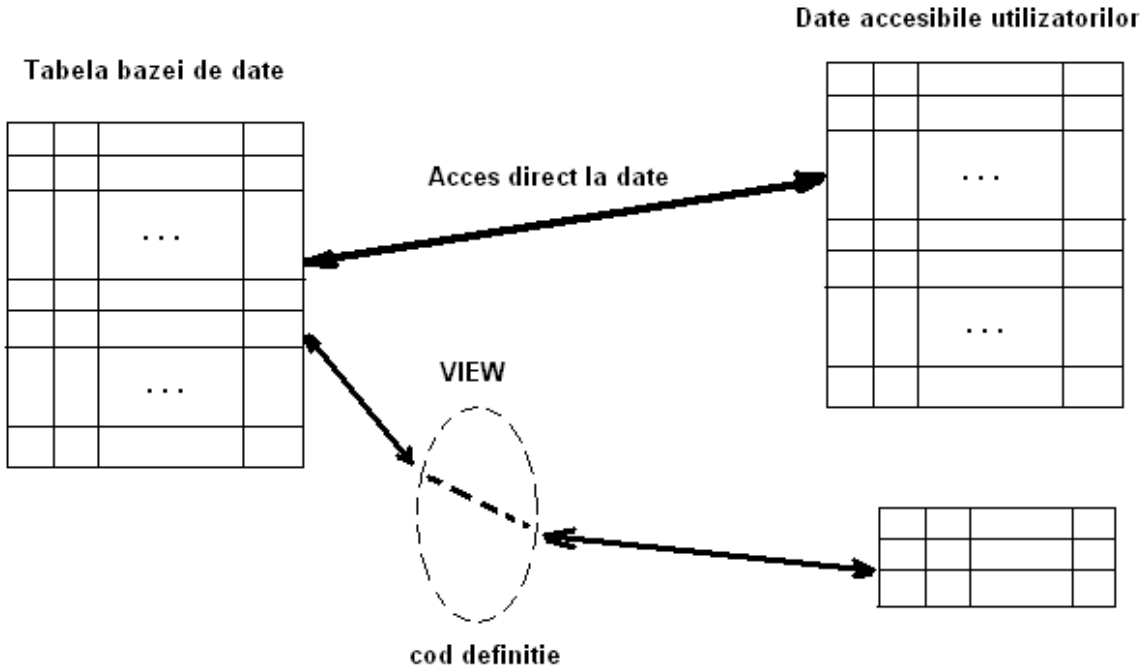
VIEW-VEDERI

- Ce este o vedere
- Care sunt avantajele folosirii vederilor
- Cum se creează o vedere
- Cum se șterge o vedere
- Cum se pot actualiza datele dintr-o tabelă prin intermediul unei vederi
- Când se pot modifica datele prin intermediul unei vederi.

- Uneori, din motive de securitate, este de dorit ca anumitor utilizatori să nu li se permită să aibă acces nelimitat la o tabelă, ci doar la datele ce se găsesc în anumite coloane ale acestei tabele.
- De exemplu, într-o firmă, contabila firmei nu va avea acces la coloanele ce se referă la proiectele în care sunt implicaţi la momentul actual fiecare angajat al firmei, însă va avea cu siguranţă acces la date privind salariul, tariful orar cu care este plătit fiecare angajat, numărul de ore lucrate etc. Pe de altă parte, bibliotecara de la biblioteca firmei, nu va avea acces la datele privind salarizarea personalului ci doar la datele personale ale angajaţilor (adresa, telefon, email etc).
- Pentru a putea da acces parţial la o tabelă, vom folosi ceea ce numim vederi (sau views).
- **O vedere este o tabelă virtuală, pentru care nu sunt memorate date propriu-zise ci doar definiţia vederii, care are rolul de filtrare a datelor.**

- Vederile sunt reprezentări logice ale tabelelor existente şi funcţionează ca nişte ferestre prin intermediul cărora pot fi vizualizate şi modificate datele din tabelele fizice



Acces direct şi indirect (printr-o vedere) la o tabelă

- Pe lângă faptul că oferă protecţie mărită a datelor, vederile mai au un mare avantaj: ele reduc în mod considerabil complexitatea interogărilor pe care utilizatorii trebuie să le scrie. O vedere poate fi construită folosind operaţii complexe de join, care rămân "ascunse" utlizatorului vederii respective, care va folosi interogări simple.
- La crearea unei vederi se va folosi o subinterogare, oricât de complexă, însă aceasta **NU** poate folosi clauza **ORDER BY**.

CREAREA ŞI ŞTERGEREA VEDERILOR

- Sintaxa generală de a comenzii pentru crearea unei vederi este:

CREATE OR REPLACE VIEW *nume_nedere*

AS *subinterogare*

- Opţiunea **OR REPLACE** poate lipsi, aceasta fiind utilă atunci când dorim să modificăm o vedere deja existentă.
- De exemplu, următoarea comandă creează o vedere simplă pe baza tabelei **employees**:

```
CREATE OR REPLACE VIEW v1 AS
( SELECT first_name||' '||last_name as Nume,
        salary
  FROM employees WHERE department_id=20)
```

- După cum am precizat, o vedere se poate construi folosind mai multe tabele, ca în exemplul următor:

```
CREATE OR REPLACE VIEW v2 AS
( SELECT a.nume ||' '|| a.prenume AS Angajat,
        b.nume ||' '|| b.prenume AS Sef,
            c.nume as Firma, d.nume as Job
  FROM angajat a, angajat b
  WHERE a.id_manager = b.id(+) and
       a.idFirm=c.idFirm(+) and a.idJob=d.idJob(+)
)
```

- **Observaţie.** În subinterogarea care defineşte o vedere, toate expresiile (nu şi coloanele simple) trebuie să aibă asociate un alias pentru a putea fi ulterior referite în interogări.

- Cum putem interoga aceste vederi? Ele pot fi folosite ca orice tabelă obişnuită, atât în interogări cât şi în operaţiile de actualizare (adăugare, modificare, ştergere), asupra acestora din urmă însă vom reveni în paragrafele următoare. Putem scrie de exemplu:

SELECT nume, salary FROM v1

WHERE nume like '%a%'

- sau

SELECT angajat, sef, firma, job

FROM v2

- O vedere poate fi şterasă cu comanda

  DROP VIEW nume_vedere

- **Atenţie!** Ştergerea unei vederi nu afectează în nici un fel datele din tabelele pe baza cărora s-a creat vederea. Toate modificările realizate asupra tabelelor prin intermediul vederii rămân valabile şi după ştergerea acesteia.

Actualizarea datelor prin intermediul vederilor

- În acest paragraf vom folosi pentru exemplificare tabelele **jucatori** şi **echipe** create cu ajutorul următoarelor comenzi:

```
CREATE TABLE jucatori(
  id NUMBER(5) PRIMARY KEY,
  nume VARCHAR2(30) NOT NULL,
  prenume VARCHAR2(30),
  rating NUMBER(1) CHECK (rating BETWEEN 1 AND
    5),
  varsta NUMBER(2),
  localitatea VARCHAR2(30) DEFAULT 'Timisoara',
  email VARCHAR2(30) UNIQUE
)
```

- Să creăm acum următoarele vederi:

```
CREATE OR REPLACE VIEW
    v1_JucatoriTm AS
( SELECT id, nume, varsta, localitatea
    FROM jucatori
  WHERE localitatea = 'Timisoara' )
```

- şi

```
CREATE OR REPLACE VIEW
    v2_Jucatori AS
( SELECT nume, prenume FROM
    jucatori
  WHERE rating IS NOT NULL)
```

- Aşadar am creat o vedere pentru toţi jucătorii din Timişoara. Putem interoga simplu această vedere:

```
SELECT * FROM v1_JucatoriTm
```

- rezultatul fiind cel din tabelul alăturat:

ID	NUME	VARSTA	LOCALITATEA
22	Vasilescu	-	Timisoara
103	Hunold	-	Timisoara
104	Ernst	-	Timisoara
107	Lorentz	-	Timisoara
37	Enescu	26	Timisoara
44	Plesca	37	Timisoara

- iar comanda

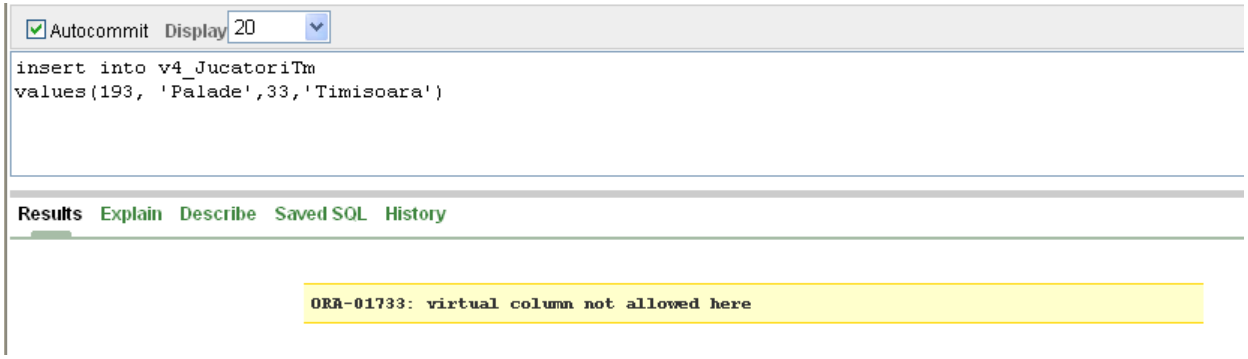
SELECT * FROM v2_Jucatori

- va afişa

NUME	PRENUME
Georgescu	Valeriu
Marin	Adriana
Ionescu	Emilian

- Vom încerca acum, pe rând, să vedem cum funcţionează fiecare operaţie de actualizare a datelor.
- O vedere poate fi creată folosind opţiunea **WITH READ OPTION**, prin intermediul unei astfel de vederi neputându-se efectua nici o operaţie de actualizare. Aceste vederi sunt folosite doar pentru vizualizarea datelor:

```
CREATE OR REPLACE VIEW v4_JucatoriTm AS
( SELECT id, nume, varsta, localitatea
  FROM jucatori
  WHERE localitatea = 'Timisoara' )
WITH READ ONLY
```



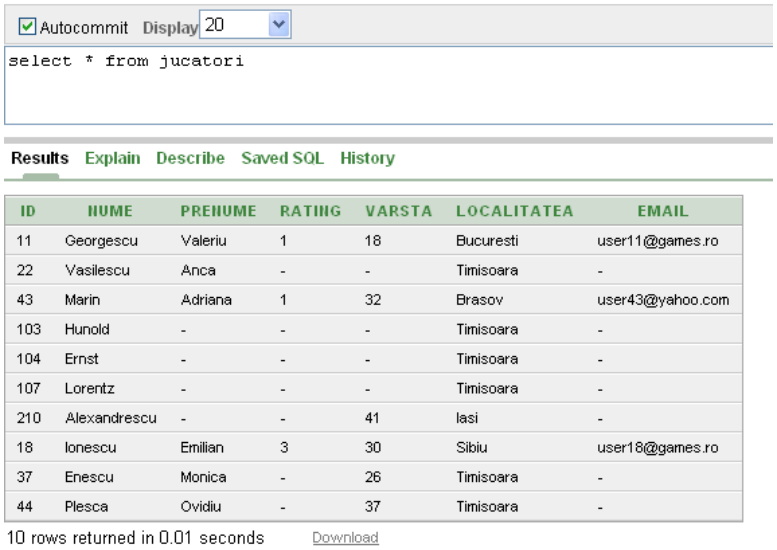
Inserarea datelor prin intermediul vederilor

- Încercăm să inserăm câte o înregistrare în tabela jucători prin intermediul celor două vederi create anterior:

```
insert into v1_JucatoriTm
values(210, 'Alexandrescu',41,'Iasi')
```

insert into v1_JucatoriTm values(210, 'Alexandrescu',41,'Iasi')

- Comanda funcţionează perfect, deşi jucătorul nou inserat nu respectă domeniul vederii **v1_JucatoriTm**, adică deşi putem vizualiza prin intermediul acestei vederi doar jucătorii din Timişoara, am reuşit totuşi să inserăm un jucător din altă localitate.
- Acest lucru ar putea crea probleme de securitate (am creat vederea tocmai pentru a restricţiona drepturile utilizatorilor).



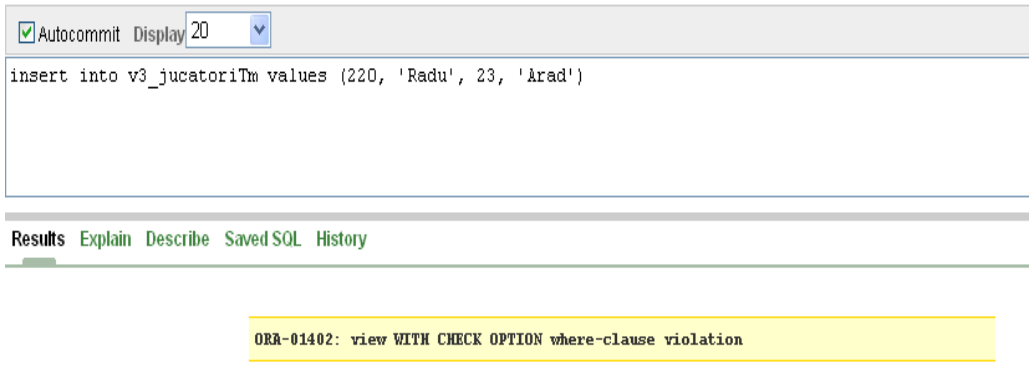
ID	NUME	PRENUME	RATING	VARSTA	LOCALITATEA	EMAIL
11	Georgescu	Valeriu	1	18	Bucuresti	user11@games.ro
22	Vasilescu	Anca	-	-	Timisoara	-
43	Marin	Adriana	1	32	Brasov	user43@yahoo.com
103	Hunold	-	-	-	Timisoara	-
104	Ernst	-	-	-	Timisoara	-
107	Lorentz	-	-	-	Timisoara	-
210	Alexandrescu	-	-	41	Iasi	-
18	Ionescu	Emilian	3	30	Sibiu	user18@games.ro
37	Enescu	Monica	-	26	Timisoara	-
44	Plesca	Ovidiu	-	37	Timisoara	-

10 rows returned in 0.01 seconds Download

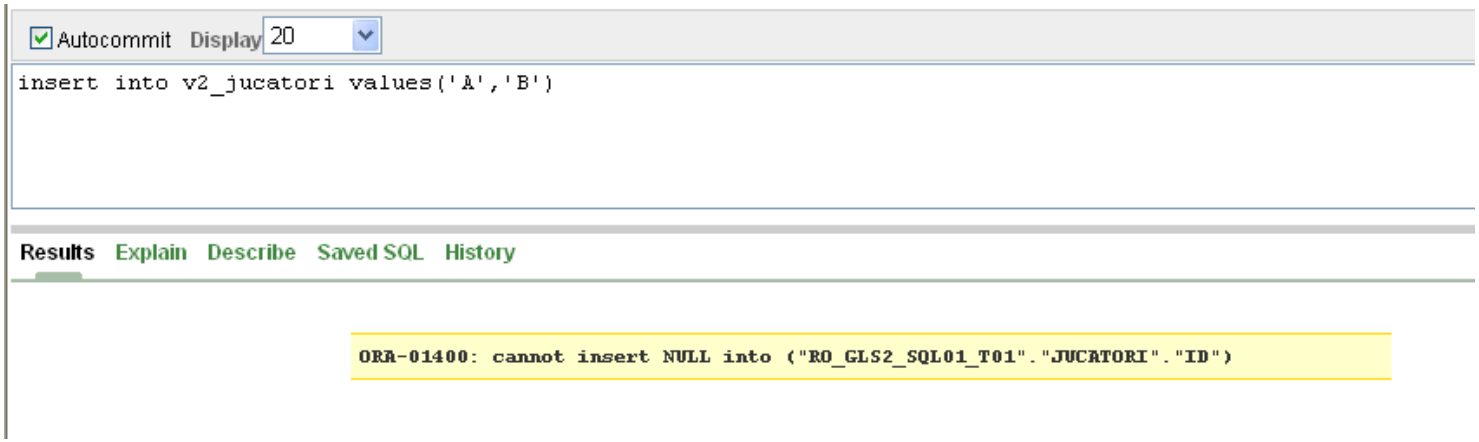
- Această problemă poate fi rezolvată prin folosirea opţiunii **WITH CHECK OPTION** la crearea vederii. Vom crea o nouă vedere **v3_jucatoriTm** folosind această opţiune:

```
CREATE OR REPLACE VIEW v3_JucatoriTm AS
( SELECT id, nume, varsta, localitatea FROM jucatori
  WHERE localitatea = 'Timisoara' )
WITH CHECK OPTION
```

- De această dată nu mai putem insera valori care sunt în afara domeniului vederii



- Prin intermediul vederii **v2_jucatori** nu vom putea insera linii în tabela **jucatori**, deoarece prin intermediul vederii nu avem acces la câmpul **id**, care fiind cheie primară nu poate fi iniţializată cu valoarea implicită **NULL**



Ştergerea datelor prin intermediul vederilor

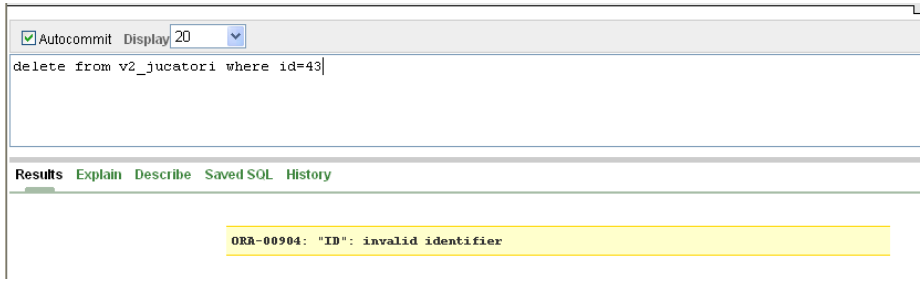
- La ştergerea unei înregistrări vom folosi comanda **DELETE** cu formatul deja cunoscut. Evident nu vom putea şterge din tabela decât liniile accesibile prin vederea respectivă. De aceea comanda:

DELETE FROM v1_jucatoriTm WHERE id=43

- nu va genera nici o eroare, însă nu va şterge nici o linie întrucât jucătorul având **id**-ul **43** este din **Brasov**, deci nu avem acces la el prin intermediul vederii **v1_jucatoriTm**.
- Similar, nu vom putea folosi în clauza **WHERE** a comenzii **DELETE** coloane care nu sunt vizibile din vederea respectivă. De exemplu comanda

DELETE FROM v2_jucatori WHERE id=43

- va genera o eroare, deoarece câmpul **id** este inaccesibil vederii



- Comenzile

delete from v2_jucatori where prenume='Emilian'

- şi

delete from jucatori where id=107

- sunt perfect funcţionale.

Modificarea datelor prin intermediul vederilor

- Ca şi în cazul celorlaltor operaţii de actualizare vom putea modifica doar valorile liniilor şi coloanelor care sunt vizibile din vederea respectivă:

update v1_jucatoriTm

set varsta=13

where id=103

Restricţii privind utilizarea vederilor

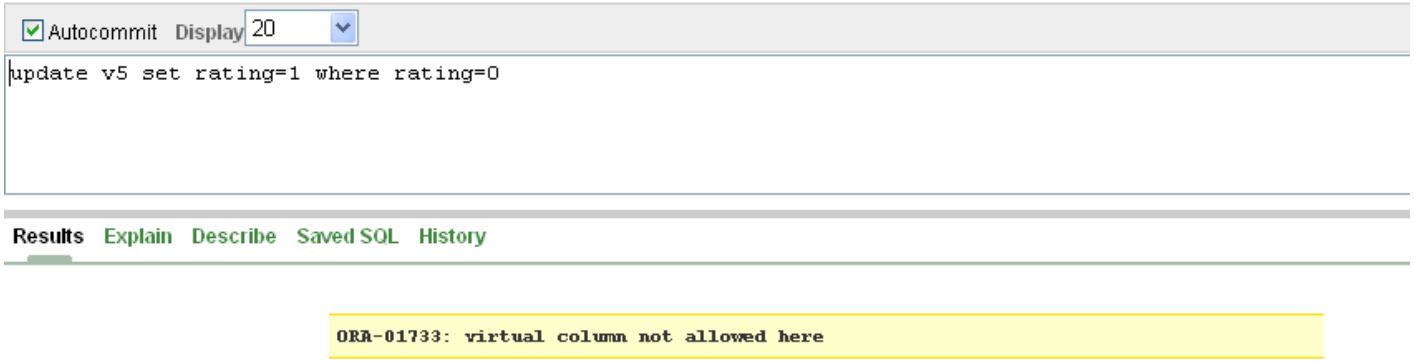
- Operaţiile de actualizare a datelor prin intermediul vederilor NU pot fi realizate în următoarele condiţii:
- actualizarea datelor (**ştergere**, **modificare**, **inserare**) nu se poate efectua dacă subinterogarea cu care s-a creat vederea foloseşte:
  - funcţii de grup
  - clauza **GROUP BY**
  - clauza **DISTINCT**
  - pseudocoloanele **ROWNUM** sau **ROWID**
- nu se poate **modifica** un câmp calculat al unei vederi:
- De exemplu, dacă s-a creat vederea

```
CREATE VIEW v5 AS
( SELECT id, nume, nvl(rating,0) rating
  FROM jucatori)
```

- vom putea actualiza câmpurile id şi nume:

```
UPDATE v5
SET nume='Eminescu'
WHERE id=37
```

- dar nu putem modifica valoarea din câmpul **rating** (fig. II.8.7.)



- Nu se poate insera o linie într-o tabelă prin intermediul unei vederi decât dacă toate coloanele **NOT NULL** ale tabelei sunt prezente în vedere.

In această lecție am învățat:

- Ce este o vedere
- Care sunt avantajele folosirii vederilor
- Cum se creează o vedere
- Cum se șterge o vedere
- Cum se pot actualiza datele dintr-o tabelă prin intermediul unei vederi
- Când se pot modifica datele prin intermediul unei vederi.