REZOLVĂRI SUBIECTE
ATESTAT 2009 – BAZE DE DATE
IINFORMATICĂ INTENSIV

SUBIECTUL 1

Să se creeze un tabel cu structura: **Nume_muncitor, Cod_atelier, Numar_piese_produse** şi să se introducă 8 înregistrări (o înregistrare - un muncitor, **codul unui atelier este un număr de la 1 la 9**. Pot fi mai mulţi muncitori într-un atelier şi toţi au nume diferite).

Cerinţe:

a) Să se afişeze codul atelierului/atelierelor din tabel cu cei mai mulţi muncitori;

b) Să se afişeze codurile atelierelor în care numărul de piese produse este mai mare decât o valoare dată de la tastatură.

REZOLVARE:

Pentru crearea tabelului se scrie comanda:

```
CREATE TABLE Muncitori (Nume_muncitor VARCHAR2(20) PRIMARY KEY, Cod_atelier
NUMBER(1) CHECK (Cod_atelier>=1 AND Cod_atelier<=9), Numar_piese_produse
NUMBER(3))
```

Regula (condiţia) de validare: Codul atelierului poate fi doar un număr de la 1 la 9 (constrângere de tip CHECK).

Dacă se încearcă introducerea unei alte valori decât cele care respectă regula de validare, se va afişa un mesaj de eroare:

```
check constraint violated
```

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Muncitori VALUES ('gigi', 1, 100)
INSERT INTO Muncitori VALUES ('ion', 1, 200)
INSERT INTO Muncitori VALUES ('mihai', 2, 200)
INSERT INTO Muncitori VALUES ('vasile', 2, 250)
INSERT INTO Muncitori VALUES ('maria', 2, 150)
INSERT INTO Muncitori VALUES ('ana', 2, 100)
INSERT INTO Muncitori VALUES ('alina', 3, 100)
INSERT INTO Muncitori VALUES ('andrei', 3, 250)
INSERT INTO Muncitori VALUES ('ionela', 3, 200)
```

Pentru a afişa conţinutul tabelului, se foloseşte comanda

```
SELECT * FROM Muncitori
```

NUME_MUNCITOR	COD_ATELIER	NUMAR_PIESE_PRODUSE
ion	1	200
mihai	2	200
maria	2	150
ionela	3	200
gigi	1	100
vasile	2	250
ana	2	100
alina	3	100
andrei	3	250

9 rows returned in 0.01 seconds Download

a) Pentru rezolvarea acestei cerinţe se scrie comanda:

```
SELECT Cod_atelier, COUNT(*) FROM Muncitori GROUP BY Cod_atelier
HAVING COUNT(*)=
(SELECT MAX(COUNT(*)) FROM Muncitori GROUP BY Cod_atelier)
```

COD_ATELIER	COUNT(*)
2	4

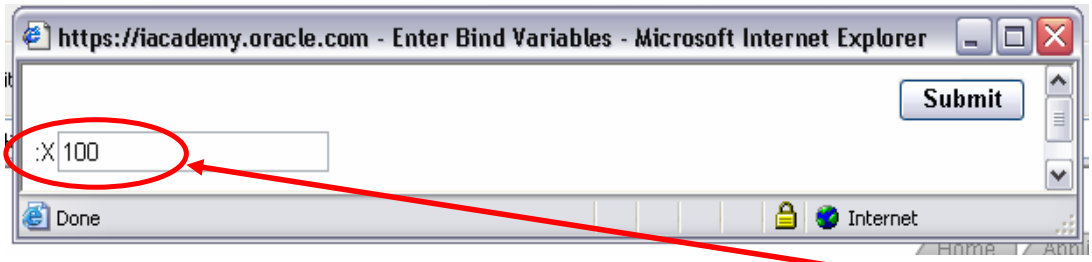
1 rows returned in 0.00 seconds

b) Pentru citirea unei valori pentru o variabilă de la tastatură se foloseşte: **:nume_variabilă** (de exemplu: pentru citirea variabilei x se scrie **:x**).

Pentru rezolvarea cerinței se foloseşte comanda:

SELECT Cod_atelier, Numar_piese_produse FROM Muncitori WHERE Numar_piese_produse >:x

Se va deschide o casetă de dialog unde se aşteaptă introducerea unei valori pentru variabila x (se va ține cont de tipul datelor care se compară cu **x**, în cazul nostru **Numar_piese_produse** care este un număr de maxim 3 cifre).



În zona de editare (în dreapta lui X) se completează valoarea dorită (de ex. 100) şi apoi clic pe **Submit.**

COD_ATELIER	NUMAR_PIESE_PRODUSE
1	200
2	200
2	150
3	200
2	250
3	250

6 rows returned in 0.00 seconds Download

SUBIECTUL 2

Să se creeze un tabel cu structura: **Nume_muncitor, Cod_atelier, Numar_piese_produse** şi să se introducă 8 înregistrări (o înregistrare - un muncitor, **codul unui atelier este un număr de la 1 la 9**. Pot fi mai mulți muncitori într-un atelier şi toți au nume diferite).

Cerințe:

a) Se şterg muncitorii pentru care producția este mai mică decât o valoare dată de la tastatură. Să se afişeze numele acestora şi codurile atelierelor din care făceau parte;

b) Să se afişeze numărul atelierelor ce au rămas cu cel mult doi muncitori în urma ştergerii articolelor de la cerința **a)**.

REZOLVARE:

Pentru crearea tabelului şi „popularea” cu date, vezi subiectul 1.

a) Pentru afişarea muncitorilor ce vor fi şterşi, se scrie comanda:

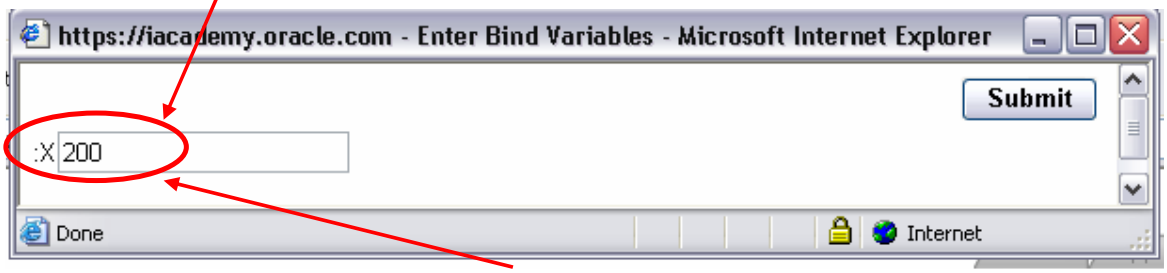
SELECT Nume_muncitor, Cod_atelier FROM Muncitori
WHERE Numar_piese_produse < :x

NUME_MUNCITOR	COD_ATELIER
maria	2
gigi	1
ana	2
alina	3

4 rows returned in 0.00 seconds Download

Pentru ştergerea muncitorilor se foloseşte comanda:

DELETE FROM Muncitori WHERE Numar_piese_produse < :x



Pentru valoarea lui x introdusă în casetă (de ex. 200) se şterg 4 linii, cele afişate mai sus.
Pentru afişarea conţinutului tabelului după ştergere, se foloseşte comanda:

SELECT * FROM Muncitori

NUME_MUNCITOR	COD_ATELIER	NUMAR_PIESE_PRODUSE
ion	1	200
mihai	2	200
ionela	3	200
vasile	2	250
andrei	3	250

5 rows returned in 0.01 seconds Download

b) Pentru afişarea numărului atelierelor ce au cel mult 2 muncitori, se poate folosi comanda:

SELECT COUNT(DISTINCT Cod_atelier) FROM Muncitori a WHERE a.Cod_atelier IN
(SELECT b.Cod_atelier FROM Muncitori b WHERE (SELECT count(Nume_muncitor)
FROM Muncitori c WHERE c.Cod_atelier=b.Cod_atelier) <=2)

COUNT(DISTINCTCOD_ATELIER)
3

1 rows returned in 0.01 seconds Download

SELECT count(Nume_muncitor) FROM Muncitori c WHERE c.Cod_atelier=b.Cod_atelier – numără muncitorii din tabelul **c** din atelierul cu codul egal cu cel curent din tabelul **b.**

SELECT b.Cod_atelier FROM Muncitori b WHERE (SELECT count(Nume_muncitor) FROM Muncitori c WHERE c.Cod_atelier=b.Cod_atelier) <=2 – selectează din tabelul **b** atelierele care au cel mult 2 muncitori.

SUBIECTUL 3

Să se creeze un tabel cu structura: **Nume_medic, Nume_pacient, Diagnostic, Data_consultului** şi să se introducă cel puţin 8 înregistrări.

Cerinţe:

a) Să se afişeze numărul pacienţilor consultaţi în luna curentă de către un medic al cărui nume se introduce de la tastatură;

b) Să se afişeze diagnosticul cel mai frecvent pus în luna curentă.

REZOLVARE:

Trebuie adăugat un câmp (coloană) tabelului care să poată fi cheie primară (să identifice unic o linie a tabelului) cu numele **Cod_consultatie**.

Pentru crearea tabelului se scrie comanda:

CREATE TABLE Consultatii (Cod_consultatie NUMBER(3) PRIMARY KEY, Nume_medic VARCHAR2(20), Nume_pacient VARCHAR2(20), Diagnostic VARCHAR2(20), Data_consultului DATE)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Consultatii VALUES (100,'Gigi','Ion','Gripa','02.feb.2009')
INSERT INTO Consultatii VALUES (101,'Gigi','Maria','Pneumonie','22.feb.2009')
INSERT INTO Consultatii VALUES (102,'Gigi','Vasile','Viroza','25.feb.2009')
INSERT INTO Consultatii VALUES (103,'Marius','Valentin','Viroza','11.jan.2009')
INSERT INTO Consultatii VALUES (104,'Marius','Liviu','Gripa','21.jan.2009')
INSERT INTO Consultatii VALUES (105,'Marius','George','Viroza','29.jan.2009')
INSERT INTO Consultatii VALUES (106,'Ana','Liliana','Hepatita','28.jan.2009')
INSERT INTO Consultatii VALUES (107,'Ana','Vlad','Horia','30.jan.2009')
```

Pentru a afişa conţinutul tabelului, se foloseşte comanda

SELECT * FROM Consultatii

COD_CONSULTATIE	NUME_MEDIC	NUME_PACIENT	DIAGNOSTIC	DATA_CONSULTULUI
101	Gigi	Maria	Pneumonie	22-FEB-09
103	Marius	Valentin	Viroza	11-JAN-09
107	Ana	Vlad	Horia	30-JAN-09
100	Gigi	ion	Gripa	02-FEB-09
102	Gigi	Vasile	Viroza	25-FEB-09
104	Marius	Liviu	Gripa	21-JAN-09
105	Marius	George	Viroza	29-JAN-09
106	Ana	Liliana	Hepatita	28-JAN-09

a) Selectarea consultaţiilor (**Cod_consultatie**) din luna curentă se poate face folosind funcţiile de prelucrare a datelor calendaristice:

- **CURRENT_DATE** – data şi ora curentă a aplicaţiei client (care, în cazul utilizării https://iacademy.oracle.com, este data serverului pe care se găseşte);
- **MONTHS_BETWEEN(data_1, data_2)** – cu semnificaţia: numărul de luni dintre cele două date calendaristice (dacă **data_1 < data_2**, atunci rezultatul este un număr negativ;
- **LAST_DAY(data_1)** – ultima zi din luna corespunzătoare datei **data_1**;

SELECT cod_consultatie, data_consultului FROM Consultatii WHERE MONTHS_BETWEEN(CURRENT_DATE, LAST_DAY(data_consultului))<=0

COD_CONSULTATIE	DATA_CONSULTULUI
101	22-FEB-09
100	02-FEB-09
102	25-FEB-09

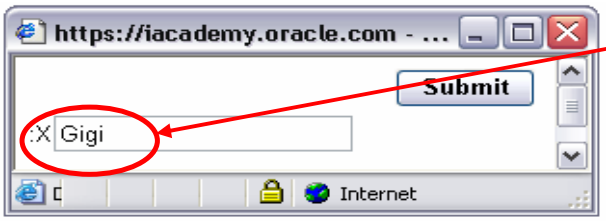
3 rows returned in 0.01 seconds Download

Ex: luna curentă „februarie" (**feb**), considerând data curentă:

CURRENT_DATE
24-FEB-09

Selectarea consultațiilor (**Cod_consultatie**) din luna curentă pentru un medic cu numele citit de la tastatură se poate face cu comanda:

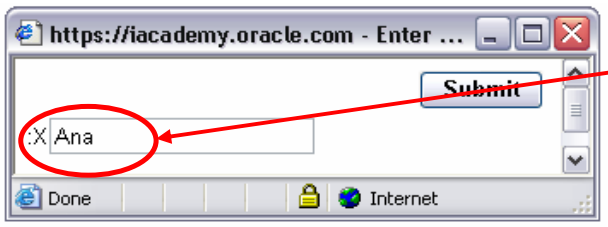
SELECT cod_consultatie, data_consultului FROM Consultatii WHERE (MONTHS_BETWEEN(CURRENT_DATE, LAST_DAY(data_consultului))<=0) AND (Nume_medic= :x)



Pentru medicul **'Gigi'** (pentru variabila **x** s-a introdus valoarea **'Gigi'**) se obține:

COD_CONSULTATIE	DATA_CONSULTULUI
101	22-FEB-09
100	02-FEB-09
102	25-FEB-09

3 rows returned in 0.01 seconds Download



Pentru medicul **'Ana'** (pentru variabila **x** s-a introdus valoarea **'Ana'**) se obține:

Results Explain
no data found

(*nu există linii în tabel care să îndeplinească aceste condiții*)

Pentru a rezolva cerința (numărul pacienților consultați în luna curentă de către un medic al cărui nume se introduce de la tastatură), se poate folosi următoarea comandă:

SELECT COUNT(cod_consultatie) FROM Consultatii WHERE (MONTHS_BETWEEN(CURRENT_DATE, LAST_DAY(data_consultului))<=0) AND (Nume_medic= :x)

Pentru medicul **Gigi** (pentru variabila **x** s-a introdus valoarea Gigi) se obține:

COUNT(COD_CONSULTATIE)
3

1 rows returned in 0.01 seconds

Pentru orice altă valoare introdusă de la tastatură, considerând tabelul cu datele de mai sus, se va obține:

Results Explain
no data found

b) Diagnosticul cel mai frecvent pus în luna curentă, presupune:

- selectarea consultațiilor din luna curentă;
- numărarea consultațiilor pe fiecare diagnostic din luna curentă;
- alegerea diagnosticului cu număr maxim de apariții (număr maxim de linii selectate pentru acel diagnostic).

SELECT DISTINCT diagnostic FROM Consultatii WHERE (SELECT MAX(COUNT(Cod_consultatie)) FROM Consultatii WHERE MONTHS_BETWEEN(CURRENT_DATE, LAST_DAY(data_consultului))<=0 GROUP BY Diagnostic) IN (SELECT COUNT(Cod_consultatie) FROM Consultatii WHERE MONTHS_BETWEEN(CURRENT_DATE, LAST_DAY(data_consultului))<=0 GROUP BY Diagnostic)

DIAGNOSTIC
pneumonie
gripa
viroza

3 rows returned

SUBIECTUL 4

Să se creeze un tabel **PACIENTI** având următoarea structură: **Nume, An_Nastere, Inaltime, Greutate, Temperatura** cu condițiile (restricții):

- anul naşterii să fie între anii 1900 si 1999
- înălțimea între 1,30m si 1,99m
- greutatea între 39Kg si 120Kg
- temperatura să fie între 36^0 si 40^0

Inserați în acest tabel cel puțin 5 înregistrări. Să se listeze persoanele cu vârsta mai mare decât o valoare dată (de la tastatură).

REZOLVARE:

Trebuie adăugat un câmp (coloană) tabelului care să poată fi cheie primară (să identifice unic o linie a tabelului) cu numele **Cod_pacient**.

Pentru crearea tabelului se scrie comanda:

CREATE TABLE Pacienti (Cod_pacient NUMBER(3) PRIMARY KEY, Nume VARCHAR2(20), An_nastere NUMBER(4) CHECK (An_nastere>=1900 AND An_nastere<=1999), Inaltime NUMBER (4,2) CHECK (Inaltime>=1.30 AND Inaltime<=1.99), Greutate NUMBER(3) CHECK (Greutate>=39 AND Greutate<=120), Temperatura NUMBER(5,2) CHECK (Temperatura>=36 AND Temperatura<=40))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Pacienti VALUES (10,'gigi', 1980, 1.84, 84, 36.7)
INSERT INTO Pacienti VALUES (11,'ion', 1970, 1.75, 77, 36.6)
INSERT INTO Pacienti VALUES (12,'mara', 1976, 1.70, 62, 36.5)
INSERT INTO Pacienti VALUES (13,'lucia', 1986, 1.68, 58, 36.5)
INSERT INTO Pacienti VALUES (14,'laurentiu', 1977, 1.78, 82, 36.6)

Pentru a afişa conținutul tabelului, se foloseşte comanda:
SELECT * FROM Pacienti

COD_PACIENT	NUME	INALTIME	GREUTATE	TEMPERATURA	AN_NASTERE
10	gigi	1.84	84	36.7	1980
13	lucia	1.68	58	36.5	1986
14	laurentiu	1.78	82	36.6	1977
11	ion	1.75	77	36.6	1970
12	mara	1.7	62	36.5	1976

5 rows returned in 0.01 seconds Download

a) Pentru rezolvarea cerinței, se folosesc următoarele funcții:

- **TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY')** – transformă data curentă în şir de caractere cu formatul: **'DD.MON.YYYY'**;
- **SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4)** – extrage anul din şirul de caractere corespunzător datei curente;
- **TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE,'DD.MON.YYYY'),8,4))** – transformă şirul de caractere corespunzător anului din data curentă în număr;

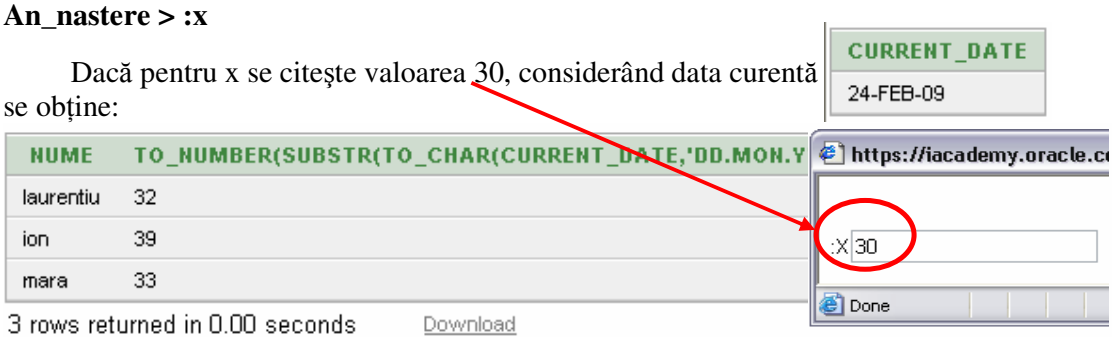
Pentru calculul vârstei unei persoane se foloseşte expresia:

TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4)) - An_nastere

Comanda următoare permite afişarea numelui şi a vârstei pentru persoanele care au vârsta mai mare decât o valoare citită în variabila x:

SELECT Nume, TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE , 'DD.MON.YYYY'),8,4)) - An_nastere FROM Pacienti WHERE TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4)) - An_nastere > :x

Dacă pentru x se citeşte valoarea 30, considerând data curentă se obţine:



SUBIECTUL 5

Să se construiască un tabel cu principalele producţii cinematografice din secolul XX. Acesta are următoarea structură: **Nume_film**, **Gen**, **Tara**, **An**, **Regizor**

a) Care sunt filmele produse în România în anul curent?
b) Câte filme de comedie au fost produse în România înainte de 1989?
c) Ştergeţi filmele regizate de regizorul filmului .... (introdus de la tastatură).
d) Afişaţi pentru fiecare regizor numărul filmelor regizate.

REZOLVARE:

Trebuie adăugat un câmp (coloană) tabelului care să poată fi cheie primară (să identifice unic o linie a tabelului) cu numele **Cod_film**.

Pentru crearea tabelului se scrie comanda:

CREATE TABLE Filme (Cod_film NUMBER(3) PRIMARY KEY, Nume_film VARCHAR2(30), Gen VARCHAR2(20), Tara VARCHAR2(20), An NUMBER(4), Regizor VARCHAR2(20))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Filme VALUES (100,'Razboiul stelelor', 'SF','SUA' , 1980,'George Lucas')
INSERT INTO Filme VALUES (101,'Harry Potter', 'Aventuri','SUA', 2001, 'Chris Columbus')
INSERT INTO Filme VALUES (102,'Dacii', 'Istoric','Romania', 1967,'Sergiu Nicolaescu')
INSERT INTO Filme VALUES (103,'Balanta', 'Drama','Romania', 1985,'Lucian Pintilie')
INSERT INTO Filme VALUES (104,'Nea Marin miliardar', 'Comedie', 'Romania', 1983, 'Sergiu Nicolaescu')
INSERT INTO Filme VALUES (105,'BD la munte si la mare', 'Comedie', 'Romania', 1972, 'Geo Saizescu')
INSERT INTO Filme VALUES (106,'Jurassyc Parck II', 'SF', 'SUA', 1996, 'Spilberg')
INSERT INTO Filme VALUES (107,'Restul e tacere', 'Drama','Romania', 2009, 'Nae Caranfil')
INSERT INTO Filme VALUES (108,'Boogie', 'Comedie','Romania', 2009, 'Radu Muntean')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a câmpului **Cod_film**, se foloseşte comanda:

SELECT * FROM Filme ORDER BY Cod_film

COD_FILM	NUME_FILM	GEN	TARA	AN	REGIZOR
100	Razboiul stelelor	SF	SUA	1980	George Lucas
101	Harry Potter	Aventuri	SUA	2001	Chris Columbus
102	Dacii	Istoric	Romania	1967	Sergiu Nicolaescu
103	Balanta	Drama	Romania	1985	Lucian Pintilie
104	Nea Marin miliardar	Comedie	Romania	1983	Sergiu Nicolaescu
105	BD la munte si la mare	Comedie	Romania	1972	Geo Saizescu
106	Jurassyc Parck II	SF	SUA	1996	Spilberg
107	Restul e tacere	Drama	Romania	2009	Nae Caranfil
108	Boogie	Comedie	Romania	2009	Radu Muntean

9 rows returned in 0.01 seconds Download

a) Filmele produse în România în anul curent.

Pentru rezolvarea cerinţei, se pot folosi următoarele funcţii:

- **TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY')** – transformă data curentă în şir de caractere cu formatul: **'DD.MON.YYYY'**;
- **SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4)** – extrage anul din şirul de caractere corespunzător datei curente;

SELECT Nume_film FROM Filme WHERE An =
TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4))

Se obţine lista filmelor produse în România în anul curent:

NUME_FILM
Boogie
Restul e tacere

2 rows returned i

COD_FILM	NUME_FILM
107	Restul e tacere
108	Boogie

2 rows returned in 0.01 seconds

Pentru a le sorta după câmpul **Cod_film**:

SELECT Cod_film, Nume_film FROM Filme WHERE An =
TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4))
ORDER BY Cod_film

b) Numărul filmelor de comedie produse în România înainte de 1989.

COUNT(COD_FILM)
2

1 rows returned in 0.01

SELECT COUNT(Cod_film) FROM Filme
WHERE Tara= 'Romania' AND Gen='Comedie' AND An<1989

Obs: Pentru a nu face diferenţă între literele mari şi mici (ex: **'Comedie'** sau **'comedie'**, sau **'COMEDIE'** sau **'CoMEdiE'**), se poate folosi una din funcţiile:

- **LOWER(şir_de_caractere)** – transformă literele mari din şir în litere mici;
- **UPPER(şir_de_caractere)** – transformă literele mici din şir în litere mari;

În acest caz, rezolvarea problemei se poate face cu una din comenzile:

SELECT COUNT(Cod_film) FROM Filme
WHERE LOWER (Tara)= 'romania' AND LOWER (Gen)='comedie' AND An<1989

SAU

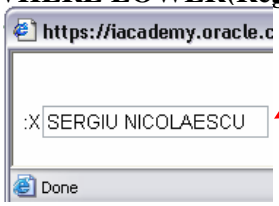
SELECT COUNT(Cod_film) FROM Filme
WHERE UPPER (Tara)= 'ROMANIA' AND UPPER (Gen)='COMEDIE' AND An<1989

c) Pentru a vedea efectul comenzii care va şterge din tabel, mai întâi afişăm filmele regizate de un regizor al cărui nume se citeşte, apoi ştergem filmele corespunzătoare şi afişăm tabelul final.

SELECT Cod_film, Nume_film FROM Filme WHERE LOWER(Regizor)= LOWER(:x)

COD_FILM	NUME_FILM
102	Dacii
104	Nea Marin miliardar

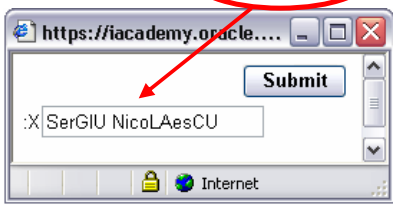
2 rows returned in 0.00 seconds



Ştergerea din tabel a liniilor selectate mai sus se face cu comanda:

DELETE Filme WHERE LOWER(Regizor) = LOWER(:x)

```
2 row(s) deleted.
```



Datele din tabelul final (sortate după codul filmului) se obţin cu comanda:

SELECT * FROM Filme ORDER BY Cod_film

COD_FILM	NUME_FILM	GEN	TARA	AN	REGIZOR
100	Razboiul stelelor	SF	SUA	1980	George Lucas
101	Harry Potter	Aventuri	SUA	2001	Chris Columbus
103	Balanta	Drama	Romania	1985	Lucian Pintilie
105	BD la munte si la mare	Comedie	Romania	1972	Geo Saizescu
106	Jurassyc Parck II	SF	SUA	1996	Spilberg
107	Restul e tacere	Drama	Romania	2009	Nae Caranfil
108	Boogie	Comedie	Romania	2009	Radu Muntean

7 rows returned in 0.00 seconds Download

d) Numărul filmelor regizate pentru fiecare regizor.

Pentru a rezolva această cerinţă, se poate folosi clauza **GROUP BY** astfel:

SELECT Regizor, COUNT(Cod_film) FROM Filme GROUP BY Regizor

REGIZOR	COUNT(COD_FILM)
Chris Columbus	1
Geo Saizescu	1
Nae Caranfil	1
Spilberg	1
Lucian Pintilie	1
Radu Muntean	1
George Lucas	1

7 rows returned in 0.01 seconds D

Tabelul final
(*după ştergere*)

Tabelul iniţial
(*înainte de ştergere*)

REGIZOR	COUNT(COD_FILM)
Chris Columbus	1
Geo Saizescu	1
Sergiu Nicolaescu	2
Nae Caranfil	1
Spilberg	1
Lucian Pintilie	1
Radu Muntean	1
George Lucas	1

8 rows returned in 0.01 seconds Dow

SUBIECTUL 6

Fie tabelul **MEDICAMENTE** (**Denumire_medicament, Cantitate, Pret**).

a) Listați medicamentele din farmacie care au cantitatea mai mare sau egală cu o valoare introdusă de la tastatură.

b) Care este prețul unui medicament cu numele introdus de la tastatură?

c) Afișați medicamentul/medicamentele cel(e) mai scump(e).

REZOLVARE:

Trebuie adăugat un câmp (coloană) tabelului care să poată fi cheie primară (să identifice unic o linie a tabelului) cu numele **Cod_medicamente**.

Pentru crearea tabelului se scrie comanda:

CREATE TABLE Medicamente (Cod_medicamente NUMBER(3) PRIMARY KEY, Nume_medicament VARCHAR2(30), Cantitate NUMBER(4), Pret NUMBER (8,3))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Medicamente VALUES (1,'Aspirina', 10, 5.25)
INSERT INTO Medicamente VALUES (2,'Piramidon', 30, 2.5)
INSERT INTO Medicamente VALUES (3,'Sirop de tuse', 50, 7.15)
INSERT INTO Medicamente VALUES (4,'Fastum gel', 20, 15.05)
INSERT INTO Medicamente VALUES (5,'Faringosept', 100, 4.75)
INSERT INTO Medicamente VALUES (6,'Paracetamol', 110, 5.25)
INSERT INTO Medicamente VALUES (7,'Saridon', 15, 15.05)
INSERT INTO Medicamente VALUES (8,'Aspirina Bayer', 15, 10.25)

Pentru a afișa conținutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_medicament**, se folosește comanda:

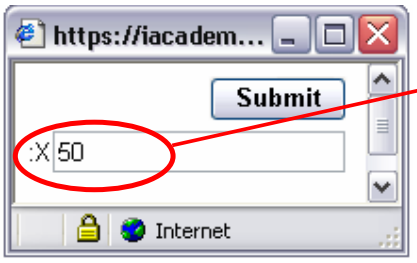
SELECT * FROM Medicamente ORDER BY Cod_medicamente

COD_MEDICAMENTE	NUME_MEDICAMENT	CANTITATE	PRET
1	Aspirina	10	5.25
2	Piramidon	30	2.5
3	Sirop de tuse	50	7.15
4	Fastum gel	20	15.05
5	Faringosept	100	4.75
6	Paracetamol	110	5.25
7	Saridon	15	15.05
8	Aspirina Bayer	15	10.25

8 rows returned in 0.00 seconds Download

a) Pentru rezolvarea cerinței, se poate folosi comanda:

SELECT Nume_medicament FROM Medicamente WHERE Cantitate >= :x

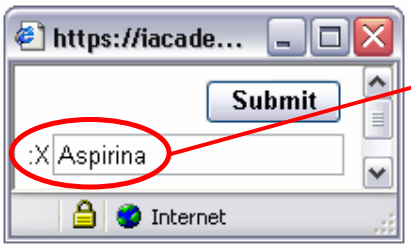


NUME_MEDICAMENT	CANTITATE
Paracetamol	110
Sirop de tuse	50
Faringosept	100

3 rows returned in 0.00 seconds D

b) Preţul unui medicament cu numele introdus de la tastatură

SELECT Pret FROM Medicamente WHERE Nume_medicament = :x



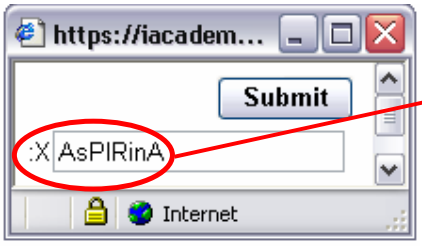
PRET
5.25

1 rows re

Obs: Pentru a nu face diferenţă între literele mari şi mici (ex: **'Aspirina'** sau **'aspirina'**, sau **'ASPIRINA'** sau **'asPIrINa'**), se poate folosi una din funcţiile:

- **LOWER(şir_de_caractere)** – transformă literele mari din şir în litere mici;
- **UPPER(şir_de_caractere)** – transformă literele mici din şir în litere mari;

SELECT Pret FROM Medicamente WHERE UPPER(Nume_medicament) = UPPER(:x)



PRET
5.25

1 rows re

c) Cele mai scumpe medicamente se pot găsi cu ajutorul funcţiei **MAX**:

SELECT Nume_medicament, Pret FROM Medicamente WHERE Pret =

(SELECT MAX(Pret) FROM Medicamente)

NUME_MEDICAMENT	PRET
Fastum gel	15.05
Saridon	15.05

2 rows returned in 0.01 seconds

Pretul maxim este:

MAX(PRET)
15.05

1 rows returned

SUBIECTUL 7

Fie tabelul **TELEFOANE** (**Nume, Telefon, Adresa, Oras**). Să se creeze structura acestuia care să corespundă următoarelor interogări:

a) Să se afişeze toate persoanele din tabel al căror nume începe cu litera "A" şi care locuiesc într-un anumit oraş X, introdus de la tastatură;

b) Să se găsească prima persoană din tabel al cărui nume este IONESCU şi să se afişeze înregistrarea respectivă.

REZOLVARE:

Trebuie adăugat un câmp (coloană) care să poată fi cheie primară (să identifice unic o linie a tabelului) cu numele **Cod_pers**.

Pentru crearea tabelului se scrie comanda:

CREATE TABLE Telefoane (Cod_pers NUMBER(3) PRIMARY KEY, Nume VARCHAR2(30), Telefon VARCHAR2(10), Oras VARCHAR2(20))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Telefoane VALUES (100,'Gigel', '0250731234', 'Rm. Valcea')
INSERT INTO Telefoane VALUES (101,'Ionel', '0250731235', 'Rm. Valcea')
INSERT INTO Telefoane VALUES (102,'Ana-Maria', '0251711234', 'Craiova')
INSERT INTO Telefoane VALUES (103,'Ioana', '0251711235', 'Craiova')
INSERT INTO Telefoane VALUES (104,'Petre', '0253701234', 'Cluj-Napoca')
INSERT INTO Telefoane VALUES (105,'Ana', '0253701235', 'Cluj-Napoca')
INSERT INTO Telefoane VALUES (106,'Adrian', '0251711236', 'Craiova')
INSERT INTO Telefoane VALUES (107,'Andreea', '0253701236', 'Cluj-Napoca')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_pers**, se foloseşte comanda:

SELECT * FROM Telefoane ORDER BY Cod_pers

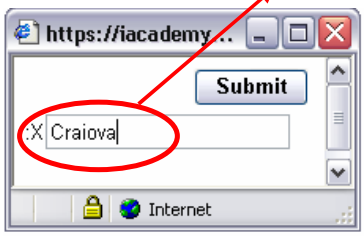
COD_PERS	NUME	TELEFON	ORAS
100	Gigel	0250731234	Rm. Valcea
101	Ionel	0250731235	Rm. Valcea
102	Ana-Maria	0251711234	Craiova
103	Ioana	0251711235	Craiova
104	Petre	0253701234	Cluj-Napoca
105	Ana	0253701235	Cluj-Napoca
106	Adrian	0251711236	Craiova
107	Andreea	0253701236	Cluj-Napoca

8 rows returned in 0.01 seconds Download

a) Pentru a rezolva această cerinţă, avem nevoie de:

- **SUBSTR(şir_de_caractere, poziţie_de_început, nr_caractere)** – întoarce un şir de caractere de lungime **nr_caractere**, începând cu caracterul de pe poziţia **poziţie_de_început**, din şirul **şir_de_caractere**.

SELECT Nume, Oras FROM Telefoane WHERE UPPER(SUBSTR(Nume,1,1))= 'A' AND UPPER(Oras) = UPPER(:x)



NUME	ORAS
Ana-Maria	Craiova
Adrian	Craiova

2 rows returned in 0.0

b) Ne asigurăm, mai întâi, că există persoana cu numele IONESCU în tabel, adică adăugăm următoarele linii:

INSERT INTO Telefoane VALUES (108,'Ionescu', '0253701237', 'Cluj-Napoca')
INSERT INTO Telefoane VALUES (109,'Ionescu', '0251701237', 'Craiova')

Pentru a găsi prima persoană (linie) din tabel al cărui nume e **Ionescu**, mai întâi selectăm liniile care au **Nume='Ionescu'** şi apoi selectăm prima linie: **ROWNUM=1**:

SELECT * FROM Telefoane WHERE Nume='Ionescu' AND ROWNUM=1

Toate liniile cu **Nume='Ionescu'**:

COD_PERS	NUME	TELEFON	ORAS
108	Ionescu	0253701237	Cluj-Napoca
109	Ionescu	0251701237	Craiova

2 rows returned in 0.00 seconds Download

Numai prima linie cu **Nume='Ionescu'** (rezultatul comenzii de mai sus):

COD_PERS	NUME	TELEFON	ORAS
108	Ionescu	0253701237	Cluj-Napoca

1 rows returned in 0.01 seconds Download

SUBIECTUL 8

La examenul de bacalaureat se trec în baza de date următoarele informații: **Nume, Prenume Nota_Rom1, Nota_LS, Nota_Rom2, Nota_probaD, Nota_probaE, Nota_ProbaF**.

Cerințe:

a) Afişarea elevilor admişi în ordinea descrescătoare a mediilor;

b) Afişarea elevilor respinşi în ordine alfabetică;

c) Afişați media aritmetică a mediilor elevilor admişi.

Pentru a fi admis, fiecare notă trebuie să fie mai mare sau egală cu 5 şi media trebuie să fie cel puțin 6.

REZOLVARE:

Trebuie adăugat un câmp (coloană) care să poată fi cheie primară (să identifice unic o linie a tabelului) cu numele **Cod_pers**.

Pentru crearea tabelului se scrie comanda:

CREATE TABLE Rezultate (Cod_pers NUMBER(3) PRIMARY KEY, Nume VARCHAR2(30) NOT NULL, Prenume VARCHAR2(30) NOT NULL, Nota_Rom1 NUMBER(4,2), Nota_LS NUMBER(4,2), Nota_Rom2 NUMBER(4,2), Nota_probaD NUMBER(4,2), Nota_probaE NUMBER(4,2), Nota_ProbaF NUMBER(4,2))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Rezultate VALUES (100,'Ionescu', 'Gigel', 10, 9.5, 9.80, 9.25, 9.75, 10)
INSERT INTO Rezultate VALUES (101,'Ionescu', 'Ramona', 10, 10, 9.5, 8.25, 8.75, 9.5)
INSERT INTO Rezultate VALUES (102,'Popescu', 'Ionel', 9, 9.5, 9.20, 9.25, 8.75, 10)
INSERT INTO Rezultate VALUES (103,'Popescu', 'Maria', 9, 9.5, 9.45, 9.75, 9.25, 10)
INSERT INTO Rezultate VALUES (104,'Radu', 'Roxana', 10, 9.75, 9.80, 9.75, 10, 10)
INSERT INTO Rezultate VALUES (105,'Radulescu', 'Ioana', 9.5, 9.15, 8.80, 8.25, 8.75, 9)
INSERT INTO Rezultate VALUES (106,'Toma', 'George', 8, 8.5, 8.80, 9.25, 8.75, 10)
INSERT INTO Rezultate VALUES (107,'Voinea', 'Ana', 10, 9.5, 9.20, 9.15, 8.75, 10)
```

Pentru a afişa conținutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_pers**, se foloseşte comanda:

SELECT * FROM Rezultate ORDER BY Cod_pers

COD_PERS	NUME	PRENUME	NOTA_ROM1	NOTA_LS	NOTA_ROM2	NOTA_PROBAD	NOTA_PROBAE	NOTA_PROBAF
100	Ionescu	Gigel	10	9.5	9.8	9.25	9.75	10
101	Ionescu	Ramona	10	10	9.5	8.25	8.75	9.5
102	Popescu	Ionel	9	9.5	9.2	9.25	8.75	10
103	Popescu	Maria	9	9.5	9.45	9.75	9.25	10
104	Radu	Roxana	10	9.75	9.8	9.75	10	10
105	Radulescu	Ioana	9.5	9.15	8.8	8.25	8.75	9
106	Toma	George	8	8.5	8.8	9.25	8.75	10
107	Voinea	Ana	10	9.5	9.2	9.15	8.75	10

8 rows returned in 0.01 seconds Download

a) Pentru afişarea elevilor admişi în ordinea descrescătoare a mediilor, se poate folosi comanda:

SELECT Nume, Prenume, TRUNC((Nota_Rom1+Nota_LS+Nota_Rom2+Nota_probaD+ Nota_probaE+ Nota_ProbaF)/6,2) AS Media FROM Rezultate
WHERE Nota_Rom1>=5 AND Nota_LS>=5 AND Nota_Rom2>=5 AND Nota_probaD>=5 AND Nota_probaE>=5 AND Nota_ProbaF>=5 AND
(Nota_Rom1+Nota_LS+Nota_Rom2+Nota_probaD+ Nota_probaE+ Nota_ProbaF)/6>=6
ORDER BY Media DESC

NUME	PRENUME	MEDIA
Radu	Roxana	9.88
Ionescu	Gigel	9.71
Popescu	Maria	9.49
Voinea	Ana	9.43
Ionescu	Ramona	9.33
Popescu	Ionel	9.28
Radulescu	Ioana	8.9
Toma	George	8.88

8 rows returned in 0.01 seconds

b) afişarea elevilor respinşi în ordine alfabetică.

Pentru că în tabelul nostru nu există elevi respinşi, mai adăugăm date:

INSERT INTO Rezultate VALUES (108,'Iancu', 'Tudor', 8, 6.5, 4.75, 6.25, 6.75, 8)
INSERT INTO Rezultate VALUES (109,'Tudoroiu', 'Rodica', 6, 6, 5.5, 5.25, 5.75, 5.15)

Se obține tabelul următor:

COD_PERS	NUME	PRENUME	NOTA_ROM1	NOTA_LS	NOTA_ROM2	NOTA_PROBAD	NOTA_PROBAE	NOTA_PROBAF
100	Ionescu	Gigel	10	9.5	9.8	9.25	9.75	10
101	Ionescu	Ramona	10	10	9.5	8.25	8.75	9.5
102	Popescu	Ionel	9	9.5	9.2	9.25	8.75	10
103	Popescu	Maria	9	9.5	9.45	9.75	9.25	10
104	Radu	Roxana	10	9.75	9.8	9.75	10	10
105	Radulescu	Ioana	9.5	9.15	8.8	8.25	8.75	9
106	Toma	George	8	8.5	8.8	9.25	8.75	10
107	Voinea	Ana	10	9.5	9.2	9.15	8.75	10
108	Iancu	Tudor	8	6.5	4.75	6.25	6.75	8
109	Tudoroiu	Rodica	6	6	5.5	5.25	5.75	5.15

10 rows returned in 0.01 seconds Download

SELECT Nume, Prenume FROM Rezultate
WHERE Nota_Rom1<5 OR Nota_LS<5 OR Nota_Rom2<5 OR Nota_probaD<5 OR Nota_probaE<5 OR Nota_ProbaF<5 OR (Nota_Rom1+Nota_LS+Nota_Rom2+Nota_probaD+ Nota_probaE+ Nota_ProbaF)/6<6 ORDER BY Nume, Prenume

NUME	PRENUME
Iancu	Tudor
Tudoroiu	Rodica

2 rows returned in 0.01 seconds

Pentru calculul şi afişarea mediilor, pentru a verifica rezultatul obținut, folosim comanda:

SELECT Nume, Prenume,
TRUNC((Nota_Rom1+Nota_LS+Nota_Rom2+
Nota_probaD+ Nota_probaE+ Nota_ProbaF)/6,2)
AS Media FROM Rezultate ORDER BY Media DESC

NUME	PRENUME	MEDIA
Radu	Roxana	9.88
Ionescu	Gigel	9.71
Popescu	Maria	9.49
Voinea	Ana	9.43
Ionescu	Ramona	9.33
Popescu	Ionel	9.28
Radulescu	Ioana	8.9
Toma	George	8.88
Iancu	Tudor	6.7
Tudoroiu	Rodica	5.6

10 rows returned in 0.01 seconds

c) Pentru afişarea mediei aritmetice a mediilor elevilor admişi putem folosi comanda:

```
SELECT TRUNC(AVG(TRUNC((Nota_Rom1+Nota_LS+Nota_Rom2+Nota_probaD+
Nota_probaE+ Nota_ProbaF)/6,2)),2) AS Media_generala FROM Rezultate
WHERE Nota_Rom1>=5 AND  Nota_LS>=5 AND Nota_Rom2>=5 AND Nota_probaD>=5
AND Nota_probaE>=5 AND Nota_ProbaF>=5 AND
(Nota_Rom1+Nota_LS+Nota_Rom2+Nota_probaD+ Nota_probaE+ Nota_ProbaF)/6>=6
```

MEDIA_GENERALA
9.36

1 rows returned in 0.00 seconds

SUBIECTUL 9

O societate de distribuție presă ține evidența abonaților săi. O persoană poate avea abonamente la mai multe publicații, iar o publicație poate fi obiectul mai multor abonamente. Pentru fiecare abonament se vor reține următoarele informații: suma, data de început şi data de sfârşit. Creați tabelele şi populați-le cu date pertinente pentru a răspunde următoarelor cerințe:

a) Afişați pentru o anumită persoană cu numele citit de la tastatură toate revistele la care este abonat;

b) Revista "Atestat" îşi majorează toate abonamentele cu x %. Să se opereze modificările corespunzătoare;

c) Ştergeți toate abonamentele expirate (data de sfârşit mai mare decât data curentă).

REZOLVARE:

Trebuie create următoarele tabele:

- **PERSOANE** cu structura:

Cod_pers – codul unei persoane (CNP) care va fi cheie primară;
Nume – numele şi prenumele persoanei;
Adresa – adresa persoanei.

- **PUBLICAȚII** cu structura:

Cod_pub – codul unei publicații care va fi cheie primară;
Nume – numele (titlul) publicației;
Pret – prețul abonamentului lunar.

- **ABONAMENTE** cu structura:

Cod_1 – codul unei publicații care va fi cheie străină;
Cod_2 – codul unei persoane (CNP) care va fi cheie străină;
Suma – valoarea (în lei) a abonamentului;
Data1 – data de început a abonamentului;
Data2 – data de sfârşit a abonamentului.

Pentru crearea tabelului **PERSOANE** se scrie comanda:

```
CREATE TABLE Persoane (Cod_pers NUMBER(3) PRIMARY KEY, Nume VARCHAR2(30)
NOT NULL, Adresa VARCHAR2(50))
```

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Persoane VALUES (100,'Ionescu Gigel', 'Aleea Trandafirilor, nr.9')
INSERT INTO Persoane VALUES (101,'Ionescu Maria', 'Str. O. Goga, nr. 12')
INSERT INTO Persoane VALUES (102,'Popescu Daniel', 'Str. M. Basarab, nr. 19')
INSERT INTO Persoane VALUES (103,'Popescu Maria', 'B-dul T. Vladimirescu, nr. 22')
```

INSERT INTO Persoane VALUES (104,'Radu Roxana', 'Str. St. Voda, nr. 16')
INSERT INTO Persoane VALUES (105,'Radulescu Ioana', 'Aleea Olanesti, nr. 3')
INSERT INTO Persoane VALUES (106,'Toma George', 'Str. O. Goga, nr. 22')
INSERT INTO Persoane VALUES (107,'Voinea Ana', 'Aleea Tuberozelor, nr. 17')

Pentru a afişa conţinutul tabelului Persoane, în ordine crescătoare a valorilor din câmpul **Cod_pers**, se foloseşte comanda:

SELECT * FROM Persoane ORDER BY Cod_pers

COD_PERS	NUME	ADRESA
100	Ionescu Gigel	Aleea Trandafirilor, nr.9
101	Ionescu Maria	Str. O. Goga, nr. 12
102	Popescu Daniel	Str. M. Basarab, nr. 19
103	Popescu Maria	B-dul T. Vladimirescu, nr. 22
104	Radu Roxana	Str. St. Voda, nr. 16
105	Radulescu Ioana	Aleea Olanesti, nr. 3
106	Toma George	Str. O. Goga, nr. 22
107	Voinea Ana	Aleea Tuberoyelor, nr. 17

8 rows returned in 0.01 seconds Download

Pentru crearea tabelului **PUBLICATII** se scrie comanda:

CREATE TABLE Publicatii (Cod_pub NUMBER(3) PRIMARY KEY, Nume VARCHAR2(30) NOT NULL, Pret NUMBER(5,2))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Publicatii VALUES (10,'Evenimentul zilei', 20)
INSERT INTO Publicatii VALUES (11,'Atestat', 11.75)
INSERT INTO Publicatii VALUES (12,'Popcorn', 4)
INSERT INTO Publicatii VALUES (13,'Bravo', 3.5)
INSERT INTO Publicatii VALUES (14,'Curierul de Valcea', 25.50)
INSERT INTO Publicatii VALUES (15,'Rebus Flacara', 5.5)

Pentru a afişa conţinutul tabelului Publicatii, în ordine crescătoare a valorilor din câmpul **Cod_pers**, se foloseşte comanda:

SELECT * FROM Publicatii ORDER BY Cod_pub

COD_PUB	NUME	PRET
10	Evenimentul zilei	20
11	Atestat	11.75
12	Popcorn	4
13	Bravo	3.5
14	Curierul de Valcea	25.5
15	Rebus Flacara	5.5

6 rows returned in 0.01 seconds Do

Pentru crearea tabelului **ABONAMENTE** se scrie comanda:

CREATE TABLE Abonamente (Cod_1 NUMBER(3) REFERENCES Persoane(Cod_pers), Cod_2 NUMBER(3) REFERENCES Publicatii(Cod_pub), Suma NUMBER(7,2), Data1 DATE, Data2 DATE)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Abonamente VALUES (100, 10, 40, '01-JAN-2009', '01-MAR-2009')
INSERT INTO Abonamente VALUES (100, 14, 51, '01-MAR-2009', '01-MAY-2009')
INSERT INTO Abonamente VALUES (101, 11, 47, '01-MAR-2009', '01-JUL-2009')
INSERT INTO Abonamente VALUES (102, 11, 47, '01-JAN-2009', '01-MAY-2009')
INSERT INTO Abonamente VALUES (103, 12, 20, '01-JAN-2009', '01-JUN-2009')
INSERT INTO Abonamente VALUES (104, 14, 102, '01-MAR-2009', '01-JUL-2009')
INSERT INTO Abonamente VALUES (105, 13, 7, '01-MAR-2009', '01-MAY-2009')
INSERT INTO Abonamente VALUES (105, 15, 11, '01-APR-2009', '01-JUN-2009')
INSERT INTO Abonamente VALUES (106, 15, 11, '01-MAR-2009', '01-MAY-2009')
INSERT INTO Abonamente VALUES (107, 11, 23.5, '01-JAN-2009', '01-MAR-2009')

Pentru a afişa conţinutul tabelului **ABONAMENTE**, în ordine crescătoare a valorilor din câmpul **Cod_pers**, se foloseşte comanda:

SELECT * FROM Abonamente ORDER BY Cod_1, Cod_2

COD_1	COD_2	SUMA	DATA1	DATA2
100	10	40	01-JAN-09	01-MAR-09
100	14	51	01-MAR-09	01-MAY-09
101	11	47	01-MAR-09	01-JUL-09
102	11	47	01-JAN-09	01-MAY-09
103	12	20	01-JAN-09	01-JUN-09
104	14	102	01-MAR-09	01-JUL-09
105	13	7	01-MAR-09	01-MAY-09
105	15	11	01-APR-09	01-JUN-09
106	15	11	01-MAR-09	01-MAY-09
107	11	23.5	01-JAN-09	01-MAR-09

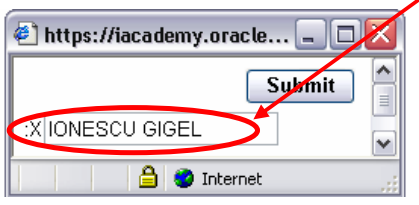
10 rows returned in 0.01 seconds Download

a) Afişaţi pentru o anumită persoană cu numele citit de la tastatură toate revistele la care este abonat.

Pentru rezolvarea acestei cerinţe, putem folosi comanda:

SELECT Nume FROM Publicatii WHERE Cod_pub IN
(SELECT Cod_2 FROM Abonamente WHERE Cod_1 =
(SELECT Cod_pers FROM Persoane WHERE UPPER(Nume) = UPPER(:x)))

Dacă se introduce numele IONESCU GIGEL, atunci se vor afişa publicaţiile:



NUME
Curierul de Valcea
Evenimentul zilei

2 rows returned in 0.02 seconds

b) Revista "Atestat" îşi majorează toate abonamentele cu x %. Să se opereze modificările corespunzătoare.

Modificăm conţinutul tabelului **ABONAMENTE**:

UPDATE Abonamente SET Suma = Suma+(Suma* :x)/100 WHERE Cod_2 =
(SELECT Cod_pub FROM Publicatii WHERE UPPER(Nume) = 'ATESTAT')

În urma execuției comenzii de mai sus, dacă pentru x se introduce valoarea 10, se obține:

COD_1	COD_2	SUMA	DATA1	DATA2
100	10	40	01-JAN-09	01-MAR-09
107	11	25.85	01-JAN-09	01-MAR-09
101	11	51.7	01-MAR-09	01-JUL-09
102	11	51.7	01-JAN-09	01-MAY-09
103	12	20	01-JAN-09	01-JUN-09
105	13	7	01-MAR-09	01-MAY-09
104	14	102	01-MAR-09	01-JUL-09
100	14	51	01-MAR-09	01-MAY-09
105	15	11	01-APR-09	01-JUN-09
106	15	11	01-MAR-09	01-MAY-09

10 rows returned in 0.01 seconds Download

COD_PUB	NUME	PRET
10	Evenimentul zilei	20
11	Atestat	11.75
12	Popcorn	4
13	Bravo	3.5
14	Curierul de Valcea	25.5
15	Rebus Flacara	5.5

6 rows returned in 0.01 seconds Do

c) Ştergeţi toate abonamentele expirate (data de sfârşit mai mare decât data curentă).

Afişăm mai întâi abonamentele de şters:

SELECT * FROM Abonamente WHERE Data2 < CURRENT_DATE ORDER BY Cod_1

COD_1	COD_2	SUMA	DATA1	DATA2
100	10	40	01-JAN-09	01-MAR-09
107	11	25.85	01-JAN-09	01-MAR-09

2 rows returned in 0.01 seconds Download

```
2 row(s) deleted.


0.01 seconds
```

Pentru ştergerea acestor abonamente, putem folosi:

DELETE FROM Abonamente WHERE Data2 < CURRENT_DATE ORDER BY Cod_1

SUBIECTUL 10

Există următoarea listă de interogări la care trebuie urgent răspunsuri din partea bibliotecarului şcolii:

a) Lista tuturor cărţilor din bibliotecă ce aparţin editurii Polirom;
b) Ce cărţi au intrat în inventarul bibliotecii în anul curent?
c) Câte cărţi ale unui autor cu numele citit de la tastatură sunt în bibliotecă?
d) Lista tuturor cărţilor din domeniul informaticii.

Construiţi baza de date corespunzătoare şi rezolvaţi cele 4 cerinţe. Ţineţi cont de faptul că o carte poate avea mai mulţi autori, iar un autor evident poate scrie mai multe cărţi.

REZOLVARE:

Trebuie create următoarele tabele:

- **CARTI** cu structura:

Cod_carte – codul unei cărţi care va fi cheie primară;
Titlu – titlul cărţii;
Domeniu – domeniul din care face parte cartea;
Editura – numele editurii la care a fost publicată;
An – anul în care a intrat în inventarul bibliotecii.

- **SCRIITORI** cu structura:

Cod_pers – codul autorului (CNP) care va fi cheie primară;;
Nume – numele şi prenume autor.

- **LISTA_CARTI** cu structura:

Cod_1 – codul cărţii care va fi cheie străină;
Cod_2 – codul autorului care va fi cheie străină;

Considerăm că în tabelul **LISTA_CARTI** se găsesc date care fac legătura între cărţile din tabelul **CARTI** şi autorii lor din tabelul **SCRIITORI**.

Pentru crearea tabelului **CARTI** se scrie comanda:

CREATE TABLE Carti (Cod_carte NUMBER(3) PRIMARY KEY, Titlu VARCHAR2(30) NOT NULL, Domeniu VARCHAR2(30), Editura VARCHAR2(30) NOT NULL, An NUMBER(4))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Carti VALUES (10, 'Fluviul diamantelor', 'Aventuri', 'Orpheus', 1994)
INSERT INTO Carti VALUES (20, 'Ciresarii', 'Aventuri','Polirom', 1998)
INSERT INTO Carti VALUES (30, 'Pearl Harbor', 'Istorie','Polirom', 2001)
INSERT INTO Carti VALUES (40, 'Contesa de Charny', 'Istorie', 'Orpheus', 1996)
INSERT INTO Carti VALUES (50, 'Limbajul Java', 'Informatica', 'Nemira', 2009)
INSERT INTO Carti VALUES (60, 'Limbajul Pascal', 'Informatica', 'LS Infomat', 2003)
INSERT INTO Carti VALUES (70, 'Imperiul atomului', 'SF', 'Nemira', 1994)
INSERT INTO Carti VALUES (80, 'Caverne de otel', 'SF', 'Univers', 2009)

Pentru a afişa conţinutul tabelului CARTI, în ordine crescătoare a valorilor din câmpul **Cod_carte**, se foloseşte comanda:

SELECT * FROM Carti ORDER BY Cod_carte

COD_CARTE	TITLU	DOMENIU	EDITURA	AN
10	Fluviul diamantelor	Aventuri	Orpheus	1994
20	Ciresarii	Aventuri	Polirom	1998
30	Pearl Harbor	Istorie	Polirom	2001
40	Contesa de Charny	Istorie	Orpheus	1996
50	Limbajul Java	Informatica	Nemira	2009
60	Limbajul Pascal	Informatica	LS Infomat	2003
70	Imperiul atomului	SF	Nemira	1994
80	Caverne de otel	SF	Univers	2009

8 rows returned in 0.01 seconds Download

Pentru crearea tabelului **SCRIITORI** se scrie comanda:

CREATE TABLE Scriitori (Cod_pers NUMBER(13) PRIMARY KEY, Nume VARCHAR2(30) NOT NULL)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Scriitori VALUES (100, 'Isaac Asimov')
INSERT INTO Scriitori VALUES (123, 'A. E. van Vogt')
INSERT INTO Scriitori VALUES (130, 'Alexandre Dumas')
INSERT INTO Scriitori VALUES (133, 'Constantin Chirita')
INSERT INTO Scriitori VALUES (145, 'Edgar Wallace')
INSERT INTO Scriitori VALUES (150, 'Walter Lord')

INSERT INTO Scriitori VALUES (160, 'Tudor Sorin')
INSERT INTO Scriitori VALUES (170, 'Carmen Popescu')

Pentru a afişa conținutul tabelului SCRIITORI, în ordine crescătoare a valorilor din câmpul **Cod_pers**, se foloseşte comanda:

SELECT * FROM Scriitori ORDER BY Cod_pers

COD_PERS	NUME
100	Isaac Asimov
123	A. E. van Vogt
130	Alexandre Dumas
133	Constantin Chirita
145	Edgar Wallace
150	Walter Lord
160	Tudor Sorin
170	Carmen Popescu

8 rows returned in 0.01 seconds

Pentru crearea tabelului **LISTA_CARTI** se scrie comanda:

CREATE TABLE Lista_carti (Cod_carte NUMBER(3) REFERENCES Carti(Cod_carte), Cod_pers NUMBER(13) REFERENCES Scriitori(Cod_pers))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Lista_carti VALUES (10, 145)
INSERT INTO Lista_carti VALUES (10, 123)
INSERT INTO Lista_carti VALUES (20, 133)
INSERT INTO Lista_carti VALUES (30, 150)
INSERT INTO Lista_carti VALUES (40, 130)
INSERT INTO Lista_carti VALUES (50, 160)
INSERT INTO Lista_carti VALUES (50, 170)
INSERT INTO Lista_carti VALUES (60, 160)
INSERT INTO Lista_carti VALUES (70, 100)
INSERT INTO Lista_carti VALUES (70, 123)
INSERT INTO Lista_carti VALUES (80, 100)

Pentru a afişa conținutul tabelului CARTI, în ordine crescătoare a valorilor din câmpul **Cod_carte**, se foloseşte comanda:

SELECT * FROM Lista_carti ORDER BY Cod_carte

COD_CARTE	COD_PERS
10	145
10	123
20	133
30	150
40	130
50	160
50	170
60	160
70	123
70	100
More than 10 rows available. Increase rows selector to view more rows.	

10 rows returned in 0.01 seconds Download

a) Lista tuturor cărților din bibliotecă ce aparțin editurii Polirom.

SELECT Cod_carte, Titlu FROM Carti WHERE UPPER(Editura) = 'POLIROM' ORDER BY Cod_carte

COD_CARTE	TITLU
20	Ciresarii
30	Pearl Harbor

2 rows returned in 0.01 seconds

b) Ce cărți au intrat în inventarul bibliotecii în anul curent?

Pentru a extrage anul din data curentă (vezi subiectul 5, punctul a) putem folosi expresia:

TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4))

Pentru a răspunde cerinței a), se poate folosi comanda:

SELECT Titlu FROM Carti WHERE An = TO_NUMBER(SUBSTR(TO_CHAR(CURRENT_DATE, 'DD.MON.YYYY'),8,4))

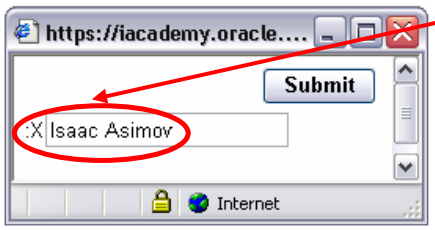
TITLU
Limbajul Java
Caverne de otel

2 rows returned i

c) Câte cărți ale unui autor cu numele citit de la tastatură sunt în bibliotecă?

SELECT COUNT(Cod_carte) FROM Lista_carti WHERE Cod_pers = (SELECT Cod_pers FROM Scriitori WHERE UPPER(Nume) = UPPER(:x))

Pentru valoarea introdusă de la tastatură **'Isaac Asimov'** se obține:



COUNT(COD_CARTE)
2

1 rows returned in 0.01 s

d) Lista tuturor cărților din domeniul informaticii.

SELECT Titlu FROM Carti WHERE LOWER(Domeniu) = 'informatica' ORDER BY Cod_carte

:

TITLU
Limbajul Java
Limbajul Pascal

2 rows returned

Tabelul **Carti:**

COD_CARTE	TITLU	DOMENIU	EDITURA	AN
10	Fluviul diamantelor	Aventuri	Orpheus	1994
20	Ciresarii	Aventuri	Polirom	1998
30	Pearl Harbor	Istorie	Polirom	2001
40	Contesa de Charny	Istorie	Orpheus	1996
50	Limbajul Java	Informatica	Nemira	2009
60	Limbajul Pascal	Informatica	LS Infomat	2003
70	Imperiul atomului	SF	Nemira	1994
80	Caverne de otel	SF	Univers	2009

8 rows returned in 0.01 seconds Download

SUBIECTUL 11

Se consideră o bază de date cu următoarele tabele: **ANGAJAȚI** (**Id, Nume, Salariu**), **ANGAJĂRI** (**Id_Angajat, Id_Departament**) şi **DEPARTAMENTE** (**Id_Departament, Nume, Id_Manager, Etaj**).

Cerințe:

a) Afişaţi numele angajaților care lucrează la etajul 10 şi au salariul mai mic decât 850;

b) Angajații din departamentul „Jucării" primesc o mărire de salariu de 10%. Afişaţi numele şi noul salariu al angajaților din acest departament;

c) Afişaţi numele angajaților cu salariu maxim pe fiecare departament.

REZOLVARE:

Pentru că la descrierea structurii tabelului **ANGAJARI** se face referire la conținutul tabelului **ANGAJATI** (**Id_Angajat** este cheie străină pentru **Angajari**) şi la conținutul tabelului **DEPARTAMENTE** (**Id_departament** este cheie străină pentru **Angajari**), trebuie create mai întâi tabelele **ANGAJAȚI** şi **DEPARTAMENTE**.

Pentru crearea tabelului **ANGAJATI** se scrie comanda:

```
CREATE TABLE Angajati (Id NUMBER(3) PRIMARY KEY, Nume VARCHAR2(30) NOT NULL, Salariu NUMBER(5))
```

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Angajati VALUES (100, 'Ionescu Gigel', 1200)
INSERT INTO Angajati VALUES (101,'Ionescu Maria', 1150)
INSERT INTO Angajati VALUES (102,'Popescu Daniel', 775)
INSERT INTO Angajati VALUES (103,'Popescu Maria', 850)
INSERT INTO Angajati VALUES (104,'Radu Roxana', 775)
INSERT INTO Angajati VALUES (105,'Radulescu Ioana', 815)
INSERT INTO Angajati VALUES (106,'Toma George', 847)
INSERT INTO Angajati VALUES (107,'Voinea Ana', 1125)
```

Pentru a afişa conținutul tabelului ANGAJATI, în ordine crescătoare a valorilor din câmpul **Id**, se foloseşte comanda:

```
SELECT * FROM Angajati ORDER BY Id
```

ID	NUME	SALARIU
100	Ionescu Gigel	1200
101	Ionescu Maria	1150
102	Popescu Daniel	775
103	Popescu Maria	850
104	Radu Roxana	775
105	Radulescu Ioana	815
106	Toma George	847
107	Voinea Ana	1125

8 rows returned in 0.01 seconds

Pentru crearea tabelului **DEPARTAMENTE** se scrie comanda:

```
CREATE TABLE Departamente (Id_Departament NUMBER(2) PRIMARY KEY, Nume VARCHAR2(30) NOT NULL, Id_Manager NUMBER(3), Etaj NUMBER(2))
```

Pentru “popularea” cu date se scriu, pe rând, comenzile:

INSERT INTO Departamente VALUES (31,'Cosmetice', 100,2)
INSERT INTO Departamente VALUES (44,'Jucarii', 101,3)
INSERT INTO Departamente VALUES (52,'Articole sportive', 107,10)

Pentru a afişa conţinutul tabelului DEPARTAMENTE, în ordine crescătoare a valorilor din câmpul **Id_Departament**, se foloseşte comanda:

SELECT * FROM Departamente ORDER BY Id_Departament

ID_DEPARTAMENT	NUME	ID_MANAGER	ETAJ
31	Cosmetice	100	2
44	Jucarii	101	3
52	Articole sportive	107	10

3 rows returned in 0.01 seconds Download

Pentru crearea tabelului **ANGAJARI** se scrie comanda:

CREATE TABLE Angajari (Id_Angajat NUMBER(3) REFERENCES Angajati(Id), Id_departament NUMBER(2) REFERENCES Departamente(Id_departament))

Pentru “popularea” cu date se scriu, pe rând, comenzile:

INSERT INTO Angajari VALUES (100, 31)
INSERT INTO Angajari VALUES (101, 31)
INSERT INTO Angajari VALUES (102, 31)
INSERT INTO Angajari VALUES (103, 44)
INSERT INTO Angajari VALUES (104, 44)
INSERT INTO Angajari VALUES (105, 52)
INSERT INTO Angajari VALUES (106, 52)
INSERT INTO Angajari VALUES (107, 52)

Pentru a afişa conţinutul tabelului ANGAJARI, în ordine crescătoare a valorilor din câmpul **Id_departament**, se foloseşte comanda:

SELECT * FROM Angajari ORDER BY Id_Departament, Id_Angajat

ID_ANGAJAT	ID_DEPARTAMENT
100	31
101	31
102	31
103	44
104	44
105	52
106	52
107	52

8 rows returned in 0.00 seconds

a) Numele angajaţilor care lucrează la etajul 10 şi au salariul mai mic decât 850:

SELECT Nume FROM Angajati WHERE Salariu < 850 AND Id IN
(SELECT Id_Angajat FROM Angajari WHERE Id_Departament =
(SELECT Id_Departament FROM Departamente WHERE Etaj = 10))

Rezultatul obținut:

NUME
Radulescu Ioana
Toma George

2 rows returned ir

ID	NUME	SALARIU
100	Ionescu Gigel	1200
101	Ionescu Maria	1150
102	Popescu Daniel	775
103	Popescu Maria	850
104	Radu Roxana	775
105	Radulescu Ioana	815
106	Toma George	847
107	Voinea Ana	1125

8 rows returned in 0.01 seconds

ID_ANGAJAT	ID_DEPARTAMENT
100	31
101	31
102	31
103	44
104	44
105	52
106	52
107	52

8 rows returned in 0.00 seconds

ID_DEPARTAMENT	NUME	ID_MANAGER	ETAJ
31	Cosmetice	100	2
44	Jucarii	101	3
52	Articole sportive	107	10

3 rows returned in 0.01 seconds Download

b) Numele şi noul salariu al angajaților din departamentul „Jucarii”:

UPDATE Angajati SET
Salariu = 1.10*Salariu WHERE Id IN (SELECT Id_Angajat FROM Angajari WHERE Id_Departament = (SELECT Id_Departament FROM Departamente WHERE UPPER(Nume) = 'JUCARII'))

În cazul datelor din tabel, se actualizează 2 înregistrări şi se obține tabelul:

ID	NUME	SALARIU
100	Ionescu Gigel	1200
101	Ionescu Maria	1150
102	Popescu Daniel	775
103	Popescu Maria	935
104	Radu Roxana	853
105	Radulescu Ioana	815
106	Toma George	847
107	Voinea Ana	1125

8 rows returned in 0.01 seconds

Tabelul final

ID	NUME	SALARIU
100	Ionescu Gigel	1200
101	Ionescu Maria	1150
102	Popescu Daniel	775
103	Popescu Maria	850
104	Radu Roxana	775
105	Radulescu Ioana	815
106	Toma George	847
107	Voinea Ana	1125

Tabelul inițial

8 rows returned in 0.01 seconds

c) Numele angajaților cu salariu maxim pe fiecare departament:

SELECT Nume, Salariu FROM Angajati WHERE Salariu IN
(SELECT MAX(Salariu) FROM Angajati, Angajari WHERE Id = Id_angajat
GROUP BY Id_Departament)

NUME	SALARIU
Ionescu Gigel	1200
Popescu Maria	935
Voinea Ana	1125

3 rows returned in 0.01 seconds

SUBIECTUL 12

Se consideră o bază de date cu următoarele tabele: **FURNIZORI** (**Id, Nume, Localitate**), **COMPONENTE** (**Id, Nume, Culoare**) şi **COMENZI** (**Id_Furnizor, Id_Componenta, Cantitate**).

Cerințe:

a) Afişaţi toţi furnizorii din oraşul .... (citit de la tastatură);

b) Afişaţi componentele de culoare „roşie" care au fost comandate de la furnizori din „Braşov";

c) Afişaţi furnizorul/furnizorii care au oferit componente de culoare „verde" în cantitate maximă.

REZOLVARE:

Pentru că tabelul **COMENZI** face referire la tabelele **FURNIZORI** şi **COMPONENTE**, mai întâi se creează acestea din urmă.

Pentru crearea tabelului **FURNIZORI** se scrie comanda:

CREATE TABLE Furnizori (Id NUMBER(3) PRIMARY KEY, Nume VARCHAR2(20), Localitate VARCHAR2(20))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Furnizori VALUES (100,'Arta ceramica', 'Brasov')
INSERT INTO Furnizori VALUES (101,'Georgia SRL', 'Bucuresti')
INSERT INTO Furnizori VALUES (102,'Vopseluri SRL', 'Pitesti')
INSERT INTO Furnizori VALUES (103,'Pictorul SA', 'Brasov')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Id**, se foloseşte comanda:

SELECT * FROM Furnizori ORDER BY Id

ID	NUME	LOCALITATE
100	Arta ceramica	Brasov
101	Georgia SRL	Bucuresti
102	Vopseluri SRL	Pitesti
103	Pictorul SA	Brasov

4 rows returned in 0.01 seconds

Pentru crearea tabelului **COMPONENTE** se scrie comanda:

CREATE TABLE Componente (Id NUMBER(3) PRIMARY KEY, Nume VARCHAR2(20), Culoare VARCHAR2(20))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Componente VALUES (500,'Rama tablou 20x20', 'Neagra')
INSERT INTO Componente VALUES (600,'Panza tablou 20x20', 'Alba')
INSERT INTO Componente VALUES (700,'Vopsea tip v1', 'Rosu')
INSERT INTO Componente VALUES (510,'Rama tablou 40x20', 'Rosu')
INSERT INTO Componente VALUES (605, 'Panza tablou 40x20' , 'Bej')
INSERT INTO Componente VALUES (701,'Vopsea tip v2', 'Verde')
INSERT INTO Componente VALUES (705,'Vopsea tip v5', 'Galben')
INSERT INTO Componente VALUES (725,'Vopsea Tempera', 'Rosu')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Id**, se foloseşte comanda:

SELECT * FROM Componente ORDER BY Id

ID	NUME	CULOARE
500	Rama tablou 20x20	Neagra
510	Rama tablou 40x20	Rosu
600	Panza tablou 20x20	Alba
605	Panza tablou 40x20	Bej
700	Vopsea tip v1	Rosu
701	Vopsea tip v2	Verde
705	Vopsea tip v5	Galben
725	Vopsea Tempera	Rosu

8 rows returned in 0.01 seconds

Pentru crearea tabelului **COMENZI** se scrie comanda:

CREATE TABLE Comenzi (Id_Furnizor NUMBER(3) REFERENCES Furnizori(Id), Id_Componenta NUMBER(3) REFERENCES Componente(Id), Cantitate NUMBER (3))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Comenzi VALUES (100, 500, 20)
INSERT INTO Comenzi VALUES (100, 600, 15)
INSERT INTO Comenzi VALUES (100, 510, 10)
INSERT INTO Comenzi VALUES (101, 605, 21)
INSERT INTO Comenzi VALUES (101, 700, 11)
INSERT INTO Comenzi VALUES (101, 705, 27)
INSERT INTO Comenzi VALUES (102, 725, 14)
INSERT INTO Comenzi VALUES (102, 700, 10)
INSERT INTO Comenzi VALUES (102, 701, 10)
INSERT INTO Comenzi VALUES (102, 701, 15)

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Id_Furnizor**, se foloseşte comanda:

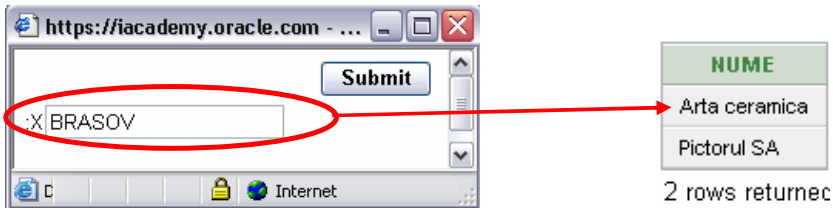
SELECT * FROM Comenzi ORDER BY Id_Furnizor

ID_FURNIZOR	ID_COMPONENTA	CANTITATE
100	600	15
100	510	10
100	510	10
100	500	20
101	605	21
101	700	11
101	705	27
102	701	10
102	701	15
102	700	10
More than 10 rows available. Increase rows selector to view more rows.		

10 rows returned in 0.00 seconds Download

a) Toți furnizorii din oraşul .... (citit de la tastatură) se obțin astfel:

SELECT Nume FROM Furnizori WHERE UPPER(Localitate) = UPPER(:X)



b) Componentele de culoare roşie comandate de la furnizorii din Braşov:

SELECT Nume FROM Componente WHERE UPPER(Culoare) = 'ROSU' AND Id IN (SELECT Id_Componenta FROM Comenzi WHERE Id_Furnitor IN (SELECT Id_Furnizor FROM Furnizori WHERE UPPER(Localitate) = 'BRASOV'))

NUME
Rama tablou 40x20

1 rows returned in C

c) Furnizorul/furnizorii care au oferit componente de culoare „verde" în cantitate maximă:

SELECT a.Nume FROM Furnizori a, Comenzi b, Componente c
WHERE a.Id = b.Id_Furnizor AND b.Id_Componenta = c.Id AND UPPER(Culoare) = 'VERDE' AND (SELECT SUM(b.Cantitate)FROM Furnizori a, Comenzi b, Componente c
WHERE a.Id =b.Id_Furnizor AND b.Id_Componenta = c.Id AND UPPER(Culoare) = 'VERDE') =
(SELECT MAX(SUM(Cantitate)) FROM Furnizori a, Comenzi b, Componente c WHERE a.Id = b.Id_Furnizor AND b.Id_Componenta = c.Id AND UPPER(Culoare) = 'VERDE' GROUP BY a.Id)

NUME
Vopseluri SRL

1 rows returned

SAU

Pentru a afişa numele furnizorului şi suma maximă (cu aliasul SUMA), se poate folosi următoarea comandă:

SELECT DISTINCT a.Nume, SUM(b.Cantitate) Suma FROM Furnizori a, Comenzi b, Componente c
WHERE a.Id = b.Id_Furnizor AND b.Id_Componenta = c.Id AND UPPER(Culoare) = 'VERDE' GROUP BY a.Nume
HAVING SUM(b.Cantitate) =
(SELECT MAX(SUM(Cantitate)) FROM Furnizori a, Comenzi b, Componente c WHERE a.Id = b.Id_Furnizor AND b.Id_Componenta = c.Id AND UPPER(Culoare) = 'VERDE' GROUP BY a.Id)

NUME	SUMA
Vopseluri SRL	25

1 rows returned in 0.01 s

SUBIECTUL 13

Se consideră o bază de date cu următoarele tabele: **CLASE** (**Cod_Clasa, Nume, Sala, Profil, Cod_Diriginte**), **PROFESORI** (**Cod, Nume, Prenume, Specializarea**), **INCADRARI** (**Cod_Profesor, Cod_Clasa, Nr_Ore**) şi **ELEVI** (**Id, Cod_Clasa, Nume, Prenume**).

Cerințe:

a) Afişaţi numele, prenumele şi specializarea pentru profesorii care au ore la clasa „IX B”.

b) Afişaţi numele şi prenumele tuturor colegilor de clasă ai elevei „Enescu Maria”.

c) Afişaţi numărul de elevi şi profilul pentru fiecare clasă.

Se considera că fiecare profesor are o singură specializare, iar un exemplu pentru numele unei clase este „IX B”.

REZOLVARE:

Pentru că tabelul **INCADRARI** face referire la tabelele **CLASE** şi **PROFESORI**, mai întâi se creează acestea din urmă.

Pentru crearea tabelului **CLASE** se scrie comanda:

CREATE TABLE Clase (Cod_Clasa NUMBER(3) PRIMARY KEY, Nume VARCHAR2(10), Sala NUMBER (3), Profil VARCHAR2(20), Cod_Diriginte NUMBER(3))

Pentru “popularea” cu date se scriu, pe rând, comenzile:

INSERT INTO Clase VALUES (1,'IX A', 16, 'Mate-info', 15)
INSERT INTO Clase VALUES (2,'IX B', 11, 'Servicii', 10)
INSERT INTO Clase VALUES (3,'IX C', 14, ' Servicii', 20)
INSERT INTO Clase VALUES (4,'X A', 8, 'Mate-info', 35)
INSERT INTO Clase VALUES (5,'X B', 7, 'Servicii', 25)
INSERT INTO Clase VALUES (6,'X C', 9, 'Servicii', 45)

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_Clasa**, se foloseşte comanda:

SELECT * FROM Clase ORDER BY Cod_Clasa

COD_CLASA	NUME	SALA	PROFIL	COD_DIRIGINTE
1	IX A	16	Mate-info	15
2	IX B	11	Servicii	10
3	IX C	14	Servicii	20
4	X A	8	Mate-info	35
5	X B	7	Servicii	25
6	X C	9	Servicii	45

6 rows returned in 0.01 seconds Download

Pentru crearea tabelului **PROFESORI** se scrie comanda:

CREATE TABLE Profesori (Cod NUMBER(2) PRIMARY KEY, Nume VARCHAR2(20), Prenume VARCHAR2(30), Specializarea VARCHAR2(15))

Pentru “popularea” cu date se scriu, pe rând, comenzile:

INSERT INTO Profesori VALUES (10,'Popescu', 'Vasile', 'Fizica')
INSERT INTO Profesori VALUES (15,'Ionescu', 'Mariana', 'Matematica')
INSERT INTO Profesori VALUES (20,'Tanase', 'Marius', 'Franceza')
INSERT INTO Profesori VALUES (25,'Grigore', 'Ionela', 'Contabilitate')
INSERT INTO Profesori VALUES (35,'Danciu', 'George', 'Informatica')
INSERT INTO Profesori VALUES (45,'Badescu', 'Viorica', 'Chimie')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod**, se foloseşte comanda:

SELECT * FROM Profesori ORDER BY Cod

COD	NUME	PRENUME	SPECIALIZAREA
10	Popescu	Vasile	Fizica
15	Ionescu	Mariana	Matematica
20	Tanase	Marius	Franceza
25	Grigore	Ionela	Contabilitate
35	Danciu	George	Informatica
45	Badescu	Viorica	Chimie

6 rows returned in 0.01 seconds Download

Pentru crearea tabelului **INCDRARI** se scrie comanda:

CREATE TABLE Incadrari (Cod_Profesor NUMBER(2) REFERENCES Profesori(Cod), Cod_clasa NUMBER(3) REFERENCES Clase(Cod_Clasa), Nr_ore NUMBER(2))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Incadrari VALUES (10, 1, 3)
INSERT INTO Incadrari VALUES (15, 1, 4)
INSERT INTO Incadrari VALUES (20, 1, 2)
INSERT INTO Incadrari VALUES (35, 1, 4)
INSERT INTO Incadrari VALUES (45, 1, 2)
INSERT INTO Incadrari VALUES (10, 2, 2)
INSERT INTO Incadrari VALUES (15, 2, 2)
INSERT INTO Incadrari VALUES (25, 2, 2)
INSERT INTO Incadrari VALUES (20, 3, 2)
INSERT INTO Incadrari VALUES (15, 3, 2)
INSERT INTO Incadrari VALUES (25, 3, 2)
INSERT INTO Incadrari VALUES (10, 4, 3)
INSERT INTO Incadrari VALUES (15, 4, 4)
INSERT INTO Incadrari VALUES (35, 4, 4)
INSERT INTO Incadrari VALUES (45, 4, 2)
INSERT INTO Incadrari VALUES (10, 5, 2)
INSERT INTO Incadrari VALUES (15, 5, 3)
INSERT INTO Incadrari VALUES (45, 5, 2)
INSERT INTO Incadrari VALUES (25, 5, 2)
INSERT INTO Incadrari VALUES (20, 6, 2)
INSERT INTO Incadrari VALUES (25, 6, 2)
INSERT INTO Incadrari VALUES (35, 6, 2)

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_Clasa**, se foloseşte comanda:

SELECT * FROM Incadrari ORDER BY Cod_Clasa

COD_PROFESOR	COD_CLASA	NR_ORE
15	1	4
20	1	2
35	1	4
45	1	2
10	1	3
25	2	2
10	2	2
15	2	2
20	3	2
25	3	2

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds Download

COD_CLASA	NUME	SALA	PROFIL	COD_DIRIGINTE
1	IX A	16	Mate-info	15
2	IX B	11	Servicii	10
3	IX C	14	Servicii	20
4	X A	8	Mate-info	35
5	X B	7	Servicii	25
6	X C	9	Servicii	45

6 rows returned in 0.01 seconds Download

Pentru crearea tabelului **ELEVI** se scrie comanda:

CREATE TABLE Elevi (Id NUMBER(4) PRIMARY KEY, Cod_Clasa NUMBER(3) REFERENCES Clase(Cod_Clasa), Nume VARCHAR2(20), Prenume VARCHAR2(30))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Elevi VALUES (911,1,'Balan', 'Vasile')
INSERT INTO Elevi VALUES (912,1,'Barbu', 'Eugen')
INSERT INTO Elevi VALUES (913,1,'Enescu', 'Maria')
INSERT INTO Elevi VALUES (921,2,'Balaban', 'Dorin')
INSERT INTO Elevi VALUES (922,2,'Chituc', 'Loredana')
INSERT INTO Elevi VALUES (923,2,'Tatu', 'Ionela')
INSERT INTO Elevi VALUES (931,3,'Manea', 'George')
INSERT INTO Elevi VALUES (932,3,'Popa', 'Dumitru')
INSERT INTO Elevi VALUES (933,3,'Tenea', 'Maria')
INSERT INTO Elevi VALUES (941,4,'Vladescu', 'Raul')
INSERT INTO Elevi VALUES (951,5,'Manu', 'Vlad')
INSERT INTO Elevi VALUES (961,6,'Tanase', 'George')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Id**, se foloseşte comanda:

SELECT * FROM Elevi ORDER BY Id

ID	COD_CLASA	NUME	PRENUME
911	1	Balan	Vasile
912	1	Barbu	Eugen
913	1	Enescu	Maria
921	2	Balaban	Dorin
922	2	Chituc	Loredana
923	2	Tatu	Ionela
931	3	Manea	George
932	3	Popa	Dumitru
933	3	Tenea	Maria
941	4	Vladescu	Raul

More than 10 rows available. Increase rows selector to view more rows.

10 rows returned in 0.01 seconds Download

a) Numele, prenumele şi specializarea pentru profesorii care au ore la clasa „IX B":

SELECT Nume, Prenume, Specializarea FROM Profesori WHERE Cod IN (SELECT Cod_Profesor FROM Incadrari WHERE Cod_Clasa IN (SELECT Cod_Clasa FROM Clase WHERE UPPER(Nume) = 'IX B'))

NUME	PRENUME	SPECIALIZAREA
Popescu	Vasile	Fizica
Ionescu	Mariana	Matematica
Grigore	Ionela	Contabilitate

3 rows returned in 0.02 seconds Down

b) Numele şi prenumele tuturor colegilor de clasă ai elevei „Enescu Maria":

SELECT Nume, Prenume FROM Elevi WHERE Cod_Clasa IN (SELECT Cod_Clasa FROM Elevi WHERE UPPER(Nume) = 'ENESCU' AND UPPER(Prenume) = 'MARIA')

NUME	PRENUME
Enescu	Maria
Barbu	Eugen
Balan	Vasile

3 rows returned in 0.01

c) Numărul de elevi şi profilul pentru fiecare clasă:

SELECT b.Nume, b.Profil, COUNT(a.Nume) Nr_elevi FROM Elevi a, Clase b WHERE a.Cod_Clasa = b.Cod_Clasa GROUP BY a.Cod_Clasa, b.Nume, b.Profil ORDER BY b.Nume

NUME	PROFIL	NR_ELEVI
IX A	Mate-info	3
IX B	Servicii	3
IX C	Servicii	3
X A	Mate-info	1
X B	Servicii	1
X C	Servicii	1

6 rows returned in 0.00 seconds

SUBIECTUL 14

Se consideră baza de date în care se ține evidența accidentelor care au avut loc în România cu următoarele tabele: **PERSOANE** (**Cod, Nume, Localitate**), **MASINI** (**Id_Masina, Model, An_Fabricație, Cod_Proprietar**) şi **ACCIDENTE** (**Data, Cod_Sofer, Daune, Loc_Accident, Id_Masina**). Ştiind că o persoană poate avea mai multe maşini, iar o maşină poate fi condusă şi de persoane diferite de proprietar, să se rezolve următoarele cerințe:

a) Afişați Id-ul şi modelul maşinilor implicate în accidente şi care au fost conduse de proprietar;

b) Determinați suma totală a daunelor produse în accidente în care au fost implicați şoferi din „Braşov";

c) Afişați oraşul/oraşele în care au avut loc cele mai multe accidente.

REZOLVARE:

Pentru că tabelul **ACCIDENTE** face referire la tabelele **PERSOANE** şi **MASINI**, mai întâi se creează acestea din urmă.

Pentru crearea tabelului **PERSOANE** se scrie comanda:

CREATE TABLE Persoane (Cod NUMBER(3) PRIMARY KEY, Nume VARCHAR2(30), Localitate VARCHAR2(20))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Persoane VALUES (100,'Dragan Adrian', 'Craiova')
INSERT INTO Persoane VALUES (101,'Marin Dana', 'Pitesti')
INSERT INTO Persoane VALUES (102,'Nicolaescu George', 'Craiova')
INSERT INTO Persoane VALUES (103,'Balan Radu', 'Brasov')
INSERT INTO Persoane VALUES (104,'Barbu Maria', 'Craiova')
INSERT INTO Persoane VALUES (105,'Tatu Vasile', 'Brasov')

Pentru a afişa conținutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod**, se foloseşte comanda:

SELECT * FROM Persoane ORDER BY Cod

COD	NUME	LOCALITATE
100	Dragan Adrian	Craiova
101	Marin Dana	Pitesti
102	Nicolaescu George	Craiova
103	Balan Radu	Brasov
104	Barbu Maria	Craiova
105	Tatu Vasile	Brasov

6 rows returned in 0.01 seconds Down

Pentru crearea tabelului **MASINI** se scrie comanda:

CREATE TABLE Masini (Id_Masina NUMBER(3) PRIMARY KEY, Model VARCHAR2(30), An_Fabricaţie NUMBER(4), Cod_Proprietar NUMBER(3))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Masini VALUES (11,'Dacia Logan', 2007,103)
INSERT INTO Masini VALUES (12,'Opel Astra', 2006,105)
INSERT INTO Masini VALUES (13,'Toyota Advences', 2008,100)
INSERT INTO Masini VALUES (14,'Opel Vectra', 2005,101)
INSERT INTO Masini VALUES (15,'Dacia Logan', 2006,102)
INSERT INTO Masini VALUES (16,'Opel Astra', 2008,104)

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod**, se foloseşte comanda:

SELECT * FROM Masini ORDER BY Id_Masina

ID_MASINA	MODEL	AN_FABRICAŢIE	COD_PROPRIETAR
11	Dacia Logan	2007	103
12	Opel Astra	2006	105
13	Toyota Advences	2008	100
14	Opel Vectra	2005	101
15	Dacia Logan	2006	102
16	Opel Astra	2008	104

6 rows returned in 0.00 seconds Download

(**Data, Cod_Sofer, Daune, Loc_Accident, Id_Masina**).

Pentru crearea tabelului **ACCIDENTE** se consideră că **Cod_Acc** reprezintă identificatorul unic al unui accident şi se scrie comanda:

CREATE TABLE Accidente (Cod_Acc NUMBER(3) PRIMARY KEY, Data DATE, Cod_Sofer NUMBER(3) REFERENCES Persoane(Cod), Daune NUMBER (10,2), Loc VARCHAR2(30), Id_Masina NUMBER(3) REFERENCES Masini(Id_Masina))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Accidente VALUES (1,'12-dec-2008',103, 1200, 'Sibiu',12)
INSERT INTO Accidente VALUES (2,'19-dec-2008',100, 1500, 'Bucuresti',13)
INSERT INTO Accidente VALUES (3,'22-dec-2008',101, 850, 'Sibiu',14)
INSERT INTO Accidente VALUES (4,'24-dec-2008',102, 1800, 'Sibiu',11)
INSERT INTO Accidente VALUES (5,'28-dec-2008',105, 800, 'Craiova',12)
INSERT INTO Accidente VALUES (6,'30-dec-2008',104, 950, 'Pitesti',16)

Pentru a afişa conținutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_Acc**, se foloseşte comanda:

SELECT * FROM Accidente ORDER BY Cod_Acc

COD_ACC	DATA	COD_SOFER	DAUNE	LOC	ID_MASINA
1	12-DEC-08	103	1200	Sibiu	12
2	19-DEC-08	100	1500	Bucuresti	13
3	22-DEC-08	101	850	Sibiu	14
4	24-DEC-08	102	1800	Sibiu	11
5	28-DEC-08	105	800	Craiova	12
6	30-DEC-08	104	950	Pitesti	16

6 rows returned in 0.00 seconds Download

a) Id-ul şi modelul maşinilor implicate în accidente şi care au fost conduse de proprietar:

SELECT Id_Masina, Model FROM Masini a WHERE Id_Masina IN
(SELECT Id_Masina FROM Accidente WHERE Cod_Sofer = a.Cod_Proprietar)
ORDER BY Id_Masina

ID_MASINA	MODEL
12	Opel Astra
13	Toyota Advences
14	Opel Vectra
16	Opel Astra

4 rows returned in 0.01 seconds

ID_MASINA	MODEL	AN_FABRICAȚIE	COD_PROPRIETAR
11	Dacia Logan	2007	103
12	Opel Astra	2006	105
13	Toyota Advences	2008	100
14	Opel Vectra	2005	101
15	Dacia Logan	2006	102
16	Opel Astra	2008	104

6 rows returned in 0.00 seconds Download

b) Suma totală a daunelor produse în accidente în care au fost implicați şoferi din „Braşov”:

SELECT SUM(Daune) FROM Accidente WHERE Cod_Sofer IN
(SELECT Cod FROM Persoane WHERE UPPER(Localitate) = 'BRASOV')

TOTAL_DAUNE
2000

1 rows returned in

COD	NUME	LOCALITATE
100	Dragan Adrian	Craiova
101	Marin Dana	Pitesti
102	Nicolaescu George	Craiova
103	Balan Radu	Brasov
104	Barbu Maria	Craiova
105	Tatu Vasile	Brasov

6 rows returned in 0.01 seconds Down

c) Oraşul (oraşele) în care au avut loc cele mai multe accidente:

SELECT Loc, COUNT(*) Nr_Accidente FROM Accidente GROUP BY Loc HAVING
COUNT(*) = (SELECT MAX(COUNT(*)) FROM Accidente GROUP BY Loc)

LOC	NR_ACCIDENTE
Sibiu	3

1 rows returned in 0.01 sec

SUBIECTUL 15

Se consideră o bază de date cu 2 tabele: **USERS** (**User_Id, Name**), **GROUPS** (**Group_Id, Title, Category**) şi **POSTS** (**Post_Id, User_Id, Group_Id, Post_Text, Date_Created**).

Cerințe:

a) Afişaţi grupurile („group_id” şi „title”) pentru care numele categoriei are între 6 şi 9 caractere;

b) Afişaţi mesajele („user_id” şi „post_text”) şi numărul de zile care a trecut de la crearea lor;

c) Pentru mesajele din categoria „Sport” create în „13 aprilie” afişaţi mesajul: „text vid” (dacă textul are 0 caractere), „mesaj scurt” (dacă textul are între 1 şi 10 caractere) sau „mesaj lung” (dacă textul are mai mult de 10 caractere).

REZOLVARE:

Pentru crearea tabelului **USERS** se scrie comanda:

CREATE TABLE Users (User_Id NUMBER(3) PRIMARY KEY, Name VARCHAR2(20))

Pentru “popularea” cu date se scriu, pe rând, comenzile:

INSERT INTO Users VALUES (100,'Dragan Adrian')
INSERT INTO Users VALUES (101,'Marin Daniela')
INSERT INTO Users VALUES (112,'Voinea George')
INSERT INTO Users VALUES (116,'Nicolaescu George')
INSERT INTO Users VALUES (203, 'Balan Radu')
INSERT INTO Users VALUES (204,'Barbu Maria')
INSERT INTO Users VALUES (215,'Tatu Vasile')
INSERT INTO Users VALUES (217,'Ionescu Ramona')

Pentru a afişa conţinutul tabelului **Users**, în ordine crescătoare a valorilor din câmpul **Name**, se foloseşte comanda:

SELECT * FROM Users ORDER BY Name

USER_ID	NAME
203	Balan Radu
204	Barbu Maria
100	Dragan Adrian
217	Ionescu Ramona
101	Marin Daniela
116	Nicolaescu George
215	Tatu Vasile
112	Voinea George

8 rows returned in 0.01 second

Pentru crearea tabelului **GROUPS** se scrie comanda:

CREATE TABLE Groups (Group_Id NUMBER(3) PRIMARY KEY, Title VARCHAR2(20), Category VARCHAR2(20))

Pentru “popularea” cu date se scriu, pe rând, comenzile:

INSERT INTO Groups VALUES (10,'Profesori info', 'Scoala')
INSERT INTO Groups VALUES (11,'Profesori mate', 'Scoala')
INSERT INTO Groups VALUES (20,'Campionat fotbal', 'Sport')
INSERT INTO Groups VALUES (21,'Tenis de camp', 'Sport')

Pentru a afişa conţinutul tabelului **Groups**, în ordine crescătoare a valorilor din câmpul **Title**, se foloseşte comanda:

SELECT * FROM Groups ORDER BY Title

GROUP_ID	TITLE	CATEGORY
20	Campionat fotbal	Sport
10	Profesori info	Scoala
11	Profesori mate	Scoala
21	Tenis de camp	Sport

4 rows returned in 0.01 seconds Downloa

Pentru crearea tabelului **POSTS** se scrie comanda:

CREATE TABLE Posts (Post_Id NUMBER(5) PRIMARY KEY, User_Id NUMBER(3) REFERENCES Users(User_Id), Group_Id NUMBER(3) REFERENCES Groups(Group_Id), Post_Text VARCHAR2(50), Date_Created DATE)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Posts VALUES (1000, 100, 10, 'Cercul de info este pe 3 aprilie', '23-mar-2009')
INSERT INTO Posts VALUES (1001, 100, 10, 'Cercul se tine la Economic', '25-mar-2009')
INSERT INTO Posts VALUES (1002, 101, 10, 'Inscrierile pt. CIA se fac pana pe 2 apr.', '13-mar-2009')
INSERT INTO Posts VALUES (1100, 112, 11, 'Concursul PI incepe la ora 10', '28-feb-2009')
INSERT INTO Posts VALUES (1101, 116, 11, 'Merg si eu la Dragasani.', '13-apr-2009')
INSERT INTO Posts VALUES (1102, 116, 11, 'Am obtinut premiul II la Olanesti', '24-mar-2009')
INSERT INTO Posts VALUES (2000, 203, 20, '', '13-apr-2009')
INSERT INTO Posts VALUES (2001, 203, 20, 'E meci?', '13-apr-2008')
INSERT INTO Posts VALUES (2101, 203, 21, 'Rezultatele de ieri sunt pe SPORT.RO', '13-apr-2009')

Pentru a afişa conţinutul tabelului **Posts**, în ordine crescătoare a valorilor din câmpul **Title**, se foloseşte comanda:

SELECT * FROM Posts ORDER BY Post_Id

POST_ID	USER_ID	GROUP_ID	POST_TEXT	DATE_CREATED
1000	100	10	Cercul de info este pe 3 aprilie	23-MAR-09
1001	100	10	Cercul se tine la Economic	25-MAR-09
1002	101	10	Inscrierile pt. CIA se fac pana pe 2 apr.	13-MAR-09
1100	112	11	Concursul PI incepe la ora 10	28-FEB-09
1101	116	11	Merg si eu la Dragasani.	13-APR-09
1102	116	11	Am obtinut premiul II la Olanesti	24-MAR-09
2000	203	20	-	13-APR-09
2001	203	20	E meci?	13-APR-08
2101	203	21	Rezultatele de ieri sunt pe SPORT.RO	13-APR-09

9 rows returned in 0.01 seconds Download

a) Grupurile („Group_id" şi „Title") pentru care numele categoriei are între 6 şi 9 caractere:

SELECT Group_Id, Title FROM Groups WHERE (LENGTH(Category)>=6) AND (LENGTH(Category)<=9)

GROUP_ID	TITLE
10	Profesori info
11	Profesori mate

2 rows returned in 0.01 seconds

GROUP_ID	TITLE	CATEGORY
20	Campionat fotbal	Sport
10	Profesori info	Scoala
11	Profesori mate	Scoala
21	Tenis de camp	Sport

4 rows returned in 0.01 seconds Downloa

b) mesajele („user_id” şi „post_text”) şi numărul de zile care a trecut de la crearea lor:

SELECT User_Id, Post_Text, TRUNC(CURRENT_DATE - Date_Created) Nr_zile FROM Posts ORDER BY User_Id

USER_ID	POST_TEXT	NR_ZILE
100	Cercul se tine la Economic	0
100	Cercul de info este pe 3 aprilie	1
101	Inscrierile pt. CIA se fac pana pe 2 apr.	11
112	Concursul PI incepe la ora 10	24
116	Merg si eu la Dragasani.	-19
116	Am obtinut premiul II la Olanesti	0
203	E meci?	345
203	Rezultatele de ieri sunt pe SPORT.RO	-19
203	-	-19

9 rows returned in 0.01 seconds Download

S-a folosit funcţia **TRUNC** pentru că **CURRENT_DATE** întoarce data curentă cu ore şi minute. Fără această funcţie, s-ar obţine numărul de zile ca număr real:

24.936388888888888888888888888888888889

S-a considerat data curentă:

CURRENT_DATE
24-MAR-09

1 rows returned in 0.01 seconds

c) Pentru mesajele din categoria „Sport” create în „13 aprilie” afişaţi mesajul: „text vid” (dacă textul are 0 caractere), „mesaj scurt” (dacă textul are între 1 şi 10 caractere) sau „mesaj lung” (dacă textul are mai mult de 10 caractere):

SELECT User_Id, Post_Text "Mesaj vid" FROM Posts WHERE (Post_Text IS NULL) AND (LOWER(SUBSTR(TO_CHAR(Date_Created, 'DD-MON-YYYY'),1,6)) = '13-apr') AND Group_Id IN
(SELECT Group_Id FROM Groups WHERE LOWER(Category) = 'sport')

SELECT User_Id, Post_Text "Mesaj scurt" FROM Posts WHERE (LENGTH(Post_Text) >=1) AND (LENGTH(Post_Text) <=10) AND (LOWER(SUBSTR(TO_CHAR(Date_Created, 'DD-MON-YYYY'),1,6))= '13-apr') AND Group_Id IN
(SELECT Group_Id FROM Groups WHERE LOWER(Category) = 'sport')

SELECT User_Id, Post_Text "Mesaj lung" FROM Posts WHERE (LENGTH(Post_Text) >=10) AND (LOWER(SUBSTR(TO_CHAR(Date_Created, 'DD-MON-YYYY'),1,6))= '13-apr') AND Group_Id IN
(SELECT Group_Id FROM Groups WHERE LOWER(Category) = 'sport')

USER_ID	Mesaj vid
203	-

1 rows returned in 0.00 se

USER_ID	Mesaj scurt
203	E meci?

1 rows returned in 0.01 secc

USER_ID	Mesaj lung
203	Rezultatele de ieri sunt pe SPORT.RO

1 rows returned in 0.01 seconds Downloa

S-a folosit expresia:

- **SUBSTR(TO_CHAR(Date_Created, 'DD-MON-YYYY'),1,6)** – extrage ziua şi luna (în forma **13-apr**) din şirul de caractere corespunzător datei postării mesajului;

Observaţie:

Se poate crea un nou câmp (coloană) în tabelul **POSTS**, pe care îl putem numi **Type_Post** în care se va scrie: „text vid” (dacă textul are 0 caractere), „mesaj scurt” (dacă textul are între 1 şi 10 caractere) sau „mesaj lung” (dacă textul are mai mult de 10 caractere).

Pentru aceasta:

- se modifică structura tabelului (se adaugă câmpul **Type_Post**) astfel:

ALTER TABLE Posts ADD Type_Post VARCHAR2(12)

- se modifică conținutul tabelului (în câmpul **Type_Post** se scrie „text vid”, „mesaj scurt” sau „mesaj lung”) astfel:

UPDATE Posts SET Type_Post = 'Mesaj vid' WHERE Post_Text IS NULL

UPDATE Posts SET Type_Post = 'Mesaj scurt' WHERE (LENGTH(Post_Text) >=1) AND (LENGTH(Post_Text) <=10)

UPDATE Posts SET Type_Post = 'Mesaj lung' WHERE LENGTH(Post_Text) >=10

Se obține tabelul:

POST_ID	USER_ID	GROUP_ID	POST_TEXT	DATE_CREATED	TYPE_POST
1001	100	10	Cercul se tine la Economic	25-MAR-09	Mesaj lung
1002	101	10	Inscrierile pt. CIA se fac pana pe 2 apr.	13-MAR-09	Mesaj lung
1100	112	11	Concursul PI incepe la ora 10	28-FEB-09	Mesaj lung
2001	203	20	E meci?	13-APR-08	Mesaj scurt
1000	100	10	Cercul de info este pe 3 aprilie	23-MAR-09	Mesaj lung
1101	116	11	Merg si eu la Dragasani.	13-APR-09	Mesaj lung
1102	116	11	Am obtinut premiul II la Olanesti	24-MAR-09	Mesaj lung
2000	203	20	-	13-APR-09	Mesaj vid
2101	203	21	Rezultatele de ieri sunt pe SPORT.RO	13-APR-09	Mesaj lung

9 rows returned in 0.00 seconds Download

- se afişează doar mesajele postate în „13 aprilie”, din categoria „Sport”:

SELECT User_Id, Post_Text, Type_Post FROM Posts WHERE (LOWER(SUBSTR(TO_CHAR(Date_Created, 'DD-MON-YYYY'),1,6)) = '13-apr') AND Group_Id IN (SELECT Group_Id FROM Groups WHERE LOWER(Category) = 'sport')

USER_ID	POST_TEXT	TYPE_POST
203	E meci?	Mesaj scurt
203	-	Mesaj vid
203	Rezultatele de ieri sunt pe SPORT.RO	Mesaj lung

3 rows returned in 0.01 seconds Download

SUBIECTUL 16

Se consideră o bază de date cu 2 tabele: **REVISTE** (**Nume**, C(30) – denumirea revistei – identifică unic o revistă; **Domeniu**, C(10) – domeniul publicației; **Editura**, C(20) – numele editurii care o tipăreşte; **Pret,** N(5.2) – prețul lunar al abonamentului în lei) şi **ABONAȚI** (**Cnp**, C(13) – identifică unic o persoană (abonat); **Nume**, C(30) – numele şi prenumele abonatului).

Ştiind că o persoană poate fi abonată la mai multe reviste pe un număr de luni, să se rezolve următoarele cerințe:

a) crearea tabelelor REVISTE, ABONAȚI şi popularea cu date;

b) crearea unui nou tabel ABONAMENTE care să conțină informațiile: persoana abonată, revista la care s-a făcut abonamentul, durata abonamentului (în luni), şi data de început a abonamentului;

c) afişarea revistelor cu cele mai multe abonamente în luna curentă;

d) afişarea abonamentelor expirate.

REZOLVARE:

a) Pentru crearea tabelului **REVISTE** se scrie comanda:

CREATE TABLE Reviste (Nume VARCHAR2(30)PRIMARY KEY, Domeniu VARCHAR2(10), Editura VARCHAR2(20), Pret NUMBER(5,2))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Reviste VALUES ('Sabrina's secrets', 'Vacanta', 'Univers', 6.9)
INSERT INTO Reviste VALUES ('Atestat', 'Educatie', 'ALL', 11.75)
INSERT INTO Reviste VALUES ('Popcorn', 'Diverse', 'Univers', 4)
INSERT INTO Reviste VALUES ('Bravo', 'Diverse', 'Flacara', 3.5)
INSERT INTO Reviste VALUES ('Rebus Flacara', 'Vacanta', 'Flacara', 5.5)
INSERT INTO Reviste VALUES ('PC Word', 'Educatie', 'ALL', 8.5)

Pentru a afişa conţinutul tabelului **Reviste**, în ordine crescătoare a valorilor din câmpul **Nume**, se foloseşte comanda:

SELECT * FROM Reviste ORDER BY Nume

NUME	DOMENIU	EDITURA	PRET
Atestat	Educatie	ALL	11.75
Bravo	Diverse	Flacara	3.5
PC Word	Educatie	ALL	8.5
Popcorn	Diverse	Univers	4
Rebus Flacara	Vacanta	Flacara	5.5
Sabrina's secrets	Vacanta	Univers	6.9

6 rows returned in 0.01 seconds Download

Pentru crearea tabelului **ABONATI** se scrie comanda:

CREATE TABLE Abonati (Cnp VARCHAR2(13)PRIMARY KEY, Nume VARCHAR2(30))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Abonati VALUES (100,'Ionescu Gigel')
INSERT INTO Abonati VALUES (101,'Ionescu Maria')
INSERT INTO Abonati VALUES (102,'Popescu Daniel')
INSERT INTO Abonati VALUES (103,'Popa Maria')
INSERT INTO Abonati VALUES (104,'Radu Roxana')
INSERT INTO Abonati VALUES (105,'Radulescu Ioana')
INSERT INTO Abonati VALUES (106,'Toma George')
INSERT INTO Abonati VALUES (107,'Voinea Ana')

Pentru a afişa conţinutul tabelului **Abonati**, în ordine crescătoare a valorilor din câmpul **Nume**, se foloseşte comanda:

SELECT * FROM Abonati ORDER BY Nume

CNP	NUME
100	Ionescu Gigel
101	Ionescu Maria
103	Popa Maria
102	Popescu Daniel
104	Radu Roxana
105	Radulescu Ioana
106	Toma George
107	Voinea Ana

8 rows returned in 0.01 s

b) Crearea tabelului **ABONAMENTE** care să conțină informațiile: persoana abonată, revista la care s-a făcut abonamentul, durata abonamentului (în luni), şi data de început a abonamentului:

Pentru crearea tabelului **ABONAMENTE** se scrie comanda:

CREATE TABLE Abonamente (Cnp VARCHAR2(13) REFERENCES Abonati(Cnp), Nume_revista VARCHAR2(30) REFERENCES Reviste(Nume), Durata NUMBER (2), Data_Inceput DATE)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Abonamente VALUES (100,'Atestat', 5, '10-jan-2009')
INSERT INTO Abonamente VALUES (101,'Atestat', 4, '11-feb-2009')
INSERT INTO Abonamente VALUES (105,'Bravo', 3, '15-dec-2008')
INSERT INTO Abonamente VALUES (107,'Bravo', 3, '25-mar-2009')
INSERT INTO Abonamente VALUES (103,'PC Word', 10, '1-dec-2008')
INSERT INTO Abonamente VALUES (104,'PC Word', 6, '1-apr-2009')
INSERT INTO Abonamente VALUES (102,'PC Word', 10, '5-feb-2009')
INSERT INTO Abonamente VALUES (100,'Bravo', 4, '15-jan-2009')
INSERT INTO Abonamente VALUES (101,'PC Word', 10, '21-dec-2008')

Pentru a afişa conținutul tabelului **Abonamente**, în ordine crescătoare a valorilor din câmpul **Nume_revista**, se foloseşte comanda:

SELECT * FROM Abonamente ORDER BY Nume_revista

CNP	NUME_REVISTA	DURATA	DATA_INCEPUT
100	Atestat	5	10-JAN-09
101	Atestat	4	11-FEB-09
107	Bravo	3	25-MAR-09
100	Bravo	4	15-JAN-09
105	Bravo	3	15-DEC-08
103	PC Word	10	01-DEC-08
101	PC Word	10	21-DEC-08
104	PC Word	6	01-APR-09
102	PC Word	10	05-FEB-09

9 rows returned in 0.01 seconds Download

c) afişarea revistelor cu cele mai multe abonamente în luna curentă:

Pentru afişarea revistelor cu abonamente în luna curentă:

SELECT DISTINCT Nume_revista FROM Abonamente WHERE ADD_MONTHS(Data_Inceput, Durata) > CURRENT_DATE

NUME_REVISTA
Bravo
PC Word
Atestat

3 rows returned in 0

SELECT Nume_revista FROM Abonamente WHERE ADD_MONTHS(Data_Inceput, Durata) > CURRENT_DATE GROUP BY Nume_Revista HAVING COUNT (*) = (SELECT MAX(COUNT(*)) FROM Abonamente WHERE ADD_MONTHS(Data_Inceput, Durata) > CURRENT_DATE GROUP BY Nume_Revista)

NUME_REVISTA
PC Word

1 rows returned in 0

S-a considerat data curentă:

CURRENT_DATE
24-MAR-09

1 rows returned in 0.01 seconds

d) Afişarea abonamentelor expirate.

SELECT Cnp, Nume_revista, Durata, Data_Inceput FROM Abonamente WHERE ADD_MONTHS(Data_Inceput, Durata) <= CURRENT_DATE

CNP	NUME_REVISTA	DURATA	DATA_INCEPUT
105	Bravo	3	15-DEC-08

1 rows returned in 0.01 seconds Download

CURRENT_DATE
24-MAR-09

1 rows returned in 0.01 seconds

S-a considerat data curentă:

SUBIECTUL 17

Se consideră o bază de date 2 tabele privind produsele vândute în mai multe magazine: **MAGAZINE** (**Cod_Mag**, N(3) – identifică unic un magazin; **Denumire**, C(20) – numele magazinului; **Adresa**, C(20) – adresa clădirii în care se găseşte magazinul) şi **PRODUSE** (**Cod**, N(3) – codul produsului; **Denumire**, C(20) – denumirea produsului; **Cantitate**, N(5) – cu semnificaţia "număr bucăţi", reprezentând stocul curent; **Pret**, N(7) – preţul pe bucată).

Se cere:

a) crearea tabelelor MAGAZINE, PRODUSE şi popularea cu date;

b) crearea unui nou tabel VÂNZĂRI care să conţină informaţiile: produsul vândut, magazinul la care s-a făcut vânzarea, cantitatea vândută (în bucăţi) şi data vânzării, (datele din tabele se consideră pertinente şi conforme cu o realitate posibilă)

c) magazinele cu cele mai slabe vânzări în perioada "3.04.2009 - 5.05.2009";

d) produsele cu stoc epuizat (stocul fiecărui produs va fi actualizat conform vânzărilor făcute).

REZOLVARE:

a) Pentru crearea tabelului **MAGAZINE** se scrie comanda:

CREATE TABLE Magazine (Cod_Mag NUMBER(3) PRIMARY KEY, Denumire VARCHAR2(20), Adresa VARCHAR2(20))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Magazine VALUES (100,'Cozia', 'Calea lui traian')
INSERT INTO Magazine VALUES (101,'Anabella', 'Calea lui traian')
INSERT INTO Magazine VALUES (102,'Super market', 'Aleea trandafirilor')

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_Mag**, se foloseşte comanda:

SELECT * FROM Magazine ORDER BY Cod_Mag

COD_MAG	DENUMIRE	ADRESA
100	Cozia	Calea lui traian
101	Anabella	Calea lui Traian
102	Super market	Aleea Trandafirilor

3 rows returned in 0.01 seconds Download

Pentru crearea tabelului **PRODUSE** se scrie comanda:

CREATE TABLE Produse (Cod NUMBER(3) PRIMARY KEY, Denumire VARCHAR2(20), Cantitate NUMBER(5), Pret NUMBER(7,2))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Produse VALUES (10,'Paine', 95, 2.5)
INSERT INTO Produse VALUES (11,'Covrig', 200, 0.5)

INSERT INTO Produse VALUES (12,'Corn', 88, 1.55)
INSERT INTO Produse VALUES (13,'Napolitane', 98, 2.5)

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_Mag**, se foloseşte comanda:

SELECT * FROM Produse ORDER BY Cod

COD	DENUMIRE	CANTITATE	PRET
10	Paine	95	2.5
11	Covrig	200	.5
12	Corn	88	1.55
13	Napolitane	98	2.5

4 rows returned in 0.01 seconds Downlo

b) Pentru crearea tabelului **VANZARI** se scrie comanda:

CREATE TABLE Vanzari (Cod_prod NUMBER(3) REFERENCES Produse(Cod), Cod_Mag NUMBER(3) REFERENCES Magazine(Cod_Mag), Cantitate NUMBER(5), Data_vanzarii DATE)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Vanzari VALUES (10, 102, 10, '12-dec-2008')
INSERT INTO Vanzari VALUES (10, 101, 20, '12-dec-2008')
INSERT INTO Vanzari VALUES (10, 101, 10, '11-dec-2008')
…

Pentru a afişa conţinutul tabelului, în ordine crescătoare a valorilor din câmpul **Cod_prod**, se foloseşte comanda:

SELECT * FROM Vanzari ORDER BY Cod_prod

COD_PROD	COD_MAG	CANTITATE	DATA_VANZARII
10	102	10	12-DEC-08
10	101	20	12-DEC-08
10	101	10	11-DEC-08
10	100	10	10-DEC-08
11	101	50	17-NOV-08
11	102	50	27-NOV-08
11	100	20	17-DEC-08
12	100	20	17-DEC-08
12	101	30	07-DEC-08
12	102	20	27-NOV-08

10 rows returned in 0.00 seconds Download

c) Pentru a afla care sunt magazinele cu cele mai slabe vânzări într-o perioadă dată de timp, putem proceda astfel:

- Pentru calculul cantităţii totale de produse vândute pentru fiecare magazin, putem utiliza funcţia de grup **SUM(expresie)** – calculează suma valorilor corespunzătoare expresiei (care poate fi numele unei coloane din tabel):

SELECT Cod_Mag, SUM(Cantitate) as Suma FROM Vanzari GROUP BY Cod_Mag

COD_MAG	SUMA
100	50
102	80
101	110

3 rows returned in 0.00

• Pentru calculul minimului cantităților de produse vândute pe magazine, putem utiliza funcția de grup **MIN(expresie)** – calculează minimul valorilor corespunzătoare expresiei (care poate fi numele unei coloane din tabel), în cazul nostru minimul cantității de produse vândute de magazine:

SELECT MIN(SUM(Cantitate)) as Suma_min FROM Vanzari GROUP BY Cod_Mag

SUMA_MIN
50

1 rows returned

• Pentru a afla care sunt magazinele cu cele mai slabe vânzări (cantitatea totală de produse vândute este egală cu minimul), putem proceda astfel:

SELECT Cod_Mag, SUM(Cantitate) as Suma FROM Vanzari GROUP BY Cod_Mag
HAVING SUM(Cantitate) =
(SELECT MIN(SUM(Cantitate)) as Suma_min FROM Vanzari GROUP BY Cod_Mag)

COD_MAG	SUMA
100	50

1 rows returned in 0.01

• Pentru a afla care sunt magazinele cu cele mai slabe vânzări, într-o perioadă dată de timp (ex: "3.04.2009 - 5.05.2009"), putem proceda astfel:

(***)
SELECT Cod_Mag, SUM(Cantitate) as Suma FROM Vanzari WHERE
(data_vanzarii > '3-apr-2009') AND (data_vanzarii < '5-may-2009') GROUP BY Cod_Mag
HAVING SUM(Cantitate)=
(SELECT MIN(SUM(Cantitate)) as Suma_min FROM Vanzari WHERE
(data_vanzarii > '3-apr-2009') AND (data_vanzarii < '5-may-2009') GROUP BY Cod_Mag)

Pentru că nu există linii în tabel care să corespundă cerințelor, se va afișa „no data found" (nu s-au găsit date).

Pentru a avea linii care să corespundă cerințelor, se vor adăuga următoarele date în tabel:

INSERT INTO Vanzari VALUES (10, 101, 10, '12-apr-2009')
INSERT INTO Vanzari VALUES (11, 102, 100, '17-apr-2009')

Vânzările făcute de magazine în perioada "3.04.2009 - 5.05.2009":

SELECT * FROM Vanzari WHERE
(data_vanzarii > '3-apr-2009') AND (data_vanzarii < '5-may-2009')

COD_PROD	COD_MAG	CANTITATE	DATA_VANZARII
10	101	10	12-APR-09
11	102	10	17-APR-09

2 rows returned in 0.01 seconds Download

Magazinele cu cele mai slabe vânzări în perioada "3.04.2009 - 5.05.2009", conform comenzii de mai sus (***):

COD_MAG	SUMA
101	10

1 rows returned in 0.00

d) Pentru găsirea produsele cu stoc epuizat (stocul fiecărui produs va fi actualizat conform vânzărilor făcute) presupune:

- **actualizarea stocului** fiecărui produs (cantitatea din fiecare produs va fi egală cu cantitatea din stoc minus suma vânzărilor):

```
UPDATE Produse a SET a.Cantitate =
(SELECT b.Cantitate - (SELECT SUM(Cantitate) FROM Vanzari
WHERE Cod_prod=b.Cod) FROM Produse b where b.cod=a.cod)
```

```
ORA-01407: cannot update ("RO_CE1_SQL01_T01"."PRODUSE"."CANTITATE") to NULL
```

Obs: Nu se pot face actualizări cu valori nule!! În acest caz, vom face actualizarea numai a acelor produse care au fost vândute, adică suma cantităților vândute este **diferită de zero.**

```
UPDATE Produse a SET a.Cantitate =
(SELECT b.Cantitate - (SELECT SUM(Cantitate) FROM Vanzari
WHERE Cod_prod=b.Cod) FROM Produse b where b.cod=a.cod)
WHERE Cod IN
(SELECT Cod FROM Produse WHERE
(SELECT SUM(Cantitate) FROM Vanzari WHERE Cod_prod=Cod) <>0)
```

După actualizarea datelor, tabelul PRODUSE are următorul conținut:

COD	DENUMIRE	CANTITATE	PRET
10	Paine	35	2.5
11	Covrig	70	.5
12	Corn	18	1.55
13	Napolitane	98	2.5

4 rows returned in 0.00 seconds Downlo

- **afişarea produselor** cu stoc epuizat (pe „zero”):

```
SELECT Cod, Denumire, Cantitate FROM Produse WHERE Cantitate=0
```

Se va afişa mesajul: „NO DATA FOUND” pentru că, în cazul nostru, nu avem produse cu stocul zero.

SUBIECTUL 18

Se consideră o bază de date cu 2 tabele: **ORAR** (**Cod_Avion**, C(10) – identifică unic un avion; **Zile**, C(7) – orarul săptămânal al unui avion (un şir de exact 7 caractere cifre binare cu semnificația că avionul circulă -“1”, sau nu circulă -“0” în a *k*-a zi, unde *k* este poziția cifrei în câmpul ZILE de la stânga la dreapta; de exemplu “0011000” semnifică faptul că avionul circulă numai miercuri şi joi); **Pilot**, C(12) – codul pilotului avionului respectiv (un pilot poate pilota mai multe avioane într-o săptămână); **Ruta**, C(20) – destinația avionului) şi **PILOTI** (**Cod_Pilot**, C(12) – identifică unic un pilot; **Nume**, C(25) – numele şi prenumele pilotului).

Se cere:

a) crearea tabelelor şi popularea cu date;

b) afişarea orarului săptămânal al avioanelor (fiecare zi a săptămânii cu avioanele care circulă în ziua respectivă);

c) numele pilotului cu cele mai multe zboruri săptămânale;
d) codul şi destinaţia avioanelor pe care le pilotează "Badea Viorel" săptămânal.

REZOLVARE:

a) Pentru crearea tabelului **PILOTI** se scrie comanda:

CREATE TABLE Piloti (Cod_pilot VARCHAR2(12) PRIMARY KEY, Nume VARCHAR2(25))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Piloti VALUES ('P01', 'Ionescu George')
INSERT INTO Piloti VALUES ('P02', 'Popescu Radu')
INSERT INTO Piloti VALUES ('P03', 'Dragan Mirabela')
INSERT INTO Piloti VALUES ('P04', 'Georgescu Victor')
INSERT INTO Piloti VALUES ('P05', 'Tatu Gabriela')
INSERT INTO Piloti VALUES ('P06', 'Badea Viorel')
```

Pentru a afişa conţinutul tabelului, se foloseşte comanda

SELECT * FROM Piloti ORDER BY Cod_pilot

COD_PILOT	NUME
P01	Ionescu George
P02	Popescu Radu
P03	Dragan Mirabela
P04	Georgescu Victor
P05	Tatu Gabriela
P06	Badea Viorel

6 rows returned in 0.00 seconds

Pentru crearea tabelului **ORAR** se scrie comanda:

CREATE TABLE Orar (Cod_avion VARCHAR2(10) PRIMARY KEY, Zile CHAR(7), Pilot VARCHAR2(12) REFERENCES Piloti(Cod_Pilot), Ruta VARCHAR2(20))

(şir de exact 7 caractere - de lungime fixă)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Orar VALUES ('A001', '1010100', 'P01', 'Cluj')
INSERT INTO Orar VALUES ('A002', '1001000', 'P02', 'Paris')
INSERT INTO Orar VALUES ('A003', '0100100', 'P03', 'Sibiu')
INSERT INTO Orar VALUES ('A004', '0010100', 'P04', 'Roma')
INSERT INTO Orar VALUES ('A005', '0010010', 'P03', 'Londra')
INSERT INTO Orar VALUES ('A006', '0101010', 'P01', 'Berlin')
INSERT INTO Orar VALUES ('A007', '0011001', 'P05', 'Atena')
INSERT INTO Orar VALUES ('A008', '0011001', 'P06', 'Lisabona')
```

Pentru a afişa conţinutul tabelului, se foloseşte comanda

SELECT * FROM Orar ORDER BY Cod_avion

COD_AVION	ZILE	PILOT	RUTA
A001	1010100	P01	Cluj
A002	1001000	P02	Paris
A003	0100100	P03	Sibiu
A004	0010100	P04	Roma
A005	0010010	P03	Londra
A006	0101010	P01	Berlin
A007	0011001	P05	Atena
A008	0011001	P06	Lisabona

8 rows returned in 0.00 seconds

b) afişarea orarului săptămânal al avioanelor (fiecare zi a săptămânii cu avioanele care circulă în ziua respectivă):

```
SELECT Cod_avion LUNI FROM Orar WHERE SUBSTR(Zile,1,1) = '1'
SELECT Cod_avion MARTI FROM Orar WHERE SUBSTR(Zile,2,1) = '1'
SELECT Cod_avion MIERCURI FROM Orar WHERE SUBSTR(Zile,3,1) = '1'
SELECT Cod_avion JOI FROM Orar WHERE SUBSTR(Zile,4,1) = '1'
SELECT Cod_avion VINERI FROM Orar WHERE SUBSTR(Zile,5,1) = '1'
SELECT Cod_avion SAMBATA FROM Orar WHERE SUBSTR(Zile,6,1) = '1'
SELECT Cod_avion DUMINICA FROM Orar WHERE SUBSTR(Zile,7,1) = '1'
```

LUNI	MARTI	MIERCURI	JOI	VINERI	SAMBATA	DUMINICA
A001	A003	A001	A002	A001	A005	A007
A002	A006	A004	A007	A003	A006	A008
2 rows r	2 rows ret	A005	A006	A004	2 rows returne	2 rows returne
		A007	A008	3 rows ret		
		A008	4 rows r			
		5 rows return				

c) pilotul cu cele mai multe zboruri săptămânale:

```
SELECT Nume FROM Piloti WHERE Cod_pilot IN
(SELECT Pilot FROM Orar GROUP BY Pilot HAVING
SUM(TO_NUMBER(SUBSTR(Zile,1,1)) + TO_NUMBER(SUBSTR(Zile,2,1)) +
TO_NUMBER(SUBSTR(Zile,3,1)) + TO_NUMBER(SUBSTR(Zile,4,1)) +
TO_NUMBER(SUBSTR(Zile,5,1)) + TO_NUMBER(SUBSTR(Zile,6,1)) +
TO_NUMBER(SUBSTR(Zile,7,1))) =
(SELECT MAX(SUM(TO_NUMBER(SUBSTR(Zile,1,1)) + TO_NUMBER(SUBSTR(Zile,2,1))
+ TO_NUMBER(SUBSTR(Zile,3,1)) + TO_NUMBER(SUBSTR(Zile,4,1)) +
TO_NUMBER(SUBSTR(Zile,5,1)) + TO_NUMBER(SUBSTR(Zile,6,1)) +
TO_NUMBER(SUBSTR(Zile,7,1)))) FROM Orar GROUP BY Pilot))
```

NUME
Ionescu George
1 rows returned i

COD_AVION	ZILE	PILOT	RUTA
A001	1010100	P01	Cluj
A006	0101010	P01	Berlin

S-au folosit funcţiile:

- **SUM(…) –** calculează suma pentru toate zborurile pilotului curent;
- **TO_NUMBER(…)** – transformă în număr caracterul corespunzător unei zile (0 sau 1);
- **SUBSTR(Zile,k,1)** – extrage câte un caracter (0 sau 1) de pe poziţiile 1, 2, …, 7;
- **MAX(…)** – calculează maximul zborurilor pentru piloţi (în cazul nostru 6).

d) Codul şi destinaţia avioanelor pe care le pilotează “Badea Viorel” săptămânal:

```
SELECT Cod_avion, Ruta FROM Orar WHERE Pilot =
(SELECT Cod_pilot FROM Piloti WHERE UPPER(Nume)='BADEA VIOREL')
```

COD_AVION	RUTA
A008	Lisabona

1 rows returned in 0.01 sec

SAU

```
SELECT Cod_avion, Ruta FROM Orar, Piloti WHERE UPPER(Nume)='BADEA VIOREL'
AND Pilot = Cod_Pilot
```

SUBIECTUL 19

Se consideră o bază de date cu 2 tabele: **PERSOANE (Cnp**, C(13) – identifică unic o persoană; **Nume**, C(30) – numele şi prenumele persoanei) şi **CREDITE (Cod_Pers**, C(13) – cod numeric personal persoană creditată; **Data1**, D – data contractării creditului; **Suma**, N(7) – suma contractată; **Perioada**, N(2) – numărul de luni pe care s-a făcut creditul; **Rata**, N(7) – rata lunară pe care trebuie să o plătească persoana creditată; **Data2**, D – termenul la care persoana creditată trebuie să plătească următoarea rată (se actualizează la plata fiecărei rate); **Rest_Plata**, N(7) – suma rămasă de plătit (se actualizează la plata fiecărei rate).

Se cere:

a) crearea tabelelor şi popularea cu date;

b) numele persoanelor care au credite mai mari decât 5 000 lei şi mai mici decât 15 000 lei;

c) numele şi suma restantă pentru persoanele care au depăşit termenul de plată (suma se va calcula ca produs între rata lunară şi numărul de luni, la care se adaugă o restanță de 10%);

d) verificați dacă Popa Ionela are vreun credit în derulare.

REZOLVARE:

a) Pentru crearea tabelului **PERSOANE** se scrie comanda:

```
CREATE TABLE Persoane (Cnp VARCHAR(13) PRIMARY KEY, Nume VARCHAR2(30))
```

*(putem folosi **CHAR(13)** = şir de exact 13 caractere - dar este dificil de completat aceste valori în tabel)*

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Persoane VALUES ('P01', 'Ionescu George')
INSERT INTO Persoane VALUES ('P02', 'Popescu Radu')
INSERT INTO Persoane VALUES ('P03', 'Dragan Mirabela')
INSERT INTO Persoane VALUES ('P04', 'Georgescu Victor')
INSERT INTO Persoane VALUES ('P05', 'Tatu Gabriela')
INSERT INTO Persoane VALUES ('P06', 'Popa Ionela')
```

Pentru a afişa conținutul tabelului, se foloseşte comanda:

```
SELECT * FROM Persoane ORDER BY Cnp
```

CNP	NUME
P01	Ionescu George
P02	Popescu Radu
P03	Dragan Mirabela
P04	Georgescu Victor
P05	Tatu Gabriela
P06	Popa Ionela

6 rows returned in 0.00 se

Pentru crearea tabelului **CREDITE** se scrie comanda:

```
CREATE TABLE Credite (Cod_Pers VARCHAR(13) REFERENCES Persoane(Cnp), Data1
DATE, Suma NUMBER(7), Perioada NUMBER(2), Rata NUMBER(7), Data2 DATE,
Rest_Plata NUMBER(7))
```

Pentru "popularea" cu date se scriu, pe rând, comenzile:

```
INSERT INTO Credite VALUES ('P01', '30-jan-2009', 10000, 24, 500, '25-feb-2009', 12000)
INSERT INTO Credite VALUES ('P02', '10-feb-2009', 15000, 36, 500, '5-mar-2009', 18000)
INSERT INTO Credite VALUES ('P03', '15-jan-2009', 10000, 12, 1000, '10-apr-2009', 10000)
INSERT INTO Credite VALUES ('P04', '10-dec-2008', 20000, 24, 1000, '05-apr-2009', 21000)
INSERT INTO Credite VALUES ('P05', '20-mar-2009', 17000, 24, 850, '15-apr-2009', 20400)
INSERT INTO Credite VALUES ('P06', '15-dec-2008', 7000, 12, 700, '05-apr-2009', 6300)
```

S-a considerat **dobânda de 20%**, indiferent de suma şi perioada creditului, iar **rata** s-a calculat după formula: **(suma+20%din suma)/perioada** (ex: pentru suma de **10000** lei creditată pe **24** de luni, în data de **'30-jan-2009'**, s-a calculat rata astfel: **(10000 + 2000)/24=500** (lei).

Termenul la care o persoană trebuie să plătească următoarea rată (**Data2**) se calculează ca fiind: **Data2 + 30 * nr_rate** (se consideră că o rată trebuie plătită lunar, deci se poate aproxima cu 30 de zile).

Pentru actualizarea datelor din tabel s-a considerat **data curentă**:

CURRENT_DATE
26-MAR-09

Pentru a afişa conţinutul tabelului, se foloseşte comanda:

SELECT * FROM Credite ORDER BY Cod_pers

COD_PERS	DATA1	SUMA	PERIOADA	RATA	DATA2	REST_PLATA
P01	30-JAN-09	10000	24	500	25-FEB-09	12000
P02	10-FEB-09	15000	36	500	05-MAR-09	18000
P03	15-JAN-09	10000	12	1000	10-APR-09	10000
P04	10-DEC-08	20000	24	1000	05-APR-09	21000
P05	20-MAR-09	17000	24	850	15-APR-09	20400
P06	15-DEC-08	7000	12	700	05-APR-09	6300

6 rows returned in 0.01 seconds Download

Să presupunem că persoana cu codul **P01** a mai plătit **două rate**, înainte de aplicarea penalităţilor. În acest caz, trebuie să modificăm conţinutul tabelului **Credite** astfel:

- **Data2** se modifică astfel: **Data2 = Data2 + 30*2**, adică, conform datelor din tabel: **'25-feb-2009' + 60 = '26-apr-2009'**;
- **Rest_plata** se modifică astfel: **Rest_plata = Rest_plata – Rata*2**, adică, conform datelor din tabel: **12000 – 500*2 = 11000**;

Modificarea (actualizarea) conţinutului tabelului pentru persoana cu codul **P01** conform datelor de mai sus se poate face cu comenzile:

UPDATE Credite SET Data2 = Data2+30*2 WHERE Cod_pers = 'P01' şi

UPDATE Credite SET Rest_plata = Rest_plata – Rata*2 WHERE Cod_pers = 'P01' şi se obţine:

COD_PERS	DATA1	SUMA	PERIOADA	RATA	DATA2	REST_PLATA
P01	30-JAN-09	10000	24	500	26-APR-09	11000
P02	10-FEB-09	15000	36	500	05-MAR-09	18000
P03	15-JAN-09	10000	12	1000	10-APR-09	10000
P04	10-DEC-08	20000	24	1000	05-APR-09	21000
P05	20-MAR-09	17000	24	850	15-APR-09	20400
P06	15-DEC-08	7000	12	700	05-APR-09	6300

6 rows returned in 0.00 seconds Download

b) Persoanele care au credite mai mari decât 5 000 lei şi mai mici decât 15 000 lei:

SELECT Nume FROM Persoane WHERE Cnp IN
(SELECT Cod_pers FROM Credite WHERE (Suma > 5000) AND (Suma < 15000))

COD_PERS	DATA1	SUMA	PERIOADA	RATA	DATA2	REST_PLATA
P01	30-JAN-09	10000	24	500	26-APR-09	11000
P02	10-FEB-09	15000	36	500	05-MAR-09	18000
P03	15-JAN-09	10000	12	1000	10-APR-09	10000
P04	10-DEC-08	20000	24	1000	05-APR-09	21000
P05	20-MAR-09	17000	24	850	15-APR-09	20400
P06	15-DEC-08	7000	12	700	05-APR-09	6300

NUME
Ionescu George
Dragan Mirabela
Popa Ionela

3 rows returned i 6 rows returned in 0.01 seconds Download

c) Numele şi suma restantă pentru persoanele care au depăşit termenul de plată (suma restantă se va calcula ca **produs între rata lunară şi numărul de luni**, la care se adaugă o **restanţă de 10%**):

SELECT Nume, 1.10*Rata*TRUNC((CURRENT_DATE-Data2)/30) Suma_restanta FROM Persoane, Credite WHERE Cnp = Cod_pers AND Data2 < CURRENT_DATE

NUME	SUMA_RESTANTA
Popescu Radu	0

1 rows returned in 0.00 seconds

CURRENT_DATE
26-MAR-09

Deşi **Popescu Radu** (cu codul **P02**) îndeplineşte condiţia: **Data2 < CURRENT_DATE**, pentru că **Data2** este **'5-mar-2009'** şi data curentă este **'26-mar-2009'**, nu plăteşte restanţe (numărul de luni de întârziere este **0**). Am aproximat o lună cu 30 de zile (numărul de luni l-am calculat ca fiind câtul împărţirii numărului de zile la 30).

Presupunând că data curentă ar fi **'16-apr-2009'**, atunci suma restantă ar fi:

NUME	SUMA_RESTANTA
Popescu Radu	550

1 rows returned in 0.01 seconds

S-a calculat după formula:
1.10*Rata*1 = 1.10*500*Trunc(to_date ('16-apr-2009')-to_date('5-mar-2009'))/30) = 550*1

Obs: Pentru a putea prelucra o valoare de tip dată calendaristică (**'5-mar-2009'**), trebuie s-o interpretăm ca pe un şir de caractere ce trebuie transformat în dată, adică: **TO_DATE('5-mar-2009')**.

Se mai poate folosi funcţia **MONTHS_BETWEEN(data1, data2)** care calculează numărul de luni dintre cele două date calendaristice date ca parametru (rezultatul este număr real):

SELECT Nume, 1.10*Rata* MONTHS_BETWEEN (CURRENT_DATE, Data2) Suma_restanta FROM Persoane, Credite WHERE Cnp = Cod_pers AND Data2 < CURRENT_DATE

NUME	SUMA_RESTANTA
Popescu Radu	390.529569892473118279569892473118279569

1 rows returned in 0.01 seconds Download

Numărul de luni este subunitar, ceea ce înseamnă că nu ar trebui penalizat (a întârziat cu plata mai puţin de o lună).

Pentru a „repara" problema de mai sus, folosim funcţia **TRUNC** astfel:

SELECT Nume, 1.10*Rata* TRUNC(MONTHS_BETWEEN (CURRENT_DATE, Data2)) Suma_restanta FROM Persoane, Credite WHERE Cnp = Cod_pers AND Data2 < CURRENT_DATE

d) Popa Ionela are vreun credit în derulare?

SELECT Suma Valoare_credit FROM Credite WHERE Cod_pers = (SELECT Cnp FROM Persoane WHERE UPPER(Nume) = 'POPA IONELA' AND Rest_plata>0)

VALOARE_CREDIT
7000

1 rows returned in 0.01

Am considerat că o persoană are credit în derulare dacă mai are de plătit rate la bancă, adică **Rest_plata>0**.

SUBIECTUL 20

Se consideră o bază de date 2 tabele: **FILME (Cod**, N(5) – codul filmului (identifică unic un film); **Nume**, C(15) – titlul filmului; **Regizor**, C(25) – numele regizorului; **Categorie**, C(15) – poate fi: "comedie", "aventură", "horror" etc.; **An_Apar**, N(4) – anul apariției filmului) şi **CINEMA** (**Nume**, C(15) – nume cinematograf; **Film**, N(5) – codul filmului care a rulat în cinematograf; **Nr_Spect**, N(4) – număr spectatori care au vizionat filmul; **Pret_Bilet**, N(6) – prețul unui bilet; **Data1**, D – data de început pentru rularea filmului; **Data2**, D – data de sfârşit pentru rularea filmului).

Se cere:

a) crearea tabelelor şi popularea cu date;

b) să se afişeze filmele care au rulat la cinematograful "Patria", cu următoarele informații: titlul filmului, numărul de spectatori, prețul biletului şi totalul încasărilor;

c) să se afişeze totalul încasărilor la toate cinematografele pentru filmele care au rulat în perioada "10.02.2009 – 10.04.2009";

d) prețul minim şi maxim al unui bilet specificând cinematograful (cinematografele), filmul (filmele) şi regizorul.

REZOLVARE:

a) Pentru crearea tabelului **FILME** se scrie comanda:

CREATE TABLE Filme (Cod NUMBER(5) PRIMARY KEY, Nume VARCHAR2(15), Regizor VARCHAR2(25), Categorie VARCHAR2(15), An_Apar NUMBER(4))

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Filme VALUES (100,'Stars world', 'George Lucas', 'SF', 1980)
INSERT INTO Filme VALUES (101,'Harry Potter', 'Chris Columbus', 'Aventuri', 2001)
INSERT INTO Filme VALUES (102,'Dacii', 'Sergiu Nicolaescu', 'Istoric', 1967)
INSERT INTO Filme VALUES (103,'Balanta', 'Drama', 'Lucian Pintilie', 1985)
INSERT INTO Filme VALUES (104,'Nea Marin', 'Sergiu Nicolaescu', 'Comedie', 1983)
INSERT INTO Filme VALUES (105,'BD la munte', 'Geo Saizescu', 'Comedie', 1972)
INSERT INTO Filme VALUES (106,'Jurassyc Parck', 'Spilberg', 'SF', 1996)
INSERT INTO Filme VALUES (107,'Restul e tacere', 'Nae Caranfil', 'Drama', 2009)
INSERT INTO Filme VALUES (108,'Boogie', 'Radu Muntean', 'Comedie', 2009)

Pentru a afişa conținutul tabelului, se foloseşte comanda:

SELECT * FROM Filme ORDER BY Cod

COD	NUME	REGIZOR	CATEGORIE	AN_APAR
100	Stars war	George Lucas	SF	1980
101	Harry Potter	Chris Columbus	Aventuri	2001
102	Dacii	Sergiu Nicolaescu	Istoric	1967
103	Balanta	Drama	Lucian Pintilie	1985
104	Nea Marin	Sergiu Nicolaescu	Comedie	1983
105	BD la munte	Geo Saizescu	Comedie	1972
106	Jurassyc Parck	Spilberg	SF	1996
107	Restul e tacere	Nae Caranfil	Drama	2009
108	Boogie	Radu Muntean	Comedie	2009

9 rows returned in 0.01 seconds Download

Pentru crearea tabelului **CINEMA** se scrie comanda:

CREATE TABLE Cinema (Nume VARCHAR2(15), Film NUMBER(5) REFERENCES Filme(Cod), Nr_spect NUMBER(4), Pret_bilet NUMBER(6), Data1 DATE, Data2 DATE)

Pentru "popularea" cu date se scriu, pe rând, comenzile:

INSERT INTO Cinema VALUES ('Flacara', 100, 553, 15, '10-dec-2008', '20-dec-2008')
INSERT INTO Cinema VALUES ('Flacara', 101, 1751, 25, '05-jan-2009', '31-jan-2009')
INSERT INTO Cinema VALUES ('Flacara', 103, 373, 10, '07-feb-2009', '20-feb-2009')
INSERT INTO Cinema VALUES ('Flacara', 104, 373, 12, '21-feb-2009', '28-feb-2009')
INSERT INTO Cinema VALUES ('Flacara', 107, 305, 10, '18-feb-2009', '24-feb-2009')
INSERT INTO Cinema VALUES ('Patria', 100, 450, 15, '20-dec-2008', '27-dec-2008')
INSERT INTO Cinema VALUES ('Patria', 101, 754, 25, '18-feb-2009', '28-feb-2009')
INSERT INTO Cinema VALUES ('Patria', 102, 250, 10, '06-jan-2009', '13-jan-2009')
INSERT INTO Cinema VALUES ('Patria', 104, 350, 12, '14-jan-2009', '27-jan-2009')
INSERT INTO Cinema VALUES ('Patria', 107, 650, 10, '04-feb-2009', '17-feb-2009')

Pentru a afişa conţinutul tabelului, se foloseşte comanda:

SELECT * FROM Cinema ORDER BY Nume

NUME	FILM	NR_SPECT	PRET_BILET	DATA1	DATA2
Flacara	100	553	15	10-DEC-08	20-DEC-08
Flacara	101	1751	25	05-JAN-09	31-JAN-09
Flacara	103	373	10	07-FEB-09	20-FEB-09
Flacara	104	373	12	21-FEB-09	28-FEB-09
Flacara	107	305	10	18-FEB-09	24-FEB-09
Patria	100	450	15	20-DEC-08	27-DEC-08
Patria	101	754	25	18-FEB-09	28-FEB-09
Patria	102	250	10	06-JAN-09	13-JAN-09
Patria	104	350	12	14-JAN-09	27-JAN-09
Patria	107	650	10	04-FEB-09	17-FEB-09

10 rows returned in 0.01 seconds Download

b) Filmele care au rulat la cinematograful "Patria", cu următoarele informaţii: titlul filmului, numărul de spectatori, preţul biletului şi totalul încasărilor:

SELECT a.Nume, Nr_spect, Pret_bilet, Nr_spect*Pret_bilet Total_incasari FROM Filme a, Cinema b WHERE Cod = Film AND UPPER(b.Nume)='PATRIA'

NUME	NR_SPECT	PRET_BILET	TOTAL_INCASARI
Stars war	450	15	6750
Nea Marin	350	12	4200
Harry Potter	754	25	18850
Dacii	250	10	2500
Restul e tacere	650	10	6500

5 rows returned in 0.01 seconds Download

c) Totalul încasărilor la toate cinematografele pentru filmele care au rulat în perioada "10.02.2009 – 10.04.2009":

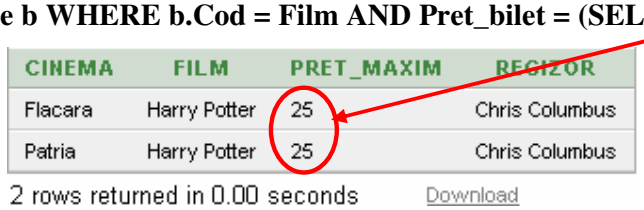
SELECT SUM(Nr_spect*Pret_bilet) Total_Incasari FROM Cinema
WHERE Data1>='10-feb-2009' AND Data2<='10-apr-2009'

TOTAL_INCASARI
26376

1 rows returned in 0.0

d) Prețul minim şi maxim al unui bilet specificând cinematograful (cinematografele), filmul (filmele) şi regizorul:

SELECT a.Nume Cinema, b.Nume Film, Pret_bilet Pret_maxim, Regizor FROM Cinema a, Filme b WHERE b.Cod = Film AND Pret_bilet = (SELECT MAX(Pret_bilet) FROM Cinema)



CINEMA	FILM	PRET_MAXIM	REGIZOR
Flacara	Harry Potter	25	Chris Columbus
Patria	Harry Potter	25	Chris Columbus

2 rows returned in 0.00 seconds Download

SELECT a.Nume Cinema, b.Nume Film, Pret_bilet Pret_minim, Regizor FROM Cinema a, Filme b WHERE b.Cod = Film AND Pret_bilet = (SELECT MIN(Pret_bilet) FROM Cinema)

CINEMA	FILM	PRET_MINIM	REGIZOR
Flacara	Balanta	10	Drama
Flacara	Restul e tacere	10	Nae Caranfil
Patria	Dacii	10	Sergiu Nicolaescu
Patria	Restul e tacere	10	Nae Caranfil

4 rows returned in 0.00 seconds Download