UTILIZAREA VEDERILOR

1. Obiectivul lucrării:

   Formarea și dezvoltarea abilitaților de interogare a bazelor de date.

2. Breviar teoretic cu exerciții și probleme rezolvate

O vedere este o descriere parțială sau particulară a datelor din unul sau mai multe tabele ale bazei de date. O vedere crează un tabel virtual al carui conţinut este definit printr-o interogare.
Vederile construiesc tabele logice dinamice cu date care provin din tabelele aflate în baza de date. Spre deosebire de tabel, vederea nu stochează date şi nici nu are alocat vreun spaţiu de stocare; vederea doar extrage sau derivă datele din tabelele la care aceasta se referă. Aceste tabele poartă denumirea de *tabele de bază* ale vederii. Tabele de bază ale vederii pot fi tabele fizice sau pot fi ele însele vederi.

O vedere este o definiţie a unei fraze SELECT memorată în baza de date. În principiu vederile se comportă şi pot fi folosite la fel ca orice tabel, dar în cazul operaţiilor de ştergere, inserare şi actualizare sunt diverse restricţii care trebuie să fie respectate.
Avantajul vederilor:

- Vederile simplifică şi specializează viziunea utilizatorilor asupra datelor. Ele oferă grupurilor de utilizatori un model al datelor mai apropiat de nevoile lor.
- Vederile oferă un mecanism suplimentar de securitate. Grupuri de utilizatori pot avea asupra vederilor drepturi specifice diferite de drepturile acestora la nivelul tabelelor de bază.
- Prin proiectarea adecvată a vederilor, interogările şi operaţiile de manipulare a datelor pot fi rezolvate mai simplu. De exemplu, este posibilă definirea unei vederi care realizează un join între o vedere care include clauza `GROUP BY` şi un alt tabel;
- Vederile menţin calculele complicate asupra datelor. Interogarea care defineşte vederea poate efectua calcule complicate aspra datelor din tabele; prin menţinerea acestei interogări ca o vedere, calculele sunt efectuate de fiecare dată când se face referire la vedere.

Crearea unei vederi
Sintaxa pentru crearea unei vederi este următoarea:

```
CREATE VIEW denumireVedere[(denumireColoana [,denumireColoana]...)]
      [WITH ENCRYPTION]
AS fraza_SELECT
      [WITH CHECK OPTION]
```

unde

- `denumireColoana` specifică denumirile expresiilor selectate de interogarea asociată vederii. Numărul coloanelor vederii trebuie să fie acelaşi cu numărul de expresii selectate de către interogarea vederii. Dacă lista denumirilor coloanelor vederii este omisă, SQL va folosi denumirile coloanelor atribuite prin fraza SELECT.

- `WITH ENCRYPTION` determină stocarea interogării în formă criptată şi prin urmare ea nu mai poate fi afişată
- `AS` indică interogarea asociată vederii. Aceasta poate fi orice frază `SELECT` care nu conţine clauzele `ORDER BY` şi `INTO`.
- `WITH CHECK OPTION` este o constrângere care arată că toate actualizările efectuate prin intermediul vederii vor afecta tabelele de bază numai dacă actualizările respective vor avea ca rezultat doar rânduri care pot fi vizualizate prin intermediul vederii.

Observaţie. Este uşor să confundăm o vedere cu un tabel. Odată definită o vedere, datele pot fi accesate prin intermediul ei ca şi cum ar fi un tabel. Din această cauză este bine să folosim convenţii de nume: putem prefixa tabelele cu "t" şi vederile cu "v".

Exemple:

```
Create view vSalariatiProductie (CodAngajat, Nume, Prenume, CNP)
AS SELECT CodAngajat, Nume,   Prenume,CNP
from tAngajati
where CodDepartament = 'Productie'
```

In loc să obligăm utilizatorii să formuleze interogări complicate, putem defini astfel de interogări sub formă de vedere, permiţându-le utilizatorilor să le acceseze:

```
Create view vFacturiClienti
AS SELECT A.NrFact,A.DataFact,A.CodClient,C.NumeClient as
Client,C.CodJudet,C.Localitate,
B.CodAngajat,B.Nume+' '+B.Prenume as Angajat,
D.CodProd,E.DenProd,E.UM,D.Cantitate,E.Pret,Cantitate*Pret as Valoare
from
   tFacturi as A inner join tAngajati as B on A.CodAngajat=B.CodAngajat
   inner join tClienti as C on A.CodClient=C.CodClient
   inner join tDetaliiFact as D on A.NrFact=D.NrFact
   inner join tProduse as E on D.CodProd=E.CodProd
```

Putem utiliza acum vederea `vFacturiClienti` ca şi cum ar fi un tabel:

```
select * from vFacturiClienti where MONTH(data)=3
```

Ca regulă, în CREATE VIEW nu este permisă folosirea clauzei ORDER BY. Există însă o posibilitate de a ocoli această restricţie prin utilizarea extensiei TOP 100 PERCENT. TOP n PERCENT permite utilizarea clauzei ORDER BY în interiorul vederii dându-ne astfel voie să solicităm întregul tabel logic pentru n=100.

```
Create view vSalariatiProductie2 (cod, nume, prenume, salariu)
AS SELECT top 100 percent
CodAngajat, nume, prenume,cnp
from tAngajati
where CodDepartament = 'Productie'
order by Nume,Prenume
```

Utilizarea vederilor în subinterogari

1)Sa se determine clienții pentru care suma valorică a facturilor (sum(cantitate*pret)) este mai mare decât valoarea medie facturată la nivel de client.

Vom crea o vedere care ne furnizează valoarea totală a facturilor la nivel de CodClient

```
Create view vValoriFacturateClienti
AS SELECT A.CodClient,sum(Cantitate*Pret) as Valoare
from
   tFacturi as A inner join tDetaliiFact as B on A.NrFact=B.NrFact
   inner join tProduse as C on B.CodProd=C.CodProd
group by A.CodClient
```

Acum selectăm clienții solicitați în enunț

```
select A.CodClient,B.NumeClient,B.CodJudet,B.Localitate,
       Valoare as [Valoare facturata]
from vValoriFacturateClienti as A
     inner join
     tClienti as B
     on A.CodClient=B.CodClient
where Valoare >(select AVG(Valoare) from vValoriFacturateClienti)
```

Operaţii DML asupra vederilor

În momentul în care în tabelele de bază sunt adăugate noi date sau sunt actualizate sau şterse cele existente, aceste modificări se reflectă corespunzător în vederile bazate pe aceste tabele. Acest lucru este adevărat şi reciproc, cu singura menţiune că există anumite restricţii la inserarea, actualizarea sau ştergerea datelor dintr-o vedere. Aceste restricţii sunt redate pe scurt în continuare:

- Nu pot fi inserate, şterse sau actualizate datele din vederi care conţin una dintre următoarele:
  - operatorul `DISTINCT` (pentru eliminarea duplicatelor);
  - cuvantul cheie TOP
  - clauzele `GROUP BY`
  - operatorul `UNION`
- Coloanele referite într-o frază UPDATE sau INSERT trebuie să aparţină unei singure tabele din cele referite în definiţia vederii, astfel încât să se poată determina fără ambiguităţi tabela de bază la care se referă operaţia.
- Nu pot fi inserate sau actualizate date care ar încălca constrângerile din tabelele de bază. De exemplu, dacă în tabela *`tAngajati`* coloana *`salariu`* este definită ca `NOT NULL`, atunci în orice vedere bazată pe acest tabel care nu conţine coloana *`salariu`* nu va fi posibilă inserarea de date deoarece aceasta ar duce la încălcarea constrângerii `NOT NULL`. Astfel următorul insert va furniza eroare

```
insert into vSalariatiProductie(CodAngajat,Nume,Prenume,CNP)
values ('100000','Popescu','Ioana','2750604123456')
```

`Următorul update` funcţionează corect.

```
update vSalariatiProductie
set nume='Radulescu' where cod=5
```

Fie vederea

```
create view vAngajatiProductie
as select * from tAngajati where CodDepartament='Productie'
```

Dacă executăm

```
update vAngajatiProductie
set CodDepartament='Administrativ'
where CodAngajat='5'
```

după modificare rândul corespunzător va dispărea din vedere (nu şi din tabel) pentru că nu mai aparţine departamentului `'Productie'`.

Dacă se adaugă opţiunea WITH CHECK la definirea vederii atunci astfel de operaţii nu mai sunt permise – produc eroare.

```
create view vAngajatiProductie
as select * from tAngajati where CodDepartament='Productie'
with check option
```

Modificarea vederilor

Modificarea unei vederi se poate face folosind comanda SQL `alter view`:

```
ALTER VIEW denumire_vedere
[(alias [,alias]...)]
 [WITH ENCRYPTION]
AS fraza_SELECT
 [WITH CHECK OPTION]
```

Ştergerea vederilor

Pentru ştergerea unei vederi se foloseşte comanda `drop view`:

```
drop view denumire_vedere;
```

Observaţie
Deşi vederile constituie un instrument extrem de convenabil pentru dezvoltatorul de aplicaţii, ele pot avea un impact negativ asupra performanţei. Dacă pentru vederi simple (de exemplu simple copii ale unui tabel) impactul asupra performantei nu este sesizabil, pentru vederi complexe, care cuprind interogări pe mai multe tabele, acest impact poate fi foarte sever. De aceea folosirea vederilor trebuie planificată cu grija, pentru a se vedea dacă avantajul creat de simplitatea în manipulare a acestora compensează impactul negativ asupra performanţei