**Proiect Sisteme de Gestiune a Bazelor de Date**

**“Companie operatoare de comenzi online”**

**Boroșeanu Darius-Ilie**

**Anul II, Grupa 244**

**2021-2022**

**Facultatea de Matematică și Informatică**

**Universitatea din București**

**1. Utilitatea bazei de date**

Proiectul constă în reprezentarea unei baze de date în care dorim să stocăm gestiunea unei companii care operează comenzi în mediul online, fiind relevantă pentru realitatea din zilele noastre, unde aproape toate produsele și bunurile se pot comanda online.

Aceasta bază de date conține informații despre o companie în cadrul căreia există manageri care se ocupă de administrarea companiei și tuturor procedurilor, există distribuitori care oferă spre vânzare diferite produse și sunt și clienții care pot plasa comenzi. De asemenea, sunt stocate informații despre produsele oferite spre vânzare, cât și despre recenziile acordate acestora de către clienți.

**2. Diagrama Entitate – Relație**

**Graphical user interface

Description automatically generated**

**3. Diagrama conceptuală**

**Graphical user interface, text

Description automatically generated**

**4. Crearea bazei de date**

CREATE TABLE PERSONAL (

id\_personal NUMBER(5) NOT NULL constraint pkey\_per PRIMARY KEY,

nume VARCHAR2(20) NOT NULL,

prenume VARCHAR2(20) NOT NULL,

rol1 VARCHAR2(20) NOT NULL,

adresa VARCHAR2(30) NOT NULL,

telefon NUMBER(15) UNIQUE,

credit NUMBER(10)

);

CREATE TABLE COMENZI (

id\_comanda NUMBER(5) NOT NULL constraint pkey\_com PRIMARY KEY,

status VARCHAR2(15) NOT NULL,

data\_comanda DATE DEFAULT SYSDATE,

id\_personal constraint fkey\_per REFERENCES PERSONAL(id\_personal)

);

CREATE TABLE CATEGORII(

id\_categorie NUMBER(5) NOT NULL constraint pkey\_cat PRIMARY KEY,

nume\_categorie VARCHAR(30) NOT NULL,

id\_manager constraint fkey\_per1 REFERENCES PERSONAL(id\_personal)

);

CREATE TABLE PRODUSE(

id\_produs NUMBER(5) NOT NULL constraint pkey\_pro PRIMARY KEY,

nume\_produs VARCHAR(30) NOT NULL,

descriere VARCHAR(255) NOT NULL,

pret NUMBER(10) NOT NULL,

cantitate NUMBER(5) NOT NULL,

id\_categorie constraint fkey\_cat REFERENCES CATEGORII(id\_categorie),

id\_distribuitor constraint fkey\_per2 REFERENCES PERSONAL(id\_personal)

);

CREATE TABLE RECENZII(

id\_recenzie NUMBER(5) NOT NULL constraint pkey\_rec PRIMARY KEY,

continut VARCHAR(255),

calificativ NUMBER(5),

id\_personal constraint fkey\_per3 REFERENCES PERSONAL(id\_personal),

id\_produs constraint fkey\_pro REFERENCES PRODUSE(id\_produs)

);

CREATE TABLE PRODUSE\_COMANDA(

id\_personal constraint fkey\_per4 REFERENCES PERSONAL(id\_personal),

id\_produs constraint fkey\_pro1 REFERENCES PRODUSE(id\_produs),

id\_comanda constraint fkey\_com REFERENCES COMENZI(id\_comanda),

cantitate\_pc NUMBER(5) NOT NULL

);

Graphical user interface, text, application, email

Description automatically generated

**5. Adăugare de date în tabelele create**

**Tabelul PERSONAL:**

INSERT INTO PERSONAL VALUES (1, 'Boroseanu', 'Darius', 'MANAGER', 'Bl. Gheorghe Magheru 17A', 899888, 0);

INSERT INTO PERSONAL VALUES (2, 'Popescu', 'Cristian', 'CLIENT', 'Strada Florilor 15', 524523, 500);

INSERT INTO PERSONAL VALUES (3, 'Popa', 'Ionut', 'CLIENT', 'Strada Muncii 21', 714935, 450);

INSERT INTO PERSONAL VALUES (4, 'Dumitru', 'Alexandru', 'DISTRIBUITOR', 'Aleea Parcului 7', 355892, 0);

INSERT INTO PERSONAL VALUES (5, 'Lazar', 'George', 'CLIENT', 'Strada Viitorului 12', 873521, 700);

INSERT INTO PERSONAL VALUES (6, 'Petrescu', 'Constantin', 'CLIENT', 'Strada Izvoarelor 98', 633349, 800);

INSERT INTO PERSONAL VALUES (7, 'Ene', 'Catalin', 'DISTRIBUITOR', 'Strada Planetelor 19', 837125, 0);

INSERT INTO PERSONAL VALUES (8, 'Banciu', 'Radu', 'MANAGER', 'Strada Franceza 1', 912548, 0);

**Tabelul COMENZI:**

INSERT INTO COMENZI VALUES (1, 'Plasata', to\_date('25-03-2019','dd-mm-yyyy'), 2);

INSERT INTO COMENZI VALUES (2, 'Plasata', to\_date('02-10-2020','dd-mm-yyyy'), 5);

INSERT INTO COMENZI VALUES (3, 'Livrata', to\_date('19-04-2018','dd-mm-yyyy'), 6);

INSERT INTO COMENZI VALUES (4, 'Livrata', to\_date('17-11-2021','dd-mm-yyyy'), 3);

INSERT INTO COMENZI VALUES (5, 'Procesata', to\_date('01-07-2020','dd-mm-yyyy'), 3);

INSERT INTO COMENZI VALUES (6, 'Plasata', to\_date('10-04-2022','dd-mm-yyyy'), 6);

INSERT INTO COMENZI VALUES (7, 'Procesata', to\_date('09-05-2022','dd-mm-yyyy'), 5);

INSERT INTO COMENZI VALUES (8, 'Livrata', to\_date('17-12-2019','dd-mm-yyyy'), 5);

INSERT INTO COMENZI VALUES (9, 'Procesata', to\_date('10-01-2022','dd-mm-yyyy'), 2);

INSERT INTO COMENZI VALUES (10, 'Livrata', to\_date('07-02-2021','dd-mm-yyyy'), 6);

INSERT INTO COMENZI VALUES (11, 'Livrata', to\_date('11-10-2021','dd-mm-yyyy'), 6);

INSERT INTO COMENZI VALUES (12, 'Plasata', to\_date('20-03-2022','dd-mm-yyyy'), 3);

**Tabelul CATEGORII:**

INSERT INTO CATEGORII VALUES (1, 'Imbracaminte', 1);

INSERT INTO CATEGORII VALUES (2, 'Accesorii', 1);

INSERT INTO CATEGORII VALUES (3, 'Incaltaminte', 1);

INSERT INTO CATEGORII VALUES (4, 'Electrocasince', 8);

INSERT INTO CATEGORII VALUES (5, 'Papetarie', 8);

INSERT INTO CATEGORII VALUES (6, 'Cosmetice', 8);

**Tabelul PRODUSE:**

INSERT INTO PRODUSE VALUES (1, 'Tricou', 'Bumbac', 20, 30, 1, 4);

INSERT INTO PRODUSE VALUES (2, 'Jacheta', 'De vara', 60, 10, 1, 4);

INSERT INTO PRODUSE VALUES (3, 'Pantaloni', 'Lungi', 45, 15, 1, 4);

INSERT INTO PRODUSE VALUES (4, 'Bluza', 'Subtire', 37, 20, 1, 4);

INSERT INTO PRODUSE VALUES (5, 'Papion', 'Rosu', 15, 45, 2, 4);

INSERT INTO PRODUSE VALUES (6, 'Cravata', 'Scurta', 22, 50, 2, 4);

INSERT INTO PRODUSE VALUES (7, 'Adidasi', 'Sport', 100, 55, 3, 4);

INSERT INTO PRODUSE VALUES (8, 'Pantofi', 'Eleganti', 120, 70, 3, 4);

INSERT INTO PRODUSE VALUES (9, 'Frigider', 'Clasa A+++', 550, 10, 4, 7);

INSERT INTO PRODUSE VALUES (10, 'Aragaz', 'Electric', 450, 12, 4, 7);

INSERT INTO PRODUSE VALUES (11, 'Caiet', 'Dictando', 5, 200, 5, 7);

INSERT INTO PRODUSE VALUES (12, 'Pix', 'Cu gel', 3, 500, 5, 7);

**Tabelul RECENZII:**

INSERT INTO RECENZII VALUES (1, 'Foarte calitativ!', 5, 2, 1);

INSERT INTO RECENZII VALUES (2, 'Ok pentru pretul sau.', 4, 3, 2);

INSERT INTO RECENZII VALUES (3, 'O varietate larga de marimi.', 5, 3, 3);

INSERT INTO RECENZII VALUES (4, 'Material slab calitativ.', 3, 6, 4);

INSERT INTO RECENZII VALUES (5, 'Nimic de reprosat!', 5, 5, 5);

INSERT INTO RECENZII VALUES (6, 'Impecabili!', 5, 2, 7);

INSERT INTO RECENZII VALUES (7, 'Instructiuni neintuitive.', 2, 3, 10);

INSERT INTO RECENZII VALUES (8, 'Culoarea negru arata bine.', 4, 6, 1);

**Tabelul PRODUSE\_COMANDA:**

INSERT INTO PRODUSE\_COMANDA VALUES (2, 3, 1, 1);

INSERT INTO PRODUSE\_COMANDA VALUES (2, 4, 1, 2);

INSERT INTO PRODUSE\_COMANDA VALUES (2, 7, 1, 1);

INSERT INTO PRODUSE\_COMANDA VALUES (3, 1, 2, 2);

INSERT INTO PRODUSE\_COMANDA VALUES (3, 2, 2, 1);

INSERT INTO PRODUSE\_COMANDA VALUES (5, 5, 3, 2);

INSERT INTO PRODUSE\_COMANDA VALUES (5, 6, 3, 3);

INSERT INTO PRODUSE\_COMANDA VALUES (5, 9, 4, 1);

INSERT INTO PRODUSE\_COMANDA VALUES (5, 10, 4, 1);

INSERT INTO PRODUSE\_COMANDA VALUES (6, 11, 5, 10);

INSERT INTO PRODUSE\_COMANDA VALUES (6, 12, 5, 15);

INSERT INTO PRODUSE\_COMANDA VALUES (3, 8, 6, 2);

Graphical user interface, text, application, email

Description automatically generated

**Cerința 6**

La cerința 6 am făcut o procedură care stochează și afișează toate produsele care au prețul mai mare decât o valoare dată.

CREATE OR REPLACE PROCEDURE ex6(val PRODUSE.pret%TYPE)

IS

TYPE tbl\_idx IS TABLE OF PRODUSE%ROWTYPE

INDEX BY PLS\_INTEGER;

prod tbl\_idx;

BEGIN

SELECT \* BULK COLLECT INTO prod

FROM PRODUSE

WHERE pret > val

ORDER BY pret;

FOR i in prod.first..prod.last LOOP

DBMS\_OUTPUT.PUT\_LINE(prod(i).id\_produs || ' ' || prod(i).nume\_produs || ' ' || prod(i).pret );

END LOOP;

END;

/

DECLARE

val PRODUSE.pret%TYPE := 30;

BEGIN

ex6(val);

END;

/

Graphical user interface, text, application, email

Description automatically generated

**Cerința 7**

La cerința 7 am făcut o procedură care stochează și afișează pentru fiecare categorie în parte câte produse conține.

CREATE OR REPLACE PROCEDURE ex7

IS

CURSOR C IS

SELECT nume\_categorie nume, COUNT(id\_produs) nr

FROM CATEGORII c, PRODUSE p

WHERE c.id\_categorie = p.id\_categorie(+)

GROUP BY nume\_categorie

ORDER BY nume\_categorie;

nume CATEGORII.nume\_categorie%type;

nr int;

BEGIN

OPEN C;

LOOP

FETCH C INTO nume, nr;

EXIT WHEN C%NOTFOUND;

IF (nr = 0) THEN

DBMS\_OUTPUT.PUT\_LINE('Categoria ' || nume || ' nu contine produse.');

ELSIF (nr = 1) THEN

DBMS\_OUTPUT.PUT\_LINE('Categoria ' || nume || ' contine doar un produs.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Categoria ' || nume || ' contine ' || nr || ' produse.');

END IF;

END LOOP;

END;

/

BEGIN

ex7;

END;

/

Graphical user interface, text, application

Description automatically generated

**Cerința 8**

La cerința 8 se cere să se afișeze numărul maxim de recenzii pe care l-a primit un produs dintr-o categorie introdusă prin nume.

CREATE OR REPLACE FUNCTION ex8(categ CATEGORII.nume\_categorie%type)

RETURN INTEGER IS

TYPE vector IS VARRAY(20) OF NUMBER(4);

prod vector;

nr NUMBER(4);

maxi NUMBER(4);

aux NUMBER(4);

id\_cat CATEGORII.id\_categorie%type;

nu\_exista EXCEPTION;

e\_goala EXCEPTION;

fara\_recenzii EXCEPTION;

BEGIN

nr := 0;

SELECT COUNT(nume\_categorie) into nr

FROM CATEGORII

WHERE nume\_categorie = categ;

IF (nr = 0) THEN RAISE nu\_exista;

END IF;

nr := 0;

SELECT id\_categorie into id\_cat

FROM CATEGORII

WHERE nume\_categorie = categ;

SELECT count(\*) INTO nr

FROM PRODUSE

WHERE id\_categorie = id\_cat;

IF (nr = 0) THEN RAISE e\_goala;

END IF;

SELECT id\_produs BULK COLLECT INTO prod

FROM PRODUSE

WHERE id\_categorie = id\_cat;

maxi := 0;

FOR i in prod.first..prod.last LOOP

aux := 0;

SELECT COUNT(\*) into aux

FROM RECENZII

WHERE id\_produs = prod(i);

IF (aux > maxi) THEN

maxi := aux;

END IF;

END LOOP;

IF (maxi = 0) THEN RAISE fara\_recenzii;

END IF;

RETURN maxi;

EXCEPTION

WHEN nu\_exista THEN

DBMS\_OUTPUT.PUT\_LINE('Categoria introdusă nu există în cadrul companiei.');

RETURN -1;

WHEN e\_goala THEN

DBMS\_OUTPUT.PUT\_LINE('Din categoria introdusă nu face parte niciun produs.');

RETURN -1;

WHEN fara\_recenzii THEN

DBMS\_OUTPUT.PUT\_LINE('În categoria introdusă, niciun produs nu a primit înca vreo recenzie.');

RETURN -1;

END;

/

BEGIN

DBMS\_OUTPUT.PUT\_LINE(ex8('Imbracaminte'));

END;

/

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, application, Teams

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

**Cerința 9**

La cerința 9 se cere să se determine și să se afișeze numărul produselor cumpărate din fiecare categorie de un client ale cărui nume și prenume sunt introduse.

CREATE OR REPLACE PROCEDURE ex9(nume1 PERSONAL.nume%type , prenume1 PERSONAL.prenume%type)

IS

TYPE tbl\_idx IS TABLE OF PRODUSE\_COMANDA%ROWTYPE

INDEX BY PLS\_INTEGER;

TYPE tbl\_idx2 is TABLE OF CATEGORII%ROWTYPE

INDEX BY PLS\_INTEGER;

nr\_id tbl\_idx;

cat tbl\_idx2;

TYPE vector IS VARRAY(40) OF CATEGORII.id\_categorie%TYPE;

cat\_tbl vector;

nume\_cat CATEGORII.nume\_categorie%TYPE;

aux number(4);

id\_pers PERSONAL.id\_personal%type := '-1';

nu\_exista\_comenzi\_livrate EXCEPTION;

BEGIN

aux := 0;

SELECT id\_personal into id\_pers

FROM PERSONAL

WHERE nume = nume1 AND prenume = prenume1;

SELECT pc.id\_personal, pc.id\_produs, pc.id\_comanda, pc.cantitate\_pc BULK COLLECT INTO nr\_id

FROM PRODUSE\_COMANDA pc, COMENZI cmz

WHERE pc.id\_personal = id\_pers AND NVL(pc.id\_comanda, 0) = cmz.id\_comanda

AND cmz.status ='Livrata';

IF (nr\_id.count = 0) THEN RAISE nu\_exista\_comenzi\_livrate;

END IF;

SELECT \* BULK COLLECT INTO cat

FROM CATEGORII;

cat\_tbl := VECTOR();

FOR i IN nr\_id.FIRST..nr\_id.LAST LOOP

SELECT id\_categorie INTO NUME\_CAT

FROM PRODUSE

WHERE nr\_id(i).id\_produs = id\_produs;

cat\_tbl.EXTEND();

cat\_tbl(i) := NUME\_CAT;

END LOOP;

FOR i IN CAT.FIRST..CAT.LAST LOOP

aux := 0;

FOR j IN cat\_tbl.FIRST..cat\_tbl.LAST LOOP

IF (CAT(i).id\_categorie = cat\_tbl(j)) THEN

aux := aux + 1;

END IF;

END LOOP;

IF (AUX != 0) THEN

DBMS\_OUTPUT.PUT(CAT(I).nume\_categorie || ' ');

AUX := 0;

FOR j IN cat\_tbl.FIRST..cat\_tbl.LAST LOOP

IF (CAT(i).id\_categorie = cat\_tbl(j)) THEN

AUX := AUX + nr\_id(j).cantitate\_pc;

END IF;

END LOOP;

DBMS\_OUTPUT.PUT(AUX);

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

END IF;

END LOOP;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu a fost gasit un client cu acest nume.');

WHEN nu\_exista\_comenzi\_livrate THEN

DBMS\_OUTPUT.PUT\_LINE('Clientul nu are comenzi livrate.');

END;

/

BEGIN

ex9('Lazar', 'George');

END;

/

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

**Cerința 10**

La cerința 10 am făcut un trigger care se declanșează în momentul în care dorim să facem modificări asupra tabelului PRODUSE in data de 1 mai sau în weekend.

CREATE OR REPLACE TRIGGER ex10

BEFORE INSERT OR UPDATE OR DELETE ON PRODUSE

BEGIN

IF (to\_char(sysdate, 'dd') = 01) AND (to\_char(sysdate, 'mm') = 05)

THEN raise\_application\_error(-20001, 'Nu se lucrează de 1 Mai!');

ELSIF (to\_char(sysdate,'D') = 6) OR (to\_char(sysdate,'D') = 7)

THEN raise\_application\_error(-20002, 'Nu se lucrează în weekend!');

END IF;

END;

/

Graphical user interface, text, application, email

Description automatically generated

**Cerința 11**

La cerința 11 am făcut un trigger care se declanșează în momentul în care se dorește ieftinirea sau scumpirea unui produs cu mai mult de 50% din valoarea sa.

CREATE OR REPLACE TRIGGER ex11

BEFORE UPDATE OF pret ON PRODUSE

FOR EACH ROW

BEGIN

IF (:NEW.pret > 1.5 \* :OLD.pret) OR (:NEW.pret < 0.5 \* :OLD.pret) THEN

RAISE\_APPLICATION\_ERROR(-20002,'Nu se poate modifica pretul unui produs cu mai mult de 50%');

END IF;

END;

/

Graphical user interface, text, application, email

Description automatically generated

**Cerința 12**

La cerința 12 am făcut un trigger care se declanșează în momentul în care se face o modificare asupra bazei de date. Datele vor fi inserate în tabelul SYS\_LOG.

CREATE TABLE sys\_log

(nume VARCHAR2(20),

nume\_bd VARCHAR2(30),

actiune VARCHAR2(20),

nume\_obiect VARCHAR2(30),

data\_actiune DATE

);

CREATE OR REPLACE TRIGGER ex12

BEFORE CREATE OR ALTER OR DROP ON SCHEMA

BEGIN

INSERT INTO sys\_log VALUES (sys.login\_user, sys.database\_name, sys.sysevent, sys.dictionary\_obj\_name, sysdate);

END;

/

Graphical user interface, text, application, email

Description automatically generated