Laborator 4
Tabele relationate.
Constrangerea FOREIGN KEY..REFERENCES.

Problema rezolvata:

Se considera baza de date XE, ca fiind baza de date a unui cabinet medical, in care se stocheaza informatii despre pacientii avuti in evidenta (date personale, date legate de consultatiile efectuate si de platile aferente).

a) Sa se creeze tabelele Pacienti, Consultatii si Plati cu urmatoarea structura:

Tabela PACIENTI :
- CNP CHAR (13) ,
- Nume VARCHAR (30),
- Adresa VARCHAR (30),
- Data_nasterii DATE.

Tabela CONSULTATII:
- CNP CHAR (13) ,
- Diagnostic VARCHAR (30),
- Id_consult INTEGER ,
- Data_consult DATE,
- Plata NUMERIC.

Tabela PLATI:
- Id_consult INTEGER,
- Rata NUMERIC,
- Data_plata DATE.

La crearea tabelelor se vor prevedea toate constrangerile de legatura dintre tabele, precum si constrangerile la nivel de coloana (valori implicite, asigurarea ca s-au introdus date, corectitudinea lor ca unicitate, alte restrictii etc).

Tabela PACIENTI reprezinta o lista cu pacientii unui cabinet medical si datele lor personale. Tabela CONSULTATII contine informatii despre fiecare consult (eventual) efectuat asupra pacientilor din tabela PACIENTI. Cele doua tabele sunt relationate intre ele prin campul CNP (cod numeric personal, unic in tabela PACIENTI). Fiecare pacient este inregistrat doar o data in tabela PACIENTI, dar CNP-ul sau se poate repeta in tabela CONSULTATII, un pacient putand beneficia de oricate consultatii. Orice consultatie implica existenta unei inregistrari a pacientului in lista din tabela PACIENTI.

In tabela CONSULTATII, campul Data_consult reprezinta data efectuarii consulatiei (avand ca valoare implicita data curenta a sistemului), iar campul Plata, costul consultatiei. Odata ce un pacient a beneficiat de o consultatie, el nu mai poate fi sters din tabela PACIENTI decat daca sunt sterse initial consultatiile aferente lui.

Tabela PLATI reprezinta tabela cu platile efectuate de un pacient, presupunand ca plata unei consultatii se poate face in una sau mai multe rate. Tabela PLATI este legata de

tabela CONSULTATII prin campul Id_consult (neputand sa apara o plata pentru un consult neinregistrat). Stergerea unei consultatii din tabela CONSULTATII, pentru care s-au facut plati, implica stergerea automata si a platilor aferente (linii din tabela PLATI). Campul Data_plata reprezinta data in care s-a efectuat o plata pentru o anumita consultative, avand ca valoare implicita data curenta a sistemului.

```
CREATE TABLE PACIENTI
      (CNP CHAR(13) PRIMARY KEY,
       Nume VARCHAR(30) NOT NULL,
       Adresa VARCHAR(30) NOT NULL,
       Data_nasterii DATE NOT NULL);

CREATE TABLE CONSULTATII
      (CNP CHAR(13) NOT NULL REFERENCES PACIENTI(CNP),
       Diagnostic VARCHAR(30),
       Id_consult INTEGER PRIMARY KEY,
       Data_consult DATE DEFAULT SYSDATE,
       Plata NUMERIC);

CREATE TABLE PLATI
     (Id_consult INTEGER NOT NULL
      REFERENCES CONSULTATII(Id_consult) ON DELETE CASCADE,
      Rata NUMERIC NOT NULL,
      Data_plata DATE DEFAULT SYSDATE);
```

Explicatii:

FOREIGN KEY …REFERENCES … - defineste o constrangere de tip cheie straina (sau cheie externa). Intr-o astfel de constrângere sunt implicate doua tabele: o tabelă ‘parinte’ (sau *master*) si o tabela ‘copil’ (sau *slave*).

Constrangerea este definita pe tabela ‘copil’ (presupunand deja creata tabela ‘parinte’, care are in mod obligatoriu definita o cheie primara sau o cheie unica). Constrangerea implica existenta in cele doua tabele a cate unei coloane cu o aceeaşi semnificatie informationala (putand avea acelasi nume sau nume diferite), prin care continuturile celor doua tabele sunt relationate intre ele. Cele doua coloane sunt practic coloane de legatura intre tabele (coloana din tabela ‘parinte’ fiind obligatoriu cheie primara/unica).

In coloana de tip FOREIGN KEY din tabela ‘copil’ pot fi introduse doar valori - oricate- care exista deja in coloana referita prin REFERENCES in tabela ‘parinte’). Cu alte cuvinte, intre tabela ‘parinte’ si tabela ‘copil’, este stabilita o relatie de tipul “una-la mai multe”: unei linii din tabela ‘parinte’ îi pot corespunde una sau mai multe linii în tabela ‘copil’ (sau nici una). De asemenea, in tabela ‘copil’ nu pot exista linii fara corespondent in tabela ‘parinte’ (nu pot exista “linii copil” fara o “linie părinte”, fiecarei “linii copil” corespunzandu-i o singură linie parinte – conditie asigurata de existenta unei chei primare/unice in tabela ‘parinte’).

Clauza ON DELETE CASCADE indică faptul că, atunci cand o linie din tabela ‘parinte’ este stearsa, vor fi sterse si liniile dependente din tabela ‘copil’. Aceasta clauza este optionala. Daca nu se specifica această clauza, liniile din tabela ‘parinte’ nu pot fi sterse daca exista linii corespondente in tabela ‘copil’, fiind necesara o stergere prealabila a acestora.

In rezolvarea de mai sus, am folosit sintaxa de definire a unei constrângeri referenţiale de tip cheie străină la nivel de coloană (a se vedea in curs cele 2 moduri de definire a constrangerilor, la nivel de tabela sau la nivel de coloana).

b) Sa se populeze tabelele cu date (in ordinea corecta), introducand-se se minim 2 pacienti, minim 3 consultatii si respectiv minim 3 plati de rate. Se vor incerca stergeri ale unui pacient, consultatie, plata, in diverse ordini, urmarind modul in care lucreaza constrangerile de legatura intre tabele (vizualizand continutul tabelelor dupa aceste operatii).

```
SET AUTOCOMMIT ON;
SELECT sysdate FROM dual;
ALTER session SET nls_date_format='DD-MM-YYYY';

INSERT INTO pacienti VALUES ('121','Ion','Strada 1','22-03-2008');
INSERT INTO pacienti VALUES ('122','Alex','Strada 2','22-03-2005');
INSERT INTO pacienti VALUES ('123','Vasile','Strada 3','22-03-2015');
INSERT INTO pacienti VALUES ('124','Maria','Strada 4','22-04-2015');

INSERT INTO consultatii VALUES ('121', 'pojar', 1, '02-09-2016', 40);
INSERT INTO consultatii VALUES ('121', 'gripa', 2, '15-09-2017', 55);
INSERT INTO consultatii VALUES ('122', 'viroza',3, '22-11-2017', 50);
INSERT INTO consultatii VALUES ('123', 'gripa', 4, '21-12-2017', 30);

INSERT INTO plati VALUES (1, 2, '02-09-2016');
INSERT INTO plati VALUES (1, 4, '24-09-2016');
INSERT INTO plati VALUES (2, 40, '25-09-2017');
INSERT INTO plati VALUES (3, 20, '25-11-2017');
INSERT INTO plati VALUES (3, 30, '27-11-2017');
```

Tabela PACIENTI:

CNP	Nume	Adresa	Data_nasterii
121	Ion	Strada 1	22-03-2008
122	Alex	Strada 2	22-03-2005
123	Vasile	Strada 3	22-03-2015
124	Maria	Strada 4	22-04-2015

Tabela CONSULTATII:

CNP	Diagnostic	Id_consult	Data_consult	Plata
121	pojar	1	02-09-2016	40
121	gripa	2	15-09-2017	55
122	viroza	3	22-11-2017	50
123	gripa	4	21-12-2017	30

Tabela PLATI:

Id_consult	Rata	Data_plata
1	2	02-09-2016
1	4	24-09-2016
2	40	25-09-2017
3	20	25-11-2017
3	30	27-11-2017

c). Sa se scrie comenzile pentru afisarea intregului continut al celor 3 tabele:

```
SELECT * FROM pacienti;
SELECT * FROM consultatii;
SELECT * FROM plati;
```

d). Presupunand existenta a minim 2 pacienti, minim 3 consultatii si respectiv minim 3 plati de rate se cere scrirea interogarilor corespunzatoare urmatoarelor operatii:

- Pentru care consultatii (Id_consult) s-au facut plati si cat s-a platit?

```
SELECT Id_consult, sum(Rata) as Platit FROM plati GROUP BY Id_consult;
```

- Pentru fiecare consultatie cat mai este de platit si care este CNP-ul pacientului aferent? Ordonati inregistrarile crescator, dupa Id_consult.

```
SELECT A.Id_consult, A.CNP, A.Plata-NVL(sum(B.Rata),0) AS De_plata
  FROM consultatii A, plati B
  WHERE A.Id_consult= B.Id_consult(+)
  GROUP BY A.Id_consult,A.CNP,A.Plata
  ORDER BY Id_consult;
```

Explicatii:

- Semnificatia (+)

Dacă unei linii din prima tabela referita in clauza FROM (in exemplul nostru, tabela consultatii) nu îi corespund linii in a doua tabela din lista (in exemplul nostru, tabela plati), aceste linii nu sunt afisate.

```
SELECT A.Id_consult, A.CNP, A.Plata-sum(B.Rata) AS De_plata
  FROM consultatii A, plati B
  WHERE A.Id_consult= B.Id_consult
  GROUP BY A.Id_consult,A.CNP,A.Plata
  ORDER BY Id_consult;
```

Id_consult	CNP	De_plata
1	121	34
2	121	15
3	122	0

Folosind urmatoarea forma sintactica a comenzii, se va obtine:

```
SELECT A.Id_consult, A.CNP, A.Plata-sum(B.Rata) AS De_plata
  FROM consultatii A, plati B
  WHERE A.Id_consult= B.Id_consult(+)
  GROUP BY A.Id_consult,A.CNP,A.Plata
  ORDER BY Id_consult;
```

Id_consult	CNP	De_plata
1	121	34
2	121	15
3	122	0
4	123	NULL

Se poate observa diferenta, fiind afisate inclusiv liniile din prima tabela pentru care nu există corespondent in cealalta tabela (DECI INCLUSIV CONSULTATIA CU ID-UL 4) (datele lipsa primind valori NULL).

- Functia NVL compara 2 valori primite ca si parametri de intrare si o returneaza pe cea nenula. Daca ambele sunt nenule, o returneaza pe prima.

```
SELECT NVL(50, NULL), NVL(NULL,70), NVL(50,70) FROM dual;
Rezultat:
NVL(50,NULL) NVL(NULL,70) NVL(50,70)
------------ ------------ ----------
          50           70         50
```

In cazul nostru este obligatorie folosirea functiei NVL. Altfel, celor care nu au facut nicio plata (deci care nu apar in tabela PLATI) li se va afisa ca suma restanta de plata - NULL (desi sunt datornici PLINI, neavand nicio rata platita).

- Pentru care consultatii (Id_consult) s-a platit integral si care este CNP-ul pacientului care a platit?

```
Varianta 1:

SELECT A.Id_consult, A.CNP, A.Plata-NVL(sum(B.Rata),0) AS De_plata
      FROM consultatii A, plati B
      WHERE A.Id_consult= B.Id_consult(+)
      GROUP by A.Id_consult,A.CNP,A.Plata
      HAVING A.Plata-NVL(sum(B.Rata),0)=0
      ;

Varianta 2:

SELECT A.Id_consult, A.CNP, A.Plata-NVL(sum(B.Rata),0) AS De_plata
FROM consultatii A
RIGHT JOIN plati B On A.Id_consult= B.Id_consult
GROUP by A.Id_consult,A.CNP,A.Plata
HAVING A.Plata-NVL(sum(B.Rata),0)=0;
```

- Pentru fiecare consult, care este numele pacientilor, CNP-ul, si cat mai au de plata pentru consult (ordonat in ordinea descrescatoare a sumelor de plata) – de folosit nume de alias obligatoriu.

```
Varianta 1:

SELECT A.Id_consult, A.CNP, C.Nume, A.Plata-NVL(sum(B.Rata),0) AS De_plata
FROM consultatii A, plati B, pacienti C
WHERE A.Id_consult= B.Id_consult(+) and A.CNP=C.CNP
GROUP BY A.Id_consult, A.CNP, A.Plata, C.Nume
ORDER BY De_plata DESC;

Varianta 2:

SELECT A.Id_consult, A.CNP, C.Nume, A.Plata-NVL(sum(B.Rata),0) AS De_plata
FROM consultatii A
LEFT JOIN plati B ON A.Id_consult = B.Id_consult
INNER JOIN pacienti C ON A.CNP=C.CNP
GROUP BY A.Id_consult, A.CNP, A.Plata, C.Nume
ORDER BY De_plata DESC;
```

Explicatii:

Strategia pentru scrierea comenzii SELECT este data de următoarele reguli:

- Se determina coloanele care vor fi vizualizate si se includ in clauza SELECT . In situatia in care tabelele interogate contin coloane cu acelasi nume, numele acestora trebuie obligatoriu precedat de numele tabelei din care coloana se selecteaza sau de alias-ul acesteia. Altfel, va aparea o eroare de ambiguitate.
- Se determina tabelele implicate si se includ in clauza FROM.
- Se determină conditiile care limiteaza selectarea. Conditiile care se refera la grupuri de date apar in clauza HAVING, iar cele care se refera la linii individuale apar in clauza WHERE. Clauza WHERE trebuie sa contina obligatoriu conditia/conditiile de relationare intre tabelele interogate;
- Daca comanda SELECT include functii de grup, atunci se utilizeaza clauza GROUP BY, gruparea facandu-se dupa toate atributele (coloanele) enumerate in SELECT. In clauza GROUP BY insa, se pot folosi si coloane care nu sunt enumerate in SELECT, in functie de cerintele problemei.
- Daca este necesară valoarea unui atribut din alt tabel sau este necesara o functie de grup in clauza WHERE, atunci se utilizeaza o subinterogare (alt SELECT).
- Cu ajutorul clauzei ORDER BY se precizeaza criteriul de ordonare a afisarii liniilor (inregistrarilor).
- Alias-urile nu pot fi folosite in clauza WHERE, GROUP BY, HAVING , insa pot fi folosite in clauza ORDER BY.

Problema propusa:

Se considera baza de date XE, ca fiind baza de date a asociatiilor de proprietari din Timisoara, in care se tine o evidenta a platilor cheltuielilor de intretinere.

a). Sa se creeze tabelele Asociatii, Locatari, Cheltuieli, cu urmatoarea structura:

Tabela ASOCIATII:

- Cod_asociatie INTEGER ,
- Nume_asociatie VARCHAR (50).

Tabela LOCATARI:

- CNP CHAR(13),
- Nume VARCHAR(30),
- Sc VARCHAR(4),
- Ap VARCHAR(2),
- Telefon VARCHAR (15),
- Email VARCHAR(20).
- Cod_asociatie INTEGER.

Tabela CHELTUIELI:

- Id INTEGER
- CNP CHAR(13),
- De_plata NUMERIC,
- Data_scadenta DATE,
- Platit NUMERIC,
- Data_plata DATE.

La crearea tabelelor se vor prevedea toate constrangerile de legatura dintre tabele, precum si constrangerile la nivel de coloana (valori implicite, asigurarea ca s-au introdus date, corectitudinea lor ca unicitate, alte restrictii etc).

Tabela ASOCIATII contine o lista cu toate asociatiile de proprietari din Timisoara. Fiecare asociatie este identificata printr-un cod unic si diferit de NULL (astfel, campul Cod_asociatie va trebui sa aiba aplicata constrangerea PRIMARY KEY). Campul Nume_asociatie trebuie sa fie unic si diferit de NULL.

Tabela LOCATARI contine o lista cu toti locatarii ce apartin de asociatiile inregistrate in tabela ASOCIATII. Astfel, orice inregistrare a unui locatar in tabela LOCATARI, implica existenta unei inregistrari a asociatiei din care acesta face parte, in tabela ASOCIATII.

Fiecare locatar este identificat in mod unic prin CNP (astfel, campul CNP va trebui sa fie PRIMARY KEY). Campurile Nume, Sc, Ap, Telefon, Email, Cod_Asociatie trebuie sa fie diferite de NULL.

Cele doua tabele sunt relationate intre ele prin campul Cod_asociatie. Fiecare asociatie este inregistrata doar o data in tabela ASOCIATII, dar codul sau, Cod_asociatie, se va repeta in tabela LOCATARI, pentru fiecare locatar ce-i apartine.

Odata ce o asociatie a fost inregistrata in tabela ASOCIATII, aceasta nu mai poate fi stearsa decat daca sunt stersi initial toti locatarii care apartin de ea.

In tabela CHELTUIELI se memoreaza un id ce identifica in mod unic cheltuielile de intretinere (campul Id, PRIMARY KEY), CNP-ul locatarilor (campul CNP, obligatoriu a fi completat), suma de plata a cheltuielilor locatarilor (campul De_plata, obligatoriu a fi completat), data scadenta a efectuarii platilor (campul Data_scadenta, obligatoriu a fi completat), sumele platite (campul Platit, necompletat inaintea efectuarii platii) si data in care s-au efectuat platile (campul Data_plata, necompletat inaintea efectuarii platii).

Tabela CHELTUIELI va fi relationata cu tabela LOCATARI prin campul CNP, pentru fiecare locatar creandu-se lunar o noua inregistrare in tabela CHELTUIELI, odata cu generarea cheltuielilor de intretinere. In Tabela CHELTUIELI nu pot fi salvate decat cheltuielile locatarilor care sunt inregistrati deja in tabela LOCATARI. Stegerea unui locatar din tabela LOCATARI presupune stergerea automata a tuturor cheltuielilor asociate acestuia.

b) Sa se populeze cele 3 tabele cu datele de mai jos, iar ulterior sa se afiseze continutul acestora.

Tabela ASOCIATII:

Cod_asociatie	Nume_asociatie
1	Asociatia de proprietari Florin Medelet, nr. 5
2	Asociatia de proprietari Oreon, nr. 10

Tabela LOCATARI:

CNP	Nume_locatar	Sc	Ap	Telefon	Email	Cod_asociatie
1234567891230	Ambrus Dan	A	1	0722000000	a@yahoo.com	1
1234567891231	Laudat Ana	A	2	0722111111	b@yahoo.com	1
1234567891232	Popovici Ioan	B	1	0722222222	c@yahoo.com	1
1234567891233	Matei Vasile	A	1	0722333333	d@yahoo.com	2
1234567891234	Blaga Ion	A	2	0722444444	e@yahoo.com	2

Tabela CHELTUIELI:

Id	CNP	De_plata	Data_scadenta	Platit	Data_plata
1	1234567891230	120	22.02.2020	120	20.02.2020
2	1234567891231	150	17.02.2020	NULL	NULL
3	1234567891231	300	17.03.2020	150	16.03.2020
4	1234567891232	200	27.03.2020	300	24.03.2020
5	1234567891233	300	15.02.2020	200	14.02.2020
6	1234567891233	400	15.03.2020	NULL	NULL
7	1234567891233	200	15.04.2020	NULL	NULL
8	1234567891234	500	25.03.2020	500	22.03.2020

c). Sa se afiseze pentru toti locatarii asociatiilor inregistrate in baza de date, urmatoarele:
- CNP-ul locatarului;
- situatia restantelor (dupa caz: 0 – daca locatarul a facut plati integrale; o valoare pozitiva, daca exista restante; o valoare negativa, daca locatarul a facut plati in avans).

Inregistrarile vor fi ordonate dupa CNP, crescator, sub forma:

CNP	REST
1234567891230	0
1234567891231	300
1234567891232	-100
1234567891233	500
1234567891234	0

d). Sa se afiseze pentru toti locatarii asociatiilor inregistrate in baza de date, urmatoarele:
- CNP-ul locatarului;
- situatia restantelor (dupa caz: 0 – daca locatarul a facut plati integrale; o valoare pozitiva, daca exista restante; o valoare negativa, daca locatarul a facut plati in avans).
- numele asociatiei.

Inregistrarile vor fi ordonate dupa numele asociatiei, crescator, sub forma:

CNP	REST	NUME_ASOCIATIE
1234567891230	**0**	Asociatia de proprietari Florin Medelet, nr. 5
1234567891231	**300**	Asociatia de proprietari Florin Medelet, nr. 5
1234567891232	**-100**	Asociatia de proprietari Florin Medelet, nr. 5
1234567891233	**500**	Asociatia de proprietari Oreon, nr. 10
1234567891234	**0**	Asociatia de proprietari Oreon, nr. 10

e). Sa se afiseze toti locatarii asociatiilor inregistrate in baza de date, care au facut plati integrale, ordonat crescator dupa numele asociatiei, sub forma:

CNP	REST	NUME_ASOCIATIE
1234567891230	**0**	Asociatia de proprietari Florin Medelet, nr. 5
1234567891234	**0**	Asociatia de proprietari Oreon, nr. 10

f). Sa se afiseze toti locatarii asociatiilor inregistrate in baza de date, care mai au sume restante de platit, ordonat descrescator dupa restul de plata, sub forma:

CNP	REST	NUME_ASOCIATIE
1234567891233	**500**	Asociatia de proprietari Oreon, nr. 10
1234567891231	**300**	Asociatia de proprietari Florin Medelet, nr. 5

g). Sa se afiseze toti locatarii asociatiilor inregistrate in baza de date, care au facut plati in avans, ordonat crescator dupa numele asociatiei, sub forma:

CNP	REST	NUME_ASOCIATIE
1234567891232	**-100**	Asociatia de proprietari Florin Medelet, nr. 5

h). Sa se afiseze locatarii ‘Asociatiei de proprietari Florin Medelet, nr. 5’ care mai au sume restante de platit, ordonat dupa nume, sub forma:

CNP	NUME	SC	AP	REST	NUME_ASOCIATIE
1234567891231	Laudat Ana	A	2	300	Asociatia de proprietari Florin Medelet, nr. 5

Identificarea asociatiei se va face pe baza codului de asociatie, care in cazul de fata are valoarea 1.

i). Sa se afiseze suma restanta existenta la nivel de fiecare asociatie, ordonat crescator dupa numele asociatiei, sub forma:

COD_ASOCIATIE	NUME_ASOCIATIE	REST
1	Asociatia de proprietari Florin Medelet, nr. 5	200
2	Asociatia de proprietari Oreon, nr. 10	500