**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 colecteaza datele despre sala in care se va desfasura trainingul. Cheia primara consta intr-un numar unic de identificare, COD\_SALA.

Feedback ul este strict legat de fisa de inscriere, continand parerile si nota oferita per total experientei. Cheia sa primara este, asadar, ID\_FISA, totodata si cheie straina.

**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 sa fie insotit de o unica tranzactie, iar o tranzactie poate corespunde unui unic contract. (1:1)

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

Un program prevede una sau mai multe fise de inscriere, iar o fisa de inscriere corespunde unui singur program. ( 1:M)

Un participant completeza una sau mai multe fise de inscriere, iar o fisa de inscriere este completata de un singur participant. (1:M)

O fisa de inscriere are asociat un singur formular de feedback, iar unui formular de feedback ii este asociata o singura fisa de inscriere. (1:1)

Pentru un training se inregistreaza zero sau mai multe fise de inscriere. (1:M)

Un training poate fi sustinut la mai multe date, iar pe o data se pot sustine mai multe traininguri. (M:M) Am transformat relatia prin intermediul tabelului asociativ ASIGNARE\_DATA. Astfel, pentru un training se aloca una sau mai multe asignari de date (1:M), iar pentru o data corespund una sau mai multe asignari de data. (1:M) Vor fi trecute in baza de data doar datele la care se iti trainingurile.

Un training se desfasoara la o locatie unica, iar la o locatie se pot desfasura unul sau mai multe traininguri. (1:M)

O sala are o singura locatie, iar o locatie are una sau mai multe sali. (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

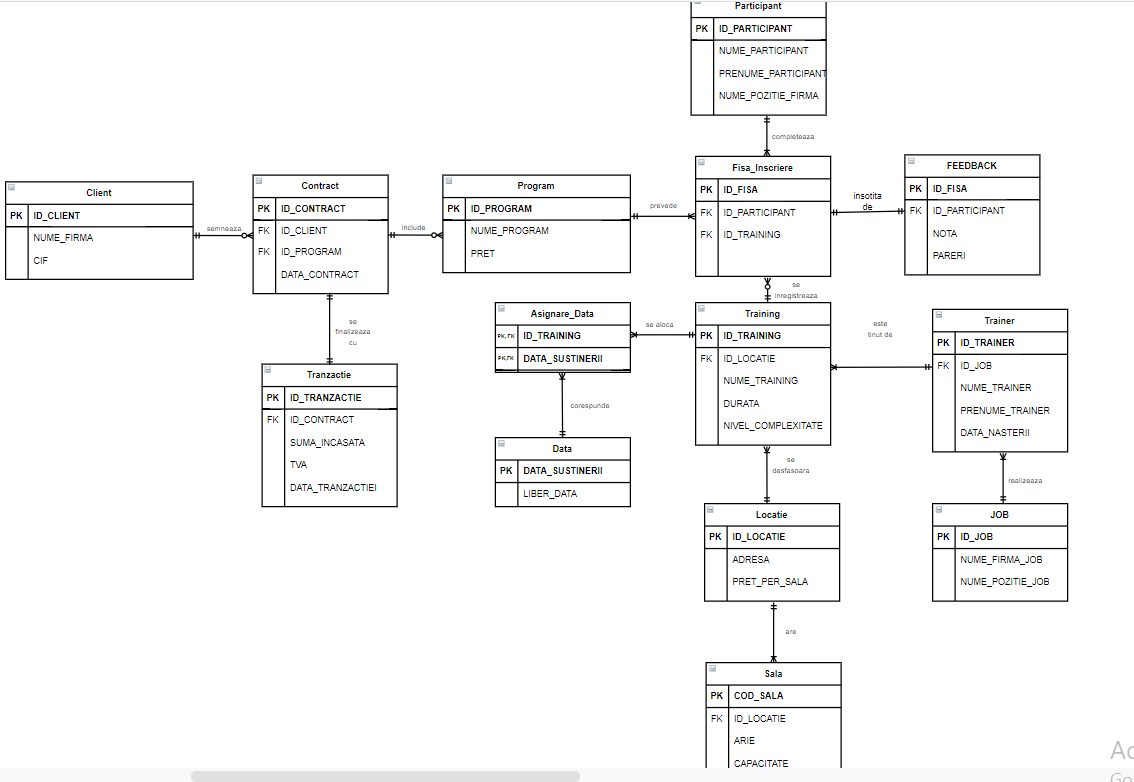
ARIA : 47

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

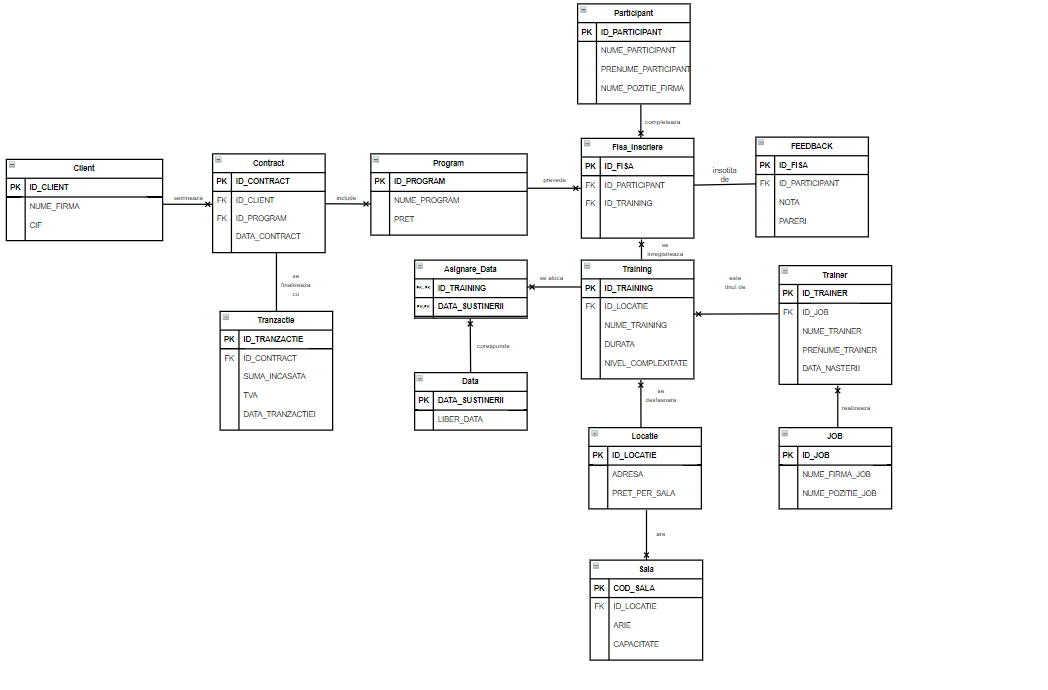
CAPACITATE: 156

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,

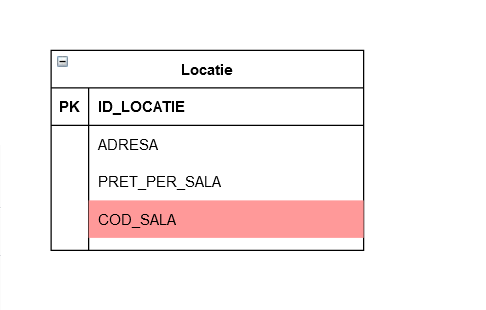
Feedback(#ID\_FISA, ID\_PARTICIPANT, NOTA, PARERI)

**Normalizare:**

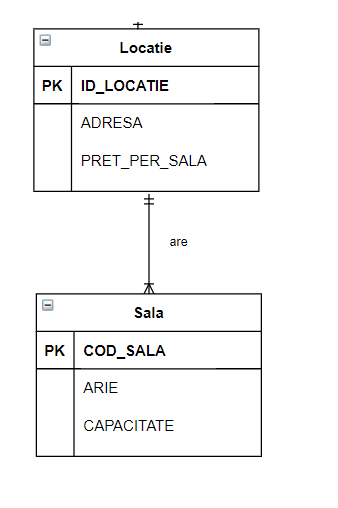
**FN1**: no multi-valued attributes

Exemplu: o locatie poate avea mai multe sali disponibile

Non-FN1:



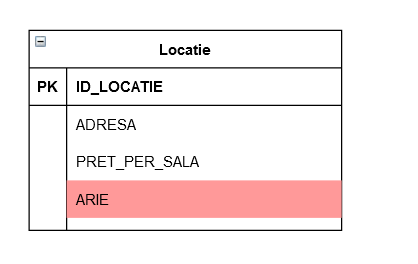
FN1:



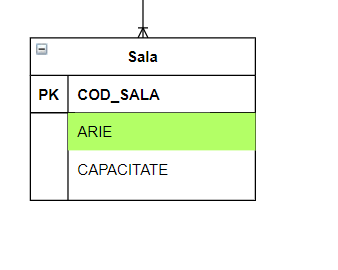
**FN2**: any non-UID attribute be dependent on the entire UID

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:



FN2:

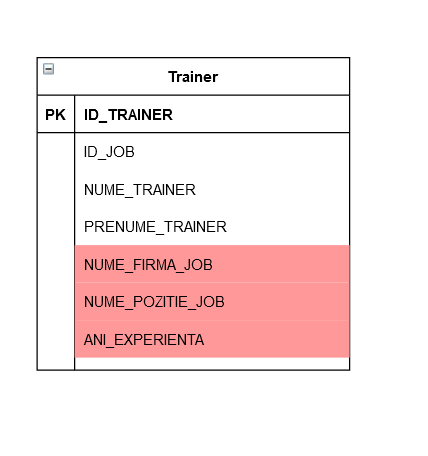


**FN3**: no non-UID attribute can be dependent on another non-UID attribute

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

Non-FN3:

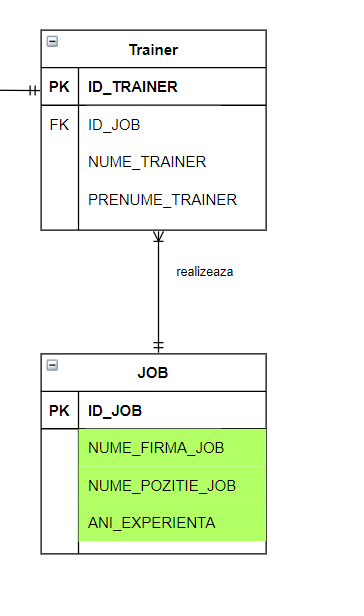
Trainer(ID\_TRAINER, ID\_JOB, NUME\_TRAINER, PRENUME\_TRAINER, NUME\_FIRMA\_JOB, NUME\_POZITIE\_JOB, ANI\_EXPERIENTA)



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', 'Statitics 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. "\*/

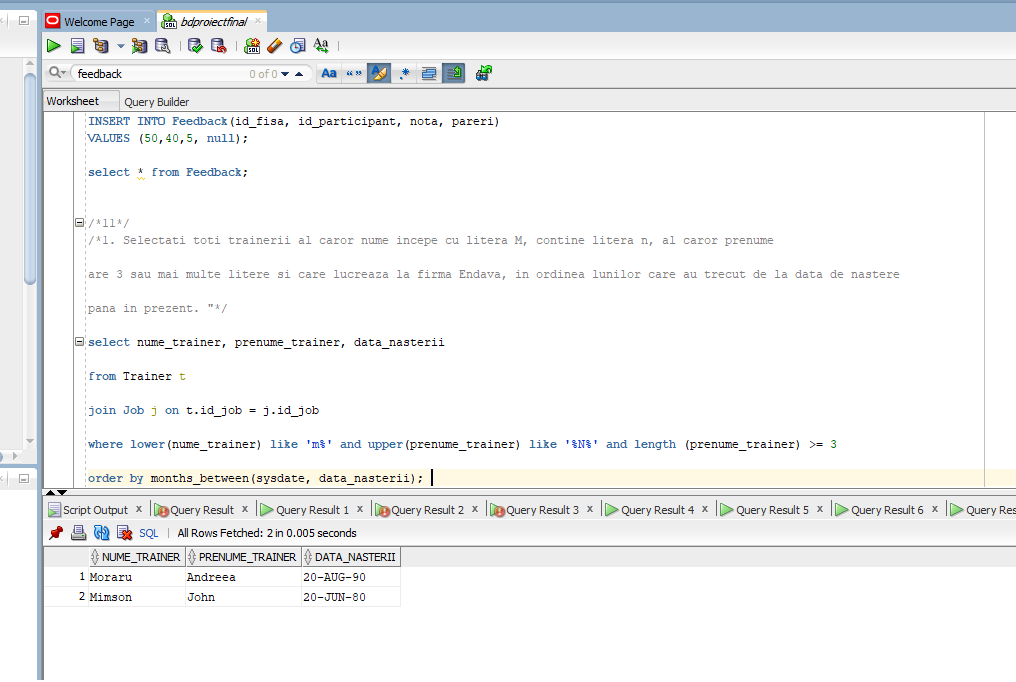
select nume\_trainer, prenume\_trainer, data\_nasterii

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);



/\*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 criterului 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

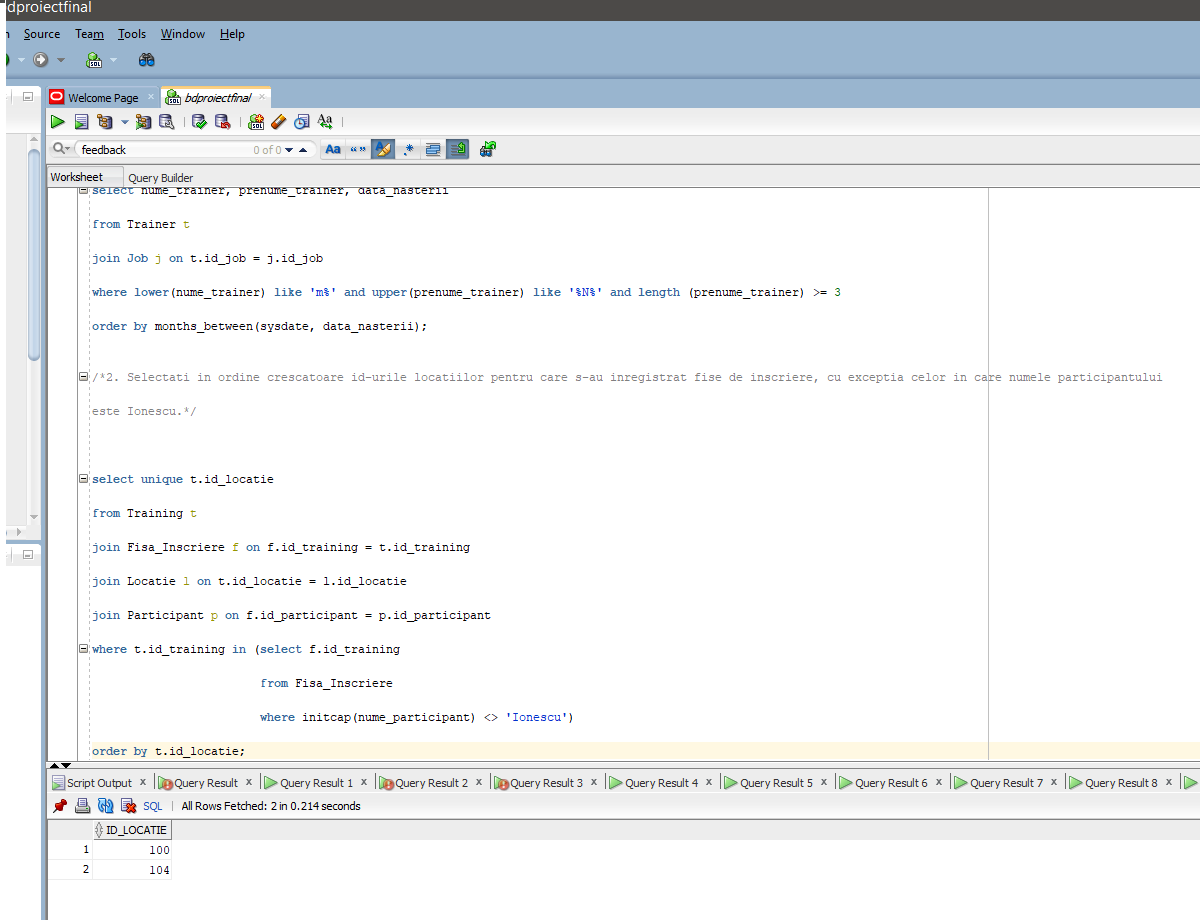
join Participant p on f.id\_participant = p.id\_participant

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 şi determină o valoare (sau o mulţime 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)

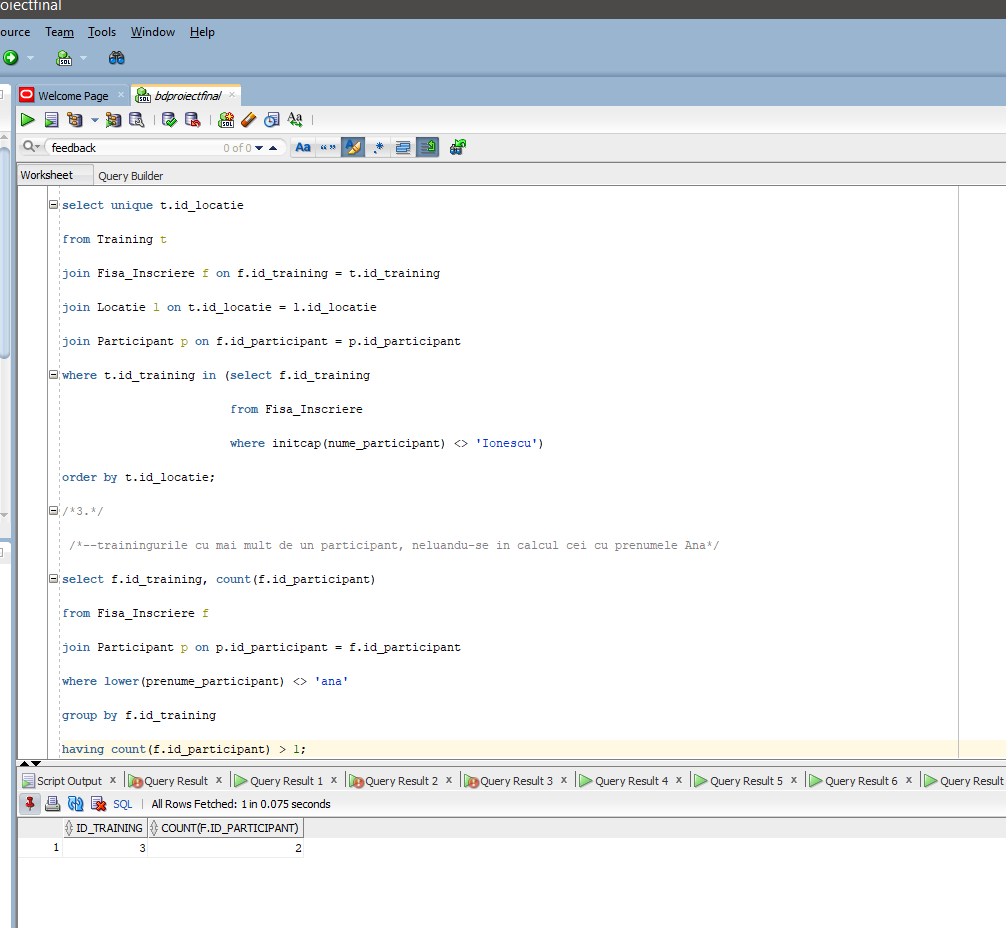
from Fisa\_Inscriere f

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 \*

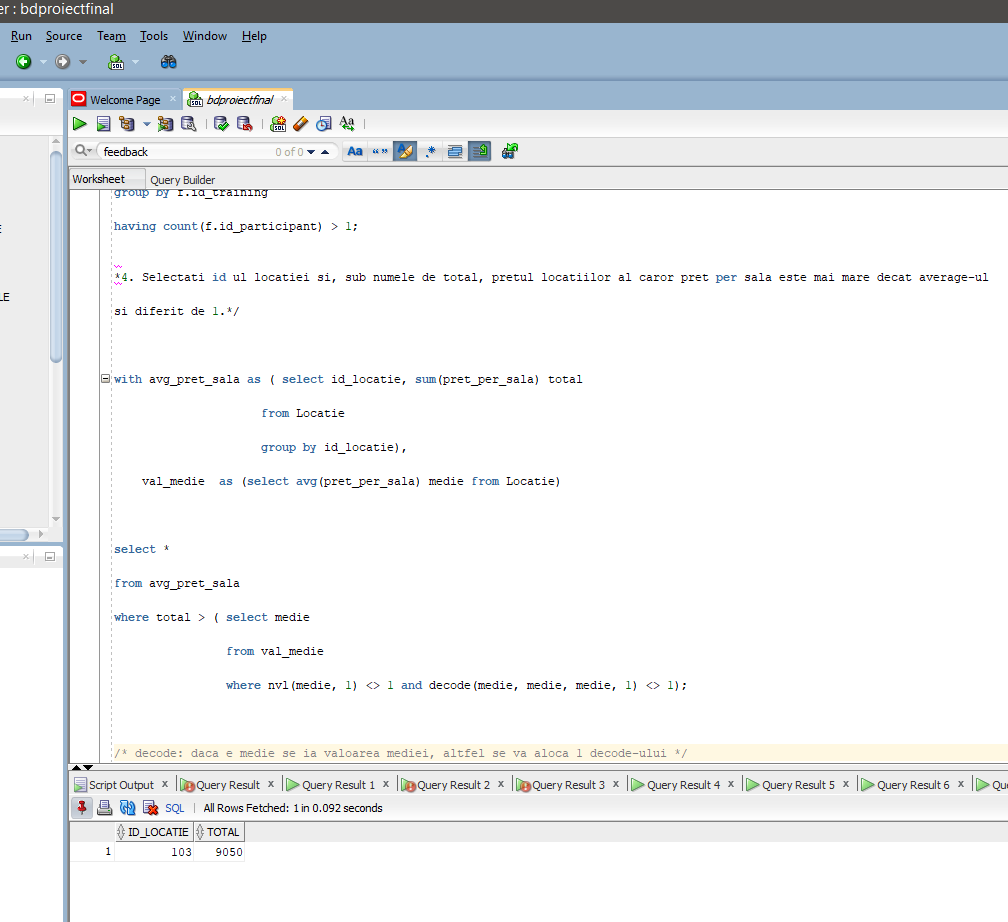
from avg\_pret\_sala

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 \*/



/\*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) >

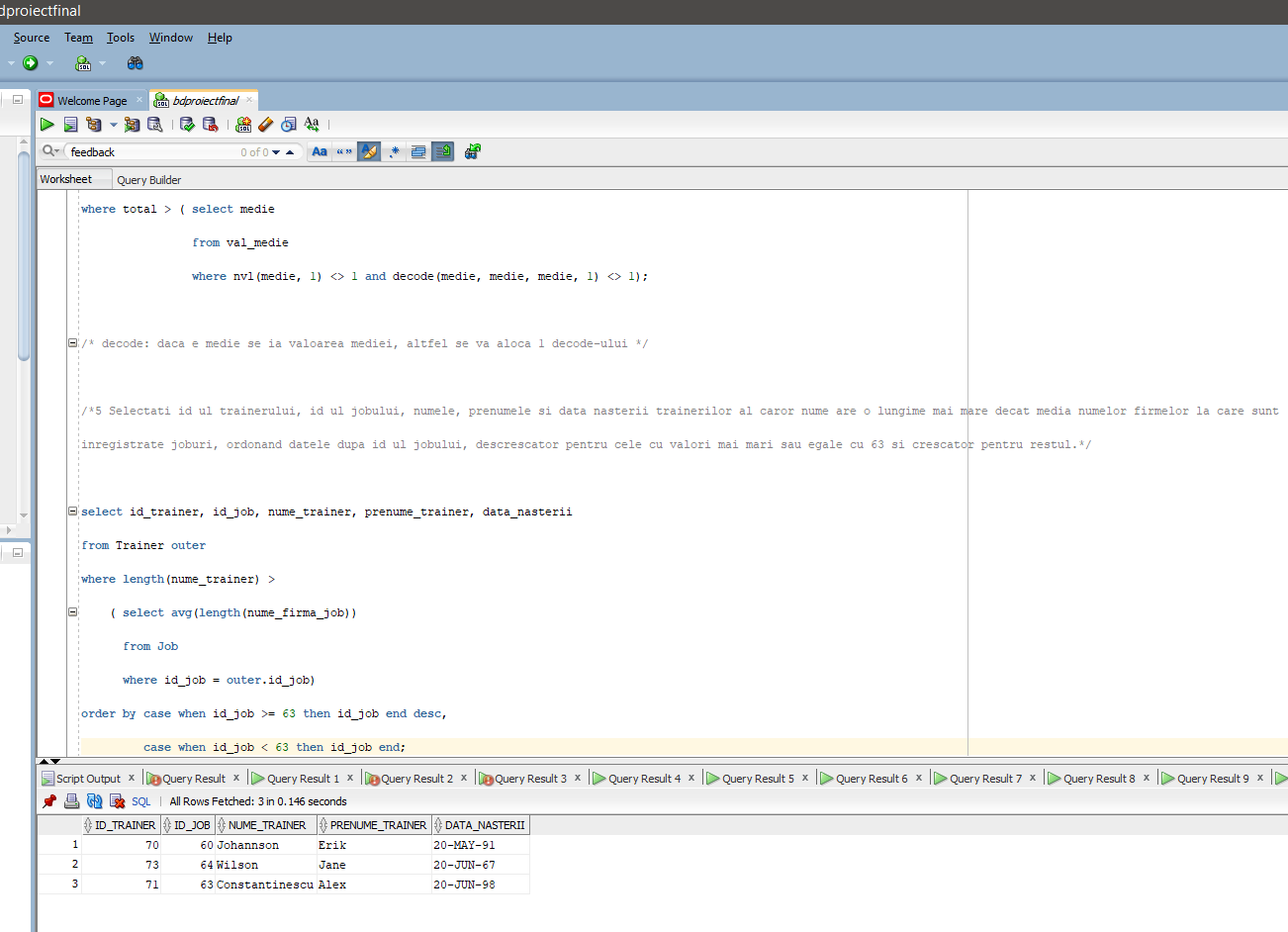
( select avg(length(nume\_firma\_job))

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;



/\*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;

paşii precedenţi 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

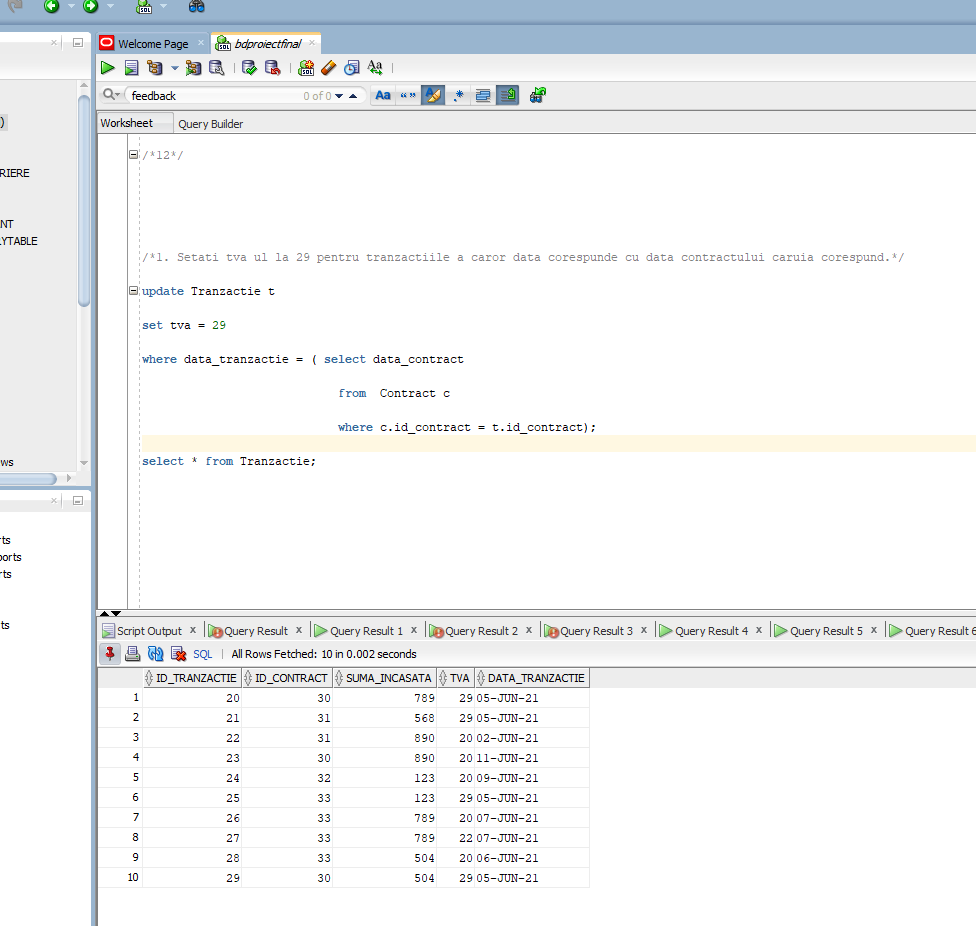
set tva = 29

where data\_tranzactie = ( select data\_contract

from Contract c

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

select \* from Tranzactie;



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

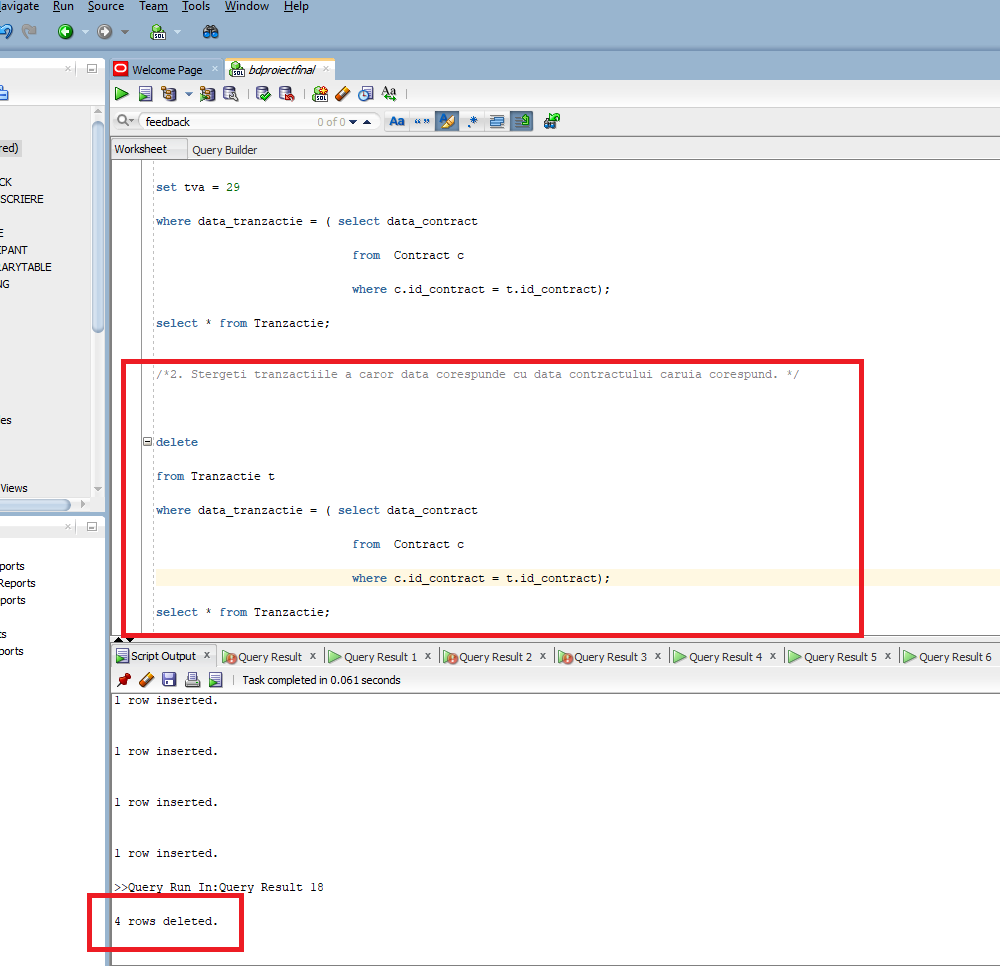
delete

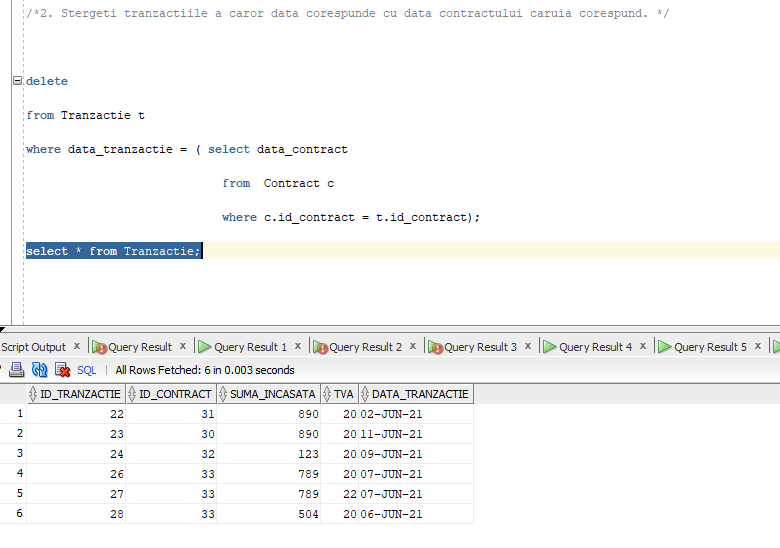
from Tranzactie t

where data\_tranzactie = ( select data\_contract

from Contract c

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





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

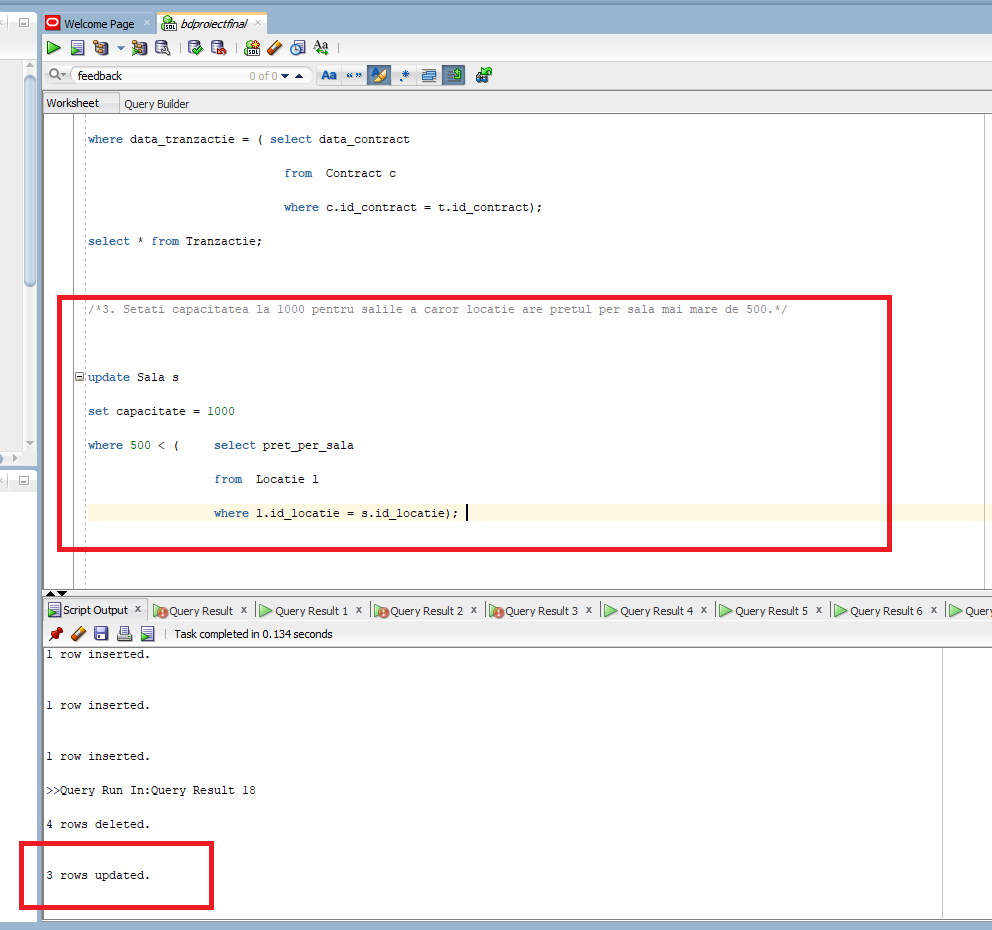
update Sala s

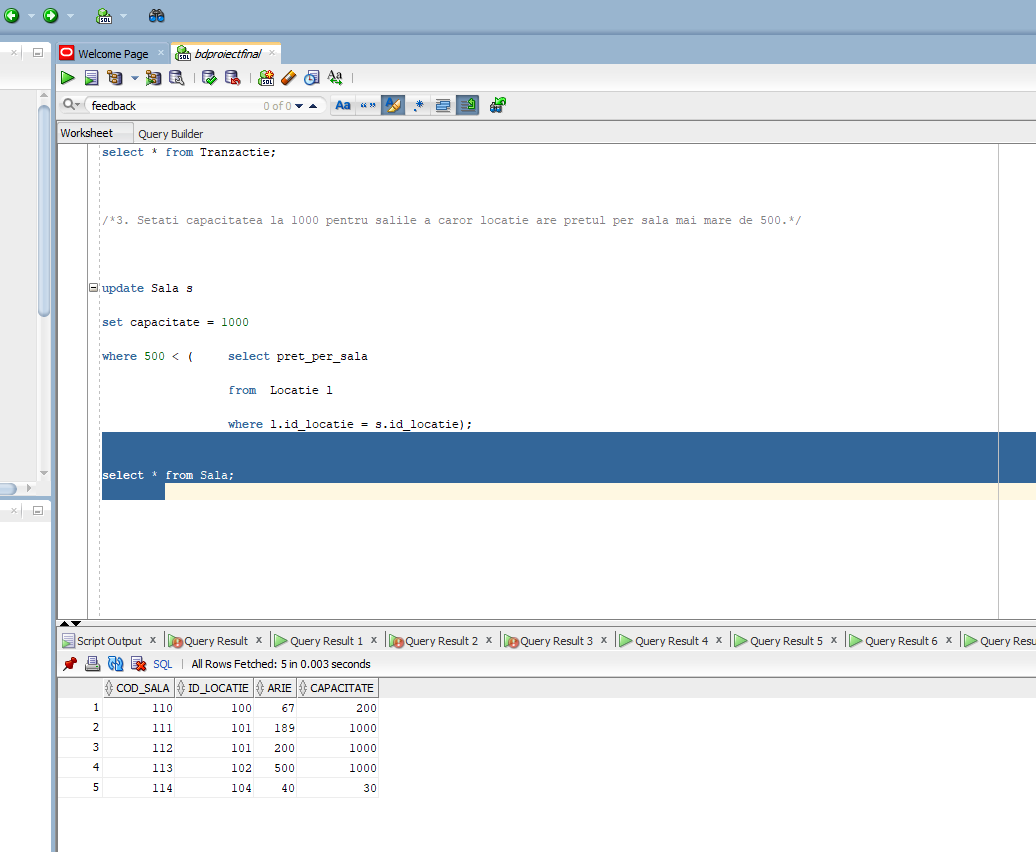
set capacitate = 1000

where 500 < ( select pret\_per\_sala

from Locatie l

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





/\*13\*/

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

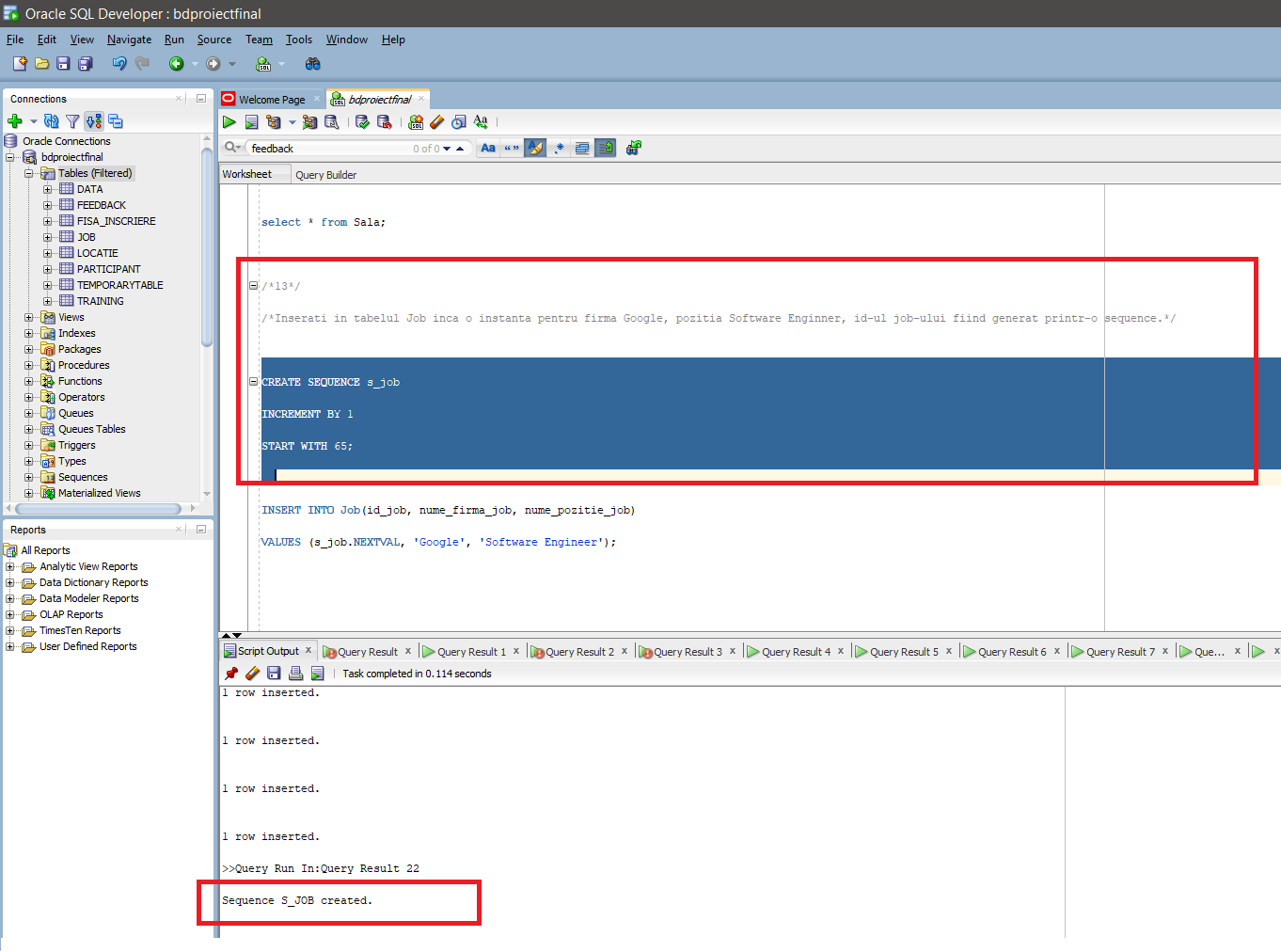
CREATE SEQUENCE s\_job

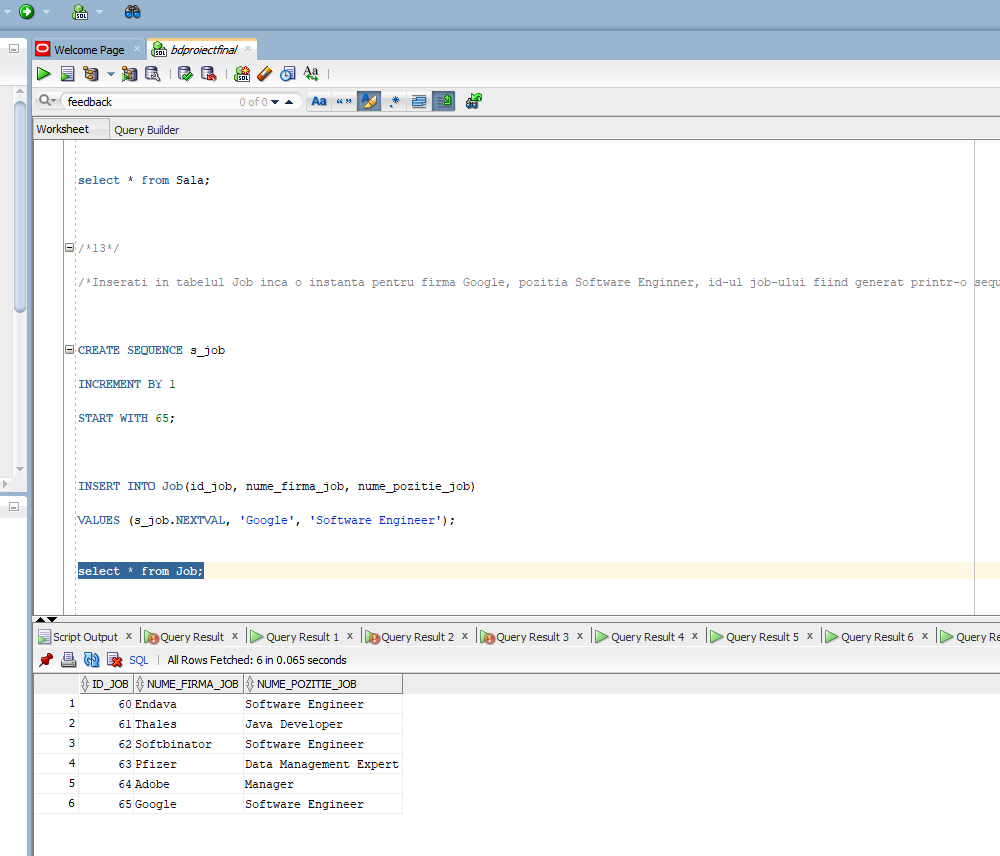
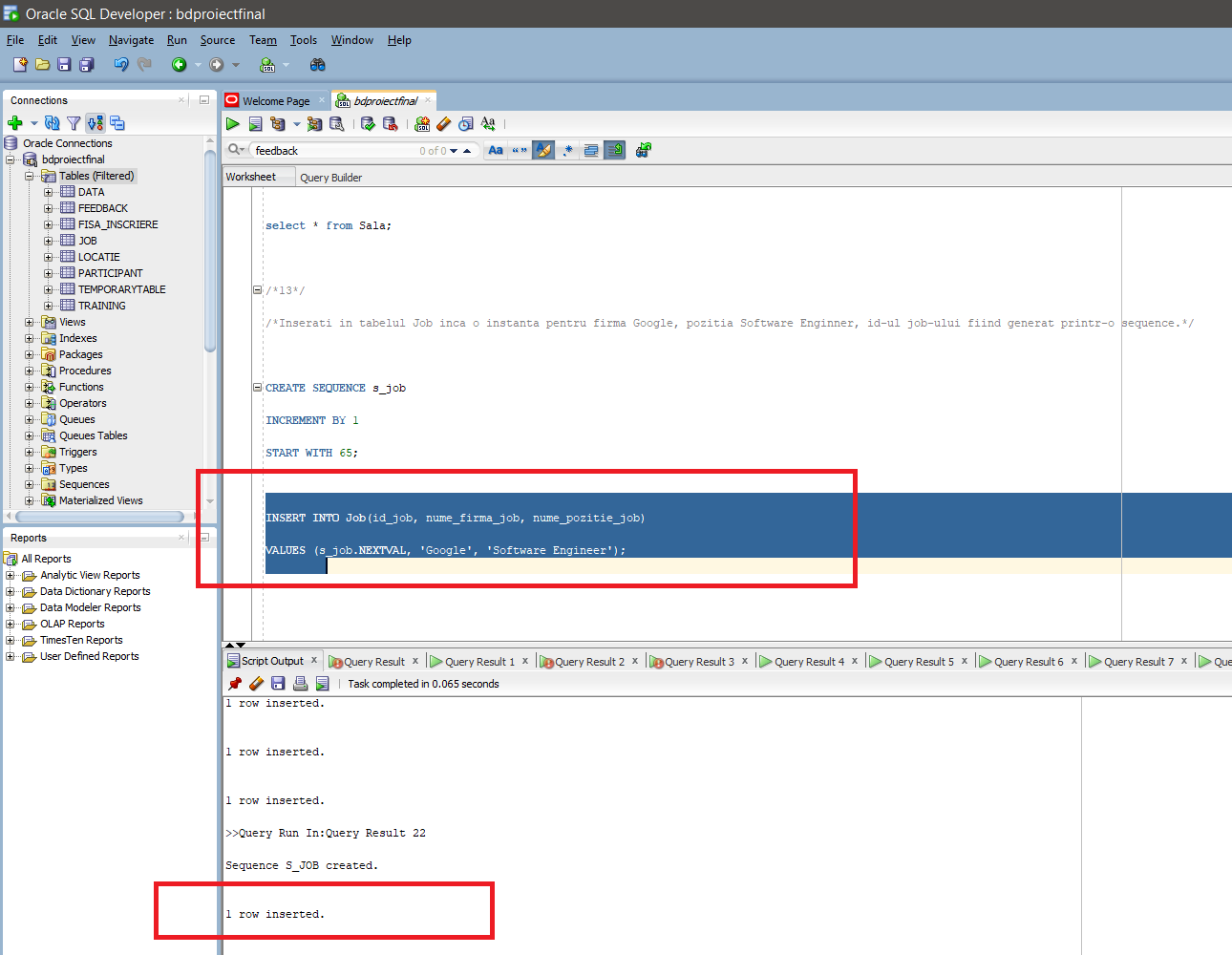
INCREMENT BY 1

START WITH 65;

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

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





/\*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

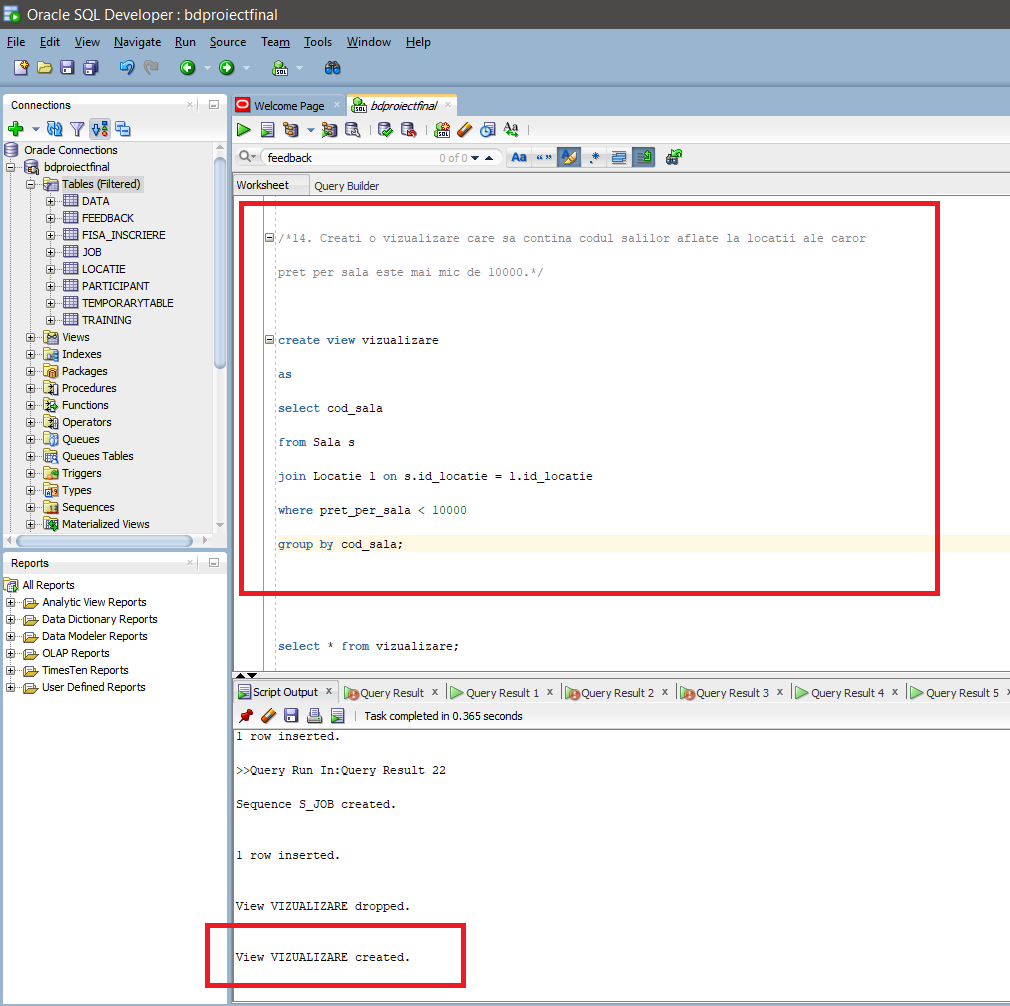
join Locatie l on s.id\_locatie = l.id\_locatie

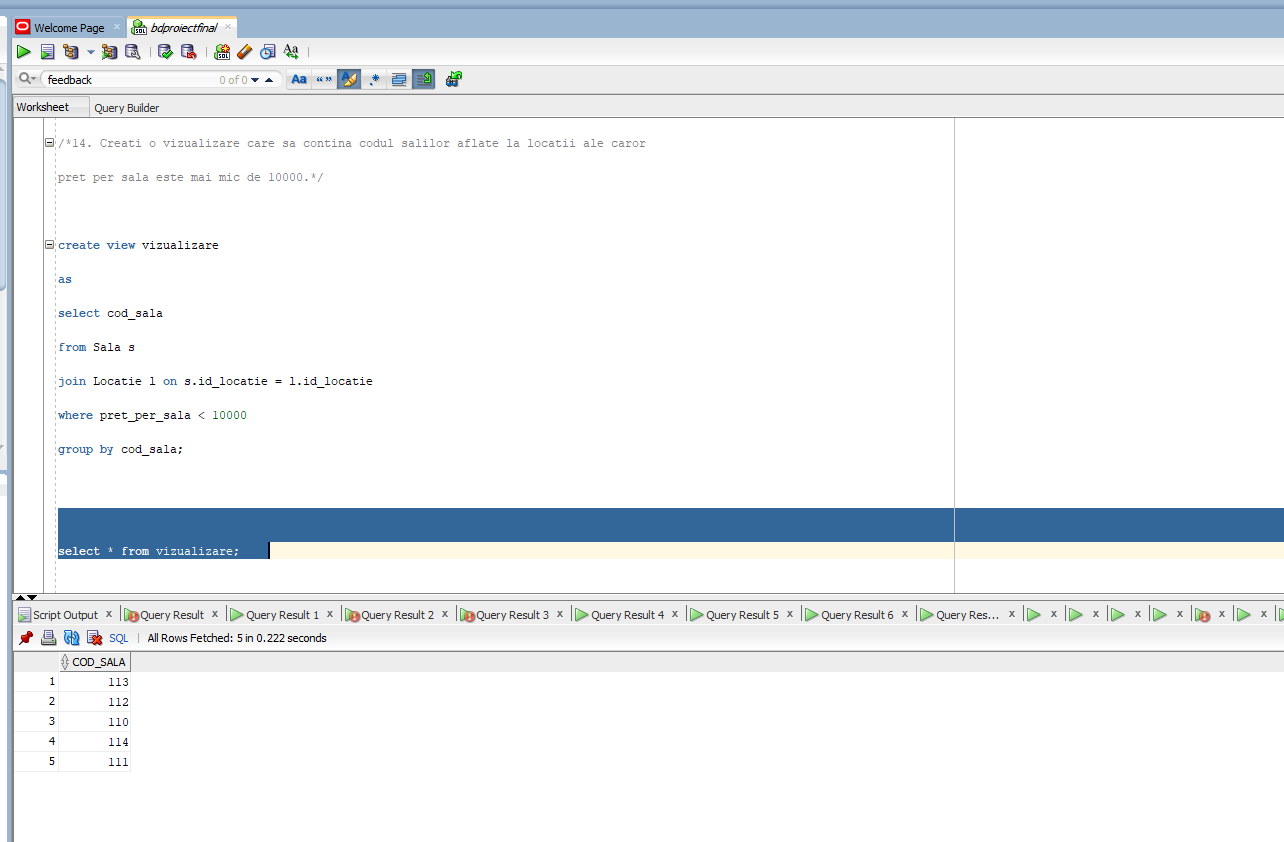
where pret\_per\_sala < 10000

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)

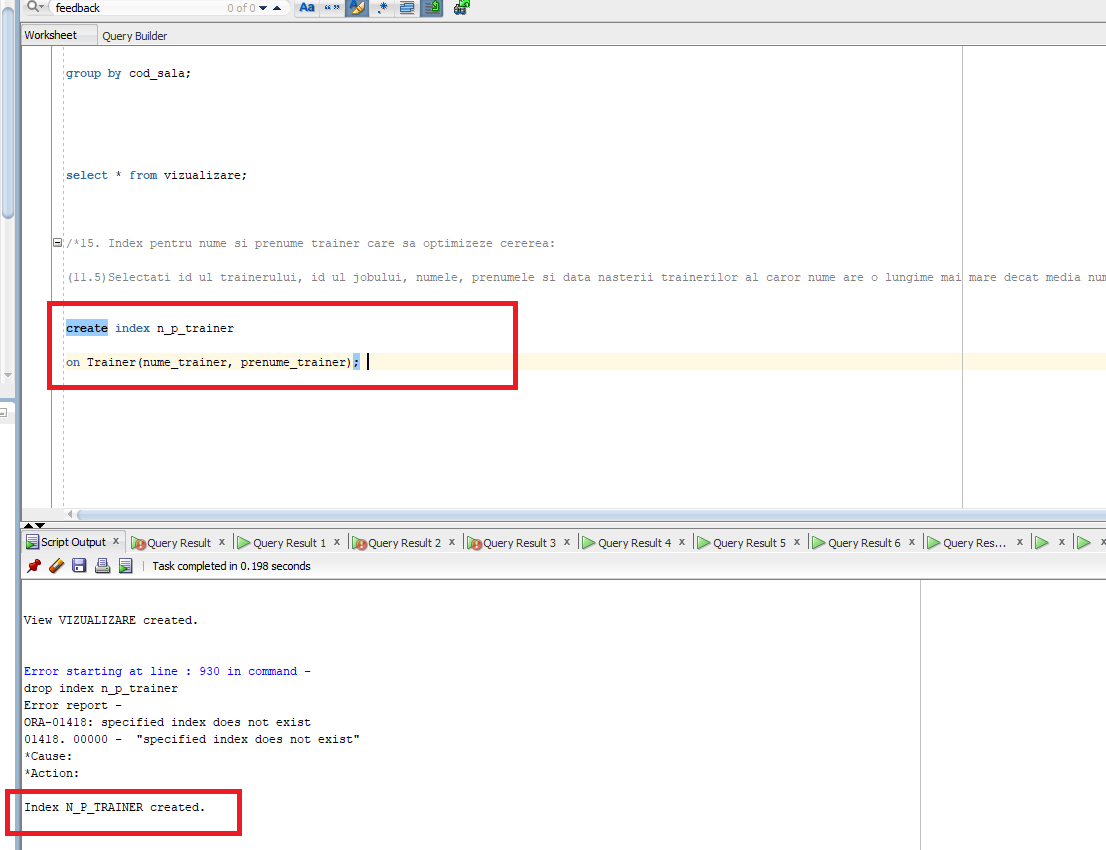
order by case when id\_job >= 63 then id\_job end desc,

case when id\_job < 63 then id\_job end;

\*/

create index n\_p\_trainer

on Trainer(nume\_trainer, prenume\_trainer);



/\*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

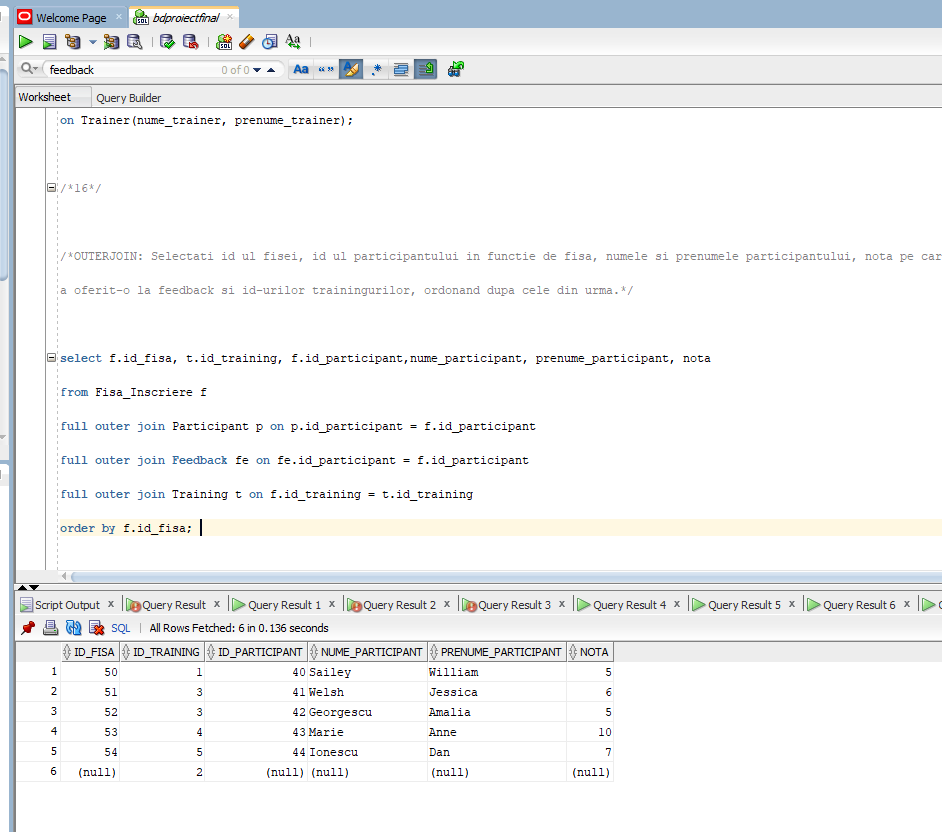
from Fisa\_Inscriere f

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;



/\* 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;

