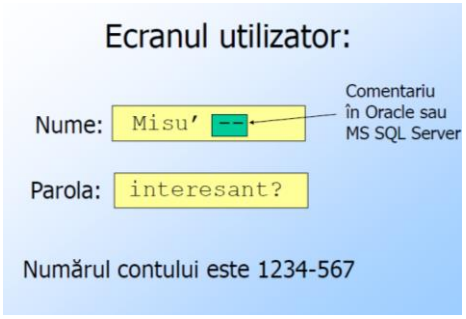
Intercalare de SQL

Interogările SQL sunt adeseori construite de către programe.

Aceste interogări primesc constante introduse de utilizator.

Atunci când codul nu este tratat cu atenţie,se pot întâmpla lucruri neaşteptate.





76. Definiți ce este aceea constrângere în contextul bazelor de date

O constrângere este **o relaţie de legătură între elemente de date**, pe care SGBD-ul este obligat să le forţeze.

77. Ce restricție există pentru a utiliza operatorul SQL, UNION?

Operatorul UNION este utilizat pentru a combina setul de rezultate a două sau mai multe instrucțiuni SELECT.

Fiecare instrucțiune SELECT din cadrul UNION trebuie să **aibă același număr de coloane.**

Coloanele trebuie să aibă, de asemenea, **tipuri de date similare.**

Coloanele din fiecare instrucțiune SELECT **trebuie să fie în aceeași ordine.**

78. Care este rolul funcțiilor următoare ale unui SGBD? Gestiunea dicționarului de date. Gestiunea fișierelor de date.

Gestiunea dicţionarului de date

Programele aplicative accesează datele prin intermediul SGBD, care caută în dicţionarul de date structura datelor şi a legăturilor necesare rezolvării problemei utilizatorului

Gestiunea fişierelor de date

Colecţia de date (BD) se materializează printr-un fişier sau colecţie de fişiere de date, fişiere index, etc.

79. Nivelul extern pentru un SGDB

Schema externă permite accesul la date să fie "croit" după nevoia grupurilor de utilizatori (de exemplu un angajat are acces la datele de salarizare proprii, iar un manager are acces la datele de salarizare ale tuturor angajaţilor din subordine)

O BD are **exact 1 schemă conceptuală şi 1 schemă fizică**

O BD **poate avea mai multe scheme externe**

80. Definiţi forma normală Boyce-Codd.

Definiție: Se spune că relaţia R este în FNBC dacă totdeauna când $X \rightarrow Y$ este o DF netrivială valabilă pentru R, X este o supercheie.

Ne reamintim: **netrivial** înseamnă **Y nu este conţinut în X.**

Ne reamintim, **o supercheie** este **orice superset al unei chei** (nu neapărat unul anume).

81. Explicaţi diferenţele între:

SELECT a

FROM R, S

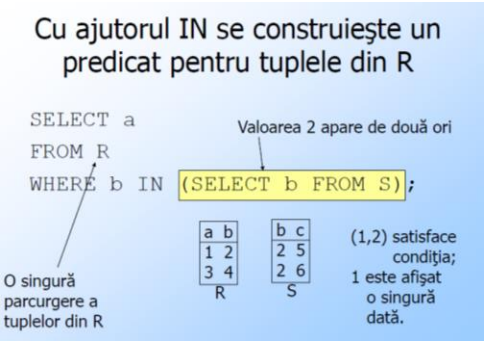
WHERE R.b = S.b;

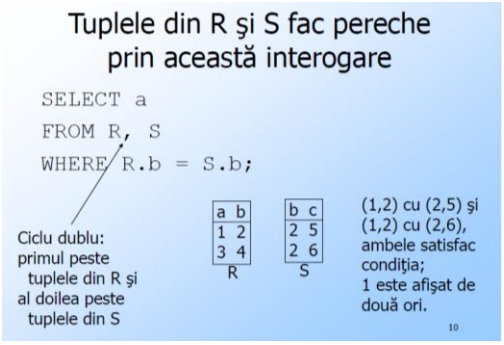
și

SELECT a

FROM R

WHERE b IN (SELECT b FROM S);





82 Explicaţi diferenţele între:

SELECT b

FROM R

WHERE a<10 or a>=10;

Si

SELECT b

FROM R;

Daca a are valoarea NULL atunci prima varianta nu va afisa acea tupla, iar a doua o va afisa