Aprob

șeful Catedrei MI

\_\_\_\_\_\_\_\_\_ M. Petic

„\_\_\_” \_\_\_\_\_\_ 2022

Cozma Denis

Proba I pentru evaluarea curentă la unitatea de curs „Practica SGBD”

Fie este dată baza de date cu următoarele tabele:

PERSOANE (idPersoana, Numele, Vârsta)

RUDE (idPersoana1, idPersoana2)

AMICI (idPersoana1, idPersoana2)

Inserturile

*/\**

*CREATE TABLE Persoane (*

*idPersoana SMALLINT PRIMARY KEY,*

*Numele VARCHAR(30) NOT NULL,*

*Virsta SMALLINT NOT NULL*

*)*

*create table Amici(*

*idpersoana1 SMALLINT NOT null,*

*idpersoana2 SMALLINT NOT null,*

*PRIMARY KEY(idpersoana1, idpersoana2),*

*foreign key (idpersoana1) references persoane (idpersoana),*

*foreign key (idpersoana2) references persoane (idpersoana)*

*);*

*create table Rude(*

*idpersoana1 SMALLINT NOT null,*

*idpersoana2 SMALLINT NOT null,*

*PRIMARY KEY(idpersoana1, idpersoana2),*

*foreign key (idpersoana1) references persoane (idpersoana),*

*foreign key (idpersoana2) references persoane (idpersoana)*

*);*

*\*/*

*/\**

*INSERT INTO Persoane(idPersoana, Numele, Virsta*

*)*

*VALUES(*

*1,*

*'Elvi',*

*19*

*),*

*(*

*2,*

*'Farouk',*

*19*

*),*

*(*

*3,*

*'Sam',*

*19*

*),*

*(*

*4,*

*'Tiany',*

*19*

*),*

*(*

*5,*

*'Nadia',*

*14*

*),*

*(*

*6,*

*'Chris',*

*12*

*),*

*(*

*7,*

*'Kris',*

*10*

*),*

*(*

*8,*

*'Bethany',*

*16*

*);*

*INSERT INTO Persoane(idPersoana, Numele, Virsta*

*)*

*VALUES(*

*9,*

*'Louis',*

*17*

*),*

*(*

*10,*

*'Austin',*

*22*

*),*

*(*

*11,*

*'Gabriel',*

*21*

*),*

*(*

*12,*

*'Jessica',*

*20*

*),*

*(*

*13,*

*'John',*

*16*

*),*

*(*

*14,*

*'Alfred',*

*19*

*),*

*(*

*15,*

*'Samantha',*

*17*

*),*

*(*

*16,*

*'Craig',*

*17*

*);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(1,2);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(1,3);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(2,4);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(2,6);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(3,9);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(4,9);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(7,5);*

*INSERT INTO amici (idpersoana1, idpersoana2)*

*VALUES(5,8);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (6, 10);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (13, 6);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (7, 6);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (8, 7);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (9, 11);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (12, 9);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (10, 15);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (12, 11);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (12, 15);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (13, 16);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (15, 13);*

*INSERT INTO Amici (idpersoana1, idpersoana2)*

*VALUES (16, 14);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (4,6);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (2,4);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (9,7);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (7,8);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (11,9);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (13,10);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (14,5);*

*INSERT INTO rude (idpersoana1, idpersoana2) VALUES (12,13);*

*\*/*

1. Să se proiecteze o interogare care ar determina numele și vârsta persoanelor cu cel mai mare număr de rude.

**SELECT** persoane.Numele, persoane.Virsta, **COUNT**(persoane.idPersoana) **AS** rude

**FROM** persoane

**INNER** **JOIN** rude r **ON** persoane.idPersoana = r.idpersoana1

**OR** persoane.idPersoana = r.idpersoana2

**GROUP** **BY** persoane.idPersoana

**HAVING** rude = (**SELECT** **COUNT**(persoane.idPersoana) **AS** rude

**FROM** persoane

**INNER** **JOIN** rude r **ON** persoane.idPersoana = r.idpersoana1

**OR** persoane.idPersoana = r.idpersoana2

**GROUP** **BY** persoane.idPersoana

**LIMIT** 1);

1. Să se proiecteze o interogare care ar returna numele persoanelor care sunt amici. Perechile de prieteni nu trebuie să se repete.

**SELECT** p1.Numele, p2.Numele

**FROM** amici a

**INNER** **JOIN** persoane p1 **ON** p1.idPersoana = a.idpersoana1

**INNER** **JOIN** persoane p2 **ON** p2.idPersoana = a.idpersoana2

1. Să se proiecteze o interogare care ar determina rudele persoanelor care sunt mai mici ca ei. Să se afișeze perechi de persoane fără repetări.

*SELECT P1.NUMELE, P1.Virsta, P2.NUMELE, P2.Virsta*

*FROM PERSOANE AS P1*

*INNER JOIN RUDE ON P1.IDPERSOANA = RUDE.IDPERSOANA1*

*INNER JOIN PERSOANE AS P2 ON P2.IDPERSOANA = RUDE.IDPERSOANA2*

*WHERE P1.VIRSTA<>P2.VIRSTA;*

1. Să se elaboreze o procedură care pentru o persoană ar determina lista membrilor familiei sale. Dacă A este membru al familiei X și B este membru al familiei X, atunci B trebuie să apară în lista membrilor familiei lui A. Ca parametru este dat numele persoanei.
2. Să se elaboreze o procedură care pentru o persoană ar determina numărul de prieteni. Ca parametru este dat numele persoanei.

delimiter $$

**CREATE** **PROCEDURE** proc7(nume\_pers **VARCHAR**(20))

**BEGIN**

**SELECT** **COUNT**(persoane.idPersoana)

**FROM** persoane

**INNER** **JOIN** amici **ON** persoane.idPersoana = amici.idpersoana1

**OR** persoane.idPersoana = amici.idpersoana2

**WHERE** persoane.Numele = nume\_pers

**GROUP** **BY** persoane.idPersoana;

**END**; $$

delimiter ;

**CALL** proc6("Farouk");

\*/

1. Să se elaboreze o funcție care returnează True sau False dacă două persoane sunt prieteni. Numele persoanelor sunt date ca parametru.

**CREATE** **FUNCTION** func10(nume1 **VARCHAR**(200),nume2 **VARCHAR**(200))

**RETURNS** **VARCHAR**(5)

**DETERMINISTIC**

**BEGIN**

**if** **EXISTS** (**SELECT** \*

**FROM** persoane

**INNER** **JOIN** amici **ON** idpersoana = idpersoana1

**OR** idpersoana=idpersoana2

**WHERE** numele = nume1 **OR** numele = nume2 )

**ELSE**

**RETURN** "TRUE";

**ELSE**

**RETURN** "FALSE";

**END** **if**;

**END** $$

delimiter ;

**SELECT** func6('Elvira','Kris');

1. Să se creeze o vedere care ar conține lista persoanelor ce au un membru de familie, dar nu au nici un prieten. Este posibil de a adăuga date în tabele prin intermediul vederii? Este posibil de a elimina date din tabele prin intermediul vederii? Este posibil de a modifica date în tabele prin intermediul vederii? Dați exemple de operații de actualizare prin intermediul vederilor, dacă este posibil.
2. Dați câte un exemplu de vedere care ar permite de a adăuga date în tabelul PERSOANE, a doua vedere ar permite modificarea, și a treia ar permite eliminarea datelor din tabelul menționat. Toate trei vederi trebuie să conțină interogări diferite.

*/\* 8 a*

*CREATE VIEW Vedere AS*

*SELECT \**

*FROM persoane*

*SELECT \* FROM Vedere*

*INSERT INTO Vedere VALUES (22, 'DENIS', 22);*

*\*/*

*/\* 8b*

*CREATE VIEW Vedere AS*

*SELECT \**

*FROM persoane*

*SELECT \* FROM V8*

*UPDATE VIEW8*

*SET Numele = "Cozma" WHERE Numele = "Ion";*

*CREATE VIEW Vedere AS*

*SELECT \**

*FROM persoane*

*SELECT \* FROM Vedere*

*DELETE FROM Vedere WHERE idPersoana <18;*

Titularul disciplinei Corina Negara