

Universitatea din Bucuresti
Facultatea de Matematica si Informatica

Închirieri auto

PROIECT - SISTEME DE GESTIUNE A BAZELOR DE DATE

Nume student: Oprea Tudor
Grupa: 241

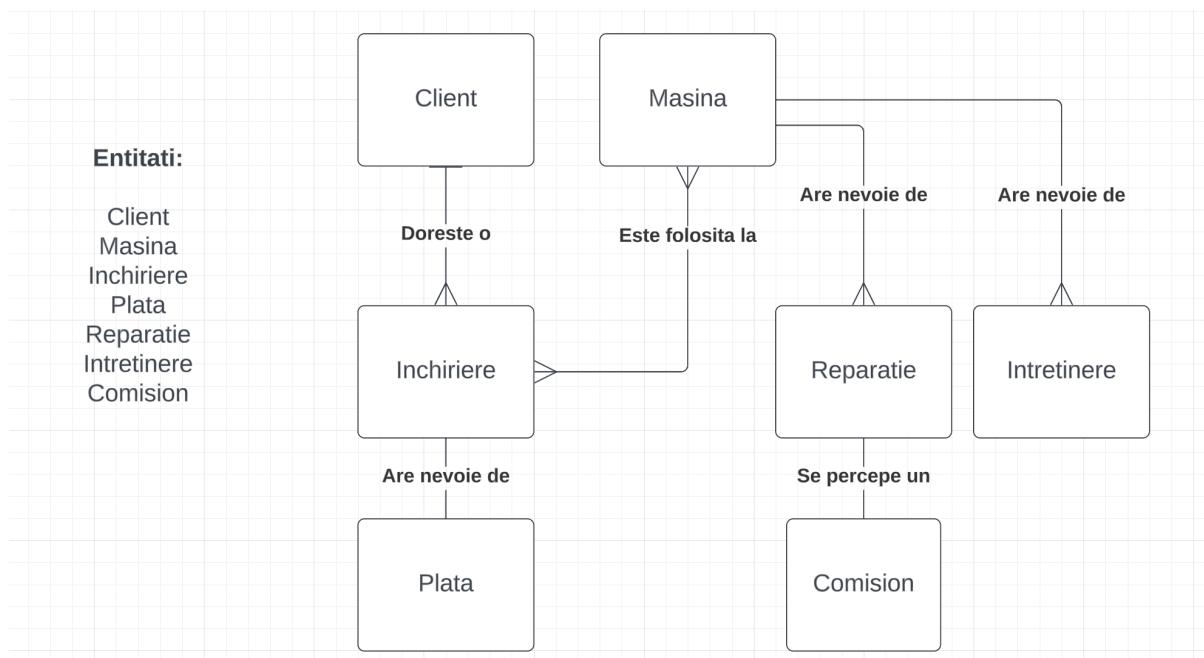
Cuprins - cerințe:

1. Prezentați pe scurt baza de date (utilitatea ei).
2. Realizați diagrama entitate-relație (ERD).
3. Pornind de la diagrama entitate-relație realizați diagrama conceptuală a modelului propus, integrând toate atributele necesare.
4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).
5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).
6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze două tipuri diferite de colecții studiate. Apelați subprogramul.
7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul.
8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 excepții. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.
9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate excepțiile care pot apărea, inclusiv excepțiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.
10. Definiți un trigger de tip LMD la nivel de comandă. Declanșați trigger-ul.
11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.
12. Definiți un trigger de tip LDD. Declanșați trigger-ul.
13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.

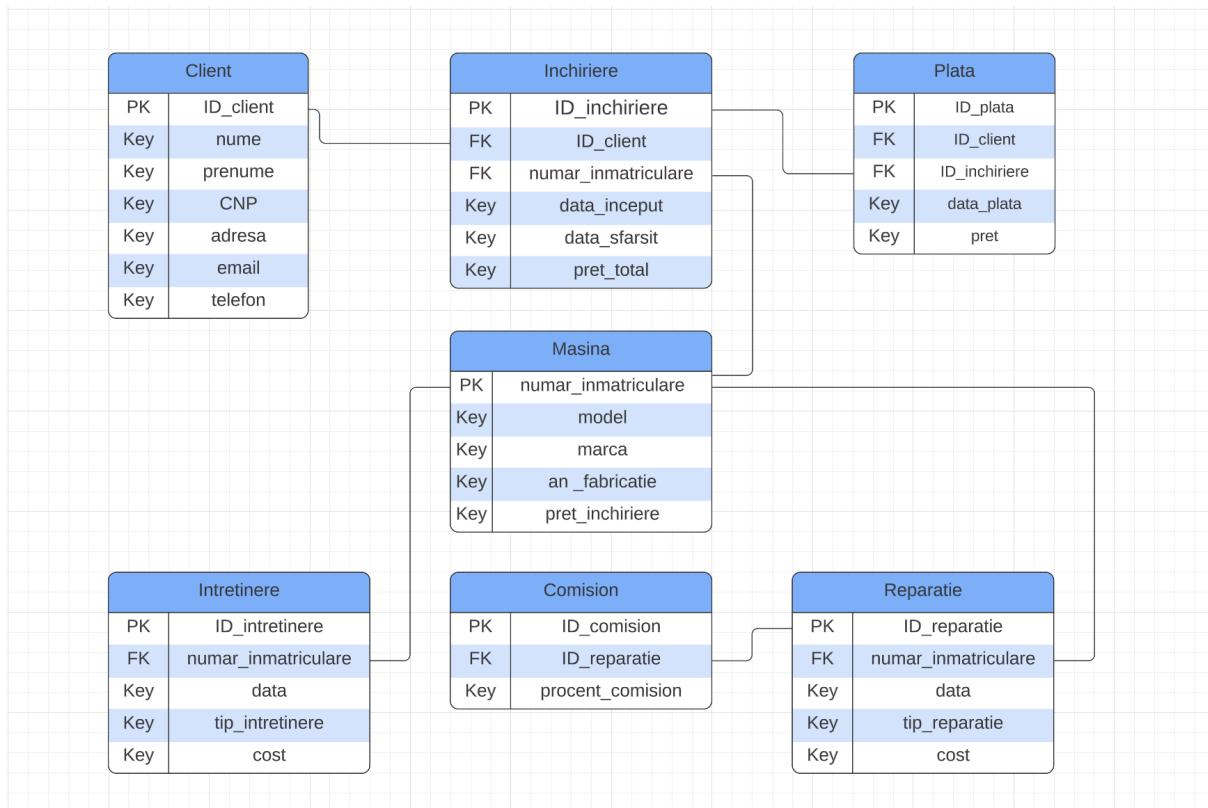
1. Prezentați pe scurt baza de date (utilitatea ei).

Proiectul propune o baza de date ce să permită gestiunea masinilor dintr-o flota ce dorește să le închirieze. Aceste mașini au nevoie de reparări, pentru care percepem și un comision, sau de întrețineri, ce se efectuează periodic. Există bineînțeles și clienți, care pot face una sau mai multe închirieri și pot alege dintre tipurile de mașini disponibile. Pentru orice închiriere trebuie să existe și o plată, care se poate efectua la o altă dată fata de cea la care se finalizează închirierea (clientul poate avea un anumit termen de plată).

2. Realizați diagrama entitate-relație (ERD).



3. Pornind de la diagrama entitate-relație realizată, creați diagrama conceptuală a modelului propus, integrând toate atributele necesare.



4. Implementați în Oracle diagrama conceptuală realizată: definiți toate tabelele, implementând toate constrângerile de integritate necesare (chei primare, cheile externe etc).

```

CREATE TABLE Client (
    ID_client INTEGER PRIMARY KEY,
    nume CHAR(255) NOT NULL,
    prenume VARCHAR(255) NOT NULL,
    CNP CHAR(13) NOT NULL,
    adresa VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    telefon CHAR(10) NOT NULL
);

```

```
CREATE TABLE Masina (
    numar_inmatriculare CHAR(7) PRIMARY KEY,
    model VARCHAR(255) NOT NULL,
    marca VARCHAR(255) NOT NULL,
    an_fabricatie INT NOT NULL,
    pret_inchiriere DECIMAL(10,2) NOT NULL,
    CHECK (an_fabricatie >= 1900 AND an_fabricatie <= 2023 );
```

```
CREATE TABLE Inchiriere (
    ID_inchiriere INTEGER PRIMARY KEY,
    ID_client INTEGER NOT NULL,
    numar_inmatriculare CHAR(7) NOT NULL,
    data_inceput DATE NOT NULL,
    data_sfarsit DATE NOT NULL,
    pret_total DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (ID_client) REFERENCES Client(ID_client),
    FOREIGN KEY (numar_inmatriculare) REFERENCES Masina(numar_inmatriculare),
    CHECK (data_inceput<=data_sfarsit)
);
```

```
CREATE TABLE Intretinere (
    ID_intretinere INTEGER PRIMARY KEY,
    numar_inmatriculare CHAR(7) NOT NULL,
    data DATE NOT NULL,
    tip_intretinere VARCHAR(255) NOT NULL,
    cost DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (numar_inmatriculare) REFERENCES Masina(numar_inmatriculare)
);
```

```
CREATE TABLE Reparatie (
    ID_reparatie INTEGER PRIMARY KEY,
    numar_inmatriculare CHAR(7) NOT NULL,
    data DATE NOT NULL,
    tip_reparatie VARCHAR(255) NOT NULL,
    cost DECIMAL(10,2) NOT NULL,
    FOREIGN KEY (numar_inmatriculare) REFERENCES Masina(numar_inmatriculare)
);
```

```
CREATE TABLE Comision (
    ID_comision INTEGER PRIMARY KEY,
    ID_reparatie INTEGER NOT NULL,
    procent_comision DECIMAL(5,2) NOT NULL,
    FOREIGN KEY (ID_reparatie) REFERENCES Reparatie(ID_reparatie)
);
```

```
CREATE TABLE Plata (
    ID_plata INTEGER PRIMARY KEY,
    ID_client INTEGER NOT NULL,
```

```

ID_inchiriere INTEGER NOT NULL,
data_plata DATE NOT NULL,
pret DECIMAL(8,2) NOT NULL,
FOREIGN KEY (ID_client) REFERENCES Client(ID_client),
FOREIGN KEY (ID_inchiriere) REFERENCES Inchiriere(ID_inchiriere)
);

```

5. Adăugați informații coerente în tabelele create (minim 5 înregistrări pentru fiecare entitate independentă; minim 10 înregistrări pentru tabela asociativă).

```

INSERT INTO Client (ID_client, nume, prenume, CNP, adresa, email, telefon)
VALUES (1, 'Popescu', 'Ion', '1234567891234', 'Str. Morii nr. 7',
'ion.popescu@gmail.com', '0712345678');

INSERT INTO Client (ID_client, nume, prenume, CNP, adresa, email, telefon)
VALUES (2, 'Mihai', 'Ioana', '1234567890000', 'Sos. Oltenitei nr.10',
'ioana.mihai@yahoo.com', '0799887766');

INSERT INTO Client (ID_client, nume, prenume, CNP, adresa, email, telefon)
VALUES (3, 'Grigore', 'Alexandru', '523456789444', 'Str.Mihai Viteazul nr.5',
'alex.grigore@gmail.com', '0712345999');

INSERT INTO Client (ID_client, nume, prenume, CNP, adresa, email, telefon)
VALUES (4, 'Argesanu', 'Rodica', '4253475891234', 'Calea Victoriei nr.35',
'argesanu.r@yahoo.com', '0778546123');

INSERT INTO Client (ID_client, nume, prenume, CNP, adresa, email, telefon)
VALUES (5, 'Oprea', 'Tudor', '5234567891004', 'Bulevardul Unirii nr.127',
'tudor.oprea@hotmail.com', '0710986338');

INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B185WHM', 'Golf', 'Volkswagen', 2018, 150);

INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B123FUG', 'Passat', 'Volkswagen', 2013, 120);

INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B07WHM', '1310', 'Dacia', 1997, 60);

INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B01TOP', 'X5', 'BMW', 2010, 175);

INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B106VHJ', 'Scala', 'Skoda', 2022, 150);

INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B06WHM', 'Supernova', 'Dacia', 2001, 90);

```

```

INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (1, 1, 'B185WHM', TO_DATE('2022-02-03', 'YYYY-MM-DD'),
TO_DATE('2022-02-07', 'YYYY-MM-DD'), 840);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (2, 4, 'B06WHM', TO_DATE('2022-05-09', 'YYYY-MM-DD'),
TO_DATE('2022-06-24', 'YYYY-MM-DD'), 1496.25);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (3, 2, 'B123FUG', TO_DATE('2022-10-01', 'YYYY-MM-DD'),
TO_DATE('2022-10-07', 'YYYY-MM-DD'), 498.54);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (4, 1, 'B185WHM', TO_DATE('2022-11-02', 'YYYY-MM-DD'),
TO_DATE('2022-12-01', 'YYYY-MM-DD'), 974.20);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (5, 2, 'B185WHM', TO_DATE('2022-09-15', 'YYYY-MM-DD'),
TO_DATE('2022-09-30', 'YYYY-MM-DD'), 758.50);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (6, 1, 'B185WHM', TO_DATE('2022-07-09', 'YYYY-MM-DD'),
TO_DATE('2022-07-14', 'YYYY-MM-DD'), 456.12);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (7, 1, 'B01TOP', TO_DATE('2022-02-07', 'YYYY-MM-DD'),
TO_DATE('2022-02-15', 'YYYY-MM-DD'), 550);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (8, 3, 'B06WHM', TO_DATE('2022-05-07', 'YYYY-MM-DD'),
TO_DATE('2022-06-30', 'YYYY-MM-DD'), 1545.25);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (9, 4, 'B123FUG', TO_DATE('2022-07-10', 'YYYY-MM-DD'),
TO_DATE('2022-07-15', 'YYYY-MM-DD'), 320.12);
INSERT INTO Inchiriere (ID_inchiriere, ID_client, numar_inmatriculare, data_inceput,
data_sfarsit, pret_total)
VALUES (10, 1, 'B06WHM', TO_DATE('2022-01-03', 'YYYY-MM-DD'),
TO_DATE('2022-01-09', 'YYYY-MM-DD'), 375.13);

INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere,
cost)
VALUES (1, 'B185WHM', TO_DATE('2022-12-20', 'YYYY-MM-DD'), 'schimb ulei',
100);

```

```

INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (2, 'B185WHM', TO_DATE('2022-09-12', 'YYYY-MM-DD'), 'schimb anvelope', 250);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (3, 'B06WHM', TO_DATE('2022-10-01', 'YYYY-MM-DD'), 'inspectie tehnica', 150);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (4, 'B123FUG', TO_DATE('2022-11-25', 'YYYY-MM-DD'), 'schimb bec far', 25);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (5, 'B185WHM', TO_DATE('2022-05-20', 'YYYY-MM-DD'), 'incarcare freon', 120);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (6, 'B07WHM', TO_DATE('2022-12-20', 'YYYY-MM-DD'), 'schimb ulei', 100);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (7, 'B01TOP', TO_DATE('2022-12-20', 'YYYY-MM-DD'), 'schimb filtru polen', 190);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (8, 'B185WHM', TO_DATE('2022-05-10', 'YYYY-MM-DD'), 'schimb anvelope', 300);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (9, 'B123FUG', TO_DATE('2022-01-05', 'YYYY-MM-DD'), 'completat antigel', 70);
INSERT INTO Intretinere (ID_intretinere, numar_inmatriculare, data, tip_intretinere, cost)
VALUES (10, 'B06WHM', TO_DATE('2022-01-15', 'YYYY-MM-DD'), 'completat solutie parbriz', 30);

```

```

INSERT INTO Reparatie (ID_reparatie, numar_inmatriculare, data, tip_reparatie, cost)
VALUES (1, 'B123FUG', TO_DATE('2022-01-15', 'YYYY-MM-DD'), 'schimb far si bara fata', 200);
INSERT INTO Reparatie (ID_reparatie, numar_inmatriculare, data, tip_reparatie, cost)
VALUES (2, 'B185WHM', TO_DATE('2022-01-30', 'YYYY-MM-DD'), 'inlocuit rezervor', 500);
INSERT INTO Reparatie (ID_reparatie, numar_inmatriculare, data, tip_reparatie, cost)

```

```

VALUES (3, 'B06WHM', TO_DATE('2022-12-04', 'YYYY-MM-DD'), 'reparatie cutie de viteze', 350);
INSERT INTO Reparatie (ID_reparatie, numar_inmatricularare, data, tip_reparatie, cost)
VALUES (4, 'B185WHM', TO_DATE('2022-11-24', 'YYYY-MM-DD'), 'macara geam stanga fata', 150);
INSERT INTO Reparatie (ID_reparatie, numar_inmatricularare, data, tip_reparatie, cost)
VALUES (5, 'B06WHM', TO_DATE('2022-10-01', 'YYYY-MM-DD'), 'schimb stop stanga', 300);

INSERT INTO Comision (ID_comision, ID_reparatie, procent_comision)
VALUES (1, 1, 10);
INSERT INTO Comision (ID_comision, ID_reparatie, procent_comision)
VALUES (2, 4, 15);
INSERT INTO Comision (ID_comision, ID_reparatie, procent_comision)
VALUES (3, 2, 5);
INSERT INTO Comision (ID_comision, ID_reparatie, procent_comision)
VALUES (4, 3, 20);
INSERT INTO Comision (ID_comision, ID_reparatie, procent_comision)
VALUES (5, 5, 15);

INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (1, 1, 1, TO_DATE('2022-02-07', 'YYYY-MM-DD'), 840);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (2, 4, 2, TO_DATE('2022-06-24', 'YYYY-MM-DD'), 1496.25);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (3, 2, 3, TO_DATE('2022-10-07', 'YYYY-MM-DD'), 498.54);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (4, 1, 4, TO_DATE('2022-12-01', 'YYYY-MM-DD'), 974.20);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (5, 2, 5, TO_DATE('2022-09-30', 'YYYY-MM-DD'), 758.5);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (6, 1, 6, TO_DATE('2022-07-14', 'YYYY-MM-DD'), 456.12);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (7, 1, 7, TO_DATE('2022-02-15', 'YYYY-MM-DD'), 550);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (8, 3, 8, TO_DATE('2022-06-30', 'YYYY-MM-DD'), 1545.25);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (9, 4, 9, TO_DATE('2022-07-15', 'YYYY-MM-DD'), 320.12);
INSERT INTO Plata (ID_plata , ID_client , ID_inchiriere , data_plata , pret )
VALUES (10, 1, 10, TO_DATE('2022-01-09', 'YYYY-MM-DD'), 375.13);

```

SQL Worksheet | History

Worksheet | Query Builder

```

1 /
2 select * from client;
3 /

```

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0,017 seconds

ID_CLIENT	NUME	PRENUME	CNP	ADRESA	EMAIL
1	Popescu	Ion	1234567891234	Str. Morii nr. 7	ion.popescu@gmail.com
2	Mihai	Ioana	1234567890000	Sos. Oltenitei nr.10	ioana.mihai@yahoo.com
3	Grigore	Alexandru	523456789444	Str.Mihai Viteazul nr.5	alex.grigore@gmail.com
4	Argesanu	Rodica	4253475891234	Calea Victoriei nr.35	argesanu.r@yahoo.com
5	Oprea	Tudor	5234567891004	Bulevardul Unirii nr.127	tudor.oprea@hotmail.com

SQL Worksheet | History

Worksheet | Query Builder

```

1 /
2 select * from masina;
3 /

```

Script Output | Query Result | SQL | All Rows Fetched: 10 in 0,016 seconds

NUMAR_INMATRICULARE	MODEL	MARCA	AN_FABRICATIE	PRET_INCHIRIERE
B185WHM	Golf	Volkswagen	2018	150
B123FUG	Passat	Volkswagen	2013	120
B07WHM	1310	Dacia	1997	60
B01TOP	X5	BMW	2010	175
B106VHJ	Scala	Skoda	2022	150
B06WHM	Supernova	Dacia	2001	90
B01SOL	Astra	Opel	2010	210
B108WOW	Vitara	Suzuki	2020	199,5
B108WOY	Vitara	Suzuki	2020	199,5
B01SOC	Astra	Opel	2010	141,7

SQL Worksheet | History

Worksheet | Query Builder

```
1 /  
2 select * from inchiriere;  
3 /
```

Script Output | Query Result | All Rows Fetched: 10 in 0,02 seconds

ID_INCHIRIERE	ID_CLIENT	NUMAR_INMATRICULARE	DATA_INCEPUT	DATA_SFARSIT	PRET_TOTAL
1	1	1 B185WHM	03-02-2022	07-02-2022	840
2	2	4 B06WHM	09-05-2022	24-06-2022	1496,25
3	3	2 B123FUG	01-10-2022	07-10-2022	498,54
4	4	1 B185WHM	02-11-2022	01-12-2022	974,2
5	5	2 B185WHM	15-09-2022	30-09-2022	758,5
6	6	1 B185WHM	09-07-2022	14-07-2022	456,12
7	7	1 B01TOP	07-02-2022	15-02-2022	550
8	8	3 B06WHM	07-05-2022	30-06-2022	1545,25
9	9	4 B123FUG	10-07-2022	15-07-2022	320,12
10	10	1 B06WHM	03-01-2022	09-01-2022	375,13

SQL Worksheet | History

Worksheet | Query Builder

```
1 /  
2 select * from inchiriere;  
3 /
```

Script Output | Query Result | All Rows Fetched: 10 in 0,02 seconds

ID_INCHIRIERE	ID_CLIENT	NUMAR_INMATRICULARE	DATA_INCEPUT	DATA_SFARSIT	PRET_TOTAL
1	1	1 B185WHM	03-02-2022	07-02-2022	840
2	2	4 B06WHM	09-05-2022	24-06-2022	1496,25
3	3	2 B123FUG	01-10-2022	07-10-2022	498,54
4	4	1 B185WHM	02-11-2022	01-12-2022	974,2
5	5	2 B185WHM	15-09-2022	30-09-2022	758,5
6	6	1 B185WHM	09-07-2022	14-07-2022	456,12
7	7	1 B01TOP	07-02-2022	15-02-2022	550
8	8	3 B06WHM	07-05-2022	30-06-2022	1545,25
9	9	4 B123FUG	10-07-2022	15-07-2022	320,12
10	10	1 B06WHM	03-01-2022	09-01-2022	375,13

SQL Worksheet | History

Worksheet | Query Builder

```
1 /  
2 select * from reparatie;  
3 /
```

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0,018 seconds

ID_REPARATIE	NUMAR_INMATRICULARE	DATA	TIP_REPARATIE	COST
1	1B123FUG	15-01-2022	schimb far si bara fata	200
2	2B185WHM	30-01-2022	inlocuit rezervor	500
3	3B06WHM	04-12-2022	reparatie cutie de viteze	350
4	4B185WHM	24-11-2022	macara_gream stanga fata	150
5	5B06WHM	01-10-2022	schimb stop stanga	300

SQL Worksheet | History

Worksheet | Query Builder

```
1 /  
2 select * from comision;  
3 /
```

Script Output | Query Result | SQL | All Rows Fetched: 5 in 0,018 seconds

ID_COMISION	ID_REPARATIE	PROCENT_COMISION
1	1	10
2	2	15
3	3	5
4	4	20
5	5	15

SQL Worksheet | History

Worksheet | Query Builder

```
1 /  
2 select * from plata;  
3 /
```

Script Output | Query Result | All Rows Fetched: 10 in 0,007 seconds

ID_PLATA	ID_CLIENT	ID_INCHIRIERE	DATA_PLATA	PRET
1	1	1	1 07-02-2022	840
2	2	4	2 24-06-2022	1496,25
3	3	2	3 07-10-2022	498,54
4	4	1	4 01-12-2022	974,2
5	6	1	6 14-07-2022	456,12
6	7	1	7 15-02-2022	550
7	8	3	8 30-06-2022	1545,25
8	9	4	9 15-07-2022	320,12
9	10	1	10 09-01-2022	375,13
10	5	2	5 30-09-2022	758,5

6. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze două tipuri diferite de colecții studiate. Apelați subprogramul.

Dacă un vector care conține numerele de înmatriculare ale mai multor mașini, afișați pentru fiecare mașină în parte numele și prenumele persoanelor care au închiriat-o.

```
-- datele de ieșire sunt stocate într-un tablou de caractere
/
CREATE OR REPLACE TYPE t_vector AS VARRAY(60) OF CHAR(7);
/
CREATE OR REPLACE PROCEDURE afiseaza_inchirieri (p_numar_inmatriculare
t_vector)
AS
    TYPE t_inchirieri IS TABLE OF VARCHAR2(4000);
    v_inchirieri t_inchirieri;

BEGIN
    for it in p_numar_inmatriculare.first .. p_numar_inmatriculare.last loop

        DBMS_OUTPUT.PUT_LINE('->'||p_numar_inmatriculare(it)||':');

        SELECT i.ID_inchiriere || ' | ' || c.nume || ' ' || c.prenume
        BULK COLLECT INTO v_inchirieri
        FROM Inchiriere i
        JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
        JOIN Client c ON i.ID_client = c.ID_client
        WHERE m.numar_inmatriculare = p_numar_inmatriculare(it);

        FOR i IN v_inchirieri.FIRST..v_inchirieri.LAST LOOP
            DBMS_OUTPUT.PUT_LINE(v_inchirieri(i));

        END LOOP;

        DBMS_OUTPUT.PUT_LINE(' ');

    end loop;
END;
/

BEGIN
afiseaza_inchirieri(t_vector('B185WHM', 'B06WHM','B01TOP'));
END;
```

/

```

SQL Worksheet: History
Worksheet Query Builder
SGBD_an2_semi Buffer Size: 20000
1 | Popescu Ion
4 | Popescu Ion
5 | Mihai Ioana
6 | Popescu Ion
->B185WHM:
1 | Popescu Ion
4 | Popescu Ion
5 | Mihai Ioana
6 | Popescu Ion
->B06WHM :
2 | Argesanu Rodica
8 | Grigore Alexandru
10 | Popescu Ion
->B01TOP :
7 | Popescu Ion
PL/SQL procedure successfully completed.

```

7. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent care să utilizeze 2 tipuri diferite de cursoare studiate, unul dintre acestea fiind cursor parametrizat. Apelați subprogramul.

Rezolvati cu ajutorul unui cursor parametrizat, ce primește un parametru număr de înmatriculare și afisează numele și prenumele persoanelor care au închiriat acea mașină. Folosiți un cursor fără parametrii care afisează marca și modelul mașinilor care au fost închiriate pentru mai mult de 1000 de lei.

CREATE OR REPLACE PROCEDURE afiseaza_inchirieri_parametrizat
(p_numar_inmatriculare CHAR)

AS

CURSOR c_inchirieri_parametrizat (p_numar_inmatriculare CHAR) IS
SELECT c.nume, c.prenume
FROM Inchiriere i
JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
JOIN Client c ON i.ID_client = c.ID_client
WHERE m.numar_inmatriculare = p_numar_inmatriculare;

CURSOR c_inchirieri_non_parametrizat IS

SELECT m.marca, m.model
FROM Inchiriere i
JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
JOIN Client c ON i.ID_client = c.ID_client
WHERE i.pret_total > 1000;

```

BEGIN
-- utilizarea cursorului parametrizat
DBMS_OUTPUT.PUT_LINE('Cursor parametrizat - clienti pentru acest numar:');
FOR r_inchiriere_parametrizat IN c_inchirieri_parametrizat (p_numar_inmatricularare)
LOOP
DBMS_OUTPUT.PUT_LINE(r_inchiriere_parametrizat.nume || ' ' ||
r_inchiriere_parametrizat.prenume);
END LOOP;

-- utilizarea cursorului non-parametrizat
DBMS_OUTPUT.PUT_LINE('Cursor non-parametrizat - marca si model:');
FOR r_inchiriere_non_parametrizat IN c_inchirieri_non_parametrizat LOOP
DBMS_OUTPUT.PUT_LINE(r_inchiriere_non_parametrizat.marca || ' ' ||
r_inchiriere_non_parametrizat.model);
END LOOP;
END;
/

```

```

BEGIN
    afiseaza_inchirieri_parametrizat('B185WHM');
END;
/

```

The screenshot shows the Oracle SQL Developer interface with a SQL Worksheet window. The script being run is as follows:

```

Run Statement (Ctrl+Enter)
265 END LOOP;
266
267 -- utilizarea cursorului non-parametrizat
268 DBMS_OUTPUT.PUT_LINE('Cursor non-parametrizat - marca si model:');
269 FOR r_inchiriere_non_parametrizat IN c_inchirieri_non_parametrizat
270 DBMS_OUTPUT.PUT_LINE(r_inchiriere_non_parametrizat.marca || ' ' ||
271 END LOOP;
272 END;
273 /
274
275 BEGIN
276     afiseaza_inchirieri_parametrizat('B185WHM');
277 END;
278 /
279
280

```

The output window displays the results of the two queries:

```

Cursor parametrizat - clienti pentru acest numar:
Popescu Ion
Popescu Ion
Mihai Ioana
Popescu Ion
Cursor non-parametrizat - marca si model:
Dacia Supernova
Dacia Supernova

```

The status bar at the bottom indicates "PL/SQL procedure successfully completed."

8. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip funcție care să utilizeze într-o singură comandă SQL 3 dintre tabelele definite. Definiți minim 2 exceptii. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Afisati datele inchirierii pentru un client care a inchiriat o singura data o masina. Daca a inchiriat de mai multe ori, se va afisa mesaj de eroare, la fel si daca nu a inchiriat niciodata.

```
CREATE OR REPLACE FUNCTION obtine_inchirieri (p_id_client INT)
RETURN VARCHAR2
AS
    v_result VARCHAR2(1000);
BEGIN
    SELECT i.ID_inchiriere || ' | ' || c.nume || ' ' || c.prenume || ' | ' || m.marca
    INTO v_result
    FROM Inchiriere i
    JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
    JOIN Client c ON i.ID_client = c.ID_client
    WHERE c.ID_client = p_id_client;
    RETURN v_result;
EXCEPTION
-- WHEN NO_DATA_FOUND THEN
--     RETURN 'Nu au fost găsite închirieri pentru acest client';
-- WHEN TOO_MANY_ROWS THEN
--     RETURN 'Acest client a inchiriat de mai multe ori';
WHEN NO_DATA_FOUND THEN
RAISE_APPLICATION_ERROR(-20010, 'Nu exista inchirieri pentru acest client');

WHEN TOO_MANY_ROWS THEN
RAISE_APPLICATION_ERROR(-20011, 'Acest client a inchiriat de mai multe ori');

END;
/
BEGIN

DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(&p_id_client));
-- apelarea functiei pentru un ID de client existent (cu o singura inchiriere)
--DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(3));

-- apelarea functiei pentru un ID de client care nu există
-- DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(999));
```

```
-- apelarea functiei pentru un ID cu mai multe inchirieri
-- DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(1));
END;
/

```

The screenshot shows the Oracle SQL Developer interface. The top window is a worksheet containing PL/SQL code. Lines 308 through 322 are visible, including the END; statement, a slash, BEGIN, and several comments about calling functions for client IDs 1 and 3. The line DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(3)); is highlighted in yellow. The bottom window, titled 'Script Output', displays the message 'PL/SQL procedure successfully completed.' indicating the code ran without errors.

The screenshot shows the Oracle SQL Developer interface. The top window is a worksheet containing the same PL/SQL code as the previous screenshot, with the line DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(3)); highlighted. The bottom window, titled 'Query Result', displays an 'Error report' section with the following details:

```
Error report -
ORA-20010: Nu exista inchirieri pentru acest client
ORA-06512: la "SYSTEM.OBTINE_INCHIRIERI", linia 20
ORA-06512: la linia 9
```

```

SQL Worksheet | History
Worksheet | Query Builder
313 --DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(&p_id_client));
314 -- apelarea functiei pentru un ID de client existent (cu o sg inchiriere)
315 --DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(3));
316
317
318 -- apelarea functiei pentru un ID de client care nu exista
319 --DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(999));
320
321 -- apelarea functiei pentru un ID cu mai multe inchirieri
322 DBMS_OUTPUT.PUT_LINE(obtine_inchirieri(1));
323 END;
324 /
325

Script Output | Query Result
Task completed in 0,046 seconds

Error report -
ORA-20011: Acest client a inchiriat de mai multe ori
ORA-06512: la "SYSTEM.OBTINE_INCHIRIERI", linia 23
ORA-06512: la linia 12

```

9. Formulați în limbaj natural o problemă pe care să o rezolvați folosind un subprogram stocat independent de tip procedură care să utilizeze într-o singură comandă SQL 5 dintre tabelele definite. Tratați toate exceptiile care pot apărea, inclusiv exceptiile NO_DATA_FOUND și TOO_MANY_ROWS. Apelați subprogramul astfel încât să evidențiați toate cazurile tratate.

Dat un id de client, afisati pentru acesta, doar daca a inchiriat o singura masina, datele masinii, numele sau, id ul platii, suma platita si apoi numarul de intrari in service pentru intretinere pe care le a avut masina respectiva. Definiți erorile: daca al nostru client a inchiriat mai multe masini sau de mai multe ori (too many rows) sau daca nu exista acest client in baza de date ori nu a inchiriat nimic.

```

CREATE OR REPLACE PROCEDURE afiseaza_inchirierile (p_id_client INT)
AS
v_memorare_inchiriere VARCHAR2(4000);
BEGIN
    SELECT i.ID_inchiriere || '|' || m.numar_inmatriculare || '|' || c.nume
    || '|' || c.prenume || '|' || p.id_plata || '|' || p.pret || '|' || COUNT(r.tip_intretinere) as
    numar_intretineri

    INTO v_memorare_inchiriere
    FROM Inchiriere i
    JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
    JOIN Client c ON i.ID_client = c.ID_client
    JOIN Plata p ON p.ID_inchiriere = i.ID_inchiriere
    JOIN Intretinere r on i.numar_inmatriculare = r.numar_inmatriculare
    WHERE c.ID_client = p_id_client

```

```

        GROUP BY i.ID_inchiriere, m.numar_inmatricularare, c.nume, c.prenume,
        p.id_plata, p.pret;

        DBMS_OUTPUT.PUT_LINE(v_memorare_inchiriere);

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu exista inchirieri pentru acest client.');
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('S-au gasit mai multe inchirieri decat era de asteptat
        pentru acest client.');
    END;
/
BEGIN
--client cu o singura inchiriere -> functioneaza corect
afiseaza_inchirierile(3);
DBMS_OUTPUT.PUT_LINE(' ');

--client cu mai multe inchirieri
afiseaza_inchirierile(1);
DBMS_OUTPUT.PUT_LINE(' ');

--client existent care nu a inchiriat nimic/ inexistent in bd
afiseaza_inchirierile(5);
DBMS_OUTPUT.PUT_LINE(' ');

END;
/

```

The screenshot shows the Oracle SQL Developer interface. In the top-left, there's a toolbar with various icons. Below it is a 'Worksheet' tab and a 'Query Builder' tab. The main area contains a code editor with a scroll bar. The code in the editor is:

```

359     DBMS_OUTPUT.PUT_LINE('Nu exista
360     WHEN TOO_MANY_ROWS THEN
361         DBMS_OUTPUT.PUT_LINE('S-au gasit
362     END;
363 /
364 BEGIN
365     --client cu o singura inchiriere ->
366     afiseaza_inchirierile(3);
367     DBMS_OUTPUT.PUT_LINE(' ');
368
369     --client cu mai multe inchirieri
370     afiseaza_inchirierile(1);
371     DBMS_OUTPUT.PUT_LINE(' ');
372
373     --client existent care nu a inchiriat
374     afiseaza_inchirierile(5);
375     DBMS_OUTPUT.PUT_LINE(' ');

```

To the right of the code editor is a 'Buffer Size: 20000' dropdown and a tab labeled 'SGBD_an2_sem1'. The 'Query Result' tab is active, showing the output of the script. The output consists of three lines of text:

```

8|B06WHM|Grigore|Alexandru|8|1545,25|2
S-au gasit mai multe inchirieri decat era de asteptat pentru acest client.
Nu exista inchirieri pentru acest client.

```

Below the tabs, there are buttons for 'Script Output' and 'Query Result', and a status message: 'Task completed in 0,09 seconds'. At the bottom of the interface, a progress bar indicates the task was completed successfully.

10. Definiți un trigger de tip LMD la nivel de comandă. Declansați trigger-ul.

```
CREATE OR REPLACE TRIGGER verifca_inserare
BEFORE INSERT OR UPDATE OR DELETE ON Masina
DECLARE
    v_numar_auto INT;
BEGIN
    SELECT COUNT(numar_inmatricularare)
    INTO v_numar_auto
    FROM MASINA;

    IF v_numar_auto < 5 THEN
        RAISE_APPLICATION_ERROR(-20002, 'Nu putem avea mai putin de 5 masini in
firma.');
    END IF;

    IF v_numar_auto > 9 THEN
        RAISE_APPLICATION_ERROR (-20003, 'Nu putem avea mai mult de 9 masini');
    END IF;
END;
```

/

```
-- DECLANSARE:
INSERT INTO Masina (numar_inmatricularare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B01ERO', 'Duster', 'Dacia', 2012, 120);
```

The screenshot shows the Oracle SQL Worksheet interface. The main pane displays the PL/SQL code for the trigger. The code includes logic to check if there are less than 5 or more than 9 cars in the database before inserting or updating a car record. The final part of the code attempts to insert a new car record ('B01ERO', 'Duster', 'Dacia', 2012, 120). The bottom pane, titled 'Script Output', shows the error messages resulting from the execution:

```
ORA-20003: Nu putem avea mai mult de 9 masini
ORA-06512: la "SYSTEM.VERIFICA_INSERARE", linia 13
ORA-04088: eroare in timpul executiei triggerului 'SYSTEM.VERIFICA_INSERARE'
```

11. Definiți un trigger de tip LMD la nivel de linie. Declanșați trigger-ul.

```
CREATE OR REPLACE TRIGGER verificare_adaugare_masina
BEFORE INSERT ON Masina
FOR EACH ROW
DECLARE
    v_numar_auto INT;
BEGIN

    SELECT COUNT (numar_inmatriculare) into v_numar_auto
    from MASINA
    where marca = :new.marca;
    DBMS_OUTPUT.PUT_LINE('S-a adaugat o noua masina cu numarul de
inmatriculare ' || :NEW.numar_inmatriculare);
    DBMS_OUTPUT.PUT_LINE('Este masina marca '||:new.marca ||' cu numarul ' ||
v_numar_auto+1);
END;
/
INSERT INTO Masina (numar_inmatriculare, model, marca, an_fabricatie,
pret_inchiriere)
VALUES ('B108XAC', 'Vitara', 'Suzuki', 2020, 190);
```

The screenshot shows the Oracle SQL Developer interface. On the left, the 'Worksheet' tab displays the PL/SQL code for creating a trigger named 'verificare_adaugare_masina'. This trigger is defined to run before an insert operation on the 'Masina' table, specifically for each row. It declares a variable 'v_numar_auto' of type INT. Inside the trigger body, it performs a SELECT COUNT query on the 'MASINA' table to find the number of rows where the 'marca' column matches the new value. It then uses DBMS_OUTPUT.PUT_LINE to print two messages: one indicating the addition of a new car with a specific license plate number, and another stating the car's model and its new assigned number. After the trigger definition, an 'INSERT INTO' statement is shown, adding a new record to the 'Masina' table with values: 'B108XAD', 'Vitara', 'Suzuki', 2020, and 190. On the right, the 'Query Result' tab shows the output of the executed code. It includes the inserted data and the two printed messages from the trigger's DBMS_OUTPUT.PUT_LINE statements.

12. Definiți un trigger de tip LDD. Declanșați trigger-ul.

```
CREATE TABLE ev_modif
(user_logat VARCHAR2(30),
eveniment VARCHAR2(20),
data TIMESTAMP(3));
/
CREATE OR REPLACE TRIGGER modif_bd
AFTER CREATE OR DROP OR ALTER ON SCHEMA

DECLARE
    ora_op TIMESTAMP;
BEGIN
    ora_op:=SYSDATE;
    INSERT INTO ev_modif
    VALUES ( SYS.LOGIN_USER,
    SYS.SYSEVENT, SYSTIMESTAMP(3));

    IF (TO_CHAR(ora_op,'HH24') NOT BETWEEN 7 AND 21)THEN
        RAISE_APPLICATION_ERROR(-20005,'Nu poti modifica tabelele in afara
orelor de program');
    END IF;
END;
/
CREATE TABLE nou (col INT);
ALTER TABLE nou ADD (col_6 INT);
--INSERT INTO nou VALUES (1,2);
/
SELECT * FROM ev_modif;
/
SELECT * from nou;
/
```

Triggerul se declanseaza cand avem o modificare in afara orelor de program (se vede ora apelarii, 23:46).

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, a query builder window displays the following PL/SQL code:

```

457    VALUES ('SYS.LOGIN_USER,
458           SYS.SYSEVENT, SYSTIMESTAMP(3));
459
460   IF (TO_CHAR(ora_op,'HH24') NOT BETWEEN 7 AND 21)THEN
461      RAISE_APPLICATION_ERROR(-20005,'Nu poti modifica tabelele in afara orelor de program');
462   END IF;
463
464 /
465 CREATE TABLE nou (col INT);
466 ALTER TABLE nou ADD (col_7 INT);
467 --INSERT INTO nou VALUES (1,2);
468 /
469 SELECT * FROM ev_modif;
470 /
471 SELECT * from nou;
472 /

```

In the bottom-left pane, the "Script Output" tab shows the execution results:

```

ORA-00604: eroare apărută la SQL recursiv nivelul 1
ORA-20005: Nu poti modifica tabelele in afara orelor de program
ORA-06512: la linia 10
04088. 00000 -  "error during execution of trigger '%s.%s'"

```

The status bar at the bottom right indicates the time as 23:46 and the date as 12.01.2023.

Triggerul tine evidenta cui modifica tabelele in tabelul ev_modif.

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, a query builder window displays the same PL/SQL code as before:

```

457    VALUES ('SYS.LOGIN_USER,
458           SYS.SYSEVENT, SYSTIMESTAMP(3));
459
460   IF (TO_CHAR(ora_op,'HH24') NOT BETWEEN 7 AND 21)THEN
461      RAISE_APPLICATION_ERROR(-20005,'Nu poti modifica tabelele in afara orelor de program');
462   END IF;
463
464 /
465 CREATE TABLE nou (col INT);
466 ALTER TABLE nou ADD (col_7 INT);
467 --INSERT INTO nou VALUES (1,2);
468 /
469 SELECT * FROM ev_modif;
470 /

```

In the bottom-left pane, the "Script Output" tab shows the execution results:

```

1 SYSTEM CREATE 12-01-2023 22:06:35,918000000
2 SYSTEM ALTER 12-01-2023 22:07:00,396000000
3 SYSTEM ALTER 12-01-2023 22:25:36,557000000
4 SYSTEM ALTER 12-01-2023 22:26:35,235000000
5 SYSTEM CREATE 12-01-2023 23:36:40,264000000
6 SYSTEM ALTER 12-01-2023 23:44:20,691000000

```

The status bar at the bottom right indicates the time as 23:46 and the date as 12.01.2023.

13. Definiți un pachet care să conțină toate obiectele definite în cadrul proiectului.

```
CREATE OR REPLACE PACKAGE PACHET_INCHIRIERE_AUTO AS
    --6
    TYPE t_vector IS VARRAY(60) OF CHAR(7);
    PROCEDURE afiseaza_inchirieri (p_numar_inmatriculare t_vector);
    --7
    PROCEDURE afiseaza_inchirieri_parametrizat (p_numar_inmatriculare CHAR);
    --8
    FUNCTION obtine_inchirieri (p_id_client INT) RETURN VARCHAR2;
    --9
    PROCEDURE afiseaza_inchirierile (p_id_client INT);

    END PACHET_INCHIRIERE_AUTO;

/
CREATE OR REPLACE PACKAGE BODY PACHET_INCHIRIERE_AUTO AS

    --6
    PROCEDURE afiseaza_inchirieri (p_numar_inmatriculare t_vector)
    IS
        TYPE t_inchirieri IS TABLE OF VARCHAR2(4000);
        v_inchirieri t_inchirieri;

    BEGIN
        for it in p_numar_inmatriculare.first .. p_numar_inmatriculare.last loop
            DBMS_OUTPUT.PUT_LINE('->'||p_numar_inmatriculare(it)||':');

            SELECT i.ID_inchiriere || ' | ' || c.nume || ' ' || c.prenume
            BULK COLLECT INTO v_inchirieri
            FROM Inchiriere i
            JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
            JOIN Client c ON i.ID_client = c.ID_client
            WHERE m.numar_inmatriculare = p_numar_inmatriculare(it);

            FOR i IN v_inchirieri.FIRST..v_inchirieri.LAST LOOP
                DBMS_OUTPUT.PUT_LINE(v_inchirieri(i));
            END LOOP;
            DBMS_OUTPUT.PUT_LINE(' ');
        end loop;
    END;
```

```
END afiseaza_inchirieri;
```

--7

```
PROCEDURE afiseaza_inchirieri_parametrizat (p_numar_inmatriculare CHAR)
```

```
IS
```

```
CURSOR c_inchirieri_parametrizat (p_numar_inmatriculare CHAR) IS
```

```
SELECT c.nume, c.prenume
```

```
FROM Inchiriere i
```

```
JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
```

```
JOIN Client c ON i.ID_client = c.ID_client
```

```
WHERE m.numar_inmatriculare = p_numar_inmatriculare;
```

```
CURSOR c_inchirieri_non_parametrizat IS
```

```
SELECT m.marca, m.model
```

```
FROM Inchiriere i
```

```
JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
```

```
JOIN Client c ON i.ID_client = c.ID_client
```

```
WHERE i.pret_total > 1000;
```

```
BEGIN
```

```
-- utilizarea cursorului parametrizat
```

```
DBMS_OUTPUT.PUT_LINE('Cursor parametrizat - clienti pentru acest  
numar:');
```

```
FOR r_inchiriere_parametrizat IN c_inchirieri_parametrizat  
(p_numar_inmatriculare) LOOP
```

```
DBMS_OUTPUT.PUT_LINE(r_inchiriere_parametrizat.nume || ' ' ||  
r_inchiriere_parametrizat.prenume);
```

```
END LOOP;
```

```
-- utilizarea cursorului non-parametrizat
```

```
DBMS_OUTPUT.PUT_LINE('Cursor non-parametrizat - marca si model:');
```

```
FOR r_inchiriere_non_parametrizat IN c_inchirieri_non_parametrizat LOOP  
DBMS_OUTPUT.PUT_LINE(r_inchiriere_non_parametrizat.marca || ' ' ||  
r_inchiriere_non_parametrizat.model);
```

```
END LOOP;
```

```
END afiseaza_inchirieri_parametrizat;
```

-- 8

```
FUNCTION obtine_inchirieri (p_id_client INT)
```

```
RETURN VARCHAR2
```

```
IS
```

```
v_result VARCHAR2(1000);
```

```
BEGIN
```

```
SELECT i.ID_inchiriere || ' | ' || c.nume || ' ' || c.prenume || ' | ' || m.marca
```

```
INTO v_result
```

```
FROM Inchiriere i
```

```
JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
```

```

        JOIN Client c ON i.ID_client = c.ID_client
        WHERE c.ID_client = p_id_client;
        RETURN v_result;
    EXCEPTION
        -- WHEN NO_DATA_FOUND THEN
        -- RETURN 'Nu au fost găsite închirieri pentru acest client';
        -- WHEN TOO_MANY_ROWS THEN
        -- RETURN 'Acest client a inchiriat de mai multe ori';

        WHEN NO_DATA_FOUND THEN
            RAISE_APPLICATION_ERROR(-20010, 'Nu exista inchirieri pentru acest
client');

        WHEN TOO_MANY_ROWS THEN
            RAISE_APPLICATION_ERROR(-20011, 'Acest client a inchiriat de mai
multe ori');
    END obtine_inchirieri;

```

```

-- 9
PROCEDURE afiseaza_inchirierile (p_id_client INT)
IS
    v_memorare_inchiriere VARCHAR2(4000);
BEGIN
    SELECT i.ID_inchiriere || '|' || m.numar_inmatriculare || '|' || c.nume
    || '|' || c.prenume || '|' || p.id_plata || '|' || p.pret || '|'
    COUNT(r.tip_intretinere) as numar_intretineri

    INTO v_memorare_inchiriere
    FROM Inchiriere i
    JOIN Masina m ON i.numar_inmatriculare = m.numar_inmatriculare
    JOIN Client c ON i.ID_client = c.ID_client
    JOIN Plata p ON p.ID_inchiriere = i.ID_inchiriere
    JOIN Intretinere r on i.numar_inmatriculare = r.numar_inmatriculare
    WHERE c.ID_client = p_id_client
    GROUP BY i.ID_inchiriere, m.numar_inmatriculare, c.nume, c.prenume,
    p.id_plata, p.pret;

```

```

    DBMS_OUTPUT.PUT_LINE(v_memorare_inchiriere);

    EXCEPTION
        WHEN NO_DATA_FOUND THEN
            DBMS_OUTPUT.PUT_LINE('Nu exista inchirieri pentru acest client.');
        WHEN TOO_MANY_ROWS THEN
            DBMS_OUTPUT.PUT_LINE('S-au gasit mai multe inchirieri decat era de
asteptat pentru acest client.');

```

```

        END afiseaza_inchirierile;

    END PACHET_INCHIRIERE_AUTO;

    /
BEGIN
--6
PACHET_INCHIRIERE_AUTO.afiseaza_inchirieri(PACHET_INCHIRIERE_AUTO.t_vector('B185WHM', 'B06WHM','B01TOP'));
DBMS_OUTPUT.PUT_LINE(' ');

--7
PACHET_INCHIRIERE_AUTO.afiseaza_inchirieri_parametrizat('B185WHM');

--8
DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(&p_id_client));
-- apelarea functiei pentru un ID de client existent (cu o singura inchiriere)
--DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(3));

-- apelarea functiei pentru un ID de client care nu există
-- DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(999));

-- apelarea functiei pentru un ID cu mai multe inchirieri
-- DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(1));

--9
--client cu o singura inchiriere -> functioneaza corect
PACHET_INCHIRIERE_AUTO.afiseaza_inchirierile(3);
DBMS_OUTPUT.PUT_LINE(' ');

--client cu mai multe inchirieri
PACHET_INCHIRIERE_AUTO.afiseaza_inchirierile(1);
DBMS_OUTPUT.PUT_LINE(' ');

--client existent care nu a inchiriat nimic/ inexistent in bd
afiseaza_inchirierile(5);
DBMS_OUTPUT.PUT_LINE(' ');

END;
/

```

The screenshot shows the Oracle SQL Developer interface during the execution of a PL/SQL procedure. The main window displays the code in the Worksheet tab:

```
606 /
607 BEGIN
608   --6
609   PACHET_INCHIRIERE_AUTO.afiseaza_inchirieri(PACHET_INCHIRIERE_AUTO.t_vec,
610   DBMS_OUTPUT.PUT_LINE(' ');
611
612   --7
613   PACHET_INCHIRIERE_AUTO.afiseaza_inchirieri_parametrizat('B185WHM');
614
615   --8
616   DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(&p_id_client));
617   -- apelarea functiei pentru un ID de client existent (cu o singura inchiriere)
618   --DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(3));
619   --DBMS_OUTPUT.PUT_LINE(PACHET_INCHIRIERE_AUTO.obtine_inchirieri(3));
```

The Script Output tab shows the completion message:

```
PL/SQL procedure successfully completed.
```

The Dbms Output tab displays the results of the procedure execution, categorized by cursor type:

- B185WHM:**
 - 1 | Popescu Ion
 - 4 | Popescu Ion
 - 5 | Mihai Ioana
 - 6 | Popescu Ion
- B06WHM :**
 - 2 | Argesanu Rodica
 - 8 | Grigore Alexandru
 - 10 | Popescu Ion
- B01TOP :**
 - 7 | Popescu Ion

Below these, two additional sections are shown:

- Cursor parametrizat - clienti pentru acest numar:**
 - Popescu Ion
 - Popescu Ion
 - Mihai Ioana
 - Popescu Ion
- Cursor non-parametrizat - marca si model:**
 - Dacia Supernova
 - Dacia Supernova