

PROIECT Sisteme de Gestiune a Bazelor de Date

-Centru de fitness-

Proiect realizat de : Codarcea Alexandru-Christian, grupa 232

1. Modelul bazei de date pentru Managementul unui Club de Fitness are scopul de a gestiona eficient diverse aspecte ale clubului de fitness. Prin intermediul acestei baze de date, se dorește o gestionare centralizată a informațiilor legate de membrii clubului, clase, echipamente, locații, sponsori, vestiare, angajați și abonamente.

Utilitatea bazei de date include:

- Gestionarea conturilor membrilor, inclusiv informații personale, detalii despre abonament.
- Programarea și gestionarea claselor, inclusiv detalii despre clasă, antrenori și prezența membrilor.
- Urmărirea și gestionarea utilizării echipamentelor din club.
- Administrarea informațiilor despre locațiile clubului și sponsorii implicați.
- Gestionarea vestiarelor și a personalului responsabil de acestea.
- Gestionarea angajaților clubului.

În cadrul modelului bazei de date pentru Managementul unui Club de Fitness, sunt impuse următoarele constrângerî și reguli:

- Fiecare entitate trebuie să aibă un identificator unic (cheie primară).
- Relațiile dintre entități trebuie să fie definite și menționate în mod corespunzător, utilizând verbe adecvate (ex: "are", "detine", "ingrijeste", "cumpara", "creeaza", "foloseste", "coordoneaza").
- Cardinalitățile relațiilor trebuie specificate în funcție de logica modelului, indicând numărul minim și maxim de legături între entități (ex: 1(0) la M(0), M(0) la M(0)).
- Atributele fiecărei entități trebuie definite, specificând tipul de date, constrângerile și valorile隐式 (dacă există).
- Un membru poate să își cumpere un singur abonament la o sală.
- Membrii trebuie să aibă peste 16 ani pentru a fi eligibili să cumpere un abonament.
- Vor exista minim 2 vestiare (unul pentru barbati și unul pentru femei).
- Capacitatea minima a unui vestiar este de 25.

- O clasa poate avea un numar intre 2 si 10 participanti.
- O sala poate avea mai multe locatii sub acelasi brand (chiar si in orase/tari diferite).
- O sala poate avea mai multi sponsori, iar sponsorii pot sponsoriza mai multe sali.
- Există un singur contract intre un sponsor si o sala.
- O sala poate avea mai multe echipamente.
- O sala poate avea mai multi angajati.
- O sala poate avea un numar infinit de membrii, atata timp cat este cerere, sala ofera abonamente oamenilor ce vor sa devina membrii.
- Angajatii salii pot fi ingrijitori, antrenori sau nelistati deocamdata.
- Mai multi ingrijitori pot ingrijii mai multe vestiare.
- Un antrenor poate sa nu aiba o specializare fixata.
- Un antrenor coordoneaza mai multe clase si mai multi membrii.
- Un antrenor creeaza programe de fitness.
- Mai multe clase pot impartasi mai multe programe.
- Clasele au structura comună pentru fiecare sala din acest lant de sali.

Pentru modelul de date referitor la Managementul unui Club de Fitness, structurile SALA, ECHIPAMENT, LOCATIE, SPONSOR, ANGAJAT, INGRIJITOR, ANTRENOR, VESTIAR, MEMBRU, ABONAMENT, CLASA, PROGRAME reprezinta entitati. INGRIJITOR si ANTRENOR sunt subentitati ale entitatii ANGAJAT.

SALA = sala de fitness. Cheia primara este id_sala#.

ECHIPAMENT = echipament ce se gaseste in sala de fitness. Cheia primara este id_echipament#.

LOCATIE = locatia salii de fitness. Cheia primara este id_adresa#.

SPONSOR = sponsor al salii de fitness. Cheia primara este id_sponsor#.

ANGAJAT = angajat al salii de fitness. Cheia primara este id_angajat#.

INGRIJITOR = angajat ce se ocupa cu ingrijirea vestiarelor si curatarea salii. Cheia primara este id_angajat#.

ANTRENOR = antrenor ce se ocupa de clase, de membrii si de crearea programelor de fitness. Cheia primara este id_angajat#.

VESTIAR = vestiarul salii. Cheia primara este id_vestiar#.

PROGRAME = program de fitness/plan de workout, facut de antrenor. Cheia primara este id_antrenament#.

CLASA = clasa la care participa membrii salii, impreuna cu un antrenor. Cheia primara este id_clasa#.

MEMBRU = membru al salii de fitness. Cheia primara este id_membru#.

ABONAMENT = abonamentul pe care il cumpara membrii salii de fitness. Cheia primara este id_abonament#.

Descrierea relațiilor:

SALA_are_SPONSOR = relație care leaga entitatile SALA si SPONSOR, reflectand legatura dintre acestea (ce sponsori au salile). Ea are cardinalitatea minima 0:0 si cardinalitatea maxima M:M.

SALA_are_ECHIPAMENT = relație care leagă entitățile SALA și ECHIPAMENT, reflectând legătura dintre acestea (ce echipamente are o sală). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:M.

SALA_detine_LOCATIE= relație care leagă entitățile SALA și LOCATIE, reflectând legătura dintre acestea (ce locatii are o sală). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:M.

SALA_are_ANGAJAT = relație care leagă entitățile SALA și ANGAJAT, reflectând legătura dintre acestea (ce angajati are o sală). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:M.

SALA_are_MEMBRU = relație care leagă entitățile SALA și MEMBRU, reflectând legătura dintre acestea (ce membrii are o sală). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:M.

SALA_poseda_VESTIAR = relație care leagă entitățile SALA și VESTIAR, reflectând legătura dintre acestea (ce vestiare are o sală). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:M.

MEMBRU_cumpara_ABONAMENT = relație care leagă entitățile MEMBRU și ABONAMENT, reflectând legătura dintre acestea (ce abonament cumpara un membru). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:1.

ANTRENOR_creeaza_PROGRAME = relație care leagă entitățile ANTRENOR și ABONAMENT, reflectând legătura dintre acestea (ce programe creeaza un antrenor). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă 1:M.

INGRIJITOR_ingrijeste_VESTIAR = relație care leagă entitățile INGRIJITOR și VESTIAR, reflectând legătura dintre acestea (ce vestiare ingrijesc ingrijitorii). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă M:M.

CLASA_foloseste_PROGRAME = relație care leagă entitățile CLASA și ABONAMENT, reflectând legătura dintre acestea (ce programe folosesc clasele). Ea are cardinalitatea minimă 0:0 și cardinalitatea maximă M:M.

ANTRENOR_coordoneaza_CLASE_coordoneaza_MEMBRU = relatie de tip 3 ce leaga entitatile ANTRENOR, CLASA si MEMBRU, reflectand clasele pe care le coordoneaza antrenorul, la care participa membrii pe care tot antrenorul ii coordoneaza.

ANGAJAT_ISA_INGRIJITOR = relatia care leaga entitatile ANGAJAT si INGRIJITOR, reflectand legatura dintre acestea (un ingrijitor este un angajat). Ea are cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

ANGAJAT_ISA_ANTRENOR = relatia care leaga entitatile ANGAJAT si ANTRENOR, reflectand legatura dintre acestea (un antrenor este un angajat). Ea are cardinalitatea minimă 1:0 și cardinalitatea maximă 1:1.

Descrierea atributelor:

SALA are urmatoarele atribute :

- id_sala = variabila de tip intreg, care reprezinta id-ul salii.
- nume_sala = variabila de tip caracter, care reprezinta numele salii.

SPONSOR are urmatoarele atribute :

- id_sponsor = variabila de tip intreg, care reprezinta id-ul sponsorului.
- nume_sponsor = variabila de tip caracter, care reprezinta numele sponsorului.

LOCATIE are urmatoarele atribute :

- id_adresa = variabila de tip intreg, care reprezinta id-ul locatiei.
- oras_locatie = variabila de tip caracter, care reprezinta numele orasului locatiei.
- tara_locatie = variabila de tip caracter, care reprezinta numele tarii locatiei.

ECHIPAMENT are urmatoarele atribute :

- id_echipament = variabila de tip intreg, care reprezinta id-ul echipamentului.
- nume = variabila de tip caracter, care reprezinta numele echipamentului.

ANGAJAT are urmatoarele atribute :

- id_angajat = variabila de tip intreg, care reprezinta id-ul angajatului.
- nume_angajat = variabila de tip caracter, care reprezinta numele angajatului.

ANTRENOR are urmatoarele atribute :

- id_angajat = variabila de tip intreg, care reprezinta id-ul angajatului.
- nume_angajat = variabila de tip caracter, care reprezinta numele angajatului.
- specializare = variabila de tip caracter, care reprezinta specializarea antrenorului.

INGRIJITOR are urmatoarele atribute :

- id_angajat = variabila de tip intreg, care reprezinta id-ul angajatului.
- nume_angajat = variabila de tip caracter, care reprezinta numele angajatului.

VESTIAR are urmatoarele atribute :

- id_vestiar = variabila de tip intreg, care reprezinta id-ul vestiarului.
- tip_vestiar = variabila de tip caracter, luand valorile M sau F, de lungime 1, care reprezinta tipul vestiarului (de barbati, sau de femei)
- capacitate = variabila de tip intreg, un numar intre 10 si 140, care reprezinta capacitatea vestiarului

PROGRAME are urmatoarele atribute :

- id_antrenament = variabila de tip intreg, care reprezinta id-ul programului.
- nume = variabila de tip caracter, care reprezinta numele programului.
- timp_start = variabila de tip data (time), care reprezinta ora de incepere a programului.
- timp_final = variabila de tip data (time), care reprezinta ora de sfarsit a programului.

CLASA are urmatoarele atribute :

- id_clasa = variabila de tip intreg, care reprezinta id-ul clasei.
- data = variabila de tip data calendaristica, care reprezinta data in care se efectueaza clasa.

MEMBRU are urmatoarele atribute :

- id_membru = variabila de tip intreg, care reprezinta id-ul membrului.
- nume = variabila de tip caracter, care reprezinta numele membrului.
- data_nasterii = variabila de tip data calendaristica care reprezinta data de nastere a membrului.

ABONAMENT are urmatoarele atribute :

- id_abonament = variabila de tip intreg, care reprezinta id-ul abonamentului.
- tip_abonament = variabila de tip caracter, care reprezinta tipul abonamentului (Bronze, Silver, Gold, Platinum, Diamond).
- pret = variabila de tip decimal care reprezinta pretul abonamentului.

Schemele relationale sunt :

SALA(id_sala#, nume_sala);

SPONSOR(id_sponsor#, nume_sponsor);

CONTRACT_SPONSOR(id_sala#, id_sponsor#, data_contract);

ECHIPAMENT(id_echipament#, nume);

LOCATIE(id_adresa#, oras_locatie, tara_locatie);

ANGAJAT(id_angajat#, nume_angajat);

INGRIJITOR(id_angajat#);

ANTRENOR(id_angajat#, specializare);

INGRIJESTE(id_ingrijitor#, id_vestiar#);

VESTIAR(id_vestiar#, tip_vestiar, capacitate);

PROGRAME(id_antrenament#, nume, timp_start, timp_final);

FOLOSESTE(id_antrenament#, id_clasa#);

CLASA(id_clasa#, data);

FORMULAR(id_clasa#, id_antrenor#, id_membru#);

MEMBRU(id_membru#, nume, data_nasterii);

ABONAMENT(id_abonament#, tip_abonament, pret);

2,3.

Diagrama entitate-relație :

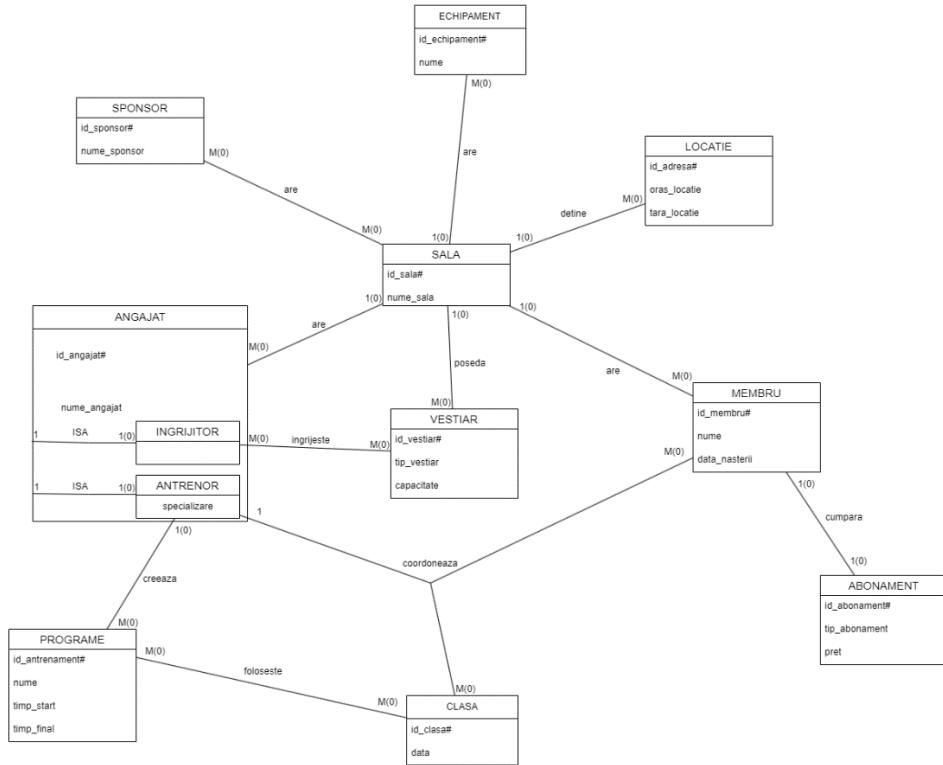
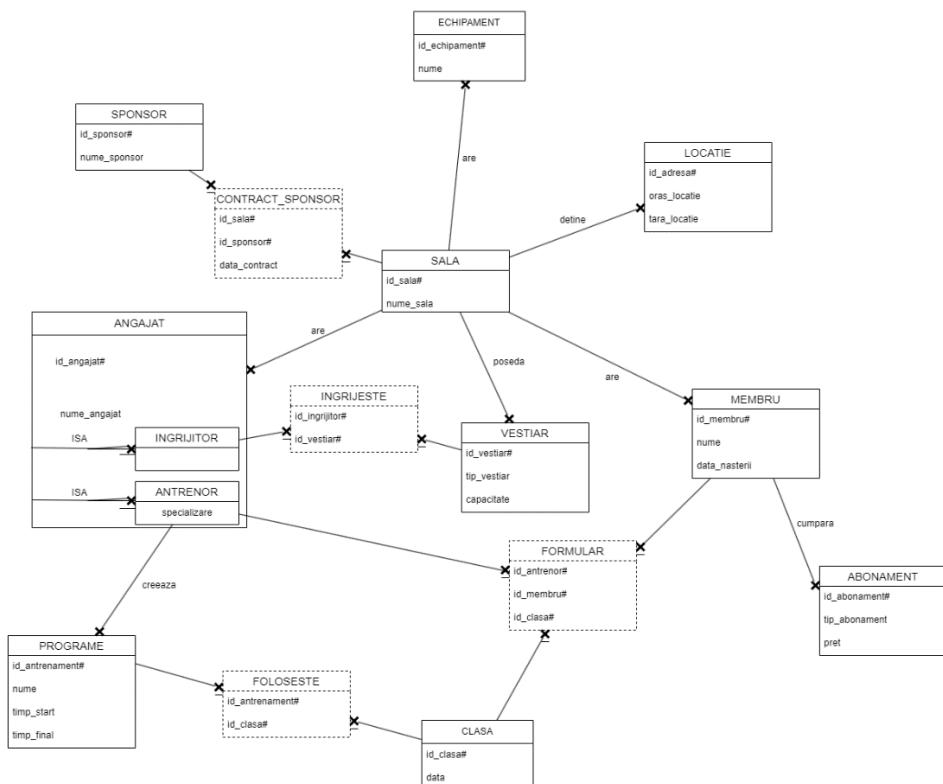


Diagrama conceptuală :



4. CREATE SEQUENCE sala_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE sponsor_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE echipament_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE locatie_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE angajat_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE vestiar_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE antrenament_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE clasa_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE membru_seq START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE abonament_seq START WITH 1 INCREMENT BY 1;

The screenshot shows the Oracle SQL Developer interface. The 'Connections' sidebar shows a connection named 'Sala'. The 'Worksheet' tab contains the SQL code for creating sequences. The 'Script Output' tab shows the results of the execution, listing each sequence created along with its start value and increment.

```
--Create Sequence
CREATE SEQUENCE sala_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE sponsor_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE echipament_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE locatie_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE angajat_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE vestiar_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE antrenament_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE clasa_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE membru_seq START WITH 1 INCREMENT BY 1;
CREATE SEQUENCE abonament_seq START WITH 1 INCREMENT BY 1;
```

Sequence	Start Value	Increment
SALA_SEQ	1	1
SPONSOR_SEQ	1	1
ECHIPAMENT_SEQ	1	1
LOCATIE_SEQ	1	1
ANGAJAT_SEQ	1	1
VESTIAR_SEQ	1	1
ANTRENNAMENT_SEQ	1	1
CLASA_SEQ	1	1
MEMBRU_SEQ	1	1
ABONAMENT_SEQ	1	1

CREATE TABLE SALA (

id_sala INT DEFAULT sala_seq.NEXTVAL PRIMARY KEY,

nume_sala VARCHAR(50)

);

CREATE TABLE SPONSOR (

```
    id_sponsor INT DEFAULT sponsor_seq.NEXTVAL PRIMARY KEY,  
    nume_sponsor VARCHAR(50)  
);
```

```
CREATE TABLE CONTRACT_SPONSOR (  
    id_sala INT,  
    id_sponsor INT,  
    data_contract DATE,  
    PRIMARY KEY (id_sala, id_sponsor),  
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala),  
    FOREIGN KEY (id_sponsor) REFERENCES SPONSOR(id_sponsor)  
);
```

```
CREATE TABLE ECHIPAMENT (  
    id_echipament INT DEFAULT echipament_seq.NEXTVAL PRIMARY KEY,  
    nume VARCHAR(50),  
    id_sala INT,  
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)  
);
```

```
CREATE TABLE LOCATIE (  
    id_adresa INT DEFAULT locatie_seq.NEXTVAL PRIMARY KEY,  
    oras_locatie VARCHAR(50),  
    tara_locatie VARCHAR(50),  
    id_sala INT,  
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)  
);
```

```
CREATE TABLE ANGAJAT (
```

```
    id_angajat INT DEFAULT angajat_seq.NEXTVAL PRIMARY KEY,  
    nume_angajat VARCHAR(50),  
    id_sala INT,  
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)  
);
```

```
CREATE TABLE INGRIJITOR (  
    id_angajat INT PRIMARY KEY REFERENCES ANGAJAT(id_angajat)  
);
```

```
CREATE TABLE ANTRENOR (  
    id_angajat INT PRIMARY KEY REFERENCES ANGAJAT(id_angajat),  
    specializare VARCHAR(50)  
);
```

```
CREATE TABLE VESTIAR (  
    id_vestiar INT DEFAULT vestiar_seq.NEXTVAL PRIMARY KEY,  
    tip_vestiar CHAR(1),  
    capacitate INT,  
    id_sala INT,  
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala),  
    CONSTRAINT ck_capacitate CHECK (capacitate >= 25)  
);
```

```
CREATE TABLE INGRIJESTE (  
    id_ingrijitor INT,  
    id_vestiar INT,  
    PRIMARY KEY (id_ingrijitor, id_vestiar),  
    FOREIGN KEY (id_ingrijitor) REFERENCES INGRIJITOR(id_angajat),
```

```
FOREIGN KEY (id_vestiar) REFERENCES VESTIAR(id_vestiar)
);

CREATE TABLE PROGRAME (
    id_antrenament INT DEFAULT antrenament_seq.NEXTVAL PRIMARY KEY,
    nume VARCHAR(50),
    timp_start DATE,
    timp_final DATE,
    id_angajat INT,
    FOREIGN KEY (id_angajat) REFERENCES ANTRENOR(id_angajat)
);
```

```
CREATE TABLE CLASA (
    id_clasa INT DEFAULT clasa_seq.NEXTVAL PRIMARY KEY,
    data DATE
);
```

```
CREATE TABLE FOLOSESTE (
    id_antrenament INT,
    id_clasa INT,
    PRIMARY KEY (id_antrenament, id_clasa),
    FOREIGN KEY (id_antrenament) REFERENCES PROGRAME(id_antrenament),
    FOREIGN KEY (id_clasa) REFERENCES CLASA(id_clasa)
);
```

```
CREATE TABLE MEMBRU (
    id_membru INT DEFAULT membru_seq.NEXTVAL PRIMARY KEY,
    nume VARCHAR(50),
    data_nasterii DATE,
```

```
    id_sala INT,  
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)  
);
```

```
CREATE TABLE ABONAMENT (  
    id_abonament INT DEFAULT abonament_seq.NEXTVAL PRIMARY KEY,  
    tip_abonament VARCHAR(50),  
    pret DECIMAL(10, 2),  
    id_membru INT,  
    FOREIGN KEY (id_membru) REFERENCES MEMBRU(id_membru),  
    CONSTRAINT uc_id_membru UNIQUE (id_membru)  
);
```

```
CREATE TABLE FORMULAR (  
    id_clasa INT,  
    id_antrenor INT,  
    id_membru INT,  
    PRIMARY KEY (id_clasa, id_antrenor, id_membru),  
    FOREIGN KEY (id_clasa) REFERENCES CLASA(id_clasa),  
    FOREIGN KEY (id_antrenor) REFERENCES ANTRENOR(id_angajat),  
    FOREIGN KEY (id_membru) REFERENCES MEMBRU(id_membru)  
);
```

```

--Create TableSal
CREATE TABLE SALA (
    id_sala INT DEFAULT sala_seq.NEXTVAL PRIMARY KEY,
    nume_sala VARCHAR(50)
);

CREATE TABLE SPONSOR (
    id_sponsor INT DEFAULT sponsor_seq.NEXTVAL PRIMARY KEY,
    nume_sponsor VARCHAR(50)
);

CREATE TABLE CONTRACT_SPONSOR (
    id_contract INT,
    id_sponsor INT,
    data_contract DATE,
    PRIMARY KEY (id_sala, id_sponsor),
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala),
    FOREIGN KEY (id_sponsor) REFERENCES SPONSOR(id_sponsor)
);

CREATE TABLE ECHIPAMENT (
    id_echipament INT DEFAULT echipament_seq.NEXTVAL PRIMARY KEY,
    nume_echipament VARCHAR(50),
    id_sala INT,
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)
);

CREATE TABLE LOCATIE (
    id_adresa INT DEFAULT locatie_seq.NEXTVAL PRIMARY KEY,
    ora_locatie VARCHAR(50),
    tarz_locatie VARCHAR(50),
    id_sala INT,
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)
);

CREATE TABLE ANGAJAT (
    id_angajat INT DEFAULT angajat_seq.NEXTVAL PRIMARY KEY,
    nume_angajat VARCHAR(50),
    id_sala INT,
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)
);

CREATE TABLE INGINIER (
    id_angajat INT PRIMARY KEY REFERENCES ANGAJAT(id_angajat)
);

CREATE TABLE ANTRENOR (
    id_angajat INT PRIMARY KEY REFERENCES ANGAJAT(id_angajat),
    specialitate VARCHAR(50)
);

```

```

--CREATE TABLE ANGREM (
--    id_angajat INT PRIMARY KEY REFERENCES ANGAJAT(id_angajat),
--    specialitate VARCHAR(50)
--);
--CREATE TABLE VESTIAR (
--    id_vestiar INT DEFAULT vestiar_seq.NEXTVAL PRIMARY KEY,
--    tip_vestiar CHAR(1),
--    capacitate INT,
--    id_sala INT,
--    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala),
--    CONSTRAINT ck_capacitate CHECK (capacitate >= 25)
--);
--CREATE TABLE INGILINSTE (
--    id_ingilinste INT,
--    id_vestiar INT,
--    FOREIGN KEY (id_vestiar) REFERENCES VESTIAR(id_vestiar),
--    FOREIGN KEY (id_ingilinste) REFERENCES INGINIER(id_angajat),
--    FOREIGN KEY (id_vestiar) REFERENCES VESTIAR(id_vestiar)
--);
--CREATE TABLE PROGRAME (
--    id_antrenament INT DEFAULT antrenament_seq.NEXTVAL PRIMARY KEY,
--    nume VARCHAR(50),
--    timp_start DATE,
--    timp_finisare DATE,
--    id_angajat INT,
--    FOREIGN KEY (id_angajat) REFERENCES ANGREM(id_angajat)
--);
--CREATE TABLE CLASA (
--    id_clasa INT DEFAULT clasa_seq.NEXTVAL PRIMARY KEY,
--    data DATE
--);
--CREATE TABLE FOLGOESTE (
--    id_antrenament INT,
--    id_clasa INT,
--    PRIMARY KEY (id_antrenament, id_clasa),
--    FOREIGN KEY (id_antrenament) REFERENCES PROGRAME(id_antrenament),
--    FOREIGN KEY (id_clasa) REFERENCES CLASA(id_clasa)
--);
--CREATE TABLE MEMBRIU (
--    id_membru INT DEFAULT membru_seq.NEXTVAL PRIMARY KEY,
--    nume VARCHAR(50),
--    data_nasterii DATE,
--    id_sala INT
--);

```

```

CREATE TABLE SALA (
    id_sala INT DEFAULT sala_seq.NEXTVAL PRIMARY KEY,
    nume_sala VARCHAR(50),
    time_start DATE,
    time_end DATE,
    id_angajat INT,
    FOREIGN KEY (id_angajat) REFERENCES ANGRENR(id_angajat)
);

CREATE TABLE CLASA (
    id_clasa INT DEFAULT clasa_seq.NEXTVAL PRIMARY KEY,
    data DATE
);

CREATE TABLE FOLGESTE (
    id_entrenament INT,
    id_clasa INT,
    PRIMAR KEY (id_entrenament, id_clasa),
    FOREIGN KEY (id_entrenament) REFERENCES PROGRAMME(id_entrenament),
    FOREIGN KEY (id_clasa) REFERENCES CLASA(id_clasa)
);

CREATE TABLE MEMBRI (
    id_membru INT DEFAULT membru_seq.NEXTVAL PRIMARY KEY,
    nume VARCHAR(50),
    data_nasterii DATE,
    id_sala INT,
    FOREIGN KEY (id_sala) REFERENCES SALA(id_sala)
);

CREATE TABLE ABONAMENT (
    id_abonament INT DEFAULT abonament_seq.NEXTVAL PRIMARY KEY,
    tip_abonament VARCHAR(50),
    pret_abonament DECIMAL(15, 2),
    id_membru INT,
    FOREIGN KEY (id_membru) REFERENCES MEMBRI(id_membru),
    CONSTRAINT uc_id_membru UNIQUE (id_membru)
);

CREATE TABLE FORMULAR (
    id_formular INT,
    id_entrenament INT,
    id_clasa INT,
    PRIMARY KEY (id_clasa, id_entrenament, id_membru),
    FOREIGN KEY (id_clasa) REFERENCES CLASA(id_clasa),
    FOREIGN KEY (id_entrenament) REFERENCES ANGRENR(id_angajat),
    FOREIGN KEY (id_membru) REFERENCES MEMBRI(id_membru)
);

```

Opened nodes (100); Saved file()

Table SALA created.

Table SPONJOR created.

Table CONTRACT_SPONJOR created.

Table EQUIPMENT created.

Table LOCATIE created.

Table ANGAJAT created.

Table INGRIGITOR created.

Table ANGRENR created.

Table VESTIAR created.

Table INGRIGESTE created.

Table PROGRAMME created.

Table CLASA created.

Table FOLGESTE created.

Table MEMBRI created.

Table ABONAMENT created.

Table FORMULAR created.

5. INSERT INTO SALA (nume_sala) VALUES ('World Class'); --SALA1

INSERT INTO SALA (nume_sala) VALUES ('Fit Gym'); --SALA2

INSERT INTO SALA (nume_sala) VALUES ('Fitness Zone'); --SALA3

INSERT INTO SALA (nume_sala) VALUES ('Gym Plus'); --SALA4

```
INSERT INTO SALA (nume_sala) VALUES ('Power Fitness'); --SALA5
```

```
SELECT * FROM SALA;
```

The screenshot shows the Oracle SQL Developer interface with a script file named 'Sala.sql' open. The code creates a table 'SALA' with columns 'id_sala', 'id_clase', 'id_entrenor', and 'id_membru'. It includes primary key and foreign key constraints linking to 'CLASA', 'ENTRENOR', and 'MEMBRI' tables. Below the table creation, a section titled '--Create Insertarilor' contains five INSERT statements for the 'SALA' table, each with a different name: 'World Class', 'Fit Gym', 'Fitness Zone', 'Gym Plus', and 'Power Fitness'. A 'Script Output' window shows the execution results, and a 'Query Result' window displays the data in the 'SALA' table:

ID_SALA	NUME_SALA
1	World Class
2	Fit Gym
3	Fitness Zone
4	Gym Plus
5	Power Fitness

SPONSOR :

```
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Nike');
```

```
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Adidas');
```

```
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Reebok');
```

```
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Puma');
```

```
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Under Armour');
```

```
SELECT * FROM SPONSOR;
```

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there's a tree view of connections and database objects. The main workspace contains a SQL Worksheet with the following code:

```
--Creates Inserturile
--SALA
INSERT INTO SALA (name_sala) VALUES ('World Class'); --SALA1
INSERT INTO SALA (name_sala) VALUES ('Fit Gym'); --SALA2
INSERT INTO SALA (name_sala) VALUES ('Fitness Center'); --SALA3
INSERT INTO SALA (name_sala) VALUES ('Gym Fit'); --SALA4
INSERT INTO SALA (name_sala) VALUES ('Power Fitness'); --SALA5
SELECT * FROM SALA;

--SPONSOR
INSERT INTO SPONSOR (name_sponsor) VALUES ('Nike');
INSERT INTO SPONSOR (name_sponsor) VALUES ('Adidas');
INSERT INTO SPONSOR (name_sponsor) VALUES ('Reebok');
INSERT INTO SPONSOR (name_sponsor) VALUES ('Puma');
INSERT INTO SPONSOR (name_sponsor) VALUES ('Under Armour');
SELECT * FROM SPONSOR;
```

In the bottom-right pane, the Script Output tab shows the results of the query:

ID_SPONSOR	NAME_SPONSOR
1	Nike
2	Adidas
3	Reebok
4	Puma
5	Under Armour

CONTRACT_SPONSOR :

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 1, TO_DATE('2023-01-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (2, 2, TO_DATE('2023-02-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (3, 3, TO_DATE('2023-03-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (4, 4, TO_DATE('2023-04-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (5, 5, TO_DATE('2023-05-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 2, TO_DATE('2023-06-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (2, 3, TO_DATE('2023-07-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (3, 4, TO_DATE('2023-08-01', 'YYYY-MM-DD'));
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (4, 5, TO_DATE('2023-09-01', 'YYYY-MM-DD'));

INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (5, 1, TO_DATE('2023-10-01', 'YYYY-MM-DD'));
```

SELECT * FROM CONTRACT_SPONSOR;

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. The left sidebar shows 'Connections' with 'Sala' selected, 'Oracle Connections', and 'Database Schema Service Connections'. The main area has tabs for 'Script', 'Query Builder', 'Worksheet', and 'Script Output'. The 'Worksheet' tab contains the following SQL code:

```

--SPONSOR
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Addida');
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Meeteo');
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Pana');
INSERT INTO SPONSOR (nume_sponsor) VALUES ('Under Armour');

SELECT * FROM SPONSOR;

--CONTRACT_SPONSOR
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 1, TO_DATE('2023-01-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (2, 2, TO_DATE('2023-02-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (3, 3, TO_DATE('2023-03-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (4, 4, TO_DATE('2023-04-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (5, 5, TO_DATE('2023-05-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 2, TO_DATE('2023-06-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (2, 3, TO_DATE('2023-07-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (3, 4, TO_DATE('2023-08-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (4, 5, TO_DATE('2023-09-01', 'YYYY-MM-DD'));
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (5, 1, TO_DATE('2023-10-01', 'YYYY-MM-DD'));

```

The 'Script Output' tab shows the results of the SELECT query:

ID_SALA	ID_SPONSOR	DATA_CONTRACT
1	1	01-JAN-23
2	2	01-FEB-23
3	3	01-MAR-23
4	4	01-APR-23
5	5	01-MAY-23
6	1	01-JUN-23
7	2	01-JUL-23
8	3	01-AUG-23
9	4	01-SEP-23
10	5	01-OCT-23

ECHIPAMENT :

```

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Bicicleta stationara', 1);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Stepper', 1);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Benzi de alergare', 1);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Banca pentru abdomene', 1);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Trambulina', 1);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Haltere', 2);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Bara de tractiuni', 2);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Greutati', 2);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Banca de antrenament', 2);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Sfoara', 2);

INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Gantere', 3);
```

```
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Saltele de yoga', 3);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Benzi elastice', 3);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Minge medicinala', 3);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Sac de box', 3);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Bara olimpica', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Discuri', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Helcometru', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Extensii de picioare', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Impins la piept inclinat', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Inele', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Scandura de echilibru', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Ghete pentru sarituri', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Squat-rack', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Coarda de sarit', 5);
```

SELECT * FROM ECHIPAMENT;

```

--ECHIPAMENT
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Bicicleta stationara', 1);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Sca de box', 3);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Banci elastice', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Trambulina', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Helometru', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Extensii de picioare', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Impins la piept inclinat', 4);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Inclina', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Scandura de echilibrul', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Ghece pentru saituri', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Squat-rack', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Coarde de sarit', 5);

SELECT * FROM ECHIPAMENT;

```

ID_EQUIPAMENT	NAME	ID_SALA
1	Bicicleta stationara	1
2	3 Sca de alergare	1
3	4 Banci pentru abdomen	1
5	5 Trambulina	1
6	6 Haltere	2
7	7 Banc de tractiuni	2
8	8 Grevatai	2
9	9 Banca de antrenament	2
10	10 Sfoara	2
11	11 Gantere	3
12	12 Saltele de yoga	3
13	13 Banci elastice	3
14	14 Minge medicinale	3
15	15 Sca de box	3
16	16 Banci olimpici	4
17	17 Banci de sarit	4
18	18 Helometru	4
19	19 Extensii de picioare	4
20	20 Impins la piept inclinat	4
21	21 Inclina	5
22	22 Scandura de echilibrul	5
23	23 Ghece pentru saituri	5
24	24 Squat-rack	5
25	25 Coarde de sarit	5

LOCATIE :

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Bucuresti', 'Romania', 1);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('New York', 'Statele Unite', 1);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Timisoara', 'Romania', 2);

```

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Iasi', 'Romania', 3);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Marseille', 'Franța', 3);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Hamburg', 'Germania', 3);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Berna', 'Elveția', 3);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Craiova', 'Romania', 4);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Nantes', 'Franța', 4);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Cologne', 'Germania', 4);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Oradea', 'Romania', 5);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Bordeaux', 'Franța', 5);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Stuttgart', 'Germania', 5);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('London', 'Marea Britanie', 5);

INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Beijing', 'China', 5);

```

SELECT * FROM LOCATIE;

The screenshot shows the Oracle SQL Developer interface with the following details:

- Toolbar:** File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help.
- Connections:** Shows a connection named "Sala".
- SQL Worksheet:** Contains the following SQL code:

```

--ECHIPAMENT
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Scandura de schilindru', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Ghetă pentru aspirație', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Sugat-rack', 5);
INSERT INTO ECHIPAMENT (nume, id_sala) VALUES ('Ocasă de sacit', 5);

SELECT * FROM ECHIPAMENT;

--LOCATIE
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('București', 'Romania', 1);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('New York', 'Statele Unite', 1);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Timișoara', 'Romania', 2);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Iasi', 'Romania', 3);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Marseille', 'Franța', 3);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Hamburg', 'Germania', 3);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Berna', 'Elveția', 3);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Craiova', 'Romania', 4);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Nantes', 'Franța', 4);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Cologne', 'Germania', 4);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Oradea', 'Romania', 5);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Bordeaux', 'Franța', 5);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Stuttgart', 'Germania', 5);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('London', 'Marea Britanie', 5);
INSERT INTO LOCATIE (oras_locatie, tara_locatie, id_sala) VALUES ('Beijing', 'China', 5);

```
- Script Output:** Shows the results of the SELECT query, which is the same as the code above.
- Table View:** Shows the LOCATIE table structure and data:

ID_LOCATIE	ID_ADRESA	ORAS_LOCATIE	TARA_LOCATIE	ID_SALA
1	1	București	Romania	1
2	2	New York	Statele Unite	1
3	3	Timișoara	Romania	2
4	4	Iasi	Romania	3
5	5	Marseille	Franța	3
6	6	Hamburg	Germania	3
7	7	Berna	Elveția	3
8	8	Craiova	Romania	4
9	9	Nantes	Franța	4
10	10	Cologne	Germania	4
11	11	Oradea	Romania	5
12	12	Bordeaux	Franța	5
13	13	Stuttgart	Germania	5
14	14	London	Marea Britanie	5
15	15	Beijing	China	5

ANGAJAT :

```

INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popescu Ion', 1);

INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Ionescu Maria', 1);

```

```
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Georgescu Andrei', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Constantin Elena', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Dumitru Radu', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popa Ana', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Gheorghe Alexandru', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Mihai Ioana', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Apostol Stefan', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Radulescu Andreea', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stancu Daniel', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Petrescu Gabriela', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Dragomir Valentin', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Preda Adriana', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Neagu Bogdan', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Gavrilă Elena', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Dobrescu Ionut', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Cojocaru Mihaela', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stanescu Andrei', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popovici Raluca', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Avram Mihai', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Florea Alexandra', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Munteanu Bogdana', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Voinea Alexandru', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Constantinescu Diana', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Moldovan Ionel', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Diaconu Gabriela', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stoica Adrian', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Gheorghiu Maria', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iacob Andrei', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stan Mirela', 5);
```

SELECT * FROM ANGAJAT;

```

INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Moldovan Ionel', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iacobina Gabriela', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popescu Andreea', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Gheorghiu Maria', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iacob Andrei', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stan Mirela', 5);

SELECT * FROM ANGAJAT;
  
```

ID_ANGAJAT	NUME_ANGAJAT	ID_SALA
1	Iopăciuc Ion	1
2	Ionescu Maria	1
3	Iordache Andrei	1
4	Constantin Elena	1
5	Dumitru Radu	1
6	Popa Ana	1
7	Gheorghie Alexandra	1
8	Mihai Ioana	2
9	Apostol Stefan	2
10	Rădulescu Andreea	2
11	Popescu Daniel	2
12	Artemescu Gheorghe	2
13	Iordache Valentin	3
14	Preda Adriana	3
15	Stoica Bogdan	3
16	Savrilici Elena	3
17	Dobrescu Ionut	3
18	Ciojoraru Mihaela	3
19	Stănescu Andrei	3
20	Popovici Radu	4
21	Avram Mihai	4
22	Florea Alexandra	4
23	Munteanu Bogdana	4
24	Voivulescu Alexandru	4
25	Constantinescu Diana	4
26	Popescu Iulian	5
27	Dianu Gabriel	5
28	Stoica Adrian	5
29	Gheorghiu Maria	5
30	Iacob Andrei	5
31	Stan Mirela	5

```

SELECT * FROM LOCATIE;

--ANGAJAT
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popescu Ion', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Ionescu Maria', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Georgescu Andrei', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Constantin Elena', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Dumitru Radu', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popa Ana', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Gheorghie Alexandra', 1);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Mihai Ioana', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Apostol Stefan', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Rădulescu Andreea', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popescu Daniel', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Artemescu Gheorghe', 2);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iordache Valentin', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Preda Adriana', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stoica Bogdan', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Savrilici Elena', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Dobrescu Ionut', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Ciojoraru Mihaela', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stănescu Andrei', 3);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Popovici Radu', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Avram Mihai', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Florea Alexandra', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Munteanu Bogdana', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Voivulescu Alexandru', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Constantinescu Diana', 4);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Moldovan Ionel', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iacobina Gabriela', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iacob Andrei', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Gheorghiu Maria', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Iacob Andrei', 5);
INSERT INTO ANGAJAT (nume_angajat, id_sala) VALUES ('Stan Mirela', 5);

SELECT * FROM ANGAJAT;
  
```

ID_ANGAJAT	NUME_ANGAJAT	ID_SALA
1	Iopăciuc Ion	1
2	Ionescu Maria	1
3	Iordache Andrei	1
4	Constantin Elena	1
5	Dumitru Radu	1
6	Popa Ana	1

INGRIJITOR :

--SALA1

```
INSERT INTO INGRIJITOR (id_angajat) VALUES (1);
INSERT INTO INGRIJITOR (id_angajat) VALUES (2);
INSERT INTO INGRIJITOR (id_angajat) VALUES (3);
--SALA2
INSERT INTO INGRIJITOR (id_angajat) VALUES (8);
INSERT INTO INGRIJITOR (id_angajat) VALUES (9);
--SALA3
INSERT INTO INGRIJITOR (id_angajat) VALUES (13);
INSERT INTO INGRIJITOR (id_angajat) VALUES (14);
INSERT INTO INGRIJITOR (id_angajat) VALUES (15);
INSERT INTO INGRIJITOR (id_angajat) VALUES (16);
--SALA4
INSERT INTO INGRIJITOR (id_angajat) VALUES (20);
INSERT INTO INGRIJITOR (id_angajat) VALUES (21);
INSERT INTO INGRIJITOR (id_angajat) VALUES (22);
--SALA5
INSERT INTO INGRIJITOR (id_angajat) VALUES (26);
INSERT INTO INGRIJITOR (id_angajat) VALUES (27);
```

```
SELECT * FROM INGRIJITOR;
```

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays 'Connections' (with one connection named 'Sala'), 'Reports' (with various report types), and a 'Script Output' section. The main area is a 'Worksheet' titled 'Query Builder'. It contains the following SQL code:

```
SELECT * FROM ANGAJAT;
--INGRIJITOR
--SALA1
INSERT INTO INGRIJITOR (id_angajat) VALUES (1);
INSERT INTO INGRIJITOR (id_angajat) VALUES (2);
INSERT INTO INGRIJITOR (id_angajat) VALUES (3);
--SALA2
INSERT INTO INGRIJITOR (id_angajat) VALUES (8);
INSERT INTO INGRIJITOR (id_angajat) VALUES (9);
--SALA3
INSERT INTO INGRIJITOR (id_angajat) VALUES (13);
INSERT INTO INGRIJITOR (id_angajat) VALUES (14);
INSERT INTO INGRIJITOR (id_angajat) VALUES (15);
INSERT INTO INGRIJITOR (id_angajat) VALUES (16);
--SALA4
INSERT INTO INGRIJITOR (id_angajat) VALUES (20);
INSERT INTO INGRIJITOR (id_angajat) VALUES (21);
INSERT INTO INGRIJITOR (id_angajat) VALUES (22);
--SALA5
INSERT INTO INGRIJITOR (id_angajat) VALUES (26);
INSERT INTO INGRIJITOR (id_angajat) VALUES (27);

SELECT * FROM INGRIJITOR;
```

Below the code, the 'Script Output' pane shows the results of the last query:

ID_ANGAJAT
1
2
3
8
9
13
14
15
16
20
21
22
26
27

ANTRENOR :

--SALA1

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (4, 'Fitness');
```

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (5, 'Yoga');
```

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (6, 'Pilates');
```

--SALA2

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (10, 'Crossfit');
```

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (11, NULL);
```

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (12, 'Aerobic');
```

--SALA3

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (17, 'Crossfit');
```

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (18, 'Aerobic');
```

```
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (19, 'Powerlifting');
```

--SALA4

```

INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (23, 'Bodybuilding');

INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (24, 'Aerobic');

--SALA5

INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (28, 'Yoga');

INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (29, 'Fitness');

INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (30, 'Aerobic');

INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (31, NULL);

```

SELECT * FROM ANTRENOR;

The screenshot shows the Oracle SQL Developer interface with the following details:

- Connections:** Shows a connection named "Sala".
- SQL Worksheet:** Contains the SQL code for inserting data into the ANTRENOR table.
- Script Output:** Shows the results of the query, displaying 15 rows of data from the ANTRENOR table.

```

SELECT * FROM ANTRENOR;

--ANTRENOR

--SALA1
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (4, 'Fitness');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (5, 'Yoga');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (6, 'Plates');
--SALA2
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (10, 'Crossfit');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (11, NULL);
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (12, 'Aerobic');
--SALA3
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (17, 'Crossfit');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (18, 'Aerobic');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (19, 'Powerlifting');
--SALA4
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (23, 'Bodybuilding');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (24, 'Aerobic');
--SALA5
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (28, 'Yoga');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (29, 'Fitness');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (30, 'Aerobic');
INSERT INTO ANTRENOR (id_angajat, specializare) VALUES (31, NULL);

SELECT * FROM ANTRENOR;

```

ID_ANGAJAT	SPECIALIZARE
1	Fitness
2	Yoga
3	Plates
4	Crossfit
5	(null)
6	Aerobic
7	Crossfit
8	Aerobic
9	Powerlifting
10	Bodybuilding
11	Aerobic
12	Yoga
13	Fitness
14	Aerobic
15	(null)

VESTIAR :

```

--SALA1

INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 50, 1);

INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 60, 1);

--SALA2

INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 40, 2);

INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 30, 2);

```

--SALA3

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 100, 3);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 80, 3);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 120, 3);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 90, 3);
```

--SALA4

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 40, 4);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 35, 4);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 55, 4);
```

--SALA5

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 70, 5);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 60, 5);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 90, 5);
```

```
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 80, 5);
```

SELECT * FROM VESTIAR;

```
--VESTIAR
--SALA1
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 50, 1);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 60, 1);
--SALA2
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 40, 2);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 30, 2);
--SALA3
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 100, 3);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 80, 3);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 120, 3);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 90, 3);
--SALA4
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 40, 4);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 35, 4);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 55, 4);
--SALA5
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 70, 5);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 60, 5);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('M', 90, 5);
INSERT INTO VESTIAR (tip_vestiar, capacitate, id_sala) VALUES ('F', 80, 5);

SELECT * FROM VESTIAR;
```

ID_VESTIAR	TIP_VESTIAR	CAPACITATE	ID_SALA
1	1 M	50	1
2	2 F	60	1
3	3 M	40	2
4	4 F	30	2
5	5 M	100	3
6	6 F	80	3
7	7 M	120	3
8	8 F	50	3
9	9 M	40	4
10	10 F	35	4
11	11 M	55	4
12	12 M	70	5
13	13 F	60	5
14	14 M	90	5
15	15 F	80	5

INGRIESTE :

--SALA1

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (1, 1);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (2, 2);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (3, 1);

--SALA2

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (8, 4);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (9, 3);

--SALA3

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (13, 5);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (14, 6);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (15, 7);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (16, 8);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (16, 7);

--SALA4

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (20, 9);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (21, 10);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (22, 11);

--SALA5

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (26, 12);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (27, 13);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (26, 14);

INSERT INTO INGRIESTE (id_ingrijitor, id_vestiar) VALUES (27, 15);

```
SELECT * FROM INGRIESTE;
```

The screenshot shows the Oracle SQL Developer interface. The 'Worksheet' tab is active, displaying the following SQL code:

```
--SALA1
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (1, 1);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (2, 2);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (3, 3);
--SALA2
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (4, 4);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (5, 5);
--SALA3
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (6, 6);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (7, 7);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (8, 8);
--SALA4
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (9, 9);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (10, 10);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (11, 11);
--SALA5
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (12, 12);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (13, 13);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (14, 14);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (15, 15);
--SALA6
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (16, 16);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (17, 17);
--SALA7
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (18, 18);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (19, 19);
--SALA8
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (20, 20);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (21, 21);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (22, 22);
--SALA9
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (23, 23);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (24, 24);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (25, 25);
--SALA10
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (26, 26);
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (27, 27);
--SALA11
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (28, 28);
--SALA12
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (29, 29);
--SALA13
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (30, 30);
--SALA14
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (31, 31);
--SALA15
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (32, 32);
--SALA16
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (33, 33);
--SALA17
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (34, 34);
--SALA18
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (35, 35);
--SALA19
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (36, 36);
--SALA20
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (37, 37);
--SALA21
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (38, 38);
--SALA22
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (39, 39);
--SALA23
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (40, 40);
--SALA24
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (41, 41);
--SALA25
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (42, 42);
--SALA26
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (43, 43);
--SALA27
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (44, 44);
--SALA28
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (45, 45);
--SALA29
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (46, 46);
--SALA30
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (47, 47);
--SALA31
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (48, 48);
--SALA32
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (49, 49);
--SALA33
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (50, 50);
--SALA34
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (51, 51);
--SALA35
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (52, 52);
--SALA36
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (53, 53);
--SALA37
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (54, 54);
--SALA38
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (55, 55);
--SALA39
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (56, 56);
--SALA40
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (57, 57);
--SALA41
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (58, 58);
--SALA42
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (59, 59);
--SALA43
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (60, 60);
--SALA44
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (61, 61);
--SALA45
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (62, 62);
--SALA46
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (63, 63);
--SALA47
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (64, 64);
--SALA48
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (65, 65);
--SALA49
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (66, 66);
--SALA50
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (67, 67);
--SALA51
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (68, 68);
--SALA52
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (69, 69);
--SALA53
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (70, 70);
--SALA54
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (71, 71);
--SALA55
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (72, 72);
--SALA56
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (73, 73);
--SALA57
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (74, 74);
--SALA58
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (75, 75);
--SALA59
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (76, 76);
--SALA60
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (77, 77);
--SALA61
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (78, 78);
--SALA62
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (79, 79);
--SALA63
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (80, 80);
--SALA64
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (81, 81);
--SALA65
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (82, 82);
--SALA66
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (83, 83);
--SALA67
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (84, 84);
--SALA68
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (85, 85);
--SALA69
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (86, 86);
--SALA70
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (87, 87);
--SALA71
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (88, 88);
--SALA72
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (89, 89);
--SALA73
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (90, 90);
--SALA74
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (91, 91);
--SALA75
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (92, 92);
--SALA76
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (93, 93);
--SALA77
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (94, 94);
--SALA78
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (95, 95);
--SALA79
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (96, 96);
--SALA80
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (97, 97);
--SALA81
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (98, 98);
--SALA82
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (99, 99);
--SALA83
INSERT INTO INGRIESTE (id_ingrileitor, id_vestiar) VALUES (100, 100);
```

The 'Script Output' tab shows the results of the execution, which is a single row of data:

ID_INGRIESTE	ID_VESTIAR
100	100

PROGRAME :

--SALA1

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Antrenament cardio',
TO_DATE('2023-05-18 09:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-05-18 10:00', 'YYYY-MM-DD
HH24:MI'), 4);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Antrenament functional',
TO_DATE('2023-06-30 16:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-06-30 17:30', 'YYYY-MM-DD
HH24:MI'), 4);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Yoga relaxare',
TO_DATE('2023-08-12 10:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-08-12 11:00', 'YYYY-MM-DD
HH24:MI'), 5);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Pilates postural',
TO_DATE('2023-09-24 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-09-24 16:30', 'YYYY-MM-DD
HH24:MI'), 6);
```

--SALA2

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Antrenament de bază',
TO_DATE('2023-11-05 14:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-11-05 15:00', 'YYYY-MM-DD
HH24:MI'), 10);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Yoga ashtanga',  
TO_DATE('2023-12-17 18:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-12-17 19:30', 'YYYY-MM-DD  
HH24:MI'), 12);
```

--SALA3

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Crossfit challenge',  
TO_DATE('2024-01-29 13:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-01-29 14:30', 'YYYY-MM-DD  
HH24:MI'), 17);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Aerobic step up',  
TO_DATE('2024-03-11 16:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-03-11 17:00', 'YYYY-MM-DD  
HH24:MI'), 18);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Aerobic dance party',  
TO_DATE('2024-04-22 19:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-04-22 20:30', 'YYYY-MM-DD  
HH24:MI'), 19);
```

--SALA4

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Pilates sculpting',  
TO_DATE('2024-06-03 11:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-06-03 12:30', 'YYYY-MM-DD  
HH24:MI'), 23);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Functional circuit',  
TO_DATE('2024-07-15 15:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-07-15 16:00', 'YYYY-MM-DD  
HH24:MI'), 24);
```

--SALA5

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Aerobic intens',  
TO_DATE('2024-08-26 17:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-08-26 18:30', 'YYYY-MM-DD  
HH24:MI'), 28);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Aerobic dance mix',  
TO_DATE('2024-10-07 20:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-10-07 21:00', 'YYYY-MM-DD  
HH24:MI'), 29);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Pilates core',  
TO_DATE('2024-11-18 12:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-11-18 13:00', 'YYYY-MM-DD  
HH24:MI'), 30);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Yoga vinyasa',  
TO_DATE('2024-12-30 14:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-12-30 15:30', 'YYYY-MM-DD  
HH24:MI'), 31);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Antrenament functional',  
TO_DATE('2025-02-10 16:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-02-10 17:30', 'YYYY-MM-DD  
HH24:MI'), 31);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Yoga relaxare',  
TO_DATE('2025-03-23 10:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-03-23 11:00', 'YYYY-MM-DD  
HH24:MI'), 31);
```

```
INSERT INTO PROGRAME (nume, timp_start, timp_final, id_angajat) VALUES ('Pilates postural',  
TO_DATE('2025-05-04 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-05-04 16:30', 'YYYY-MM-DD  
HH24:MI'), 31);
```

```
SELECT * FROM PROGRAME;
```

Oracle SQL Developer : C:\Users\Alex\Desktop\Sala.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

Sala

SQL Worksheet - History

Worksheet - Query Builder

```
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament functional', 'TO_DATE('2023-05-18 09:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-05-18 10:00', 'YYYY-MM-DD HH24:MI'), 4);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament functional', 'TO_DATE('2023-06-30 16:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-06-30 17:30', 'YYYY-MM-DD HH24:MI'), 5);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates postural', 'TO_DATE('2023-09-24 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-09-24 16:30', 'YYYY-MM-DD HH24:MI'), 6);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament de basă', 'TO_DATE('2023-11-05 14:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-11-05 15:00', 'YYYY-MM-DD HH24:MI'), 10);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Yoga ashtanga', 'TO_DATE('2023-12-17 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-12-17 19:30', 'YYYY-MM-DD HH24:MI'), 12);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Crossfit challenge', 'TO_DATE('2024-01-29 13:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-01-29 14:30', 'YYYY-MM-DD HH24:MI'), 17);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic step up', 'TO_DATE('2024-03-11 16:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-03-11 17:00', 'YYYY-MM-DD HH24:MI'), 18);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic dance party', 'TO_DATE('2024-04-22 19:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-04-22 20:30', 'YYYY-MM-DD HH24:MI'), 19);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates sculpting', 'TO_DATE('2024-06-05 11:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-06-05 12:30', 'YYYY-MM-DD HH24:MI'), 23);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Functional circuits', 'TO_DATE('2024-07-03 15:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-07-03 16:00', 'YYYY-MM-DD HH24:MI'), 24);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic intense', 'TO_DATE('2024-09-10 17:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-09-10 18:30', 'YYYY-MM-DD HH24:MI'), 29);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic dance mix', 'TO_DATE('2024-10-07 20:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-10-07 21:00', 'YYYY-MM-DD HH24:MI'), 30);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates core', 'TO_DATE('2024-11-18 12:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-11-18 13:00', 'YYYY-MM-DD HH24:MI'), 30);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Yoga vinyasa', 'TO_DATE('2024-12-05 14:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-12-05 15:00', 'YYYY-MM-DD HH24:MI'), 31);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament functional', 'TO_DATE('2025-02-10 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-02-10 17:30', 'YYYY-MM-DD HH24:MI'), 31);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Yoga relaxare', 'TO_DATE('2025-03-23 10:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-03-23 11:00', 'YYYY-MM-DD HH24:MI'), 31);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates postural', 'TO_DATE('2025-05-04 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-05-04 16:30', 'YYYY-MM-DD HH24:MI'), 31);  
SELECT * FROM PROGRAM;
```

Reports

All Reports

Analytic View Reports

Data Dictionary Reports

Data Model Reports

OLAP Reports

TimetTen Reports

User Defined Reports

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Query Result 7 | Query Result 8 | Query Result 9 | Query Result 10

All Rows Fetched: 18 in 0.004 seconds

SQL

ID ANTRÉNAMENT | NUME | TEMP_START | TEMP_FINAL | ID_ANGAJAT

ID ANTRÉNAMENT	NUME	TEMP_START	TEMP_FINAL	ID_ANGAJAT
1	Antrenament cardio	15-MAY-23	18-MAY-23	4
2	Antrenament functional	30-JUN-23	30-JUN-23	4
3	Yoga relaxare	12-AUG-23	12-AUG-23	5
4	Pilates postural	24-SEP-23	24-SEP-23	6
5	Antrenament de basă	05-NOV-23	05-NOV-23	10
6	Yoga ashtanga	17-DEC-23	17-DEC-23	12
7	Crossfit challenge	28-JAN-24	28-JAN-24	17
8	Aerobic intense up	11-FEB-24	11-FEB-24	18
9	Aerobic dance party	22-APR-24	22-APR-24	19
10	Pilates sculpting	03-JUN-24	03-JUN-24	23
11	Functional circuit	14-JUL-24	14-JUL-24	24
12	Aerobic intense	26-AUG-24	26-AUG-24	26
13	Aerobic dance mix	07-OCT-24	07-OCT-24	29
14	Pilates core	19-NOV-24	19-NOV-24	30
15	Yoga vinyasa	30-DEC-24	30-DEC-24	31
16	Antrenamente functional	10-FEB-25	10-FEB-25	31
17	Yoga relaxare	23-MAR-25	23-MAR-25	31
18	Pilates postural	04-MAR-25	04-MAR-25	31

Opened nodes (100): Saved files (1)

Line 418 Column 24 | Insert | Modified | Versions

Oracle SQL Developer : C:\Users\Alex\Desktop\Sala.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections

Sala

SQL Worksheet - History

Worksheet - Query Builder

```
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament cardio', 'TO_DATE('2023-05-18 09:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-05-18 10:00', 'YYYY-MM-DD HH24:MI'), 4);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament functional', 'TO_DATE('2023-06-30 16:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-06-30 17:30', 'YYYY-MM-DD HH24:MI'), 5);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates postural', 'TO_DATE('2023-09-24 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-09-24 16:30', 'YYYY-MM-DD HH24:MI'), 6);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament de basă', 'TO_DATE('2023-11-05 14:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-11-05 15:00', 'YYYY-MM-DD HH24:MI'), 10);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Yoga ashtanga', 'TO_DATE('2023-12-17 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2023-12-17 19:30', 'YYYY-MM-DD HH24:MI'), 12);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Crossfit challenge', 'TO_DATE('2024-01-29 13:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-01-29 14:30', 'YYYY-MM-DD HH24:MI'), 17);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic step up', 'TO_DATE('2024-03-11 16:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-03-11 17:00', 'YYYY-MM-DD HH24:MI'), 18);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic dance party', 'TO_DATE('2024-04-22 19:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-04-22 20:30', 'YYYY-MM-DD HH24:MI'), 19);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates sculpting', 'TO_DATE('2024-06-05 11:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-06-05 12:30', 'YYYY-MM-DD HH24:MI'), 23);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Functional circuits', 'TO_DATE('2024-07-03 15:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-07-03 16:00', 'YYYY-MM-DD HH24:MI'), 24);  
--SALA  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic intense', 'TO_DATE('2024-09-10 17:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-09-10 20:00', 'YYYY-MM-DD HH24:MI'), 29);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic dance mix', 'TO_DATE('2024-10-07 20:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-10-07 21:00', 'YYYY-MM-DD HH24:MI'), 30);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates core', 'TO_DATE('2024-11-18 12:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-11-18 13:00', 'YYYY-MM-DD HH24:MI'), 30);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Yoga vinyasa', 'TO_DATE('2024-12-05 14:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-12-05 15:00', 'YYYY-MM-DD HH24:MI'), 31);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament functional', 'TO_DATE('2025-02-10 16:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-02-10 17:00', 'YYYY-MM-DD HH24:MI'), 31);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Yoga relaxare', 'TO_DATE('2025-03-23 10:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-03-23 11:00', 'YYYY-MM-DD HH24:MI'), 31);  
INSERT INTO PROGRAM (name, temp_start, temp_final, id_angajat) VALUES ('Pilates postural', 'TO_DATE('2025-05-04 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-05-04 16:30', 'YYYY-MM-DD HH24:MI'), 31);  
SELECT * FROM PROGRAM;
```

Reports

All Reports

Analytic View Reports

Data Dictionary Reports

Data Model Reports

OLAP Reports

TimetTen Reports

User Defined Reports

Script Output | Query Result | Query Result 1 | Query Result 2 | Query Result 3 | Query Result 4 | Query Result 5 | Query Result 6 | Query Result 7 | Query Result 8 | Query Result 9 | Query Result 10

All Rows Fetched: 18 in 0.004 seconds

SQL

ID ANTRÉNAMENT | NUME | TEMP_START | TEMP_FINAL | ID_ANGAJAT

ID ANTRÉNAMENT	NUME	TEMP_START	TEMP_FINAL	ID_ANGAJAT
1	Antrenament cardio	15-MAY-23	18-MAY-23	4
2	Antrenament functional	30-JUN-23	30-JUN-23	4
3	Yoga relaxare	12-AUG-23	12-AUG-23	5
4	Pilates postural	24-SEP-23	24-SEP-23	6
5	Antrenament de basă	05-NOV-23	05-NOV-23	10
6	Yoga ashtanga	17-DEC-23	17-DEC-23	12
7	Crossfit challenge	28-JAN-24	28-JAN-24	17
8	Aerobic step up	11-FEB-24	11-FEB-24	18
9	Aerobic dance party	22-APR-24	22-APR-24	19
10	Pilates sculpting	03-JUN-24	03-JUN-24	23
11	Functional circuit	14-JUL-24	14-JUL-24	24
12	Aerobic intense	26-AUG-24	26-AUG-24	26
13	Aerobic dance mix	07-OCT-24	07-OCT-24	29
14	Pilates core	19-NOV-24	19-NOV-24	30

Opened nodes (100): Saved files (1)

Line 418 Column 24 | Insert | Modified | Versions

CLASA :

```
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-18', 'YYYY-MM-DD'))
```

```
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-19', 'YYYY-MM-DD'))
```

```

INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-20', 'YYYY-MM-DD'));

INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-21', 'YYYY-MM-DD'));

INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-22', 'YYYY-MM-DD'));

```

SELECT * FROM CLASA;

The screenshot shows the Oracle SQL Developer interface. The SQL Worksheet tab contains the following SQL code:

```

--SALA
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Pilates sculpting', TO_DATE('2024-04-03 11:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-04-03 12:30', 'YYYY-MM-DD HH24:MI'), 23);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Funcional circuit', TO_DATE('2024-07-15 15:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-07-15 16:00', 'YYYY-MM-DD HH24:MI'), 24);
--SALA
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic intens', TO_DATE('2024-08-26 17:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-08-26 18:30', 'YYYY-MM-DD HH24:MI'), 25);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Aerobic dansă mix', TO_DATE('2024-09-10 19:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-09-10 21:00', 'YYYY-MM-DD HH24:MI'), 26);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Yoga relaxant', TO_DATE('2024-12-01 13:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-12-01 14:00', 'YYYY-MM-DD HH24:MI'), 27);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Yoga vinçate', TO_DATE('2024-12-30 14:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2024-12-30 15:30', 'YYYY-MM-DD HH24:MI'), 28);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Antrenament funcional', TO_DATE('2025-02-10 16:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-02-10 17:30', 'YYYY-MM-DD HH24:MI'), 29);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Yoga relaxans', TO_DATE('2025-03-23 10:00', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-03-23 11:00', 'YYYY-MM-DD HH24:MI'), 30);
INSERT INTO PROGRAMA (name, temp_start, temp_final, id_angajat) VALUES ('Pilates postural', TO_DATE('2025-05-04 15:30', 'YYYY-MM-DD HH24:MI'), TO_DATE('2025-05-04 16:30', 'YYYY-MM-DD HH24:MI'), 31);

SELECT * FROM CLASA;

--CLASA
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-18', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-19', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-20', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-21', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-22', 'YYYY-MM-DD'));

```

The Results tab shows the output of the SELECT query:

ID_CLASA	DATA
1	1.19-MAY-23
2	2.19-MAY-23
3	3.20-MAY-23
4	4.21-MAY-23
5	5.22-MAY-23

FOLOSESTE :

```

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (11, 1);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (12, 2);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (18, 1);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (12, 3);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (3, 2);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (7, 4);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (8, 3);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (15, 5);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (9, 1);

INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (3, 4);

```

SELECT * FROM FOLOSESTE;

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, and Help. The left sidebar displays 'Connections' (with one connection named 'Sala'), 'Reports' (with various report types listed), and 'Opened nodes (108); Saved files (1)'. The main workspace has a title bar 'SQL Worksheet: Sala.sql' and a tab 'Worksheet - Query Builder'. The code area contains two sections of SQL:

```
--CLASA
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-18', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-18', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-18', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-19', 'YYYY-MM-DD'));
INSERT INTO CLASA (data) VALUES (TO_DATE('2023-05-19', 'YYYY-MM-DD'));

--FOLOSESTE
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 1);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 2);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 1);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 3);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 2);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 4);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 3);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 1);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 2);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 3);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 4);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 2);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 3);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 4);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 1);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 2);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 3);
INSERT INTO FOLOSESTE (id_entrenament, id_clasa) VALUES (1, 4);

SELECT * FROM FOLOSESTE;
```

The results pane shows a table titled 'Script Output' with the following data:

ID_ENTRENAMENT	ID_CLASA
1	11
2	12
3	18
4	12
5	3
6	2
7	7
8	6
9	15
9	9
10	1
10	3
10	4

MEMBRU :

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Popescu Ion', TO_DATE('1990-03-15', 'YYYY-MM-DD'), 1);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Radu Maria', TO_DATE('1985-07-22', 'YYYY-MM-DD'), 1);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Ionescu Andrei', TO_DATE('1995-11-05', 'YYYY-MM-DD'), 1);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Georgescu Ana', TO_DATE('1988-09-18', 'YYYY-MM-DD'), 1);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Stanescu Mihaela', TO_DATE('1992-06-27', 'YYYY-MM-DD'), 1);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Dumitru Dan', TO_DATE('1991-02-10', 'YYYY-MM-DD'), 2);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Gheorghiu Elena', TO_DATE('1987-04-25', 'YYYY-MM-DD'), 2);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Mihai Andrei', TO_DATE('1994-08-12', 'YYYY-MM-DD'), 2);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Florescu Alina', TO_DATE('1989-01-03', 'YYYY-MM-DD'), 2);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Stefanescu Ioana', TO_DATE('1993-12-08', 'YYYY-MM-DD'), 2);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Diaconu Ionut', TO_DATE('1990-05-20', 'YYYY-MM-DD'), 2);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Popa George', TO_DATE('1986-10-17', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Gavril Alexandra', TO_DATE('1995-09-02', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Marinescu Adrian', TO_DATE('1991-06-14', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Barbu Gabriela', TO_DATE('1988-03-28', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Constantinescu Ana', TO_DATE('1993-11-10', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Dobre Radu', TO_DATE('1992-07-06', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Andrei Mihaela', TO_DATE('1987-04-19', 'YYYY-MM-DD'), 3);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Iacob Marius', TO_DATE('1990-09-12', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Voicu Ana', TO_DATE('1986-05-25', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Cristescu Andrei', TO_DATE('1992-12-03', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Munteanu Elena', TO_DATE('1989-02-15', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Alexandrescu Ana', TO_DATE('1994-11-07', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Popescu Ioan', TO_DATE('1993-07-01', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Dinu Mihaela', TO_DATE('1988-03-22', 'YYYY-MM-DD'), 4);
```

```
INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Ionescu Florin', TO_DATE('1991-06-09', 'YYYY-MM-DD'), 4);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Andrei Catalin', TO_DATE('1990-01-29', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Stan Maria', TO_DATE('1987-08-14', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Gheorghe Mihai', TO_DATE('1992-04-05', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Dumitrescu Elena', TO_DATE('1989-10-23', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Stefan Maria', TO_DATE('1994-03-11', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Popa Ionut', TO_DATE('1993-12-02', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Radu Alexandra', TO_DATE('1988-07-16', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Marin Florina', TO_DATE('1991-05-03', 'YYYY-MM-DD'), 5);

INSERT INTO MEMBRU (nume, data_nasterii, id_sala) VALUES ('Dobre Andrei', TO_DATE('1996-02-27', 'YYYY-MM-DD'), 5);
```

SELECT * FROM MEMBRU;

```

INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dobres Andrei', TO_DATE('1994-02-27', 'YYYY-MM-DD'), 1)
SELECT * FROM MEMBRU;

```

ID_MEMBRU	NAME	DATA_NASTERII	ID_SALA
1	Ionescu Ion	15-MAR-56	1
2	Rade Maria	22-JUL-55	1
3	Ionescu Andrei	05-NOV-56	1
4	Georgescu Ana	18-SEP-58	1
5	Stanescu Mihaila	27-JUN-52	1
6	Dumitru Dan	10-FEB-51	2
7	Gheorghici Elena	25-APR-57	2
8	Mihai Andrei	12-APR-54	2
9	Florescu Alina	03-JAN-59	2
10	Stefanescu Ioan	08-DEC-51	2
11	Radulescu Dumitru	10-JUN-50	2
12	Popa George	17-OCT-56	3
13	Gavrilii Alexandru	02-SEPT-55	3
14	Martinescu Adrian	14-JUL-51	3
15	Baciu Gabriela	20-MAR-58	3
16	Constantinescu Ana	10-NOV-53	3
17	Dobre Radu	04-JUL-52	3
18	Andrei Mihaila	19-APR-57	3
19	Iacob Marius	12-SEP-50	4
20	Voiu Ana	25-MAY-56	4
21	Cristescu Andrei	03-DEC-52	4
22	Munteanu Elena	15-PEST-59	4
23	Alexandrescu Ana	07-NOV-54	4
24	Dumitrescu Stefan	05-APR-57	4
25	Dianu Mihaila	22-MAR-56	4
26	Ionescu Florin	09-JUN-51	4
27	Andrei Catalin	28-JUL-50	5
28	Stan Maria	14-APR-57	5
29	Gheorghici Mihail	05-APR-52	5
30	Dumitrescu Ileana	23-OCT-59	5
31	Stefan Maria	11-MAR-54	5
32	Popa Ioana	02-DEC-53	5
33	Rada Alexandra	16-JUL-58	5
34	Marin Florin	03-MAY-51	5
35	Dobre Andrei	27-FEB-56	5

```

INSERT INTO FOLOSESTE (ID_ANTRENAMENT, ID_CLASA) VALUES (15, 3)
INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (5, 1)
INSERT INTO FOLOSESTE (id_antrenament, id_clasa) VALUES (3, 4)

SELECT * FROM FOLOSESTE;

--MEMBRU
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Popescu Ion', TO_DATE('1950-05-15', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Rade Maria', TO_DATE('1950-07-21', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Ionescu Andrei', TO_DATE('1955-1-15', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Georgescu Ana', TO_DATE('1956-09-15', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Stanescu Mihaila', TO_DATE('1952-06-27', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dumitru Dan', TO_DATE('1951-12-15', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Mihai Andrei', TO_DATE('1954-05-12', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Florescu Alina', TO_DATE('1959-01-03', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Stefanescu Ioan', TO_DATE('1953-12-08', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Cristescu Ionut', TO_DATE('1950-05-20', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Munteanu Elena', TO_DATE('1955-06-01', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Gavrilii Alexandru', TO_DATE('1958-09-02', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Marinescu Adrian', TO_DATE('1991-06-14', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Baciu Gabriela', TO_DATE('1958-03-25', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Constantinescu Ana', TO_DATE('1955-03-01', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dobres Andrei', TO_DATE('1952-07-04', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Andrei Mihaila', TO_DATE('1957-04-19', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Iacob Maria', TO_DATE('1950-09-12', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Voiu Ana', TO_DATE('1955-05-25', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dumitrescu Stefan', TO_DATE('1955-01-15', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Mihail Andrei', TO_DATE('1954-11-07', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Alexandrescu Ana', TO_DATE('1959-03-01', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Popescu Ionut', TO_DATE('1953-07-01', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dianu Mihaila', TO_DATE('1955-03-01', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Ionescu Florin', TO_DATE('1955-06-09', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Andrei Catalin', TO_DATE('1950-05-20', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Stan Maria', TO_DATE('1957-08-14', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Gheorghici Mihail', TO_DATE('1952-04-05', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dobres Andrei', TO_DATE('1955-03-12', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Rade Maria', TO_DATE('1955-03-11', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Popa Ioana', TO_DATE('1955-12-02', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Rada Alexandra', TO_DATE('1958-03-15', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Marin Florin', TO_DATE('1951-03-03', 'YYYY-MM-DD'), 1)
INSERT INTO MEMBRU (name, DATA_NASTERII, ID_NASTERII) VALUES ('Dobre Andrei', TO_DATE('1954-02-27', 'YYYY-MM-DD'), 1)

```

ABONAMENT :

--SALA1

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 100.00, 1);

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 150.00, 2);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 200.00, 3);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 250.00, 4);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 300.00, 5);
```

--SALA2

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 120.00, 6);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 170.00, 7);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 220.00, 8);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 270.00, 9);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 320.00, 10);
```

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 320.00, 11);
```

--SALA3

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 110.00, 12);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 160.00, 13);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 210.00, 14);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 260.00, 15);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 310.00, 16);
```

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 310.00, 17);
```

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 110.00, 18);
```

--SALA4

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 180.00, 19);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 230.00, 20);
```

```
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 280.00, 21);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330.00, 22);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330.00, 23);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330.00, 24);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330.00, 25);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 140.00, 26);

--SALA5

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 190.00, 27);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 240.00, 28);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 290.00, 29);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 290.00, 30);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 290.00, 31);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 340.00, 32);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 130.00, 33);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 340.00, 34);

INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 130.00, 35);
```

SELECT * FROM ABONAMENT;

ID_ABONAMENT	TIP_ABONAMENT	PRET	ID_MEMBRU
1	Abonament Bronze	100	1
2	Abonament Silver	150	2
3	Abonament Gold	200	3
4	Abonament Platinum	250	4
5	Abonament Diamond	300	5
6	Abonament Bronze	120	6
7	Abonament Silver	170	7
8	Abonament Gold	220	8
9	Abonament Platinum	270	9
10	Abonament Diamond	320	10
11	Abonament Bronze	120	11
12	Abonament Bronze	110	12
13	Abonament Silver	160	13
14	Abonament Gold	210	14
15	Abonament Platinum	260	15
16	Abonament Diamond	310	16
17	Abonament Bronze	110	17
18	Abonament Bronze	110	18
19	Abonament Silver	180	19
20	Abonament Gold	230	20
21	Abonament Platinum	280	21
22	Abonament Diamond	330	22
23	Abonament Diamond	330	23
24	Abonament Diamond	330	24
25	Abonament Diamond	330	25
26	Abonament Bronze	140	26
27	Abonament Silver	190	27
28	Abonament Gold	240	28
29	Abonament Platinum	290	29
30	Abonament Platinum	290	30
31	Abonament Platinum	290	31
32	Abonament Diamond	340	32
33	Abonament Bronze	130	33
34	Abonament Diamond	340	34
35	Abonament Bronze	120	35

```

--ABONAMENT
--SALA1
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 100,00, 1);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 150,00, 2);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 200,00, 3);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 250,00, 4);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 300,00, 5);

--SALA2
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 120,00, 6);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 170,00, 7);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 220,00, 8);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 270,00, 9);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 320,00, 10);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 320,00, 11);

--SALA3
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 110,00, 12);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 160,00, 13);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 210,00, 14);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 260,00, 15);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 310,00, 16);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 310,00, 17);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 110,00, 18);

--SALA4
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 180,00, 19);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 230,00, 20);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 280,00, 21);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330,00, 22);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330,00, 23);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330,00, 24);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 330,00, 25);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 140,00, 26);

--SALA5
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Silver', 190,00, 27);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Gold', 240,00, 28);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 290,00, 29);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 340,00, 30);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Platinum', 290,00, 31);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 340,00, 32);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 130,00, 33);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Diamond', 340,00, 34);
INSERT INTO ABONAMENT (tip_abonament, pret, id_membru) VALUES ('Abonament Bronze', 120,00, 35);

SELECT * FROM ABONAMENT;

```

Opened nodes (100); Saved files(1)

FORMULAR :

--SALA1

INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 4, 1);

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 4, 2);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 4, 3);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 4, 4);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 4, 5);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (2, 6, 3);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (2, 6, 4);
```

```
--SALA2
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 11, 6);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 11, 7);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 11, 8);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 11, 9);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 11, 10);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 11, 11);
```

```
--SALA3
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 18, 12);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 18, 13);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 18, 14);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 18, 15);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 17, 16);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 17, 17);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 19, 18);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 19, 12);
```

```
--SALA4
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 23, 19);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 23, 20);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 23, 21);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (4, 23, 22);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (5, 24, 23);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (5, 24, 24);
```

```
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (5, 24, 25);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (5, 24, 19);
--SALA5
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 28, 27);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (3, 28, 28);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (2, 29, 29);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (2, 29, 30);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 30, 31);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (1, 30, 32);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (5, 31, 33);
INSERT INTO FORMULAR (id_clasa, id_antrenor, id_membru) VALUES (5, 31, 34);
```

SELECT * FROM FORMULAR;

The screenshot shows the Oracle SQL Developer interface. The title bar reads "Oracle SQL Developer C:\Users\Alex\Desktop\Sala.sql". The main window displays a query result from a table named "FORMULARIS". The table has three columns: "ID_CLASA", "ID_ANTRIENOR", and "ID_MEMBRI". The data consists of 37 rows, each containing a unique combination of values for the three columns. The results are displayed in a grid format with horizontal and vertical scroll bars.

ID_CLASA	ID_ANTRIENOR	ID_MEMBRI
1	1	4
2	1	2
3	1	4
4	1	4
5	1	4
6	2	6
7	2	6
8	3	11
9	3	11
10	3	11
11	3	11
12	3	11
13	3	11
14	4	18
15	4	18
16	4	18
17	4	18
18	3	17
19	3	17
20	3	19
21	3	19
22	4	23
23	4	23
24	4	23
25	4	23
26	5	24
27	5	24
28	5	24
29	5	24
30	3	28
31	3	28
32	2	29
33	2	29
34	1	30
35	1	30
36	5	31
37	5	31

6. Se dorește aflarea unor informații legate de săli. Prin intermediul unei proceduri stocate care primește ca parametru id-ul sălii despre care dorim să aflăm detalii, ni se furnizează următoarele informații :

-numele sălii respective

- lista angajaților sălii respective (s-a utilizat un tablou indexat unde index-ul este id-ul angajatului, iar conținutul este numele acestuia)
- lista membrilor sălii respective (id-urile lor) (s-a utilizat un tablou imbricat)
- lista locațiilor sălii respective (numele orașelor) (s-a utilizat un tablou de tip vector)

```
CREATE OR REPLACE PROCEDURE afisare_informatii_sala
```

```
    (v_id_sala sala.id_sala%TYPE DEFAULT 1)
```

```
IS
```

```
    TYPE tablou_indexat IS TABLE OF VARCHAR(50) INDEX BY PLS_INTEGER;
```

```
    TYPE tablou_imbricat IS TABLE OF NUMBER;
```

```
    TYPE tablou_vector IS VARRAY(20) OF VARCHAR(50);
```

```
    v_lista_angajati tablou_indexat := tablou_indexat();
```

```
    v_lista_membrii tablou_imbricat := tablou_imbricat();
```

```
    v_lista_locatii tablou_vector := tablou_vector();
```

```
    v_nume_sala VARCHAR(50);
```

```
BEGIN
```

```
    SELECT nume_sala
```

```
        INTO v_nume_sala
```

```
        FROM sala
```

```
        WHERE id_sala = v_id_sala;
```

```
    FOR angajat IN (SELECT id_angajat, nume_angajat FROM angajat WHERE id_sala = v_id_sala)
```

```
    LOOP
```

```
        v_lista_angajati(angajat.id_angajat) := angajat.nume_angajat;
```

```
    END LOOP;
```

```
FOR membru IN (SELECT id_membru FROM membru WHERE id_sala = v_id_sala)
LOOP
    v_lista_membrii.extend;
    v_lista_membrii(v_lista_membrii.last) := membru.id_membru;
END LOOP;
```

```
FOR locatie IN (SELECT oras_locatie FROM locatie WHERE id_sala = v_id_sala)
LOOP
    v_lista_locatii.extend;
    v_lista_locatii(v_lista_locatii.last) := locatie.oras_locatie;
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('Informatii despre sala ' || v_nume_sala || ':');
```

```
DBMS_OUTPUT.PUT_LINE('Lista angajati:');
FOR i IN v_lista_angajati.first .. v_lista_angajati.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_angajati(i));
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('Lista membrii (id-urile):');
FOR i IN v_lista_membrii.first .. v_lista_membrii.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_membrii(i));
END LOOP;
```

```
DBMS_OUTPUT.PUT_LINE('Lista locatii (orasele):');
FOR i IN v_lista_locatii.first .. v_lista_locatii.last
```

```
LOOP  
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_locatii(i));  
END LOOP;
```

```
END afisare_informatii_sala;
```

```
/
```

```
DROP PROCEDURE afisare_informatii_sala;
```

```
BEGIN  
    afisare_informatii_sala;  
END;  
/
```

```
BEGIN  
    afisare_informatii_sala(1);  
END;  
/
```

```
BEGIN  
    afisare_informatii_sala(2);  
END;  
/
```

```
BEGIN  
    afisare_informatii_sala(4);  
END;  
/
```

```

CREATE OR REPLACE PROCEDURE afastra.informatii_sala
(
    v_id_sala salas.id_sala%TYPE DEFAULT 1
)
IS
    TYPE tablo_l_indeks IS TABLE OF VARCHAR(50) INDEX BY PLS_INTEGER;
    TYPE tablo_l_imbrat IS TABLE OF NUMBER;
    TYPE tablo_l_vector IS VARAY(20) OF VARCHAR(50);

    v_lista_angajati tablo_l_indeks := tablo_l_indeks();
    v_lista_membrii tablo_l_imbrat := tablo_l_imbrat();
    v_lista_locatii tablo_l_vector := tablo_l_vector();

    v_nume_sala VARCHAR(50);

BEGIN

    SELECT nume_sala
    INTO v_nume_sala
    FROM sala
    WHERE id_sala = v_id_sala;

    FOR angajat IN (SELECT id_angajat, nume_angajat FROM angajat WHERE id_sala = v_id_sala)
    LOOP
        v_lista_angajati(angajat.id_angajat) := angajat.nume_angajat;
    END LOOP;

    FOR membru IN (SELECT id_membru FROM membru WHERE id_sala = v_id_sala)
    LOOP
        v_lista_membrii.extend;
        v_lista_membrii(v_lista_membrii.last) := membru.id_membru;
    END LOOP;

    FOR locatie IN (SELECT oras_locatie FROM locatie WHERE id_sala = v_id_sala)
    LOOP
        v_lista_locatii.extend;
        v_lista_locatii(v_lista_locatii.last) := locatie.oras_locatie;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Informatii despre sala ' || v_nume_sala || ':');

    DBMS_OUTPUT.PUT_LINE('Liste angajati:');
    FOR i IN v_lista_angajati.first .. v_lista_angajati.last
    LOOP
        DBMS_OUTPUT.PUT_LINE(i);
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Liste membri:');
    FOR i IN v_lista_membrii.first .. v_lista_membrii.last
    LOOP
        DBMS_OUTPUT.PUT_LINE(i);
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Liste locatii (orasele):');
    FOR i IN v_lista_locatii.first .. v_lista_locatii.last
    LOOP
        DBMS_OUTPUT.PUT_LINE(i);
    END LOOP;

    END PROCEDURE;
/

```

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySALA Tables (Filtered) Views Procedures Functions Operators Outlines Sequences Materialized Views Triggers Types Seconaries Materialized View Logs Synonyms Public Synonyms Database Links Public Database Links Directories Editors Java XML Schemas XML DB Repository OJAP Options Reports All Reports Analytic View Reports Data Discovery Reports Data Modeler Reports OLAP Reports TimeTen Reports User Defined Reports

Worksheet: Query Builder

```

LOOP
    v_lista_locatii.extend;
    v_lista_locatii(v_lista_locatii.last) := locatii.oraclie;
END LOOP;

DBMS_OUTPUT.PUT_LINE('Informatii despre sala ' || v_numar_sala || ':');

DBMS_OUTPUT.PUT_LINE('Lista angajati:');
FOR i IN v_lista_angajati.first .. v_lista_angajati.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_angajati(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Lista membrui (id-urile):');
FOR i IN v_lista_membrui.first .. v_lista_membrui.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_membrui(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Lista locatii (oraclie):');
FOR i IN v_lista_locatii.first .. v_lista_locatii.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_locatii(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Afisare informatii sala:');

```

Script Output: A [Query Result] Task completed in 0.035 seconds

Procedure AFISARE_INFORMATII_SALA compiled

DBMS Output: Buffer Size[20000]

MySALA *

Click on an identifier with the Control key down to perform "Go to Declaration".

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySALA Tables (Filtered) Views Procedures Functions Operators Outlines Sequences Materialized Views Triggers Types Seconaries Materialized View Logs Synonyms Public Synonyms Database Links Public Database Links Directories Editors Java XML Schemas XML DB Repository OJAP Options Reports All Reports Analytic View Reports Data Discovery Reports Data Modeler Reports OLAP Reports TimeTen Reports User Defined Reports

Worksheet: Query Builder

```

LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_membrui(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Lista locatii (oraclie):');
FOR i IN v_lista_locatii.first .. v_lista_locatii.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_locatii(i));
END LOOP;

END afisare_informatii_sala;
/
DROP PROCEDURE afisare_informatii_sala;
BEGIN
    afisare_informatii_sala;
END;

```

PL/SQL procedure successfully completed.

Script Output: A [Query Result] Task completed in 0.032 seconds

DBMS Output: Buffer Size[20000]

MySALA *

Informatii despre sala World Class:
 Lista angajati:
 Popescu Ion
 Cozecu Maria
 Georgescu Andrei
 Constantinescu Elena
 Dumitru Radu
 Popescu Bogdan
 Gheorghe Alexandru
 Lista membrui (id-urile):
 1
 2
 3
 4
 5
 Lista locatii (oraclie):
 Bucuresti
 New York

Oracle SQL Developer : MySALA

Connections

- MySALA
 - Tables (Filtered)
 - Views
 - Indices
 - Indexes
 - Procedures
 - Functions
 - Operators
 - Outlines
 - Queues Tables
 - Triggers
 - Types
 - Sequences
 - Materialized Views
 - Materialized View Logs
 - Synonyms
 - Public Synonyms
 - Database Links
 - Public Database Links
 - Directories
 - Editors
 - Java
 - XML Schemas
 - VN-DB Repository
 - OB-AP Repository

Reports

- All Reports
- Analytic View Reports
- Data Discovery Reports
- Data Modeler Reports
- OLAP Reports
- Timeline Reports
- User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySALA

Connections

- MySALA
 - Tables (Filtered)
 - Views
 - Indices
 - Indexes
 - Procedures
 - Functions
 - Operators
 - Outlines
 - Queues Tables
 - Triggers
 - Types
 - Sequences
 - Materialized Views
 - Materialized View Logs
 - Synonyms
 - Public Synonyms
 - Database Links
 - Public Database Links
 - Directories
 - Editors
 - Java
 - XML Schemas
 - VN-DB Repository
 - OB-AP Repository

Reports

- All Reports
- Analytic View Reports
- Data Discovery Reports
- Data Modeler Reports
- OLAP Reports
- Timeline Reports
- User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

SQL Worksheet

```

    LOCAL;
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_locatii(i));
  END LOOP;

END afisare_informatii_sala;
/
DROP PROCEDURE afisare_informatii_sala;
/
BEGIN
  afisare_informatii_sala();
END;
/

```

Script Output

PL/SQL procedure successfully completed.

Items Output

Buffer Size [20000]

MySALA

Informații despre sala World Class:

Lista angajați:

- Ropean Ion
- Ionescu Maria
- Georgescu Andrei
- Constantinescu Elena
- Danciu Radu
- Pope Ana
- Gheorghe Alexandru

Lista membrilor (id=urile):

- 1
- 2
- 3
- 4
- 5

Lista locații (orasele):

- București
- New York

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySALA

Connections

- MySALA
 - Tables (Filtered)
 - Views
 - Indices
 - Indexes
 - Procedures
 - Functions
 - Operators
 - Outlines
 - Queues Tables
 - Triggers
 - Types
 - Sequences
 - Materialized Views
 - Materialized View Logs
 - Synonyms
 - Public Synonyms
 - Database Links
 - Public Database Links
 - Directories
 - Editors
 - Java
 - XML Schemas
 - VN-DB Repository
 - OB-AP Repository

Reports

- All Reports
- Analytic View Reports
- Data Discovery Reports
- Data Modeler Reports
- OLAP Reports
- Timeline Reports
- User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySALA

Connections

- MySALA
 - Tables (Filtered)
 - Views
 - Indices
 - Indexes
 - Procedures
 - Functions
 - Operators
 - Outlines
 - Queues Tables
 - Triggers
 - Types
 - Sequences
 - Materialized Views
 - Materialized View Logs
 - Synonyms
 - Public Synonyms
 - Database Links
 - Public Database Links
 - Directories
 - Editors
 - Java
 - XML Schemas
 - VN-DB Repository
 - OB-AP Repository

Reports

- All Reports
- Analytic View Reports
- Data Discovery Reports
- Data Modeler Reports
- OLAP Reports
- Timeline Reports
- User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

SQL Worksheet

```

DROP PROCEDURE afisare_informatii_sala;
/
BEGIN
  afisare_informatii_sala();
END;
/

```

PL/SQL procedure successfully completed.

Items Output

Buffer Size [20000]

MySALA

Informații despre sala Fit Gym:

Lista angajați:

- Marin Ioana
- Apcar Stefan
- Rădulescu Andreea
- Stancu Daniel
- Petreacu Gabriel

Lista membrilor (id=urile):

- 6
- 7
- 8
- 9
- 10
- 11

Lista locații (orasele):

- Timisoara

```

CREATE OR REPLACE PROCEDURE afisare_informatii_sala(id IN NUMBER)
IS
    v_nume_sala sala.id_sala%TYPE DEFAULT 1;
    v_id_vestiar vestiar.id_vestiar%TYPE;
    v_tip CHAR(1);
    v_capacitate INT;
    v_id_ingrijitor INT;
BEGIN
    dbms_output.put_line('Afisare informatii sala');
    dbms_output.put_line('Sala: ' || v_nume_sala);
    dbms_output.put_line('Lista angajati:');
    dbms_output.put_line('Popovici Radu');
    dbms_output.put_line('Avram Mihai');
    dbms_output.put_line('Flores Alin Andrei');
    dbms_output.put_line('Matei Bogdan');
    dbms_output.put_line('Voinea Alexandru');
    dbms_output.put_line('Constantinescu Diana');
    dbms_output.put_line('Lista membrii (id-uri):');
    dbms_output.put_line(v_id_vestiar);
    dbms_output.put_line(v_id_vestiar + 1);
    dbms_output.put_line(v_id_vestiar + 2);
    dbms_output.put_line(v_id_vestiar + 3);
    dbms_output.put_line(v_id_vestiar + 4);
    dbms_output.put_line(v_id_vestiar + 5);
    dbms_output.put_line(v_id_vestiar + 6);
    dbms_output.put_line(v_id_vestiar + 7);
    dbms_output.put_line(v_id_vestiar + 8);
    dbms_output.put_line(v_id_vestiar + 9);
    dbms_output.put_line(v_id_vestiar + 10);
    dbms_output.put_line('Lista locatii (orasele):');
    dbms_output.put_line('Craiova');
    dbms_output.put_line('Roma');
    dbms_output.put_line('Cologne');
END;

```

7. Se dorește aflarea unor informații legate de săli. Prin intermediul unei proceduri stocate care primește ca parametru id-ul sălii despre care dorim să aflăm detalii, ni se furnizează următoarele informații :

- numele sălii respective
- informații despre vestiarele sălii (id, tip și capacitate)
- informații despre îngrijitorii vestiarilor respective (id, nume)

S-a utilizat un cursor clasic (ce selectează informații despre vestiarele din sala respectivă) și un ciclu cursor (ce selectează informații despre îngrijitorii vestiarului respectiv) cu parametru (parametrul fiind id-ul vestiarului respectiv) ce este dependent de cursorul clasic.

`CREATE OR REPLACE PROCEDURE afisare_informatii_vestiare`

`(v_id_sala sala.id_sala%TYPE DEFAULT 1)`

`IS`

`v_nume_sala sala.nume_sala%TYPE;`

`v_id_vestiar vestiar.id_vestiar%TYPE;`

`v_tip CHAR(1);`

`v_capacitate INT;`

`v_id_ingrijitor INT;`

```

v_nume_angajat VARCHAR(50);

CURSOR c_vestiar IS
SELECT id_vestiar, tip_vestiar, capacitate
FROM vestiar
WHERE id_sala=v_id_sala;

CURSOR c_ingrijitor (vestiar_id vestiar.id_vestiar%TYPE) IS
SELECT id_ingrijitor, nume_angajat
FROM ingrijeste i, angajat a
WHERE i.id_vestiar = vestiar_id and a.id_angajat=i.id_ingrijitor;

BEGIN

SELECT nume_sala
INTO v_nume_sala
FROM sala
WHERE id_sala = v_id_sala;

DBMS_OUTPUT.PUT_LINE('-----');
DBMS_OUTPUT.PUT_LINE('Informatii despre vestiarele salii ' || v_nume_sala || ':');

OPEN c_vestiar;

LOOP
FETCH c_vestiar INTO v_id_vestiar, v_tip, v_capacitate;
EXIT WHEN c_vestiar%NOTFOUND;

DBMS_OUTPUT.PUT_LINE('-----');

```

```

DBMS_OUTPUT.PUT_LINE('Vestiarul ' || v_id_vestiar || ' de tipul ' || v_tip || ' si cu capacitatea ' || v_capacitate || '.');
DBMS_OUTPUT.PUT_LINE('Lista ingrijitorilor: ');

FOR ingrijitor_i IN c_ingrijitor(v_id_vestiar) LOOP
    DBMS_OUTPUT.PUT_LINE(ingrijitor_i.id_ingrijitor || '.' || ingrijitor_i.nume_angajat);
END LOOP;

END LOOP;

DBMS_OUTPUT.PUT_LINE('-----');
CLOSE c_vestiar;

END afisare_informatii_vestiare;
/

```

DROP PROCEDURE afisare_informatii_vestiare;

```

BEGIN
    afisare_informatii_vestiare;
END;
/

```

```

BEGIN
    afisare_informatii_vestiare(1);
END;
/

```

```
BEGIN  
afisare_informatii_vestiare(2);  
END;  
/
```

```
BEGIN  
afisare_informatii_vestiare(4);  
END;  
/
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Workshop Query Builder

```

CREATE OR REPLACE PROCEDURE afisare_informatii_vestiare
(v_id_vesta sal.v_id_vesta%TYPE DEFAULT 1)
IS
  v_name_vesta sal.v_name_vesta%TYPE;
  v_id_vestiar vestiar.v_id_vestiar%TYPE;
  v_tip CHAR(1);
  v_capacitate INT;
  v_id_angajator INT;
  v_nume_angajat VARCHAR(50);

  CURSOR c_vestiar IS
    SELECT id_vestiar, tip_vestiar, capacitate
    FROM vestiar
    WHERE id_vesta = id_vesta;

  CURSOR c_angajitor (vestiar_id_vestiar%TYPE) IS
    SELECT id_angajitor, nume_angajat
    FROM angajator a, angajat
    WHERE a.id_vestiar = vestiar_id_vestiar AND a.id_angajat = i.id_angajitor;

  BEGIN
    SELECT nume_vesta
    INTO v_name_vesta
    FROM sal
    WHERE id_vesta = v_id_vesta;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE('Informatii despre vestiarele salii ' || v_name_vesta || '-----');

    OPEN c_vestiar;

    LOOP
      FETCH c_vestiar INTO v_id_vestiar, v_tip, v_capacitate;
      EXIT WHEN c_vestiar%NOTFOUND;

      DBMS_OUTPUT.PUT_LINE('Vestiarul ' || v_id_vestiar || ' de tipul ' || v_tip || ' si cu capacitate ' || v_capacitate || ',');
      DBMS_OUTPUT.PUT_LINE('Liste angajitorilor: ');

      FOR angajitor_i IN c_angajitor(v_id_vestiar) LOOP
        DBMS_OUTPUT.PUT_LINE(angajitor_i.id_angajitor || ', ' || angajitor_i.nume_angajat);
      END LOOP;
    END LOOP;
  END;

```

Script Output: Task completed in 0.025 seconds

Dmls Output Compiler - Log Messages Logging Page Summaries Console

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Workshop Query Builder

```

CREATE OR REPLACE PROCEDURE afisare_informatii_vestiare
(v_id_vesta sal.v_id_vesta%TYPE DEFAULT 1)
IS
  v_name_vesta sal.v_name_vesta%TYPE;
  v_capacitate INT;
  v_id_angajator INT;
  v_nume_angajat VARCHAR(50);

  CURSOR c_vestiar IS
    SELECT id_vestiar, tip_vestiar, capacitate
    FROM vestiar
    WHERE id_vesta = id_vesta;

  CURSOR c_angajitor (vestiar_id_vestiar%TYPE) IS
    SELECT id_angajitor, nume_angajat
    FROM angajator a, angajat
    WHERE a.id_vestiar = vestiar_id_vestiar AND a.id_angajat = i.id_angajitor;

  BEGIN
    SELECT nume_vesta
    INTO v_name_vesta
    FROM sal
    WHERE id_vesta = v_id_vesta;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE('Informatii despre vestiarele salii ' || v_name_vesta || '-----');

    OPEN c_vestiar;

    LOOP
      FETCH c_vestiar INTO v_id_vestiar, v_tip, v_capacitate;
      EXIT WHEN c_vestiar%NOTFOUND;

      DBMS_OUTPUT.PUT_LINE('Vestiarul ' || v_id_vestiar || ' de tipul ' || v_tip || ' si cu capacitate ' || v_capacitate || ',');
      DBMS_OUTPUT.PUT_LINE('Liste angajitorilor: ');

      FOR angajitor_i IN c_angajitor(v_id_vestiar) LOOP
        DBMS_OUTPUT.PUT_LINE(angajitor_i.id_angajitor || ', ' || angajitor_i.nume_angajat);
      END LOOP;
    END LOOP;
  END afisare_informatii_vestiare;

```

Script Output: Task completed in 0.025 seconds

Dmls Output Compiler - Log Messages Logging Page Summaries Console

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySALA

Workshop Query Builder

```

OPEN c_vestiar;
LOOP
  FETCH c_vestiar INTO v_id_vestiar, v_tip, v_capacitate;
  EXIT WHEN c_vestiar%NOTFOUND;
  DBMS_OUTPUT.PUT_LINE('Vestiarul ' || v_id_vestiar || ' de tipul ' || v_tip || ' si cu capacitatea ' || v_capacitate || '.');
  DBMS_OUTPUT.PUT_LINE('Lista ingrijitorilor:');
  FOR ingrijitor_i IN c_ingrijitor(v_id_vestiar) LOOP
    DBMS_OUTPUT.PUT_LINE(ingrijitor_i.id_ingrijitor || ' ' || ingrijitor_i.name_angajat);
  END LOOP;
  DBMS_OUTPUT.PUT_LINE('-----');
  CLOSE c_vestiar;
END afisare_informatii_vestiare;

```

Script Output: Task completed in 0.028 seconds

Procedure AFISARE_INFORMATII_VESTIARE compiled

DBMS Output Buffer Size:20000 MySALA

Compiler - Log

Messages Logging Page Statements Compiler

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySALA

Workshop Query Builder

```

FOR ingrijitor_i IN c_afisare_informatii_vestiare LOOP
  DBMS_OUTPUT.PUT_LINE(ingrijitor_i.id_ingrijitor || ' ' || ingrijitor_i.name_angajat);
END LOOP;
END LOOP;
DBMS_OUTPUT.PUT_LINE('-----');
CLOSE c_vestiar;
END afisare_informatii_vestiare;
/

```

DESCRIBE PROCEDURE afisare_informatii_vestiare;

BEGIN
 afisare_informatii_vestiare();
END;
/

BEGIN
 afisare_informatii_vestiare();
END;
/

PL/SQL procedure successfully completed.

DBMS Output Buffer Size:20000 MySALA

Informatii despre vestiarele salii World Class:
Vestiarul 1 de tipul M si cu capacitatea 50.
Lista ingrijitorilor:
1. Popescu Ion
2. Georgescu Andrei
Vestiarul 2 de tipul F si cu capacitatea 40.
Lista ingrijitorilor:
2. Ionescu Maria

Compiler - Log

Messages Logging Page Statements Compiler

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySALA

Reports All Reports Analytic View Reports Database Model Reports OLAP Reports Time/Fan Reports User Defined Reports

Script Output MySQLA Buffer Size[20000]

Compiler - Log

Worksheet Query Builder

```

DBMS_OUTPUT.PUT_LINE(ingritor_i.id_ngritor || ', ' || ingritor_i.nume_angajat);
END LOOP;
END LOOP;

DBMS_OUTPUT.PUT_LINE('-----');
CLOSE C_VESTIARE;

END afisare_informatii_vestiare;
/
DROP PROCEDURE afisare_informatii_vestiare;
BEGIN
afisare_informatii_vestiare();
END;
/

```

PL/SQL procedure successfully completed.

Domn. Output MySQLA

Informații despre vestiarele salii World Class:

Vestiarul 1 de tipul M și cu capacitatea 50.
Lista îngrijitorilor:
1. Popescu Ion
3. Georgescu Andrei
Vestiarul 2 de tipul F și cu capacitatea 60.
Lista îngrijitorilor:
2. Ionescu Maria

Messages Logging Page Statements Compiler

Line 195 Column 2 Insert Modify Windows

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySALA

Reports All Reports Analytic View Reports Database Model Reports OLAP Reports Time/Fan Reports User Defined Reports

Script Output MySQLA Buffer Size[20000]

Compiler - Log

Worksheet Query Builder

```

/
DROP PROCEDURE afisare_informatii_vestiare;
BEGIN
afisare_informatii_vestiare();
END;
/
BEGIN
afisare_informatii_vestiare();
END;
/
BEGIN
afisare_informatii_vestiare(2);
END;
/
BEGIN
afisare_informatii_vestiare(4);
END;
/

```

PL/SQL procedure successfully completed.

Domn. Output MySQLA

Informații despre vestiarele salii Fit Gym:

Vestiarul 3 de tipul M și cu capacitatea 40.
Lista îngrijitorilor:
9. Apostol Stefan
Vestiarul 4 de tipul F și cu capacitatea 30.
Lista îngrijitorilor:
8. Mihai Ioana

Messages Logging Page Statements Compiler

Line 193 Column 2 Insert Modify Windows

```

DECLARE PROCEDURE afisare_informatii_vestiare;
BEGIN
    afisare_informatii_vestiare();
END;
/
BEGIN
    afisare_informatii_vestiare(1);
END;
/
BEGIN
    afisare_informatii_vestiare(2);
END;
/
BEGIN
    afisare_informatii_vestiare(3);
END;
/

```

Script Output: Task completed in 0.032 seconds

PL/SQL procedure successfully completed.

Dbms Output: MySALA

```

Informatii despre vestierile salii Gym Plus:
-----
Vestiarul 9 de tipul M si cu capacitatea 40.
Lista ingrijitorilor:
20. Popescu Radu
-----
Vestiarul 10 de tipul F si cu capacitatea 35.
Lista ingrijitorilor:
21. Avram Mihai
-----
Vestiarul 11 de tipul M si cu capacitatea 55.
Lista ingrijitorilor:
22. Flores Alexandru
-----
```

Compiler Log: Messages Logging Page Statements Compiler

8. Se dorește aflarea totalului încasat de sală din abonamentele membrilor sălii respective. S-a folosit o funcție stocată care returnează acest total. S-au utilizat 3 tabele într-o singură comandă SQL (SALA, MEMBRU, ABONAMENT) și s-au inclus 2 excepții proprii (exceptie1 și exceptie2).

```

CREATE OR REPLACE FUNCTION suma_bani
(v_id_sala sala.id_sala%TYPE DEFAULT 1)

RETURN abonament.pret%TYPE IS

v_suma abonament.pret%TYPE;

exception1 EXCEPTION;
exception2 EXCEPTION;

BEGIN

BEGIN

SELECT SUM(a.pret)

INTO v_suma

FROM abonament a

JOIN membru m ON m.id_membru = a.id_membru

```

```
JOIN sala s ON s.id_sala = m.id_sala  
WHERE s.id_sala = v_id_sala;
```

```
IF v_suma <= 1000 THEN  
    RAISE exceptie1;  
END IF;
```

```
IF v_suma <= 2000 THEN  
    RAISE exceptie2;  
END IF;
```

```
EXCEPTION  
    WHEN exceptie1 THEN  
        DBMS_OUTPUT.PUT_LINE('Sala este in faliment!');  
    WHEN exceptie2 THEN  
        DBMS_OUTPUT.PUT_LINE('Sala este aproape de faliment!');  
    END;
```

```
RETURN v_suma;
```

```
EXCEPTION  
    WHEN OTHERS THEN  
        DBMS_OUTPUT.PUT_LINE('Alta eroare!');
```

```
END suma_bani;  
/
```

```
DROP FUNCTION suma_bani;
```

```
BEGIN  
  DBMS_OUTPUT.PUT_LINE(suma_bani);  
END;  
/
```

```
BEGIN  
  DBMS_OUTPUT.PUT_LINE(suma_bani(1));  
END;  
/
```

```
BEGIN  
  DBMS_OUTPUT.PUT_LINE(suma_bani(2));  
END;  
/
```

```
BEGIN  
  DBMS_OUTPUT.PUT_LINE(suma_bani(5));  
END;  
/
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE FUNCTION suma_banl
(V_id_sala salas.id_sala%TYPE DEFAULT 1)
RETURN Abramant.suma%TYPE;
BEGIN
    v_suma := Abramant.suma;
    exception1 EXCEPTION;
    exception2 EXCEPTION;
    BEGIN
        BEGIN
            SELECT sum(a.pret)
            INTO v_suma
            FROM Abramant a
            JOIN membru m ON m.id_membru = a.id_membru
            JOIN sala s ON s.id_sala = m.id_sala
            WHERE s.id_sala = V_id_sala;

            IF v_suma <= 1000 THEN
                RAISE exception1;
            END IF;

            IF v_suma <= 2000 THEN
                RAISE exception2;
            END IF;
        EXCEPTION
            WHEN exception1 THEN
                DBMS_OUTPUT.PUT_LINE('Sala este in faliment!');
            WHEN exception2 THEN
                DBMS_OUTPUT.PUT_LINE('Sala este aproape de faliment!');
        END;
    RETURN v_suma;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Alte eroare!');
    END suma_banl;

```

Script Output X | Query Result X | Task completed in 0.022 seconds

Dime Output

Click on an identifier with the Control key down to perform 'Go to Declaration'

Line 234 (Column 2) | Insert | Modified | Windows |

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet - Query Builder

```

IF v_suma <= 2000 THEN
    RAISE exception1;
END IF;

```

EXCEPTION
 WHEN exception1 THEN
 DBMS_OUTPUT.PUT_LINE('Sala este in faliment!');
 WHEN exception2 THEN
 DBMS_OUTPUT.PUT_LINE('Sala este aproape de faliment!');
 END;

RETURN v_suma;

EXCEPTION
 WHEN OTHERS THEN
 DBMS_OUTPUT.PUT_LINE('Alte eroare!');
 END suma_banl;

Function SUMA_BANI compiled

Dime Output

MySQLA

Click on an identifier with the Control key down to perform 'Go to Declaration'

Line 234 (Column 2) | Insert | Modified | Windows |

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```
END;
    RETURN v_suma;
EXCEPTION
    WHEN OTHERS THEN
        DBMS_OUTPUT.PUT_LINE('Alta errónea');
END suma_bani;
/
DROP FUNCTION suma_bani;

BEGIN
    DBMS_OUTPUT.PUT_LINE(suma_bani);
END;
```

Script Output x | Query Result x | Task completed in 0.031 seconds

Function SUMA_BANI compiled

PL/SQL procedure successfully completed.

Items Output

MySQLA x

Sala este in faliment!

1000

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```
/
```

DROP FUNCTION suma_bani;

BEGIN
 DBMS_OUTPUT.PUT_LINE(suma_bani);
END;

BEGIN
 DBMS_OUTPUT.PUT_LINE(suma_bani(1));
END;

BEGIN
 DBMS_OUTPUT.PUT_LINE(suma_bani(2));
END;

Function SUMA_BANI compiled

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Items Output

MySQLA x

Sala este in faliment!

1000

Sala este in faliment!

1000

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet Query Builder

```

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani');
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani(i)');
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani(2)');
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani(5)');
END;
/

```

Script Output x | Query Result x | Task completed in 0.03 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Items Output

MySQLA x

```

Sala este in faliment!
1990
Sala este in faliment!
1990
Sala este aproape de faliment!
1420

```

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet Query Builder

```

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani');
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani(i)');
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani(2)');
END;
/

BEGIN
  DBMS_OUTPUT.PUT_LINE('suma_bani(5)');
END;
/

```

Script Output x | Query Result x | Task completed in 0.021 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Items Output

MySQLA x

```

Sala este in faliment!
1990
Sala este in faliment!
1990
Sala este aproape de faliment!
1420
2240

```

Click on an identifier with the Control key down to perform "Go to Declaration"

9. Se dorește aflarea antrenorului anului pentru o sala respectivă. Pentru acest lucru, managerul s-a gândit la o formulă pentru a determina acest lucru : antrenorul anului va fi acel antrenor care are numărul de programe create și utilizate de clase egal cu numărul maxim de programe create, utilizate de clase, dintre toți antrenorii sălilor. Prin intermediul unei proceduri stocate care primește ca parametru id-

ul sălii, aflăm antrenorul anului. Dacă nu există niciun antrenor eligibil, vom avea o eroare. Dacă există mai mulți antrenori eligibili, vom avea o eroare, deoarece managerul a spus că vrea ca el să decidă antrenorul anului, în cazul acesta. Pentru orice altă eroare ce poate apărea, avem o altă eroare. S-au utilizat 5 tabele într-o singură comandă SQL (SALA, ANGAJAT, ANTRENOR, PROGRAME, FOLOSESTE) și s-au inclus excepțiile NO_DATA_FOUND și TOO_MANY_ROWS.

```
CREATE OR REPLACE PROCEDURE afisare_antrenorul_anului
  (v_id_sala sala.id_sala%TYPE DEFAULT 1)
IS
  v_nume_angajat angajat.nume_angajat%type;
BEGIN
  SELECT a1.nume_angajat
    INTO v_nume_angajat
   FROM angajat a1
  JOIN antrenor a2 ON a1.id_angajat = a2.id_angajat
  JOIN programe p ON p.id_angajat = a1.id_angajat
  JOIN foloseste f ON f.id_antrenament = p.id_antrenament
  JOIN sala s ON s.id_sala = a1.id_sala
 WHERE s.id_sala = v_id_sala
 GROUP BY a1.nume_angajat
 HAVING COUNT(*) = (
   SELECT MAX(cnt)
     FROM (
       SELECT COUNT(*) as cnt
         FROM angajat a3
        JOIN antrenor a4 ON a3.id_angajat = a4.id_angajat
        JOIN programe p2 ON p2.id_angajat = a3.id_angajat
        JOIN foloseste f2 ON f2.id_antrenament = p2.id_antrenament
        JOIN sala s2 ON s2.id_sala = a3.id_sala
```

```

        GROUP BY s2.nume_sala, a3.nume_angajat
    )
);

DBMS_OUTPUT.PUT_LINE('Antrenorul anului este ' || v_nume_angajat || '.');

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in sala cu id-ul dat.');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi antrenori eligibili pentru castigarea titlului. Va rugam frumos contactati managerul salii respective.');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');

END afisare_antrenorul_anului;
/

```

DROP PROCEDURE afisare_antrenorul_anului;

```

BEGIN
    afisare_antrenorul_anului;
END;
/

```

```

BEGIN
    afisare_antrenorul_anului(1);
END;
/

```

```
BEGIN  
afisare_antrenorul_anului(2);  
END;  
/
```

```
BEGIN  
afisare_antrenorul_anului(5);  
END;  
/
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE PROCEDURE afisare_antrenorul_anului
  (v_id_seala salas.id_salatTYPE DEFAULT 1)
IS
  v_name_angajat angajat.name_angajatTYPE;
BEGIN

  SELECT al.name_angajat
  INTO v_name_angajat
  FROM angajat
  JOIN antrenor a2 ON a1.id_angajat = a2.id_angajat
  JOIN program p ON p.id_angajat = a1.id_angajat
  JOIN foloseste f ON f.id_entrenament = p.id_entrenament
  JOIN seala s ON s.id_seala = a1.id_seala
  WHERE s.id_seala = v_id_seala
  GROUP BY al.name_angajat
  HAVING COUNT(*) = (
    SELECT MAX(cnt)
    FROM (
      SELECT COUNT(*) AS cnt
      FROM angajat a3
      JOIN antrenor a4 ON a1.id_angajat = a4.id_angajat
      JOIN program p4 ON p4.id_angajat = a4.id_angajat
      JOIN foloseste f4 ON f4.id_entrenament = p4.id_entrenament
      JOIN seala s4 ON s4.id_seala = a4.id_seala
      GROUP BY s4.name_seala, a3.name_angajat
    )
  );

  DBMS_OUTPUT.PUT_LINE('Antrenorul anului este ' || v_name_angajat || '.');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in seala cu id-ul dat.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Vă rugăm să contactați managerul salii respective.');
  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Aleră eroare!');
END afisare_antrenorul_anului;
/

```

Drop PROCEDURE afisare_antrenorul_anului;

Begin afisare_antrenorul_anului;

End;

Begin

Click on an identifier with the Control key down to perform 'Go to Declaration'

Line 16 Column 20 | Insert | Modified | Windows |

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet - Query Builder

```

SELECT MAX(cnt)
  FROM (
    SELECT COUNT(*) AS cnt
    FROM angajat a3
    JOIN antrenor a4 ON a1.id_angajat = a4.id_angajat
    JOIN program p4 ON p4.id_angajat = a4.id_angajat
    JOIN foloseste f4 ON f4.id_entrenament = p4.id_entrenament
    JOIN seala s4 ON s4.id_seala = a4.id_seala
    GROUP BY s4.name_seala, a3.name_angajat
  )
;

DBMS_OUTPUT.PUT_LINE('Antrenorul anului este ' || v_name_angajat || '.');

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu există antrenori eligibili pentru castigarea titlului în seala cu id-ul dat.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Vă rugăm să contactați managerul salii respective.');
  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Aleră eroare!');
END afisare_antrenorul_anului;
/

```

Script Output A | Task completed in 0.041 seconds

Procedure AFISARE_ANTRENORUL_ANULUI compiled

Done Output | Buffer Size [20000]

MySQLA

Click on an identifier with the Control key down to perform 'Go to Declaration'

Line 16 Column 20 | Insert | Modified | Windows |

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

DBMS_OUTPUT.PUT_LINE('Antrenorul anului este ' || v_name_angajat || '.');
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in sezon cu id-ul dat.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Vă rugăm să contactați managerul salii respective.');
  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Alte eroare!');

END afisare_antrenorul_anului;
/
DROP PROCEDURE afisare_antrenorul_anului;

BEGIN
  afisare_antrenorul_anului();
END;

/

```

Script Output A Task completed in 0.019 seconds

PL/SQL procedure successfully completed.

Done Output Buffer Size [20000]

MySQLA Antrenorul anului este Dumitru Radu.

Click on an identifier with the Control key down to perform "Go to Declaration".

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

DBMS_OUTPUT.PUT_LINE('Antrenorul anului este ' || v_name_angajat || '.');
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu există antrenori eligibili pentru castigarea titlului în sezon cu id-ul dat.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există prea multe antrenori eligibili pentru castigarea titlului. Vă rugăm să contactați managerul salii respective.');
  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Alte eroare!');

END afisare_antrenorul_anului;
/
DROP PROCEDURE afisare_antrenorul_anului;

BEGIN
  afisare_antrenorul_anului();
END;

/

```

Script Output A Task completed in 0.027 seconds

PL/SQL procedure successfully completed.

Done Output Buffer Size [20000]

MySQLA Antrenorul anului este Dumitru Radu.

Antrenorul anului este Dumitru Radu.

Click on an identifier with the Control key down to perform "Go to Declaration".

The screenshot shows the Oracle SQL Developer interface. The top menu bar includes File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help. The left sidebar displays the schema structure under 'Connections' (MySALA), including tables, views, procedures, and triggers. The central area is a 'Worksheet' titled 'Query Editor' containing PL/SQL code:

```
WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Va ruga frumos contactați managerul săili respectiv.');
WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Aラー eroare!');
END IF;
END afiseaza_antrenorul_anului;
/

DECLARE PROCEDURE afiseaza_antrenorul_anului;
BEGIN
    afiseaza_antrenorul_anului;
END;
/
BEGIN
    afiseaza_antrenorul_anului(1);
END;
/
BEGIN
    afiseaza_antrenorul_anului(2);
END;
/
BEGIN
    afiseaza_antrenorul_anului(5);
END;
/

```

The 'Script Output' tab at the bottom shows the execution results:

```
Error starting at line : 111 in command -
BEGIN
afiseaza_antrenorul_anului(2);
END;
Error report -
ORA-20000: Nu există antrenori eligibili pentru castigarea titlului în sezon cu id-ul dat.
ORA-06512: at "SYSTEM.AFISEAZA_ANTRENORUL_ANULUI", line 34
ORA-06512: at line 2
20000. 00000
*Cause:   The stored procedure 'raise_application_error'
          was called which causes this error to be generated.
*Action:  Correct the problem as described in the error message or contact
          the application administrator or DBA for more information.
```

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays the 'Connections' tree, which includes the 'MySALA' connection. The 'Reports' section is also visible. The main workspace contains a 'Worksheet' tab with a 'Query Builder' interface. A script is being run to create a stored procedure named 'afisare_entrenorul_anului'. The script includes error handling logic using RAISE_APPLICATION_ERROR. The 'Script Output' tab at the bottom shows the execution results, including the creation of the stored procedure and an error message indicating that the 'raise_application_error' call was the cause of the error.

```
CREATE OR REPLACE PROCEDURE afisare_entrenorul_anului
  AS
    BEGIN
      WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru cșastigarea titlului. Va rugam frumos contactați managerul salii respective.');
        RAISE_APPLICATION_ERROR(-20002, 'Alegeți +roare!');

      END afisare_entrenorul_anului;
    /

DROP PROCEDURE afisare_entrenorul_anului;

BEGIN
  afisare_entrenorul_anului;
END;
/

BEGIN
  afisare_entrenorul_anului();
END;
/

BEGIN
  afisare_entrenorul_anului(2);
END;
/

BEGIN
  afisare_entrenorul_anului(5);
END;
```

```
Task completed in 0.05 seconds
*Cause:  The stored procedure 'raise_application_error'
          was called which causes this error to be generated.
*Action:  Correct the problem as described in the error message or contact
          the application administrator or DBA for more information.

Error starting at line 136 in command -
BEGIN
  afisare_entrenorul_anului(5);
END;
Error report -
ORA-20001: Există mai mulți antrenori eligibili pentru cșastigarea titlului. Va rugam frumos contactați managerul salii respective.
ORA-06512: at "SYSTEM.AFISARE_ENTRENORUL_ANULUI", line 36
ORA-06512: at line 2
```

10. Manager-ul a decis că nu se vor modifica contractele cu sponsorii, decât în intervalul 8:00 – 14:00, de luni până vineri. Având în vedere acest lucru, se definește un trigger care să permită lucrul asupra tabelului CONTRACT_SPONSOR (INSERT, UPDATE, DELETE) decât în intervalul precizat mai sus (se folosește un trigger de tip LMD la nivel de comandă).

```
CREATE OR REPLACE TRIGGER trig_contract
  BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
  IF (TO_CHAR(SYSDATE, 'DY') IN ('Sun', 'Sat'))
    OR (TO_CHAR(SYSDATE, 'HH24') NOT BETWEEN 8 AND 14)
  THEN
    RAISE_APPLICATION_ERROR(-20001, 'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
  END IF;
END;
/
```

```
DROP TRIGGER trig_contract;
```

```
INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));
```

```
UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5;
```

```
DELETE FROM contract_sponsor
WHERE id_sala = 1 and id_sponsor = 5;
```

```
SELECT SYSDATE FROM DUAL;
SELECT TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS') FROM DUAL;
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet Query Builder

```

CREATE OR REPLACE TRIGGER trig_contract
BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
  IF (TO_CHAR(TSYNDATE, 'DY') IN ('Sun', 'Sat')
    OR (TO_CHAR(TSYNDATE, 'H24') NOT BETWEEN 8 AND 14)
    THEN
      RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
    END IF;
  END;

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_spnsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5;

DELETE FROM contract_spnsor
WHERE id_sala = 1 and id_sponsor = 5;

```

Script Output A [Query Result X] Task completed in 0.029 seconds

Trigger TRIG_CONTRACT compiled

Console Output Buffer Size[20000]

MySQLA

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet Query Builder

```

CREATE OR REPLACE TRIGGER trig_contract
BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
  IF (TO_CHAR(TSYNDATE, 'DY') IN ('Sun', 'Sat')
    OR (TO_CHAR(TSYNDATE, 'H24') NOT BETWEEN 8 AND 14)
    THEN
      RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
    END IF;
  END;

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_spnsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5;

DELETE FROM contract_spnsor
WHERE id_sala = 1 and id_sponsor = 5;

```

Script Output A [Query Result X] Task completed in 0.03 seconds

Error starting at line 14 in command -
 INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'))
 Error at Command Line : 14 Column : 13
 Error report -
 SQL error: ORA-20001: Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.
 ORA-06512: at "SYSTEM.TRIG_CONTRACT", line 5
 ORA-04081: error during execution of trigger 'SYSTEM.TRIG_CONTRACT'

Console Output Buffer Size[20000]

MySQLA

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_contract
    BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
    BEGIN
        IF (TO_CHAR(SYSDATE, 'DY') IN ('Sun', 'Sat')
            OR (TO_CHAR(SYSDATE, 'HH24') NOT BETWEEN 8 AND 14)
            THEN
                RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
            END IF;
        END;

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5;

DELETE FROM contract_sponsor
WHERE id_sala = 1 and id_sponsor = 5;

```

Script Output - [Query Result] X

Task completed in 0.046 seconds

```

UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5
Error at Command line : 16 Column : 8
Error report -
SQL error: ORA-20001: Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.
ORA-06512: at "SYSTEM.TRIG_CONTRACT", line 5
ORA-04001: error during execution of trigger 'SYSTEM.TRIG_CONTRACT'

```

Done Output

MySQLA

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_contract
    BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
    BEGIN
        IF (TO_CHAR(SYSDATE, 'DY') IN ('Sun', 'Sat')
            OR (TO_CHAR(SYSDATE, 'HH24') NOT BETWEEN 8 AND 14)
            THEN
                RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
            END IF;
        END;

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5;

DELETE FROM contract_sponsor
WHERE id_sala = 1 and id_sponsor = 5;

```

Script Output - [Query Result] X

Task completed in 0.03 seconds

Error starting at line 1: 20 in command -

```

DELETE FROM contract_sponsor
WHERE id_sala = 1 and id_sponsor = 5
Error at Command line : 20 Column : 13
Error report -
SQL error: ORA-20001: Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.
ORA-06512: at "SYSTEM.TRIG_CONTRACT", line 5
ORA-04001: error during execution of trigger 'SYSTEM.TRIG_CONTRACT'

```

Done Output

MySQLA

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

MySQLA

Worksheet Query Builder

```

CREATE OR REPLACE TRIGGER trig_contract
BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
IF (TO_CHAR(SYSTIME, 'DT') IN ('Sun', 'Sat'))
OR (TO_CHAR(SYSTIME, 'HRS24') NOT BETWEEN 6 AND 14)
THEN
RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
END IF;
END;
/

```

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_seala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor

SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')

WHERE id_seala = 1 and id_sponsor = 5;

DELETE FROM contract_sponsor

WHERE id_seala = 1 and id_sponsor = 5;

select systime from dual;

Script Output Query Result All Rows Fetched: 1 in 0.002 seconds

SYSTIME

1 07-ZAN-24

Items Output Buffer Size: 20000

MySQLA

Click on an identifier with the Control key down to perform "Go to Declaration".

Oracle SQL Developer : C:\Users\Alex\MySALA2.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

MySQLA

SQL Worksheet History

Worksheet Query Builder

```

CREATE OR REPLACE TRIGGER trig_contract
BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
IF (TO_CHAR(SYSTIME, 'DT') IN ('Sun', 'Sat'))
OR (TO_CHAR(SYSTIME, 'HRS24') NOT BETWEEN 6 AND 14)
THEN
RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
END IF;
END;
/

```

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_seala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor

Task completed in 0.071 seconds

1 row inserted.

Items Output Buffer Size: 20000

MySQLA

Messages+Log

Messages Logging Page Statements

MySQLA

Oracle SQL Developer : C:\Users\Alex\MySALA2.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections MySALA SQL Worksheet History

Worksheet Query Builder

```
CREATE OR REPLACE TRIGGER trig_contract
BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
IF (TO_CHAR(SYSTIME, 'DD') IN ('Sun', 'Sat'))
OR (TO_CHAR(SYSTIME, 'HRS24') NOT BETWEEN 6 AND 14)
THEN
RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
END IF;
END;
```

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_seia, id_sponsor, date_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_seia = 1 and id_sponsor = 5;

DELETE FROM contract_sponsor
WHERE id_seia = 1 and id_sponsor = 5;

select systime from dual;

Script Output X Task completed in 0.029 seconds

1 row inserted.

1 row updated.

Items Output Buffer Size [20000]

MySALA x

Messages - Log

Messages Logging Page Statements

Line 18 Column 30 Insert Windows

Oracle SQL Developer : C:\Users\Alex\MySALA2.sql

File Edit View Navigate Run Source Team Tools Window Help

Connections MySALA SQL Worksheet History

Worksheet Query Builder

```
CREATE OR REPLACE TRIGGER trig_contract
BEFORE INSERT OR UPDATE OR DELETE ON contract_sponsor
BEGIN
IF (TO_CHAR(SYSTIME, 'DD') IN ('Sun', 'Sat'))
OR (TO_CHAR(SYSTIME, 'HRS24') NOT BETWEEN 6 AND 14)
THEN
RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 8-14.');
END IF;
END;
```

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_seia, id_sponsor, date_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-03', 'YYYY-MM-DD')
WHERE id_seia = 1 and id_sponsor = 5;

DELETE FROM contract_sponsor
WHERE id_seia = 1 and id_sponsor = 5;

select systime from dual;

Script Output X Task completed in 0.025 seconds

1 row updated.

1 row deleted.

Items Output Buffer Size [20000]

MySALA x

Messages - Log

Messages Logging Page Statements

Line 21 Column 30 Insert Windows

```

CREATE OR REPLACE TRIGGER trig_contract
BEFORE UPDATE ON contract_sponsor
FOR EACH ROW
BEGIN
    IF (TO_CHAR(SYSDATE, 'D') IN ('Sun', 'Sat'))
        OR (TO_CHAR(SYSDATE, 'HH24') NOT BETWEEN 5 AND 14)
    THEN
        RAISE_APPLICATION_ERROR(-20001,'Contractele se pot modifica doar de luni pana vineri, in intervalul orar 5-14.');
    END IF;
END;
/

```

```

DROP TRIGGER trig_contract;

INSERT INTO CONTRACT_SPONSOR (id_sala, id_sponsor, data_contract) VALUES (1, 5, TO_DATE('2023-01-01', 'YYYY-MM-DD'));

UPDATE contract_sponsor
SET data_contract = TO_DATE('2024-01-01', 'YYYY-MM-DD')
WHERE id_sala = 1 and id_sponsor = 5;

DELETE FROM contract_sponsor
WHERE id_sala = 1 and id_sponsor = 5;

SELECT TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS') FROM DUAL;

```

11. Manager-ul dorește să se creeze tabelul info_sala_capacitate cu următoarele coloane:

- id_sala (codul sălii) – cheie primară;
- nume_sala (numele sălii);
- capacitate_totala (capacitatea alocată pentru vestiarele existente în sala respectivă)

Se introduc date în tabelul creat anterior corespunzătoare informațiilor existente în schemă. Se definește un trigger care va actualiza automat câmpul capacitate_totala atunci când se introduce un nou vestiar, respectiv se șterge un vestiar sau se modifică capacitatea unui vestiar (se folosește un trigger de tip LMD la nivel de linie). Se folosește o procedură stocată pentru a calcula noua capacitate.

```
select * from vestiar;
```

```
select * from sala;
```

```

CREATE TABLE info_sala_capacitate (
    id_sala INT PRIMARY KEY,
    nume_sala VARCHAR(50),
    capacitate_totala INT
);

```

```
drop table info_sala_capacitate;

INSERT INTO info_sala_capacitate (id_sala, nume_sala, capacitate_totala) VALUES (1, 'World Class', 110 );
select * from info_sala_capacitate;
```

```
CREATE TABLE vestiar_copie AS
SELECT *
FROM vestiar;
```

```
drop table vestiar_copie;

select * from vestiar_copie;
```

```
CREATE OR REPLACE PROCEDURE modific_capacitate_totala
(v_id_sala info_sala_capacitate.id_sala%TYPE,
v_capacitate_totala info_sala_capacitate.capacitate_totala%TYPE) AS
BEGIN
UPDATE info_sala_capacitate
SET capacitate_totala = NVL (capacitate_totala, 0) + v_capacitate_totala
WHERE id_sala = v_id_sala;
END;
/
```

```
CREATE OR REPLACE TRIGGER trig_capacitate
AFTER DELETE OR UPDATE OR INSERT OF capacitate ON vestiar_copie
FOR EACH ROW
BEGIN
IF DELETING THEN
```

```
modific_capacitate_totala (:OLD.id_sala, -1*:OLD.capacitate);

ELSIF UPDATING THEN

modific_capacitate_totala(:OLD.id_sala,:NEW.capacitate-:OLD.capacitate);

ELSE

modific_capacitate_totala(:NEW.id_sala, :NEW.capacitate);

END IF;

END;

/
```

```
SELECT * FROM info_sala_capacitate WHERE id_sala=1;
```

```
INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (16, 'M', 90, 1);
```

```
SELECT * FROM info_sala_capacitate WHERE id_sala=1;
```

```
SELECT * FROM info_sala_capacitate WHERE id_sala=1;
UPDATE vestiar_copie
SET capacitate = capacitate + 1000
WHERE id_vestiar=16;
```

```
SELECT * FROM info_sala_capacitate WHERE id_sala=1;
```

```
DELETE FROM vestiar_copie
WHERE id_vestiar=16;
```

```
SELECT * FROM info_sala_capacitate WHERE id_sala=1;
```

```
DROP PROCEDURE modif_capacitate_totala;
```

```
DROP TRIGGER trig_capacitate;
```

The screenshot shows the Oracle SQL Developer interface with the 'MySALA' connection selected. In the 'Worksheet' tab, a query builder window displays the following SQL code:

```
select * from vestiar;
select * from sala;

CREATE TABLE info_sala_capacitate (
    id_sala INT PRIMARY KEY,
    nume_sala VARCHAR(50),
    capacitate_totala INT
);

drop table info_sala_capacitate;

INSERT INTO info_sala_capacitate (id_sala, nume_sala, capacitate_totala) VALUES (1, 'World Class', 110 );
select * from info_sala_capacitate;

CREATE TABLE vestiar_copie AS
SELECT * FROM vestiar;
TRUNC vestiar;

drop table vestiar_copie;

select * from vestiar_copie;

CREATE OR REPLACE PROCEDURE modif_capacitate_totala
    (v_id_sala info_sala_capacitate.id_sala%TYPE,
     v_capacitate_totala info_sala_capacitate.capacitate_totala%TYPE)
AS
```

The 'Script Output' and 'Query Result' panes show the results of the 'select * from vestiar;' query, which returns five rows of data:

ID_SALA	NUME_SALA
1	World Class
2	Fit Gym
3	Fitness Zone
4	Gym Plus
5	Power Fitness

The screenshot shows the Oracle SQL Developer interface with the 'MySALA' connection selected. In the 'Worksheet' tab, a query builder window displays the same SQL code as the previous screenshot, but the results in the 'Query Result' pane are different, indicating a change in the data:

ID_VESTIAR	TP_VESTIAR	CAPACITATE	ID_SALA
1	LM	50	1
2	2T	60	1
3	3M	40	2
4	4T	30	2
5	5M	100	3
6	6F	80	3
7	7M	120	3
8	8M	...	-

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

MySQLA

Tables (Filtered)

Views

Sequences

Packages

Procedures

Functions

Operators

UML

Queues Tables

Triggers

Types

Sequences

Materialized Views

Materialized View Logs

Synonyms

Public Synonyms

Public Object Info

Public Database Links

Directories

Editors

Java

XML Schemas

VML DB Repository

OLAP Option

Reports

All Reports

Analytic View Reports

Data Dictionary Reports

Data Modeler Reports

OLAP Reports

Timeline Reports

User Defined Reports

Worksheet Query Builder

```

select * from vestiar;
select * from sala;

CREATE TABLE info_sala_capacitate (
    id_sala INT PRIMARY KEY,
    nume_sala VARCHAR(50),
    capacitate_totala INT
);

drop table info_sala_capacitate;

INSERT INTO info_sala_capacitate (id_sala, nume_sala, capacitate_totala) VALUES (1, 'World Class', 110);

CREATE TABLE vestiar_copie AS
SELECT *
FROM vestiar;

drop table vestiar_copie;

select * from vestiar_copie;

CREATE OR REPLACE PROCEDURE modific_capacitate_totala
(
    v_id_sala info_sala_capacitate.id_sala%TYPE,
    v_capacitate_totala info_sala_capacitate.capacitate_totala%TYPE
)
AS
```

Script Output X | Query Result X | Task completed in 0.023 seconds

Table INFO_SALA_CAPACITATE created.

Items Output

MySQLA

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

MySQLA

Tables (Filtered)

Views

Sequences

Packages

Procedures

Functions

Operators

UML

Queues Tables

Triggers

Types

Sequences

Materialized Views

Materialized View Logs

Synonyms

Public Synonyms

Public Object Info

Public Database Links

Directories

Editors

Java

XML DB Repository

OLAP Option

Reports

All Reports

Analytic View Reports

Data Dictionary Reports

Data Modeler Reports

OLAP Reports

Timeline Reports

User Defined Reports

Worksheet Query Builder

```

select * from vestiar;
select * from sala;

CREATE TABLE info_sala_capacitate (
    id_sala INT PRIMARY KEY,
    nume_sala VARCHAR(50),
    capacitate_totala INT
);

drop table info_sala_capacitate;

INSERT INTO info_sala_capacitate (id_sala, nume_sala, capacitate_totala) VALUES (1, 'World Class', 110);

CREATE TABLE vestiar_copie AS
SELECT *
FROM vestiar;

drop table vestiar_copie;

select * from vestiar_copie;

CREATE OR REPLACE PROCEDURE modific_capacitate_totala
(
    v_id_sala info_sala_capacitate.id_sala%TYPE,
    v_capacitate_totala info_sala_capacitate.capacitate_totala%TYPE
)
AS
```

Script Output X | Query Result X | All Rows Fetched: 1 in 0.04 seconds

ID_SALA	NUME_SALA	CAPACITATE_TOTALA
1	World Class	110

Items Output

MySQLA

Oracle SQL Developer : MySQL

```

CREATE TABLE info_sala_capacitate (
    id_sala INT PRIMARY KEY,
    nume_sala VARCHAR(50),
    capacitate_totala INT
);

drop table info_sala_capacitate;

INSERT INTO info_sala_capacitate (id_sala, nume_sala, capacitate_totala) VALUES (1, 'World Class', 110 );

select * from info_sala_capacitate;

CREATE TABLE vestiar_copie AS
SELECT *
FROM vestiar;

drop table vestiar_copie;

select * from vestiar_copie;

CREATE OR REPLACE PROCEDURE modific_capacitate_totala
(
    v_id_sala info_sala_capacitate.id_sala%TYPE,
    v_capacitate_totala info_sala_capacitate.capacitate_totala%TYPE
)
AS
BEGIN
    UPDATE info_sala_capacitate
    SET capacitate_totala = NVL(capacitate_totala, 0) + v_capacitate_totala
    WHERE id_sala = v_id_sala;
END;
/

```

Table INFO_SALA_CAPACITATE created.

1 row inserted.

>>Query Run In/Query Result 2

Table VESTIAR_COPIE created.

Oracle SQL Developer : MySQL

```

CREATE TABLE info_sala_capacitate (
    id_sala INT PRIMARY KEY,
    nume_sala VARCHAR(50),
    capacitate_totala INT
);

drop table info_sala_capacitate;

INSERT INTO info_sala_capacitate (id_sala, nume_sala, capacitate_totala) VALUES (1, 'World Class', 110 );

select * from info_sala_capacitate;

CREATE TABLE vestiar_copie AS
SELECT *
FROM vestiar;

drop table vestiar_copie;

select * from vestiar_copie;

CREATE OR REPLACE PROCEDURE modific_capacitate_totala
(
    v_id_sala info_sala_capacitate.id_sala%TYPE,
    v_capacitate_totala info_sala_capacitate.capacitate_totala%TYPE
)
AS
BEGIN
    UPDATE info_sala_capacitate
    SET capacitate_totala = NVL(capacitate_totala, 0) + v_capacitate_totala
    WHERE id_sala = v_id_sala;
END;
/

```

Table INFO_SALA_CAPACITATE created.

All Rows Fetched 15 in 0.005 seconds

ID_VESTIAR	TIP_VESTIAR	CAPACITATE	ID_SALA
1	1M	50	1
2	2T	60	1
3	3M	40	2
4	4F	30	2
5	5M	100	3
6	6F	80	3
7	7M	120	3
8	8M	100	3

Items Copied

MySALA x

Oracle SQL Developer : MySQLA

```

CREATE OR REPLACE PROCEDURE modif_capacitate_totala
( v_id_sala info_sala_capacitate.id_sala%TYPE,
  v_capacitate_totala info_sala_capacitate.capacitate_totals%TYPE ) AS
BEGIN
  UPDATE info_sala_capacitate
  SET capacitate_totala = INV (capacitate_totals, 0) + v_capacitate_totala
  WHERE id_sala = v_id_sala;
END;

CREATE OR REPLACE TRIGGER trig_capacitate
  AFTER INSERT OR UPDATE OR INSERT OF capacitate ON vestiar_copy
  FOR EACH ROW
  BEGIN
    IF DELETING THEN
      modif_capacitate_totala (OLD.id_sala, -1*OLD.capacitate);
    ELSEIF UPDATING THEN
      modif_capacitate_totala(OLD.id_sala,NEW.capacitate-OLD.capacitate);
    ELSE
      modif_capacitate_totala(NEW.id_sala, NEW.capacitate);
    END IF;
  END;

```

1 ROW inserted.

>>Query Run In:Query Result 2

Table VESTIAR_COPY created.

Procedure MODIFIC_CAPACITATE_TOTALA compiled

Oracle SQL Developer : MySQLA

```

CREATE OR REPLACE PROCEDURE modif_capacitate_totala
( v_id_sala info_sala_capacitate.id_sala%TYPE,
  v_capacitate_totala info_sala_capacitate.capacitate_totals%TYPE ) AS
BEGIN
  UPDATE info_sala_capacitate
  SET capacitate_totala = INV (capacitate_totals, 0) + v_capacitate_totala
  WHERE id_sala = v_id_sala;
END;

CREATE OR REPLACE TRIGGER trig_capacitate
  AFTER INSERT OR UPDATE OR INSERT OF capacitate ON vestiar_copy
  FOR EACH ROW
  BEGIN
    IF DELETING THEN
      modif_capacitate_totala (OLD.id_sala, -1*OLD.capacitate);
    ELSEIF UPDATING THEN
      modif_capacitate_totala(OLD.id_sala,NEW.capacitate-OLD.capacitate);
    ELSE
      modif_capacitate_totala(NEW.id_sala, NEW.capacitate);
    END IF;
  END;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

```

Table VESTIAR_COPY created.

Procedure MODIFIC_CAPACITATE_TOTALA compiled

Trigger TRIG_CAPACITATE compiled

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_capacitate
AFTER DELETE OR UPDATE OR INSERT ON capacitate
FOR EACH ROW
BEGIN
    IF DELETING THEN
        modifica_capacitate_totala(:OLD.id_sala, -1,:OLD.capacitate);
    ELSIF UPDATING THEN
        modifica_capacitate_totala(:OLD.id_sala,:NEW.capacitate-:OLD.capacitate);
    ELSE
        modifica_capacitate_totala(:NEW.id_sala, :NEW.capacitate);
    END IF;
END;
/

```

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (16, 'M', 90, 1);

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

All Rows Fetched 0 in 0.02 seconds

ID_SALA	NAME_SALA	CAPACITATE_TOTALA
1	World Class	110

Items Output Buffer Size [20000]

MySQLA

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_capacitate
AFTER DELETE OR UPDATE OR INSERT ON capacitate
FOR EACH ROW
BEGIN
    IF DELETING THEN
        modifica_capacitate_totala(:OLD.id_sala, -1,:OLD.capacitate);
    ELSIF UPDATING THEN
        modifica_capacitate_totala(:OLD.id_sala,:NEW.capacitate-:OLD.capacitate);
    ELSE
        modifica_capacitate_totala(:NEW.id_sala, :NEW.capacitate);
    END IF;
END;
/

```

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (16, 'M', 90, 1);

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;
UPDATE vestiar_copie
SET capacitate = capacitate + 1000

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

Task completed in 0.013 seconds

Procedure MODIFIC_CAPACITATE_TOTALA compiled

Trigger TRIG_CAPACITATE_TOTALA compiled

1 row inserted.

Items Output Buffer Size [20000]

MySQLA

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet Query Builder

```

CREATE OR REPLACE TRIGGER trig_capacitate
  AFTER DELETE OR UPDATE OR INSERT ON capacitate
  FOR EACH ROW
  BEGIN
    IF DELETING THEN
      modificar_capacitate_totala (:OLD.id_sala, -1::OLD.capacitate);
    ELSIF UPDATING THEN
      modificar_capacitate_totala(:OLD.id_sala,:NEW.capacitate-:OLD.capacitate);
    ELSE
      modificar_capacitate_totala(:NEW.id_sala, :NEW.capacitate);
    END IF;
  END;
/
SELECT * FROM info_sala_capacitate WHERE id_sala=1;

INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (16, 'N', 90, 1);

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;
UPDATE vestiar_copie
SET capacitate = capacitate + 1000

```

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

All Rows Fetched 0 in 0.001 seconds

ID_SALA	NAME_SALA	CAPACITATE_TOTALA
1	World Class	200

Items Output Buffer Size [20000]

MySQLA

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet Query Builder

```

modificar_capacitate_totala (:NEW.id_sala, :NEW.capacitate);
END IF;
END;
/

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (16, 'N', 90, 1);

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;
UPDATE vestiar_copie
SET capacitate = capacitate + 1000
WHERE id_vestiar=16;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

DELETE FROM vestiar_copie
WHERE id_vestiar=16;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

```

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

Trigger TRIG_CAPACITATE compiled

1 row inserted.

1 row updated.

Items Output Buffer Size [20000]

MySQLA

Oracle SQL Developer : MySALA

Connections Worksheet Script Output Query Result 1 Query Result 2 Query Result 3 Query Result 4 MySQLA

```

modific_capacitate_totala(:NEW.id_sala, :NEW.capacitate);
END IF;
END;

/
SELECT * FROM info_sala_capacitate WHERE id_sala=1;

INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (1, 'M', 90, 1);

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;
UPDATE vestiar_copie
SET capacitate = capacitate + 1000
WHERE id_vestiar=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

DELETE FROM vestiar_copie
WHERE id_vestiar=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

```

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

All Rows Fetched 1 in 0.001 seconds

ID_SALA	NAME_SALA	CAPACITATE_TOTALA
1	World Class	1200

Items Output Buffer Size [20000] MySQLA

Oracle SQL Developer : MySALA

File Edit View Navigate Run Source Team Tools Window Help

Connections Worksheet Script Output Query Result 1 Query Result 2 Query Result 3 Query Result 4 MySQLA

```

INSERT INTO vestiar_copie (id_vestiar, tip_vestiar, capacitate, id_sala)
VALUES (1, 'M', 90, 1);

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;
UPDATE vestiar_copie
SET capacitate = capacitate + 1000
WHERE id_vestiar=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

DELETE FROM vestiar_copie
WHERE id_vestiar=1;

SELECT * FROM info_sala_capacitate WHERE id_sala=1;

DROP PROCEDURE modific_capacitate_totala;
DROP TRIGGER trig_capacitate;
```

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

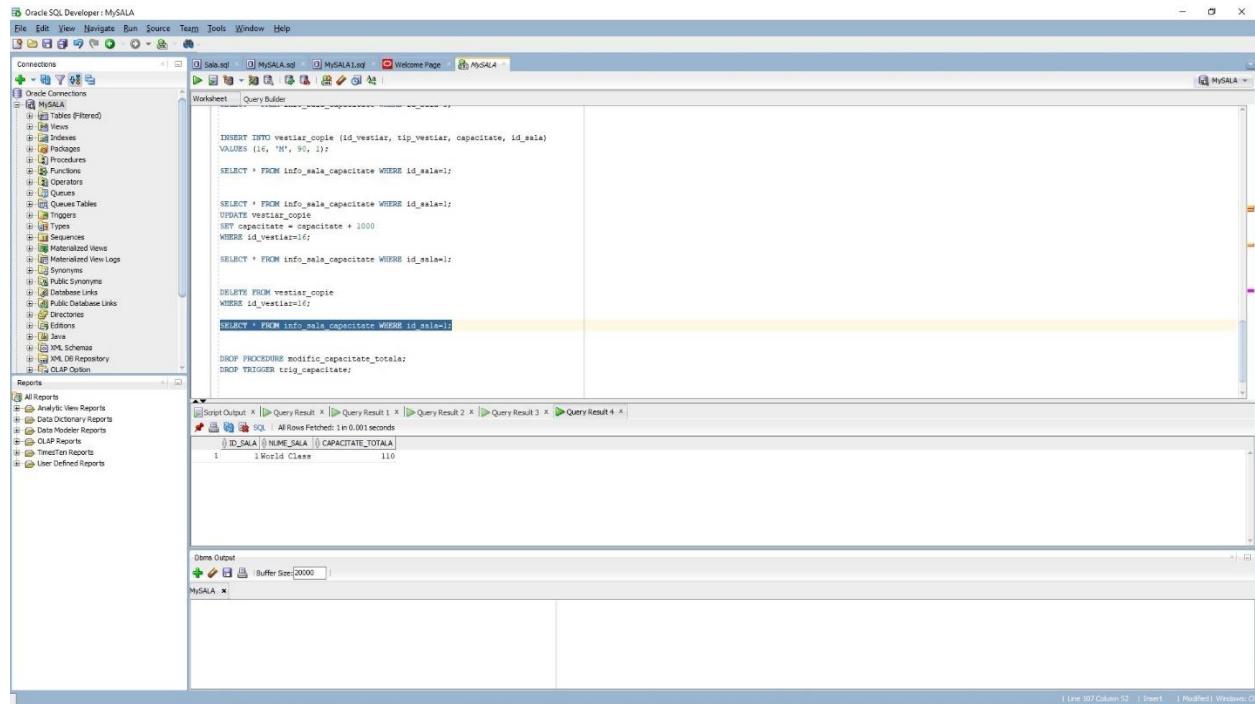
Task completed in 0.021 seconds

1 row inserted.

1 row updated.

1 row deleted.

Items Output Buffer Size [20000] MySQLA



12. Manager-ul dorește ca de fiecare dată când se modifică baza de date, acest lucru să fie anunțat automat printr-un mesaj. Având în vedere acest lucru, se definește un trigger de tip LDD pentru a rezolva dorința manager-ului, mesajul conținând data și ora exactă a momentului în care s-au făcut modificările.

CREATE OR REPLACE TRIGGER trig_global

AFTER CREATE OR ALTER OR DROP ON SCHEMA

DECLARE

v_mesaj VARCHAR2(100);

BEGIN

v_mesaj := 'Atenție! Acțiune LDD asupra bazei de date la data: ' || TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS');

DBMS_OUTPUT.PUT_LINE(v_mesaj);

END;

/

DROP TRIGGER trig_global;

```
CREATE TABLE contract_sponsor_copie AS  
SELECT *  
FROM contract_sponsor;
```

```
ALTER TABLE contract_sponsor_copie ADD (valoare INT);
```

```
DROP TABLE contract_sponsor_copie;
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_global
AFTER CREATE OR ALTER OR DROP ON SCHEMA
DECLARE
    _v_mesa VARCHAR2(100);
BEGIN
    _v_mesa := 'Atentie! Actiunea IUD asupra bazei de date la data: ' || TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS');
    DBMS_OUTPUT.PUT_LINE(_v_mesa);
END;
/
DROP TRIGGER trig_global;

CREATE TABLE contract_sponsor_copie AS
SELECT *
FROM contract_sponsor;

ALTER TABLE contract_sponsor_copie ADD (veloare INT);

DROP TABLE contract_sponsor_copie;

```

Script Output: A Task completed in 0.028 seconds

Trigger TRIG_GLOBAL compiled

Done Output Buffer Size[20000]

MySQLA

Compiler - Log

Messages Logging Page Statements Compiler

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_global
AFTER CREATE OR ALTER OR DROP ON SCHEMA
DECLARE
    _v_mesa VARCHAR2(100);
BEGIN
    _v_mesa := 'Atentie! Actiunea IUD asupra bazei de date la data: ' || TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS');
    DBMS_OUTPUT.PUT_LINE(_v_mesa);
END;
/
DROP TRIGGER trig_global;

CREATE TABLE contract_sponsor_copie AS
SELECT *
FROM contract_sponsor;

ALTER TABLE contract_sponsor_copie ADD (veloare INT);

DROP TABLE contract_sponsor_copie;

```

Script Output: A Task completed in 0.028 seconds

Trigger TRIG_GLOBAL compiled

Table CONTRACT_SPONSOR_COPIE created.

Done Output Buffer Size[20000]

MySQLA

Compiler - Log

Messages Logging Page Statements Compiler

Atentie! Actiunea IUD asupra bazei de date la data: 2024-01-07 16:57:48

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_global
AFTER CREATE OR ALTER OR DROP ON SCHEMA
DECLARE
    v_messa VARCHAR2(100);
BEGIN
    v_messa := 'Atentie! Actiunea LOD asupra bazei de date la data: ' || TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS');
    DBMS_OUTPUT.PUT_LINE(v_messa);
END;
/

```

DROP TRIGGER trig_global;

CREATE TABLE contract_sponsor_copie AS

```

SELECT *
FROM contract_sponsor;

```

ALTER TABLE contract_sponsor_copie ADD (valoare INT);

DROP TABLE contract_sponsor_copie;

Script Output A Task completed in 0.028 seconds

Trigger TRIG_GLOBAL compiled

Table CONTRACT_SPONSOR_COPIE created.

Table CONTRACT_SPONSOR_COPIE altered.

Done Output Buffer Size[20000]

MySQLA x

Atentie! Actiunea LOD asupra bazei de date la data: 2024-01-07 16:57:48

Atentie! Actiunea LOD asupra bazei de date la data: 2024-01-07 16:57:55

Compiler - Log

Messages Logging Page Statements Compiler

J Line 130 Column 54 Input Modified Windows

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE TRIGGER trig_global
AFTER CREATE OR ALTER OR DROP ON SCHEMA
DECLARE
    v_messa VARCHAR2(100);
BEGIN
    v_messa := 'Atentie! Actiunea LOD asupra bazei de date la data: ' || TO_CHAR(SYSDATE, 'YYYY-MM-DD HH24:MI:SS');
    DBMS_OUTPUT.PUT_LINE(v_messa);
END;
/

```

DROP TRIGGER trig_global;

CREATE TABLE contract_sponsor_copie AS

```

SELECT *
FROM contract_sponsor;

```

ALTER TABLE contract_sponsor_copie ADD (valoare INT);

DROP TABLE contract_sponsor_copie;

Script Output A Task completed in 0.028 seconds

Table CONTRACT_SPONSOR_COPIE created.

Table CONTRACT_SPONSOR_COPIE altered.

Table CONTRACT_SPONSOR_COPIE dropped.

Done Output Buffer Size[20000]

MySQLA x

Atentie! Actiunea LOD asupra bazei de date la data: 2024-01-07 16:57:48

Atentie! Actiunea LOD asupra bazei de date la data: 2024-01-07 16:57:55

Atentie! Actiunea LOD asupra bazei de date la data: 2024-01-07 16:58:05

Compiler - Log

Messages Logging Page Statements Compiler

J Line 130 Column 55 Input Modified Windows

13. Manager-ul dorește crearea unui pachet care să conțină tot ce s-a realizat mai sus.

CREATE OR REPLACE PACKAGE pachet AS

PROCEDURE afisare_informatii_sala (v_id_sala sala.id_sala%TYPE DEFAULT 1);

```
PROCEDURE afisare_informatii_vestiare (v_id_sala sala.id_sala%TYPE DEFAULT 1);
FUNCTION suma_bani (v_id_sala sala.id_sala%TYPE DEFAULT 1) RETURN abonament.pret%TYPE;
PROCEDURE afisare_antrenorul_anului (v_id_sala sala.id_sala%TYPE DEFAULT 1);
END pachet;
/
```

```
CREATE OR REPLACE PACKAGE BODY pachet AS
PROCEDURE afisare_informatii_sala
(v_id_sala sala.id_sala%TYPE DEFAULT 1)
IS
TYPE tablou_indexat IS TABLE OF VARCHAR(50) INDEX BY PLS_INTEGER;
TYPE tablou_imbricat IS TABLE OF NUMBER;
TYPE tablou_vector IS VARRAY(20) OF VARCHAR(50);
```

```
v_lista_angajati tablou_indexat := tablou_indexat();
v_lista_membrii tablou_imbricat := tablou_imbricat();
v_lista_locatii tablou_vector := tablou_vector();
```

```
v_nume_sala VARCHAR(50);
```

```
BEGIN
```

```
SELECT nume_sala
INTO v_nume_sala
FROM sala
WHERE id_sala = v_id_sala;
```

```
FOR angajat IN (SELECT id_angajat, nume_angajat FROM angajat WHERE id_sala = v_id_sala)
LOOP
```

```

v_lista_angajati(angajat.id_angajat) := angajat.nume_angajat;
END LOOP;

FOR membru IN (SELECT id_membru FROM membru WHERE id_sala = v_id_sala)
LOOP
    v_lista_membrii.extend;
    v_lista_membrii(v_lista_membrii.last) := membru.id_membru;
END LOOP;

FOR locatie IN (SELECT oras_locatie FROM locatie WHERE id_sala = v_id_sala)
LOOP
    v_lista_locatii.extend;
    v_lista_locatii(v_lista_locatii.last) := locatie.oras_locatie;
END LOOP;

DBMS_OUTPUT.PUT_LINE('Informatii despre sala ' || v_nume_sala || ':');

DBMS_OUTPUT.PUT_LINE('Lista angajati:');
FOR i IN v_lista_angajati.first .. v_lista_angajati.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_angajati(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Lista membrii (id-uri):');
FOR i IN v_lista_membrii.first .. v_lista_membrii.last
LOOP
    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_membrii(i));
END LOOP;

```

```
DBMS_OUTPUT.PUT_LINE('Lista locatii (orasele):');

FOR i IN v_lista_locatii.first .. v_lista_locatii.last

LOOP

    DBMS_OUTPUT.PUT_LINE(' ' || v_lista_locatii(i));

END LOOP;

END afisare_informatii_sala;
```

```
PROCEDURE afisare_informatii_vestiare
(v_id_sala sala.id_sala%TYPE DEFAULT 1)

IS

    v_nume_sala sala.nume_sala%TYPE;
    v_id_vestiar vestiar.id_vestiar%TYPE;
    v_tip CHAR(1);
    v_capacitate INT;
    v_id_ingrijitor INT;
    v_nume_angajat VARCHAR(50);

CURSOR c_vestiar IS
SELECT id_vestiar, tip_vestiar, capacitate
FROM vestiar
WHERE id_sala=v_id_sala;

CURSOR c_ingrijitor (vestiar_id vestiar.id_vestiar%TYPE) IS
SELECT id_ingrijitor, nume_angajat
FROM ingrijeste i, angajat a
WHERE i.id_vestiar = vestiar_id and a.id_angajat=i.id_ingrijitor;
```

```

BEGIN

    SELECT nume_sala
        INTO v_nume_sala
        FROM sala
       WHERE id_sala = v_id_sala;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE('Informatii despre vestiarele salii ' || v_nume_sala || ':');

    OPEN c_vestiar;

    LOOP
        FETCH c_vestiar INTO v_id_vestiar, v_tip, v_capacitate;
        EXIT WHEN c_vestiar%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('-----');
        DBMS_OUTPUT.PUT_LINE('Vestiarul ' || v_id_vestiar || ' de tipul ' || v_tip || ' si cu capacitatea ' || v_capacitate || '.');
        DBMS_OUTPUT.PUT_LINE('Lista ingrijitorilor: ');

        FOR ingrijitor_i IN c_ingrijitor(v_id_vestiar) LOOP
            DBMS_OUTPUT.PUT_LINE(ingrijitor_i.id_ingrijitor || '.' || ingrijitor_i.nume_angajat);
        END LOOP;

    END LOOP;

    DBMS_OUTPUT.PUT_LINE('-----');
    CLOSE c_vestiar;

```

```

END afisare_informatii_vestiare;

FUNCTION suma_bani
(v_id_sala sala.id_sala%TYPE DEFAULT 1)
RETURN abonament.pret%TYPE IS
v_suma abonament.pret%TYPE;
exceptie1 EXCEPTION;
exceptie2 EXCEPTION;
BEGIN
BEGIN
SELECT SUM(a.pret)
INTO v_suma
FROM abonament a
JOIN membru m ON m.id_membru = a.id_membru
JOIN sala s ON s.id_sala = m.id_sala
WHERE s.id_sala = v_id_sala;

IF v_suma <= 1000 THEN
RAISE exceptie1;
END IF;

IF v_suma <= 2000 THEN
RAISE exceptie2;
END IF;

EXCEPTION
WHEN exceptie1 THEN
DBMS_OUTPUT.PUT_LINE('Sala este in faliment!');


```

```
WHEN exceptie2 THEN  
    DBMS_OUTPUT.PUT_LINE('Sala este aproape de faliment!');  
END;
```

```
RETURN v_suma;
```

```
EXCEPTION
```

```
WHEN OTHERS THEN  
    DBMS_OUTPUT.PUT_LINE('Alta eroare!');
```

```
END suma_bani;
```

```
PROCEDURE afisare_antrenorul_anului  
(v_id_sala sala.id_sala%TYPE DEFAULT 1)  
IS  
    v_nume_angajat angajat.nume_angajat%type;  
BEGIN  
  
    SELECT a1.nume_angajat  
    INTO v_nume_angajat  
    FROM angajat a1  
    JOIN antrenor a2 ON a1.id_angajat = a2.id_angajat  
    JOIN programe p ON p.id_angajat = a1.id_angajat  
    JOIN foloseste f ON f.id_antrenament = p.id_antrenament  
    JOIN sala s ON s.id_sala = a1.id_sala  
    WHERE s.id_sala = v_id_sala  
    GROUP BY a1.nume_angajat  
    HAVING COUNT(*) = (  
        SELECT MAX(cnt)
```

```
        FROM (
            SELECT COUNT(*) as cnt
            FROM angajat a3
            JOIN antrenor a4 ON a3.id_angajat = a4.id_angajat
            JOIN programe p2 ON p2.id_angajat = a3.id_angajat
            JOIN foloseste f2 ON f2.id_antrenament = p2.id_antrenament
            JOIN sala s2 ON s2.id_sala = a3.id_sala
            GROUP BY s2.nume_sala, a3.nume_angajat
        )
    );
```

```
DBMS_OUTPUT.PUT_LINE('Antrenorul anului este ' || v_nume_angajat || '.');
```

```
EXCEPTION
```

```
    WHEN NO_DATA_FOUND THEN
```

```
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in sala cu id-ul dat.');
```

```
    WHEN TOO_MANY_ROWS THEN
```

```
        RAISE_APPLICATION_ERROR(-20001, 'Exista mai multi antrenori eligibili pentru castigarea titlului. Va rugam frumos contactati managerul salii respective.');
```

```
    WHEN OTHERS THEN
```

```
        RAISE_APPLICATION_ERROR(-20002, 'Alta eroare!');
```

```
END afisare_antrenorul_anului;
```

```
END pachet;
```

```
/
```

```
drop package pachet;
```

```
EXECUTE pachet.afisare_informatii_sala(1);
EXECUTE pachet.afisare_informatii_vestiare(1);
DECLARE
  v_rezultat abonament.pret%TYPE;
BEGIN
  v_rezultat := pachet.suma_bani(1);
  DBMS_OUTPUT.PUT_LINE(v_rezultat);
END;
/
EXECUTE pachet.afisare_antrenorul_anului(1);
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE PACKAGE pacchet AS
  PROCEDURE afisare_informatii_sala (v_id_sala sala.id_salaTYPE DEFAULT 1);
  PROCEDURE afisare_informatii_vestimente (v_id_sala sala.id_salaTYPE DEFAULT 1);
  FUNCTION suma_bani (v_id_sala sala.id_salaTYPE DEFAULT 1) RETURN abonament.pretTVPS;
  PROCEDURE afisare_antrenorul_angajat (v_id_sala sala.id_salaTYPE DEFAULT 1);
  END pacchet;
/

CREATE OR REPLACE PROCEDURE scrieI_pacchet AS
  PROCEDURE afisare_informatii_sala
  IS
    TYPE tabluu_indexat IS TABLE OF VARCHAR(50) INDEX BY PLS_INTEGER;
    TYPE tabluu_imbricat IS TABLE OF NUMBER;
    TYPE tabluu_vector IS VARAY(20) OF VARCHAR(50);

    v_lista_angajati tabluu_indexat := tabluu_indexat();
    v_lista_membrilor tabluu_imbricat := tabluu_imbricat();
    v_lista_locatii tabluu_vector := tabluu_vector();

    v_name_sala VARCHAR(50);

  BEGIN
    SELECT nume_sala
    INTO v_name_sala
    FROM sala
    WHERE id_sala = v_id_sala;

    FOR angajat IN (SELECT id_angajat, nume_angajat FROM angajat WHERE id_sala = v_id_sala)
    LOOP
      v_lista_angajati(angajat.id_angajat) := angajat.nume_angajat;
    END LOOP;

    FOR membru IN (SELECT id_membru FROM membru WHERE id_sala = v_id_sala)
    LOOP
      v_lista_membrilor.extend;
      v_lista_membrilor(v_lista_membrilor.last) := membru.id_membru;
    END LOOP;
  END;
/

```

Script Output | Task completed in 0.03 seconds

Package PACCHET compiled

Dime Output Compiler - Log Messages Logging Help Characters Compiler | Log (150 Columns) Input Modified Windows

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

CREATE OR REPLACE PACKAGE BODY pacchet AS
  PROCEDURE afisare_informatii_sala
  IS
    TYPE tabluu_indexat IS TABLE OF VARCHAR(50) INDEX BY PLS_INTEGER;
    TYPE tabluu_imbricat IS TABLE OF NUMBER;
    TYPE tabluu_vector IS VARAY(20) OF VARCHAR(50);

    v_lista_angajati tabluu_indexat := tabluu_indexat();
    v_lista_membrilor tabluu_imbricat := tabluu_imbricat();
    v_lista_locatii tabluu_vector := tabluu_vector();

    v_name_sala VARCHAR(50);

  BEGIN
    SELECT nume_sala
    INTO v_name_sala
    FROM sala
    WHERE id_sala = v_id_sala;

    FOR angajat IN (SELECT id_angajat, nume_angajat FROM angajat WHERE id_sala = v_id_sala)
    LOOP
      v_lista_angajati(angajat.id_angajat) := angajat.nume_angajat;
    END LOOP;

    FOR membru IN (SELECT id_membru FROM membru WHERE id_sala = v_id_sala)
    LOOP
      v_lista_membrilor.extend;
      v_lista_membrilor(v_lista_membrilor.last) := membru.id_membru;
    END LOOP;

    FOR locatie IN (SELECT oras_locatie FROM locatie WHERE id_sala = v_id_sala)
    LOOP
      v_lista_locatii.extend;
      v_lista_locatii(v_lista_locatii.last) := locatie.oras_locatie;
    END LOOP;

    DBMS_OUTPUT.PUT_LINE('Informatii despre sala ' || v_name_sala || ':');

    DBMS_OUTPUT.PUT_LINE('Liste angajati:');
    FOR i IN v_lista_angajati.first .. v_lista_angajati.last
    LOOP
      DBMS_OUTPUT.PUT_LINE(' ' || v_lista_angajati(i));
    END LOOP;
  END;
/

```

Script Output | Task completed in 0.03 seconds

Dime Output Compiler - Log Messages Logging Help Characters Compiler | Log (150 Columns) Input Modified Windows

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet Query Builder

```

INTO v_nume_sala
FROM sala
WHERE id_sala = v_id_sala;

FOR angajat IN (SELECT id_angajat, nume_angajat FROM angajat WHERE id_sala = v_id_sala)
LOOP
  v_lista_angajati(v_angajat.id_angajat) := angajat.nume_angajat;
END LOOP;

FOR membru IN (SELECT id_membru FROM membru WHERE id_sala = v_id_sala)
LOOP
  v_lista_membrui.extend;
  v_lista_membrui(v_lista_membrui.last) := membru.id_membru;
END LOOP;

FOR locatie IN (SELECT ora_locatie FROM locatie WHERE id_sala = v_id_sala)
LOOP
  v_lista_locatii.extend;
  v_lista_locatii(v_lista_locatii.last) := locatie.ora_locatie;
END LOOP;

DBMS_OUTPUT.PUT_LINE('Informații despre sală ' || v_nume_sala || ':');

DBMS_OUTPUT.PUT_LINE('Listă angajați:');
FOR i IN v_lista_angajati.first .. v_lista_angajati.last
LOOP
  DBMS_OUTPUT.PUT_LINE(' ' || v_lista_angajati(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Listă membru (id-urile):');
FOR i IN v_lista_membrui.first .. v_lista_membrui.last
LOOP
  DBMS_OUTPUT.PUT_LINE(' ' || v_lista_membrui(i));
END LOOP;

DBMS_OUTPUT.PUT_LINE('Listă locatii (orașele):');
FOR i IN v_lista_locatii.first .. v_lista_locatii.last
LOOP
  DBMS_OUTPUT.PUT_LINE(' ' || v_lista_locatii(i));
END LOOP;

BND afisare_informatii_sala;

PROCEDURE afisare_informatii_vestiare
  (v_id_sala sala.id_sala%TYPE DEFAULT 1)
IS
  v_name_sala sala.name_sala%TYPE;
  v_id_verstilar vestilar.id_vestilar%TYPE;
  v_tip CHAR(1);
  v_capacitate INT;
  v_id_ingritor INGRITOR%TYPE;
  v_name_angajat VARCHAR2(50);

  CURSOR c_vestilar IS
    SELECT id_vestilar, tip_vestilar, capacitate
    FROM vestilar
    WHERE id_sala=v_id_sala;

  CURSOR c_ingritor (vestilar_id vestilar.id_vestilar%TYPE) IS
    SELECT id_ingritor, nume_angajat
    FROM ingridente
    WHERE id_vestilar = vestilar_id AND e.id_angajat=c.id_angajat;

  BEGIN

    SELECT nume_sala
    INTO v_name_sala
    FROM sala
    WHERE id_sala = v_id_sala;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE('Informații despre vestiarele salii ' || v_name_sala || ':');
    DBMS_OUTPUT.PUT_LINE('-----');

    OPEN c_vestilar;

    LOOP
      FETCH c_vestilar INTO v_id_vestilar, v_tip, v_capacitate;
      EXIT WHEN c_vestilar%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.PUT_LINE('Vestilarul ' || v_id_vestilar || ' de tipul ' || v_tip || ' și cu capacitatea ' || v_capacitate || '.');
      DBMS_OUTPUT.PUT_LINE('Listă ingritorilor:');

      FOR ingritor_i IN c_ingritor(v_id_vestilar) LOOP
        DBMS_OUTPUT.PUT_LINE(ingritor_i.id_ingritor || ' ' || ingritor_i.nume_angajat);
      END LOOP;
    END LOOP;
  END;

```

Script Output

Done Output

Compiler Log

Messages Logging Scripts Subscriptions Compiler

Line 100 Column 41 Input Modified Windows

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet Query Builder

```

PROCEDURE afisare_informatii_vestiare
  (v_id_sala sala.id_sala%TYPE DEFAULT 1)
IS
  v_name_sala sala.name_sala%TYPE;
  v_id_verstilar vestilar.id_vestilar%TYPE;
  v_tip CHAR(1);
  v_capacitate INT;
  v_id_ingritor INGRITOR%TYPE;
  v_name_angajat VARCHAR2(50);

  CURSOR c_vestilar IS
    SELECT id_vestilar, tip_vestilar, capacitate
    FROM vestilar
    WHERE id_sala=v_id_sala;

  CURSOR c_ingritor (vestilar_id vestilar.id_vestilar%TYPE) IS
    SELECT id_ingritor, nume_angajat
    FROM ingridente
    WHERE id_vestilar = vestilar_id AND e.id_angajat=c.id_angajat;

  BEGIN

    SELECT nume_sala
    INTO v_name_sala
    FROM sala
    WHERE id_sala = v_id_sala;

    DBMS_OUTPUT.PUT_LINE('-----');
    DBMS_OUTPUT.PUT_LINE('Informații despre vestiarele salii ' || v_name_sala || ':');
    DBMS_OUTPUT.PUT_LINE('-----');

    OPEN c_vestilar;

    LOOP
      FETCH c_vestilar INTO v_id_vestilar, v_tip, v_capacitate;
      EXIT WHEN c_vestilar%NOTFOUND;
      DBMS_OUTPUT.PUT_LINE(' ');
      DBMS_OUTPUT.PUT_LINE('Vestilarul ' || v_id_vestilar || ' de tipul ' || v_tip || ' și cu capacitatea ' || v_capacitate || '.');
      DBMS_OUTPUT.PUT_LINE('Listă ingritorilor:');

      FOR ingritor_i IN c_ingritor(v_id_vestilar) LOOP
        DBMS_OUTPUT.PUT_LINE(ingritor_i.id_ingritor || ' ' || ingritor_i.nume_angajat);
      END LOOP;
    END LOOP;
  END;

```

Script Output

Done Output

Compiler Log

Messages Logging Scripts Subscriptions Compiler

Line 100 Column 41 Input Modified Windows

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet Query Builder

```

EXIT WHEN c_vestilar NOTFOUND;
DBMS_OUTPUT.PUT_LINE('-----');
DBMS_OUTPUT.PUT_LINE('Vestiarul cu id-ul '|| v_id_vestiar || ' nu exista');
DBMS_OUTPUT.PUT_LINE('Lista ingrijitorilor care au capacitate de '|| v_capacitate || '.');
FOR ingrijitor_i IN c_ingrijitor(v_id_vestiar) LOOP
  DBMS_OUTPUT.PUT_LINE(ingrijitor_i.id_ingrijitor || ' '|| ingrijitor_i.nume_angajat);
END LOOP;
END LOOP;

DBMS_OUTPUT.PUT_LINE('-----');
CLOSE c_vestilar;

END afisare_informatii_vestiare;
```

FUNCTION suma_bani

```

(v_id_sala salas.id_sala%TYPE DEFAULT 1)
RETURNS decimal_precision IS
  v_suma decimal_precision;
  exception1 EXCEPTION;
  exception2 EXCEPTION;
BEGIN
  BEGIN
    SELECT SUM(a.pret)
    INTO v_suma
    FROM achizitie a
    JOIN achizitie_mesebru m ON a.id_mesebru = m.id_mesebru
    JOIN sala s ON s.id_sala = m.id_sala
    WHERE s.id_sala = v_id_sala;

    IF v_suma < 1000 THEN
      RAISE exception1;
    END IF;

    IF v_suma >= 2000 THEN
      RAISE exception2;
    END IF;

  EXCEPTION
    WHEN exception1 THEN
      DBMS_OUTPUT.PUT_LINE('Sala este in faliment!');
    WHEN exception2 THEN
      DBMS_OUTPUT.PUT_LINE('Sala este aproape de faliment!');
  END;
END;
```

Script Output Task completed in 0.03 seconds

Dom Output Compiler Log Messages Logging Scripts Subscriptions Compiler Log (100 Columns) Input Modified Windows

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet Query Builder

```

DBMS_OUTPUT.PUT_LINE('Sala este in faliment!');
WHEN exception1 THEN
  DBMS_OUTPUT.PUT_LINE('Sala este aproape de faliment!');
END;

RETURN v_suma;

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE('Alte eroare!');

END suma_bani;

PROCEDURE afiseaza_antrenorul_anului
(v_id_sala salas.id_sala%TYPE DEFAULT 1)
IS
  v_name_angajat angajat.nume_angajat%TYPE;
BEGIN
  SELECT a1.nume_angajat
  INTO v_name_angajat
  FROM antrenor a1
  JOIN antrenor a2 ON a1.id_angajat = a2.id_angajat
  JOIN program p ON a1.id_angajat = a2.id_angajat
  JOIN foloseste f1 ON p.id_antrenament = f1.id_antrenament
  JOIN sala s ON a1.id_sala = s.id_sala
  WHERE s.id_sala = v_id_sala
  GROUP BY a1.nume_angajat
  HAVING COUNT(*) = 1
  SELECT MAX(cnt)
  FROM (
    SELECT COUNT(*) as cnt
    FROM angajat a3
    GROUP BY a3.id_angajat
    HAVING COUNT(*) > 1 AND a3.id_angajat = a1.id_angajat
    JOIN program p2 ON a3.id_angajat = a2.id_angajat
    JOIN foloseste f2 ON p2.id_antrenament = f1.id_antrenament
    JOIN sala s2 ON a2.id_sala = s2.id_sala
    GROUP BY a2.nume_sala, a3.nume_angajat
  );
  DBMS_OUTPUT.PUT_LINE('Antrenorul anului este '|| v_name_angajat || '.');
EXCEPTION
  WHEN NO_DATA_FOUND THEN
    DBMS_OUTPUT.PUT_LINE('Nu exista antrenori pentru acest sala');
```

Script Output Task completed in 0.03 seconds

Dom Output Compiler Log Messages Logging Scripts Subscriptions Compiler Log (100 Columns) Input Modified Windows

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Workshop Query Builder

```

SELECT MAX(cnt)
FROM (
    SELECT COUNT(*) AS cnt
    FROM angajat a
    JOIN angajat s ON a.id_angajat = s.id_angajat
    WHERE s.id_angajat = v_id_angajat
    GROUP BY s.id_angajat
    HAVING COUNT(*) = (
        SELECT MAX(cnt)
        FROM (
            SELECT COUNT(*) AS cnt
            FROM angajat a
            JOIN angajat s ON a.id_angajat = s.id_angajat
            JOIN proiecte p ON a.id_angajat = s.id_angajat
            JOIN angajat s2 ON p.id_angajat = s2.id_angajat
            JOIN sala s3 ON s2.id_angajat = s3.id_angajat
            GROUP BY s3.nume_sala, s3.nume_angajat
        )
    )

    DBMS_OUTPUT.PUT_LINE('Antrenorul analui este ' || v_nume_angajat || ' ');

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in sala cu id-ul dat.');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Există mai multi antrenori eligibili pentru castigarea titlului. Vă rugăm frumos contactați managerul salii respective.');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alte eroare!');
END afiseaza_antrenorul_anului;
END pacchet;

```

Script Output Task completed in 0.037 seconds

Package PACHECT compiled

Package Body PACHECT compiled

Dime Output

Compiler - Log

All Reports

Al Reports New Reports Data Dictionary Reports Data Modeler Reports OLAP Reports Time/Env Reports User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Workshop Query Builder

```

EXCEPTION
    WHEN NO_DATA_FOUND THEN
        RAISE_APPLICATION_ERROR(-20000, 'Nu există antrenori eligibili pentru castigarea titlului în sălă cu id-ul dat.');
    WHEN TOO_MANY_ROWS THEN
        RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Vă rugăm frumos contactați managerul salii respective.');
    WHEN OTHERS THEN
        RAISE_APPLICATION_ERROR(-20002, 'Alte eroare!');
END afiseaza_antrenorul_anului;
END pacchet;
/
drop package pacchet;

EXECUTE pacchet.afiseaza_informatii_sala();
EXECUTE pacchet.afiseaza_informatii_vestiare();
END;

```

Script Output Task completed in 0.03 seconds

PL/SQL procedure successfully completed.

Dime Output Buffer Size [20000]

MySQLA

Înainte de seara sala World Class:
 Lista membrilor:
 Popescu Ion
 Ionescu Maria
 Georgescu Andrei
 Constantin Tiefu
 Dumitru Radu
 Pop Ana
 Gheorghe Alexandru
 Lista membrilor (id-urile):

Compiler - Log

Messages Logging Page Statements Compiler

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in sala cu id-ul dat.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Vă rugăm să contactați managerul salii respective.');
  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Alte eroare!');
END afisare_antrenorul_anual;

END pachet;
/
drop package pachet;

EXECUTE pachet.afisare_informatii_sala();
EXECUTE pachet.afisare_informatii_vestiare();
DECR
  v_resultat abonament.pret47NPR;
BEGIN
  v_resultat := pachet.suma_bani();
  DBMS_OUTPUT.PUT_LINE(v_resultat);
END;
/

```

Script Output: Task completed in 0.023 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Items Output: Buffer Size [20000]

MySQLA *

```

Informatii despre vestiarele salii World Class:
-----
Vestiarul 1 de tipul M si cu capacitatea 50.
Lista angajatorilor:
 1. Popescu Ion
 2. Georgescu Andrei
 3. Ionescu Maria
-----
Vestiarul 2 de tipul F si cu capacitatea 40.
Lista angajatorilor:
 2. Ionescu Maria

```

Compiler - Log

Messages Logging Page Statements Compiler

J (col 105 Column 47) | Input | Modified | Windows

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections MySQLA

Worksheet - Query Builder

```

EXCEPTION
  WHEN NO_DATA_FOUND THEN
    RAISE_APPLICATION_ERROR(-20000, 'Nu există antrenori eligibili pentru castigarea titlului în sălă cu id-ul dat.');
  WHEN TOO_MANY_ROWS THEN
    RAISE_APPLICATION_ERROR(-20001, 'Există prea multe antrenori eligibili pentru castigarea titlului. Vă rugăm să contactați managerul salii respective.');
  WHEN OTHERS THEN
    RAISE_APPLICATION_ERROR(-20002, 'Alte eroare!');
END afisare_antrenorul_anual;

END pachet;
/
drop package pachet;

EXECUTE pachet.afisare_informatii_sala();
EXECUTE pachet.afisare_informatii_vestiare();
DECR
  v_resultat abonament.pret47NPR;
BEGIN
  v_resultat := pachet.suma_bani();
  DBMS_OUTPUT.PUT_LINE(v_resultat);
END;
/
EXECUTE pachet.afisare_antrenorul_anual();


```

Script Output: Task completed in 0.03 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Items Output: Buffer Size [20000]

MySQLA *

```

-- DECALARE
  v_resultat NUMBER;
  Vestiarul 2 de tipul F si cu capacitatea 40.
  Lista angajatorilor:
  2. Ionescu Maria
  -----
  Săia este în fațament!
  1000

```

Compiler - Log

Messages Logging Page Statements Compiler

J (col 105 Column 2) | Input | Modified | Windows

The screenshot shows the Oracle SQL Developer interface with the following details:

- Connections:** MySQLA
- File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help**
- Worksheet:** Query Builder
- Script Content:**

```
Raise_APPLICATION_ERROR(-20000, 'Nu exista antrenori eligibili pentru castigarea titlului in sala cu id-ul dat.');
WHEN NO_ERROR THEN
  Raise_APPLICATION_ERROR(-20001, 'Există mai mulți antrenori eligibili pentru castigarea titlului. Va rugam frumos contactați managerul salii respective.');
WHEN OTHERS THEN
  Raise_APPLICATION_ERROR(-20002, 'Aラー eroare');
END if;
END afisare_antrenorul_anului;
```

```
END pachet;
/
drop package pachet;

EXECUTE pachet.afisare_informatii_sele();
EXECUTE pachet.afisare_informatii_westiare();
END DECODE
v_rezultat abonent.pretTFP;
BEGIN
  v_rezultat := pachet.numar_bani();
  DBMS_OUTPUT.PUT_LINE(v_rezultat);
END;
/
EXECUTE pachet.afisare_antrenorul_anului();
```
- Script Output:**
 - Task completed in 0.028 seconds
 - PL/SQL procedure successfully completed.
 - PL/SQL procedure successfully completed.
- Compiler-Log:**

```
None Output
```

MySQLA [1]
verificam daca exista un utilizator care a raspuns pozitiv.
Listam ingrijitorii:
1. Ionescu Maria

Sala este în felină!
1000
Antrenorul anului este Dumitru Radu.
- Messages:** Click on identifier with the Control key down to perform "Go to Declaration".
- Log:** Line 363 Column 45 | Insert | Modified | Versions

14. Manager-ul dorește implementarea pachetului "pachet_premium" pentru a defini și a implementa funcționalități specifice pentru gestionarea centrului de fitness. Acesta conține următoarele tipuri de date complexe:

- **informatii_membru**: Un tip de înregistrare (RECORD) care stochează informații despre un membru al clubului de fitness, inclusiv id-ul membrului, numele său, tipul sau de abonament și prețul abonamentului.

- **lista_nume**: Un tip de vector (VARRAY) care poate stoca până la 10 nume de membri.

- **informatii_antrenor**: Un tip de înregistrare (RECORD) care păstrează informații despre un antrenor, inclusiv id-ul antrenorului, numele său, specializarea să, id-ul clasei la care predă și o listă de membri ai clasei (numele acestora).

- lista_clase: Un tip de vector (VARRAY) care poate stoca până la 100 de id-uri de clase.

Functiile definite în pachet sunt:

- `numar_clase_membru`: O funcție care primește id-ul unui membru și returnează numărul de clase la care acesta participă.

- `data_minima_clase`: O funcție care primește o listă de id-uri de clase și returnează datea celei mai apropiate clase în viitor din această listă. Funcția este utilă pentru ca antrenorii să știe când este următoarea lor clasa.

Procedurile definite în pachet sunt:

- afisare_informatii_membru: O procedură care primește un parametru de tip "informatii_membru" și afișează informațiile membrului.
- afisare_informatii_antrenor: O procedură care primește un parametru de tip "informatii_antrenor" și afișează informațiile antrenorului.

```
CREATE OR REPLACE PACKAGE pachet_premium AS
```

```
    TYPE informatii_membru IS RECORD (
        id_membru membru.id_membru%TYPE,
        nume_membru membru.nume%TYPE,
        tip_abonament abonament.tip_abonament%TYPE,
        pret_abonament abonament.pret%TYPE
    );
```

```
    TYPE lista_nume IS VARRAY(10) OF membru.nume%TYPE;
```

```
    TYPE informatii_antrenor IS RECORD (
        id_antrenor angajat.id_angajat%TYPE,
        nume_antrenor angajat.nume_angajat%TYPE,
        specializare_antrenor antrenor.specializare%TYPE,
        id_clasa clasa.id_clasa%TYPE,
        lista_membrii lista_nume
    );
```

```
    FUNCTION numar_clase_membru (v_id_membru membru.id_membru%TYPE) RETURN NUMBER;
```

```
    TYPE lista_clase IS VARRAY(100) OF clasa.id_clasa%TYPE;
```

```
    FUNCTION data_minima_clase (v_lista_clase lista_clase) RETURN DATE;
```

```
    PROCEDURE afisare_informatii_membru(v_informatii_membru informatii_membru);
```

```
PROCEDURE afisare_informatii_antrenor(v_informatii_antrenor informatii_antrenor);
```

```
END pachet_premium;
```

```
/
```

```
CREATE OR REPLACE PACKAGE BODY pachet_premium AS
```

```
FUNCTION numar_clase_membru (v_id_membru membru.id_membru%TYPE) RETURN NUMBER IS
    v_numar_clase NUMBER;
BEGIN
    SELECT COUNT(*)
    INTO v_numar_clase
    FROM FORMULAR f
    WHERE f.id_membru=v_id_membru;
    RETURN v_numar_clase;
END numar_clase_membru;
```

```
FUNCTION data_minima_clase (v_lista_clase lista_clase) RETURN DATE IS
    v_data_minima DATE;
BEGIN
    SELECT MIN(data)
    INTO v_data_minima
    FROM clasa c
    WHERE c.id_clasa IN (SELECT * FROM TABLE(v_lista_clase));
    RETURN v_data_minima;
END data_minima_clase;
```

```
PROCEDURE afisare_informatii_membru(v_informatii_membru informatii_membru) IS
```

```
BEGIN  
    DBMS_OUTPUT.PUT_LINE('ID Membru: ' || v_informatii_membru.id_membru);  
    DBMS_OUTPUT.PUT_LINE('Nume Membru: ' || v_informatii_membru.nume_membru);  
    DBMS_OUTPUT.PUT_LINE('Tip Abonament: ' || v_informatii_membru.tip_abonament);  
    DBMS_OUTPUT.PUT_LINE('Pret Abonament: ' || v_informatii_membru.pret_abonament);  
END afisare_informatii_membru;
```

```
PROCEDURE afisare_informatii_antrenor(v_informatii_antrenor informatii_antrenor) IS  
BEGIN  
    DBMS_OUTPUT.PUT_LINE('ID Antrenor: ' || v_informatii_antrenor.id_antrenor);  
    DBMS_OUTPUT.PUT_LINE('Nume Antrenor: ' || v_informatii_antrenor.nume_antrenor);  
    DBMS_OUTPUT.PUT_LINE('Specializare Antrenor: ' || v_informatii_antrenor.specializare_antrenor);  
    DBMS_OUTPUT.PUT_LINE('ID Clasa: ' || v_informatii_antrenor.id_clasa);  
    DBMS_OUTPUT.PUT_LINE('Lista Membrii:');  
  
    FOR i IN 1..v_informatii_antrenor.lista_membrii.COUNT LOOP  
        DBMS_OUTPUT.PUT_LINE(' ' || v_informatii_antrenor.lista_membrii(i));  
    END LOOP;  
END afisare_informatii_antrenor;
```

```
END pachet_premium;  
/
```

```
drop package pachet_premium;
```

```
DECLARE  
    v_id_membru membru.id_membru%TYPE := 3;  
    v_numar_clase NUMBER;
```

```
BEGIN  
    v_numar_clase := pachet_premium.numar_clase_membru(v_id_membru);  
    DBMS_OUTPUT.PUT_LINE('Numărul de clase la care participă membrul respectiv: ' || v_numar_clase);  
END;
```

```
DECLARE  
    v_lista_clase pachet_premium.lista_clase;  
    v_data_minima DATE;  
BEGIN  
    v_lista_clase := pachet_premium.lista_clase(1, 2, 3);  
    v_data_minima := pachet_premium.data_minima_clase(v_lista_clase);  
    DBMS_OUTPUT.PUT_LINE('Data cea mai apropiată dintre clasele respective: ' ||  
        TO_CHAR(v_data_minima, 'DD-MON-YYYY'));  
END;
```

```
DECLARE  
    v_informatii_membru pachet_premium.informatii_membru;  
BEGIN  
    v_informatii_membru.id_membru := 1;  
    v_informatii_membru.nume_membru := 'Popescu Ion';  
    v_informatii_membru.tip_abonament := 'Abonament Bronze';  
    v_informatii_membru.pret_abonament := 100;  
  
    pachet_premium.afisare_informatii_membru(v_informatii_membru);  
END;
```

```
DECLARE  
    v_informatii_antrenor pachet_premium.informatii_antrenor;  
BEGIN
```

```
v_informatii_antrenor.id_antrenor := 4;  
v_informatii_antrenor.nume_antrenor := 'Constantin Elena';  
v_informatii_antrenor.specializare_antrenor := 'Fitness';  
v_informatii_antrenor.id_clasa := 1;  
v_informatii_antrenor.lista_membrii := pachet_premium.lista_nume('Popescu Ion', 'Radu Maria',  
'Ionescu Andrei', 'Georgescu Ana', 'Stanescu Mihaela');  
  
pachet_premium.afisare_informatii_antrenor(v_informatii_antrenor);  
END;
```

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

- Tables (Filtered)
- Views
- Sequences
- Triggers
- Procedures
- Functions
- Operators
- Types
- Queues Tables
- Triggers
- Types
- Sequences
- Materialized Views
- Materialized View Logs
- Synonyms
- Public Synonyms
- Partitions
- Public Database Links
- Directories
- Jobs
- XML Schemas
- XML DB Repository
- OBAP Options

Reports All Reports Data Ware Reports Data Dictionary Reports Data Modeler Reports OLAP Reports Time/Env Reports User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

CREATE OR REPLACE PACKAGE pacchet_premium AS

TYPE informatii_membru IS RECORD

 id_membru membru.id_memberTYPE;

 nume_membru membru.nameTYPE;

 tip_abonament abonent.tip_abonamentTYPE;

 pret_abonament abonent.pretTYPE;

END pacchet_premium;

TYPE lista_nume IS VARARRAY(10) OF membru.nameTYPE;

TYPE informatii_antrenor IS RECORD

 id_antrenor antrenor.id_antrenorTYPE;

 nume_antrenor antrenor.name_antrenorTYPE;

 specializare_antrenor antrenor.specializareTYPE;

 id_clasa classe.id_claseTYPE;

 lista_membru lista_membru;

FUNCTION numar_clase_membru (v_id_membru membru.id_memberTYPE) RETURN NUMBER;

TYPE lista_clase IS VARARRAY(100) OF classe.id_claseTYPE;

FUNCTION data_minima_clase (v_lista_clase lista_clase) RETURN DATE;

PROCEDURE afisare_informatii_membru(v_informatii_membru informatii_membru);

PROCEDURE afisare_informatii_antrenor(v_informatii_antrenor informatii_antrenor);

END pacchet_premium;

CREATE OR REPLACE PACKAGE BODY pacchet_premium AS

FUNCTION numar_clase_membru (v_id_membru membru.id_memberTYPE) RETURN NUMBER IS

 v_numar_clase NUMBER;

BEGIN

 SELECT COUNT(*)

 INTO v_numar_clase

 FROM FORUMTABLE

 WHERE 1=1 AND v_id_membru=id_membru;

 RETURN v_numar_clase;

END numar_clase_membru;

FUNCTION data_minima_clase (v_lista_clase lista_clase) RETURN DATE IS

 v_data_minima DATE;

BEGIN

 SELECT MIN(data)

 INTO v_data_minima

 FROM classe

 WHERE id_clase IN (SELECT * FROM TABLE(v_lista_clase));

 RETURN v_data_minima;

END data_minima_clase;

PROCEDURE afisare_informatii_membru(v_informatii_membru informatii_membru) IS

BEGIN

 DBMS_OUTPUT.PUT_LINE('ID Membru ' || v_informatii_membru.id_membru);

 DBMS_OUTPUT.PUT_LINE('Nume Membru ' || v_informatii_membru.name_membru);

 DBMS_OUTPUT.PUT_LINE('Tip Abonament: ' || v_informatii_membru.tip_abonament);

 DBMS_OUTPUT.PUT_LINE('Pret Abonament: ' || v_informatii_membru.pret_abonament);

 END afisare_informatii_membru;

PROCEDURE afisare_informatii_antrenor(v_informatii_antrenor informatii_antrenor) IS

BEGIN

 FOR i IN 1..v_informatii_antrenor.lista_membru.COUNT LOOP

 DBMS_OUTPUT.PUT_LINE('ID Antrenor ' || v_informatii_antrenor.id_antrenor);

 DBMS_OUTPUT.PUT_LINE('Nume Antrenor ' || v_informatii_antrenor.name_antrenor);

 DBMS_OUTPUT.PUT_LINE('Specializare Antrenor: ' || v_informatii_antrenor.specializare_antrenor);

 DBMS_OUTPUT.PUT_LINE('ID Clasa: ' || v_informatii_antrenor.id_clase);

 DBMS_OUTPUT.PUT_LINE('Lista Membru:');

 END LOOP;

 END afisare_informatii_antrenor;

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

- Tables (Filtered)
- Views
- Sequences
- Triggers
- Procedures
- Functions
- Operators
- Types
- Queues Tables
- Triggers
- Types
- Sequences
- Materialized Views
- Materialized View Logs
- Synonyms
- Public Synonyms
- Partitions
- Public Database Links
- Directories
- Jobs
- XML Schemas
- XML DB Repository
- OBAP Options

Reports All Reports Data Ware Reports Data Dictionary Reports Data Modeler Reports OLAP Reports Time/Env Reports User Defined Reports

Click on an identifier with the Control key down to perform "Go to Declaration"

/

CREATE OR REPLACE PACKAGE BODY pacchet_premium AS

FUNCTION numar_clase_membru (v_id_membru membru.id_memberTYPE) RETURN NUMBER IS

 v_numar_clase NUMBER;

BEGIN

 SELECT COUNT(*)

 INTO v_numar_clase

 FROM FORUMTABLE

 WHERE 1=1 AND v_id_membru=id_membru;

 RETURN v_numar_clase;

END numar_clase_membru;

FUNCTION data_minima_clase (v_lista_clase lista_clase) RETURN DATE IS

 v_data_minima DATE;

BEGIN

 SELECT MIN(data)

 INTO v_data_minima

 FROM classe

 WHERE id_clase IN (SELECT * FROM TABLE(v_lista_clase));

 RETURN v_data_minima;

END data_minima_clase;

PROCEDURE afisare_informatii_membru(v_informatii_membru informatii_membru) IS

BEGIN

 DBMS_OUTPUT.PUT_LINE('ID Membru ' || v_informatii_membru.id_membru);

 DBMS_OUTPUT.PUT_LINE('Nume Membru ' || v_informatii_membru.name_membru);

 DBMS_OUTPUT.PUT_LINE('Tip Abonament: ' || v_informatii_membru.tip_abonament);

 DBMS_OUTPUT.PUT_LINE('Pret Abonament: ' || v_informatii_membru.pret_abonament);

 END afisare_informatii_membru;

PROCEDURE afisare_informatii_antrenor(v_informatii_antrenor informatii_antrenor) IS

BEGIN

 FOR i IN 1..v_informatii_antrenor.lista_membru.COUNT LOOP

 DBMS_OUTPUT.PUT_LINE('ID Antrenor ' || v_informatii_antrenor.id_antrenor);

 DBMS_OUTPUT.PUT_LINE('Nume Antrenor ' || v_informatii_antrenor.name_antrenor);

 DBMS_OUTPUT.PUT_LINE('Specializare Antrenor: ' || v_informatii_antrenor.specializare_antrenor);

 DBMS_OUTPUT.PUT_LINE('ID Clasa: ' || v_informatii_antrenor.id_clase);

 DBMS_OUTPUT.PUT_LINE('Lista Membru:');

 END LOOP;

 END afisare_informatii_antrenor;

Click on an identifier with the Control key down to perform "Go to Declaration"

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet Query Builder

```

    DBMS_OUTPUT.PUT_LINE('ID Membru: ' || v_informatii_membru.id_membru);
    DBMS_OUTPUT.PUT_LINE('Nume Membru: ' || v_informatii_membru.nume_membru);
    DBMS_OUTPUT.PUT_LINE('Prenume Membru: ' || v_informatii_membru.prenume_membru);
    DBMS_OUTPUT.PUT_LINE('Faza Abonament: ' || v_informatii_membru.prez_abonament);
    END afisare_informatii_membru;

  PROCEDURE afisare_informatii_antrenor(v_informatii_antrenor informatii_antrenor) IS
  BEGIN
    DBMS_OUTPUT.PUT_LINE('ID Antrenor: ' || v_informatii_antrenor.id_antrenor);
    DBMS_OUTPUT.PUT_LINE('Nume Antrenor: ' || v_informatii_antrenor.nume_antrenor);
    DBMS_OUTPUT.PUT_LINE('Specializare Antrenor: ' || v_informatii_antrenor.specializare_antrenor);
    DBMS_OUTPUT.PUT_LINE('ID Clase: ' || v_informatii_antrenor.id_clase);
    DBMS_OUTPUT.PUT_LINE('Lista membrui!');

    FOR i IN 1..v_informatii_antrenor.lista_membrui.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE(' ' || v_informatii_antrenor.lista_membrui(i));
    END LOOP;
    END afisare_informatii_antrenor;

  END pacchet_premium;
  
```

Script Output A | Query Result X

Task completed in 0.059 seconds

Package PACCHET_PPREMIUM compiled

Package Body PACCHET_PPREMIUM compiled

Done Output Compiler - Log Messages Logging Page Statements Compiler

Click on an identifier with the Control key down to perform 'Go to Declaration'

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA

Worksheet Query Builder

```

    DBMS_OUTPUT.PUT_LINE('Lista Membrui:');
    FOR i IN 1..v_informatii_antrenor.lista_membrui.COUNT LOOP
      DBMS_OUTPUT.PUT_LINE(' ' || v_informatii_antrenor.lista_membrui(i));
    END LOOP;
    END afisare_informatii_antrenor;

  END pacchet_premium;
  /
drop package pacchet_premium;

  DECLARE
    v_id_membru membru.id_membru%TYPE := 3;
    v_nrclasse_clase NUMBER;
  BEGIN
    v_nrclasse := pacchet_premium.numar_clase_membru(v_id_membru);
    DBMS_OUTPUT.PUT_LINE('Numarul de clase la care participă membrul respectiv: ' || v_nrclasse);
  END;

  DECLARE
    v_lista_clase pacchet_premium.lista_clase;
    v_data_mijloace DATE;
  BEGIN
    
```

Script Output A | Query Result X

Task completed in 0.025 seconds

Package PACCHET_PPREMIUM compiled

Package Body PACCHET_PPREMIUM compiled

PL/SQL procedure successfully completed.

Done Output Compiler - Log

MySQLA x Buffer Size[20000]

Numarul de clase la care participă membrul respectiv: 2

Messages Logging Page Statements Compiler

Click on an identifier with the Control key down to perform 'Go to Declaration'

The screenshot shows the Oracle SQL Developer interface with the following details:

- File Bar:** File, Edit, View, Navigate, Run, Source, Team, Tools, Window, Help.
- Connections:** MySQLA
- Toolbars:** Standard toolbar with icons for New, Open, Save, Print, Copy, Paste, Find, Replace, Undo, Redo, Cut, Copy, Paste, Find, Replace, Undo, Redo, and a Help icon.
- Connections Tab:** Shows connections to MySQLA, MySQLA1, MySQL/A2, and Welcome Page. The MySQLA connection is selected.
- Worksheet:** Query Builder. The code is a PL/SQL procedure:

```
DELIMITER //
CREATE PROCEDURE sp_get_member_clase(v_id_membru INT, v_nr_clase NUMBER)
BEGIN
    v_nr_clase := packet_premium.numer_clase_membru(v_id_membru);
    SELECT OUTOUT.vt_nr_clase, 'Numarul de clase la care participă membrul respectiv: ' || v_nr_clase;
END;

DELIMITER //
CREATE PROCEDURE sp_get_minima_clase(v_id_membru INT)
BEGIN
    v_nr_clase := packet_premium.lista_clase(1, 2, 3);
    v_data_minima := packet_premium.data_minima_clase(v_nr_clase);
    SELECT OUTOUT.vt_nr_clase, 'Data cea mai apropiată dintre clasele respective: ' || TO_CHAR(v_data_minima, 'DD-MON-YYYY');
END;

DELIMITER //
CREATE PROCEDURE sp_informatii_membru(v_id_membru INT)
BEGIN
    v_nr_clase := packet_premium.informatii_membru(v_id_membru);
    SELECT OUTOUT.vt_nr_clase;
END;

```
- Script Output:** Task completed in 0.032 seconds.
 - PL/SQL procedure successfully completed.
 - PL/SQL procedure successfully completed.
- Logs:** Shows the output of the executed procedures:

```
Numarul de clase la care participă membrul respectiv: 2
Data cea mai apropiată dintre clasele respective: 18-MAY-2023
```
- Compiler Log:** Displays compiler messages.

The screenshot shows the Oracle SQL Developer interface with the following details:

- Connections:** Oracle Connections, MySALA
- File menu:** File, Edit, View, Navigate, Run Source, Team, Tools, Window, Help
- Connections pane:** Oracle Connections, MySALA, MySALA (selected), MySALA1.sql, MySALA2.sql, Welcome Page, MySALA
- Worksheet:** Query Builder, displaying a PL/SQL block:

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

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

```
PL/SQL procedure successfully completed.
```
- Script Output:** A tab labeled "Query Result" shows the output of the executed PL/SQL block.
- Reports:** All Reports, Data Delivery Reports, Data Monitor Reports, Time/Tim Reports, User Defined Reports
- Compiler - Log:** Shows the log for the current session.
- Bottom status bar:** Click on an identifier with the Control key down to perform "Go to Declaration". Line 97 Column 3 | Insert | Modified | Windows

Oracle SQL Developer : MySQLA

File Edit View Navigate Run Source Team Tools Window Help

Connections Oracle Connections MySQLA Tables (Filtered) Views Synonyms Packages Functions Operators Jobs Queues Tables Triggers Types Materialized Views Materialized View Log Synonyms Public Synonyms Procedures Directories Jobs XML DB Schemas XML DB Repository MySQLA

Worksheet Query Builder

```

DECLARE
    v_informatii_membru.pachet_premium.informatii_membru;
BEGIN
    v_informatii_membru.id_membru := 1;
    v_informatii_membru.nume_membru := 'Popescu Ion';
    v_informatii_membru.tip_abonament := 'Abonament Bronze';
    v_informatii_membru.pret_abonament := 100;

    pachet_premium.afisare_informatii_membru(v_informatii_membru);
END;

DECLARE
    v_informatii_antrenor.pachet_premium.informatii_antrenor;
BEGIN
    v_informatii_antrenor.id_antrenor := 4;
    v_informatii_antrenor.nume_antrenor := 'Constantin Elena';
    v_informatii_antrenor.specialitate_antrenor := 'Fitness';
    v_informatii_antrenor.id_clasa := 1;
    v_informatii_antrenor.lista_membril := pachet_premium.lista_nume('Popescu Ion', 'Radu Maria', 'Ionescu Andrei', 'Georgescu Ana', 'Stanescu Mihaila');

    pachet_premium.afisare_informatii_antrenor(v_informatii_antrenor);
END;

```

Script Output * | Query Result * Task completed in 0.03 seconds PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

DBMS Output * Buffer Size: 20000 MySQLA

```

Name Membru: Popescu Ion
Tip Abonament: Abonament Bronze
Pret Abonament: 100

ID Antrenor: 4
Nume Antrenor: Constantin Elena
Specialitate Antrenor: Fitness
ID Clasa: 1
Lista Membril:
Popescu Ion
Radu Maria
Ionescu Andrei
Georgescu Ana
Stanescu Mihaila

```

Compiler - Log

Messages Logging Page Statements Compiler

Line 120 Column 5 Input Next Window