

# **Proiect final Baze de date : Firma de Traininguri**

Popescu Ioana-Livia, grupa 152

## **Model real:**

O firma de traininguri pune la dispozitie companiilor cateva programe dintr-o editie limitata. Fiecare program contine o serie specifica de traininguri care nu se regasesc in alte programe. Fiecare training este tinut de un trainer, intr-o sala de la o anumita locatie. Angajatii pot participa la un training in functie de pozitie in cadrul companiei, spre a se evita efectuarea unui training de doua ori. Pentru fiecare trainer se retin numele firmei la care lucreaza si pozitia. Pentru a putea beneficia de un program, se realizeaza un contract in urma caruia se va efectua si plata. Se va realiza o fisa de inscriere pentru fiecare participant, careia i se va atasa si un formular de feedback.

## **Constrangeri si restrictii:**

Un trainer poate sustine un singur tip de training si poate fi angajat la o singura firma, fiind cautati oameni din alte companii cu mai multe experienta ce pot oferi sfaturi practice bazate pe proiecte aflate in desfasurare.

Programele scoase la licitatie sunt unice, astfel ca fiecare poate fi achizitionat o singura data, de catre un singur client, prin intermediul unui contract unic.

Nu se pot sustine mai mult de 2 traininguri la o anumita data.

Pentru fiecare program se va face un contract separat.

Fiecare training se tine la o locatie specifica.

## **Descrierea Entitatilor:**

Clientul reprezinta una din entitatile principale, determinand firma care doreste sa achizitioneze unul din programele scoase la licitatie. Cheia primara consta intr-un id unic, ID\_CLIENT.

Contractul, identificat printr-un id unic, are rolul de a conecta datele clientului cu cele ale programului ales. Cheia primara consta intr-un id unic, ID\_CONTRACT.

Prin intermediul contractului se face legatura cu entitatea Tranzactie care contine detaliile despre tranzactia efectuata. Cheia primara consta intr-un id unic, ID\_TRANZACTIE.

Entitatea Program retine datele aferente unui program. Cheia primara consta intr-un id unic, ID\_PROGRAM.

Entitatea Fisa inscriere conecteaza entitatile Program, Participant si Training. Cheia primara este ID\_FISA.

Entitatea Participant retine detaliile necesare despre participant, facilitand identificarea angajatilor care vor lua parte la traininguri. Cheia primara consta intr-un id unic, ID\_PARTICIPANT.

Entitatea Training retine detaliile importante despre un training. Cheia primara consta intr-un id unic, ID\_TRAINING.

Este asociata cu entitatea Trainer care retine detaliile necesare identificarii unui trainer. Cheia primara consta intr-un id unic, ID\_TRAINER.

In plus, entitatea Job ofera mai multe informatii despre trainer, respectiv rolul in firma de care apartine si anii de experienta. Cheia primara consta intr-un id unic, ID\_JOB.

Entitatea Data retine data la care se desfasoara un training si daca este valabila. Cheia primara este de tip date, DATA\_SUSTINERII.

Asignare\_Data este necesara tabelului asociativ generat in urma spargerii relatiei M:M dintre Training si Data. Cheia primara este compusa din 2 Foreign Keys, ID\_TRAINING si DATA\_SUSTINERII.

Entitatea Locatie retine detaliile despre locatia la care se va desfasura trainingul. Cheia primara consta intr-un id unic, ID\_LOCATIE.

Entitatea Sala colectează datele despre sala în care se va desfășura trainingul. Cheia primară constă într-un număr unic de identificare, COD\_SALA.

Feedback-ul este strict legat de fișa de înscriere, conținând parerile și nota oferită pe total experienței. Cheia secundară este, așadar, ID\_FISA, totodată și cheie străină.

### **Descrierea Relatiilor + Cardinalitate:**

Un client poate semna zero sau mai multe contracte, dar un contract poate fi semnat de un singur client. (1:M)

Un contract trebuie să fie însoțit de o unică tranzacție, iar o tranzacție poate corespunde unui unic contract. (1:1)

Un contract include unul sau mai multe programe, iar un program poate fi inclus într-un singur contract. (1:M)

Un program prevede una sau mai multe fișe de înscriere, iar o fișă de înscriere corespunde unui singur program. (1:M)

Un participant completează una sau mai multe fișe de înscriere, iar o fișă de înscriere este completată de un singur participant. (1:M)

O fișă de înscriere are asociat un singur formular de feedback, iar unui formular de feedback îi este asociată o singură fișă de înscriere. (1:1)

Pentru un training se înregistrează zero sau mai multe fișe de înscriere. (1:M)

Un training poate fi susținut la mai multe date, iar pe o dată se pot susține mai multe traininguri. (M:M) Am transformat relația prin intermediul tabelului asociativ ASIGNARE\_DATA. Astfel, pentru un training se alocă una sau mai multe asignări de date (1:M), iar pentru o dată corespund una sau mai multe asignări de date. (1:M) Vor fi trecute în baza de date doar datele la care se țin trainingurile.

Un training se desfășoară la o locație unică, iar la o locație se pot desfășura unul sau mai multe traininguri. (1:M)

O sală are o singură locație, iar o locație are una sau mai multe săli. (1:M)

Un training este tinut de trainer, iar un trainer poate tine unul sau mai multe traininguri (1:M).

Un trainer are un singur job, iar un job poate fi realizat de unul sau mai multi traineri. (1:M)

Un training apartine unui Un program ofera unul sau mai multe traininguri, iar un training apartine unui program specific. (1:M)

### **Descrierea Atributelor** (tip de date, constrangeri, valori implicite, valori posibile):

Atribute precum „NUME\_FIRMA”, „NUME\_PROGRAM”, „NUME\_FIRMA\_JOB”, „NUME\_TRAINER”, „NUME\_POZITIE\_FIRMA”, „NUME\_POZITIE\_JOB” si „PRENUME\_PARTICIPANT”, „PRENUME\_TRAINER” definesc numele, respectiv prenumele retinut pentru o anumita entitate, sunt de tip VARCHAR2 si nu pot depasi 30 de caractere. (VARCHAR2(30))

Valori posibile:

NUME\_FIRMA : „Mario et Company”

NUME\_PARTICIPANT: „Popescu”

PRENUME\_TRAINER : „Anca”

NUME\_POZITIE: „Manager”

NUME\_FIRMA\_JOB: „IT Engineer”

Atributele de forma „ ID\_x”, unde x adesea corespunde denumirii entitatii (ID\_CLIENT, ID\_PARTICIPANT, ID\_TRANZACTIE, ID\_CONTRACT, ID\_LOCATIE, ID\_JOB, ID\_TRAINER, ID\_FISA) sunt de tipul number. Nu pot lua valoarea null.

Valori posibile:

ID\_CLIENT: 438789

ID\_FISA: 20

ID\_LOCATIE: 212

Atributele de forma „DATA\_x”, unde x adesea corespunde denumirii entitatii („DATA\_CONTRACT”, „DATA\_TRANZACTIEI”, „DATA\_SUSTINERII” - se refera la data la care va fi sustinut trainingul) sunt de tip date. Valori posibile:

DATA\_CONTRACT: 2021-06-15

DATA\_TRANZACTIEI: 2021-05-03

DATA\_SUSTINERII: 2021-08-20

### CLIENT

Atributul CIF reprezinta un cod de identificare primit de catre firma in momentul inregistrarii la Registrul Comertului. Va fi retinut in format varchar2 si nu poate avea mai mult de 12 caractere. Valori posibile:

CIF : RO36289

CIF: RO7392831

### TRANZACTIE

Atributul „SUMA\_INCASATA” determina suma incasa de firma (in RON ) de traininguri de la client in urma semnarii contractului. Este de tip number. Valori posibile:

SUMA\_INCASATA: 2345

SUMA\_INCASATA: 10000

Atributul „TVA” reda suma perceputa drept TVA in urma efectuarii tranzactiei . Valori posibile:

TVA: 45

TVA: 720

## PROGRAM

Atributul „PRET” se refera la pretul de piata al programului, respectiv cel pe care il va achita clientul si reprezinta suma in RON. Este de tip number, implicit nu poate depasi valoarea de 2,147,483,647 . Valori posibile:

PRET: 8379

PRET: 10000

## TRAINING

Atributul „DURATA” reprezinta numarul de ore al trainingului, fiind de tip number  
Valori posibile:

DURATA: 1

DURATA: 2

DURATA: 8

Atributul „NIVEL\_COMPLEXITATE” este de tip number, poate lua valori de la 1 la 3, in functie de cat de complex este considerat trainingul. Valori posibile:

NIVEL\_COMPLEXITATE: 2

NIVEL\_COMPLEXITATE: 3

## DATA

Atributul „LIBER\_DATA” este de tip number, insa va actiona ca o variabila booleana: 1 daca data este libera, adica se mai poate tine un training in acea zi, ori 0 in caz contrar. Valori posibile:

LIBER\_DATA: 1

LIBER\_DATA: 0

### LOCATIE

Atributul „PRET\_PER\_SALA” se refera la pretul inchirierii unei sali, in RON. Valori posibile:

PRET\_PER\_SALA: 1200

PRET\_PER\_SALA: 3500

Atributul „ADRESA” este de tip varchar si va contine adresa locatiei. Nu poate depasi 2000 de caractere. Valori posibile:

ADRESA: „Bucharest, Romania, Strada Lalelor, nr. 9”

### SALA

Atributul „COD\_SALA” functioneaza dupa aceleasi principii ca cele de tipul „ID\_x”, in scopul identificarii unice.

Valori posibile:

COD\_SALA: 2

COD\_SALA: 3

Atributul ARIA este de tip number exprima suprafata salii in metrii patrati. Valori posibile:

ARIA : 100

ARIA : 47

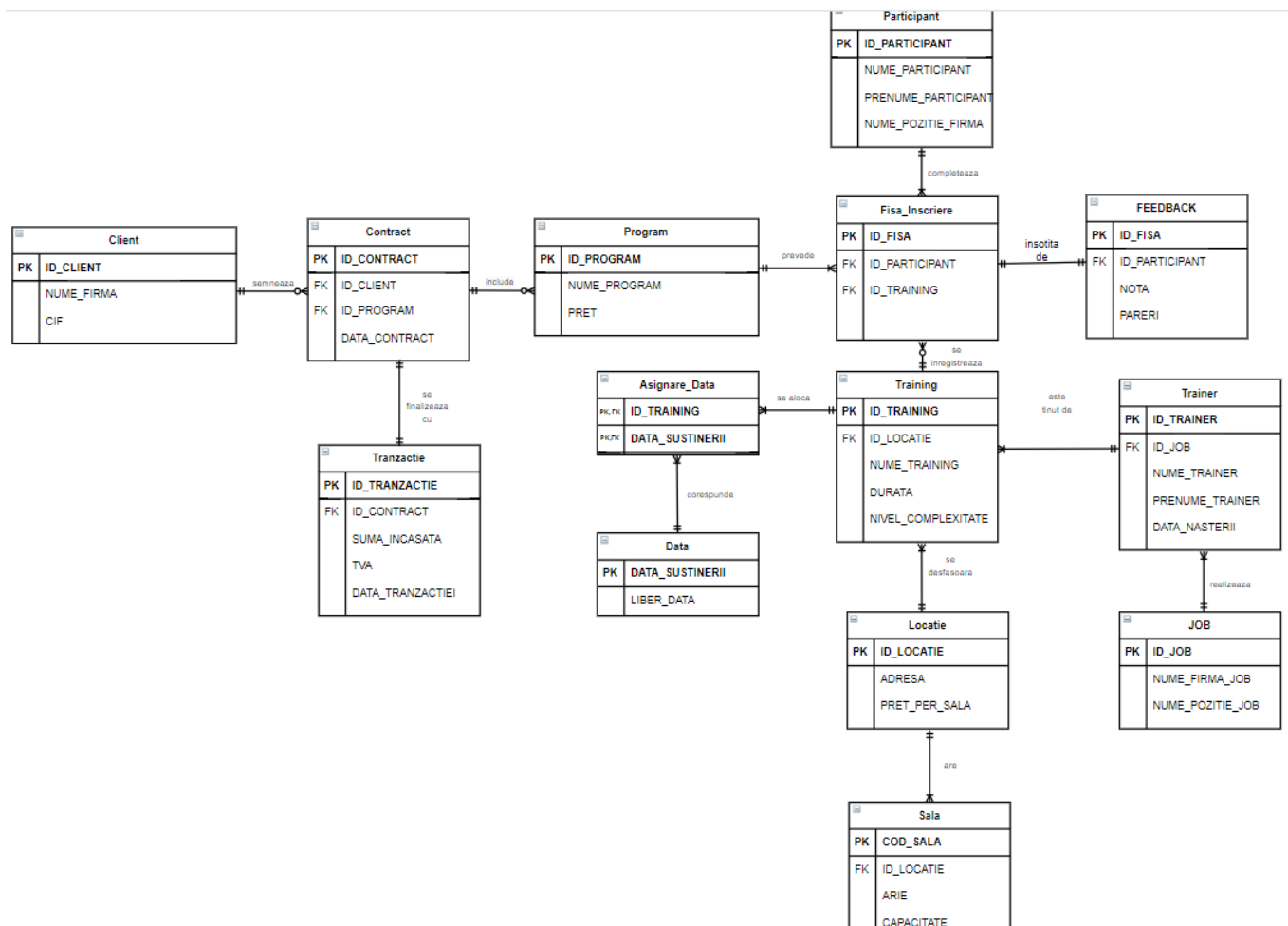
Atributul CAPACITATE este tip number se refera la numarul maxim de persoane dintr-o sala. Valori posibile:

CAPACITATE: 156

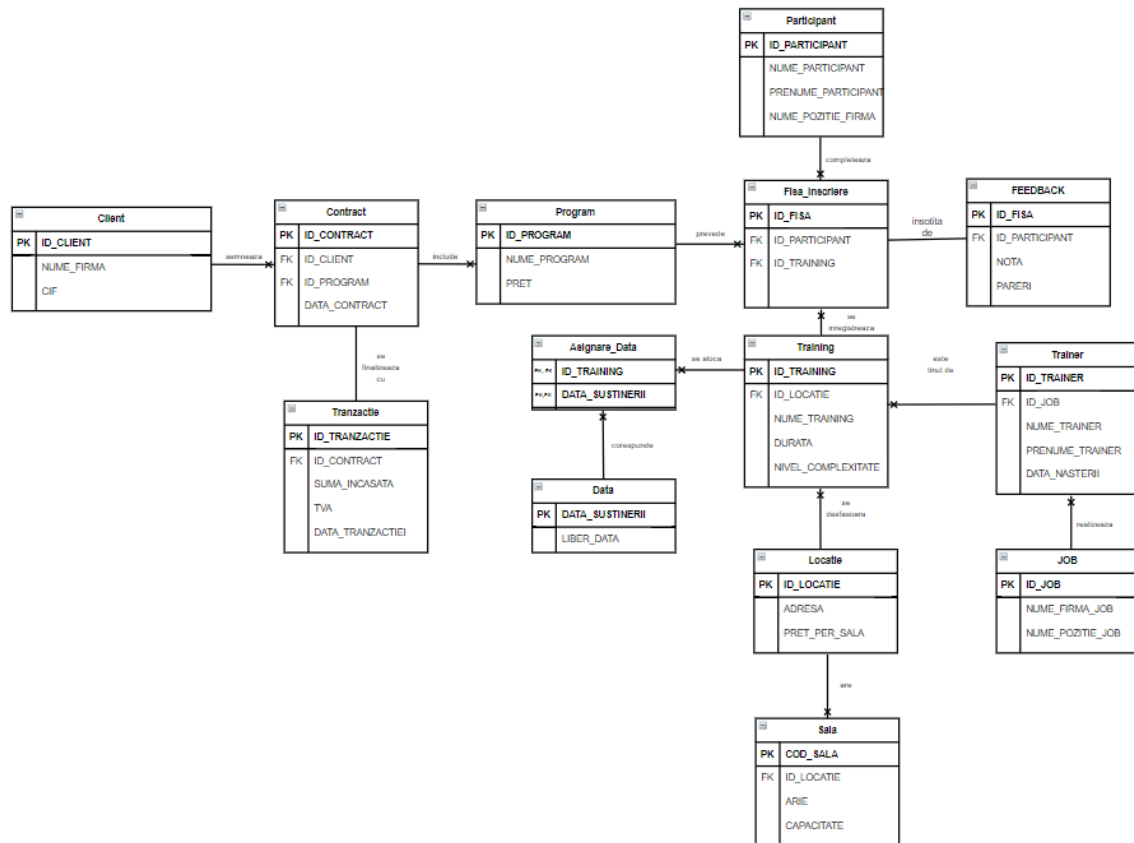
CAPACITATE: 100

**ERD:**





**DIAGRAMA CONCEPTUALA:**



## Scheme relationale:

Client(#ID\_CLIENT, NUME\_FIRMA, CIF)

Contract(#ID\_CONTRACT, ID\_CLIENT (FK) , ID\_PROGRAM (FK) , DATA\_CONTRACT)

Tranzactie(#ID\_TRANZACTIE, ID\_CONTRACT (FK), SUMA\_INCASATA, TVA, DATA\_TRANZACTIEI)

Program(#ID\_PROGRAM, NUME\_PROGRAM, PRET)

Fisa\_inscriere(#ID\_FISA, ID\_PARTICIPANT(FK), ID\_TRAINING(FK))

Participant(#ID\_PARTICIPANT, NUME\_PARTICIPANT, PRENUME\_PARTICIPANT, NUME\_POZITIE\_FIRMA)

Training(#ID\_TRAINING, NUME\_TRAINING, DURATA, NIVEL\_COMPLEXITATE)

Asignare\_Data(#ID\_TRAINING, #DATA\_SUSTINERII)

Data(#DATA\_SUSTINERII, LIBER\_DATA)

Locatie(#ID\_LOCATIE, ADRESA, PRET\_PER\_SALA)

Sala(#COD\_SALA, ARIE, CAPACITATE)

Trainer(#ID\_TRAINER, NUME\_TRAINER, PRENUME\_TRAINER)

Job(#ID\_JOB, NUME\_FIRMA\_JOB, NUME\_POZITIE\_JOB,

Feedback(#ID\_FISA, ID\_PARTICIPANT, NOTA, PARERI)

### Normalizare:

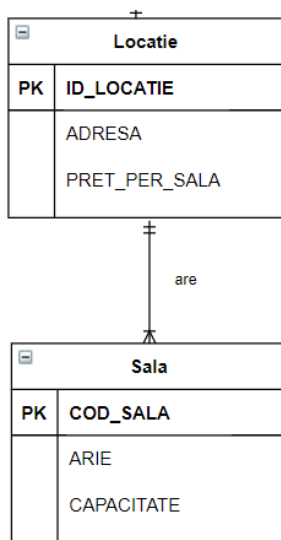
**FN1:** no multi-valued attributes

Exemplu: o locatie poate avea mai multe sali disponibile

Non-FN1:

Locatie	
PK	ID_LOCATIE
	ADRESA
	PRET_PER_SALA
	COD_SALA

FN1:



**FN2:** any non-UID attribute be dependent on the entire UID

Exemplu: Atributul ARIE se refera la aria salii si depinde de SALA, nu de LOCATIE, intrucat o locatie poate avea mai multe sali.

Non-FN2:

Locatie	
PK	ID_LOCATIE
	ADRESA
	PRET_PER_SALA
	ARIE

FN2:

Sala	
PK	COD_SALA
	ARIE
	CAPACITATE

**FN3:** no non-UID attribute can be dependent on another non-UID attribute

Exemplu: attributele nume\_firma\_job, nume\_pozitie\_job, ani\_experienta ( la un job specific) depinde de id\_job

Non-FN3:

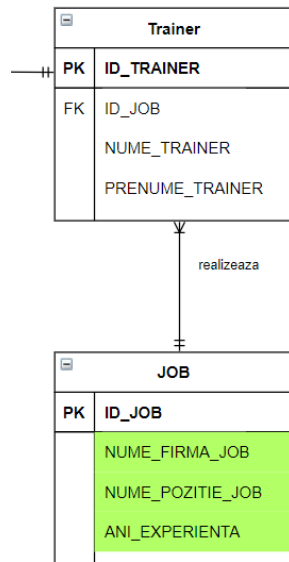
Trainer(ID\_TRAINER, ID\_JOB, NUME\_TRAINER, PRENUME\_TRAINER, NUME\_FIRMA\_JOB, NUME\_POZITIE\_JOB, ANI\_EXPERIENTA)

Trainer	
PK	ID_TRAINER
	ID_JOB
	NUME_TRAINER
	PRENUME_TRAINER
	NUME_FIRMA_JOB
	NUME_POZITIE_JOB
	ANI_EXPERIENTA

FN3:

Trainer(ID\_TRAINER, ID\_JOB, NUME\_TRAINER, PRENUME\_TRAINER)

Job(ID\_JOB, NUME\_FIRMA\_JOB, NUME\_POZITIE\_JOB, ANI\_EXPERIENTA)



10.

```

CREATE TABLE Client(
    id_client number(4) not null,
    nume_firma varchar2(50) not null,
    cif varchar2(50) not null,
    constraint pk_Client primary key (id_client)
);
  
```

```

INSERT INTO Client (id_client, nume_firma, cif)
VALUES (1, 'Thales', 'RO123C');
  
```

```

INSERT INTO Client (id_client, nume_firma, cif)
VALUES (2, 'Apple', 'RO1234C');
  
```

```

INSERT INTO Client (id_client, nume_firma, cif)
VALUES (3, 'Microsoft', 'RO12345C');
  
```

```
INSERT INTO Client (id_client, nume_firma, cif)
VALUES (4, 'Endava', 'RO123456C');
```

```
INSERT INTO Client (id_client, nume_firma, cif)
VALUES (5, 'Adobe', 'RO1234567C');
```

```
select * from Client;
```

```
CREATE TABLE Program(
    id_program number(4) not null,
    nume_program varchar2(50) not null,
    pret number(4) not null,
    constraint pk_Program primary key (id_program)
);
```

```
INSERT INTO Program (id_program, nume_program, pret)
VALUES (11, 'Software Beginner', 67);
```

```
INSERT INTO Program (id_program, nume_program, pret)
VALUES (12, 'Software Junior', 87);
```

```
INSERT INTO Program (id_program, nume_program, pret)
VALUES (13, 'Software Advanced', 227);
```

```
INSERT INTO Program (id_program, nume_program, pret)
VALUES (14, 'Software Senior', 167);
```

```
INSERT INTO Program (id_program, nume_program, pret)
VALUES (15, 'Software Expert', 307);
```

```
select * from Program;
```

```
CREATE TABLE Contract(
    id_contract number(4) not null,
    id_client number(4) not null,
    id_program number(4) not null,
    data_contract date,
    constraint pk_Contract primary key (id_contract),
    constraint client_contract_fk foreign key (id_client) references Client(id_client),
    constraint program_contract_fk foreign key (id_program) references Program(id_program)
);
```

```
INSERT INTO Contract (id_contract, id_client, id_program, data_contract)
VALUES (30, 1, 12, to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Contract (id_contract, id_client, id_program, data_contract)
VALUES (31, 2, 13,to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Contract (id_contract, id_client, id_program,data_contract)
VALUES (32, 3, 11, to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Contract (id_contract, id_client, id_program, data_contract)
VALUES (33, 4, 15,to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Contract (id_contract, id_client, id_program,data_contract)
```



```
VALUES (34, 5, 14, to_date('05-06-2021','dd-mm-yyyy'));
```

```
select * from Contract;
```

```
CREATE TABLE Tranzactie(  
    id_tranzactie number(5) not null,  
    id_contract number(4) not null,  
    suma_incasata number(8,2) not null,  
    tva number(4,2) not null,  
    data_tranzactie date default sysdate,  
    constraint tranzactie_contract_fk foreign key (id_contract) references Contract(id_contract),  
    constraint pk_Tranzactie primary key (id_tranzactie)  
);
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)  
VALUES (20, 30, 789 , 20, to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)  
VALUES (21, 31, 568 , 18, to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)  
VALUES (22, 31, 890 , 20, to_date('02-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)  
VALUES (23, 30, 890 , 20, to_date('11-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)
VALUES (24, 32, 123 , 20, to_date('09-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)
VALUES (25, 33, 123 , 20, to_date('05-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)
VALUES (26, 33, 789 , 20, to_date('07-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)
VALUES (27, 33, 789 , 22, to_date('07-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)
VALUES (28, 33, 504 , 20, to_date('06-06-2021','dd-mm-yyyy'));
```

```
INSERT INTO Tranzactie (id_tranzactie, id_contract, suma_incasata, tva, data_tranzactie)
VALUES (29, 30, 504 , 21, to_date('05-06-2021','dd-mm-yyyy'));
```

```
select *
from Tranzactie;
```

```
CREATE TABLE Participant(
    id_participant number(4) not null,
    nume_participant varchar2(50 char),
    prenume_participant varchar2(50 char),
    nume_pozitie_firma varchar2(50 char),
    constraint pk_Participant primary key (id_participant)
);
```

```
INSERT INTO Participant (id_participant, nume_participant, prenume_participant, nume_pozitie_firma)
VALUES (40, 'Sailey', 'William', 'HR');
```

```
INSERT INTO Participant (id_participant, nume_participant, prenume_participant, nume_pozitie_firma)
VALUES (41, 'Welsh', 'Jessica', 'CEO');
```

```
INSERT INTO Participant (id_participant, nume_participant, prenume_participant, nume_pozitie_firma)
VALUES (42, 'Georgescu', 'Amalia', 'Software Engineer');
```

```
INSERT INTO Participant (id_participant, nume_participant, prenume_participant, nume_pozitie_firma)
VALUES (43, 'Marie', 'Anne', 'Software Engineer');
```

```
INSERT INTO Participant (id_participant, nume_participant, prenume_participant, nume_pozitie_firma)
VALUES (44, 'Ionescu', 'Dan', 'Statistics Expert');
```

```
select * from Participant;
```

```
CREATE TABLE Training(
    id_training number(4) not null,
    id_locatie number(4) not null,
    nume_training varchar2(50 char),
    durata number(2) not null, /*nr de ore*/
    nivel_complexitate number(1) not null,
    constraint pk_Training primary key (id_training),
    constraint training_locatie_fk foreign key (id_locatie) references Locatie(id_locatie)
);
```

```
INSERT INTO Training (id_training, id_locatie, nume_training, durata, nivel_complexitate)
VALUES (01, 100, 'Style Code', 5, 1);
```

```
INSERT INTO Training (id_training, id_locatie, nume_training, durata, nivel_complexitate)
VALUES (02, 101, 'Beauty in Code', 5, 2);
```

```
INSERT INTO Training (id_training, id_locatie, nume_training, durata, nivel_complexitate)
VALUES (03, 100, 'Complexity' , 6, 3);
```

```
INSERT INTO Training (id_training, id_locatie, nume_training, durata, nivel_complexitate)
VALUES (04, 104, 'Java Intro', 10, 2);
```

```
INSERT INTO Training (id_training, id_locatie, nume_training, durata, nivel_complexitate)
VALUES (05, 104, 'Phython Hacks', 3, 1);
```

```
select * from Training;
```

```
CREATE TABLE Fisa_Inscriere(
    id_fisa number(4) not null,
    id_participant number(4) not null,
    id_training number(4) not null,
    constraint participant_fisa_fk foreign key (id_participant) references Participant(id_participant),
    constraint fisa_training_fk foreign key (id_training) references Training(id_training),
    constraint pk_Fisa primary key (id_fisa)
);
```

```
INSERT INTO Fisa_Inscriere(id_fisa, id_participant, id_training)
VALUES (50, 40, 01);
```

```
INSERT INTO Fisa_Inscriere(id_fisa, id_participant, id_training)
VALUES (51, 41, 02);
```

```
INSERT INTO Fisa_Inscriere(id_fisa, id_participant, id_training)
VALUES (52, 42, 03);
```

```
INSERT INTO Fisa_Inscriere(id_fisa, id_participant, id_training)
VALUES (53, 43, 04);
```

```
INSERT INTO Fisa_Inscriere(id_fisa, id_participant, id_training)
VALUES (54, 44, 05);
```

```
select * from Fisa_Inscriere;
```

```
CREATE TABLE Data(
    data_sustinerii date default sysdate,
    liber_data number(1),
    constraint pk_Data primary key (data_sustinerii)
);
```

```
INSERT INTO Data(data_sustinerii, liber_data)
VALUES (to_date('19-06-2021','dd-mm-yyyy'), 0);
```

```
INSERT INTO Data(data_sustinerii, liber_data)
VALUES (to_date('20-06-2021','dd-mm-yyyy'), 1);
```

```
INSERT INTO Data(data_sustinerii, liber_data)
VALUES (to_date('21-06-2021','dd-mm-yyyy'), 0);
```

```
INSERT INTO Data(data_sustinerii, liber_data)
VALUES (to_date('22-06-2021','dd-mm-yyyy'), 0);
```

```
INSERT INTO Data(data_sustinerii, liber_data)
VALUES (to_date('23-06-2021','dd-mm-yyyy'), 0);
```

```
INSERT INTO Data(data_sustinerii, liber_data)
VALUES (to_date('24-06-2021','dd-mm-yyyy'), 1);
```

```
select * from Data;
```

```
CREATE TABLE Locatie(
    id_locatie number(4) not null,
    adresa varchar2(100 char),
    pret_per_sala number(5) not null,
    constraint pk_Locatie primary key (id_locatie)
);
```

```
INSERT INTO Locatie(id_locatie, adresa, pret_per_sala)
VALUES(100, 'Romania, Bucharest, Street Lalelor, number 4', 450);
```

```
INSERT INTO Locatie(id_locatie, adresa, pret_per_sala)
VALUES(101, 'Romania, Bucharest, Street Saint Nicolas, number 3', 1450 );
```

```
INSERT INTO Locatie(id_locatie, adresa, pret_per_sala)
VALUES(102, 'Romania, Bucharest, Street Garoafelor, number 12', 900 );
```

```
INSERT INTO Locatie(id_locatie, adresa, pret_per_sala)
VALUES(103, 'Romania, Bucharest, Street Roses, number 98', 9050 );
```

```
INSERT INTO Locatie(id_locatie, adresa, pret_per_sala)
VALUES(104, 'Romania, Bucharest, Street Trenurilor, number 45', 139);
```

```
select * from Locatie;
```

```
CREATE TABLE Sala(
    cod_sala number(4) not null,
    id_locatie number(4) not null,
    arie number(10, 2) not null,
    capacitate number(4) not null,
    constraint pk_Sala primary key(cod_sala),
    constraint pk_sala_locatie foreign key (id_locatie) references Locatie(id_locatie)
);
```

```
INSERT INTO Sala(cod_sala, id_locatie, arie, capacitate)
VALUES(110, 100, 67, 200);
```

```
INSERT INTO Sala(cod_sala, id_locatie, arie, capacitate)
VALUES(111, 101, 189, 300);
```

```
INSERT INTO Sala(cod_sala, id_locatie, arie, capacitate)
VALUES(112, 101, 200, 300);
```

```
INSERT INTO Sala(cod_sala, id_locatie, arie, capacitate)
VALUES(113, 102, 500, 2000);
```

```
INSERT INTO Sala(cod_sala, id_locatie, arie, capacitate)
VALUES(114, 104, 40, 30);
```

```
select * from Sala;
```

```
CREATE TABLE Job(
    id_job number(4) not null,
    nume_firma_job varchar2(50 char),
    nume_pozitie_job varchar2(50 char),
    constraint pk_Job primary key (id_job)
);
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
VALUES (60, 'Endava', 'Software Engineer');
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
VALUES (61, 'Thales', 'Java Developer');
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
VALUES (62, 'Softbinator', 'Software Engineer');
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
VALUES (63, 'Pfizer', 'Data Management Expert');
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
VALUES (64, 'Adobe', 'Manager');
```



```
select * from Job;
```

```
CREATE TABLE Trainer(  
    id_trainer number(4) not null,  
    id_job number(4) not null,  
    nume_trainer varchar2(50 char),  
    prenume_trainer varchar2(50 char),  
    data_nasterii date,  
    constraint pk_Trainer primary key (id_trainer),  
    constraint trainer_job_fk foreign key (id_job) references Job(id_job)  
);
```

```
INSERT INTO Trainer(id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii)  
VALUES(70, 60, 'Johannson' , 'Erik', to_date('20-05-1991','dd-mm-yyyy'));
```

```
INSERT INTO Trainer(id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii)  
VALUES(71, 63, 'Constantinescu', 'Alex', to_date('20-06-1998','dd-mm-yyyy'));
```

```
INSERT INTO Trainer(id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii)  
VALUES(72, 60, 'Moraru', 'Andreea', to_date('20-08-1990','dd-mm-yyyy'));
```

```
INSERT INTO Trainer(id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii)  
VALUES(73, 64, 'Wilson' , 'Jane', to_date('20-06-1967','dd-mm-yyyy'));
```

```
INSERT INTO Trainer(id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii)  
VALUES(74, 62, 'Timson', 'Jim', to_date('20-06-1999','dd-mm-yyyy'));
```

```
INSERT INTO Trainer(id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii)  
VALUES(75, 60, 'Mimson', 'John', to_date('20-06-1980','dd-mm-yyyy'));
```

```
select * from Trainer;
```

```
CREATE TABLE Feedback(  
    id_fisa number(4) not null,  
    id_participant number(4) not null,  
    nota number(4) not null,  
    pareri varchar2(100 char),  
    constraint id_fisa primary key (id_fisa),  
    constraint id_feedback_fk foreign key (id_fisa) references Fisa_Inscriere(id_fisa),  
    constraint id_feedback_fk_2 foreign key (id_participant) references Participant(id_participant)  
);
```

```
INSERT INTO Feedback(id_fisa, id_participant, nota, pareri)  
VALUES (51,41,6,'Better');
```

```
INSERT INTO Feedback(id_fisa, id_participant, nota, pareri)  
VALUES (52,42,5,'Good');
```

```
INSERT INTO Feedback(id_fisa, id_participant, nota, pareri)  
VALUES (53,43,10,'Bad');
```

```
INSERT INTO Feedback(id_fisa, id_participant, nota, pareri)  
VALUES (54,44,7,'Could be better');
```

```
INSERT INTO Feedback(id_fisa, id_participant, nota, pareri)  
VALUES (50,40,5, null);
```

```
select * from Feedback;
```

```
/*11*/
```

/\*1. Selectati toti trainerii al caror nume incepe cu litera M, contine litera n, al caror prenume are 3 sau mai multe litere si care lucreaza la firma Endava, in ordinea lunilor care au trecut de la data de nastere pana in prezent. \*/

```
select nume_trainer, prenume_trainer, data_nasterii
from Trainer t
join Job j on t.id_job = j.id_job
where lower(nume_trainer) like 'm%' and upper(prenume_trainer) like '%N%' and length
(prenume_trainer) >= 3
order by months_between(sysdate, data_nasterii);
```

The screenshot shows a SQL IDE window with a 'Query Builder' tab. The query editor contains the following SQL code:

```

INSERT INTO Feedback(id_fisa, id_participant, nota, pareri)
VALUES (50,40,5, null);

select * from Feedback;

/*11*/
/*1. Selectati toti trainerii al caror nume incepe cu litera M, contine litera n, al caror prenume
are 3 sau mai multe litere si care lucreaza la firma Endava, in ordinea lunilor care au trecut de la data de nastere
pana in prezent. */
select nume_trainer, prenume_trainer, data_nasterii
from Trainer t
join Job j on t.id_job = j.id_job
where lower(nume_trainer) like 'm%' and upper(prenume_trainer) like '%N%' and length (prenume_trainer) >= 3
order by months_between(sysdate, data_nasterii);

```

Below the query editor, the 'Script Output' tab shows the results of the query in a table:

	NUME_TRAINER	PRENUME_TRAINER	DATA_NASTERII
1	Moraru	Andreea	20-AUG-90
2	Mimson	John	20-JUN-80

/\*length() : intoarce lungimea sirului de caractere, lower(): transforma toate caracterele sirului in minuscule, iar upper()

transforma toate caracterele sirului in majuscule\*/

/\* am folosit 2 functii pe tipul de date date: sysdate care intoarce data si timpul curent

si months\_between(expr\_date2, expr\_date1) care returneaza nr de luni dintre cele 2 date calendaristice; am pus sysdate ca prima

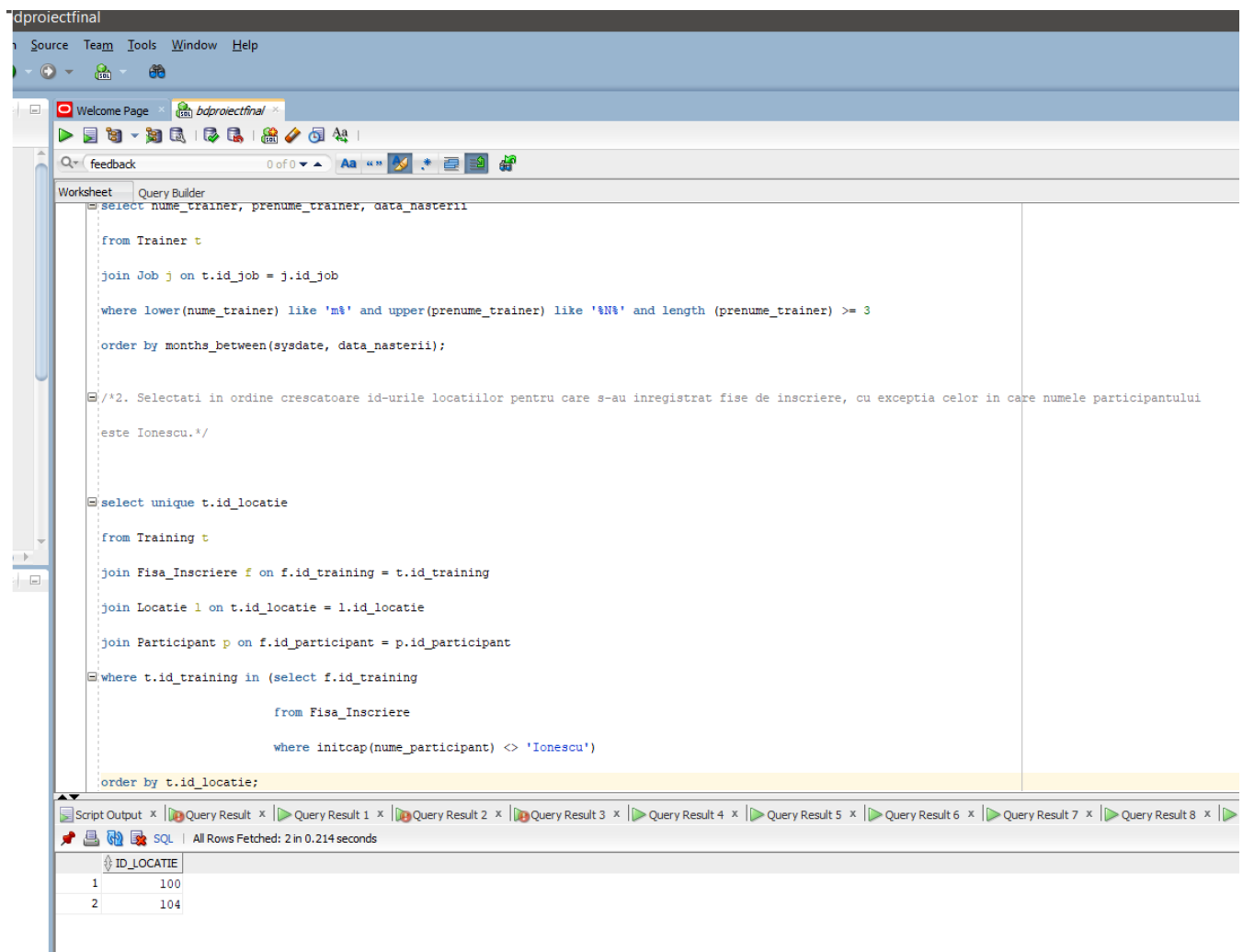
data pentru a nu obtine un nr negativ, intrucat sigur va fi mai mare \*/

/\*order by : ordoneaza datele dupa criteriului selectat\*/

/\*join pe doua tabele: Trainer si Job \*/

/\*2. Selectati in ordine crescatoare id-urile locatiilor pentru care s-au inregistrat fise de inscriere, cu exceptia celor in care numele participantului este Ionescu.\*/\*

```
select unique t.id_locatie
from Training t
join Fisa_Inscriere f on f.id_training = t.id_training
join Locatie l on t.id_locatie = l.id_locatie
join Participant p on f.id_participant = p.id_participant
where t.id_training in (select f.id_training
                        from Fisa_Inscriere
                        where initcap(nume_participant) <> 'Ionescu')
order by t.id_locatie;
```



/\* subcerere necorelata, filtrare la nivel de linii ( from join + where) \*/

/\*cererea internă este executată prima și determină o valoare (sau o mulțime de valori); cererea externă se execută o singură dată, utilizând valorile returnate de cererea internă.\*/

/\*3.\*/

/\*--trainingurile cu mai mult de un participant, neluandu-se in calcul cei cu prenumele Ana\*/

select f.id\_training, count(f.id\_participant)

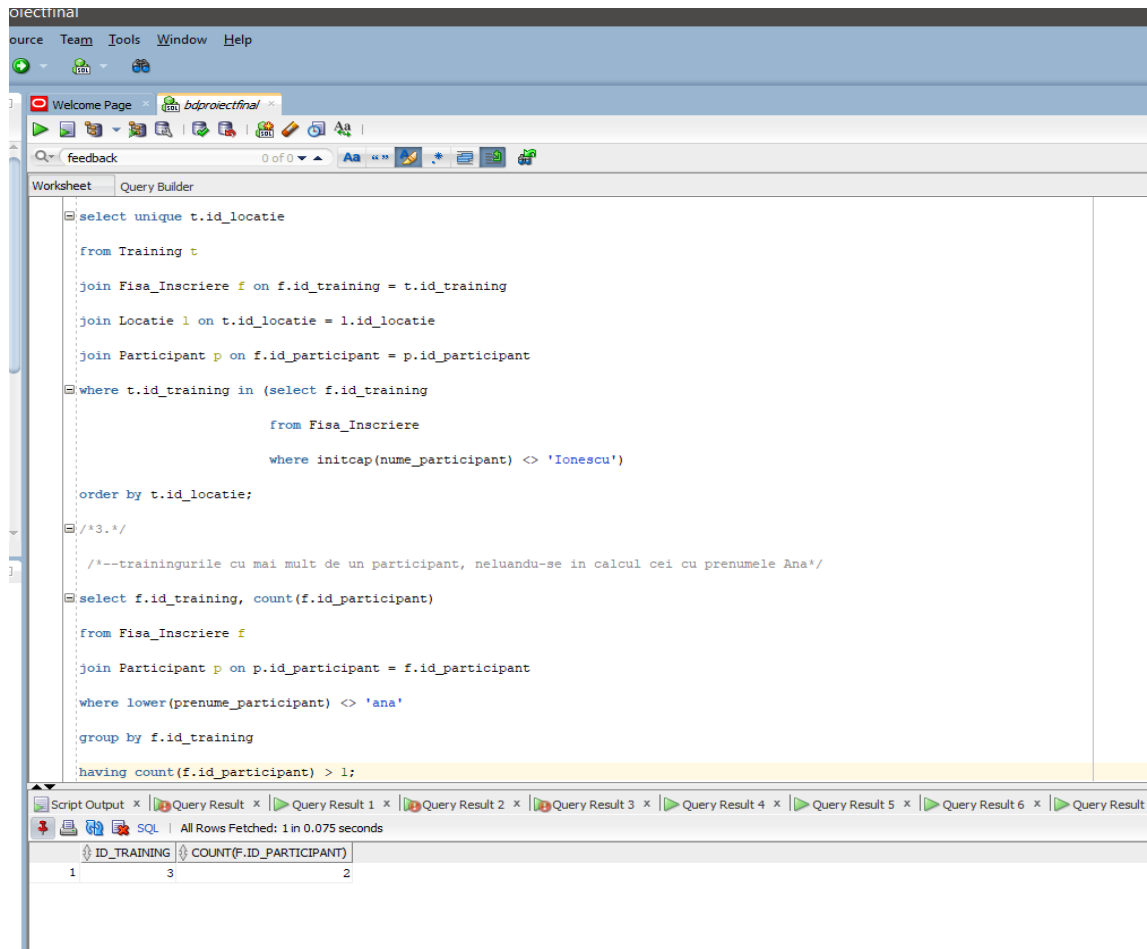
from Fisa\_Inscriere f

join Participant p on p.id\_participant = f.id\_participant

where lower(prenume\_participant) <> 'ana'

group by f.id\_training

having count(f.id\_participant) > 1;



/\*4. Selectati id ul locatiei si, sub numele de total, pretul locatiilor al caror pret per sala este mai mare decat average-ul

si diferit de 1.\*/

with avg\_pret\_sala as ( select id\_locatie, sum(pret\_per\_sala) total

from Locatie

group by id\_locatie),

val\_medie as (select avg(pret\_per\_sala) medie from Locatie)

```

select *
from avg_pret_sala
where total > ( select medie
                from val_medie
                where nvl(medie, 1) <> 1 and decode(medie, medie, medie, 1) <> 1);

/* decode: daca e medie se ia valoarea mediei, altfel se va aloca 1 decode-ului */

```

The screenshot shows a SQL IDE window titled "bdproiectfinal". The main editor displays a SQL query. The query includes a subquery to calculate the average price per room and a main query to select rooms where the total price is greater than the average price per room. The query is as follows:

```

group by f.id_training

having count(f.id_participant) > 1;

/*4. Selectati id ul locatiei si, sub numele de total, pretul locatiilor al caror pret per sala este mai mare decat average-ul
si diferit de 1.*/

with avg_pret_sala as ( select id_locatie, sum(pret_per_sala) total
                        from Locatie
                        group by id_locatie),
val_medie as (select avg(pret_per_sala) medie from Locatie)

select *
from avg_pret_sala
where total > ( select medie
                from val_medie
                where nvl(medie, 1) <> 1 and decode(medie, medie, medie, 1) <> 1);

/* decode: daca e medie se ia valoarea mediei, altfel se va aloca 1 decode-ului */

```

The IDE interface includes a menu bar (Run, Source, Team, Tools, Window, Help), a toolbar with various icons, and a status bar at the bottom. The status bar indicates "All Rows Fetched: 1 in 0.092 seconds". Below the status bar, a table with two columns, "ID\_LOCATIE" and "TOTAL", is displayed. The table contains one row with the values "1" and "9050".

ID_LOCATIE	TOTAL
1	9050



/\*5 Selectati id ul trainerului, id ul jobului, numele, prenumele si data nasterii trainerilor al caror nume are o lungime mai mare decat media numelor firmelor la care sunt

inregistrate joburi, ordonand datele dupa id ul jobului, descrescator pentru cele cu valori mai mari sau egale cu 63 si crescator pentru restul.\*/

```
select id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii
```

```
from Trainer outer
```

```
where length(nume_trainer) >
```

```
( select avg(length(nume_firma_job))
```

```
from Job
```

```
where id_job = outer.id_job)
```

```
order by case when id_job >= 63 then id_job end desc,
```

```
case when id_job < 63 then id_job end;
```

bdprojectfinal

Source Team Tools Window Help

Welcome Page bdprojectfinal

feedback 0 of 0

Worksheet Query Builder

```

where total > ( select medie
                from val_medie
                where nvl(medie, 1) <> 1 and decode(medie, medie, medie, 1) <> 1);

/* decode: daca e medie se ia valoarea mediei, altfel se va aloca 1 decode-ului */

/*5 Selectati id ul trainerului, id ul jobului, numele, prenumele si data nasterii trainerilor al caror nume are o lungime mai mare decat media numelor firmelor la care sunt
inregistrate joburi, ordonand datele dupa id ul jobului, descrescator pentru cele cu valori mai mari sau egale cu 63 si crescator pentru restul.*/

select id_trainer, id_job, nume_trainer, prenume_trainer, data_nasterii
from Trainer outer
where length(nume_trainer) >
( select avg(length(nume_firma_job))
  from Job
  where id_job = outer.id_job)
order by case when id_job >= 63 then id_job end desc,
         case when id_job < 63 then id_job end;

```

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

SQL All Rows Fetched: 3 in 0.146 seconds

ID_TRAINER	ID_JOB	NUME_TRAINER	PRENUME_TRAINER	DATA_NASTERII
1	70	60 Johansson	Erik	20-MAY-91
2	73	64 Wilson	Jane	20-JUN-67
3	71	63 Constantinescu	Alex	20-JUN-98

/\*subcerere corelata + CASE \*/

/\*cererea externă determină o linie candidat; cererea internă este executată utilizând valoarea liniei candidat;

valorile rezultatedin cererea internă sunt utilizate pentru calificarea sau descalificarea liniei candidat;

pașii precedenți se repetă până cand nu mai există linii candidat.\*/

/\*12\*/

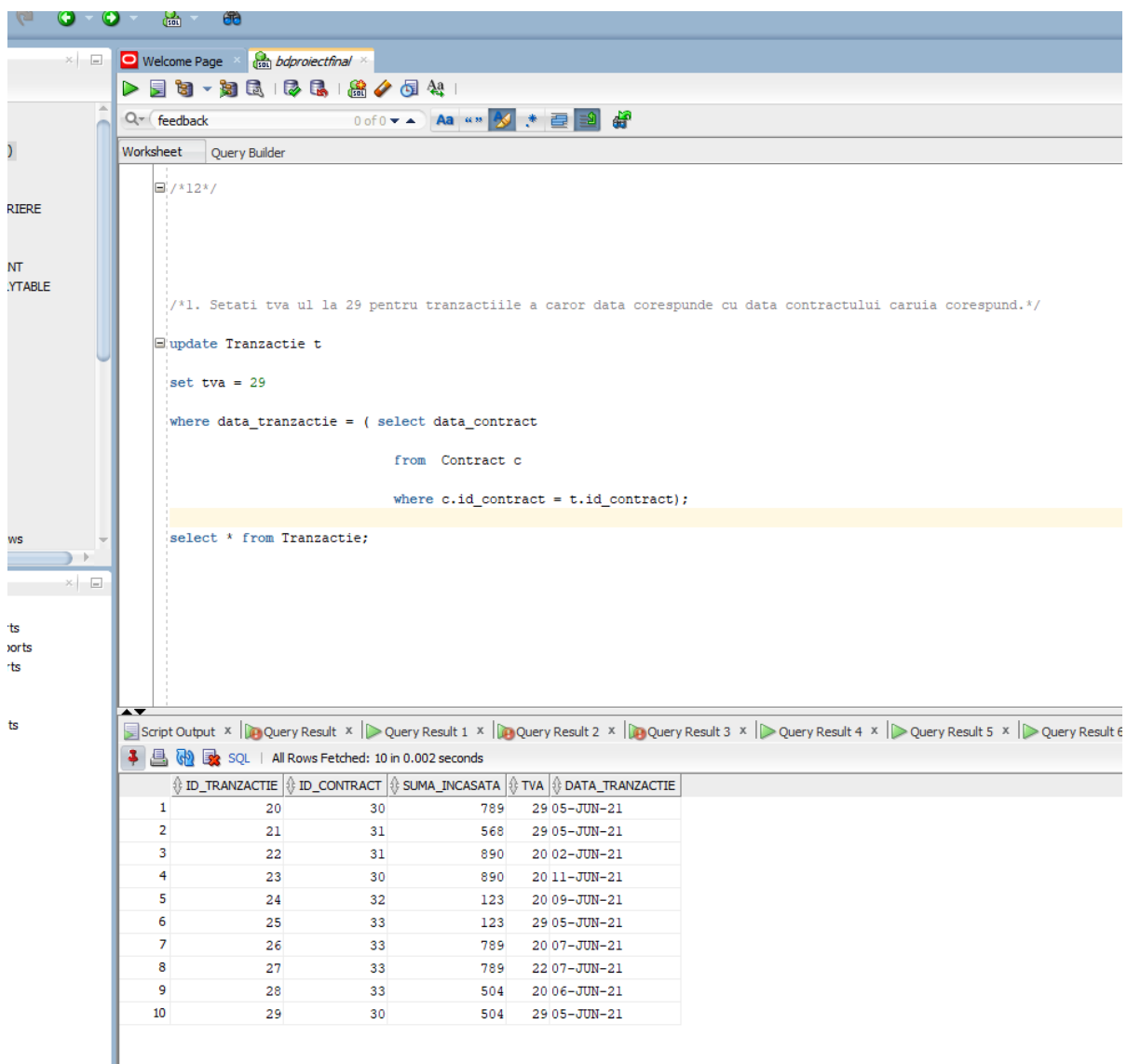
/\*1. Setati tva ul la 29 pentru tranzactiile a caror data corespunde cu data contractului caruia corespund.\*/

update Tranzactie t

set tva = 29

where data\_tranzactie = ( select data\_contract  
from Contract c  
where c.id\_contract = t.id\_contract);

select \* from Tranzactie;



The screenshot displays a SQL query editor with the following script:

```
/*1. Setati tva ul la 29 pentru tranzactiile a caror data corespunde cu data contractului caruia corespund.*/  
  
update Tranzactie t  
  
set tva = 29  
  
where data_tranzactie = ( select data_contract  
from Contract c  
where c.id_contract = t.id_contract);  
  
select * from Tranzactie;
```

Below the query editor, the results window shows the output of the final query. It displays a table with 10 rows of transaction data:

ID_TRANZACTIE	ID_CONTRACT	SUMA_INCASATA	TVA	DATA_TRANZACTIE
1	20	30	789	29 05-JUN-21
2	21	31	568	29 05-JUN-21
3	22	31	890	20 02-JUN-21
4	23	30	890	20 11-JUN-21
5	24	32	123	20 09-JUN-21
6	25	33	123	29 05-JUN-21
7	26	33	789	20 07-JUN-21
8	27	33	789	22 07-JUN-21
9	28	33	504	20 06-JUN-21
10	29	30	504	29 05-JUN-21

/\*2. Stergeti tranzactiile a caror data corespunde cu data contractului caruia corespund. \*/

delete

from Tranzactie t

where data\_tranzactie = ( select data\_contract

from Contract c

where c.id\_contract = t.id\_contract);

The screenshot shows a database management tool interface. The main window displays a SQL script in a text editor. The script includes a comment in Romanian: `/*2. Stergeti tranzactiile a caror data corespunde cu data contractului caruia corespund. */`. Below the comment, the script contains a `delete` statement: `delete from Tranzactie t where data_tranzactie = ( select data_contract from Contract c where c.id_contract = t.id_contract);`, followed by a `select * from Tranzactie;` statement. The script is highlighted with a red box. Below the script, the 'Script Output' pane shows the execution results: 'Task completed in 0.061 seconds', '1 row inserted.', '1 row inserted.', '1 row inserted.', '1 row inserted.', and '>>Query Run In:Query Result 18'. The final result, '4 rows deleted.', is highlighted with a red box.

```
set tva = 29

where data_tranzactie = ( select data_contract
                           from Contract c
                           where c.id_contract = t.id_contract);

select * from Tranzactie;

/*2. Stergeti tranzactiile a caror data corespunde cu data contractului caruia corespund. */

delete
from Tranzactie t
where data_tranzactie = ( select data_contract
                           from Contract c
                           where c.id_contract = t.id_contract);

select * from Tranzactie;
```

Script Output x Query Result x Query Result 1 x Query Result 2 x Query Result 3 x Query Result 4 x Query Result 5 x Query Result 6

Task completed in 0.061 seconds

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

>>Query Run In:Query Result 18

4 rows deleted.

/\*2. Stergeti tranzactiile a caror data corespunde cu data contractului caruia corespund. \*/

delete

from Tranzactie t

where data\_tranzactie = ( select data\_contract

from Contract c

where c.id\_contract = t.id\_contract);

select \* from Tranzactie;

ID_TRANZACTIE	ID_CONTRACT	SUMA_INCASATA	TVA	DATA_TRANZACTIE
1	22	31	890	20 02-JUN-21
2	23	30	890	20 11-JUN-21
3	24	32	123	20 09-JUN-21
4	26	33	789	20 07-JUN-21
5	27	33	789	22 07-JUN-21
6	28	33	504	20 06-JUN-21

/\*3. Setati capacitatea la 1000 pentru salile a caror locatie are pretul per sala mai mare de 500.\*/

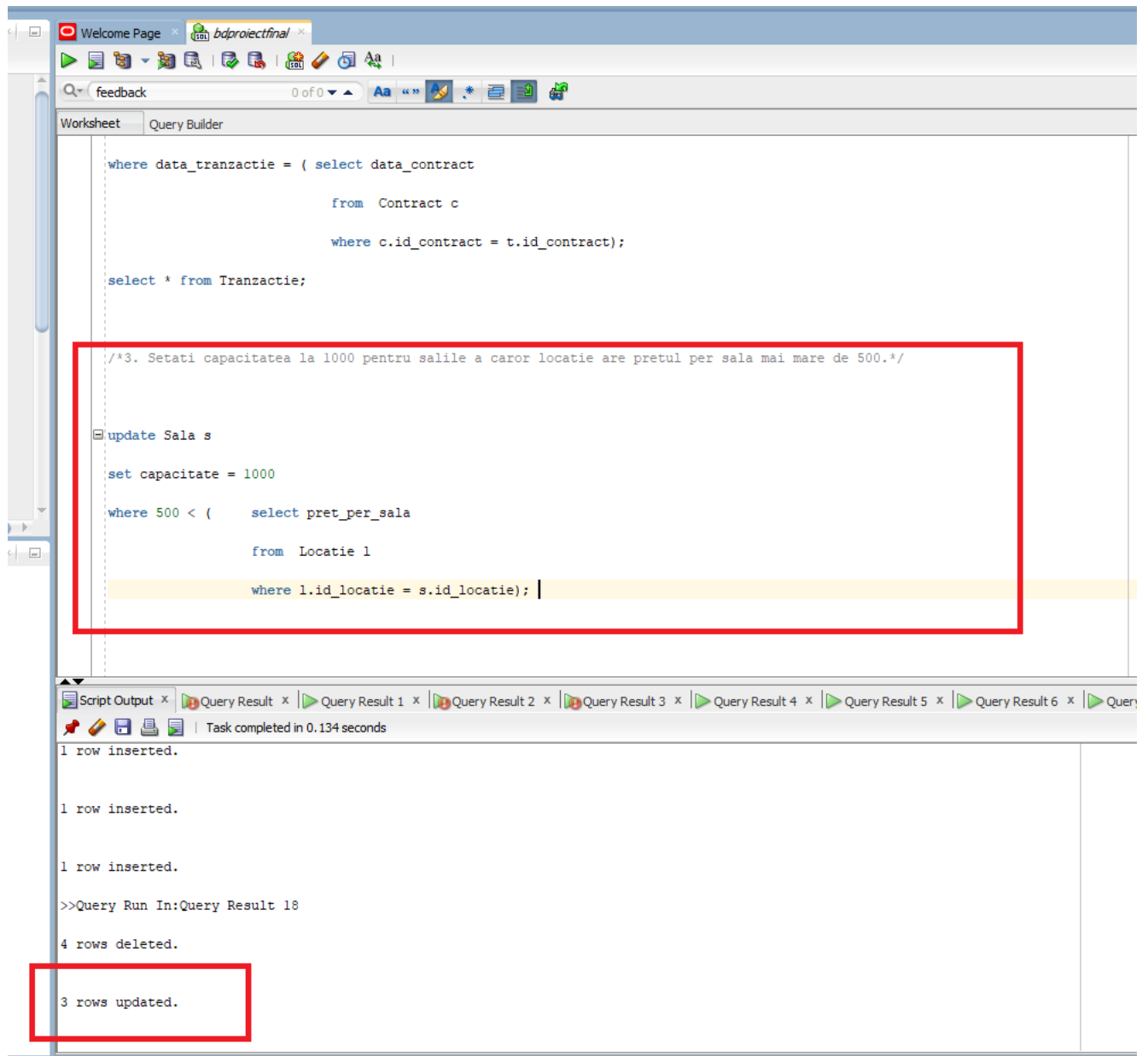
update Sala s

set capacitate = 1000

where 500 < ( select pret\_per\_sala

from Locatie l

where l.id\_locatie = s.id\_locatie);



The screenshot shows a SQL IDE window with a query editor and a results pane. The query editor contains the following SQL code:

```

select * from Tranzactie;

/*3. Setati capacitatea la 1000 pentru salile a caror locatie are pretul per sala mai mare de 500.*/

update Sala s
set capacitate = 1000
where 500 < (
    select pret_per_sala
    from Locatie l
    where l.id_locatie = s.id_locatie);

select * from Sala;
  
```

The results pane shows the output of the last query, displaying a table with 5 rows and 5 columns: COD\_SALA, ID\_LOCATIE, ARIE, and CAPACITATE. The data is as follows:

	COD_SALA	ID_LOCATIE	ARIE	CAPACITATE
1	110	100	67	200
2	111	101	189	1000
3	112	101	200	1000
4	113	102	500	1000
5	114	104	40	30

/\*13\*/

/\*Inserati in tabelul Job inca o instanta pentru firma Google, pozitia Software Enginner, id-ul job-ului fiind generat printr-o sequence.\*/

CREATE SEQUENCE s\_job

INCREMENT BY 1

START WITH 65;

INSERT INTO Job(id\_job, nume\_firma\_job, nume\_pozitie\_job)

VALUES (s\_job.NEXTVAL, 'Google', 'Software Engineer');

Oracle SQL Developer : bdproiectfinal

File Edit View Navigate Run Source Team Tools Window Help

Connections

Oracle Connections

bdproiectfinal

- Tables (Filtered)
  - DATA
  - FEEDBACK
  - FISA\_INSCRIERE
  - JOB
  - LOCATIE
  - PARTICIPANT
  - TEMPORARYTABLE
  - TRAINING
- Views
- Indexes
- Packages
- Procedures
- Functions
- Operators
- Queues
- Queues Tables
- Triggers
- Types
- Sequences
- Materialized Views

Reports

- All Reports
- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimesTen Reports
- User Defined Reports

Worksheet

Query Builder

```
select * from Sala;
```

```
/*13*/
```

```
/*Inserati in tabelul Job inca o instanta pentru firma Google, pozitia Software Engineer, id-ul job-ului fiind generat printr-o sequence.*/
```

```
CREATE SEQUENCE s_job
```

```
INCREMENT BY 1
```

```
START WITH 65;
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
```

```
VALUES (s_job.NEXTVAL, 'Google', 'Software Engineer');
```

Script Output x | Query Result x | Query Result 1 x | Query Result 2 x | Query Result 3 x | Query Result 4 x | Query Result 5 x | Query Result 6 x | Query Result 7 x | Que... x |

Task completed in 0.114 seconds

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

>>Query Run In:Query Result 22

Sequence S\_JOB created.



Oracle SQL Developer : bdprojectfinal

File Edit View Navigate Run Source Team Tools Window Help

Connections

- Oracle Connections
- bdprojectfinal
  - Tables (Filtered)
    - DATA
    - FEEDBACK
    - FISA\_INSCRIERE
    - JOB
    - LOCATIE
    - PARTICIPANT
    - TEMPORARYTABLE
    - TRAINING
  - Views
  - Indexes
  - Packages
  - Procedures
  - Functions
  - Operators
  - Queues
  - Queues Tables
  - Triggers
  - Types
  - Sequences
  - Materialized Views

Reports

- All Reports
- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimesTen Reports
- User Defined Reports

Welcome Page

feedback 0 of 0

Worksheet Query Builder

```
select * from Sala;
```

```
/*13*/
```

```
/*Inserati in tabelul Job inca o instanta pentru firma Google, pozitia Software Engineer, id-ul job-ului fiind generat printr-o sequence.*/
```

```
CREATE SEQUENCE s_job
```

```
INCREMENT BY 1
```

```
START WITH 65;
```

```
INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)
```

```
VALUES (s_job.NEXTVAL, 'Google', 'Software Engineer');
```

Script Output

Task completed in 0.065 seconds

1 row inserted.

1 row inserted.

1 row inserted.

>>Query Run In:Query Result 22

Sequence S\_JOB created.

1 row inserted.

The screenshot shows a SQL IDE window with a script editor and a results pane. The script editor contains the following SQL commands:

```

select * from Sala;

/*13*/

/*Inserati in tabelul Job inca o instanta pentru firma Google, pozitia Software Enginner, id-ul job-ului fiind generat printr-o sequ

CREATE SEQUENCE s_job

INCREMENT BY 1

START WITH 65;

INSERT INTO Job(id_job, nume_firma_job, nume_pozitie_job)

VALUES (s_job.NEXTVAL, 'Google', 'Software Engineer');

select * from Job;

```

The results pane shows the output of the last query, displaying a table with 6 rows and 3 columns: ID\_JOB, NUME\_FIRMA\_JOB, and NUME\_POZITIE\_JOB.

ID_JOB	NUME_FIRMA_JOB	NUME_POZITIE_JOB
1	60 Endava	Software Engineer
2	61 Thales	Java Developer
3	62 Softbinator	Software Engineer
4	63 Pfizer	Data Management Expert
5	64 Adobe	Manager
6	65 Google	Software Engineer

/\*14. Creati o vizualizare care sa contina codul salilor aflate la locatii ale caror pret per sala este mai mic de 10000.\*/

create view vizualizare

as

select cod\_sala

from Sala s

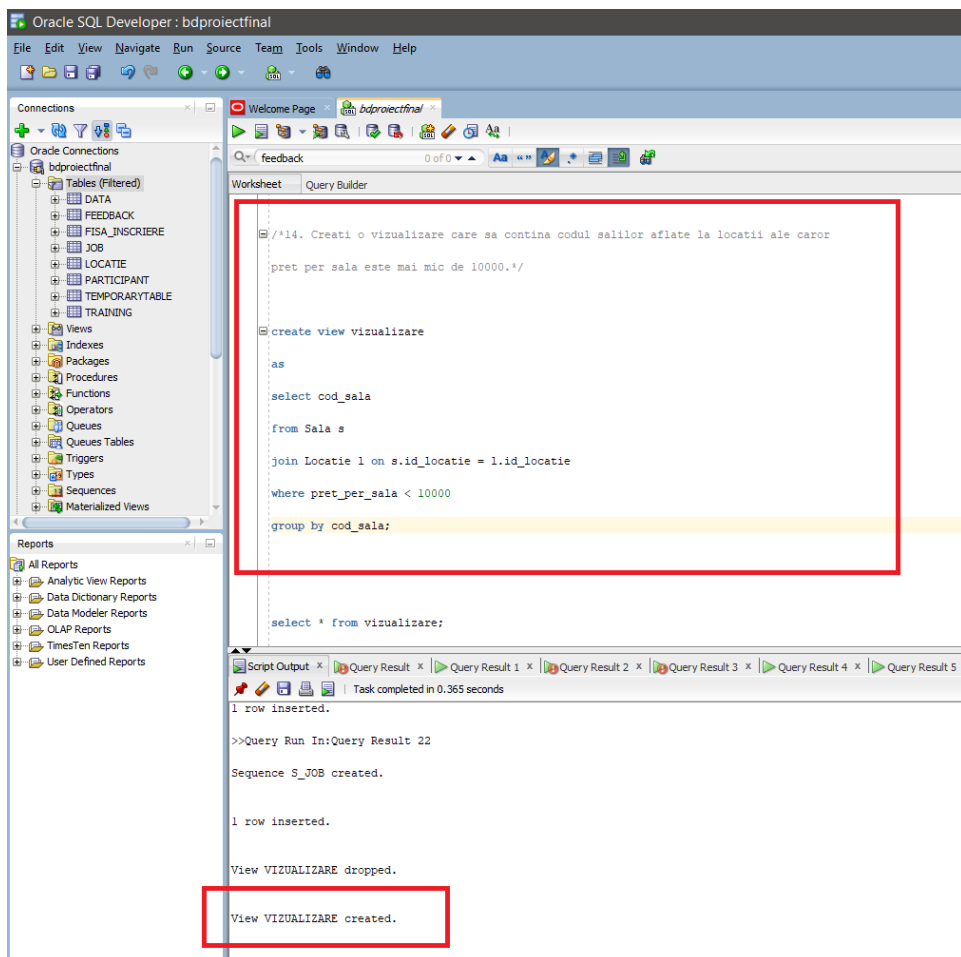
join Locatie l on s.id\_locatie = l.id\_locatie

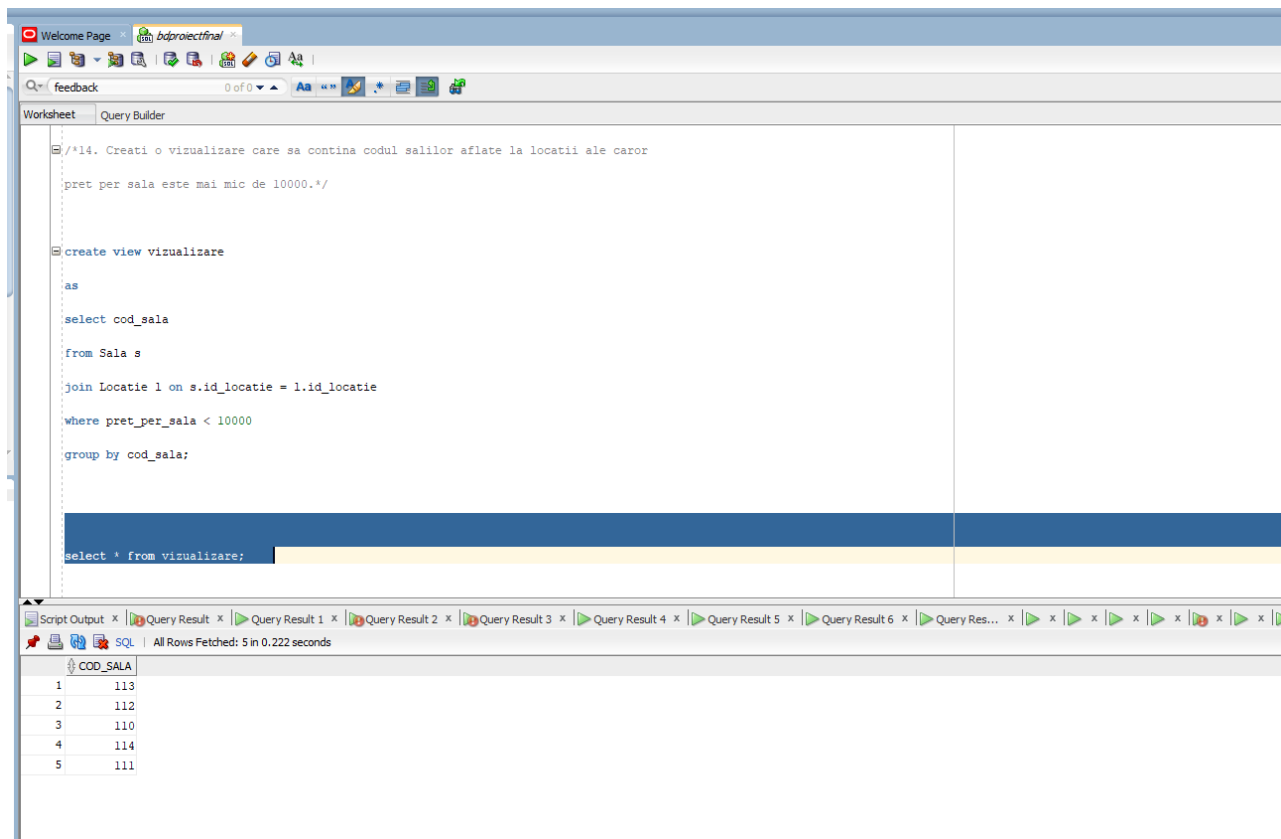
where pret\_per\_sala < 10000

group by cod\_sala;

select \* from vizualizare;

drop view vizualizare;





/\*15. Index pentru nume si prenume trainer care sa optimizeze cererea:

(11.5)Selectati id ul trainerului, id ul jobului, numele, prenumele si data nasterii trainerilor al caror nume are o lungime mai mare decat media numelor firmelor la care sunt inregistrate joburi, ordonand datele dupa id ul jobului, descrescator pentru cele cu valori mai mari sau egale cu 63 si crescator pentru restul.\*/

/\* 11.5

select id\_trainer, id\_job, nume\_trainer, prenume\_trainer, data\_nasterii

from Trainer outer

where length(nume\_trainer) >

( select avg(length(nume\_firma\_job))

from Job

```
where id_job = outer.id_job)
```

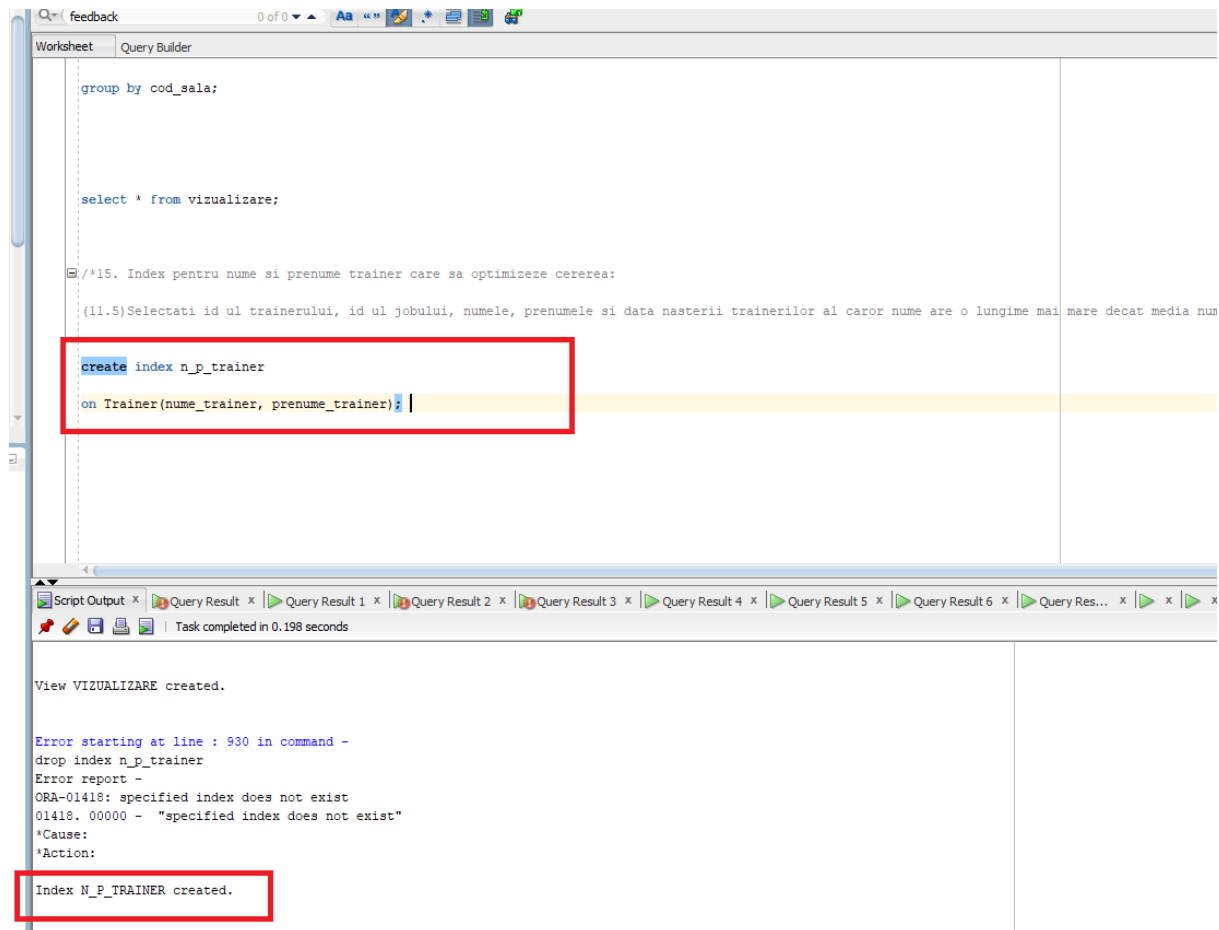
```
order by case when id_job >= 63 then id_job end desc,
```

```
case when id_job < 63 then id_job end;
```

```
*/
```

```
create index n_p_trainer
```

```
on Trainer(num_train, prenum_train);
```



```
/*16*/
```

/\*OUTERJOIN: Selectati id ul fisei, id ul participantului in functie de fisa, numele si prenumele participantului, nota pe care

a oferit-o la feedback si id-urilor trainingurilor, ordonand dupa cele din urma.\*/

select f.id\_fisa, t.id\_training, f.id\_participant,nume\_participant, prenume\_participant, nota

from Fisa\_Inscriere f

full outer join Participant p on p.id\_participant = f.id\_participant

full outer join Feedback fe on fe.id\_participant = f.id\_participant

full outer join Training t on f.id\_training = t.id\_training

order by f.id\_fisa;

The screenshot shows a database query editor interface. The top toolbar includes icons for running queries, saving, and other standard database operations. The main text area contains a SQL query that uses a full outer join to combine data from Fisa\_Inscriere, Participant, Feedback, and Training tables. The query is ordered by f.id\_fisa. Below the query, a status bar indicates 'All Rows Fetched: 6 in 0.136 seconds'. The bottom section displays a table with 6 rows and 6 columns: ID\_FISA, ID\_TRAINING, ID\_PARTICIPANT, NUME\_PARTICIPANT, PRENUME\_PARTICIPANT, and NOTA. The data is as follows:

ID_FISA	ID_TRAINING	ID_PARTICIPANT	NUME_PARTICIPANT	PRENUME_PARTICIPANT	NOTA
1	50	1	40 Sailey	William	5
2	51	3	41 Welsh	Jessica	6
3	52	3	42 Georgescu	Amalia	5
4	53	4	43 Marie	Anne	10
5	54	5	44 Ionescu	Dan	7
6	(null)	2	(null)	(null)	(null)

```
/* DIVISION */
```

```
select t.id_training
```

```
from Training t
```

```
where not exists (
```

```
    select 'x'
```

```
    from Locatie l
```

```
    where l.id_locatie = t.id_locatie and pret_per_sala < 500/*
```

```
    where not exists (select 'x'
```

```
        from Sala s
```

```
        where s.id_locatie = l.id_locatie)*/
```

```
)
```

```
order by t.id_training;
```

Welcome Page xbdprojectfinal x

feedback

0 of 0

Aa

“”

WorksheetQuery Builder

```
/* DIVISION */

select t.id_training
from Training t
where not exists (
    select 'x'
    from Locatie l
    where l.id_locatie = t.id_locatie and pret_per_sala < 500/*
    where not exists (select 'x'
        from Sala s
        where s.id_locatie = l.id_locatie)*/
)

order by t.id_training;
```

Script Output xQuery Result xQuery Result 1 xQuery Result 2 xQuery Result 3 xQuery Result 4 xQuery Result 5 xQuery Result 6 xQ... x

SQL

All Rows Fetched: 2 in 0.002 seconds

ID_TRAINING
1
2