ACADEMIA DE STUDII ECONOMICE DIN BUCURESTI

FACULTATEA DE CIBERNETICA, STATISTICA SI INFORMATICA ECONOMICA

SPECIALIZAREA INFORMATICA ECONOMICA



**PROIECT SGBD ORACLE**

**IMPLEMENTAREA UNEI BAZE DE DATE PENTRU O FIRMA DE ÎNCHIRIERI AUTO**

Bucuresti

2024

**PREZENTARE SCURTA A PROIECTULUI**

Baza de date este concepută pentru gestionarea unui serviciu de închiriere auto. Scopul principal al proiectului este să ofere o platformă pentru înregistrarea contractelor de închiriere între clienți și firmă, evidențierea informațiilor despre mașinile disponibile pentru închiriere, precum și gestionarea activităților de service auto.



**CREAREA TABELELOR**

CREATE TABLE TARIFE(

clasa CHAR(1) CONSTRAINT pk\_tarife PRIMARY KEY,

taxa\_zi NUMBER(8,2),

taxa\_luna NUMBER(8,2)

); -- TARIFE /\*P = premium class | B = business class | E = economic class \*/

CREATE TABLE MASINI(

id\_masina VARCHAR2(7) CONSTRAINT pk\_masini PRIMARY KEY,

model\_masina VARCHAR2(30),

caroserie\_masina VARCHAR2(15),

tip\_masina VARCHAR2(15),

an\_masina NUMBER(4),

clasa CHAR(1),

CONSTRAINT fk\_masini\_tarife FOREIGN KEY (clasa) REFERENCES TARIFE(clasa)

);

CREATE TABLE CLIENTI\_SOFERI(

id\_client NUMBER(5) CONSTRAINT pk\_clienti PRIMARY KEY,

nume VARCHAR2(20),

prenume VARCHAR2(20),

varsta NUMBER(2) CONSTRAINT ck\_clienti\_varsta CHECK (varsta>=18),

telefon VARCHAR2(20),

email VARCHAR2(30),

sex CHAR(1)

);

CREATE TABLE CONTRACTE(

id\_contract NUMBER(5) CONSTRAINT pk\_contracte PRIMARY KEY,

id\_client NUMBER(5),

id\_masina VARCHAR2(7),

data\_incepere DATE DEFAULT SYSDATE,

data\_incheiere DATE NOT NULL,

CONSTRAINT fk\_contracte\_clienti FOREIGN KEY (id\_client) REFERENCES CLIENTI\_SOFERI(id\_client),

CONSTRAINT fk\_contracte\_masini FOREIGN KEY(id\_masina) REFERENCES MASINI(id\_masina),

CONSTRAINT ck\_contracte\_data CHECK (data\_incheiere > data\_incepere)

);

CREATE TABLE MECANICI(

id\_mecanic NUMBER(5) CONSTRAINT pk\_mecanici PRIMARY KEY,

nume VARCHAR2(20),

prenume VARCHAR2(20),

telefon VARCHAR2(20),

data\_angajare DATE DEFAULT SYSDATE,

salariu NUMBER(8,2)

);

CREATE TABLE SERVICE(

nr\_service NUMBER(5) CONSTRAINT pk\_service PRIMARY KEY,

id\_mecanic NUMBER(5) CONSTRAINT fk\_service\_mecanici REFERENCES MECANICI(id\_mecanic),

id\_masina VARCHAR2(7) CONSTRAINT fk\_service\_masini REFERENCES MASINI(id\_masina),

data\_service DATE DEFAULT SYSDATE

);

CREATE TABLE PIESE(

id\_piesa NUMBER(6) CONSTRAINT pk\_testare PRIMARY KEY,

denumire VARCHAR2(20),

pret\_min NUMBER(8,2)

);

CREATE TABLE PIESE\_SERVICE(

id\_piesa NUMBER(6),

nr\_service NUMBER(5),

cantitate NUMBER(5),

pret NUMBER(8),

CONSTRAINT fk\_ps\_piese FOREIGN KEY (id\_piesa) REFERENCES PIESE(id\_piesa),

CONSTRAINT fk\_ps\_service FOREIGN KEY (nr\_service) REFERENCES SERVICE(nr\_service)

);

**Aplicatii**

### 1) Calculul mediei preturilor pieselor pentru fiecare serviciu + pret total

SET SERVEROUTPUT ON

DECLARE

v\_pret\_total NUMBER;

v\_media\_preturi NUMBER(8,2);

v\_count NUMBER;

BEGIN

FOR serviciu IN (SELECT nr\_service FROM SERVICE) LOOP

v\_media\_preturi := 0;

v\_count:=0;

v\_pret\_total:=0;

FOR piesa IN (SELECT pret FROM PIESE\_SERVICE WHERE nr\_service = serviciu.nr\_service) LOOP

v\_count:=v\_count+1;

v\_pret\_total:=v\_pret\_total + piesa.pret;

END LOOP;

v\_media\_preturi := v\_pret\_total / v\_count;

DBMS\_OUTPUT.PUT\_LINE('Media preturilor pentru serviciul ' || serviciu.nr\_service || ': ' || v\_media\_preturi);

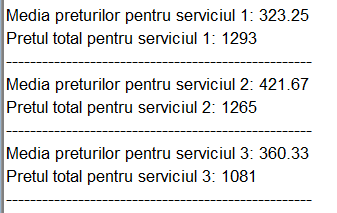
DBMS\_OUTPUT.PUT\_LINE('Pretul total pentru serviciul ' || serviciu.nr\_service || ': ' || v\_pret\_total);

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

END LOOP;

END;

/



**2) Modificarea tarifelor in functie de anul masinilor**

SET SERVEROUTPUT ON

DECLARE

v\_total\_masini NUMBER;

v\_masini\_vechi NUMBER;

v\_procent\_vechi NUMBER(5,2);

v\_masini\_noi NUMBER;

v\_procent\_noi NUMBER(5,2);

BEGIN

SELECT COUNT(\*) INTO v\_total\_masini FROM MASINI;

SELECT COUNT(\*) INTO v\_masini\_vechi FROM MASINI WHERE an\_masina <= 2020;

SELECT COUNT(\*) INTO v\_masini\_noi FROM MASINI WHERE an\_masina > 2020;

v\_procent\_vechi := (v\_masini\_vechi / v\_total\_masini) \* 100;

v\_procent\_noi:= (v\_masini\_noi / v\_total\_masini) \* 100;

DBMS\_OUTPUT.PUT\_LINE('Procentul de masini VECHI: ' || v\_procent\_vechi|| ' %');

DBMS\_OUTPUT.PUT\_LINE('Procentul de masini NOI: ' || v\_procent\_noi|| ' %');

IF v\_procent\_vechi > v\_procent\_noi THEN

UPDATE TARIFE

SET taxa\_zi = taxa\_zi \* 0.8, taxa\_luna = taxa\_luna \* 0.8

WHERE clasa = 'E' OR clasa = 'B';

DBMS\_OUTPUT.PUT\_LINE('||||S-A MODIFICAT PRETUL MASINILOR VECHI||||');

ELSE

UPDATE TARIFE

SET taxa\_zi = taxa\_zi \* 1.2, taxa\_luna = taxa\_luna \* 1.2

WHERE clasa = 'P';

DBMS\_OUTPUT.PUT\_LINE('||||S-A MODIFICAT PRETUL MASINILOR NOI||||');

END IF;

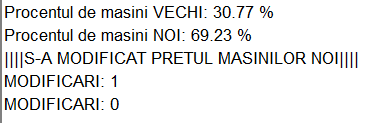
DBMS\_OUTPUT.PUT\_LINE('MODIFICARI: '||to\_char(SQL%ROWCOUNT));

ROLLBACK;

DBMS\_OUTPUT.PUT\_LINE('MODIFICARI: '||to\_char(SQL%ROWCOUNT));

END;

/



### 3) Sa se afiseze mecanicii care au lucrat in service, cu data si sa se afiseze toate piesele utilizate in acel service, cu tot cu costurile totale ale produselor (pret\*cantitate) si al service-ului. Daca mecanicii au cheltuit peste 1500 pe piese, sa se mareasca salariul cu 10%. La final, sa fie afisate modificarile.

SET SERVEROUTPUT ON

DECLARE

v\_nume\_mecanic VARCHAR2(50);

v\_piesa\_utilizata VARCHAR2(50);

v\_mecanic VARCHAR2(50);

v\_pret\_total NUMBER;

v\_nr\_modificari NUMBER:=0;

CURSOR c\_service\_piese(nr SERVICE.nr\_service%TYPE) IS

SELECT

m.nume || ' ' || m.prenume AS mecanic,

s.data\_service AS data\_service,

p.denumire AS piesa\_utilizata,

ps.pret\*ps.cantitate AS pret\_total

FROM MECANICI m,SERVICE s,PIESE\_SERVICE ps,PIESE p

WHERE m.id\_mecanic = s.id\_mecanic

AND s.nr\_service = ps.nr\_service

AND ps.id\_piesa = p.id\_piesa

AND s.nr\_service = nr;

BEGIN

FOR rec IN (SELECT nr\_service FROM SERVICE) LOOP

v\_pret\_total:=0;

FOR rec\_mecanic IN c\_service\_piese(rec.nr\_service) LOOP

IF v\_mecanic IS NULL OR v\_mecanic != rec\_mecanic.mecanic THEN

v\_mecanic := rec\_mecanic.mecanic;

DBMS\_OUTPUT.PUT\_LINE(rec\_mecanic.mecanic || ', Data service: ' || TO\_CHAR(rec\_mecanic.data\_service, 'DD-MM-YYYY'));

END IF;

DBMS\_OUTPUT.PUT\_LINE(' Piesa utilizata: ' || rec\_mecanic.piesa\_utilizata|| ' || Pret Total: ' || rec\_mecanic.pret\_total);

v\_pret\_total:=v\_pret\_total+rec\_mecanic.pret\_total;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE(' --------------------- TOTAL PE NR.SERVICE: ' || rec.nr\_service|| ' || ' ||v\_pret\_total ||' ||' );

FOR rec\_upd IN (SELECT

m.id\_mecanic

FROM MECANICI m,SERVICE s,PIESE\_SERVICE ps,PIESE p

WHERE m.id\_mecanic = s.id\_mecanic

AND s.nr\_service = ps.nr\_service

AND ps.id\_piesa = p.id\_piesa

AND s.nr\_service = rec.nr\_service

GROUP BY m.id\_mecanic

HAVING SUM(ps.pret \* ps.cantitate) > 1500) LOOP

UPDATE MECANICI

SET salariu = salariu \* 1.1

WHERE id\_mecanic = rec\_upd.id\_mecanic;

v\_nr\_modificari:=v\_nr\_modificari+1;

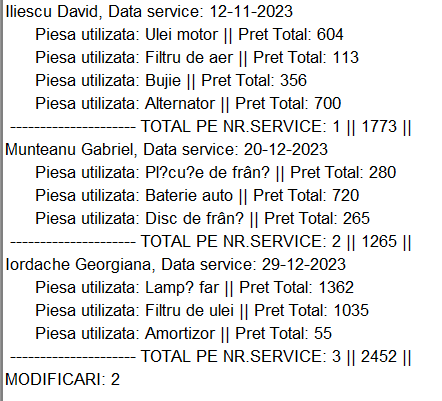
END LOOP;

END LOOP;

DBMS\_OUTPUT.PUT\_LINE('MODIFICARI: '||to\_char(v\_nr\_modificari));

END;

/



**4) Calculul costurilor de service pentru fiecare masina**

DECLARE

v\_cost\_total NUMBER;

BEGIN

FOR masina IN (SELECT id\_masina,nr\_service FROM SERVICE) LOOP

v\_cost\_total := 0;

FOR piesa IN (SELECT ps.cantitate \* ps.pret AS pret FROM PIESE\_SERVICE ps WHERE ps.nr\_service = masina.nr\_service) LOOP

v\_cost\_total := v\_cost\_total + piesa.pret;

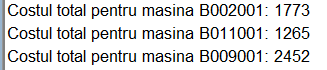
END LOOP;

DBMS\_OUTPUT.PUT\_LINE('Costul total pentru masina ' || masina.id\_masina || ': ' || v\_cost\_total);

END LOOP;

END;

/



**5) Afisarea detaliilor tuturor contractelor și a tarifelor aferente fiecărui contract**

SET SERVEROUTPUT ON

DECLARE

CURSOR c\_info IS

SELECT c.id\_contract, c.id\_client, c.id\_masina, c.data\_incepere, c.data\_incheiere, m.model\_masina, t.taxa\_zi, t.taxa\_luna

FROM CONTRACTE c

INNER JOIN MASINI m ON c.id\_masina = m.id\_masina

INNER JOIN TARIFE t ON m.clasa = t.clasa;

BEGIN

FOR c IN c\_info

LOOP

DBMS\_OUTPUT.PUT\_LINE('Contractul ' || c.id\_contract || ' pentru clientul ' || c.id\_client || ' cu masina ' || c.id\_masina ||

' (' || c.model\_masina || ') incepe la data de ' || TO\_CHAR(c.data\_incepere, 'DD-MON-YYYY') ||

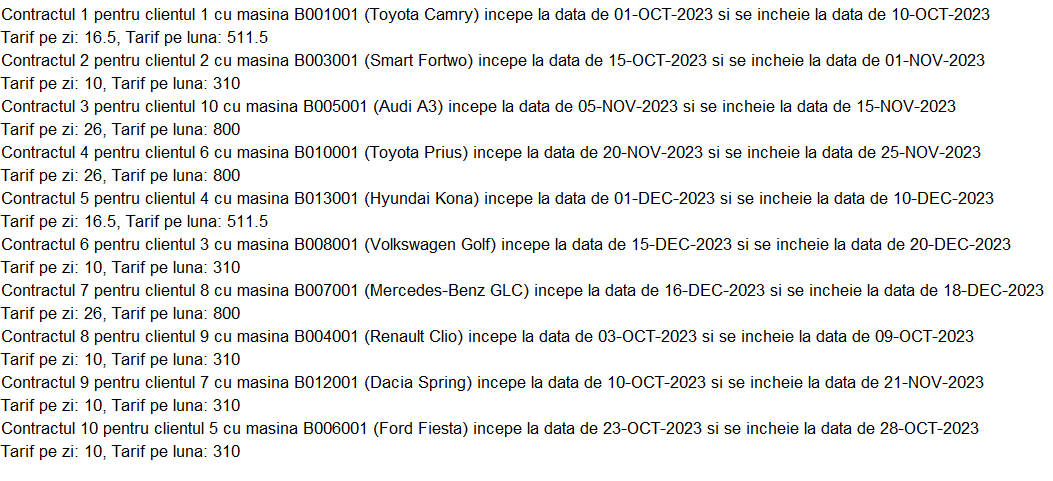
' si se incheie la data de ' || TO\_CHAR(c.data\_incheiere, 'DD-MON-YYYY'));

DBMS\_OUTPUT.PUT\_LINE('Tarif pe zi: ' || c.taxa\_zi || ', Tarif pe luna: ' || c.taxa\_luna);

END LOOP;

END;

/



**6) Afisarea masinilor dintr-un an scris de catre utilizator si afisarea numarului total de masini din anul respectiv**

SET SERVEROUTPUT ON

ACCEPT an PROMPT 'Anul masinii';

DECLARE

CURSOR c\_masini IS

SELECT \*

FROM MASINI

WHERE an\_masina = &an;

rec\_masina c\_masini%ROWTYPE;

v\_clasa VARCHAR2(15);

BEGIN

IF NOT c\_masini%ISOPEN THEN

OPEN c\_masini;

END IF;

LOOP

FETCH c\_masini INTO rec\_masina;

EXIT WHEN c\_masini%NOTFOUND;

IF rec\_masina.clasa='P' THEN v\_clasa := 'PREMIUM';

ELSIF rec\_masina.clasa='B' THEN v\_clasa := 'BUSINESS';

ELSE v\_clasa := 'ECONOMIC';

END IF;

IF c\_masini%ROWCOUNT > 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Masina: '||rec\_masina.model\_masina|| ' a fost realizata in anul '||rec\_masina.an\_masina|| ' si apartile clasei ' ||v\_clasa);

END IF;

END LOOP;

IF c\_masini%ROWCOUNT = 0 THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista masini din acest an');

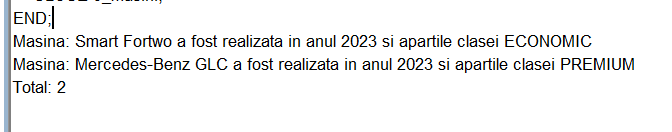
ELSE DBMS\_OUTPUT.PUT\_LINE('Total: '||c\_masini%ROWCOUNT);

END IF;

CLOSE c\_masini;

END;

/



**7) Actualizați prețul minim al unei piese. Dacă prețul minim nou este mai mic decât prețul minim existent, să se genereze o excepție definită de utilizator, iar în caz contrar să se actualizeze prețul și să se afișeze un mesaj de confirmare.**

SET SERVEROUTPUT ON

ACCEPT g\_id PROMPT 'Introduceti ID-ul '

ACCEPT g\_pret PROMPT 'Introduceti noul pret minim '

DECLARE

v\_pret\_min\_nou NUMBER := &g\_pret;

v\_pret\_min\_existent NUMBER;

pret\_min\_error EXCEPTION;

BEGIN

SELECT pret\_min INTO v\_pret\_min\_existent FROM PIESE WHERE id\_piesa = &g\_id;

IF v\_pret\_min\_nou < v\_pret\_min\_existent THEN

RAISE pret\_min\_error;

ELSE

UPDATE PIESE SET pret\_min = v\_pret\_min\_nou WHERE id\_piesa = &g\_id;

DBMS\_OUTPUT.PUT\_LINE('Pretul minim al piesei a fost actualizat cu succes.');

ROLLBACK;

END IF;

EXCEPTION

WHEN pret\_min\_error THEN

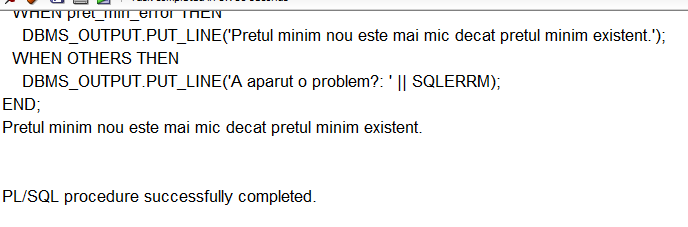
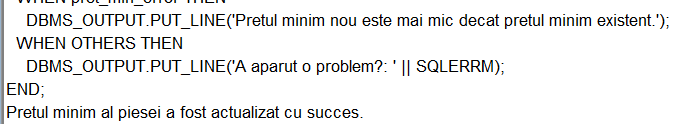
DBMS\_OUTPUT.PUT\_LINE('Pretul minim nou este mai mic decat pretul minim existent.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o problemă: ' || SQLERRM);

END;

/

**8) Sa se insereze intr-un record angajatul cu sex masculin din tabela clienti\_soferi. In caz de eroare, sa se lanseze o exceptie potrivita si, in cadrul acelei exceptii, sa se afiseze toti clientii de sex masculin**

SET SERVEROUTPUT ON

DECLARE

v\_angajat CLIENTI\_SOFERI%ROWTYPE;

BEGIN

SELECT \* INTO v\_angajat FROM CLIENTI\_SOFERI WHERE upper(sex) LIKE 'M';

DBMS\_OUTPUT.PUT\_LINE('Angajatul de sex masculin a fost selectat cu succes: ');

DBMS\_OUTPUT.PUT\_LINE('Nume: ' || v\_angajat.nume || ', Prenume: ' || v\_angajat.prenume || ', Varsta: ' || v\_angajat.varsta ||

', Telefon: ' || v\_angajat.telefon || ', Email: ' || v\_angajat.email || ', ID Client: ' || v\_angajat.id\_client);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista angajat de sex masculin.');

WHEN TOO\_MANY\_ROWS THEN

DBMS\_OUTPUT.PUT\_LINE('Prea multe inregistrari de sex masculin.');

DECLARE

CURSOR c\_clienti IS

SELECT nume, prenume, varsta, telefon, email, id\_client

FROM CLIENTI\_SOFERI

WHERE sex = 'M';

BEGIN

FOR rec\_client IN c\_clienti LOOP

DBMS\_OUTPUT.PUT\_LINE(' Nume: ' || rec\_client.nume || ', Prenume: ' || rec\_client.prenume ||

', Varsta: ' || rec\_client.varsta || ', Telefon: ' || rec\_client.telefon ||

', Email: ' || rec\_client.email || ', ID Client: ' || rec\_client.id\_client);

END LOOP;

END;

END;

/



**9) Sa se insereze in tabela contracte un nou contract cu data incheierii mai devreme decat data inceperii**

SET SERVEROUTPUT ON

DECLARE

ex\_data\_incheiere EXCEPTION;

PRAGMA EXCEPTION\_INIT(ex\_data\_incheiere, -02290);

BEGIN

INSERT INTO CONTRACTE (id\_contract, id\_client, id\_masina, data\_incepere,data\_incheiere)

VALUES (1001, 1, 'B001001',TO\_DATE('23-08-2024','DD-MM-YYYY'),SYSDATE); -- data\_incheiere are restrictie sa fie mai mare decat data\_incepere

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

EXCEPTION

WHEN ex\_data\_incheiere THEN

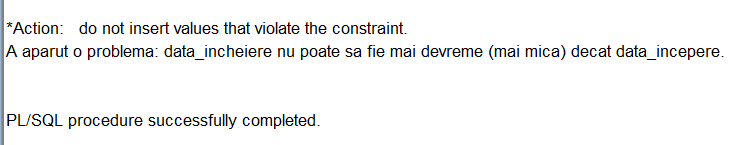
DBMS\_OUTPUT.PUT\_LINE('A aparut o problema: data\_incheiere nu poate sa fie mai devreme (mai mica) decat data\_incepere.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o alta problema: ' || SQLERRM);

END;

/



**10) Sa se mareasca salariul unui mecanic al carui ID este introdus de la tastatura, iar in cazul unei mariri nejustificate de peste 40%, sa se arunce o exceptie**

SET SERVEROUTPUT ON

ACCEPT g\_id PROMPT 'Introduceti ID-ul mecanicului';

ACCEPT g\_marire PROMPT 'Cu cat % sa fie marit salariul? ';

DECLARE

v\_marire NUMBER:=&g\_marire/100;

v\_salariu mecanici.salariu%TYPE;

v\_id mecanici.id\_mecanic%TYPE:=&g\_id;

ex\_crestere EXCEPTION;

BEGIN

SELECT salariu INTO v\_salariu FROM mecanici WHERE id\_mecanic = v\_id;

IF v\_marire >= 0.4 THEN

RAISE ex\_crestere;

ELSE

UPDATE mecanici SET salariu=salariu+salariu\*v\_marire WHERE id\_mecanic = v\_id;

DBMS\_OUTPUT.PUT\_LINE('Salariul mecanicului a fost actualizat cu succes.');

--ROLLBACK;

END IF;

EXCEPTION

WHEN ex\_crestere THEN

DBMS\_OUTPUT.PUT\_LINE('Cresterea salariala depaseste 40% din salariul actual. Actualizare anulata.');

WHEN NO\_DATA\_FOUND THEN

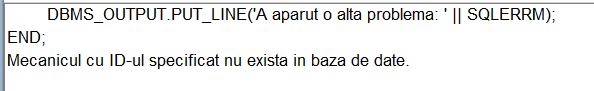
DBMS\_OUTPUT.PUT\_LINE('Mecanicul cu ID-ul specificat nu exista în baza de date.');

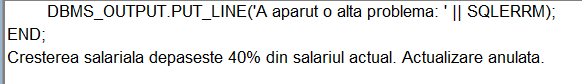
WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o alta problema: ' || SQLERRM);

END;

/





**PACHETE**

**PACKAGE 1:**

* **Proceduri: 1,2**
* **Functii: 1,2,4,5**

**PACKAGE 2:**

* **Proceduri: 3,4**
* **Functii: 3**

*PROCEDURI:*

*1/ Procedura pentru actualizarea zilei de incheiere al unui contract si rezolvarea exceptiilor*

*2/ Procedura pentru afisarea detaliilor unui contract*

*3/ Procedura pentru afisarea costurilor totale al unui service*

*4/ Procedura care mareste salariul tuturor mecanicilor cu 5% daca au lucrat in service la o masina dintr-o clasa introdusa ca parametru*

*FUNCTII:*

*1/ Functie pentru calcularea costului total al unui contract, luand in considerare timpul de inchiriere*

*2/ Functie pentru verificarea disponibilitatii unei masini pentru inchiriere*

*3/ Functie pentru gasirea mecanicului cu cele mai multe servicii*

*4/ Functie pentru afisarea duratei de inchiriere a unei masini*

*5/ Functie care returneaza masina cu cea mai lunga durata de inchiriere dintr-o anumita clasa*

*-------------------------------------------------------------------------------------------------------------------------------*

**PACKAGE 1**

create or replace PACKAGE analize\_contracte

AS

PROCEDURE actualizare\_data\_contract(

p\_id\_contract IN CONTRACTE.id\_contract%TYPE,

p\_noua\_data\_incheiere IN CONTRACTE.data\_incheiere%TYPE);

PROCEDURE afisare\_detalii\_contract(

p\_id\_contract IN CONTRACTE.id\_contract%TYPE);

FUNCTION calculeaza\_cost\_total\_contract(

p\_id\_contract CONTRACTE.id\_contract%TYPE) RETURN NUMBER;

FUNCTION verifica\_disponibilitate\_masina(

p\_id\_masina MASINI.id\_masina%TYPE,

p\_data\_incepere DATE,

p\_data\_incheiere DATE) RETURN BOOLEAN;

FUNCTION calculeaza\_durata\_inchiriere(

p\_id\_contract CONTRACTE.id\_contract%TYPE) RETURN VARCHAR2;

FUNCTION masina\_max\_durata\_inchiriere(

p\_clasa\_masina MASINI.clasa%TYPE) RETURN VARCHAR2;

END analize\_contracte;

**BODY-------------------------------------------------------------**

create or replace PACKAGE BODY analize\_contracte

AS

PROCEDURE actualizare\_data\_contract(

p\_id\_contract IN CONTRACTE.id\_contract%TYPE,

p\_noua\_data\_incheiere IN CONTRACTE.data\_incheiere%TYPE

)

IS

no\_data EXCEPTION;

ex\_data\_incheiere EXCEPTION;

PRAGMA EXCEPTION\_INIT(ex\_data\_incheiere, -02290);

BEGIN

UPDATE CONTRACTE

SET data\_incheiere = p\_noua\_data\_incheiere

WHERE id\_contract = p\_id\_contract;

IF SQL%ROWCOUNT = 0 THEN

RAISE no\_data;

ELSE

DBMS\_OUTPUT.PUT\_LINE('Data de incheiere pentru contractul ' || p\_id\_contract || ' a fost actualizata.');

END IF;

EXCEPTION

WHEN ex\_data\_incheiere THEN

DBMS\_OUTPUT.PUT\_LINE('A aparut o problema: data\_incheiere nu poate sa fie mai devreme (mai mica) decat data\_incepere.');

WHEN no\_data THEN

DBMS\_OUTPUT.PUT\_LINE('Contractul cu ID-ul ' || p\_id\_contract || ' nu exista.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Eroare la actualizarea contractului: ' || SQLERRM);

END;

PROCEDURE afisare\_detalii\_contract(

p\_id\_contract IN CONTRACTE.id\_contract%TYPE

)

IS

v\_client CLIENTI\_SOFERI.nume%TYPE;

v\_masina MASINI.model\_masina%TYPE;

v\_data\_inceput CONTRACTE.data\_incepere%TYPE;

v\_data\_incheiere CONTRACTE.data\_incheiere%TYPE;

v\_numar\_zile NUMBER;

v\_taxa\_zi TARIFE.taxa\_zi%TYPE;

v\_cost\_total NUMBER(10, 2);

BEGIN

SELECT cs.nume, m.model\_masina, c.data\_incepere, c.data\_incheiere

INTO v\_client, v\_masina, v\_data\_inceput, v\_data\_incheiere

FROM CONTRACTE c

INNER JOIN CLIENTI\_SOFERI cs ON cs.id\_client = c.id\_client

INNER JOIN MASINI m ON m.id\_masina = c.id\_masina

WHERE c.id\_contract = p\_id\_contract;

DBMS\_OUTPUT.PUT\_LINE('Detalii pentru contractul ' || p\_id\_contract || ':');

DBMS\_OUTPUT.PUT\_LINE('Client: ' || v\_client);

DBMS\_OUTPUT.PUT\_LINE('Masina: ' || v\_masina);

v\_numar\_zile := v\_data\_incheiere - v\_data\_inceput;

DBMS\_OUTPUT.PUT\_LINE('Numarul de zile de inchiriere: ' || v\_numar\_zile);

SELECT taxa\_zi

INTO v\_taxa\_zi

FROM TARIFE t

INNER JOIN MASINI m ON m.clasa = t.clasa

WHERE m.id\_masina = (SELECT id\_masina FROM CONTRACTE WHERE id\_contract = p\_id\_contract);

v\_cost\_total := v\_numar\_zile \* v\_taxa\_zi;

DBMS\_OUTPUT.PUT\_LINE('Costul total al inchirierii: ' || v\_cost\_total);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Contractul cu ID-ul ' || p\_id\_contract || ' nu exista.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Eroare la afișarea detaliilor contractului: ' || SQLERRM);

END;

FUNCTION calculeaza\_cost\_total\_contract(

p\_id\_contract CONTRACTE.id\_contract%TYPE

) RETURN NUMBER

IS

v\_cost\_total NUMBER := 0;

v\_durata\_zile NUMBER :=0;

BEGIN

SELECT data\_incheiere - data\_incepere

INTO v\_durata\_zile

FROM CONTRACTE

WHERE id\_contract = p\_id\_contract;

SELECT (t.taxa\_zi \* v\_durata\_zile)

INTO v\_cost\_total

FROM CONTRACTE c

JOIN MASINI m ON c.id\_masina = m.id\_masina

JOIN TARIFE t ON m.clasa = t.clasa

WHERE c.id\_contract = p\_id\_contract;

RETURN v\_cost\_total;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN -1;

WHEN OTHERS THEN

RETURN 0;

END;

FUNCTION verifica\_disponibilitate\_masina(

p\_id\_masina MASINI.id\_masina%TYPE,

p\_data\_incepere DATE,

p\_data\_incheiere DATE

) RETURN BOOLEAN

IS

ex\_data EXCEPTION;

v\_count NUMBER := 0;

BEGIN

IF (p\_data\_incepere > p\_data\_incheiere ) THEN

RAISE ex\_data;

END IF;

SELECT COUNT(\*)

INTO v\_count

FROM CONTRACTE

WHERE id\_masina = p\_id\_masina

AND ((p\_data\_incepere BETWEEN data\_incepere AND data\_incheiere)

OR (p\_data\_incheiere BETWEEN data\_incepere AND data\_incheiere));

RETURN v\_count = 0;

EXCEPTION

WHEN ex\_data THEN

RETURN FALSE;

WHEN NO\_DATA\_FOUND THEN

RETURN FALSE;

WHEN OTHERS THEN

RETURN FALSE;

END;

FUNCTION calculeaza\_durata\_inchiriere(

p\_id\_contract CONTRACTE.id\_contract%TYPE

) RETURN VARCHAR2

IS

v\_durata\_zile NUMBER := 0;

v\_durata\_text VARCHAR2(100);

BEGIN

SELECT data\_incheiere - data\_incepere

INTO v\_durata\_zile

FROM CONTRACTE

WHERE id\_contract = p\_id\_contract;

IF MOD(v\_durata\_zile, 7) = 0 THEN

v\_durata\_text := TO\_CHAR(v\_durata\_zile / 7) || ' saptamani';

ELSE

v\_durata\_text := TO\_CHAR(v\_durata\_zile) || ' zile';

END IF;

RETURN v\_durata\_text;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'Contractul nu exista.';

END;

FUNCTION masina\_max\_durata\_inchiriere(

p\_clasa\_masina MASINI.clasa%TYPE

) RETURN VARCHAR2

IS

v\_result VARCHAR2(400);

v\_max NUMBER:=0;

v\_masina MASINI%ROWTYPE;

BEGIN

FOR rec IN (

SELECT m.\*, (c.data\_incheiere - c.data\_incepere) AS durata\_inchiriere

FROM MASINI m

JOIN CONTRACTE c ON m.id\_masina = c.id\_masina

WHERE m.clasa = p\_clasa\_masina

ORDER BY durata\_inchiriere DESC

)

LOOP

IF (rec.durata\_inchiriere > v\_max) THEN

v\_max := rec.durata\_inchiriere;

v\_masina.id\_masina := rec.id\_masina;

v\_masina.model\_masina := rec.model\_masina;

END IF;

END LOOP;

IF v\_max > 0 THEN

v\_result := 'Masina cu cea mai lunga durata de inchiriere in clasa ' || p\_clasa\_masina || ' este: ' || v\_masina.id\_masina || ' --- ' || v\_masina.model\_masina || ', Durata: ' || v\_max || ' zile.';

ELSE

v\_result := 'Nu s-a gasit nicio masina in clasa specificata.';

END IF;

RETURN v\_result;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN 'Nu s-a gasit nicio masina in clasa specificata.';

END;

END analize\_contracte;

------------------------------------------------------------------------------------------------------------------------------------------

**PACKAGE 2**

create or replace PACKAGE analize\_service

AS

PROCEDURE calculeaza\_cost\_service(

p\_nr\_service IN SERVICE.nr\_service%TYPE);

PROCEDURE marire\_salariu\_mecanici\_clasa\_masina(

p\_clasa\_masina MASINI.clasa%TYPE);

FUNCTION mecanic\_max\_servicii RETURN MECANICI%ROWTYPE;

END analize\_service;

**BODY-------------------------------------------------------------**

create or replace PACKAGE BODY analize\_service

AS

PROCEDURE calculeaza\_cost\_service(

p\_nr\_service IN SERVICE.nr\_service%TYPE

)

IS

v\_total\_minim NUMBER := 0;

v\_total\_actual NUMBER := 0;

no\_data EXCEPTION;

v\_nr NUMBER;

BEGIN

SELECT COUNT(nr\_service)

INTO v\_nr

FROM SERVICE

WHERE nr\_service=p\_nr\_service;

IF v\_nr=0 THEN

RAISE no\_data;

END IF;

FOR rec1 IN (

SELECT ps.cantitate, p.pret\_min

FROM PIESE\_SERVICE ps

JOIN PIESE p ON ps.id\_piesa = p.id\_piesa

WHERE ps.nr\_service = p\_nr\_service

) LOOP

v\_total\_minim := v\_total\_minim + (rec1.cantitate \* rec1.pret\_min);

END LOOP;

FOR rec2 IN(

SELECT cantitate, pret

FROM PIESE\_SERVICE

WHERE nr\_service = p\_nr\_service

) LOOP

v\_total\_actual := v\_total\_actual + (rec2.cantitate \* rec2.pret);

END LOOP;

DBMS\_OUTPUT.PUT\_LINE( 'Cost minim: '|| v\_total\_minim);

DBMS\_OUTPUT.PUT\_LINE( 'Cost total: '||v\_total\_actual);

EXCEPTION

WHEN no\_data THEN

DBMS\_OUTPUT.PUT\_LINE( 'Nu exista service-ul cautat!');

END;

PROCEDURE marire\_salariu\_mecanici\_clasa\_masina(

p\_clasa\_masina MASINI.clasa%TYPE

)

IS

v\_count NUMBER;

v\_procent NUMBER := 0.05;

no\_data EXCEPTION;

BEGIN

SELECT COUNT(nr\_service)

INTO v\_count

FROM SERVICE s

INNER JOIN MASINI ma ON ma.id\_masina=s.id\_masina

WHERE clasa = p\_clasa\_masina;

IF v\_count=0 THEN

RAISE no\_data;

END IF;

FOR rec IN (

SELECT me.\*

FROM MASINI ma

INNER JOIN SERVICE s ON ma.id\_masina = s.id\_masina

INNER JOIN MECANICI me ON me.id\_mecanic = s.id\_mecanic

WHERE ma.clasa = p\_clasa\_masina )

LOOP

UPDATE MECANICI

SET salariu = salariu \* (1 + v\_procent)

WHERE id\_mecanic = rec.id\_mecanic;

DBMS\_OUTPUT.PUT\_LINE('Salariul mecanicului ' || rec.nume || ' ' || rec.prenume || ' a fost marit cu 5%.');

END LOOP;

EXCEPTION

WHEN no\_data THEN

DBMS\_OUTPUT.PUT\_LINE('Nu exista masini din clasa specificata in SERVICE.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Eroare: ' || SQLERRM);

END;

FUNCTION mecanic\_max\_servicii RETURN MECANICI%ROWTYPE

IS

v\_max\_servicii NUMBER := 0;

v\_mecanic MECANICI%ROWTYPE;

BEGIN

FOR rec IN (

SELECT id\_mecanic, COUNT(\*) AS numar\_servicii

FROM SERVICE

GROUP BY id\_mecanic

ORDER BY COUNT(\*) DESC

) LOOP

IF rec.numar\_servicii > v\_max\_servicii THEN

v\_max\_servicii := rec.numar\_servicii;

SELECT \*

INTO v\_mecanic

FROM MECANICI

WHERE id\_mecanic = rec.id\_mecanic;

END IF;

END LOOP;

RETURN v\_mecanic;

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

RETURN NULL;

WHEN OTHERS THEN

RETURN NULL;

END;

END analize\_service;

------------------------------------------------------------------------------------------------------------------------------------------

**APELURI**

**PROCEDURI:**

1)

ACCEPT g\_id PROMPT 'Introduceti ID-ul';

ACCEPT g\_data PROMPT 'Introduceti noua data in formatul DD-MM-YYYY';

BEGIN

analize\_contracte.actualizare\_data\_contract(&g\_id, TO\_DATE('&g\_data', 'DD-MM-YYYY'));

END;

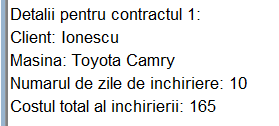
/

2)

BEGIN

analize\_contracte.AFISARE\_DETALII\_CONTRACT(1);

END;



3)

BEGIN

analize\_service.CALCULEAZA\_COST\_SERVICE(3);

END;

/



4)

BEGIN

analize\_service.MARIRE\_SALARIU\_MECANICI\_CLASA\_MASINA('B');

END;



**FUNCTII:**

1)

BEGIN

DBMS\_OUTPUT.PUT\_LINE (analize\_contracte.calculeaza\_cost\_total\_contract(&g));

END; 

2)

DECLARE

v\_result BOOLEAN;

v\_id\_masina MASINI.id\_masina%TYPE;

v\_data\_incepere DATE;

v\_data\_incheiere DATE;

BEGIN

v\_id\_masina := '&id\_masina';

v\_data\_incepere := TO\_DATE('&data\_incepere', 'DD-MM-YYYY');

v\_data\_incheiere := TO\_DATE('&data\_incheiere', 'DD-MM-YYYY');

v\_result := analize\_contracte.verifica\_disponibilitate\_masina(v\_id\_masina, v\_data\_incepere, v\_data\_incheiere);

IF v\_result THEN

DBMS\_OUTPUT.PUT\_LINE('Masina este disponibila.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Masina nu este disponibila.');

END IF;

END;

/



3)

DECLARE

v\_result MECANICI%ROWTYPE;

BEGIN

v\_result := analize\_service.mecanic\_max\_servicii;

DBMS\_OUTPUT.PUT\_LINE('Mecanicul cu cele mai multe servicii este: ' || v\_result.nume || ' ' || v\_result.prenume);

END;

/ 

4)

BEGIN

DBMS\_OUTPUT.PUT\_LINE(analize\_contracte.calculeaza\_durata\_inchiriere(&v\_id));

end;

/ 

5)

BEGIN

DBMS\_OUTPUT.PUT\_LINE(analize\_contracte.masina\_max\_durata\_inchiriere('&v\_clasa\_masina'));

END;

/ 

**TRIGGERI**

**1) Sa se realizeze un trigger pentru calculul automat al tarifelor de inchiriere la crearea contractului**

CREATE OR REPLACE TRIGGER afisare\_tarif\_inchiriere

AFTER INSERT ON CONTRACTE

FOR EACH ROW

DECLARE

v\_taxa\_zi TARIFE.taxa\_zi%TYPE;

v\_durata\_zile NUMBER;

v\_tarif\_total NUMBER;

BEGIN

SELECT taxa\_zi

INTO v\_taxa\_zi

FROM MASINI m, TARIFE t

WHERE m.clasa=t.clasa

AND id\_masina = :new.id\_masina;

v\_durata\_zile := TRUNC(:new.data\_incheiere - :new.data\_incepere);

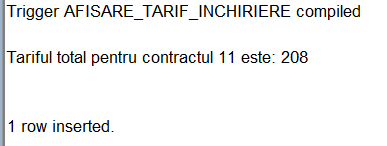
v\_tarif\_total := v\_taxa\_zi \* v\_durata\_zile;

DBMS\_OUTPUT.PUT\_LINE('Tariful total pentru contractul ' || :new.id\_contract || ' este: ' || v\_tarif\_total);

END;

/

INSERT INTO CONTRACTE VALUES(11,1,'B005001',TO\_DATE('01-05-2024','DD-MM-YYYY'),SYSDATE) ;



**2) Sa se realizeze un trigger care calculeaza suma cheltuita pe service dupa un update sau insert si care afiseaza cat a cheltuit mecanicul respectiv**

CREATE OR REPLACE TRIGGER calcul\_suma\_cheltuita\_service

BEFORE INSERT OR UPDATE OF cantitate ON PIESE\_SERVICE

FOR EACH ROW

DECLARE

v\_pret\_total NUMBER;

v\_nume\_mecanic VARCHAR2(50);

BEGIN

v\_pret\_total := :new.pret \* :new.cantitate;

SELECT m.nume

INTO v\_nume\_mecanic

FROM MECANICI m

JOIN SERVICE s ON m.id\_mecanic = s.id\_mecanic

WHERE s.nr\_service = :new.nr\_service;

IF INSERTING THEN

DBMS\_OUTPUT.PUT\_LINE('Mecanicul ' || v\_nume\_mecanic || ' a cheltuit pe piese ' || v\_pret\_total || ' în service-ul ' || :new.nr\_service);

ELSIF UPDATING THEN

IF :new.cantitate > :old.cantitate THEN

DBMS\_OUTPUT.PUT\_LINE('Mecanicul ' || v\_nume\_mecanic || ' a necesitat un surplus de piese si a cheltuit ' || v\_pret\_total || ' în service-ul ' || :new.nr\_service);

ELSE

RAISE\_APPLICATION\_ERROR(-20100, 'Actualizarea nu este permisa intrucat noua cantitate este mai mica.');

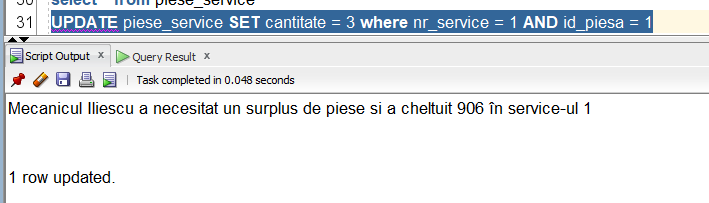
END IF;

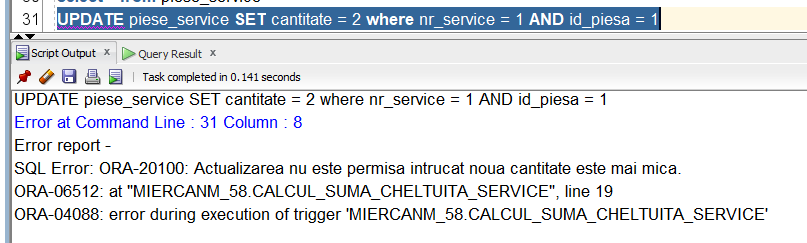
END IF;

END;

/

UPDATE piese\_service SET cantitate = 3 where nr\_service = 1 AND id\_piesa = 1





**3) Sa se realizeze un trigger care evidentiaza modificarea salariului mecanicilor**

CREATE OR REPLACE TRIGGER modificare\_salariu\_mecanic

BEFORE UPDATE OF salariu ON MECANICI

FOR EACH ROW

DECLARE

v\_diferenta\_salariu NUMBER;

BEGIN

v\_diferenta\_salariu := :new.salariu - :old.salariu;

IF :new.salariu > :old.salariu THEN

DBMS\_OUTPUT.PUT\_LINE('Mecanicului ' || :new.nume || ' ' || :new.prenume || ' a primit o marire de salariu de ' || v\_diferenta\_salariu || ' unitati.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Salariul mecanicului ' || :new.nume || ' ' || :new.prenume || ' a fost modificat de la ' || :old.salariu || ' unitati, la ' || :new.salariu || ' unitati.');

END IF;

END;

/

