



---

# PROIECT SGBD

---

*Rețele de trotinete electrice*



AN UNIVERSITAR 2024-2025  
*LICĂ AMALIA GRUPA 1081*

# CUPRINS

<b>B. Introducere.....</b>	<b>2</b>
<b>C. Prezentarea schemei conceptuale .....</b>	<b>2</b>
<b>D. Crearea tabelelor și adăugarea datelor .....</b>	<b>4</b>
<b>E. Blocuri PL/SQL conținând structuri de control variate.....</b>	<b>13</b>
<b>F. Utilizarea cursorilor și a excepțiilor în cadrul blocurilor PL/SQL .....</b>	<b>16</b>
I. CURSORI.....	16
II. EXCEPȚII .....	20
<b>G. Funcții, proceduri .....</b>	<b>22</b>
I. PROCEDURI.....	22
II. FUNCȚII .....	26
<b>H. Concluzii și Perspective de Dezvoltare.....</b>	<b>30</b>
<b>I. Cod aferent din SQL Developer .....</b>	<b>31</b>

## Proiect SGBD

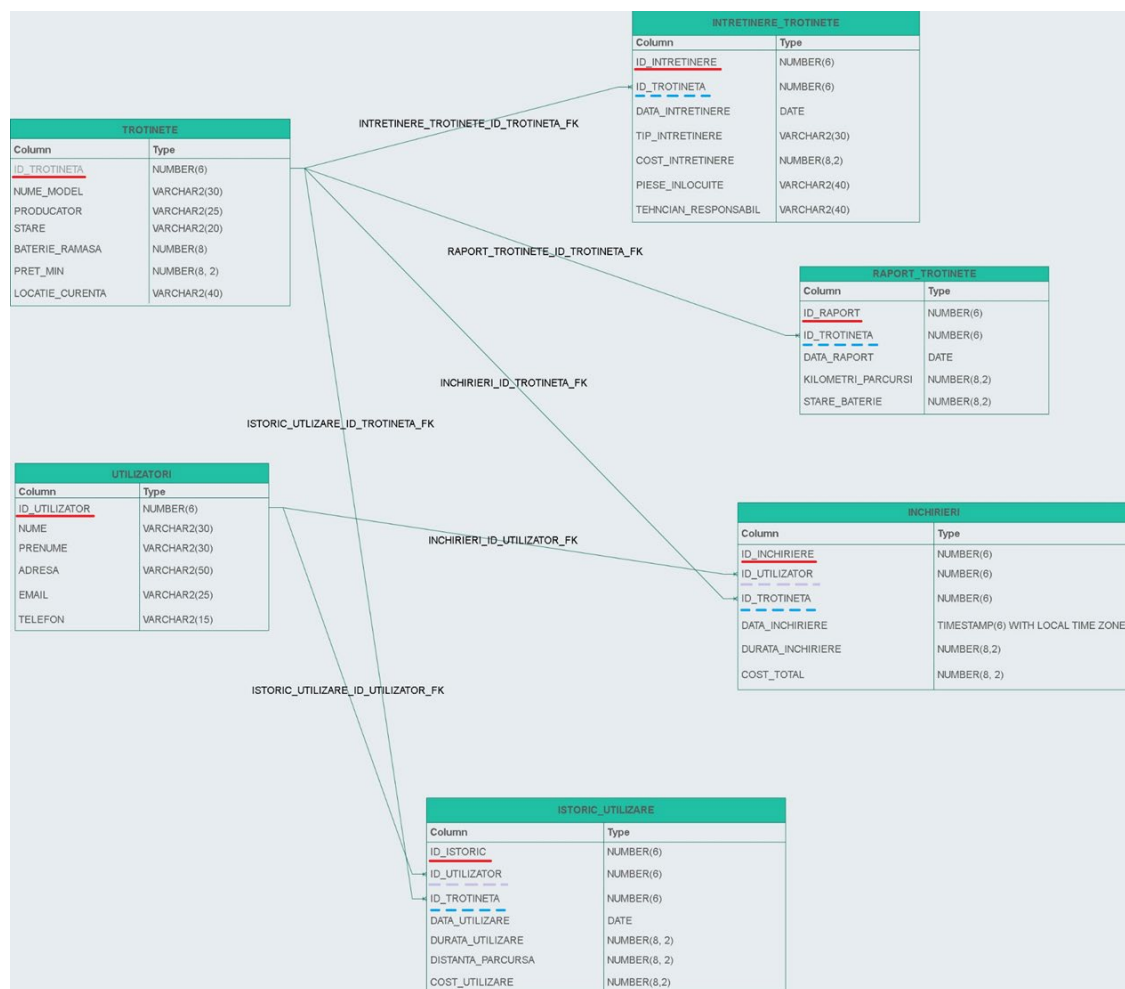
### B. Introducere

Proiectul propus se încadrează în domeniul mobilității urbane sustenabile, abordând provocările asociate gestionării rețelelor de trotinete electrice. Acest sistem își propune să optimizeze utilizarea acestor vehicule electrice prin dezvoltarea unei aplicații care integrează o bază de date complexă, destinat să îmbunătățească experiența utilizatorilor și să eficientizeze operațiunile.

Prin facilitarea închirierii rapide și monitorizarea în timp real a stării trotinetelor, aplicația răspunde nevoilor utilizatorilor pentru o mobilitate flexibilă și ecologică. Baza de date bine structurată permite o gestionare eficientă a resurselor, asigurând transparența și accesibilitatea informațiilor esențiale. Astfel, utilizatorii pot beneficia de date precise despre disponibilitatea trotinetelor, nivelul bateriei și locația acestora, contribuind la o utilizare optimă.

De asemenea, aplicația se angajează în menținerea integrității datelor, asigurându-se că informațiile sunt corecte și consistente. Prin analiza datelor istorice, sistemul nu doar facilitează servicii personalizate, dar și contribuie la îmbunătățirea continuă a ofertei, prin înțelegerea obiceiurilor de utilizare ale clienților.

### C. Prezentarea schemei conceptuale



Baza de date include tabele esențiale precum UTILIZATORI, TROTINETE, INCHIRIERI, RAPORT\_TROTINETE, INTRETINERE\_TROTINETE, ISTORIC\_UTILIZARE, acoperind astfel toate aspectele importante ale sistemului.

Tabela "TROTINETE" oferă informații detaliate despre fiecare trotinetă, inclusiv starea curentă, nivelul bateriei și locația curentă. Această tabelă formează nucleul sistemului, permițând monitorizarea și gestionarea eficientă a întregii rețele de trotinete.

Tabela "UTILIZATORI" gestionează datele utilizatorilor, cu accent pe identificare și contact. Această tabelă asigură o bază solidă pentru administrarea utilizatorilor și personalizarea serviciilor oferite.

Tabela "INCHIRIERI" cuprinde detalii esențiale despre fiecare închiriere, inclusiv durata și costul total. Această tabelă facilitează urmărirea și facturarea precisă a serviciilor de închiriere.

Tabela "ISTORIC\_UTILIZARE" furnizează înregistrările istorice ale utilizării trotinetelor, inclusiv distanța parcursă și costul asociat. Această tabelă oferă o perspectivă detaliată asupra obiceiurilor și comportamentului utilizatorilor.

Utilizatorii pot închiria trotinete în mod eficient, monitorizând în timp real starea acestora și selectând opțiunile de închiriere potrivite. Sistemul oferă notificări și informații în timp real despre nivelul bateriei, asigurând o utilizare fără probleme și minimalizând întreruperile. Prin funcționalitatea "ISTORIC\_UTILIZARE", se pot analiza detaliile privind utilizarea trotinetelor, furnizând date esențiale pentru îmbunătățirea serviciilor. Tabela "INTRETINERE\_TROTINETE" permite programarea și evidența activităților de întreținere, contribuind la menținerea în stare optimă a întregului sistem de trotinete.

Integritatea datelor contribuie la menținerea consistenței și corectitudinii informațiilor dintr-o bază de date. Prin stabilirea unei chei primare pentru fiecare înregistrare, se evită duplicarea datelor și se garantează unicitatea acestora. În același timp, cheile străine leagă tabelele între ele, asigurând coerența relațiilor și prevenind referințele către înregistrări inexistente. Astfel, aceste mecanisme nu numai că contribuie la structurarea eficientă a bazei de date, dar și la protejarea integrității informațiilor și la facilitarea unor interogări precise și rapoarte exacte.

Scopul final al proiectului este de a furniza o soluție software eficientă, capabilă să monitorizeze și să administreze în mod integral întreaga rețea de trotinete electrice, contribuind la optimizarea operațiunilor și la îmbunătățirea experienței utilizatorilor.

## D. Crearea tabelelor și adăugarea datelor

-- Crearea tabelii "TROTINETE"

```
CREATE TABLE LICAAMALIA_TROTINETE (  
    ID_TROTINETA NUMBER(6) PRIMARY KEY,  
    NUME_MODEL VARCHAR2(30),  
    PRODUCATOR VARCHAR2(25),  
    STARE VARCHAR2(20),  
    BATERIE_RAMASA NUMBER(8),  
    PRET_MINUT NUMBER(8,2),  
    LOCATIE_CURENTA VARCHAR2(40)  
);  
  
describe LICAAMALIA_TROTINETE
```

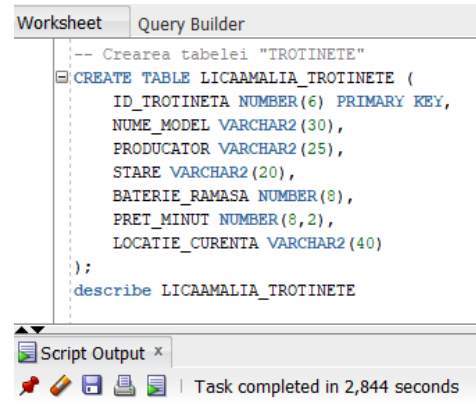


Table LICAAMALIA\_TROTINETE created.

Name	Null?	Type
ID_TROTINETA	NOT NULL	NUMBER(6)
NUME_MODEL		VARCHAR2(30)
PRODUCATOR		VARCHAR2(25)
STARE		VARCHAR2(20)
BATERIE_RAMASA		NUMBER(8)
PRET_MINUT		NUMBER(8,2)
LOCATIE_CURENTA		VARCHAR2(40)

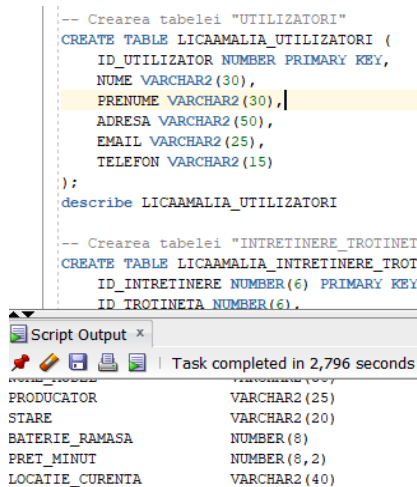


Table LICAAMALIA\_UTILIZATORI created.

Name	Null?	Type
ID_UTILIZATOR	NOT NULL	NUMBER
NUME		VARCHAR2(30)
PRENUME		VARCHAR2(30)
ADRESA		VARCHAR2(50)
EMAIL		VARCHAR2(25)
TELEFON		VARCHAR2(15)

describe LICAAMALIA\_UTILIZATORI

-- Crearea tabelii "UTILIZATORI"

```
CREATE TABLE LICAAMALIA_UTILIZATORI (  
    ID_UTILIZATOR NUMBER PRIMARY KEY,  
    NUME VARCHAR2(30),  
    PRENUME VARCHAR2(30),  
    ADRESA VARCHAR2(50),  
    EMAIL VARCHAR2(25),  
    TELEFON VARCHAR2(15)  
);  
  
describe LICAAMALIA_UTILIZATORI
```

```
+++ ALTER TABLE LICAAMALIA_UTILIZATORI  
ADD VARSTA NUMBER(3);
```

-- Crearea tabelii "INTRETINERE\_TROTINETE"

```
CREATE TABLE
LICAAMALIA_INTRETINERE_TROTINETE (

    ID_INTRETINERE NUMBER(6) PRIMARY
    KEY,

    ID_TROTINETA NUMBER(6),

    DATA_INTRETINERE DATE,

    TIP_INTRETINERE VARCHAR2(30),

    COST_INTRETINERE NUMBER(8,2),

    PIESE_INLOCUIE VARCHAR2(40),

    TEHNICIAN_RESPONSABIL VARCHAR2(40),

    FOREIGN KEY (ID_TROTINETA)
    REFERENCES
    LICAAMALIA_TROTINETE(ID_TROTINETA)
);

describe LICAAMALIA_INTRETINERE_TROTINETE
```

```
-- Crearea tabelii "INTRETINERE_TROTINETE"
CREATE TABLE LICAAMALIA_INTRETINERE_TROTINETE (
    ID_INTRETINERE NUMBER(6) PRIMARY KEY,
    ID_TROTINETA NUMBER(6),
    DATA_INTRETINERE DATE,
    TIP_INTRETINERE VARCHAR2(30),
    COST_INTRETINERE NUMBER(8,2),
    PIESE_INLOCUIE VARCHAR2(40),
    TEHNICIAN_RESPONSABIL VARCHAR2(40),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_INTRETINERE_TROTINETE

-- Crearea tabelii "RAPORT_TROTINETE"
```

Script Output x

Task completed in 2,449 seconds

Table LICAAMALIA\_INTRETINERE\_TROTINETE created.

Name	Null?	Type
ID_INTRETINERE	NOT NULL	NUMBER(6)
ID_TROTINETA		NUMBER(6)
DATA_INTRETINERE		DATE
TIP_INTRETINERE		VARCHAR2(30)
COST_INTRETINERE		NUMBER(8,2)
PIESE_INLOCUIE		VARCHAR2(40)
TEHNICIAN_RESPONSABIL		VARCHAR2(40)

-- Crearea tabelii "RAPORT\_TROTINETE"

```
-- Crearea tabelii "RAPORT_TROTINETE"
CREATE TABLE LICAAMALIA_RAPORT_TROTINETE (
    ID_RAPORT NUMBER(6) PRIMARY KEY,
    ID_TROTINETA NUMBER(6),
    DATA_RAPORT DATE,
    KILOMETRI_PARCURI NUMBER(8,2),
    STARE_BATERIE NUMBER(8,2),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_RAPORT_TROTINETE
```

Script Output x

Task completed in 2,284 seconds

Table LICAAMALIA\_RAPORT\_TROTINETE created.

Name	Null?	Type
ID_RAPORT	NOT NULL	NUMBER(6)
ID_TROTINETA		NUMBER(6)
DATA_RAPORT		DATE
KILOMETRI_PARCURI		NUMBER(8,2)
STARE_BATERIE		NUMBER(8,2)

```
CREATE TABLE
LICAAMALIA_RAPORT_TROTINETE (

    ID_RAPORT NUMBER(6) PRIMARY KEY,

    ID_TROTINETA NUMBER(6),

    DATA_RAPORT DATE,

    KILOMETRI_PARCURI NUMBER(8,2),

    STARE_BATERIE NUMBER(8,2),

    FOREIGN KEY (ID_TROTINETA) REFERENCES
    LICAAMALIA_TROTINETE(ID_TROTINETA)
);

describe LICAAMALIA_RAPORT_TROTINETE
```

-- Crearea tabelii "INCHIRIERI"

```
CREATE TABLE LICAAMALIA_INCHIRIERI (

    ID_INCHIRIERE NUMBER(6) PRIMARY KEY,

    ID_UTILIZATOR NUMBER(6),

    ID_TROTINETA NUMBER(6),

    DATA_INCHIRIERE TIMESTAMP(6) WITH
LOCAL TIME ZONE,

    DURATA_INCHIRIERE NUMBER(6),

    COST_TOTAL NUMBER(8,2),

    FOREIGN KEY (ID_UTILIZATOR)
REFERENCES
LICAAMALIA_UTILIZATORI(ID_UTILIZATOR),

    FOREIGN KEY (ID_TROTINETA)
REFERENCES
LICAAMALIA_TROTINETE(ID_TROTINETA)

);

describe LICAAMALIA_INCHIRIERI
```

```
-- Crearea tabelii "INCHIRIERI"
CREATE TABLE LICAAMALIA_INCHIRIERI (
    ID_INCHIRIERE NUMBER(6) PRIMARY KEY,
    ID_UTILIZATOR NUMBER(6),
    ID_TROTINETA NUMBER(6),
    DATA_INCHIRIERE TIMESTAMP(6) WITH LOCAL TIME ZONE,
    DURATA_INCHIRIERE NUMBER(6),
    COST_TOTAL NUMBER(8,2),
    FOREIGN KEY (ID_UTILIZATOR) REFERENCES LICAAMALIA_UTILIZATORI(ID_UTILIZATOR),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_INCHIRIERI

-- Crearea tabelii "ISTORIC_UTILIZARE"
CREATE TABLE LICAAMALIA_ISTORIC_UTILIZARE (
```

Script Output x

Task completed in 2,027 seconds

Table LICAAMALIA\_INCHIRIERI created.

Name	Null?	Type
ID_INCHIRIERE	NOT NULL	NUMBER(6)
ID_UTILIZATOR		NUMBER(6)
ID_TROTINETA		NUMBER(6)
DATA_INCHIRIERE		TIMESTAMP(6) WITH LOCAL TIME ZONE
DURATA_INCHIRIERE		NUMBER(6)
COST_TOTAL		NUMBER(8,2)

-- Crearea tabelii "ISTORIC\_UTILIZARE"

```
-- Crearea tabelii "ISTORIC_UTILIZARE"
CREATE TABLE LICAAMALIA_ISTORIC_UTILIZARE (
    ID_ISTORIC NUMBER(6) PRIMARY KEY,
    ID_UTILIZATOR NUMBER(6),
    ID_TROTINETA NUMBER(6),
    DATA_UTILIZARE DATE,
    DURATA_UTILIZARE NUMBER(6),
    DISTANTA_PARCURSA NUMBER(8,2),
    COST_UTILIZARE NUMBER(8,2),
    FOREIGN KEY (ID_UTILIZATOR) REFERENCES LICAAMALIA_UTILIZATORI(ID_UTILIZATOR),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_ISTORIC_UTILIZARE
```

```
CREATE TABLE
LICAAMALIA_ISTORIC_UTILIZARE (

    ID_ISTORIC NUMBER(6) PRIMARY KEY,

    ID_UTILIZATOR NUMBER(6),

    ID_TROTINETA NUMBER(6),

    DATA_UTILIZARE DATE,

    DURATA_UTILIZARE NUMBER(6),

    DISTANTA_PARCURSA NUMBER(8,2),

    COST_UTILIZARE NUMBER(8,2),

    FOREIGN KEY (ID_UTILIZATOR)
REFERENCES
LICAAMALIA_UTILIZATORI(ID_UTILIZATOR),

    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)

);

describe LICAAMALIA_ISTORIC_UTILIZARE
```

Script Output x

Task completed in 2,32 seconds

Table LICAAMALIA\_ISTORIC\_UTILIZARE created.

Name	Null?	Type
ID_ISTORIC	NOT NULL	NUMBER(6)
ID_UTILIZATOR		NUMBER(6)
ID_TROTINETA		NUMBER(6)
DATA_UTILIZARE		DATE
DURATA_UTILIZARE		NUMBER(6)
DISTANTA_PARCURSA		NUMBER(8,2)
COST_UTILIZARE		NUMBER(8,2)

FOREIGN KEY (ID\_TROTINETA) REFERENCES LICAAMALIA\_TROTINETE(ID\_TROTINETA)

);

describe LICAAMALIA\_ISTORIC\_UTILIZARE

- În fiecare tabel creat am adăugat înregistrări:

### --1. Adăugarea de noi înregistrări în tabela "TROTINETE"

```
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(1, 'E10', 'XIAOMI', 'BUNA', '70', 1.5, 'Piata Unirii');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(2, 'E12', 'XIAOMI', 'BUNA', '75', 1.7, 'Parcul Carol');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(3, 'ES4', 'SEGWAY', 'BUNA', '60', 1.6, 'Parcul Herastrau');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(4, 'M365', 'XIAOMI', 'OK', '80', 1.4, 'Parcul Tineretului');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(5, 'BOOSTED REV', 'BOOSTED', 'BUNA', '90', 2.0, 'Aleea Privighetorilor');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(6, 'CITYBUG 2S', 'CITYBUG', 'OK', '65', 1.8, 'Parcul Izvor');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(7, 'Ninebot MAX', 'SEGWAY', 'BUNA', '85', 1.9, 'Bulevardul Magheru');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(8, 'Zero 9', 'ZERO', 'BUNA', '70', 1.5, 'Parcul Circului');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(9, 'Hollyburn P5', 'HOLLYBURN', 'OK', '75', 1.6, 'Parcul Titan');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(10, 'Air T15', 'AIRWHEEL', 'BUNA', '80', 1.7, 'Bulevardul Unirii');

INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA ) VALUES
(11, 'GXL V2', 'GOTRAX', 'BUNA', '78', 1.6, 'Piata Victoriei');
```

```
SELECT* FROM LICAAMALIA_TROTINETE
```

```
ORDER BY ID_TROTINETA ASC;
```

Columns		Data	Model	Constraints	Grants	Statistics	Triggers	Flashback	Dependencies	Details	Partitions	Indexes	SQL
ID_TROTINETA	NUME_MODEL	PRODUCATOR	STARE	BATERIE_RAMASA	PRET_MINUT	LOCATIE_CURENTA							
1	E10	XIAOMI	BUNA	70	1,5	Piata Unirii							
2	E12	XIAOMI	BUNA	75	1,7	Parcul Carol							
3	ES4	SEGWAY	BUNA	60	1,6	Parcul Herastrau							
4	M365	XIAOMI	OK	80	1,4	Parcul Tineretului							
5	BOOSTED REV	BOOSTED	BUNA	90	2	Aleea Privighetorilor							
6	CITYBUG 2S	CITYBUG	OK	65	1,8	Parcul Izvor							
7	Ninebot MAX	SEGWAY	BUNA	85	1,9	Bulevardul Magheru							
8	Zero 9	ZERO	BUNA	70	1,5	Parcul Circului							
9	Hollyburn P5	HOLLYBURN	OK	75	1,6	Parcul Titan							
10	Air T15	AIRWHEEL	BUNA	80	1,7	Bulevardul Unirii							
11	GXL V2	GOTRAX	BUNA	78	1,6	Piata Victoriei							



## --2. Adăugarea de noi înregistrări în tabela "UTILIZATORI"

```

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(1, 'LICA', 'AMALIA', 'Str. Oltului 3', 'amalialica@gmail.com', '0771665319', 20);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(2, 'POPESCU', 'ION', 'Str. Muresului 7', 'ion.popescu@gmail.com', '0770123456', 25);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(3, 'IONESCU', 'MARIA', 'Str. Dorobantilor 15', 'maria.ionescu@gmail.com', '0770789123', 30);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(4, 'GEORGESCU', 'ANDREI', 'Str. Bucuresti 21', 'andrei.georgesc@gmail.com', '0770555678', 28);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(5, 'RADULESCU', 'ELENA', 'Str. Unirii 8', 'elena.radulescu@gmail.com', '0770998765', 22);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(6, 'POPA', 'MIHAI', 'Str. Timisului 12', 'mihai.popa@gmail.com', '0770888234', 26);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(7, 'STANCU', 'GABRIELA', 'Str. Clujului 5', 'gabriela.stancu@gmail.com', '0770333678', 32);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(8, 'CONSTANTIN', 'ALEXANDRU', 'Str. Iasi 19', 'alex.constantin@gmail.com', '0770111222', 29);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(9, 'DUMITRU', 'CRISTINA', 'Str. Vaslui 4', 'cris.dumitru@gmail.com', '0770456123', 23);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(10, 'BARBU', 'DANIEL', 'Str. Bihorului 11', 'daniel.barbu@gmail.com', '0770345876', 27);

INSERT INTO LICAAMALIA_UTILIZATORI(ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(11, 'VASILESCU', 'ANA', 'Str. Sibiu 2', 'ana.vasilescu@gmail.com', '0770998765', 21);

```

```

SELECT* FROM LICAAMALIA_UTILIZATORI
ORDER BY ID_UTILIZATOR ASC;

```

sgbd1 LICAAMALIA_UTILIZATORI							
Columns   Data   Model   Constraints   Grants   Statistics   Triggers   Flashback   Dependencies   Details   Partitions   Indexes   SQL							
	ID_UTILIZATOR	NUME	PRENUME	ADRESA	EMAIL	TELEFON	VARSTA
1	1	LICA	AMALIA	Str. Oltului 3	amalialica@gmail.com	0771665319	20
2	2	POPESCU	ION	Str. Muresului 7	ion.popescu@gmail.com	0770123456	25
3	3	IONESCU	MARIA	Str. Dorobantilor 15	maria.ionescu@gmail.com	0770789123	30
4	4	GEORGESCU	ANDREI	Str. Bucuresti 21	andrei.georgesc@gmail.com	0770555678	28
5	5	RADULESCU	ELENA	Str. Unirii 8	elena.radulescu@gmail.com	0770998765	22
6	6	POPA	MIHAI	Str. Timisului 12	mihai.popa@gmail.com	0770888234	26
7	7	STANCU	GABRIELA	Str. Clujului 5	gabriela.stancu@gmail.com	0770333678	32
8	8	CONSTANTIN	ALEXANDRU	Str. Iasi 19	alex.constantin@gmail.com	0770111222	29
9	9	DUMITRU	CRISTINA	Str. Vaslui 4	cris.dumitru@gmail.com	0770456123	23
10	10	BARBU	DANIEL	Str. Bihorului 11	daniel.barbu@gmail.com	0770345876	27
11	11	VASILESCU	ANA	Str. Sibiu 2	ana.vasilescu@gmail.com	0770998765	21

### --3. Adăugarea de noi înregistrări în tabela "INTRETINERE\_TROTINETE"

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(1, 3, TO_DATE('2023-01-01', 'YYYY-MM-DD'), 150.00, 'Baterie, Frâne', 'Ion Popescu');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(2, 7, TO_DATE('2023-02-15', 'YYYY-MM-DD'), 80.00, 'Acumulator', 'Maria Ionescu');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(3, 4, TO_DATE('2023-03-20', 'YYYY-MM-DD'), 50.00, 'Cablu electric', 'Andrei Georgescu');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(4, 9, TO_DATE('2023-04-10', 'YYYY-MM-DD'), 60.00, 'Roți', 'Elena Radulescu');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(5, 5, TO_DATE('2023-05-05', 'YYYY-MM-DD'), 45.00, 'Plăcuțe frână', 'Mihai Popa');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(6, 2, TO_DATE('2023-06-18', 'YYYY-MM-DD'), 120.00, 'Baterie, Sistem de direcție', 'Gabriela Stancu');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(7, 10, TO_DATE('2023-07-22', 'YYYY-MM-DD'), 30.00, 'Becuri', 'Alexandru Constantin');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(8, 6, TO_DATE('2023-08-12', 'YYYY-MM-DD'), 70.00, 'Plăcuțe frână', 'Cristina Dumitru');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(9, 8, TO_DATE('2023-09-28', 'YYYY-MM-DD'), 40.00, 'Amortizoare', 'Daniel Barbu');
```

```
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE(ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE,
TEHNICIAN_RESPONSABIL) VALUES
```

```
(10, 1, TO_DATE('2023-10-15', 'YYYY-MM-DD'), 25.00, 'Lampă', 'Ana Vasilescu');
```

```
SELECT* FROM LICAAMALIA_INTRETINERE_TROTINETE
```

```
ORDER BY ID_INTRETINERE ASC;
```

ID_INTRETINERE	ID_TROTINETA	DATA_INTRETINERE	TIP_INTRETINERE	COST_INTRETINERE	PIESE_INLOCUITE	TEHNICIAN_RESPONSABIL
1	1	01-01-2023	(null)	150	Baterie, Frâne	Ion Popescu
2	2	15-02-2023	(null)	80	Acumulator	Maria Ionescu
3	3	20-03-2023	(null)	50	Cablu electric	Andrei Georgescu
4	4	10-04-2023	(null)	60	Roți	Elena Radulescu
5	5	05-05-2023	(null)	45	Plăcuțe frână	Mihai Popa
6	6	18-06-2023	(null)	120	Baterie, Sistem de direcție	Gabriela Stancu
7	7	22-07-2023	(null)	30	Becuri	Alexandru Constantin
8	8	12-08-2023	(null)	70	Plăcuțe frână	Cristina Dumitru
9	9	28-09-2023	(null)	40	Amortizoare	Daniel Barbu
10	10	15-10-2023	(null)	25	Lampă	Ana Vasilescu

#### --4. Adăugarea de noi înregistrări în tabela "RAPORT\_TROTINETE"

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (1, 3, TO_DATE('2023-01-05', 'YYYY-MM-DD'), 10.3, 80.0);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (2, 7, TO_DATE('2023-02-10', 'YYYY-MM-DD'), 15.8, 75.5);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (3, 4, TO_DATE('2023-03-15', 'YYYY-MM-DD'), 20.1, 85.2);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (4, 9, TO_DATE('2023-04-20', 'YYYY-MM-DD'), 11, 70.0);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (5, 5, TO_DATE('2023-05-25', 'YYYY-MM-DD'), 5, 78.5);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (6, 2, TO_DATE('2023-06-30', 'YYYY-MM-DD'), 4.9, 90.0);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (7, 10, TO_DATE('2023-07-05', 'YYYY-MM-DD'), 8, 82.3);
```

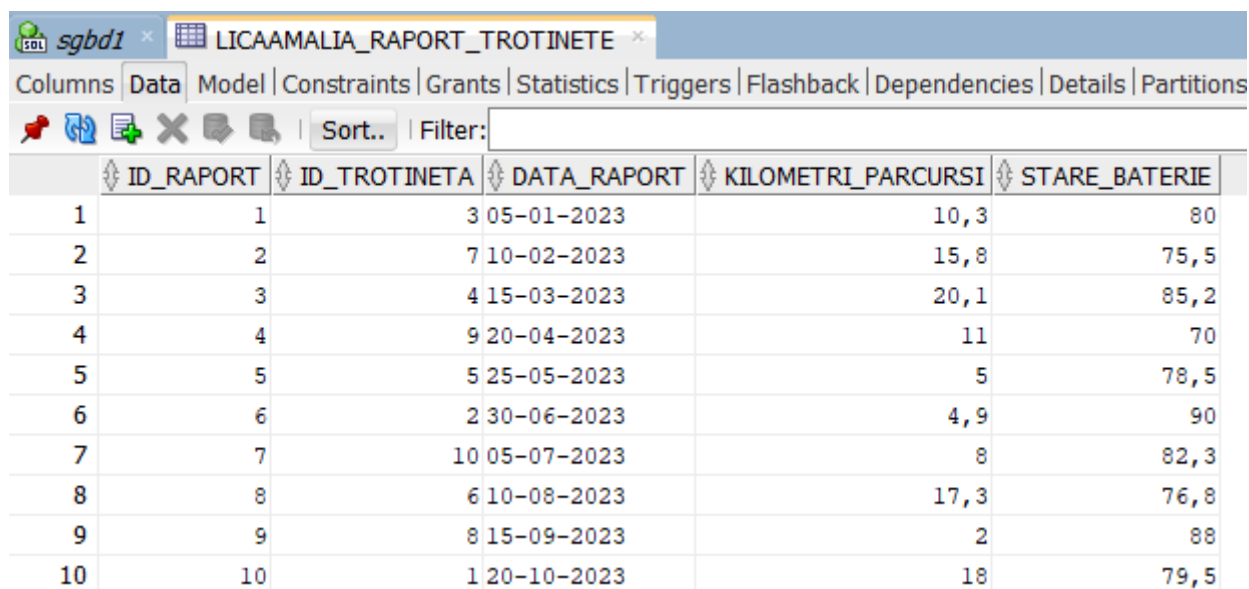
```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (8, 6, TO_DATE('2023-08-10', 'YYYY-MM-DD'), 17.3, 76.8);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (9, 8, TO_DATE('2023-09-15', 'YYYY-MM-DD'), 2, 88.0);
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE (ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES (10, 1, TO_DATE('2023-10-20', 'YYYY-MM-DD'), 18, 79.5);
```

```
SELECT* FROM LICAAMALIA_RAPORT_TROTINETE
```

```
ORDER BY ID_RAPORT ASC;
```



The screenshot shows a database management interface with a table named "LICAAMALIA\_RAPORT\_TROTINETE". The table has five columns: ID\_RAPORT, ID\_TROTINETA, DATA\_RAPORT, KILOMETRI\_PARCURSI, and STARE\_BATERIE. The data is displayed in a grid with 10 rows, numbered 1 to 10. The interface includes tabs for Columns, Data, Model, Constraints, Grants, Statistics, Triggers, Flashback, Dependencies, Details, and Partitions. The Data tab is selected, and the table is sorted by ID\_RAPORT in ascending order.

	ID_RAPORT	ID_TROTINETA	DATA_RAPORT	KILOMETRI_PARCURSI	STARE_BATERIE
1	1	3	05-01-2023	10,3	80
2	2	7	10-02-2023	15,8	75,5
3	3	4	15-03-2023	20,1	85,2
4	4	9	20-04-2023	11	70
5	5	5	25-05-2023	5	78,5
6	6	2	30-06-2023	4,9	90
7	7	10	05-07-2023	8	82,3
8	8	6	10-08-2023	17,3	76,8
9	9	8	15-09-2023	2	88
10	10	1	20-10-2023	18	79,5

### --5. Adăugarea de noi înregistrări în tabela "INCHIRIERI"

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (1, 3, 2, TO_TIMESTAMP('2023-01-05 10:30:00', 'YYYY-MM-DD HH24:MI:SS'), 2, 10.50);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (2, 7, 6, TO_TIMESTAMP('2023-02-10 15:45:00', 'YYYY-MM-DD HH24:MI:SS'), 1, 7.20);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (3, 4, 8, TO_TIMESTAMP('2023-03-15 12:00:00', 'YYYY-MM-DD HH24:MI:SS'), 3, 18.90);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (4, 9, 1, TO_TIMESTAMP('2023-04-20 09:15:00', 'YYYY-MM-DD HH24:MI:SS'), 4, 26.40);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (5, 5, 5, TO_TIMESTAMP('2023-05-25 17:30:00', 'YYYY-MM-DD HH24:MI:SS'), 1, 8.50);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (6, 2, 9, TO_TIMESTAMP('2023-06-30 14:00:00', 'YYYY-MM-DD HH24:MI:SS'), 2, 15.60);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (7, 10, 3, TO_TIMESTAMP('2023-07-05 11:45:00', 'YYYY-MM-DD HH24:MI:SS'), 3, 21.30);
```

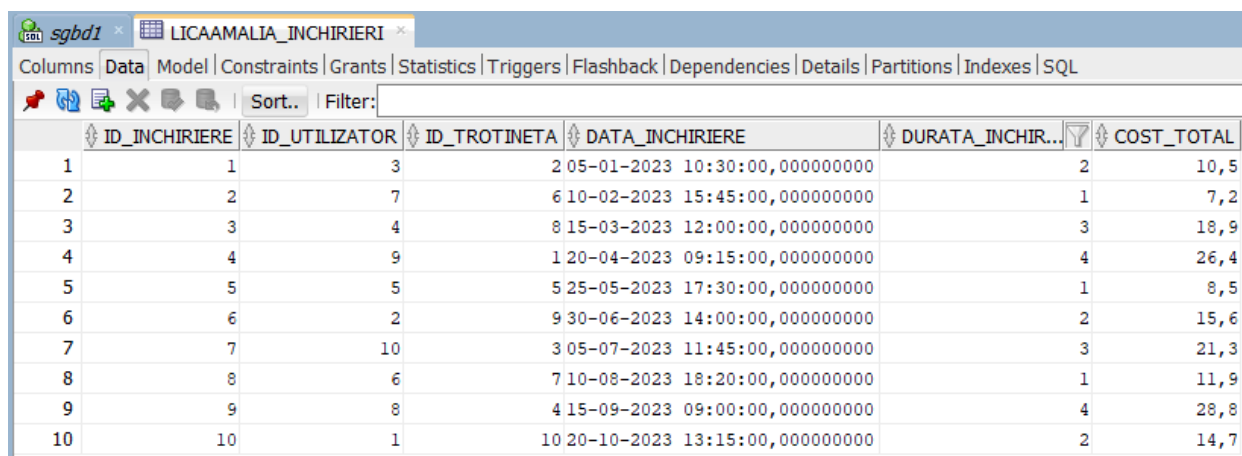
```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (8, 6, 7, TO_TIMESTAMP('2023-08-10 18:20:00', 'YYYY-MM-DD HH24:MI:SS'), 1, 11.90);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (9, 8, 4, TO_TIMESTAMP('2023-09-15 09:00:00', 'YYYY-MM-DD HH24:MI:SS'), 4, 28.80);
```

```
INSERT INTO LICAAMALIA_INCHIRIERI (ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES (10, 1, 10, TO_TIMESTAMP('2023-10-20 13:15:00', 'YYYY-MM-DD HH24:MI:SS'), 2, 14.70);
```

```
SELECT* FROM LICAAMALIA_INCHIRIERI
```

```
ORDER BY ID_INCHIRIERE ASC;
```



ID_INCHIRIERE	ID_UTILIZATOR	ID_TROTINETA	DATA_INCHIRIERE	DURATA_INCHIRIERE	COST_TOTAL
1	1	3	2023-01-05 10:30:00	2	10,5
2	2	7	2023-02-10 15:45:00	1	7,2
3	3	4	2023-03-15 12:00:00	3	18,9
4	4	9	2023-04-20 09:15:00	4	26,4
5	5	5	2023-05-25 17:30:00	1	8,5
6	6	2	2023-06-30 14:00:00	2	15,6
7	7	10	2023-07-05 11:45:00	3	21,3
8	8	6	2023-08-10 18:20:00	1	11,9
9	9	8	2023-09-15 09:00:00	4	28,8
10	10	1	2023-10-20 13:15:00	2	14,7

## --6.Adăugarea de noi înregistrări în tabela "ISTORIC\_UTILIZARE"

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (1, 3, 2, TO_DATE('2023-01-05', 'YYYY-MM-DD'), 2, 5.8, 8.20);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (2, 7, 6, TO_DATE('2023-02-10', 'YYYY-MM-DD'), 1, 3.5, 4.50);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (3, 4, 8, TO_DATE('2023-03-15', 'YYYY-MM-DD'), 3, 10.2, 14.70);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (4, 9, 1, TO_DATE('2023-04-20', 'YYYY-MM-DD'), 4, 15.0, 20.40);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (5, 5, 5, TO_DATE('2023-05-25', 'YYYY-MM-DD'), 1, 4.1, 6.80);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (6, 2, 9, TO_DATE('2023-06-30', 'YYYY-MM-DD'), 2, 7.3, 10.50);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (7, 10, 3, TO_DATE('2023-07-05', 'YYYY-MM-DD'), 3, 9.8, 13.20);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (8, 6, 7, TO_DATE('2023-08-10', 'YYYY-MM-DD'), 1, 5.5, 7.90);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (9, 8, 4, TO_DATE('2023-09-15', 'YYYY-MM-DD'), 4, 14.7, 19.80);
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE (ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES (10, 1, 10, TO_DATE('2023-10-20', 'YYYY-MM-DD'), 2, 6.2, 9.30);
```

```
SELECT* FROM LICAAMALIA_ISTORIC_UTILIZARE
```

```
ORDER BY ID_ISTORIC ASC;
```

ID_ISTORIC	ID_UTILIZATOR	ID_TROTINETA	DATA_UTILIZARE	DURATA_UTILIZARE	DISTANTA_PARCURSA	COST_UTILIZARE
1	1	3	2 05-01-2023	2	5, 8	8, 2
2	2	7	6 10-02-2023	1	3, 5	4, 5
3	3	4	8 15-03-2023	3	10, 2	14, 7
4	4	9	1 20-04-2023	4	15	20, 4
5	5	5	5 25-05-2023	1	4, 1	6, 8
6	6	2	9 30-06-2023	2	7, 3	10, 5
7	7	10	3 05-07-2023	3	9, 8	13, 2
8	8	6	7 10-08-2023	1	5, 5	7, 9
9	9	8	4 15-09-2023	4	14, 7	19, 8
10	10	1	10 20-10-2023	2	6, 2	9, 3

## E. Blocuri PL/SQL conținând structuri de control variate

### 1. Utilizare FOR LOOP- Afișarea tuturor trotinetelor

Iterează prin toate trotinetele din tabel și afișează ID-ul, modelul și producătorul acestora.

```
SET SERVEROUTPUT ON;
```

```
BEGIN

    FOR v_trotineta IN

        (SELECT ID_TROTINETA,
            NUME_MODEL, PRODUCATOR FROM
            LICAAMALIA_TROTINETE)

        LOOP

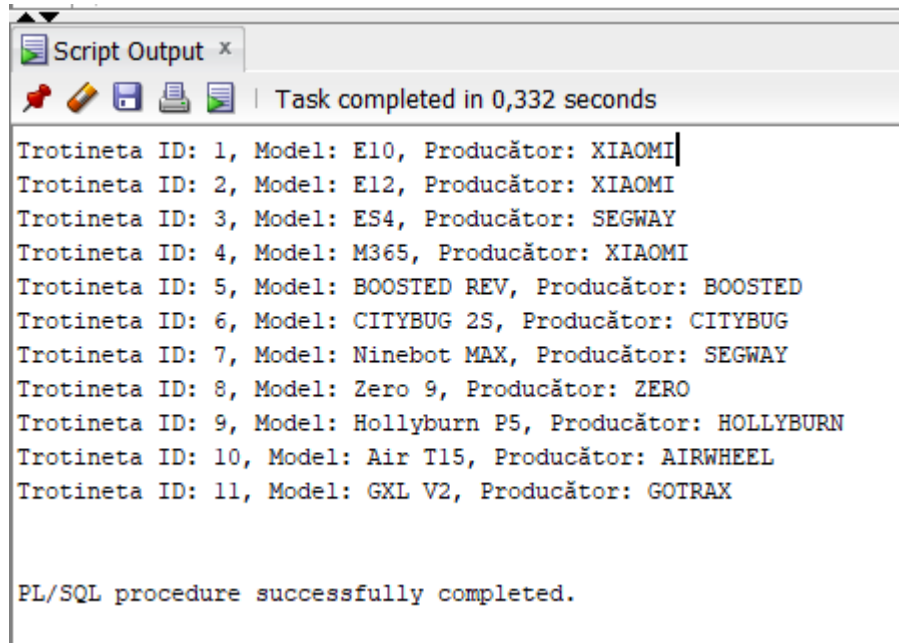
            DBMS_OUTPUT.PUT_LINE('Trotineta
ID: ' || v_trotineta.ID_TROTINETA ||

                                ', Model: ' ||
v_trotineta.NUME_MODEL ||

                                ', Producător: ' ||
v_trotineta.PRODUCATOR);

        END LOOP;

END;
```



```
Script Output x
Task completed in 0,332 seconds

Trotineta ID: 1, Model: E10, Producător: XIAOMI
Trotineta ID: 2, Model: E12, Producător: XIAOMI
Trotineta ID: 3, Model: ES4, Producător: SEGWAY
Trotineta ID: 4, Model: M365, Producător: XIAOMI
Trotineta ID: 5, Model: BOOSTED REV, Producător: BOOSTED
Trotineta ID: 6, Model: CITYBUG 2S, Producător: CITYBUG
Trotineta ID: 7, Model: Ninebot MAX, Producător: SEGWAY
Trotineta ID: 8, Model: Zero 9, Producător: ZERO
Trotineta ID: 9, Model: Hollyburn P5, Producător: HOLLYBURN
Trotineta ID: 10, Model: Air T15, Producător: AIRWHEEL
Trotineta ID: 11, Model: GXL V2, Producător: GOTRAX

PL/SQL procedure successfully completed.
```

### 2. Utilizare WHILE LOOP-Calcularea distanței totale parcurse

Calculează și afișează distanța totală parcursă utilizând informațiile din istoricul utilizării trotinetelor.

```
SET SERVEROUTPUT ON;
```

```
DECLARE

    v_total_distanța NUMBER := 0;

    v_index NUMBER := 1;

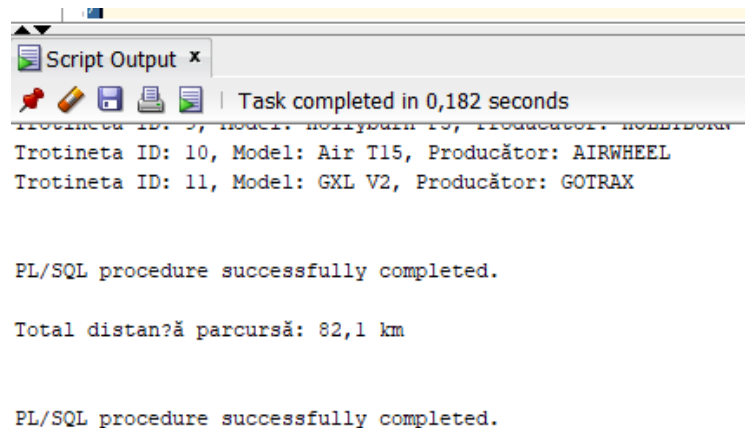
    v_max_id NUMBER;

    v_distanța NUMBER;

BEGIN

    SELECT MAX(ID_ISTORIC) INTO v_max_id
    FROM LICAAMALIA_ISTORIC_UTILIZARE;

    WHILE v_index <= v_max_id LOOP
```



```
Script Output x
Task completed in 0,182 seconds

Trotineta ID: 9, Model: Hollyburn P5, Producător: HOLLYBURN
Trotineta ID: 10, Model: Air T15, Producător: AIRWHEEL
Trotineta ID: 11, Model: GXL V2, Producător: GOTRAX

PL/SQL procedure successfully completed.

Total distanță parcursă: 82,1 km

PL/SQL procedure successfully completed.
```

```

SELECT NVL(DISTANTA_PARCURSA, 0) INTO v_distanta
FROM LICAAMALIA_ISTORIC_UTILIZARE
WHERE ID_ISTORIC = v_index;

v_total_distanta := v_total_distanta + v_distanta;
v_index := v_index + 1;
END LOOP;

DBMS_OUTPUT.PUT_LINE('Total distanță parcursă: ' || v_total_distanta || ' km');
END;
/

```

### 3. Utilizare cursor- *Top 5 trotinete după nivelul bateriei*

*Afișează primele 5 trotinete cu cel mai mare procent de baterie rămasă.*

```

SET SERVEROUTPUT ON;

DECLARE

CURSOR c IS

    SELECT ID_TROTINETA, NUME_MODEL,
    BATERIE_RAMASA

    FROM LICAAMALIA_TROTINETE

    ORDER BY BATERIE_RAMASA DESC;

v_count NUMBER := 0;
v_trotineta c%ROWTYPE;
BEGIN

OPEN c;

LOOP

    FETCH c INTO v_trotineta;

    EXIT WHEN c%NOTFOUND OR v_count = 5;

    DBMS_OUTPUT.PUT_LINE('Trotineta ID: ' || v_trotineta.ID_TROTINETA ||

        ', Model: ' || v_trotineta.NUME_MODEL ||

        ', Baterie rămasă: ' || v_trotineta.BATERIE_RAMASA || '%');

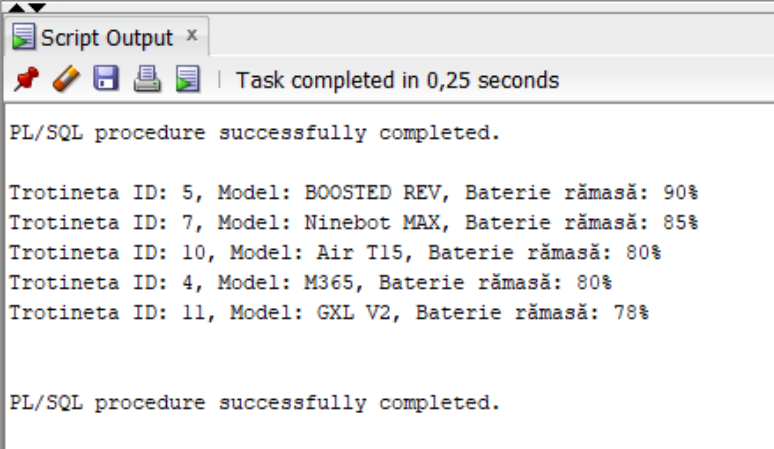
    v_count := v_count + 1;

END LOOP;

CLOSE c;

END;
/

```



```

Script Output x
Task completed in 0,25 seconds

PL/SQL procedure successfully completed.

Trotineta ID: 5, Model: BOOSTED REV, Baterie rămasă: 90%
Trotineta ID: 7, Model: Ninebot MAX, Baterie rămasă: 85%
Trotineta ID: 10, Model: Air T15, Baterie rămasă: 80%
Trotineta ID: 4, Model: M365, Baterie rămasă: 80%
Trotineta ID: 11, Model: GXL V2, Baterie rămasă: 78%

PL/SQL procedure successfully completed.

```

#### 4. Utilizare IF/ELSE- Categoria de vârstă a utilizatorilor

Afișează informațiile despre un utilizator cu o închiriere de peste 3 ore și determină categoria de vârstă a acestuia.

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_nume_complet VARCHAR2(100);
```

```
    v_varsta NUMBER;
```

```
    v_durata_inchiriere NUMBER;
```

```
BEGIN
```

```
    SELECT u.NUME || ' ' || u.PRENUME, u.VARSTA,  
           i.DURATA_INCHIRIERE
```

```
    INTO v_nume_complet, v_varsta, v_durata_inchiriere
```

```
    FROM LICAAMALIA_UTILIZATORI u
```

```
    JOIN LICAAMALIA_INCHIRIERI i ON u.ID_UTILIZATOR = i.ID_UTILIZATOR
```

```
    WHERE i.DURATA_INCHIRIERE > 3 AND ROWNUM = 1; -- Luăm doar primul utilizator
```

```
    DBMS_OUTPUT.PUT_LINE('Utilizator: ' || v_nume_complet ||  
                          ', Durată închiriere: ' || v_durata_inchiriere || ' ore');
```

```
    IF v_varsta < 25 THEN
```

```
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Tânăr');
```

```
    ELSIF v_varsta BETWEEN 25 AND 35 THEN
```

```
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Adult');
```

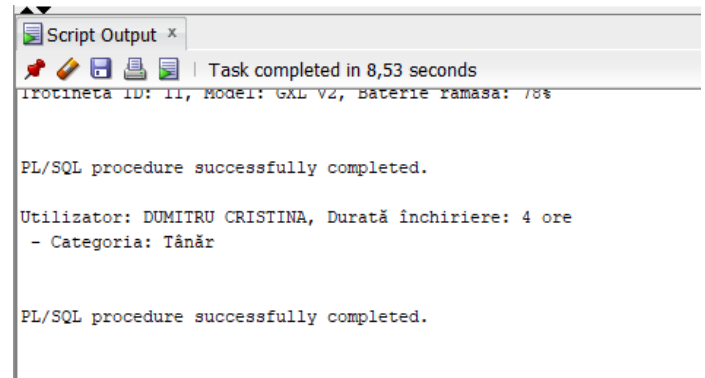
```
    ELSE
```

```
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Senior');
```

```
    END IF;
```

```
END;
```

```
/
```



```
Script Output x
Task completed in 8,53 seconds
Proiect ID: 11, Model: GXL V2, Baterie ramasa: 78%

PL/SQL procedure successfully completed.

Utilizator: DUMITRU CRISTINA, Durată închiriere: 4 ore
- Categoria: Tânăr

PL/SQL procedure successfully completed.
```

#### 5. Utilizare FOR LOOP - Cost total și clasificarea utilizatorilor

Calculează costul total al închirierilor pentru fiecare utilizator și îi clasifică drept "Client VIP" sau "Client obișnuit" în funcție de cheltuieli.

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    CURSOR c IS
```

```
    SELECT u.ID_UTILIZATOR, u.NUME || ' ' || u.PRENUME AS NUME_COMPLET, SUM(i.COST_TOTAL) AS  
    TOTAL_COST
```

```
    FROM LICAAMALIA_UTILIZATORI u
```



```

JOIN LICAAMALIA_INCHIRIERI i ON u.ID_UTILIZATOR = i.ID_UTILIZATOR
GROUP BY u.ID_UTILIZATOR, u.NUME, u.PRENUME;

```

```

BEGIN

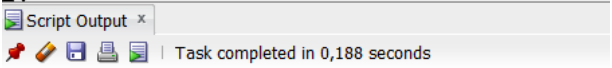
```

```

FOR v_utilizator IN c LOOP
    DBMS_OUTPUT.PUT_LINE('Utilizator: ' ||
v_utilizator.NUME_COMPLET ||
        ', Cost total închirieri: ' ||
v_utilizator.TOTAL_COST);

    IF v_utilizator.TOTAL_COST > 50 THEN
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Client VIP');
    ELSE
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Client obisnuit');
    END IF;
END LOOP;
END;
/

```



```

Utilizator: LICA AMALIA, Cost total închirieri: 14,7
- Categoria: Client obisnuit
Utilizator: POPESCU ION, Cost total închirieri: 15,6
- Categoria: Client obisnuit
Utilizator: IONESCU MARIA, Cost total închirieri: 10,5
- Categoria: Client obisnuit
Utilizator: GEORGESCU ANDREI, Cost total închirieri: 18,9
- Categoria: Client obisnuit
Utilizator: RADULESCU ELENA, Cost total închirieri: 8,5
- Categoria: Client obisnuit
Utilizator: POPA MIHAI, Cost total închirieri: 11,9
- Categoria: Client obisnuit
Utilizator: STANCU GABRIELA, Cost total închirieri: 7,2
- Categoria: Client obisnuit
Utilizator: CONSTANTIN ALEXANDRU, Cost total închirieri: 28,8
- Categoria: Client obisnuit
Utilizator: DUMITRU CRISTINA, Cost total închirieri: 26,4
- Categoria: Client obisnuit
Utilizator: BARBU DANIEL, Cost total închirieri: 21,3
- Categoria: Client obisnuit

PL/SQL procedure successfully completed.

```

## F. Utilizarea cursorilor și a excepțiilor în cadrul blocurilor PL/SQL

### I. CURSORI

1. *Cursor explicit OPEN-LOOP-FETCH-CLOSE - Afișarea numelui complet, vârstei și adresei utilizatorilor: Folosește un cursor explicit pentru a parcurge rândurile din tabelul utilizatorilor și afișează numele complet, vârsta și adresa fiecărui utilizator.*

```

SET SERVEROUTPUT ON;

DECLARE

CURSOR c IS

    SELECT NUME || ' ' || PRENUME AS NUME_COMPLET, VARSTA, ADRESA

    FROM LICAAMALIA_UTILIZATORI;

v c%ROWTYPE;

BEGIN

    OPEN c;

    LOOP

        FETCH c INTO v;

        EXIT WHEN c%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Nume complet: ' || v.NUME_COMPLET || ', Vârsta: ' || v.VARSTA || ', Adresa: ' ||
v.ADRESA);

```

```

END LOOP;

CLOSE c;

END;

```

```

/
SET SERVEROUTPUT ON;

DECLARE
    CURSOR c IS
        SELECT NUME || ' ' || PRENUME AS NUME_COMPLET, VARSTA, ADRESA
        FROM LICAAMALIA_UTILIZATORI;
    v c%ROWTYPE;
BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v;
        EXIT WHEN c%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Nume complet: ' || v.NUME_COMPLET || ', Vârsta: ' || v.VARSTA || ', Adresa: ' || v.ADRESA);
    END LOOP;
    CLOSE c;
END;
/

```

Script Output x

Task completed in 1,373 seconds

```

Nume complet: LICA AMALIA, Vârsta: 20, Adresa: Str. Oltului 3
Nume complet: POPESCU ION, Vârsta: 25, Adresa: Str. Muresului 7
Nume complet: IONESCU MARIA, Vârsta: 30, Adresa: Str. Dorobantilor 15
Nume complet: GEORGESCU ANDREI, Vârsta: 28, Adresa: Str. Bucuresti 21
Nume complet: RADULESCU ELENA, Vârsta: 22, Adresa: Str. Unirii 8
Nume complet: POPA MIHAI, Vârsta: 26, Adresa: Str. Timisului 12
Nume complet: STANCU GABRIELA, Vârsta: 32, Adresa: Str. Clujului 5
Nume complet: CONSTANTIN ALEXANDRU, Vârsta: 29, Adresa: Str. Iasi 19
Nume complet: DUMITRU CRISTINA, Vârsta: 23, Adresa: Str. Vaslui 4
Nume complet: BARBU DANIEL, Vârsta: 27, Adresa: Str. Bihorului 11
Nume complet: VASILESCU ANA, Vârsta: 21, Adresa: Str. Sibiu 2

```

## 2. Cursor explicit FOR-LOOP - Afişarea trotinetei şi numărului total de închirieri:

Utilizează un cursor explicit cu FOR-LOOP pentru a parcurge trotinetele şi a afişa ID-ul fiecărei trotinete împreună cu numărul total de închirieri asociate.

```

SET SERVEROUTPUT ON;

```

```

DECLARE

    CURSOR c IS

        SELECT ID_TROTINETA, COUNT(*) AS
        NUMAR_INCHIRIERI

        FROM LICAAMALIA_INCHIRIERI

        GROUP BY ID_TROTINETA;

BEGIN

    FOR v IN c LOOP

        DBMS_OUTPUT.PUT_LINE('ID Trotineta: ' ||
        v.ID_TROTINETA || ' - Număr Închirieri: ' ||
        v.NUMAR_INCHIRIERI);

    END LOOP;

END;

/

```

```

ID Trotineta: 2 - Număr Închirieri: 1
ID Trotineta: 6 - Număr Închirieri: 1
ID Trotineta: 8 - Număr Închirieri: 1
ID Trotineta: 1 - Număr Închirieri: 1
ID Trotineta: 5 - Număr Închirieri: 1
ID Trotineta: 9 - Număr Închirieri: 1
ID Trotineta: 3 - Număr Închirieri: 1
ID Trotineta: 7 - Număr Închirieri: 1
ID Trotineta: 4 - Număr Închirieri: 1
ID Trotineta: 10 - Număr Închirieri: 1

PL/SQL procedure successfully completed.

```

3. *Cursor explicit FOR-LOOP - Afișarea primelor 3 trotinete cu cel mai mare cost total de utilizare:* Folosește un cursor explicit cu FOR-LOOP pentru a afișa primele 3 trotinete ordonate descrescător după costul total de utilizare.

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
CURSOR c IS
```

```
SELECT ID_TROTINETA, SUM(COST_UTILIZARE) AS COST_TOTAL
```

```
FROM LICAAMALIA_ISTORIC_UTILIZARE
```

```
GROUP BY ID_TROTINETA
```

```
ORDER BY COST_TOTAL DESC
```

```
FETCH FIRST 3 ROWS ONLY;
```

```
BEGIN
```

```
FOR v IN c LOOP
```

```
DBMS_OUTPUT.PUT_LINE('ID  
Trotineta: ' || v.ID_TROTINETA || ' -  
Cost Total: ' || v.COST_TOTAL);
```

```
END LOOP;
```

```
END;
```

```
/
```

```
ID Trotineta: 1 - Cost Total: 20,4
```

```
ID Trotineta: 4 - Cost Total: 19,8
```

```
ID Trotineta: 8 - Cost Total: 14,7
```

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

4. *Cursor explicit OPEN-FETCH-CLOSE - Actualizarea trotinetei cu cel mai mic cost de întreținere:* Utilizează un cursor explicit pentru a selecta trotineta cu cel mai mic cost de întreținere și actualizează starea acesteia la "BUNĂ".

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
CURSOR c IS
```

```
SELECT ID_TROTINETA
```

```
FROM LICAAMALIA_INTRETINERE_TROTINETE
```

```
ORDER BY COST_INTRETINERE ASC
```

```
FETCH FIRST 1 ROW ONLY;
```

```
v_trotineta_id LICAAMALIA_INTRETINERE_TROTINETE.ID_TROTINETA%TYPE;
```

```
BEGIN
```

```
OPEN c;
```

```

FETCH c INTO v_trotineta_id;

CLOSE c;

UPDATE LICAAMALIA_TROTINETE

SET STARE = 'BUNA'

WHERE ID_TROTINETA = v_trotineta_id;

```

```

Trotineta cu ID 1 a fost actualizată la stare BUNĂ.

PL/SQL procedure successfully completed.

```

```

DBMS_OUTPUT.PUT_LINE('Trotineta cu ID ' || v_trotineta_id || ' a fost actualizată la stare BUNĂ.');
```

END;

/

5. *Cursor implicit - Actualizarea trotinetelor cu baterie sub 50%: Actualizează starea trotinetelor cu bateria rămasă sub 50% și utilizează attributele cursorului implicit pentru a afișa numărul total de trotinete afectate.*

```

SET SERVEROUTPUT ON;

BEGIN

    UPDATE
    LICAAMALIA_TROTINETE

    SET STARE = 'NECESITĂ ÎNCĂRCARE'

    WHERE BATERIE_RAMASA < 50;

    IF SQL%FOUND THEN

        DBMS_OUTPUT.PUT_LINE('S-au
actualizat ' || SQL%ROWCOUNT || '
trotinete cu bateria sub 50%.');

    ELSE

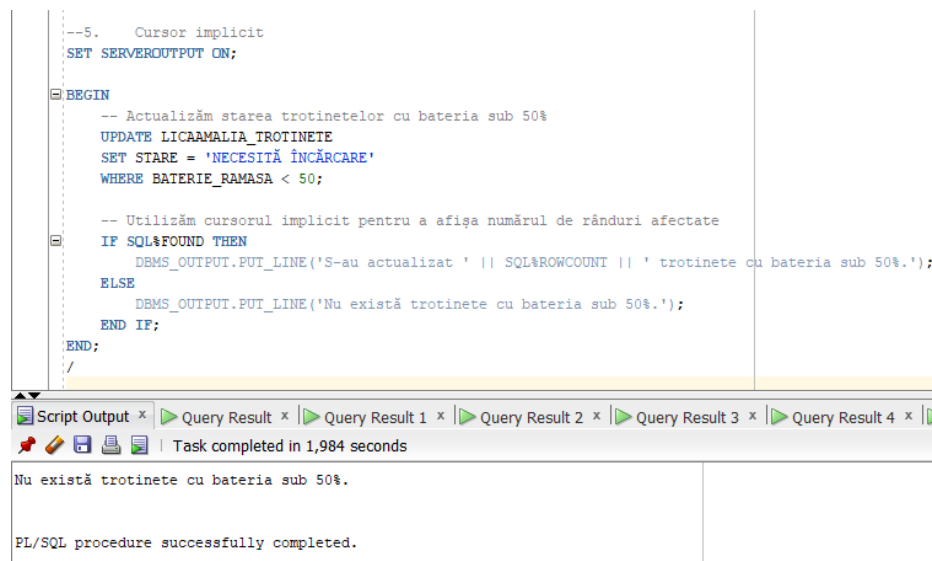
        DBMS_OUTPUT.PUT_LINE('Nu există trotinete cu bateria sub 50%.');

    END IF;

END;

/

```



```

--5. Cursor implicit
SET SERVEROUTPUT ON;

BEGIN
    -- Actualizăm starea trotinetelor cu bateria sub 50%
    UPDATE LICAAMALIA_TROTINETE
    SET STARE = 'NECESITĂ ÎNCĂRCARE'
    WHERE BATERIE_RAMASA < 50;

    -- Utilizăm cursorul implicit pentru a afișa numărul de rânduri afectate
    IF SQL%FOUND THEN
        DBMS_OUTPUT.PUT_LINE('S-au actualizat ' || SQL%ROWCOUNT || ' trotinete cu bateria sub 50%.');
    ELSE
        DBMS_OUTPUT.PUT_LINE('Nu există trotinete cu bateria sub 50%.');
    END IF;
END;
/

```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x |

Task completed in 1,984 seconds

Nu există trotinete cu bateria sub 50%.

PL/SQL procedure successfully completed.

## II. EXCEPȚII

1. *Excepție implicită (NO\_DATA\_FOUND) - Verificarea existenței unui utilizator după adresă:* Caută un utilizator după o adresă specificată. Dacă nu există niciun utilizator la acea adresă, excepția implicită NO\_DATA\_FOUND este tratată și se afișează un mesaj corespunzător.

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_num_e_utilizator LICAAMALIA_UTILIZATORI.NUME%TYPE;
```

```
BEGIN
```

```
    SELECT NUME INTO v_num_e_utilizator
```

```
    FROM LICAAMALIA_UTILIZATORI
```

```
    WHERE ADRESA = 'Str. Inexistenta';
```

```
    DBMS_OUTPUT.PUT_LINE('Utilizator găsit: ' ||  
v_num_e_utilizator);
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
```

```
        DBMS_OUTPUT.PUT_LINE('Nu există niciun  
utilizator la această adresă.');
```

```
END;
```

```
/
```



```
--1. Excepție implicită (NO_DATA_FOUND)  
SET SERVEROUTPUT ON;  
  
DECLARE  
    v_num_e_utilizator LICAAMALIA_UTILIZATORI.NUME%TYPE;  
BEGIN  
    SELECT NUME INTO v_num_e_utilizator  
    FROM LICAAMALIA_UTILIZATORI  
    WHERE ADRESA = 'Str. Inexistenta';  
  
    DBMS_OUTPUT.PUT_LINE('Utilizator găsit: ' || v_num_e_utilizator);  
  
EXCEPTION  
    WHEN NO_DATA_FOUND THEN  
        DBMS_OUTPUT.PUT_LINE('Nu există niciun utilizator la această adresă.');
```

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

```
Nu există niciun utilizator la această adresă.
```

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

2. *Excepție explicită (RAISE) - Verificarea trotinetelor cu baterie slabă:* Verifică dacă există trotinete cu bateria sub 65%. Dacă nu sunt găsite astfel de trotinete, o excepție explicită definită de utilizator este declanșată și tratată pentru a afișa un mesaj corespunzător.

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    baterie_slabă EXCEPTION;
```

```
    v_count NUMBER;
```

```
BEGIN
```

```

SELECT COUNT(*) INTO v_count
FROM LICAAMALIA_TROTINETE
WHERE BATERIE_RAMASA < 65;

IF v_count = 0 THEN
    RAISE baterie_slabă;
ELSE
    DBMS_OUTPUT.PUT_LINE('Există ' || v_count
|| ' trotinete cu baterie slabă.');
```

END IF;

EXCEPTION

WHEN baterie\_slabă THEN

DBMS\_OUTPUT.PUT\_LINE('Nu există  
trotinete cu baterie mai mică de 65%.');

END;

/

```

--2. Excepție explicită (RAISE)
SET SERVEROUTPUT ON;

DECLARE
    baterie_slabă EXCEPTION;
    v_count NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_count
    FROM LICAAMALIA_TROTINETE
    WHERE BATERIE_RAMASA < 65;

    IF v_count = 0 THEN
        RAISE baterie_slabă;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Există ' || v_count || ' trotinete cu baterie slabă.');
```

END IF;

EXCEPTION

WHEN baterie\_slabă THEN

DBMS\_OUTPUT.PUT\_LINE('Nu există trotinete cu baterie mai mică de 65%.');

END;

/

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Q

Task completed in 0,214 seconds

PL/SQL procedure successfully completed.

Există 1 trotinete cu baterie slabă.

PL/SQL procedure successfully completed.

### 3. Excepție generală (WHEN OTHERS) - Actualizarea prețului pe minut pentru trotinetele Xiaomi: Actualizează prețul pe minut pentru trotinetele producătorului Xiaomi. În cazul apariției oricărei erori neașteptate, excepția generală WHEN OTHERS capturează eroarea și afișează un mesaj detaliat cu explicația.

SET SERVEROUTPUT ON;

```

BEGIN
    UPDATE LICAAMALIA_TROTINETE
    SET PRET_MINUT = PRET_MINUT + 0.1
    WHERE PRODUCATOR = 'XIAOMI';
```

```

Pretul pe minut a fost actualizat pentru trotinetele Xiaomi.

PL/SQL procedure successfully completed.
```

```

DBMS_OUTPUT.PUT_LINE('Pretul pe minut a fost actualizat pentru trotinetele Xiaomi.');
```

EXCEPTION

WHEN OTHERS THEN

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

END;

/

4. *Excepție predefinită (TOO\_MANY\_ROWS) - Găsirea trotinetelor de la un producător specific: Caută o trotinetă produsă de un producător specific. Dacă sunt găsite mai multe trotinete ale aceluiași producător, excepția implicită TOO\_MANY\_ROWS este tratată, iar un mesaj informativ este afișat.*

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
v_trotineta_id LICAAMALIA_TROTINETE.ID_TROTINETA%TYPE;
```

```
BEGIN
```

```
SELECT ID_TROTINETA INTO v_trotineta_id  
FROM LICAAMALIA_TROTINETE  
WHERE PRODUCATOR = 'XIAOMI';
```

```
DBMS_OUTPUT.PUT_LINE('Trotineta găsită:  
' || v_trotineta_id);
```

```
EXCEPTION
```

```
WHEN TOO_MANY_ROWS THEN
```

```
DBMS_OUTPUT.PUT_LINE('Există mai  
multe trotinete pentru acest producător.');
```

```
END;
```

```
/
```

```
--4. Excepție predefinită (TOO_MANY_ROWS)  
SET SERVEROUTPUT ON;  
  
DECLARE  
    v_trotineta_id LICAAMALIA_TROTINETE.ID_TROTINETA%TYPE;  
BEGIN  
    SELECT ID_TROTINETA INTO v_trotineta_id  
    FROM LICAAMALIA_TROTINETE  
    WHERE PRODUCATOR = 'XIAOMI';  
  
    DBMS_OUTPUT.PUT_LINE('Trotineta găsită: ' || v_trotineta_id);  
  
EXCEPTION  
    WHEN TOO_MANY_ROWS THEN  
        DBMS_OUTPUT.PUT_LINE('Există mai multe trotinete pentru acest producător.');
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x  
Task completed in 0,2 seconds

PL/SQL procedure successfully completed.

Există mai multe trotinete pentru acest producător.

PL/SQL procedure successfully completed.

## G. Funcții, proceduri

### I. PROCEDURI

**1. Calculare cost total întreținere:** Calculează și afișează costul total al întreținerii pentru toate trotinetele, utilizând datele din tabelul LICAAMALIA\_INTRETINERE\_TROTINETE.

```
CREATE OR REPLACE PROCEDURE Calculare_Cost_Total_Intretinere IS
```

```
v_total_cost NUMBER := 0;
```

```
BEGIN
```

```
SELECT SUM(COST_INTRETINERE) INTO v_total_cost  
FROM LICAAMALIA_INTRETINERE_TROTINETE;
```

```
DBMS_OUTPUT.PUT_LINE('Costul total al întreținerii este: ' || v_total_cost || ' lei.');
```

```

END;

/

EXECUTE
Calculare_Cost_Total_Intretinere;

```

```

--PROCEDURE
--1. Calculare cost total intretinere
CREATE OR REPLACE PROCEDURE Calculare_Cost_Total_Intretinere IS
    v_total_cost NUMBER := 0;
BEGIN
    SELECT SUM(COST_INTRETINERE) INTO v_total_cost
    FROM LICAAMALIA_INTRETINERE_TROTINETE;

    DBMS_OUTPUT.PUT_LINE('Costul total al intretinerii este: ' || v_total_cost || ' lei.'):
END;
/
EXECUTE Calculare_Cost_Total_Intretinere;

```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x

Task completed in 0,471 seconds

```

Procedure CALCULARE_COST_TOTAL_INTRETINERE compiled

Costul total al intretinerii este: 670 lei.

PL/SQL procedure successfully completed.

```

**2. Găsire trotinetă cu baterie minimă:** Găsește și afișează trotineta cu cel mai mic procent de baterie rămasă, utilizând tabelul LICAAMALIA\_TROTINETE.

```
CREATE OR REPLACE PROCEDURE Gaseste_Trotineta_Minima_Baterie IS
```

```

    v_id_trotineta NUMBER;

    v_baterie_min NUMBER;

BEGIN

    SELECT ID_TROTINETA,
    BATERIE_RAMASA

    INTO v_id_trotineta, v_baterie_min

    FROM LICAAMALIA_TROTINETE

    WHERE BATERIE_RAMASA = (SELECT MIN(BATERIE_RAMASA) FROM LICAAMALIA_TROTINETE);

```

```

    DBMS_OUTPUT.PUT_LINE('Trotineta cu
bateria minimă este ID: ' || v_id_trotineta ||

    ', Baterie rămasă: ' || v_baterie_min || '%');

```

```
END;
```

```
/
```

```
EXECUTE Gaseste_Trotineta_Minima_Baterie;
```

```

--2. Găsire trotinetă cu baterie minimă
CREATE OR REPLACE PROCEDURE Gaseste_Trotineta_Minima_Baterie IS
    v_id_trotineta NUMBER;
    v_baterie_min NUMBER;
BEGIN
    SELECT ID_TROTINETA, BATERIE_RAMASA
    INTO v_id_trotineta, v_baterie_min
    FROM LICAAMALIA_TROTINETE
    WHERE BATERIE_RAMASA = (SELECT MIN(BATERIE_RAMASA) FROM LICAAMALIA_TROTINETE);

    DBMS_OUTPUT.PUT_LINE('Trotineta cu bateria minimă este ID: ' || v_id_trotineta ||
    ', Baterie rămasă: ' || v_baterie_min || '%');
END;
/
EXECUTE Gaseste_Trotineta_Minima_Baterie;

```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x

Task completed in 0,438 seconds

```

Procedure GASESTE_TROTINETA_MINIMA_BATERIE compiled

Trotineta cu bateria minimă este ID: 3, Baterie rămasă: 60%

PL/SQL procedure successfully completed.

```



**3. Afișare utilizatori și vârste:** Parcurge toți utilizatorii și afișează numele complet și vârsta fiecăruia.

CREATE OR REPLACE PROCEDURE Afișare\_Utilizatori IS

```

    CURSOR c_utilizatori IS
        SELECT NUME || ' ' || PRENUME AS
        NUME_COMPLET, VARSTA
        FROM LICAAMALIA_UTILIZATORI;

    v_utilizator c_utilizatori%ROWTYPE;

BEGIN
    OPEN c_utilizatori;

    LOOP

        FETCH c_utilizatori INTO v_utilizator;

        EXIT WHEN c_utilizatori%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Utilizator: ' ||
        v_utilizator.NUME_COMPLET ||
            ', Vârsta: ' || v_utilizator.VARSTA);

    END LOOP;

    CLOSE c_utilizatori;

END;

/

EXECUTE Afișare_Utilizatori;

```

```

--3. Afișare utilizatori și vârste
CREATE OR REPLACE PROCEDURE Afișare_Utilizatori IS
    CURSOR c_utilizatori IS
        SELECT NUME || ' ' || PRENUME AS NUME_COMPLET, VARSTA
        FROM LICAAMALIA_UTILIZATORI;
    v_utilizator c_utilizatori%ROWTYPE;
BEGIN
    OPEN c_utilizatori;
    LOOP
        FETCH c_utilizatori INTO v_utilizator;
        EXIT WHEN c_utilizatori%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Utilizator: ' || v_utilizator.NUME_COMPLET ||
            ', Vârsta: ' || v_utilizator.VARSTA);

    END LOOP;
    CLOSE c_utilizatori;
END;
/
EXECUTE Afișare_Utilizatori;

```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x

Task completed in 0,408 seconds

Procedure AFISARE\_UTILIZATORI compiled

```

Utilizator: LICA AMALIA, Vârsta: 20
Utilizator: POPESCU ION, Vârsta: 25
Utilizator: IONESCU MARIA, Vârsta: 30
Utilizator: GEORGESCU ANDREI, Vârsta: 28
Utilizator: RADULESCU ELENA, Vârsta: 22
Utilizator: POPA MIHAI, Vârsta: 26
Utilizator: STANCU GABRIELA, Vârsta: 32
Utilizator: CONSTANTIN ALEXANDRU, Vârsta: 29
Utilizator: DUMITRU CRISTINA, Vârsta: 23
Utilizator: BARBU DANIEL, Vârsta: 27
Utilizator: VASILESCU ANA, Vârsta: 21

```

PL/SQL procedure successfully completed.

**4. Calculare distanță totală parcursă de trotinete** - Această procedură calculează și afișează distanța totală parcursă de toate trotinetele pe baza rapoartelor din tabelul LICAAMALIA\_RAPORT\_TROTINETE.

```

CREATE OR REPLACE
PROCEDURE
Calculare_Distanta_Totala IS

    v_distanta_totala NUMBER := 0;

BEGIN

    SELECT
    SUM(KILOMETRI_PARCURSI)
    INTO v_distanta_totala

```

```

--4. Calculare distanță totală parcursă de trotinete
CREATE OR REPLACE PROCEDURE Calculare_Distanta_Totala IS
    v_distanta_totala NUMBER := 0;
BEGIN
    SELECT SUM(KILOMETRI_PARCURSI) INTO v_distanta_totala
    FROM LICAAMALIA_RAPORT_TROTINETE;
    DBMS_OUTPUT.PUT_LINE('Distanța totală parcursă de toate trotinetele este: ' || v_distanta_totala || ' km');
END;
/
BEGIN
    Calculare_Distanta_Totala;
END;
/

```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x

Task completed in 3,551 seconds

Procedure CALCULARE\_DISTANTA\_TOTALA compiled

Distanța totală parcursă de toate trotinetele este: 112,4 km

PL/SQL procedure successfully completed.

```

FROM LICAAMALIA_RAPORT_TROTINETE;

DBMS_OUTPUT.PUT_LINE('Distanța totală parcursă de toate trotinetele este: ' || v_distanța_totala || ' km');

END;

/

BEGIN

    Calculare_Distanța_Totală;

END;

/

```

**5. Afișare închirieri peste o anumită durată:** Afișează detalii (ID închiriere, ID utilizator, durata, costul total) pentru toate închirierile cu o durată mai mare decât o valoare specificată, utilizând tabelul *LICAAMALIA\_INCHIRIERI*.

```
CREATE OR REPLACE PROCEDURE Afișare_Inchirieri (
```

```

    p_durata_minima IN NUMBER
) IS

    CURSOR c_inchirieri IS

        SELECT ID_INCHIRIERE, ID_UTILIZATOR,
               DURATA_INCHIRIERE, COST_TOTAL

        FROM LICAAMALIA_INCHIRIERI

        WHERE DURATA_INCHIRIERE > p_durata_minima;

    v_inchiriere c_inchirieri%ROWTYPE;

```

```
BEGIN
```

```
    OPEN c_inchirieri;
```

```
    LOOP
```

```
        FETCH c_inchirieri INTO v_inchiriere;
```

```
        EXIT WHEN c_inchirieri%NOTFOUND;
```

```
        DBMS_OUTPUT.PUT_LINE('ID Închiriere: ' ||
                               v_inchiriere.ID_INCHIRIERE ||
```

```
                               ', ID Utilizator: ' || v_inchiriere.ID_UTILIZATOR ||
```

```
                               ', Durată: ' || v_inchiriere.DURATA_INCHIRIERE || ' ore' ||
```

```
                               ', Cost: ' || v_inchiriere.COST_TOTAL || ' lei');
```

```
    END LOOP;
```

```
    CLOSE c_inchirieri;
```

```
END;
```

```
/
```

```
EXEC Afișare_Inchirieri(2);
```

```

--5. Afișare închirieri peste o anumită durată
CREATE OR REPLACE PROCEDURE Afișare_Inchirieri (
    p_durata_minima IN NUMBER
) IS
    CURSOR c_inchirieri IS
        SELECT ID_INCHIRIERE, ID_UTILIZATOR, DURATA_INCHIRIERE, COST_TOTAL
        FROM LICAAMALIA_INCHIRIERI
        WHERE DURATA_INCHIRIERE > p_durata_minima;
    v_inchiriere c_inchirieri%ROWTYPE;
BEGIN
    OPEN c_inchirieri;
    LOOP
        FETCH c_inchirieri INTO v_inchiriere;
        EXIT WHEN c_inchirieri%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('ID Închiriere: ' || v_inchiriere.ID_INCHIRIERE ||
                               ', ID Utilizator: ' || v_inchiriere.ID_UTILIZATOR ||
                               ', Durată: ' || v_inchiriere.DURATA_INCHIRIERE || ' ore' ||
                               ', Cost: ' || v_inchiriere.COST_TOTAL || ' lei');

    END LOOP;
    CLOSE c_inchirieri;
END;
/
EXEC Afișare_Inchirieri(2);

```

Procedure AFISARE\_INCHIRIERI compiled

```

ID Închiriere: 3, ID Utilizator: 4, Durată: 3 ore, Cost: 18,9 lei
ID Închiriere: 4, ID Utilizator: 9, Durată: 4 ore, Cost: 26,4 lei
ID Închiriere: 7, ID Utilizator: 10, Durată: 3 ore, Cost: 21,3 lei
ID Închiriere: 9, ID Utilizator: 8, Durată: 4 ore, Cost: 28,8 lei

```

PL/SQL procedure successfully completed.

## II. FUNCȚII

**1. Calculare cost total întreținere pentru o trotinetă:** Returnează costul total de întreținere pentru o trotinetă, identificată prin ID.

```
CREATE OR REPLACE FUNCTION Cost_Intretinere_Trotineta (
```

```
    p_id_trotineta IN NUMBER
```

```
) RETURN NUMBER IS
```

```
    v_cost_total NUMBER;
```

```
BEGIN
```

```
    SELECT NVL(SUM(COST_INTRETINERE), 0)
```

```
    INTO v_cost_total
```

```
    FROM
```

```
LICAAMALIA_INTRETINERE_TROTINETE
```

```
    WHERE ID_TROTINETA = p_id_trotineta;
```

```
    RETURN v_cost_total;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
```

```
        RETURN 0;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_cost_total NUMBER;
```

```
BEGIN
```

```
    v_cost_total := Cost_Intretinere_Trotineta(1);
```

```
    DBMS_OUTPUT.PUT_LINE('Cost total întreținere pentru trotineta 1: ' || v_cost_total);
```

```
END;
```

```
/
```

```
--1.Calculare cost total întreținere pentru o trotinetă
CREATE OR REPLACE FUNCTION Cost_Intretinere_Trotineta (
    p_id_trotineta IN NUMBER
) RETURN NUMBER IS
    v_cost_total NUMBER;
BEGIN
    SELECT NVL(SUM(COST_INTRETINERE), 0)
    INTO v_cost_total
    FROM LICAAMALIA_INTRETINERE_TROTINETE
    WHERE ID_TROTINETA = p_id_trotineta;

    RETURN v_cost_total;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 0; -- Dacă trotineta nu are înregistrări de întreținere
END;
/
SET SERVEROUTPUT ON;

DECLARE
    v_cost_total NUMBER;
BEGIN
    v_cost_total := Cost_Intretinere_Trotineta(1);
    DBMS_OUTPUT.PUT_LINE('Cost total întreținere pentru trotineta 1: ' || v_cost_total);
END;
/
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x

Task completed in 2,943 seconds

PL/SQL procedure successfully completed.

Function COST\_INTRETINERE\_TROTINETA compiled

Cost total întreținere pentru trotineta 1: 25

PL/SQL procedure successfully completed.

**2. Calculare distanța totală parcursă de o trotinetă:** Returnează distanța totală parcursă de o trotinetă, identificată prin ID.

```
CREATE OR REPLACE FUNCTION Distanța_Totală_Trotineta (
```

```
    p_id_trotineta IN NUMBER
```

```
) RETURN NUMBER IS
```

```
    v_distanța_totala NUMBER;
```

```
BEGIN
```

```
    SELECT NVL(SUM(KILOMETRI_PARCURSI), 0)
```

```
    INTO v_distanța_totala
```

```
    FROM LICAAMALIA_RAPORT_TROTINETE
```

```
    WHERE ID_TROTINETA = p_id_trotineta;
```

```
    RETURN v_distanța_totala;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
```

```
        RETURN 0; -- Dacă trotineta nu are  
        înregistrări în raport
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_distanța_totala NUMBER;
```

```
BEGIN
```

```
    v_distanța_totala := Distanța_Totală_Trotineta(3);
```

```
    DBMS_OUTPUT.PUT_LINE('Distanța totală parcursă de trotineta 3: ' || v_distanța_totala || ' km');
```

```
END;
```

```
/
```

```
--2. Calculare distanța totală parcursă de o trotineta
CREATE OR REPLACE FUNCTION Distanța_Totală_Trotineta (
    p_id_trotineta IN NUMBER
) RETURN NUMBER IS
    v_distanța_totala NUMBER;
BEGIN
    SELECT NVL(SUM(KILOMETRI_PARCURSI), 0)
    INTO v_distanța_totala
    FROM LICAAMALIA_RAPORT_TROTINETE
    WHERE ID_TROTINETA = p_id_trotineta;

    RETURN v_distanța_totala;
EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 0; -- Dacă trotineta nu are înregistrări în raport
END;
/

SET SERVEROUTPUT ON;

DECLARE
    v_distanța_totala NUMBER;
BEGIN
    v_distanța_totala := Distanța_Totală_Trotineta(3);
    DBMS_OUTPUT.PUT_LINE('Distanța totală parcursă de trotineta 3: ' || v_distanța_totala || ' km');
END;
/
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x

Task completed in 3,243 seconds

PL/SQL procedure successfully completed.

Function DISTANȚA\_TOTALĂ\_TROTINETA compiled

Distanța totală parcursă de trotineta 3: 10,3 km

PL/SQL procedure successfully completed.

**3. Calculare cost total închirieri pentru un utilizator:** Returnează costul total al închirierilor realizate de un utilizator, identificat prin ID.

```
CREATE OR REPLACE FUNCTION Cost_Total_Inchirieri (
```

```
    p_id_utilizator IN NUMBER
```

```
) RETURN NUMBER IS
```

```
    v_cost_total NUMBER;
```

```
BEGIN
```

```
    SELECT
    NVL(SUM(COST_TOTAL), 0)
```

```
    INTO v_cost_total
```

```
    FROM
    LICAAMALIA_INCHIRIERI
```

```
    WHERE ID_UTILIZATOR =
    p_id_utilizator;
```

```
    RETURN v_cost_total;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND
    THEN
```

```
        RETURN 0;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_cost_total NUMBER;
```

```
BEGIN
```

```
    v_cost_total := Cost_Total_Inchirieri(7);
```

```
    DBMS_OUTPUT.PUT_LINE('Cost total al închirierilor pentru utilizatorul cu ID 7: ' || v_cost_total || ' lei');
END;
```

```
/
```

```
--3. Calculare cost total închirieri pentru un utilizator
CREATE OR REPLACE FUNCTION Cost_Total_Inchirieri (
    p_id_utilizator IN NUMBER
) RETURN NUMBER IS
    v_cost_total NUMBER;
BEGIN
    -- Calcularea costului total al închirierilor pentru utilizatorul dat
    SELECT NVL(SUM(COST_TOTAL), 0)
    INTO v_cost_total
    FROM LICAAMALIA_INCHIRIERI
    WHERE ID_UTILIZATOR = p_id_utilizator;

    RETURN v_cost_total;

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RETURN 0; -- Dacă utilizatorul nu are închirieri, returnăm 0
END;
/
SET SERVEROUTPUT ON;

DECLARE
    v_cost_total NUMBER;
BEGIN
    v_cost_total := Cost_Total_Inchirieri(7);
    DBMS_OUTPUT.PUT_LINE('Cost total al închirierilor pentru utilizatorul cu ID 7: ' || v_cost_total || ' lei');
END;
/

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x
Task completed in 0,418 seconds
PL/SQL procedure successfully completed.

Function COST_TOTAL_INCHIRIERI compiled
Cost total al închirierilor pentru utilizatorul cu ID 7: 7,2 lei

PL/SQL procedure successfully completed.
```

**4. Calculare vârstă utilizator:** Returnează vârsta unui utilizator, identificat prin ID. Dacă utilizatorul nu există, returnează NULL.

```
CREATE OR REPLACE FUNCTION Varsta_Utilizator (
```

```
    p_id_utilizator IN NUMBER
```

```
) RETURN NUMBER IS
```

```
    v_varsta NUMBER;
```

```
BEGIN
```

```
    SELECT VARSTA
```

```
    INTO v_varsta
```

```
    FROM LICAAMALIA_UTILIZATORI
```

```
    WHERE ID_UTILIZATOR =  
p_id_utilizator;
```

```
    RETURN v_varsta;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
```

```
        RETURN NULL;
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_varsta NUMBER;
```

```
BEGIN
```

```
    v_varsta := Varsta_Utilizator(5);
```

```
    IF v_varsta IS NOT NULL THEN
```

```
        DBMS_OUTPUT.PUT_LINE('Vârsta utilizatorului cu ID 5: ' || v_varsta || ' ani');
```

```
    ELSE
```

```
        DBMS_OUTPUT.PUT_LINE('Nu există utilizator cu ID-ul 5.');
```

```
    END IF;
```

```
END;
```

```
/
```

```
--4. Calculare vârstă utilizator
```

```
CREATE OR REPLACE FUNCTION Varsta_Utilizator (
```

```
    p_id_utilizator IN NUMBER
```

```
) RETURN NUMBER IS
```

```
    v_varsta NUMBER;
```

```
BEGIN
```

```
    SELECT VARSTA
```

```
    INTO v_varsta
```

```
    FROM LICAAMALIA_UTILIZATORI
```

```
    WHERE ID_UTILIZATOR = p_id_utilizator;
```

```
    RETURN v_varsta;
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
```

```
        RETURN NULL; -- Dacă utilizatorul nu există
```

```
END;
```

```
/
```

```
SET SERVEROUTPUT ON;
```

```
DECLARE
```

```
    v_varsta NUMBER;
```

```
BEGIN
```

```
    v_varsta := Varsta_Utilizator(5);
```

```
IF v_varsta IS NOT NULL THEN
```

```
    DBMS_OUTPUT.PUT_LINE('Vârsta utilizatorului cu ID 5: ' || v_varsta || ' ani');
```

```
ELSE
```

```
    DBMS_OUTPUT.PUT_LINE('Nu există utilizator cu ID-ul 5.');
```

```
END IF;
```

```
END;
```

```
/
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Q

Task completed in 0,679 seconds

PL/SQL procedure successfully completed.

Function VARSTA\_UTILIZATOR compiled

Vârsta utilizatorului cu ID 5: 22 ani

PL/SQL procedure successfully completed.

## H. Concluzii și Perspective de Dezvoltare

Proiectul realizat, având ca obiectiv dezvoltarea unei baze de date pentru gestionarea rețelelor de trotinete electrice, reprezintă un exemplu de integrare a tehnologiilor bazelor de date pentru optimizarea proceselor operaționale. Structura relațională bine fundamentată permite o administrare eficientă a datelor esențiale, precum informațiile despre trotinete, utilizatori, întreținere și istoricul utilizării acestora. Implementarea procedurilor și funcțiilor PL/SQL a contribuit la automatizarea unor operațiuni complexe, simplificând astfel gestionarea resurselor și oferind un grad crescut de control.

Prin intermediul soluțiilor dezvoltate, a fost posibilă monitorizarea precisă a trotinetelor, atât din punct de vedere al locației, cât și al stării tehnice. Totodată, sistemul permite calcularea automată a costurilor de întreținere și a distanțelor parcurse, oferind informații critice pentru luarea deciziilor manageriale. Aceste funcționalități nu doar că eficientizează activitatea operațională, dar contribuie și la o mai bună satisfacție a utilizatorilor prin asigurarea unei disponibilități ridicate a trotinetelor.

Procesul de dezvoltare a evidențiat și câteva provocări. Printre acestea se numără gestionarea eficientă a relațiilor dintre tabele în contextul unui volum mare de date, precum și asigurarea unei performanțe optime pentru procedurile și funcțiile complexe. Abordarea acestor provocări a necesitat o înțelegere profundă a principiilor de proiectare a bazelor de date și utilizarea unor soluții scalabile, adaptate cerințelor proiectului.

În ceea ce privește perspectivele de dezvoltare, sistemul propus poate fi extins în mai multe direcții. Integrarea unui modul avansat de raportare ar permite generarea de statistici detaliate cu privire la utilizarea trotinetelor, costurile de întreținere și alte aspecte esențiale pentru management. De asemenea, un sistem de recomandări, bazat pe istoricul de utilizare al trotinetelor, ar contribui la o experiență mai personalizată pentru utilizatori. Automatizarea proceselor de întreținere prin notificări proactive ar aduce beneficii suplimentare, asigurând reducerea timpului de inactivitate a trotinetelor. În plus, o aplicație mobilă dedicată utilizatorilor ar putea facilita accesul la servicii, incluzând funcționalități precum rezervarea trotinetelor, urmărirea locației acestora în timp real și gestionarea plăților direct din aplicație.

În concluzie, proiectul propus demonstrează rolul esențial al bazelor de date relaționale în gestionarea eficientă a unui sistem complex de mobilitate urbană. Soluția oferită nu doar că răspunde cerințelor actuale, dar asigură și o infrastructură solidă pentru extinderea ulterioară a funcționalităților. Prin integrarea unor tehnologii suplimentare și adaptarea continuă la nevoile utilizatorilor, sistemul poate deveni un model de succes în domeniul mobilității sustenabile.

# I. Cod aferent din SQL Developer

```
-- Crearea tabelci "TROTINETE"
CREATE TABLE LICAAMALIA_TROTINETE (
    ID_TROTINETA NUMBER(6) PRIMARY KEY,
    NUME_MODEL VARCHAR2(30),
    PRODUCATOR VARCHAR2(25),
    STARE VARCHAR2(20),
    BATERIE_RAMASA NUMBER(8),
    PRET_MINUT NUMBER(8,2),
    LOCATIE_CURENTA VARCHAR2(40)
);
describe LICAAMALIA_TROTINETE

-- Crearea tabelci "UTILIZATORI"
CREATE TABLE LICAAMALIA_UTILIZATORI (
    ID_UTILIZATOR NUMBER PRIMARY KEY,
    NUME VARCHAR2(30),
    PRENUME VARCHAR2(30),
    ADRESA VARCHAR2(50),
    EMAIL VARCHAR2(25),
    TELEFON VARCHAR2(15)
);
describe LICAAMALIA_UTILIZATORI

-- Crearea tabelci "INTRETINERE_TROTINETE"
CREATE TABLE LICAAMALIA_INTRETINERE_TROTINETE (
    ID_INTRETINERE NUMBER(6) PRIMARY KEY,
    ID_TROTINETA NUMBER(6),
    DATA_INTRETINERE DATE,
    TIP_INTRETINERE VARCHAR2(30),
    COST_INTRETINERE NUMBER(8,2),
    PIESE_INLOCUITE VARCHAR2(40),
    TEHNICIAN_RESPONSABIL VARCHAR2(40),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_INTRETINERE_TROTINETE

-- Crearea tabelci "RAPORT_TROTINETE"
CREATE TABLE LICAAMALIA_RAPORT_TROTINETE (
    ID_RAPORT NUMBER(6) PRIMARY KEY,
    ID_TROTINETA NUMBER(6),
    DATA_RAPORT DATE,
    KILOMETRI_PARCURSI NUMBER(8,2),
    STARE_BATERIE NUMBER(8,2),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_RAPORT_TROTINETE

-- Crearea tabelci "INCHIRIERI"
CREATE TABLE LICAAMALIA_INCHIRIERI (
    ID_INCHIRIERE NUMBER(6) PRIMARY KEY,
    ID_UTILIZATOR NUMBER(6),
    ID_TROTINETA NUMBER(6),
    DATA_INCHIRIERE TIMESTAMP(6) WITH LOCAL TIME ZONE,
    DURATA_INCHIRIERE NUMBER(6),
    COST_TOTAL NUMBER(8,2),
    FOREIGN KEY (ID_UTILIZATOR) REFERENCES LICAAMALIA_UTILIZATORI(ID_UTILIZATOR),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_INCHIRIERI

-- Crearea tabelci "ISTORIC_UTILIZARE"
CREATE TABLE LICAAMALIA_ISTORIC_UTILIZARE (
    ID_ISTORIC NUMBER(6) PRIMARY KEY,
    ID_UTILIZATOR NUMBER(6),
    ID_TROTINETA NUMBER(6),
    DATA_UTILIZARE DATE,
    DURATA_UTILIZARE NUMBER(6),
    DISTANTA_PARCURSA NUMBER(8,2),
    COST_UTILIZARE NUMBER(8,2),
    FOREIGN KEY (ID_UTILIZATOR) REFERENCES LICAAMALIA_UTILIZATORI(ID_UTILIZATOR),
    FOREIGN KEY (ID_TROTINETA) REFERENCES LICAAMALIA_TROTINETE(ID_TROTINETA)
);
describe LICAAMALIA_ISTORIC_UTILIZARE

-- +++
ALTER TABLE LICAAMALIA_UTILIZATORI
ADD VARSTA NUMBER(3);
describe LICAAMALIA_UTILIZATORI

--1. Adăugarea de noi înregistrări în tabela "TROTINETE"
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(1, 'E10', 'XIAOMI', 'BUNA', '70', 1.5, 'Pata Unirii');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(2, 'E12', 'XIAOMI', 'BUNA', '75', 1.7, 'Parcul Carol');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(3, 'ES4', 'SEGWAY', 'BUNA', '80', 1.6, 'Parcul Herăstrău');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(4, 'M365', 'XIAOMI', 'OK', '80', 1.4, 'Parcul Tinerețului');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(5, 'BOOSTED', 'SEGWAY', 'BUNA', '85', 1.9, 'Bulevardul Maghera');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(6, 'CITYBUG 2S', 'CITYBUG', 'OK', '65', 1.8, 'Parcul Izvor');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(7, 'Ninebot MAX', 'SEGWAY', 'BUNA', '85', 1.9, 'Bulevardul Maghera');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(8, 'Zero 9', 'ZERO', 'BUNA', '70', 1.5, 'Parcul Circuliui');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(9, 'Hibrytum P5', 'HOLLYBURN', 'OK', '75', 1.6, 'Parcul Titia');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(10, 'Air T15', 'AIRWHEEL', 'BUNA', '80', 1.7, 'Bulevardul Unirii');
INSERT INTO LICAAMALIA_TROTINETE (ID_TROTINETA, NUME_MODEL, PRODUCATOR, STARE, BATERIE_RAMASA, PRET_MINUT, LOCATIE_CURENTA) VALUES
(11, 'GX1 V2', 'GOTRAX', 'BUNA', '78', 1.6, 'Pata Victoriei);

SELECT* FROM LICAAMALIA_TROTINETE
ORDER BY ID_TROTINETA ASC;

--2. Adăugarea de noi înregistrări în tabela "UTILIZATORI"
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(1, 'LICA', 'AMALIA', 'Str. Omlui 3', 'amaliaica@gmail.com', '0771665319', 20);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(2, 'POPESCU', 'ION', 'Str. Muresului 7', 'ion.popescu@gmail.com', '0770123456', 25);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(3, 'IONESCU', 'MARIA', 'Str. Dorobantilor 15', 'maria.ionescu@gmail.com', '0770789123', 30);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(4, 'GEORGESCU', 'ANDREI', 'Str. Buceuresti 21', 'andrei.georgescu@gmail.com', '0770555678', 28);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(5, 'RADULESCU', 'ELENA', 'Str. Unirii 8', 'elena.radulescu@gmail.com', '0770998765', 22);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(6, 'POPA', 'MIHAEL', 'Str. Timisului 12', 'mihael.popa@gmail.com', '077088834', 26);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(7, 'STANCU', 'GABRIELA', 'Str. Clujului 5', 'gabriela.stancu@gmail.com', '0770333678', 32);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(8, 'CONSTANTIN', 'ALEXANDRU', 'Str. Iasi 19', 'alex.constantin@gmail.com', '077011222', 29);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(9, 'DUMITRU', 'CRISTINA', 'Str. Vaslui 4', 'cris.dumitru@gmail.com', '0770456123', 23);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(10, 'BARBU', 'DANIEL', 'Str. Bihorul 11', 'daniel.barbu@gmail.com', '077045876', 27);
INSERT INTO LICAAMALIA_UTILIZATORI (ID_UTILIZATOR, NUME, PRENUME, ADRESA, EMAIL, TELEFON, VARSTA) VALUES
(11, 'VASILESCU', 'ANA', 'Str. Sibiu 2', 'ana.vasilescu@gmail.com', '0770998765', 21);

SELECT* FROM LICAAMALIA_UTILIZATORI
ORDER BY ID_UTILIZATOR ASC;

--3. Adăugarea de noi înregistrări în tabela "INTRETINERE_TROTINETE"
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE (ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(1, 3, TO_DATE('2023-01-01', 'YYYY-MM-DD'), 150.00, 'Baterie, Frână', 'Ion Popescu');
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE (ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(2, 7, TO_DATE('2023-02-15', 'YYYY-MM-DD'), 80.00, 'Accumulator', 'Maria Ionescu');
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE (ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(3, 4, TO_DATE('2023-03-20', 'YYYY-MM-DD'), 50.00, 'Cablu electric', 'Andrei Georgescu');
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE (ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(4, 9, TO_DATE('2023-04-10', 'YYYY-MM-DD'), 60.00, 'Roți', 'Elena Radulescu');
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE (ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(5, 5, TO_DATE('2023-05-05', 'YYYY-MM-DD'), 45.00, 'Plicuțe de frână', 'Mihai Popa');
INSERT INTO LICAAMALIA_INTRETINERE_TROTINETE (ID_INTRETINERE, ID_TROTINETA, DATA_INTRETINERE, COST_INTRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
```



```
(6, 2, TO_DATE('2023-06-18', 'YYYY-MM-DD'), 120.00, 'Baterie, Sistem de direcție', 'Gabriela Stancu');
INSERT INTO LICAAMALIA_INTRRETINERE_TROTINETE(ID_INTRRETINERE, ID_TROTINETA, DATA_INTRRETINERE, COST_INTRRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(7, 10, TO_DATE('2023-07-22', 'YYYY-MM-DD'), 30.00, 'Becuri', 'Alexandru Constantin');
INSERT INTO LICAAMALIA_INTRRETINERE_TROTINETE(ID_INTRRETINERE, ID_TROTINETA, DATA_INTRRETINERE, COST_INTRRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(8, 6, TO_DATE('2023-08-12', 'YYYY-MM-DD'), 70.00, 'Plăcuțe frână', 'Cristina Dumitru');
INSERT INTO LICAAMALIA_INTRRETINERE_TROTINETE(ID_INTRRETINERE, ID_TROTINETA, DATA_INTRRETINERE, COST_INTRRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(9, 8, TO_DATE('2023-09-15', 'YYYY-MM-DD'), 40.00, 'Amortizor', 'Daniel Barbu');
INSERT INTO LICAAMALIA_INTRRETINERE_TROTINETE(ID_INTRRETINERE, ID_TROTINETA, DATA_INTRRETINERE, COST_INTRRETINERE, PIESE_INLOCUITE, TEHNICIAN_RESPONSABIL) VALUES
(10, 1, TO_DATE('2023-10-15', 'YYYY-MM-DD'), 25.00, 'Lampă', 'Ana Vasilescu');
```

```
SELECT* FROM LICAAMALIA_INTRRETINERE_TROTINETE
ORDER BY ID_INTRRETINERE ASC;
```

```
--4. Adăugarea de noi înregistrări în tabela "RAPORT_TROTINETE"
```

```
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(1, 3, TO_DATE('2023-01-05', 'YYYY-MM-DD'), 103.3, 80);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(2, 7, TO_DATE('2023-02-10', 'YYYY-MM-DD'), 15.8, 75.5);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(3, 4, TO_DATE('2023-03-15', 'YYYY-MM-DD'), 201, 85.2);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(4, 9, TO_DATE('2023-04-20', 'YYYY-MM-DD'), 11, 70.0);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(5, 5, TO_DATE('2023-05-25', 'YYYY-MM-DD'), 5, 78.5);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(6, 2, TO_DATE('2023-06-30', 'YYYY-MM-DD'), 4.9, 90.0);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(7, 10, TO_DATE('2023-07-05', 'YYYY-MM-DD'), 17.8, 82.3);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(8, 6, TO_DATE('2023-08-10', 'YYYY-MM-DD'), 17.3, 76.8);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(9, 8, TO_DATE('2023-09-15', 'YYYY-MM-DD'), 2, 82.8);
INSERT INTO LICAAMALIA_RAPORT_TROTINETE(ID_RAPORT, ID_TROTINETA, DATA_RAPORT, KILOMETRI_PARCURSI, STARE_BATERIE) VALUES
(10, 1, TO_DATE('2023-10-20', 'YYYY-MM-DD'), 18, 79.5);
```

```
SELECT* FROM LICAAMALIA_RAPORT_TROTINETE
ORDER BY ID_RAPORT ASC;
```

```
--5. Adăugarea de noi înregistrări în tabela "INCHIRIERI"
```

```
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(1, 3, 2, TO_TIMESTAMP('2023-01-05 10:30:00', 'YYYY-MM-DD HH24:MI:SS'), 2, 10.50);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(2, 7, 6, TO_TIMESTAMP('2023-02-10 15:45:00', 'YYYY-MM-DD HH24:MI:SS'), 1, 7.20);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(3, 4, 8, TO_TIMESTAMP('2023-03-15 17:30:00', 'YYYY-MM-DD HH24:MI:SS'), 3, 18.90);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(4, 9, 1, TO_TIMESTAMP('2023-04-20 09:15:00', 'YYYY-MM-DD HH24:MI:SS'), 4, 26.40);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(5, 5, 5, TO_TIMESTAMP('2023-05-25 17:30:00', 'YYYY-MM-DD HH24:MI:SS'), 1, 8.50);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(6, 2, 9, TO_TIMESTAMP('2023-06-30 14:00:00', 'YYYY-MM-DD HH24:MI:SS'), 2, 15.60);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(7, 10, 3, TO_TIMESTAMP('2023-07-05 11:45:00', 'YYYY-MM-DD HH24:MI:SS'), 3, 21.30);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(8, 6, 7, TO_TIMESTAMP('2023-08-10 18:20:00', 'YYYY-MM-DD HH24:MI:SS'), 1, 11.90);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(9, 8, 4, TO_TIMESTAMP('2023-09-15 09:00:00', 'YYYY-MM-DD HH24:MI:SS'), 4, 28.80);
INSERT INTO LICAAMALIA_INCHIRIERI(ID_INCHIRIERE, ID_UTILIZATOR, ID_TROTINETA, DATA_INCHIRIERE, DURATA_INCHIRIERE, COST_TOTAL ) VALUES
(10, 1, 10, TO_TIMESTAMP('2023-10-20 13:15:00', 'YYYY-MM-DD HH24:MI:SS'), 2, 14.70);
```

```
SELECT* FROM LICAAMALIA_INCHIRIERI
ORDER BY ID_INCHIRIERE ASC;
```

```
--6. Adăugarea de noi înregistrări în tabela "ISTORIC_UTILIZARE"
```

```
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(1, 3, 2, TO_DATE('2023-01-05', 'YYYY-MM-DD'), 2, 5.8, 20);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(2, 7, 6, TO_DATE('2023-02-10', 'YYYY-MM-DD'), 1, 3.5, 4.50);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(3, 4, 8, TO_DATE('2023-03-15', 'YYYY-MM-DD'), 3, 10.2, 14.70);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(4, 9, 1, TO_DATE('2023-04-20', 'YYYY-MM-DD'), 4, 15.0, 20.40);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(5, 5, 5, TO_DATE('2023-05-25', 'YYYY-MM-DD'), 1, 4.1, 6.80);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(6, 2, 9, TO_DATE('2023-06-30', 'YYYY-MM-DD'), 2, 7.3, 10.50);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(7, 10, 3, TO_DATE('2023-07-05', 'YYYY-MM-DD'), 3, 9.8, 13.20);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(8, 6, 7, TO_DATE('2023-08-10', 'YYYY-MM-DD'), 1, 5.5, 7.90);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(9, 8, 4, TO_DATE('2023-09-15', 'YYYY-MM-DD'), 4, 14.7, 19.80);
INSERT INTO LICAAMALIA_ISTORIC_UTILIZARE(ID_ISTORIC, ID_UTILIZATOR, ID_TROTINETA, DATA_UTILIZARE, DURATA_UTILIZARE, DISTANTA_PARCURSA, COST_UTILIZARE) VALUES
(10, 1, 10, TO_DATE('2023-10-20', 'YYYY-MM-DD'), 2, 6.2, 9.30);
```

```
SELECT* FROM LICAAMALIA_ISTORIC_UTILIZARE
ORDER BY ID_ISTORIC ASC;
```

```
--E. Blocuri PL/SQL conținând structuri de control variate
```

```
--1. Utilizare FOR LOOP
```

```
SET SERVEROUTPUT ON;

BEGIN
  FOR v_trotineta IN
    (SELECT ID_TROTINETA, NUME_MODEL, PRODUCATOR FROM LICAAMALIA_TROTINETE)
  LOOP
    DBMS_OUTPUT.PUT_LINE('Trotineta ID: ' || v_trotineta.ID_TROTINETA ||
      ', Model: ' || v_trotineta.NUME_MODEL ||
      ', Producator: ' || v_trotineta.PRODUCATOR);
  END LOOP;
END;
/
```

```
--2. Utilizare WHILE LOOP
```

```
SET SERVEROUTPUT ON;

DECLARE
  v_total_distanța NUMBER := 0;
  v_index NUMBER := 1;
  v_max_id NUMBER;
  v_distanța NUMBER;
BEGIN
  SELECT MAX(ID_ISTORIC) INTO v_max_id FROM LICAAMALIA_ISTORIC_UTILIZARE;
  WHILE v_index <= v_max_id LOOP
    SELECT NVL(DISTANTA_PARCURSA, 0) INTO v_distanța
    FROM LICAAMALIA_ISTORIC_UTILIZARE
    WHERE ID_ISTORIC = v_index;

    v_total_distanța := v_total_distanța + v_distanța;
    v_index := v_index + 1;
  END LOOP;

  DBMS_OUTPUT.PUT_LINE('Total distanță parcursă: ' || v_total_distanța || ' km');
END;
/
```

```
--3. Utilizare cursor
```

```
SET SERVEROUTPUT ON;

DECLARE
  CURSOR c IS
    SELECT ID_TROTINETA, NUME_MODEL, BATERIE_RAMASA
    FROM LICAAMALIA_TROTINETE
    ORDER BY BATERIE_RAMASA DESC;
  v_count NUMBER := 0;
  v_trotineta c%ROWTYPE;
BEGIN
  OPEN c;
  LOOP
    FETCH c INTO v_trotineta;
    EXIT WHEN c%NOTFOUND OR v_count = 5;

    DBMS_OUTPUT.PUT_LINE('Trotineta ID: ' || v_trotineta.ID_TROTINETA ||
      ', Model: ' || v_trotineta.NUME_MODEL ||
      ', Baterie rămasă: ' || v_trotineta.BATERIE_RAMASA || '%');

    v_count := v_count + 1;
  END LOOP;
CLOSE c;
```

```

END;
/

--4. Utilizare IF/ELSE
SET SERVEROUTPUT ON;

DECLARE
    v_name_complet VARCHAR2(100);
    v_varsta NUMBER;
    v_durata_inchiriere NUMBER;
BEGIN
    SELECT u.NUME || ' ' || u.PRENUME, u.VARSTA, i.DURATA_INCHIRIERE
    INTO v_name_complet, v_varsta, v_durata_inchiriere
    FROM LICAAMALIA_UTILIZATORI u
    JOIN LICAAMALIA_INCHIRIERI i ON u.ID_UTILIZATOR = i.ID_UTILIZATOR
    WHERE i.DURATA_INCHIRIERE > 3 AND ROWNUM = 1; -- Luăm doar primul utilizator
    DBMS_OUTPUT.PUT_LINE('Utilizator: ' || v_name_complet ||
        ', Durată închiriere: ' || v_durata_inchiriere || ' ore');

    IF v_varsta < 25 THEN
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Tânăr');
    ELSEIF v_varsta BETWEEN 25 AND 35 THEN
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Adult');
    ELSE
        DBMS_OUTPUT.PUT_LINE(' - Categoria: Senior');
    END IF;
END;
/

--5. Utilizare FOR LOOP
SET SERVEROUTPUT ON;
DECLARE
    CURSOR c IS
        SELECT u.ID_UTILIZATOR, u.NUME || ' ' || u.PRENUME AS NUME_COMPLET, SUM(i.COST_TOTAL) AS TOTAL_COST
        FROM LICAAMALIA_UTILIZATORI u
        JOIN LICAAMALIA_INCHIRIERI i ON u.ID_UTILIZATOR = i.ID_UTILIZATOR
        GROUP BY u.ID_UTILIZATOR, u.NUME, u.PRENUME;
BEGIN
    FOR v_utilizator IN c LOOP
        DBMS_OUTPUT.PUT_LINE('Utilizator: ' || v_utilizator.NUME_COMPLET ||
            ', Cost total închirieri: ' || v_utilizator.TOTAL_COST);

        IF v_utilizator.TOTAL_COST > 50 THEN
            DBMS_OUTPUT.PUT_LINE(' - Categoria: Client VIP');
        ELSE
            DBMS_OUTPUT.PUT_LINE(' - Categoria: Client obișnuit');
        END IF;
    END LOOP;
END;
/

--Utilizarea cursorilor ?i a excepțiilor în cadrul blocurilor PL/SQL
--CURSORI
--1. Cursor explicit OPEN-LOOP-FETCH-CLOSE
SET SERVEROUTPUT ON;
DECLARE
    CURSOR c IS
        SELECT NUME || ' ' || PRENUME AS NUME_COMPLET, VARSTA, ADRESA
        FROM LICAAMALIA_UTILIZATORI;
    v c%ROWTYPE;
BEGIN
    OPEN c;
    LOOP
        FETCH c INTO v;
        EXIT WHEN c%NOTFOUND;
        DBMS_OUTPUT.PUT_LINE('Nume complet: ' || v.NUME_COMPLET || ', Vârsta: ' || v.VARSTA || ', Adresa: ' || v.ADRESA);
    END LOOP;
    CLOSE c;
END;
/

--2. Cursor explicit FOR-LOOP
SET SERVEROUTPUT ON;

DECLARE
    CURSOR c IS
        SELECT ID_TROTINETA, COUNT(*) AS NUMAR_INCHIRIERI
        FROM LICAAMALIA_INCHIRIERI
        GROUP BY ID_TROTINETA;
BEGIN
    FOR v IN c LOOP
        DBMS_OUTPUT.PUT_LINE('ID Trotinetă: ' || v.ID_TROTINETA || ' - Număr Închirieri: ' || v.NUMAR_INCHIRIERI);
    END LOOP;
END;
/

--3. Cursor explicit FOR-LOOP
SET SERVEROUTPUT ON;

DECLARE
    CURSOR c IS
        SELECT ID_TROTINETA, SUM(COST_UTILIZARE) AS COST_TOTAL
        FROM LICAAMALIA_ISTORIC_UTILIZARE
        GROUP BY ID_TROTINETA
        ORDER BY COST_TOTAL DESC
        FETCH FIRST 3 ROWS ONLY;
BEGIN
    FOR v IN c LOOP
        DBMS_OUTPUT.PUT_LINE('ID Trotinetă: ' || v.ID_TROTINETA || ' - Cost Total: ' || v.COST_TOTAL);
    END LOOP;
END;
/

--4. Cursor explicit OPEN-FETCH-CLOSE
SET SERVEROUTPUT ON;

DECLARE
    CURSOR c IS
        SELECT ID_TROTINETA
        FROM LICAAMALIA_INTRETINERE_TROTINETE
        ORDER BY COST_INTRETINERE ASC
        FETCH FIRST 1 ROW ONLY;
    v_trotineta_id LICAAMALIA_INTRETINERE_TROTINETE.ID_TROTINETA%TYPE;
BEGIN
    OPEN c;
    FETCH c INTO v_trotineta_id;
    CLOSE c;

    UPDATE LICAAMALIA_TROTINETE
    SET STARE = 'BUNĂ'
    WHERE ID_TROTINETA = v_trotineta_id;

    DBMS_OUTPUT.PUT_LINE('Trotinetă cu ID ' || v_trotineta_id || ' a fost actualizată la stare BUNĂ.');
```

```

END;
/

--EXCEPTII
--1. Exceptie implicită (NO_DATA_FOUND)

```

```

SET SERVEROUTPUT ON;

DECLARE
    v_nume_utilizator LICAAMALIA_UTILIZATORI.NUME%TYPE;
BEGIN
    SELECT NUME INTO v_nume_utilizator
    FROM LICAAMALIA_UTILIZATORI
    WHERE ADRESA = 'Str. Inexistenta';

    DBMS_OUTPUT.PUT_LINE('Utilizator găsit: ' || v_nume_utilizator);

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        DBMS_OUTPUT.PUT_LINE('Nu există niciun utilizator la această adresă.');
```

END;

--2. Excepție explicită (RAISE)

```

SET SERVEROUTPUT ON;

DECLARE
    baterie_slabă EXCEPTION;
    v_count NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_count
    FROM LICAAMALIA_TROTINETE
    WHERE BATERIE_RAMASA < 65;

    IF v_count = 0 THEN
        RAISE baterie_slabă;
    ELSE
        DBMS_OUTPUT.PUT_LINE('Există ' || v_count || ' trotinete cu baterie slabă.');
```

END IF;

```

EXCEPTION
    WHEN baterie_slabă THEN
        DBMS_OUTPUT.PUT_LINE('Nu există trotinete cu baterie mai mică de 65%.');
```

END;

/

--3. Excepție generală (WHEN OTHERS)

```

SET SERVEROUTPUT ON;

BEGIN
    UPDATE LICAAMALIA_TROTINETE
    SET PRET_MINUT = PRET_MINUT + 0.1
    WHERE PRODUCATOR = 'XIAOMI';

    DBMS_OUTPUT.PUT_LINE('Prețul pe minut a fost actualizat pentru trotinetele Xiaomi.');
```

```

EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('A apărut o eroare neașteptată: ' || SQLERRM);
```

END;

/

--4. Excepție predefinită (TOO\_MANY\_ROWS)

```

SET SERVEROUTPUT ON;

DECLARE
    v_trotineta_id LICAAMALIA_TROTINETE.ID_TROTINETA%TYPE;
BEGIN
    SELECT ID_TROTINETA INTO v_trotineta_id
    FROM LICAAMALIA_TROTINETE
    WHERE PRODUCATOR = 'XIAOMI';

    DBMS_OUTPUT.PUT_LINE('Trotineta găsită: ' || v_trotineta_id);

EXCEPTION
    WHEN TOO_MANY_ROWS THEN
        DBMS_OUTPUT.PUT_LINE('Există mai multe trotinete pentru acest producător.');
```

END;

/

--PROCEDURI

--1. Calculare cost total întreținere

```

CREATE OR REPLACE PROCEDURE Calculare_Cost_Total_Intretinere IS
    v_total_cost NUMBER := 0;
BEGIN
    SELECT SUM(COST_INTRETINERE) INTO v_total_cost
    FROM LICAAMALIA_INTRETINERE_TROTINETE;

    DBMS_OUTPUT.PUT_LINE('Costul total al întreținerii este: ' || v_total_cost || ' lei.');
```

END;

/

EXECUTE Calculare\_Cost\_Total\_Intretinere;

--2. Găsire trotinetă cu baterie minimă

```

CREATE OR REPLACE PROCEDURE Gaseste_Trotineta_Minima_Baterie IS
    v_id_trotineta NUMBER;
    v_baterie_min NUMBER;
BEGIN
    SELECT ID_TROTINETA, BATERIE_RAMASA
    INTO v_id_trotineta, v_baterie_min
    FROM LICAAMALIA_TROTINETE
    WHERE BATERIE_RAMASA = (SELECT MIN(BATERIE_RAMASA) FROM LICAAMALIA_TROTINETE);

    DBMS_OUTPUT.PUT_LINE('Trotineta cu bateria minimă este ID: ' || v_id_trotineta ||
        ', Bateria rămasă: ' || v_baterie_min || '%');
```

END;

/

EXECUTE Gaseste\_Trotineta\_Minima\_Baterie;

--3. Afisare utilizatori și vârste

```

CREATE OR REPLACE PROCEDURE Afisare_Utilizatori IS
    CURSOR c_utilizatori IS
        SELECT NUME || ' ' || PRENUME AS NUME_COMPLET, VARSTA
        FROM LICAAMALIA_UTILIZATORI;
    v_utilizator c_utilizatori%ROWTYPE;
BEGIN
    OPEN c_utilizatori;
    LOOP
        FETCH c_utilizatori INTO v_utilizator;
        EXIT WHEN c_utilizatori%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('Utilizator: ' || v_utilizator.NUME_COMPLET ||
            ', Vârsta: ' || v_utilizator.VARSTA);

    END LOOP;
    CLOSE c_utilizatori;
END;
```

/

EXECUTE Afisare\_Utilizatori;

--4. Calculare distanță totală parcursă de trotinete

```

CREATE OR REPLACE PROCEDURE Calculare_Distanța_Totală IS
    v_distanța_totală NUMBER := 0;
BEGIN
    SELECT SUM(KILOMETRI_PARCURSI) INTO v_distanța_totală
    FROM LICAAMALIA_RAPORT_TROTINETE;
    DBMS_OUTPUT.PUT_LINE('Distanța totală parcursă de toate trotinetele este: ' || v_distanța_totală || ' km');
```

END;

/

```

BEGIN
    Calculare_Distanța_Totală;
END;
```

/

--5. Afisare închirieri peste o anumită durată

```

CREATE OR REPLACE PROCEDURE Afisare_Inchirieri (
    p_durată_minimă IN NUMBER
) IS
    CURSOR c_inchirieri IS
        SELECT ID_INCHIRIERE, ID_UTILIZATOR, DURATA_INCHIRIERE, COST_TOTAL
```

```

FROM LICAAMALIA_INCHIRIERI
WHERE DURATA_INCHIRIERE > p_durata_minima;
v_inchiriere c_inchirieri%ROWTYPE;
BEGIN
OPEN c_inchirieri;
LOOP
FETCH c_inchirieri INTO v_inchiriere;
EXIT WHEN c_inchirieri%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('ID Inchiriere: ' || v_inchiriere.ID_INCHIRIERE ||
', ID Utilizator: ' || v_inchiriere.ID_UTILIZATOR ||
', Durată: ' || v_inchiriere.DURATA_INCHIRIERE || ' ore' ||
', Cost: ' || v_inchiriere.COST_TOTAL || ' lei');

END LOOP;
CLOSE c_inchirieri;
END;
/
EXEC Afisare_Inchirieri(2);

--FUNCTII
--1.Calculare cost total între?inere pentru o trotinetă
CREATE OR REPLACE FUNCTION Cost_Intretinere_Trotineta (
p_id_trotineta IN NUMBER
) RETURN NUMBER IS
v_cost_total NUMBER;
BEGIN
SELECT NVL(SUM(COST_INTRETINERE), 0)
INTO v_cost_total
FROM LICAAMALIA_INTRETINERE_TROTINETE
WHERE ID_TROTINETA = p_id_trotineta;

RETURN v_cost_total;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RETURN 0; -- Dacă trotineta nu are înregistrări de între?inere
END;
/
SET SERVEROUTPUT ON;

DECLARE
v_cost_total NUMBER;
BEGIN
v_cost_total := Cost_Intretinere_Trotineta(1);
DBMS_OUTPUT.PUT_LINE('Cost total întreținere pentru trotineta 1: ' || v_cost_total);
END;
/

--2. Calculare distan?a totală parcursă de o trotinetă
CREATE OR REPLACE FUNCTION Distanța_Totala_Trotineta (
p_id_trotineta IN NUMBER
) RETURN NUMBER IS
v_distanța_totala NUMBER;
BEGIN
SELECT NVL(SUM(KILOMETRI_PARCURSI), 0)
INTO v_distanța_totala
FROM LICAAMALIA_RAPORT_TROTINETE
WHERE ID_TROTINETA = p_id_trotineta;

RETURN v_distanța_totala;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RETURN 0; -- Dacă trotineta nu are înregistrări în raport
END;
/

SET SERVEROUTPUT ON;

DECLARE
v_distanța_totala NUMBER;
BEGIN
v_distanța_totala := Distanța_Totala_Trotineta(3);
DBMS_OUTPUT.PUT_LINE('Distanța totală parcursă de trotineta 3: ' || v_distanța_totala || ' km');
END;
/

--3. Calculare cost total închirieri pentru un utilizator
CREATE OR REPLACE FUNCTION Cost_Total_Inchirieri (
p_id_utilizator IN NUMBER
) RETURN NUMBER IS
v_cost_total NUMBER;
BEGIN
-- Calcularea costului total al închirierilor pentru utilizatorul dat
SELECT NVL(SUM(COST_TOTAL), 0)
INTO v_cost_total
FROM LICAAMALIA_INCHIRIERI
WHERE ID_UTILIZATOR = p_id_utilizator;

RETURN v_cost_total;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RETURN 0; -- Dacă utilizatorul nu are închirieri, returnăm 0
END;
/
SET SERVEROUTPUT ON;

DECLARE
v_cost_total NUMBER;
BEGIN
v_cost_total := Cost_Total_Inchirieri(7);
DBMS_OUTPUT.PUT_LINE('Cost total al închirierilor pentru utilizatorul cu ID 7: ' || v_cost_total || ' lei');
END;
/

--4. Calculare vârstă utilizator
CREATE OR REPLACE FUNCTION Varsta_Utilizator (
p_id_utilizator IN NUMBER
) RETURN NUMBER IS
v_varsta NUMBER;
BEGIN
SELECT VARSTA
INTO v_varsta
FROM LICAAMALIA_UTILIZATORI
WHERE ID_UTILIZATOR = p_id_utilizator;

RETURN v_varsta;
EXCEPTION
WHEN NO_DATA_FOUND THEN
RETURN NULL; -- Dacă utilizatorul nu există
END;
/

SET SERVEROUTPUT ON;

DECLARE
v_varsta NUMBER;
BEGIN
v_varsta := Varsta_Utilizator(5);
IF v_varsta IS NOT NULL THEN
DBMS_OUTPUT.PUT_LINE('Vârsta utilizatorului cu ID 5: ' || v_varsta || ' ani');
ELSE
DBMS_OUTPUT.PUT_LINE('Nu există utilizator cu ID-ul 5. ');
END IF;
END;
/

```