



UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Facultatea de Automatică și Calculatoare

Departamentul de Calculatoare

Disciplina Baze de Date

Anul 2021-2022

Online Shop

Data: 11.01.2022

Echipa de proiect: Mirițoiu Cristian

Alexa Andrei Ștefan

Cuprins

Introducere:	2
Specificatiile de proiect:.....	2
Modelul de date și descrierea acestuia	3
Tabele:.....	3
Normalizarea datelor	6
Diagrama EER.....	7
Detalii de implementare	8
MySQL	8
Interogari	9
Proceduri stocate:.....	14
Funcții.....	19
Trigger-e.....	19
View-uri.....	21
Utilizare Java	24
Instalare și utilizare	24

Introducere:

Acest proiect reprezinta o aplicatie pentru cumparaturi online prin intermediul careia se pot achizitiona orice fel de produse necesare.

Proiectul a fost creat cu scopul de a proiecta o interfata grafica care sa fie accesibila cumparatorilor si sa poata fi accesata de pe orice laptop, calculator personal

Acest online shop este un site foarte popular in zilele noastre, avand in vedere conditiile pandemice in care traim. Aplicatia trebuie sa fie simplu si usor de utilizat, trebuie sa ofere utilizatorilor posibilitatea de a se loga pe un cont personal, de a accesa o lista cu produsele puse la dispozitie si de a avea access la review-urile lasate de clienti anteriori pentru a stii la ce sa se astepte.

Aplicatia trebuie sa fie simplu si usor de utilizat , trebuie sa ofere utilizatorilor posibilitatea de a se loga pe un cont personal.

Specificatiile de proiect:

Tema proiectulu este un magazine online de diferite produse necesare in viata de zi cu zi. Gama de produse cuprinde tot ce este necesar in viata de zi cu zi.

Proiectul consta intr-o baza de date implementata in MySQL Workbench si o interfata scrisa in Java care ofera utilizatorilor posibilitatea de a achizitiona rapid produsele care se regasesc in stocul magazinului.

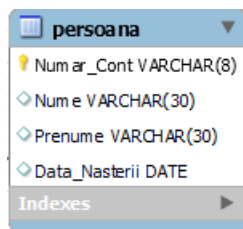
Baza de date a fost optimizata pentru a ocupa cat mai putina memorie si pentru a putea fi actualizata usor. Aproape toate operatiile bazei pot fi efectuate cu ajutorul celor 1 functii si 8 de proceduri stocate, acesta fiind unul dintre principalele obiective ale acestui proiect. De asemenea pe langa acestea in cadrul bazei de date mai exista 2 trigger-e, 4 view-uri si 20 selectii a datelor.

Modelul de date si descrierea acestuia

Tabele:

Tabela “Persoana”:

-contine informatii despre persoana care comanda un produs

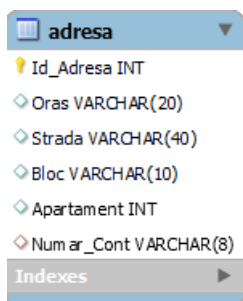


persoana	
Numar_Con	VARCHAR(8)
Nume	VARCHAR(30)
Prenume	VARCHAR(30)
Data_Nasterii	DATE

- Numar_Con – cheie primara
- Nume – numele persoanei
- Prenume – prenumele persoanei
- Data_Nasterii – data de nastere a persoanei

Tabela “Adresa”:

-contine informatii despre destinatia comenzii



adresa	
Id_Adresa	INT
Oras	VARCHAR(20)
Strada	VARCHAR(40)
Bloc	VARCHAR(10)
Apartament	INT
Numar_Con	VARCHAR(8)

- Id_Adresa – cheie primara
- Oras, strada, bloc, apartament – adresa comenzii
- Numar_Con – cheie straina din tabela “Persoana”

Tabela “Angajat”

-contine informatii despre angajati

angajat
Id_Angajat INT
Salariu DECIMAL(6,2)
Deducere DECIMAL(6,2)
Numar_Cont VARCHAR(8)
Brut DECIMAL(6,2)
Indexes

- Id_Angajat - cheie primara
- Salariu – salariul angajatului
- Deducere –
- Numar_Cont – cheie straina din tabela “Persoana”
- Brut – salariul brut al angajatului

Tabela “Client”

-contine informatii despre contul utilizatorului

client
Id_Persoana INT
Username VARCHAR(30)
Parola VARCHAR(20)
Numar_Cont VARCHAR(8)
Indexes

- Id_Persoana - cheie primara
- Username - numele contului – unique
- Parola - parola contului
- Numar_Cont – cheie straina din tabela “Persoana”

Tabela “Furnizor”

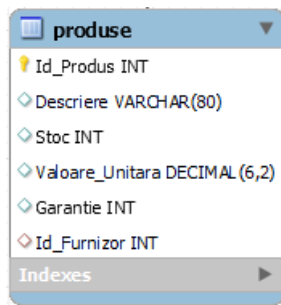
-contine informatii despre furnizorul produselor

furnizor
Id_Furnizor INT
Nume VARCHAR(30)
Tara VARCHAR(20)
Indexes

- Id_Furnizor – cheie primara
- Nume – numele furnizorului
- Tara – tara din care provin produsele

Tabela “Produse”

-contine informatii despre produsele vandute

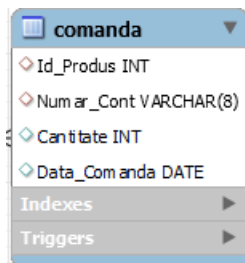


produse	
Id_Produs	INT
Descriere	VARCHAR(80)
Stoc	INT
Valoare_Unitara	DECIMAL(6,2)
Garantie	INT
Id_Furnizor	INT
Indexes	

- Id_Produs - cheie primara
- Descriere – proprietati ale produsului
- Stoc – stocul produsului
- Valoare_Unitara – pretul unui singur produs
- Garantie – garantia oferita asupra produsului
- Id_furnizor – cheie straina din tabela "Furnizor"

Tabela "Comanda"

-contine informatii despre comanda initializata

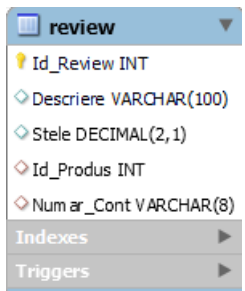


comanda	
Id_Produs	INT
Numar_Cont	VARCHAR(8)
Cantitate	INT
Data_Comanda	DATE
Indexes	
Triggers	

- Id_Comanda – cheie primara
- Id_Produs – cheie straina din tabela "Produse"
- Numar_Cont – cheie straina din tabela "Persoana"
- Cantitate – cantitatea de produse comandate
- Data_Comanda – data la care a fost initializata comanda

Tabela "Review"

-contine parerea clientilor asupra produselor comandate



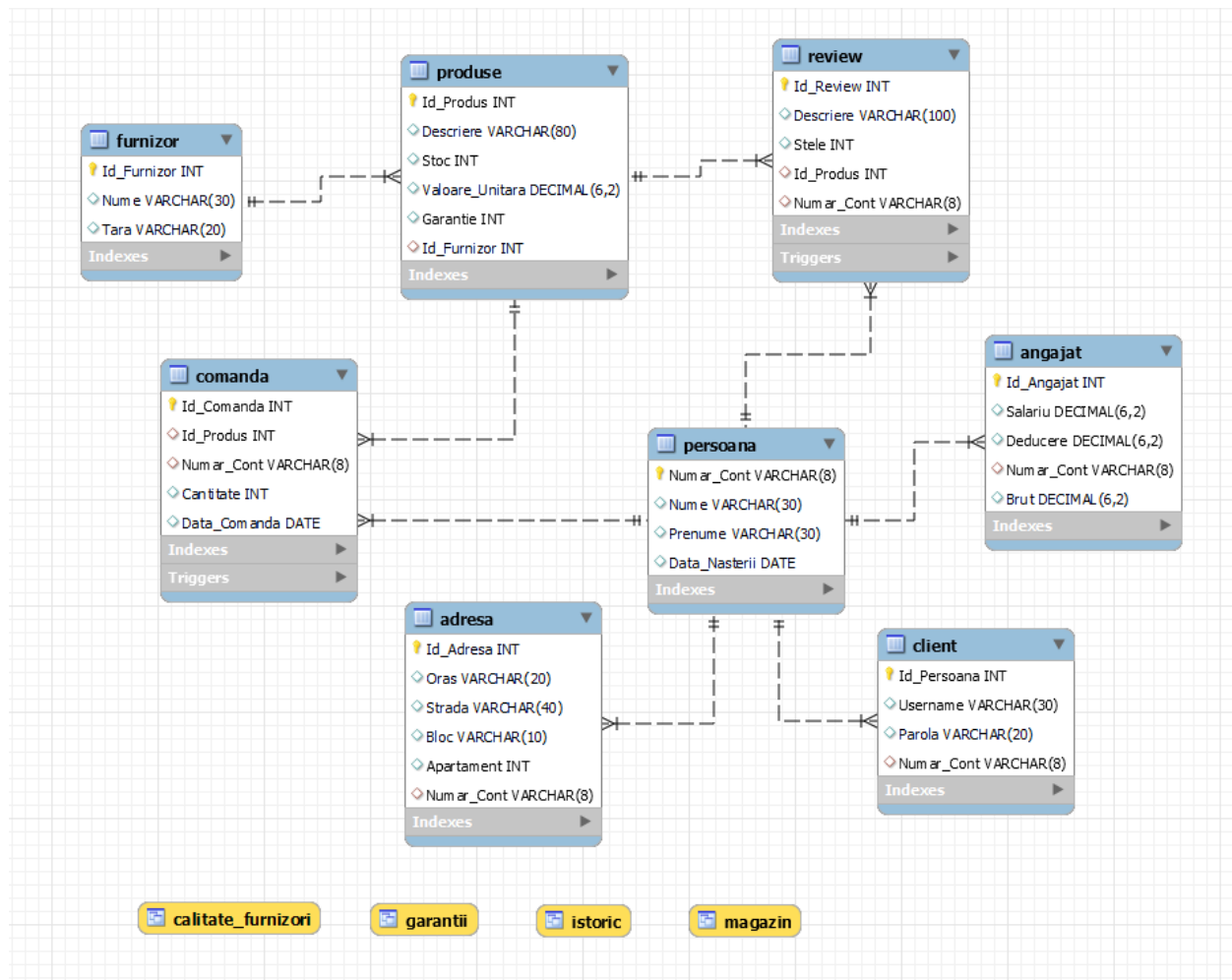
review	
Id_Review	INT
Descriere	VARCHAR(100)
Stele	DECIMAL(2,1)
Id_Produs	INT
Numar_Cont	VARCHAR(8)
Indexes	
Triggers	

- Id_Review – cheie primara
- Descriere – opinia clientului in legatura cu produsul
- Stele –
- Id_Produs – cheie straina din tabela "Produs"
- Numar_Cont – cheie straina din tabela "Persoana"

Normalizarea datelor

Baza de date prezentata in acest proiect face parte din forma normala Boyce-Codd deoarece in cadrul fiecarui tabel avem o cheie primara de care depind toate attributele, iar fiecare cheie prima este simpla. Astfel este respectata definitia formei normale Boyce-Codd: O schema de relatie este in FNBC daca si numai daca fiecare dependenta are in partea stanga o supercheie.

Diagrama EER



Detalii de implementare

MySQL

MySQL este un sistem de gestiune a bazelor de date relationale, produs de compania suedeza MySQL AB si distribuit sub Licenta Publica Generala GNU. Este cel mai popular SGBD open-source la ora actuala, fiind o componenta cheie a stivei LAMP (Linux, Apache, MySQL, PHP).

Desi este folosit foarte des impreuna cu limbajul de programare PHP, cu MySQL se pot construi aplicatii in orice limbaj major. Exista multe scheme API disponibile pentru MySQL ce permit scrierea aplicatiilor in numeroase limbaje de programare pentru accesarea bazelor de date MySQL, cum are fi: C, C++, C#, Java, Perl, PHP, Python, FreeBasic, etc., fiecare dintre acestea folosind un tip specific API. O interfata de tip ODBC denumita MyODBC permite altor limbaje de programare ce folosesc aceasta interfata, sa interactioneze cu bazele de date MySQL cum ar fi ASP sau Visual Basic. In sprijinul acestor limbaje de programare, unele companii produc componente de tip COM/COM+ sau .NET (pentru Windows) prin intermediul carora respectivele limbaje sa poata folosi acest SGBD mult mai usor decat prin intermediul sistemului ODBC. Aceste componente pot fi gratuite (ca de exemplu MyVBQL) sau comerciale.

Licenta GNU GPL nu permite incorporarea MySQL in soft-uri comerciale; cei care doresc sa faca acest lucru pot achizitiona, contra cost, o licenta comerciala de la compania producatoare, MySQL AB.

MySQL este componenta integrata a platformelor LAMP sau WAMP (Linux/Windows-Apache-MySQL-PHP/Perl/Python). Popularitatea sa ca aplicatie web este strans legata de cea a PHP-ului care este adesea combinat cu MySQL si denumit Duo-ul Dinamic. In multe carti de specialitate este precizat faptul ca MySQL este mult mai usor de invatat si folosit decat multe din aplicatiile de gestiune a bazelor de date, ca exemplu comanda de iesire fiind una simpla si evidenta: „exit” sau „quit”.

Pentru a administra bazele de date MySQL se poate folosi modul linie de comanda sau, prin descarcare de pe internet, o interfata grafica MySQL Administrator si MySQL Query Browser. Un alt instrument de management al acestor baze de date este aplicatia gratuita, scrisa in PHP, phpMyAdmin.

MySQL poate fi rulat pe multe dintre platformele software existente: AIX, FreeBSD, GNU/Linux, Mac OS X, NetBSD, Solaris, SunOS, Windows 9x/NT/2000/XP/Vista.

Interogari

In continuare vor fi prezentate cateva interogari utile pentru operarea pe baze de date din cadrul proiectului:

#Pentru afisarea celui mai vandut produs

SELECT

Descriere as produs, MAX(vanzari) as vanzari

FROM

(SELECT

produse.Descriere, COUNT(*) AS vanzari

FROM

produse

JOIN comanda ON comanda.Id_Produs = produse.Id_Produs

GROUP BY produse.Descriere) AS x;

#Pentru afisarea datei în care au avut loc cele mai multe vânzări;

SELECT

Data_comanda, MAX(vanzari) AS vanzari

FROM

(SELECT

comanda.Data_Comanda, COUNT(*) AS vanzari

FROM

comanda

GROUP BY comanda.Data_Comanda) AS x;

#Pentru a se afisa clientul (nume,prenume) care a cumpărat cele mai multe produse (și valoarea totală a cumparaturilor).

```
select Prenume,Nume,max(produse) as produse from(  
SELECT count(comanda.Numar_Cont) as produse ,persoana.Nume,persoana.Prenume FROM  
persoana  
join comanda on comanda.Numar_Cont=persoana.Numar_Cont  
GROUP BY comanda.Numar_Cont)as x;
```

#Pentru a se afisa toate produsele fabricate in canada

```
Select produse.Descriere,furnizor.Nume from produse join furnizor on  
furnizor.Id_Furnizor=produse.Id_Furnizor  
where furnizor.Tara="Canada";
```

#Pentru a se afisa numele angajatiilor cu deducere intre 500 si 800

```
select persoana.Nume,persoana.Prenume ,angajat.Deducere from angajat JOIN persoana  
on persoana.Numar_Cont=angajat.Numar_Cont  
where angajat.Deducere between 500 and 800  
order by Nume,Prenume;
```

#Pentru a se afisa numele produselor mai scumpe decat 50 lei

```
select produse.Descriere ,produse.Valoare_Unitara from  
produse where produse.Valoare_Unitara>=50  
order by Valoare_Unitara;
```

#Pentru afisarea clientilor care locuiesc in Nampa

```
select client.Username from client join persoana on persoana.Numar_Cont=client.Numar_Cont  
join adresa on adresa.Numar_Cont=client.Numar_Cont  
and adresa.Oras='Nampa';
```

#Pentru afisarea produselor livrate in orasul Kearney

```
select produse.Descriere from produse join comanda on comanda.Id_Produs=  
produse.Id_Produs join  
adresa on adresa.Numar_Cont=comanda.Numar_Cont  
where adresa.Oras='Kearney';
```

#Pentru a se afisa cate produse au fost livrate din Indonesia

```
select sum(produse.Stoc) as produse_livrate FROM produse JOIN furnizor on  
furnizor.Id_Furnizor=produse.Id_Furnizor  
where furnizor.Tara="Indonesia";
```

#Pentru a se afisa clientii care au dat reviews de 5 stele;

```
select distinct client.Username from client join review on  
review.Numar_Cont=client.Numar_Cont  
where review.Stele=5;
```

#Pentru a se afisa produsele cu o medie de reviews mai mica decat 3

```
select produse.Descriere,avg(review.stele) as medie_reviews from produse join review on  
review.Id_Produs=produse.Id_Produs  
  
group by produse.Descriere  
  
having medie_reviews<3;
```

#Pentru selectarea persoanelor si a produselor cumparate de acestia pentru care garantia este inca valabila

```
select persoana.Nume,persoana.Prenume,produse.Descriere,expira(comanda.Id_Comanda) as  
Data_termen  
  
from produse join comanda on comanda.Id_Produs=produse.Id_Produs join persoana on  
persoana.Numar_Cont=comanda.Numar_Cont  
  
HAVING Data_termen>curdate();
```

#Pentru afisare persoanelor care au cumparat de la furnizori cu o calitate/reviews de minim 4 stele

```
select persoana.Nume,persoana.Prenume from persoana join comanda on  
comanda.Numar_Cont=persoana.Numar_Cont join  
  
produse on produse.Id_Produs=comanda.Id_Produs join calitate_furnizori on  
produse.Descriere=calitate_furnizori.produs  
  
where calitate_furnizori.calitate>=4;
```

#Pentru afisarea produselor pentru care avem un stoc de minim 100 dar un maxim de 150;

```
select produse.Descriere,produse.Stoc from produse where produse.Stoc between 100 and 150;
```

#Pentru afisarea angajatilor care locuiesc in College

```
select persoana.Nume,persoana.Prenume from angajat join adresa on  
adresa.Numar_Cont=angajat.Numar_Cont  
join persoana on persoana.Numar_Cont=angajat.Numar_Cont  
where adresa.Oras="College";
```

#Pentru afisarea clientilor care nu sunt majori

```
select client.Username from client join persoana on persoana.Numar_Cont=client.Numar_Cont  
where datediff(persoana.Data_Nasterii,curDate())<18;
```

#Pentru afisarea adresei unde s-au livrat produse din Canada

```
select adresa.Oras,adresa.Strada from adresa join comanda on  
adresa.Numar_Cont=comanda.Numar_Cont join produse  
on produse.Id_Produs=comanda.Id_Produs join furnizor on  
furnizor.Id_Furnizor=produse.Id_Furnizor where  
furnizor.Tara="Canada";
```

#Pentru afisarea produselor cu furnizor Fivechat

```
select produse.Descriere from produse join furnizor on furnizor.Id_Furnizor=produse.Id_Furnizor  
where furnizor.Nume="Fivechat";
```

#Pentru afisarea clientilor a caror parola incepe cu litera M

```
select client.Username from client where client.Parola like 'M%';
```

#Pentru afisarea persoanelor care au cumparat produse cu reviews intre 2 si 4

```
select distinct persoana.Nume,persoana.Prenume from persoana join comanda on  
comanda.Numar_Cont=persoana.Numar_Cont join produse on produse.Id_Produs  
=comanda.Id_Produs join review on review.Id_Produs=comanda.Id_Produs  
where review.Stele between 2 and 4;
```

Proceduri stocate:

/*Insereaza datele necesare, o noua instanta de date pentru tabelul comanda/realizarea unei comenzi*/

```
drop PROCEDURE if EXISTS comandare;
```

```
DELIMITER $$
```

```
Create PROCEDURE comandare(Numar_Cont VARCHAR(8),Id_Produs int,cantitate int)
```

```
begin
```

```
Insert into Comanda(Id_Produs, Numar_Cont, Cantitate,Data_Comanda)
```

```
VALUES
```

```
(Id_Produs,Numar_Cont,cantitate, CURRENT_DATE);
```

```
end $$
```

```
delimiter ;
```

/*efectueaza 3 comenzi diferite de catre aceeași persoana*/

```
drop PROCEDURE if EXISTS cumparamult;
```

```
Delimiter //
```

```
Create PROCEDURE cumparamult(Cont varchar(8),Produs1 int,cantitate1 int ,Produs2  
int,cantitate2 int,Produs3 int,cantitate3 int)
```

```
BEGIN
```

```
        call comandare(Cont,Produs1,cantitate1,current_date);
```

```
        call comandare(Cont,Produs2,cantitate2,current_date);
```

```
        call comandare(Cont,Produs3,cantitate3,current_date);
```

```
END//
```

```
Delimiter ;
```

```
/*Calculeaza salariul brut al unui angajat avand in vedere deducerea*/
```

```
drop PROCEDURE if EXISTS salarii;
```

```
DELIMITER $$
```

```
Create PROCEDURE salarii()
```

```
begin
```

```
        Declare maxim int;
```

```
        Declare i int default 1;
```

```
        select max(id_angajat)
```

```
        from angajat into maxim;
```

```
        while i<=maxim do
```

```
        UPDATE angajat
```

```
        set brut=salariu+deducere where id_angajat=i;
```

```
        set i=i+1;
```

```
        end while;
```

```
end $$
```

```
delimiter ;
```

*/*creeaza un cont utilizator*/*

drop PROCEDURE if EXISTS create_user;

DELIMITER \$\$

Create PROCEDURE create_user(Numecont VARCHAR(30),Parola2 VARCHAR(20),Id VARCHAR(8),
Data_Nasterii2 DATE,Nume2 VARCHAR(30),Prenume2 VARCHAR(30))

begin

INSERT INTO persoana(Numar_Cont,Nume,Prenume,Data_Nasterii)

VALUES

(Id,Nume2,Prenume2, Data_Nasterii2);

Insert into Client(Numar_Cont,Username,Parola)

VALUES

(Id,Numecont,Parola2);

end \$\$

delimiter ;

*/*creeaza un review de catre un cont*/*

drop PROCEDURE if EXISTS create_review;

DELIMITER \$\$

Create PROCEDURE create_review(Id_Produs int,Stele DECIMAL(6,2),Numar_Cont
VARCHAR(8),Descriere varchar(100))

begin

Insert into Review(Id_Produs,Stele,Numar_Cont,Descriere)

VALUES

(Id_Produs,Stele,Numar_Cont,Descriere);

end \$\$

delimiter ;

*/*pentru a creea un cont de angajat*/*


```
drop PROCEDURE if EXISTS create_angajat;
```

```
DELIMITER $$
```

```
Create PROCEDURE create_angajat(Pay int,User_name VARCHAR(30),Parola2  
VARCHAR(20),NumarCont VARCHAR(8), Data_Nasterii2 DATE,Num2 VARCHAR(30),Prenume2  
VARCHAR(30))
```

```
begin
```

```
INSERT INTO persoana(Numar_Cont,Num2,Prenume,Data_Nasterii)
```

```
VALUES
```

```
(NumarCont,Num2,Prenume2, Data_Nasterii2);
```

```
Insert into Client(Numar_Cont,Username,Parola)
```

```
VALUES
```

```
(NumarCont,User_name,Parola2);
```

```
INSERT into angajat(Numar_Cont,salariu)
```

```
values(NumarCont,Pay);
```

```
end $$
```

```
delimiter ;
```

#Adauga in tabelul produse un stoc nou pentru produs

```
DROP Procedure if exists comanda_stoc
```

```
DELIMITER //
```

```
create procedure comanda_stoc(Id int,Produs varchar(30),stoc int)
```

```
begin
```

```
update produse
```

```
set produse.stoc=produse.stoc+stoc where produse.Descriere=Produs or  
produse.Id_Produs=Id;
```

```
end //
```

```
DELIMITER;
```

#sterge un cont de client

Drop procedure if exists stergecont

DELIMITER //

Create procedure stergecont(Nume varchar(30),Prenume varchar(30))

BEGIN

Declare Id varchar(8);

SELECT

persoana.Numar_Cont

FROM persoana

WHERE persoana.Nume = Nume

AND persoana.Prenume = Prenume INTO Id;

DELETE FROM client

WHERE

client.Numar_Cont = Id;

end

// DELIMITER ;

#sterge un angajat din tabela de angajati in functie de nume

DROP procedure if exists concediere

DELIMITER \$\$

CREATE procedure concediere(numa varchar(30),prenume varchar(30))

begin

Declare Id varchar(8);

SELECT

persoana.Numar_Cont

FROM

persoana

WHERE

persoana.nume = nume

AND persoana.prenume = prenume INTO Id;

DELETE FROM angajat

WHERE

angajat.Numar_Cont = Id;

end \$\$

DELIMITER ;

Functii

/*Returneaza data la care va expira garantia*/

drop FUNCTION if EXISTS expira;

DELIMITER \$\$

CREATE FUNCTION expira(idComanda int)

returns Date

DETERMINISTIC

BEGIN

Declare data_Expirare date;

select comanda.Data_Comanda from comanda WHERE comanda.Id_Comanda=idComanda

into data_Expirare;

set data_Expirare=date_add(data_Expirare,Interval 5 year);

return data_Expirare;

end \$\$

DELIMITER ;

Trigger-e

/*dupa efectuarea unei comenzi, din stocul magazinului se va extrage cantitatea comandata de catre o persoana*/

Delimiter //

create TRIGGER cumpar AFTER INSERT on comanda

FOR EACH ROW

BEGIN

DECLARE Id INT;

DECLARE Cantitate1 int;

DECLARE NewComanda INT;

SELECT max(new.Id_Comanda)

from Comanda INTO NewComanda;

select Id_Produs from comanda

where Id_Comanda=NewComanda into Id;

select cantitate

from comanda

where Id_Comanda=NewComanda

into Cantitate1;

UPDATE produse

set stoc=stoc-Cantitate1

where Id_Produs=Id;

END//

Delimiter ;

/*calculeaza deducerea unui angajat dupa ce acesta scrie un review*/

```
Delimiter //
```

```
create TRIGGER bonus after insert on review
```

```
FOR EACH ROW
```

```
BEGIN
```

```
            DECLARE maxim int;
```

```
    DECLARE i int DEFAULT 1;
```

```
    Declare cont varchar(8);
```

```
    SELECT max(Id_Review)
```

```
    from review into maxim;
```

```
    select Numar_Cont from review WHERE Id_Review=maxim into cont;
```

```
    UPDATE angajat
```

```
    set Deducere=Deducere+20 where Numar_Cont=cont;
```

```
End //
```

```
Delimiter ;
```

View-uri

#istoricul comenzilor

```
CREATE OR REPLACE VIEW istoric AS
```

```
SELECT
```

```
    nume,
```

```
    prenume,
```

```
    descriere,
```

```
    cantitate,
```

```
    cantitate * valoare_unitara AS ValoareTotala
```

```
FROM
```

```
    persoana AS p,
```

```
comanda AS c,  
produse AS o  
WHERE  
p.Numar_Cont = c.Numar_Cont  
AND o.Id_Produs = c.Id_Produs;
```

#Produsul caruia i-a expirat garantia

```
CREATE OR REPLACE VIEW garantii AS  
SELECT  
    descriere, EXPIRA(c.Id_Comanda) AS data_expirarii  
FROM  
    produse AS p,  
    comanda AS c  
WHERE  
    p.Id_Produs = c.Id_Produs  
HAVING data_expirarii < CURDATE();
```

#Media review-urilor pentru fiecare produs si furnizorul.

```
CREATE OR REPLACE VIEW Calitate_furnizori AS  
SELECT  
    f.Id_Furnizor AS Id,  
    f.nume AS Companie,  
    p.descriere AS produs,  
    AVG(r.stele) AS Calitate  
FROM  
    review AS r,  
    produse AS p,
```

```
furnizor AS f
WHERE
    r.Id_Produs = p.Id_Produs
    AND f.Id_Furnizor = p.Id_Furnizor
GROUP BY p.descriere
ORDER BY id;
```

#Utilizat pentru interfata

```
CREATE OR REPLACE VIEW Magazin AS
SELECT
    produse.Descriere AS Produs,
    produse.Valoare_Unitara AS Pret,
    furnizor.Nume AS Companie,
    CAST(AVG(review.Stele) AS DECIMAL (3 , 2 )) AS Stele
FROM
    produse
    JOIN
    furnizor ON produse.Id_Furnizor = furnizor.Id_Furnizor
    LEFT JOIN
    review ON produse.Id_Produs = review.Id_Produs
GROUP BY produse.Descriere
ORDER BY Produs;
```

Utilizare Java

Pentru a folosi limbajul java in relatie cu o baza de date se foloseste tehnologia JDBC. Pentru a asigura conexiunea si accesul la comenzile necesare pentru a realiza o conexiune se utilizeaza [mysql-connector-java-8.0.22.jar](https://dev.mysql.com/doc/connector-java/8.0.22/jar) pus la dispozitie in laborator. Acest document se insereaza in path-ul unui nou java document ->Proprieties->Java build path->Libraries.

Pentru conectarea la baza de date se realizeaza o conexiune noua astfel

```
Connection connect=DriverManager.getConnection(servername, user, password);
```

Pentru a verifica daca conexiunea a avut loc vom scrie

```
if (conect == null) {  
    System.out.println("JDBC connection is not established");  
    return;}  
}
```

Conectiunea stabilita ne ofera accesul la noi comenzi din interiorul java pentru a executa query si a utiliza proceduri stocate;

Pentru a executa un query vom scrie astfel:

```
Statement stm=conect.createStatement();
```

```
ResultSet rez=stm.executeQuery("query");
```

Rez va returna rezultatul selectiei si se va itera prin acestea utilizand rez.next(), iar rez.next() este null query-ul este gresit sau nu returneaza date;

Pentru a utiliza procedurile stocate se utilizeaza

```
CallableStatement pstmt = c.getConect().prepareCall("call procedure(?,?,?,?,?);");
```

Unde ? sunt parametri de intrare. Acestia se seteaza precum pstmt.setType(parameter_index,value);

Instalare si utilizare

Pentru utilizarea programului create se executa fisierele sql in ordinea baza->proceduri->views-> select -> users. Dupa care se creaza un proiect nou java in care se adauga fisierele puse la dispozitie. Se adauga la proiect fisierul jar de mai devreme. In fisierul conexiune sa va schima variabila parola cu parola actuala a utilizatorului curent. Se va lua programul main din proiect.