**Proiect SGBD**

**Baza de date pentru un atelier auto**

Pomparau Renato-Emil

Grupa 241

Cuprins

[1. Descrierea modelului real și a regulilor de funcționare 2](#_Toc124519637)

[2. Constrangeri 2](#_Toc124519638)

[3. Entitati 3](#_Toc124519639)

[4. Relatii 3](#_Toc124519640)

[5. Atribute 5](#_Toc124519641)

[6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5. 7](#_Toc124519642)

[7. Diagrama conceptuala cu crearea unui tabel asociativ intre doua tabele many to many. 7](#_Toc124519643)

[9. FN1-3 9](#_Toc124519644)

[10. Crearea tabelelor în SQL și inserarea de date. 11](#_Toc124519645)

[11. Cerintele 6-13 din proiect: 18](#_Toc124519646)

1. Descrierea modelului real și a regulilor de funcționare

In vederea crearii unei baze de date, am ales proiectarea unei baze de date ce ofera informatii despre un atelier auto care sa ofere o gestionare usoara a comenzilor, a clientilor, a angajatilor, a pieselor auto si a distribuitorilor de piese auto.

Reguli de functionare:

* Atelierul auto are clienti care pot solicita reparatii pentru unul sau mai multe autovehicule de categorii diferite (masini, motociclete). Fiecare comanda este achitata in unul dintre cele doua moduri: cash sau card sau suma poate fi acoperita de asigurare.
* Angajatii ocupa posturi diferite, mecanicii se ocupa de repararea masinilor pe partea tehnica, vulcanizatorul se ocupa de anvelope, vopsitorul de vopsirea masinii iar personalul relatii-clineti se ocupa de interactiunea directa cu clientii.
* Comanda unui client poate fi indeplinida de unul sau mai multi angajati
* Piesele sau schimburile necesare sunt furnizate prin intermediul unor distribuitori.
* Comanda clientului consta in reparatia pentru un autovehicul care face parte dintr-o anume categorie: masina sau motocicleta

1. Constrangeri

Restrictiile de functionare sunt:

* O comanda/reparatie este plasata de un client si trebuie achitata printr-o modalitate disponibila.
* Fiecare piesa de schimb sau materie prima necesara (vopsea) este adusa de un singur furnizor.
* O reparatie presupuna schimbul macar al unei piese sau a interventiei unui angajat, dar nu fiecare piesa a autovehiculului trebuie schimbata pentru fiecare comanda.
* Mecanicii se pot ocupa de schimbul macar a unei piese, dar fiecare angajat lucreaza individual la cate o problema
* Comanda de reparatie trebuie sa fie pentru un autovehicul care face parte dintr-o categorie mentionata.

1. Entitati

Entitatile enumerate sunt independente, cu exceptia subentitatilor Mecanic, Vulcanizator, Vopsitor, Spalator si Relatii-clienti.

*REPARATIE*: cerere prin care o persoana solicita o reparatie sau mai multe pentru autovehiculul sau, solicitari indeplinite de angajatii atelierului auto. *REPARATIA* este platita sub mai multe forme. Cheia primara a entitatii este *ID\_Comanda*.

*CLIENT*: cu cheia primara *ID\_Client*, aceasta entitate reprezinta un client care solicita o reparatie din partea atelierului auto.

*MOD\_DE\_PLATA*: modul prin care clientul plateste reparatia plasata în cadrul atelierului auto. Atributul *ID\_Mod\_Plata* este cheia primară a acestei entități.

*ANGAJAT:* Cheie primara: *ID\_Angajat.* Lucreaza pentru atelierul auto si poate fi *Mecanic, Vulcanizator, Vopsitor, Relatiii-clienti.* Aceste subentitati au cheia primara *ID\_Angajat.*

*CATEGORIE: ID\_Categorie,* denumeste tipul autovehiculului, motocicleta sau masina.

*AUTOVEHICUL:* Obiectul asupra caruia s-a solicitat reparatia, urmand a fi reparat. Autovehiculele pot fi impartite in 2 categorii. Cheia ID\_Autovehicul reprezinta cheia primara a acestei entitati.

*DISTRIBUITOR*: persoană fizică sau juridică care furnizează diferite ingrediente necesare în prepararea produselor. Cheia primară a acestei entități este *ID\_Distribuitor*.

*PIESA/Schimb\_necesar*: defectiunea care trebuie remediata, cu *ID\_Piesa*.

1. Relatii

*CLIENT\_solicita\_ REPARATIE* = relaţie care leagă entităţile CLIENT şi *REPARATIE*, reflectând

legătura dintre acestea (ce reparatie solicita un anumit client). Ea are cardinalitatea minimă 1:1 (un client trebuie să solicite cel puțin o reparatie, iar o reparatie trebuie să fie cerută de un client) şi cardinalitatea maximă 1:n (un client poate solicita mai multe reparatii, iar o reparatie poate fi cerută de un singur client).

*REPARATIE\_platita\_MOD\_DE\_PLATA* = relaţie care leagă entităţile REPARATIE și MOD\_DE\_PLATA, reflectând legătura dintre acestea (prin ce mod de plată a fost plătită o comandă). Relația are cardinalitatea minimă 0:1 (o comandă trebuie să fie plătită intr-un anumit mod, dar nu trebuie să fie neapărat o comandă plătită într-un anumit mod) şi cardinalitatea maximă n:1 (o comandă poate fi plătită într-un singur mod, iar în același mod de plată pot fi plătite mai multe comenzi).

*ANGAJAT\_repara\_AUTOVEHICUL\_din\_REPARATII = relaţie de tip 3 ce leagă entităţile ANGAJAT, AUTOVEHICUL şi REPARATII, reflectând ce angajat a reparat un anumit produs și pentru ce comandă. Denumirea acestei relaţii va fi repara.*

*PIESA/SCHIMB\_NECESAR\_furnizat\_de\_DISTRIBUITOR* = relaţie care leagă entităţile *PIESA/SCHIMB\_NECESAR* şi *DISTRIBUITOR*, reflectând legătura dintre acestea (ce ingrediente aduce un anumit distribuitor). Ea are cardinalitatea minimă 0:1 (un ingredient trebuie să fie adus de un furnizor, iar un furnizor nu trebuie să aducă neapărat un ingredient) şi cardinalitatea maximă n:1 (un ingredient poate fi adus de un singur distribuitor, iar un distribuitor poate aduce mai multe ingrediente).

*AUTOVEHICUL\_primeste\_ PIESA/SCHIMB\_NECESAR* = relaţie care leagă entităţile *AUTOVEHICUL* şi *PIESA/SCHIMB\_NECESAR*, reflectând legătura dintre acestea (piesa trebuie inlocuit in autovehicul).Relația are cardinalitatea minimă 0:1 (un autovehicul trebuie să aiba cel puțin o piesa schimbata, iar o piesa nu trebuie să fie schimnata neaparat intr-un autovehicul decat daca este defecta si la cererea clientului) şi cardinalitatea maximă m:n (un autovehicul poate avea nevoie de mai multe piese inlocuite, iar o piesa poate fi schimbata in mai multe autovehicule).

*AUTOVEHICUL\_apartine \_CATEGORIE* = relaţie care leagă entităţile *AUTOVEHICUL* şi CATEGORIE, reflectând legătura dintre acestea (din ce categorie face parte un AUTOVEHICUL). Ea are cardinalitatea minimă 1:1 (dintr-o categorie trebuie să facă parte cel puțin un autovehicul , iar un autovehicul trebuie să facă parte dintr-o categorie) şi cardinalitatea maximă 1:n (dintr-o categorie pot face parte mai multe produse, iar un produs poate face parte dintr- o singura categorie).

1. Atribute

ENTITATI:

*CLIENT:*

* *id\_client* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui client.
* *nume* = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele clientului.
* *prenume* = variabilă de tip caracter, de lungime maximă 25, care reprezintă prenumele clientului.
* *nr\_telefon* = variabilă de tip caracter, de lungime 10, care reprezintă numarul de telefon al clientului.

*REPARATIE*:

* *id\_comanda* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unei comenzi.
* *data\_comanda* = variabilă de tip dată calendaristică, care reprezintă data la care a fost plasată comanda.
* *id\_mod\_plata* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui mod de plată. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul MOD\_PLATA.
* *id\_client* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul clientului care a plasat comanda. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CLIENT.

*MOD\_DE\_PLATA*:

* *id\_mod\_plata* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui mod de plată.
* *denumire* = variabilă de tip caracter, de lungime maximă 20, care reprezintă denumirea modului de plată al comenzii. Atributul poate lua valorile cash, card, asigurare.

*ANGAJAT*:

* *id\_angajat* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui angajat.
* *nume* = variabilă de tip caracter, de lungime maximă 25, care reprezintă numele angajatului. *prenume* = variabilă de tip caracter, de lungime maximă 25, care reprezintă prenumele angajatului.
* *nr\_telefon* = variabilă de tip caracter, de lungime 10, care reprezintă numarul de telefon al angajatului.
* *tip\_angajat* = variabilă de tip caracter, de lungime maxima 10, care reprezintă funcţia pentru care a fost angajată persoana respectivă. De exemplu, poate să fie mecanic vopsitor, vulcanizator sau relatii-clienti.

*Subentitatile* mecanic vopsitor, vulcanizator sau relatii-clienti au ca atribute id\_angajat si anii de experienta.

*CATEGORIE*:

* *id\_categorie* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unei categorii.
* *nume\_categorie* = variabilă de tip caracter, de lungime maximă 20, care reprezintă numele categoriei (moto, masina)

*AUTOVEHICUL*:

* *id\_autovehicul* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui produs.
* *Id\_client =* variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui client. Trebuie sa corespunda cu cheia primara din tabelul CLIENT.
* *tip* = variabilă de tip caracter, de lungime maximă 5, care reprezintă denumirea produsului. (masina sau moto)
* *pret* = variabilă de tip întreg, de lungime maximă 5, care reprezintă prețul reparatiilor facute la autovehicul.
* *id\_categorie* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unei categorii. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul CATEGORIE.

*PIESA/Schimburi\_necesare*:

* *id\_piesa* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unei piese. Daca e 0 inseamna ca s-a efectuat doar o manopera care nu a implicat schimbarea unei piese(presiune, vopsit etc).
* *denumire* = variabilă de tip caracter, de lungime maximă 20, care reprezintă denumirea Piesei sau a manoperei care s-a realizat (presiune, vopsit etc).
* *id\_distribuitor* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui furnizor. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul DISTRIBUITOR.

*DISTRIBUITOR*:

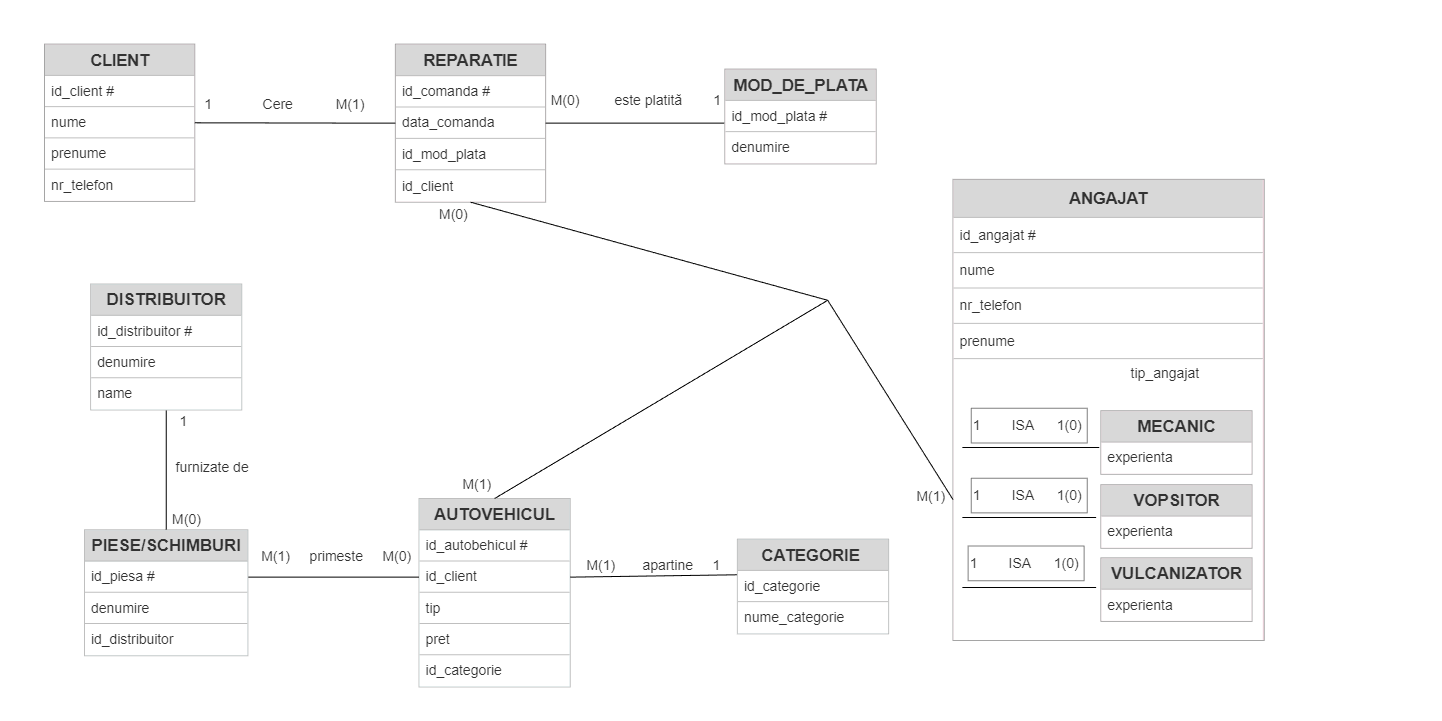
* *id\_distribuitor* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui furnizor.
* *denumire* = variabilă de tip caracter, de lungime maximă 20, care reprezintă denumirea furnizorului.

RELATII:

Relația *ANGAJAT*\_ repara/vopseste/vulcanizeaza *\_AUTOVEHICUL\_din\_REPARATIE* are ca atribute:

* *id\_angajat* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui angajat. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul ANGAJAT.
* *id\_autovehicul* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unui produs. Atributul trebuie să corespundă la o valoare a cheii primare din tabelul AUTOVEHICUL.
* *id\_comanda* = variabilă de tip întreg, de lungime maximă 5, care reprezintă codul unei comenzi. Atributul trebuie să corespundă la o valoare a cheii prim are din tabelul *REPARATIE*.

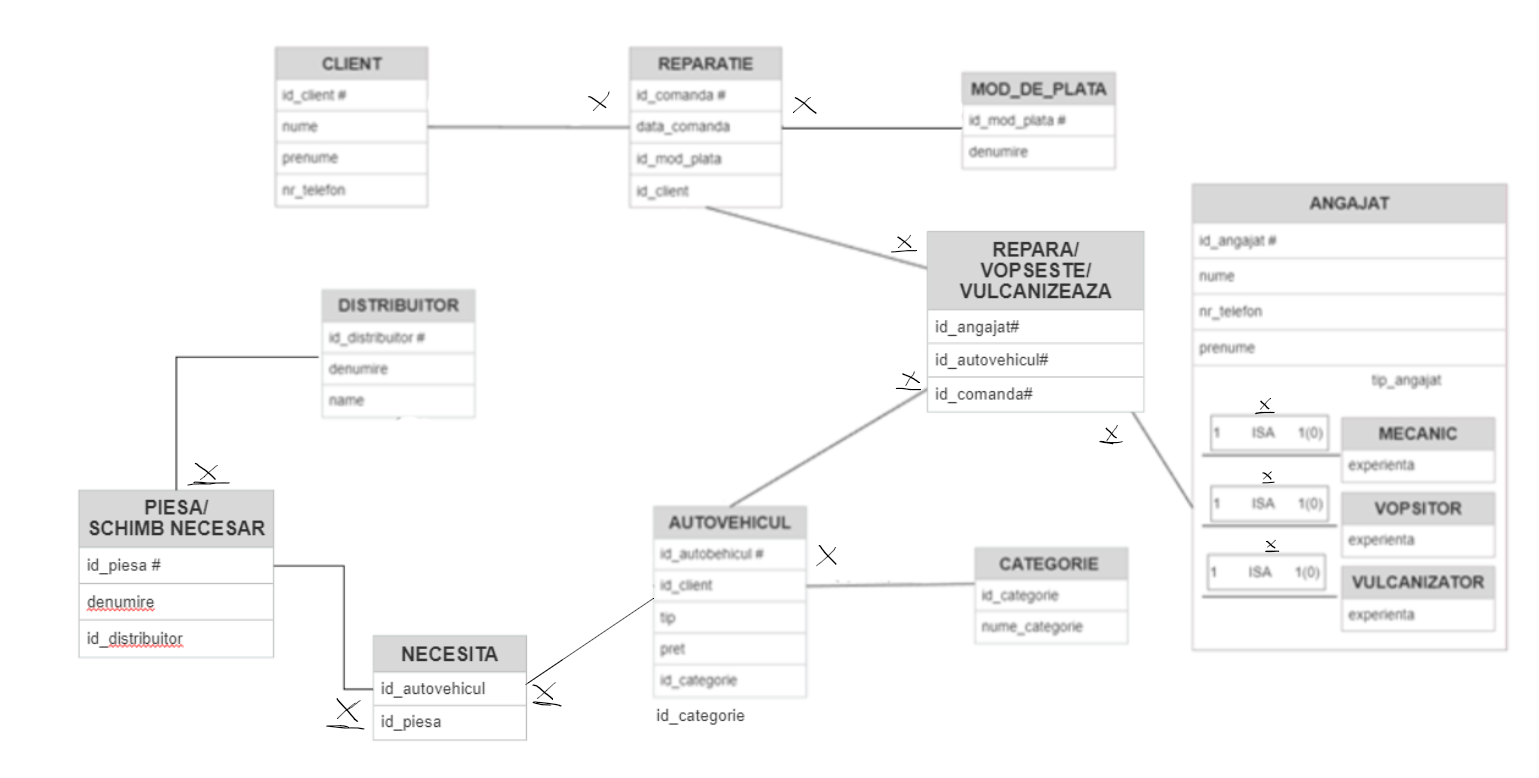
1. Realizarea diagramei entitate-relație.



repara/vopseste



1. Diagrama conceptuala.





1. Schemele relationale corespunzatoare diagramei conceptuale.

REPARATIE (id\_comanda#, data\_comanda, id\_mod\_plata(FK), id\_client(FK))

CLIENT (id\_client#, nume, prenume, nr\_telefon)

MOD\_DE\_PLATA (id\_mod\_plata #, denumire)

REPARA/VOPSESTE/VULCANIZEAZA (id\_angajat#, id\_autovehicul#, id\_comanda#)

ANGAJAT (id\_angajat#, nume, nr\_telefon, prenume)

MECANIC (id\_angajat#, experienta)

VOPSITOR (id\_angajat#, experienta)

VULCANIZATOR (id\_angajat#, experienta)

AUTOVEHIUL (#id\_autovhicul#, id\_client(FK), tip, pret, id\_categorie(FK))

CATEGORIE (#id\_categorie, nume\_categorie)

NECESITA (id\_autovehicul#, id\_piesa#)

PIESA/SCHIMB NECESAR (id\_piesa#,denumire, id\_distribuitor(FK))

DISTRIBUITOR (id\_distribuitior#, denumire, nume)

1. FN1-3

**Pasii pentru FN1:**

-Toate randurile trebuie sa fie identificate in mod unic de aceea adaug ID\_client.

-Fiecare celula trebuie sa contina o singura valoare de accea adaug o linite pentru fiecare element din lista de reparatie.

-Toate datele trebuie sa fie atomice

|  |  |  |
| --- | --- | --- |
| ID\_client | Nume | Reparatie |
| 1 | Renato Pomparau | Audi,BMW,Kawasaki |
| 2 | Alex  Alex | Dacia, Skoda |
| 3 | Renato Pomparau | Audi,BMW,Kawasaki |

|  |  |
| --- | --- |
| Nume | Reparatie |
| Renato Pomparau | Audi,BMW,Kawasaki |
| Alex  Alex | Dacia, Skoda |
| Renato Pomparau | Audi,BMW,Kawasaki |

|  |  |
| --- | --- |
| ID\_client | Nume |
| 1 | Renato Pomparau |
| 2 | Alex  Alex |
| 3 | Renato Pomparau |

|  |  |
| --- | --- |
| ID\_client | Reparatie |
| 1 | Audi |
| 1 | BMW |
| 1 | Kawasaki |
| 2 | Dacia |
| 2 | Skoda |
| 3 | Audi, |
| 3 | BMW |
| 3 | Kawasaki |

|  |  |
| --- | --- |
| ID\_client | Reparatie |
|  |  |
| 1 | Audi |
| 1 | BMW |
| 1 | Kawasaki |
| 2 | Dacia |
| 2 | Skoda |
| 3 | Audi, |
| 3 | BMW |
| 3 | Kawasaki |

|  |  |  |
| --- | --- | --- |
| ID\_client | Nume | Prenume |
| 1 | Renato | Pomparau |
| 2 | Alex | Alex |
| 3 | Renato | Pomparau |

**Pasii pentru FN2:**

- Trebiuie sa fie in FN1.

- Ficare atribuit care nu este cheie este dependent de intreaga cheie primara.

Spre exemplu, in tabelul urmator obeserv ca cele 3 atribute, tip/marca si nr.reparatii nu sunt chei primare iar tip/marca nu depinde de cheia compusa formata din cele 3 chei primare (id\_angajat, id\_autovehicul, id\_comanda) deoarece tip/marca depinde doar de id\_autovehicul, in timp de nr. reparatii depinde de intreaga cheie compusa, reprezentand numarul de reparatii la care va fi supus autovehiculul cu id\_autovehicul, reparatii efectuate de angajatul cu id\_angajat, conform comenii id\_comanda.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id\_angajat | Id\_autovehicul | Id\_comanda | Tip / Marca | Nr.reparatii |
| 123 | **100** | **1** | **Mercedes** | **1** |
| 456 | **101** | **2** | **Mercedes** | **2** |
| 123 | **102** | **3** | **Maybach** | **5** |

**Tabel Repara... Tabel Autovehicul**

|  |  |  |  |
| --- | --- | --- | --- |
| Id\_angajat | Id\_autovehicul | Id\_comanda | Nr.reparatii |
| 123 | **100** | **1** | **1** |
| 456 | **101** | **2** | **2** |
| 123 | **102** | **3** | **5** |

|  |  |
| --- | --- |
| Id\_autovehicul | Tip / Marca |
| 100 | Mercedes |
| 101 | Mercedes |
| 102 | Maybach |

**Pasii pentru FN3:**

**- sa fie in FN2**

**- sa nu exista dependente tranzitive**

|  |  |  |  |
| --- | --- | --- | --- |
| PROGRAMARE | | | |
| id\_programare# | data\_programarii | Locul\_Reparatie | Problema |
| 001 | 21-03-2002 | Anvelope | Schimbare cauciucuri |
| 010 | 22-03-2002 | Vopsit | Vopsea exterior |
| 011 | 23-03-2002 | Vopsit | Vopsea exterior |
| 100 | 24-03-2002 | Anvelope | Presiune |

Locul \_reparatii si data\_programarii depind de id\_programare.

Problema depinde de locul\_reparatii. Creez un nou tabel indetificat unic in functie de id\_reparatie.

|  |  |  |
| --- | --- | --- |
| id\_reparatie# | Locul\_Reparatie | Problema |
| 1 | Anvelope | Schimbare cauciucuri |
| 2 | Vopsit | Vopsea exterior |
| 3 | Anvelope | Presiunea |
| 4 | Vopsit | Vopsea exterior |

|  |  |  |
| --- | --- | --- |
| id\_programare# | data\_programarii | id\_reparatie |
| 001 | 21-03-2002 | 1 |
| 010 | 21-03-2002 | 2 |
| 011 | 21-03-2002 | 3 |
| 100 | 21-03-2002 | 4 |

1. Crearea tabelelor în SQL și inserarea de date.

CREATE TABLE CLIENT

(id\_client number(5),

nume varchar2(25) constraint c\_nume not null, prenume varchar2(25),

nr\_telefon char(10),

CONSTRAINT client\_pk PRIMARY KEY (id\_client), unique(nr\_telefon)

);

CREATE SEQUENCE SEQ\_CLIENT

INCREMENT by 1

START WITH 1

MINVALUE 1

MAXVALUE 1000 NOCYCLE;

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Pomparau', 'Renato', '0720005401');

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Duduman', 'Cristian', '0720005402');

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Moisii', 'Lucian', '0720005403');

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Stacy ', 'Hughes', '0720005404');

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Kathy ', 'George', '0720005405');

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Adelyn', 'Pomparau', '0735874350');

INSERT INTO CLIENT(id\_client, nume, prenume, nr\_telefon)

VALUES (SEQ\_CLIENT.NEXTVAL, 'Eric', 'King', '0720005409');

SELECT \* FROM CLIENT;

CREATE TABLE MOD\_DE\_PLATA

(id\_plata number(5), denumire varchar2(20) check (denumire in ('cash','card','asigurare')),

CONSTRAINT id\_plata\_pk PRIMARY KEY (id\_plata)

);

drop table MOD\_DE\_PLATA;

drop table reparatii;

INSERT INTO MOD\_DE\_PLATA

VALUES (1, 'cash');

INSERT INTO MOD\_DE\_PLATA

VALUES (2, 'card') ;

INSERT INTO MOD\_DE\_PLATA

VALUES (3 ,'cash') ;

INSERT INTO MOD\_DE\_PLATA

VALUES (4,'asigurare') ;

INSERT INTO MOD\_DE\_PLATA

VALUES (5, 'card') ;

commit;

CREATE TABLE REPARATII

(id\_comanda number(5),

data\_comanda date default sysdate,

id\_plata number(5) constraint mod\_not\_null not null,

id\_client number(5) ,

CONSTRAINT reparatii\_pk PRIMARY KEY (id\_comanda),

CONSTRAINT reparatie\_id\_plata\_fk foreign key(id\_plata) references MOD\_DE\_PLATA(id\_plata),

CONSTRAINT reparatie\_client\_fk foreign key(id\_client) references CLIENT(id\_client)

);

create SEQUENCE SEQ\_REPARATII

INCREMENT by 2

START WITH 21

MINVALUE 21

MAXVALUE 10000 NOCYCLE;

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('09-03-2021','dd-mm-yyyy'), 1, 5);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('15-10-2020','dd-mm-yyyy'), 2, 4);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('03-05-2020','dd-mm-yyyy'), 3, 3);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('05-07-2021','dd-mm-yyyy'), 4, 2);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('18-10-2019','dd-mm-yyyy'), 5, 1);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('18-10-2023','dd-mm-yyyy'), 5, 1);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('19-10-2023','dd-mm-yyyy'), 5, 1);

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('20-10-2023','dd-mm-yyyy'), 5, 1);

SELECT \* FROM REPARATII;

CREATE TABLE categorie

(id\_categorie number(7),

nume\_categorie varchar(20)

constraint nume\_categ\_not\_null not null,

CONSTRAINT categorii\_pk PRIMARY KEY (id\_categorie), unique(nume\_categorie)

);

drop table categorie;

CREATE SEQUENCE SEQ\_CATEGORIE

INCREMENT by 10

START WITH 300

MINVALUE 300

MAXVALUE 10000 NOCYCLE;

INSERT INTO categorie

VALUES (SEQ\_CATEGORIE.NEXTVAL, 'Masini');

INSERT INTO categorie

VALUES (SEQ\_CATEGORIE.NEXTVAL, 'Motociclete');

INSERT INTO categorie

VALUES (SEQ\_CATEGORIE.NEXTVAL, 'Barci');

INSERT INTO categorie

VALUES (SEQ\_CATEGORIE.NEXTVAL, 'Microbuze');

INSERT INTO categorie

VALUES (SEQ\_CATEGORIE.NEXTVAL, 'Autobuze');

CREATE TABLE distribuitor

(id\_furnizor number(5),

denumire varchar2(20) not null, nume varchar(10),

CONSTRAINT furnizori\_pk PRIMARY KEY (id\_furnizor)

);

create sequence SQL\_distribuitor

INCREMENT by 10

START WITH 1000

MINVALUE 1000

MAXVALUE 90000 NOCYCLE;

INSERT INTO distribuitor

VALUES (SQL\_distribuitor.NEXTVAL, 'SC 1', 'Strada 1');

INSERT INTO distribuitor

VALUES (SQL\_distribuitor.NEXTVAL, 'SC 2', 'Strada 2');

INSERT INTO distribuitor

VALUES (SQL\_distribuitor.NEXTVAL, 'SC 3', 'Strada 3');

INSERT INTO distribuitor

VALUES (SQL\_distribuitor.NEXTVAL, 'SC 4', 'Strada 4');

INSERT INTO distribuitor

VALUES (SQL\_distribuitor.NEXTVAL, 'SC 5', 'Strada 5');

CREATE TABLE Piese

(id\_piesa number(5),

denumire varchar2(20) not null,

id\_furnizor number(5) not null,

CONSTRAINT piese\_pk PRIMARY KEY (id\_piesa),

CONSTRAINT piese\_distribuitor\_fk foreign key(id\_furnizor) references distribuitor(id\_furnizor), unique(denumire)

);

Create Sequence sqe\_piese

INCREMENT by 1

START WITH 170

MINVALUE 170

MAXVALUE 90000 NOCYCLE;

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Anvelope', 1010);

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Luneta', 1020);

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Faruri', 1030);

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Vopsea exterior', 1040);

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Retapitare', 1000);

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Retapitare2', 1000);

INSERT INTO Piese

VALUES(sqe\_piese.NEXTVAL, 'Retapitare3', 1000);

CREATE TABLE autovehicul

(id\_autovehicul number(5),

id\_client number(5),

nume varchar(20),

pret number(3) not null,

id\_categorie number(5) not null,

check(pret>0),

CONSTRAINT autovehicul\_pk PRIMARY KEY (id\_autovehicul),

CONSTRAINT autovhicul\_categorie\_fk foreign key(id\_categorie) references categorie(id\_categorie),

CONSTRAINT autovehicul\_client\_fk foreign key(id\_client) references CLIENT(id\_client)

);

CREATE SEQUENCE SEQ\_PRODUS

INCREMENT by 1

START WITH 60

MINVALUE 50

MAXVALUE 90000 NOCYCLE;

drop table autovehicul;

drop table necesita;

drop sequence SEQ\_PRODUS;

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,1, 'Bugatti Chiron', 300, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,2, 'Mc LarenP1', 40,300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,3, 'Ferrari LaFerrari.', 600, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,4, 'Koenigsegg Agera RS.', 900, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,3, 'Pagani Huarya.', 900, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,5, 'Bugatti Veyron.', 900, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,6, 'Kawasaki Ninja', 900, 310);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,7, 'Cuirasate', 900, 320);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,6 ,'Crucisatoare', 900, 320);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,7,'Distrugatoare', 900, 320);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,1, 'Dacia Logan ', 300, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,1, 'Dacia Logan1 ', 300, 300);

INSERT INTO autovehicul

VALUES(SEQ\_PRODUS.NEXTVAL,1, 'Dacia Logan2 ', 300, 300);

commit;

CREATE TABLE necesita

(id\_autovehicul number(5) constraint produs\_not\_null not null, id\_piesa number(5) constraint ingred\_not\_null not null,

CONSTRAINT necesita\_autovehicul\_fk foreign key(id\_autovehicul) references autovehicul(id\_autovehicul),

CONSTRAINT necesita\_piese\_fk foreign key(id\_piesa) references piese(id\_piesa),

CONSTRAINT pk\_necesita primary key(id\_autovehicul, id\_piesa)

);

commit;

select \* from autovehicul;

select \* from piese;

INSERT INTO necesita VALUES(61,170);

INSERT INTO necesita VALUES(61,172);

INSERT INTO necesita VALUES(61,173);

INSERT INTO necesita VALUES(62,174);

INSERT INTO necesita VALUES(62,171);

INSERT INTO necesita VALUES(63,172);

INSERT INTO necesita VALUES(63,173);

INSERT INTO necesita VALUES(63,171);

INSERT INTO necesita VALUES(64,171);

INSERT INTO necesita VALUES(65,172);

INSERT INTO necesita VALUES(66,173);

INSERT INTO necesita VALUES(67,174);

INSERT INTO necesita VALUES(67,175);

INSERT INTO necesita VALUES(70,174);

INSERT INTO necesita VALUES(71,175);

INSERT INTO necesita VALUES(72,174);

commit;

drop table angajat;

CREATE TABLE angajat

(id\_angajat number(5),

nume varchar(25) ,

prenume varchar(25),

email varchar(30),

nr\_telefon varchar(25),

data\_angajare date default sysdate,

tip\_angajat varchar2(20),

experienta number(2),

CONSTRAINT angajati\_pk PRIMARY KEY (id\_angajat),

unique(nr\_telefon)

);

select \* from angajat;

INSERT INTO angajat

VALUES(200, 'Moise', 'Lucain', 'lucika@angajat.ro', '0712345678',to\_date('01-10-2019','dd-mm-yyyy'), 'Vulcanizator', null);

INSERT INTO angajat

VALUES(201, 'Duduman', 'Cristy ', 'kitydudu@angajat.ro', '0712345648', to\_date('01-10-2019','dd-mm-yyyy'), 'Mecanic', 2);

INSERT INTO angajat

VALUES(202, 'Adelyn', 'Pomarau', 'adelyn@angajat.ro', '0712345688', to\_date('02-07- 2019','dd-mm-yyyy'), 'Vopsitor', null);

INSERT INTO angajat

VALUES(203, 'Renato', 'Pompatrau', 'pompa@angajat.ro', '0712345478', to\_date('03-02- 2021','dd-mm-yyyy'), 'Mecanic', 3);

INSERT INTO angajat

VALUES(204, 'Yamil', 'Davos', 'habibi@angajat.ro', '0712345668', to\_date('04-05- 2015','dd-mm-yyyy'), 'Vulcanizator', null);

SELECT \* FROM reparatii;

CREATE TABLE repara

(id\_angajat number(5) ,

id\_comanda number(5) ,

id\_autovehicul number(5),

data\_inceput date,

data\_finalizare date default null,

CONSTRAINT repara\_angajat\_fk foreign key(id\_angajat) references angajat(id\_angajat),

CONSTRAINT repara\_autovehicul\_fk foreign key(id\_autovehicul) references autovehicul(id\_autovehicul),

CONSTRAINT reoara\_reparatie\_fk foreign key(id\_comanda) references reparatii(id\_comanda),

Constraint pk\_repara primary key(id\_angajat, id\_autovehicul,id\_comanda)

);

select \* from reparatii; -- id comanda 43-2-51 auto 60-69 200-204

INSERT INTO repara VALUES(200,21,61,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(200,23,62,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(200,23,63,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(200,25,64,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(201,25,65,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(201,27,66,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(201,27,67,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(202,25,61,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(203,27,63,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(203,27,65,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(204,21,66,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(203,35,72,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(203,33,71,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

INSERT INTO repara VALUES(204,31,70,TO\_DATE('2003/05/03 21:02:44', 'yyyy/mm/dd hh24:mi:ss'),null);

select \* from reparatii;

SELECT \* FROM repara;

COMMIT;

select a.nume,count(\*) nr\_de\_reparatii

from necesita n join autovehicul a

on a.id\_autovehicul=n.id\_autovehicul

group by a.nume;

### Cerintele 6-13:

Cerinta 6:

---Cerinta 6: Sa se afiseze autovehiculele din categoria data de la tastatura

---care au nevoie de interventii si numarul pieselor necesare.

CREATE OR REPLACE procedure vehicule\_categorie

(Categorie IN autovehicul.id\_categorie%TYPE)

as

--Categorie varchar(8) := '&categorie';

TYPE vector\_masini IS TABLE OF autovehicul%rowtype index by binary\_integer;

masini vector\_masini;

TYPE nume\_numar IS RECORD(

nume\_masina autovehicul.nume%TYPE,

numar\_piese number

);

TYPE vector\_masini\_piese IS VARRAY(20)

OF nume\_numar;

vec1 vector\_masini\_piese;

BEGIN

select \* bulk collect into masini from autovehicul where id\_categorie=Categorie;

select a.nume,count(\*)

bulk collect into

from autovehicul a,necesita n

where a.id\_autovehicul = n.id\_autovehicul

group by a.nume;

FOR i in masini.first..masini.last loop

FOR j in vec1.first..vec1.last loop

if masini(i).nume=vec1(j).nume\_masina then

DBMS\_OUTPUT.PUT\_LINE(vec1(j).nume\_masina || ' necesita o schimbare de '||vec1(j).numar\_piese ||' piese.');

end if;

end loop;

end loop;

end;

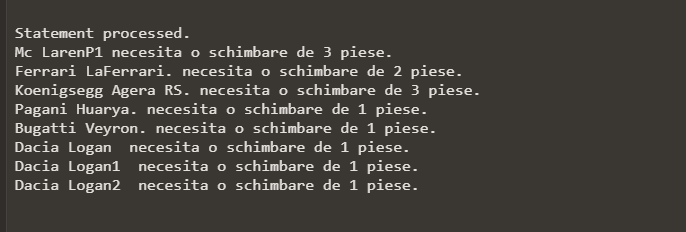
DECLARE

cat autovehicul.id\_categorie%TYPE := '&id\_categorie';

BEGIN

vehicule\_categorie(cat);

END;



--Cerinta 7:

--Pentru un angajat angajat al atelierului auto sa se afiseze masinile pe care le-a

--reparat si piesele pe care le-a folosit la fiecare masina.

-- folosesc 2 cursori,macar unul cu parametru.

-- deschid cursorul cu parametru id\_autovehicul care va stoca masinile pe care le repara

-- pentru fiecare masina iterez prin piesele pe care le are nevoie.

select denumire from piese where id\_piesa= TO\_CHAR(170);

create or replace procedure angajati\_masini

(id\_emp IN angajat.id\_angajat%type)

as

nume\_piesa piese.denumire%type;

c2\_variable necesita.id\_piesa%type;

c1\_variable autovehicul.id\_autovehicul%type;

cursor c1 is

select id\_autovehicul

from repara

where id\_angajat=id\_emp;

cursor c2(id\_masina necesita.id\_autovehicul%type) is

select id\_piesa

from necesita

where id\_autovehicul=id\_masina;

employee\_exist NUMBER;

masina\_curenta autovehicul.nume%type;

begin

SELECT id\_angajat INTO employee\_exist

FROM angajat WHERE id\_angajat = id\_emp;

open c1;

loop fetch c1 into c1\_variable;

exit when c1%notfound;

select nume into masina\_curenta from autovehicul where id\_autovehicul=c1\_variable;

dbms\_output.put(masina\_curenta ||' are nevoie de urmatoarele piese: ');

open c2(c1\_variable);

loop fetch c2 into c2\_variable;

exit when c2%notfound;

select denumire into nume\_piesa from piese where id\_piesa= c2\_variable;

dbms\_output.put(nume\_piesa ||' ');

end loop;

dbms\_output.put\_line('');

close c2;

end loop;

close c1;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista angajatul');

--WHEN OTHERS THEN

--DBMS\_OUTPUT.PUT\_LINE('An error occurred');

end;

/

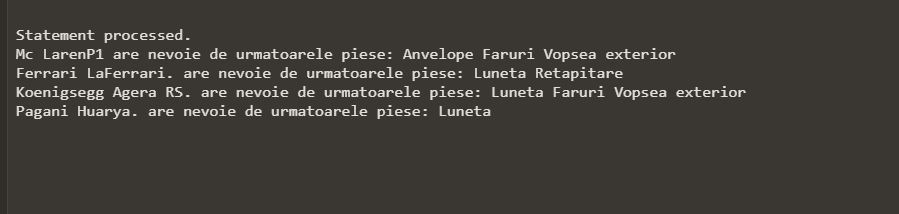
declare

angajatul angajat.id\_angajat%TYPE := '200';

BEGIN

angajati\_masini(angajatul);

end;



--Cerinta 8:

--Se da id-ul unui distribuitor si o categorie de autovehicle.

--sa se afizeze la cate masini a contribuit piesa furnizata

create or replace function nr\_contributii(idul distribuitor.id\_furnizor%type,categ autovehicul.id\_categorie%type)

return number is

total number :=0;

no\_category EXCEPTION;

PRAGMA exception\_init(no\_category, -20001);

no\_furnizor EXCEPTION;

PRAGMA exception\_init(no\_furnizor, -20002);

type tabel is table of autovehicul.id\_autovehicul%type index by pls\_integer;

type tabel2 is table of autovehicul%rowtype index by pls\_integer;

iduri\_autovehicule tabel;

autouri tabel2;

begin

select count(\*) into total from categorie where id\_categorie=categ;

if total=0 then raise\_application\_error(-20001, '');

end if;

select count(\*) into total

from distribuitor

where id\_furnizor=idul;

if total=0 then raise\_application\_error(-20002, '');

end if;

select distinct n.id\_autovehicul

bulk collect into iduri\_autovehicule -- idurile masinilor care au primit piese de la furnizor

from piese p join distribuitor d on p.id\_furnizor=d.id\_furnizor

join necesita n on n.id\_piesa=p.id\_piesa

where p.id\_furnizor=idul;

select \* bulk collect into autouri

from autovehicul a

where a.id\_categorie=categ;

for i in autouri.first..autouri.last loop

for j in iduri\_autovehicule.first..iduri\_autovehicule.last loop

if iduri\_autovehicule(j)=autouri(i).id\_autovehicul then total:=total+1;

end if;

end loop;

end loop;

return total;

exception

when no\_category then

dbms\_output.put\_line('Nu exista categoria');

return -1;

when no\_furnizor then

dbms\_output.put\_line('Nu exista furnizorul');

return -1;

end;

/

declare

id\_cat number;

id\_dist number;

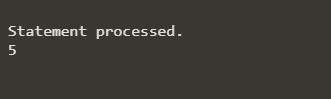
numar\_total number;

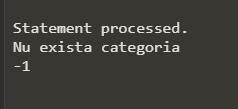
begin

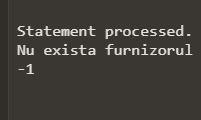
numar\_total := nr\_contributii(1000,300);

dbms\_output.put\_line(numar\_total);

end;







--Cerinta 9

--Pentru numele dat, sa se afiseze autovehiculele din categoria respectiva adusa pe parcusrul anului curent.

create or replace procedure cerinta\_noua(nume\_client client.nume%type,

nume\_categ categorie.nume\_categorie%type)

is

cursor c1 (nc client.nume%type,

ng categorie.nume\_categorie%type) is

select a.nume

from client cl join reparatii re on cl.id\_client=re.id\_client

join repara rep on rep.id\_comanda = re.id\_comanda

join autovehicul a on a.id\_autovehicul=rep.id\_autovehicul

join categorie c on a.id\_categorie=c.id\_categorie

where c.nume\_categorie=ng and

EXTRACT( year from re.data\_comanda) = EXTRACT(YEAR FROM sysdate)

and cl.nume=nc;

contor number :=0;

begin

for i in c1(nume\_client,nume\_categ) loop

dbms\_output.put\_line(i.nume);

end loop;

if contor =0 then

dbms\_output.put\_line('Nu s-a gasit nimic in baza de date');

end if;

end cerinta\_noua;

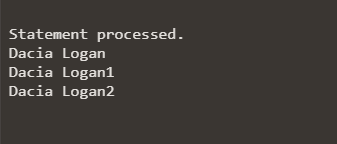
/

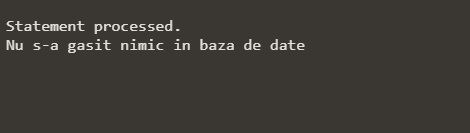
declare

begin

cerinta\_noua('Pomparau','Masini');

end;





--Cerinta 10

--Trigger care nu permite adaugarea, stergerea si modificarea reparatiilor in afara programului.

--Program: 8:00-17:00, iar in weekend intre 10:00-14:00.

CREATE OR REPLACE TRIGGER program\_atelierauto

BEFORE INSERT OR UPDATE OR DELETE ON reparatii

BEGIN

IF TO\_CHAR(SYSDATE, 'DAY') = 'SATURDAY' OR TO\_CHAR(SYSDATE, 'DAY') = 'SUNDAY' THEN

IF TO\_NUMBER(TO\_CHAR(SYSDATE, 'HH24')) < 10 THEN

RAISE\_APPLICATION\_ERROR(-20010, 'Program D-S intre orele 0:00-14:00');

END IF;

IF TO\_NUMBER(TO\_CHAR(SYSDATE, 'HH24')) > 14 THEN

RAISE\_APPLICATION\_ERROR(-20011, 'Program D-S intre orele 0:00-14:00');

END IF;

ELSE

IF TO\_NUMBER(TO\_CHAR(SYSDATE, 'HH24')) < 8 THEN

RAISE\_APPLICATION\_ERROR(-20012, 'Program L-V intre orele 8:00-17:00');

END IF;

IF TO\_NUMBER(TO\_CHAR(SYSDATE, 'HH24')) > 17 THEN

RAISE\_APPLICATION\_ERROR(-20013, 'Program L-V intre orele 8:00-17:00');

END IF;

END IF;

END;

/

Declare

BEGIN

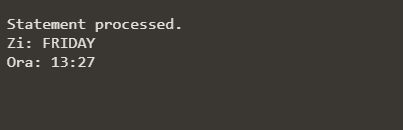
DBMS\_OUTPUT.PUT\_LINE('Zi: ' || TO\_CHAR(SYSDATE, 'DAY'));

DBMS\_OUTPUT.PUT\_LINE('Ora: ' || TO\_CHAR(SYSDATE, 'HH24:MI'));

INSERT INTO REPARATII

VALUES(SEQ\_REPARATII.NEXTVAL, to\_date('09-03-2021','dd-mm-yyyy'), 2, 5);

END;



--Cerinta 11:

--Pentru a preda o comanda, trebui sa verificam daca a fost finalizata comanda.

-- se afiseaza daca e in curs de lucru sau daca inca nu a fost alocat un angajat pentru comanda.

select \* from repara where data\_finalizare is null;

create or replace trigger finalizare\_comanda

before delete

on repara

for each row

declare

pragma autonomous\_transaction;

v\_data repara.data\_finalizare%type;

begin

if deleting then

select data\_finalizare into v\_data

from repara

where id\_angajat=:old.id\_angajat and id\_comanda=:old.id\_comanda and id\_autovehicul=:old.id\_autovehicul;

if v\_data is null then DBMS\_OUTPUT.PUT\_LINE('Reparatia nu a fost inca finalizata');

elsif v\_data<sysdate then

DBMS\_OUTPUT.PUT\_LINE('Reparatia a fost finalizara in data ' || v\_data ||' si a fost stearsa cu succes');

else DBMS\_OUTPUT.PUT\_LINE('Nimic');

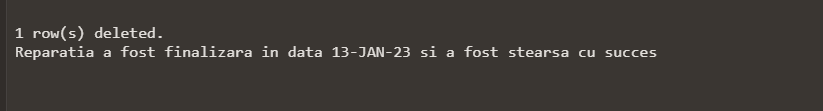
end if;

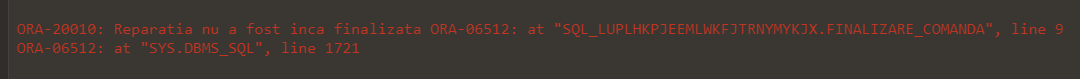
end if;

end;

delete from repara;

alter trigger finalizare\_comanda enable;





--Cerinta 12

--APEX\_PUBLIC\_USER  
  
CREATE OR REPLACE TRIGGER adminul  
    BEFORE CREATE OR DROP OR ALTER ON SCHEMA  
BEGIN  
    IF USER != UPPER('APEX\_PUBLIC\_USER') THEN  
        RAISE\_APPLICATION\_ERROR(-20010,'Doar APEX\_PUBLIC\_USER poate executa comenzi LDD!');  
    END IF;  
END;  
/  
create table try(  
tries varchar2(10)  
);  
drop table try;

--Cerinta 13

create or replace package pachetul as

procedure vehicule\_categorie (Categorie IN autovehicul.id\_categorie%TYPE);

procedure angajati\_masini (id\_emp IN angajat.id\_angajat%type);

function nr\_contributii(idul distribuitor.id\_furnizor%type,categ autovehicul.id\_categorie%type)

return number ;

procedure cerinta\_noua(nume\_client client.nume%type,

nume\_categ categorie.nume\_categorie%type) ;

end pachetul;

create or replace package body pachetul as

--9

procedure cerinta\_noua(nume\_client client.nume%type,

nume\_categ categorie.nume\_categorie%type)

is

cursor c1 (nc client.nume%type,

ng categorie.nume\_categorie%type) is

select a.nume

from client cl join reparatii re on cl.id\_client=re.id\_client

join repara rep on rep.id\_comanda = re.id\_comanda

join autovehicul a on a.id\_autovehicul=rep.id\_autovehicul

join categorie c on a.id\_categorie=c.id\_categorie

where c.nume\_categorie=ng and

EXTRACT( year from re.data\_comanda) = EXTRACT(YEAR FROM sysdate)

and cl.nume=nc;

contor number :=0;

begin

for i in c1(nume\_client,nume\_categ) loop

dbms\_output.put\_line(i.nume);

end loop;

if contor =0 then

dbms\_output.put\_line('Nu s-a gasit nimic in baza de date');

end if;

end cerinta\_noua;

--8

function nr\_contributii(idul distribuitor.id\_furnizor%type,categ autovehicul.id\_categorie%type)

return number is

total number :=0;

no\_category EXCEPTION;

PRAGMA exception\_init(no\_category, -20001);

no\_furnizor EXCEPTION;

PRAGMA exception\_init(no\_furnizor, -20002);

type tabel is table of autovehicul.id\_autovehicul%type index by pls\_integer;

type tabel2 is table of autovehicul%rowtype index by pls\_integer;

iduri\_autovehicule tabel;

autouri tabel2;

begin

select count(\*) into total from categorie where id\_categorie=categ;

if total=0 then raise\_application\_error(-20001, '');

end if;

select count(\*) into total

from distribuitor

where id\_furnizor=idul;

if total=0 then raise\_application\_error(-20002, '');

end if;

select distinct n.id\_autovehicul

bulk collect into iduri\_autovehicule -- idurile masinilor care au primit piese de la furnizor

from piese p join distribuitor d on p.id\_furnizor=d.id\_furnizor

join necesita n on n.id\_piesa=p.id\_piesa

where p.id\_furnizor=idul;

select \* bulk collect into autouri

from autovehicul a

where a.id\_categorie=categ;

for i in autouri.first..autouri.last loop

for j in iduri\_autovehicule.first..iduri\_autovehicule.last loop

if iduri\_autovehicule(j)=autouri(i).id\_autovehicul then total:=total+1;

end if;

end loop;

end loop;

return total;

exception

when no\_category then

dbms\_output.put\_line('Nu exista categoria');

return -1;

when no\_furnizor then

dbms\_output.put\_line('Nu exista furnizorul');

return -1;

end;

--7

procedure angajati\_masini

(id\_emp IN angajat.id\_angajat%type)

as

nume\_piesa piese.denumire%type;

c2\_variable necesita.id\_piesa%type;

c1\_variable autovehicul.id\_autovehicul%type;

cursor c1 is

select id\_autovehicul

from repara

where id\_angajat=id\_emp;

cursor c2(id\_masina necesita.id\_autovehicul%type) is

select id\_piesa

from necesita

where id\_autovehicul=id\_masina;

employee\_exist NUMBER;

masina\_curenta autovehicul.nume%type;

begin

SELECT id\_angajat INTO employee\_exist

FROM angajat WHERE id\_angajat = id\_emp;

open c1;

loop fetch c1 into c1\_variable;

exit when c1%notfound;

select nume into masina\_curenta from autovehicul where id\_autovehicul=c1\_variable;

dbms\_output.put(masina\_curenta ||' are nevoie de urmatoarele piese: ');

open c2(c1\_variable);

loop fetch c2 into c2\_variable;

exit when c2%notfound;

select denumire into nume\_piesa from piese where id\_piesa= c2\_variable;

dbms\_output.put(nume\_piesa ||' ');

end loop;

dbms\_output.put\_line('');

close c2;

end loop;

close c1;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista angajatul');

--WHEN OTHERS THEN

--DBMS\_OUTPUT.PUT\_LINE('An error occurred');

end;

--6

procedure vehicule\_categorie

(Categorie IN autovehicul.id\_categorie%TYPE)

as

--Categorie varchar(8) := '&categorie';

TYPE vector\_masini IS TABLE OF autovehicul%rowtype index by binary\_integer;

masini vector\_masini;

TYPE nume\_numar IS RECORD(

nume\_masina autovehicul.nume%TYPE,

numar\_piese number

);

TYPE vector\_masini\_piese IS VARRAY(20)

OF nume\_numar;

vec1 vector\_masini\_piese;

BEGIN

select \* bulk collect into masini from autovehicul where id\_categorie=Categorie;

select a.nume,count(\*)

bulk collect into vec1

from autovehicul a,necesita n

where a.id\_autovehicul = n.id\_autovehicul

group by a.nume;

FOR i in masini.first..masini.last loop

FOR j in vec1.first..vec1.last loop

if masini(i).nume=vec1(j).nume\_masina then

DBMS\_OUTPUT.PUT\_LINE(vec1(j).nume\_masina || ' necesita o schimbare de '||vec1(j).numar\_piese ||' piese.');

end if;

end loop;

end loop;

end;

end pachetul;

