Student: Сozma Denis

**Laborator 4 - Declansatori**

1. ***Completați tabelele cu date folosind scriptul SQL creat în lucrarea nr. 2.***

Trebuie să obțineți următoarea structură.

***Tabelul Persoane***

|  |  |  |
| --- | --- | --- |
| **idPersoana** | **Numele** | **Vîtsta** |
| 1 | Elvi | 19 |
| 2 | Farouk | 19 |
| 3 | Sam | 19 |
| 4 | Tiany | 19 |
| 5 | Nadia | 14 |
| 6 | Chris | 12 |
| 7 | Kris | 10 |
| 8 | Bethany | 16 |
| 9 | Louis | 17 |
| 10 | Austin | 22 |
| 11 | Gabriel | 21 |
| 12 | Jessica | 20 |
| 13 | John | 16 |
| 14 | Alfred | 19 |
| 15 | Samantha | 17 |
| 16 | Craig | 17 |

***Tabelul Amici***

|  |  |
| --- | --- |
| **Idpersoana1** | **Idpersoana2** |
| 1 | 2 |
| 1 | 3 |
| 2 | 4 |
| 2 | 6 |
| 3 | 9 |
| 4 | 9 |
| 7 | 5 |
| 5 | 8 |
| 6 | 10 |
| 13 | 6 |
| 7 | 6 |
| 8 | 7 |
| 9 | 11 |
| 12 | 9 |
| 10 | 15 |
| 12 | 11 |
| 12 | 15 |
| 13 | 16 |
| 15 | 13 |
| 16 | 14 |

***Tabelul Rude***

|  |  |
| --- | --- |
| **Idpersoana1** | **Idpersoana2** |
| 4 | 6 |
| 2 | 4 |
| 9 | 7 |
| 7 | 8 |
| 11 | 9 |
| 13 | 10 |
| 14 | 5 |
| 12 | 13 |

**CREATE** **DATABASE** laborator4;

**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);

***Sarcini de realizat:***

* 1. Să se creeze un declanșator care după adăugarea unei persoane în tabelul Persoane, va adăuga o legătură de prietenie dintre persoana dată și Elvia. Să se verifice lucrul declanșatorului prin adăugarea unei persoane în tabelul Persoane. Care este efectul declanșatorului?

delimiter $$

**CREATE** **TRIGGER** trig1

**AFTER** **INSERT**

**ON** persoane

**FOR** **EACH** **row**

**BEGIN**

**INSERT** **INTO** amici(idpersoana1, idpersoana2)

**VALUES** (**NEW**.idPersoana, (**SELECT** persoane.idPersoana

**FROM** persoane

**WHERE** persoane.Numele = "Elvi"));

**END** $$

delimiter ;

**INSERT** **INTO** persoane **VALUES** (17, 'Dumitru', 21);

* 1. Să se creeze un declanșator care înainte de adăugare a unei persoane în tabelul Persoane, verifică dacă această persoană există deja în tabel. Dacă există deja o persoană cu același nume (ca și cel inserat) să se afișeze un mesaj de eroare „Atenție! Așa persoană există!”. Să se verifice lucrul declanșatorului prin adăugarea unei persoane în tabelul Persoane cu același nume, cu un nume nou diferit. Care este efectul declanșatorului?

Obs. Se va folosi următoarea structură pentru afișarea mesajului:

signal sqlstate '45000' set message\_text = 'MESAJ'

delimiter $$

**CREATE** **TRIGGER** trig2

**BEFORE** **INSERT**

**ON** persoane

**FOR** **EACH** **ROW**

**BEGIN**

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

**FROM** persoane

**WHERE** persoane.Numele = **NEW**.Numele)

**then**

**SIGNAL** **SQLSTATE** '45000' **SET** **MESSAGE\_TEXT** = 'Atentie! Asa persoana exista.';

**END** **if**;

**END** $$

delimiter ;

**INSERT** **INTO** persoane **VALUES** (18, 'Dumitru', 21);

**INSERT** **INTO** persoane **VALUES** (18, 'Dima', 21);

* 1. Să se creeze un declanșator care după inserarea a două persoane în tabelul rude creează o legătură de prietenie dintre aceste două persoane. Dacă inserăm perechea (x, y) în tabelul Rude, în tabelul Amici se va insera aceeași pereche de persoane (x, y). Dacă perechea introdusă există, se va returna un mesaj de eroare. Să se verifice lucrul declanșatorului.

DELIMITER $$

**CREATE** **TRIGGER** trig3

**AFTER** **INSERT** *-- schimbat before in after*

**ON** rude

**FOR** **EACH** **ROW**

**BEGIN**

**IF** **EXISTS** (**SELECT** \*

**FROM** rude

**WHERE** (rude.idPersoana1 = **NEW**.idPersoana1

**AND** rude.idPersoana2 = **NEW**.idPersoana2)

**OR** (rude.idPersoana1 = **NEW**.idPersoana2

**AND** rude.idPersoana2 = **NEW**.idPersoana1))

**THEN**

**SIGNAL** **SQLSTATE** '45000' **SET** **MESSAGE\_TEXT** = 'Asa pereche exista.';

**ELSEIF** **NOT** **EXISTS** (**SELECT** \*

**FROM** amici

**WHERE** (amici.idPersoana1 = **NEW**.idPersoana1

**AND** amici.idPersoana2 = **NEW**.idPersoana2)

**OR** (amici.idPersoana1 = **NEW**.idPersoana2

**AND** amici.idPersoana2 = **NEW**.idPersoana1))

**THEN**

**INSERT** **INTO** amici(idPersoana1, idPersoana2) **VALUES** (**NEW**.idPersoana1, **NEW**.idPersoana2);

**END** **IF**;

**END** $$

DELIMITER ;

**INSERT** **INTO** rude **VALUES** (1,2);

* 1. Să se creeze un declanșator care va elimina legătura de prietenie între două persoane, dacă legătura de rudenie între ele este eliminată. Dacă perechea (x, y) nu mai este în tabelul Rude, atunci perechea dată este eliminată automat din tabelul Amici. Demonstrați lucrul declanșatorului.

delimiter $$

**CREATE** **TRIGGER** trig4

**AFTER** **DELETE**

**ON** rude

**FOR** **EACH** **ROW**

**BEGIN**

**DELETE** **FROM** amici

**WHERE** (amici.idpersoana1 = **OLD**.idpersoana1 **AND** amici.idpersoana2 = **OLD**.idpersoana1)

**OR** (amici.idpersoana1 = **OLD**.idpersoana2 **AND** amici.idpersoana2 = **OLD**.idpersoana1);

**END** $$

delimiter ;

**DELETE** **FROM** rude **WHERE** idpersoana1 = 17 **AND** idpersoana2 = 18;

* 1. Să se creeze un declanșator care ar elimina toate referințele la o persoană din tabelele Rude și Amici, când această persoană este eliminată din tabelul Persoane. Demonstrați lucrul declanșatorului.

delimiter $$

**CREATE** **TRIGGER** trig5

**BEFORE** **DELETE**

**ON** persoane

**FOR** **EACH** **ROW**

**BEGIN**

**DELETE** **FROM** rude

**WHERE** **OLD**.idPersoana = rude.idpersoana1 **OR** **OLD**.idPersoana = rude.idpersoana2;

**DELETE** **FROM** amici

**WHERE** **OLD**.idPersoana = amici.idpersoana1 **OR** **OLD**.idPersoana = amici.idpersoana2;

**END** $$

delimiter ;

**DELETE** **FROM** persoane **WHERE** idPersoana = 18;

* 1. Pentru această sarcină eliminați cheile străine din tabelul Amici. Să se creeze un declanșator care va lucra ca cheile străine eliminate. Dacă în tabelul Amici se va adăuga o pereche de persoane (x, y), declanșatorul se va asigura că persoanele cu codul x și y există în tabelul Persoane. În caz de depistare a încercării de a introduce persoane care nu există în tabelul Persoane se va afișa un mesaj de eroare. Să se demonstreze situațiile în care acționează declanșatorul dat.

delimiter $$

**CREATE** **TRIGGER** trig6

**BEFORE** **INSERT**

**ON** amici

**FOR** **EACH** **ROW**

**BEGIN**

**if** **NOT** **EXISTS** (**SELECT** \*

**FROM** persoane

**INNER** **JOIN** persoane p

**WHERE** persoane.idPersoana = **NEW**.idPersoana1 **AND** p.idPersoana = **NEW**.idPersoana2)

**then**

**SIGNAL** **SQLSTATE** '45000' **SET** **MESSAGE\_TEXT** = 'Asa persoana nu exista';

**END** **if**;

**END** $$

delimiter ;

**INSERT** **INTO** amici **VALUES** (18,17);

* 1. Să se creeze un declanșator care va implementa modificarea în cascadă a datelor, astfel la modificarea datelor din tabelul Persoane (idPersoana) aceste modificări vor fi propagate în tabelul Amici. Demonstrați lucrul declanșatorului.

delimiter $$

**CREATE** **TRIGGER** trig7

**AFTER** **UPDATE**

**ON** persoane

**FOR** **EACH** **ROW**

**BEGIN**

**UPDATE** amici

**SET** amici.idpersoana1 = **NEW**.idPersoana

**WHERE** amici.idpersoana1 = **OLD**.idPersoana;

**UPDATE** amici

**SET** amici.idpersoana2 = **NEW**.idPersoana

**WHERE** amici.idpersoana2 = **OLD**.idPersoana;

**END** $$

delimiter ;

**UPDATE** persoane

**SET** persoane.idPersoana = 20 **WHERE** persoane.idPersoana = 17;

* 1. Să se creeze un declanșator care va implementa eliminarea în cascadă a datelor, astfel la eliminarea unei peroane din tabelul Persoane (idPersoana) va fi eliminată legăturile persoanei date din tabelul Amici. Demonstrați lucrul declanșatorului.

delimiter $$

**CREATE** **TRIGGER** trig8

**BEFORE** **DELETE**

**ON** persoane

**FOR** **EACH** **ROW**

**BEGIN**

**DELETE**

**FROM** amici

**WHERE** amici.idpersoana1 = **OLD**.idPersoana

**OR** amici.idpersoana2 = **OLD**.idPersoana;

**END** $$

delimiter ;

**DELETE** **FROM** persoane **WHERE** persoane.idPersoana = 20;