Maria-Catalina Isache

Grupa 133

Mai 2022

Proiect baze de date

Olimpiada națională

1. Am proiectat baza de date pentru olimpiada națională din România. Aceasta conține informații despre elevii care au participat măcar într-un an la această etapă a olimpiadei, indiferent de materia la care au participat, precum și despre profesorii lor îndrumători, profesorii care i-au însoțit și unde au fost toți aceștia cazați.

Am luat în considerare faptul că un elev poate participa în același an la mai multe materii la olimpiada, de aceea am modelat relațiile în funcție de probele la care participă elevul.

Toate probele din toți anii, la toate clasele și la toate materiile au câte un id atribuit, ce ne ajută sa le identificăm unic. Modelul reține când a avut loc proba, unde s-a desfășurat, la ce materie a fost și pentru ce clasă.

Baza de date ține informații și despre profesorii îndrumători ai elevilor participanți, precum și despre orașul și școala pe care le reprezintă la olimpiadă. Aceasta a fost proiectată astfel încât să păstreze un istoric al participărilor, care să ia în calcul că un elev poate să-și schimbe profesorii, orașul sau școala de la an la an, iar atributele Oras\_id și Scoala\_id ale elevilor se referă la cele curente. De asemenea, am luat în considerare că un profesor poate să predea mai multe materii (ex. TIC și Informatică), că un elev poate avea mai mulți profesor îndrumători la o materie, de unde nevoia de o relație ternală între elev, materie și profesor.

Întrucât elevii dintr-un oraș merg la olimpiada la o materie într-un grup împreună cu niște profesori însoțitori, cazați în același loc de către comisia de organizare a olimpiadei, am creat entitatea lot, pentru a reține în model unde au fost cazați participanții. Fiecare lot din fiecare an, pentru fiecare materie și din fiecare oraș are câte un ID atribuit la introducerea în baza de date.

Acest model ar putea fi utilizat de către organizatorii olimpiadelor naționale pentru accesul la istoricul locațiilor, cazărilor, profesorilor, etc., în vederea organizării viitoarelor olimpiade. De asemenea, baza de date este utilă și în a centraliza performanțele școlare ale elevilor români, putându-se crea diverse statistici pe baza lui. De exemplu, putem să vedem școlile cu cei mai mulți olimpici, olimpicii cu cele mai multe participări la olimpiade, olimpicii cu participări la olimpiade la materii diferite, profesorii din țară care au îndrumat olimpici, etc.

1. Constrângeri:
   * Un elev poate avea o singură școală și un singur oraș de proveniență în baza de date în momentul actual
   * Un elev apare în baza de date doar dacă a participat la cel puțin o olimpiadă
   * Un profesor îndrumător aparține maxim unei școli, iar aceasta nu e neapărat să fie cea a elevilor pe care îi îndrumă, sau poate să nu aibă nicio școală.
   * În cadrul unei participări la o probă a olimpiadei, un elev poate reprezenta un singur oraș și o singură școală.
   * O probă a olimpiadei nu poate fi alcătuită din mai multe evaluări (ex: cum se desfășoară olimpiada de informatică cu probe în două zile diferite – pentru astfel de situații se vor introduce în bază doar datele aferente primei probe împreună cu rezultatul obținut per total).
   * O probă se desfășoară doar într-o locație
   * Un lot este reprezentant pentru un singur oraș la o singură materie și într-un singur an
   * Un întreg lot(alcătuit din elevi și professor) este cazat la o singură cazare.
   * O cazare se află la o singură adresă.
   * O școală se află la o singură adresă
2. Entități:
   * ELEVI
     + are ca atribute: Elev\_id, Nume, Prenume, Clasa, Varsta, Sex
     + conține informații despre elevii participanți la olimpiade
     + cheia primară este reprezentată de Elev\_id, ce reține un ID atribuit la introducerea în baza de date
   * PROFESOR\_INDRUMATORI
     + are ca atribute: Profesor\_id, Nume, Prenume
     + conține numele profesorii îndrumători ai elevilor participanți la olimpiade
     + cheia primară este reprezentată de Elev\_id, ce reține un ID atribuit la introducerea în baza de date
   * SCOLI
     + are ca atribute: Scoala\_id, Nume\_scoala
     + conține numele scolilor elevilor și ale profesorilor îndrumători de olimpici
     + cheia primară este reprezentată de Scoala\_id
   * MATERII
     + are ca atribute: Materie\_id, Nume\_materie, Profil\_materie
     + conține numele și profilul materiilor la care se dă olimpiadă
     + cheia primară este reprezentată de Materie\_id
   * PROBE
     + are ca atribute: Proba\_id, Data, Durata, Clasa
     + conține detalii despre probele olimpiadei; fiecare combinație diferită între materiile olimpiadei, claselor la care este dată și a anilor în care s-a ținut este considerată ca fiind o probă independentă
     + cheia primară este reprezentată de Proba\_id, ce reține un ID atribuit la introducerea în baza de date
   * LOTURI
     + are ca atribute: Lot\_id, Denumire, Nr\_persoane
     + prin această entitate sunt centralizați toți elevii dint-un oraș, indiferent de clasa, participanți într-un anumit an la olimpiada unei singure materii, alături de profesorii care i-au însoțit; de asemenea, prin aceasta se identifică unde au fost cazați elevii și profesorii în fiecare an;
     + cheia primara este reprezentată de Lod\_id, ce reține un ID atribuit la introducerea în baza de date
   * PROFESORI\_INSOTITORI
     + are ca atribute: Profesor\_id, Nume, Prenume, Sex, Nr\_telefon
     + conține informații despre profesorii însoțitori ai elevilor participanți, mai precis ale loturilor fiecărui oraș, pentru fiecare an și materie; aceștia sunt cazați alături de elevii de care au grijă, făcând practic parte din lot;
     + cheia primară este reprezentată de Profesor\_id, ce reține un ID atribuit la introducerea în baza de date
   * ORASE
     + are ca atribute: Oras\_id, Nume\_oraș, Populație
     + prin această entitate se identifică ce oraș a reprezentat fiecare student, respectiv fiecare lot la olimpiadă și totodată contribuie la descrierea completă a locațiilor stocate în baza de date
     + cheia primară este reprezentată de Oras\_id,
   * LOCATII
     + are ca atribute: Locatie\_id, Strada, Nr\_strada, Cod\_postal
     + stochează diferitele adrese aferente altor entități din baza de date: adresa probelor, cea a cazărilor și cea a școlilor
     + cheia primară este reprezentată de Locatie\_id,
   * CAZARI
     + are ca atribute: Cazare\_id, Nume\_cazare, Capacitate
     + stochează informații despre pensiunile/hotelurile la care au fost cazate loturile olimpice de-a lungul anilor, la toate materiile
     + cheia primară este reprezentată de Cazare\_id,
   * CAMERE
     + are ca atribute: Cazare\_id, Nr\_camera, Capacitate
     + stochează informații despre camerele pensiunilor/hotelurilor la care au fost cazate loturile olimpice de-a lungul anilor, la toate materiile
     + cheia primară este reprezentată de Nr\_camera+Cazare\_id
3. Relații
   * Elev aparține\_de Scoala
     + se referă la școala curentă la care studiază elevul
     + cardinalitate: M(0):1 (mai multi elevi, sau chiar niciunul, pot apartine aceleeași școli, însă un elev poate studia doar la o școală la un anumit moment în timp)
   * Elev provine\_din Oras
     + se referă la orașul de proveniență al elevului
     + cardinalitate: M(0):1 (mai multi elevi, sau chiar niciunul, pot proveni dintr-un anumit oraș, însă un elev poate proveni dintr-un singur oraș)
   * Elev are Profesor\_Indrumator la Materie (într-un anumit an)
     + este o relație ternală ce descrie pentru un elev cine i-a fost profesor îndrumător la o materie la care a participat la olimpiadă;
     + este nevoie de o astfel de relație întrucât un profesor poate preda mai multe materii la diferiți studenți, iar un elev poate participa la mai multe materii la olimpiadă și poate avea mai mulți profesori îndrumători la aceeași materie, de la an la an.
     + cardinalitate: M(0):M(1):M(1)
   * Elev participa la Proba, reprezentand un Oras si o Scoala (și obține un premiu)
     + este o relație între 4 entități ce descrie participarea la o olimpiadă a unui elev; proba este la o singură materie, într-un singur an și la o singură clasa, iar elevul poate reprezenta la o probă doar un oraș și o școală
     + cardinalitate: M(1):M(1):1:1
   * Profesor\_indrumator aparține\_de Scoala
     + se referă la școala curentă la care predă profesorul
     + cardinalitate: M(0):1 (mai multi profesori, sau chiar niciunul, pot preda la aceeași școală, însă un profesor poate preda doar la o școală la un anumit moment în timp)
   * Proba data\_la Materie
     + relevă materia la care este dată o probă a olimpiadei
     + cardinalitate: M(0):1 (fiecare probă trebuie să aibă o singură materie la care să fie dată, iar la o materie se pod da mai multe probe (ani diferiți, clase diferite) sau chiar niciuna
   * Proba are Locație
     + determină locația la care se desfășoară proba
     + cardinalitate: M(0):1 (orice probă se desfășoară la o locație, iar la o locație pot avea loc mai multe probe sau niciuna)
   * Locație aflata\_in Oras
     + completează informațiile locației cu orașul în care aceasta se află
     + cardinalitate: 1:M(0) (orice locație se află într-un oraș, iar un oraș poate avea mai multe locații sau niciuna)
   * Scoala are Locație
     + determină locația la care se află școala
     + cardinalitate: 1(0):1 (orice școală se află la o locație, iar la o locație se poate afla o singură școală sau niciuna)
   * Cazare are Locatie
     + determină locația la care se află proba
     + cardinalitate: 1(0):1 (orice cazare are o locație, iar la o locație poate fi maxim o cazare sau niciuna)
   * Lot reprezintă Oras
     + determină orașul pe care îl reprezintă lotul la olimpiadă
     + cardinalitate: M(0):1 (un lot reprezintă un oraș, iar un oraș poate fi reprezentat de mai multe loturi sau niciunul)
   * Profesor\_insotitor insoteste Lot
     + determină profesorii care însoțesc și care fac practic parte din lot
     + cardinalitate: M(1):M(0) (un lot poate fi însoțit de mai mulți profesori, dar măcar de unul, iar un profesor poate însoți mai multe loturi sau chiar niciunul)
   * Lot cazat\_la Cazare
     + determină la ce pensiuni/hoteluri au fost cazate loturile participante la olimpiadă
     + cardinalitate: M(0):1 (un lot este cazat la o singură cazare, iar la o cazare pot fi cazate mai multe loturi sau chiar niciunul)
   * Lot participa\_la Materie
     + determină pentru la ce materie participă loturile la olimpiadă
     + cardinalitate: M(0):1 (un lot participă la olimpiadă la o materie, iar la o materie participă mai multe loturi, aferente orașelor participante, sau chiar niciunul)
   * Camera situata\_in Cazare
     + determină de ce camere dispune o cazare
     + cardinalitate: M(1):1 (o cazare are mai multe camera și măcar un, iar o cameră face parte dintr-o cazare)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Entitate | Atribut | Descriere | Tip de date | Constrângeri | Valori implicite | Valori posibile |
| ELEVI | Elev\_id | Un cod unic pentru fiecare elev | int | Not null, unic |  | 1-100000 |
| Nume | Numele elevului | varchar | Max 25 caractere, not null |  | Orice string |
| Prenume | Prenumele elevului | varchar | Max 25 caractere |  | Orice string |
| Data\_nastere | Data de nastere a elevului | date | <sysdate |  | 01-01-1980 - sysdate |
| Sex | Daca elevul este fată sau băiat | char | Poate fi doar F sau M |  | F,M |
| PROFESORI\_  INDRUMATORI | Profesor\_id | Un cod unic pentru fiecare profesor indrumator | int | Not null, unic |  | 1-100000 |
| Nume | Numele profesorului indrumator | varchar | Max 25 caractere, not null |  | Orice string |
| Prenume | Prenumele profesorului indrumator | varchar | Max 25 caractere |  | Orice string |
| SCOLI | Scoala\_id | Un cod unic pentru fiecare scoala | int | Not null, unic |  | 1-100000 |
| Nume\_scoala | Numele scolii | varchar | Max 50 caractere, not null |  | Orice string |
| MATERII | Materie\_id | Un cod unic pentru fiecare materie | int | Not null, unic |  | 1-100000 |
| Nume\_materie | Numele materiei | varchar | Max 25 caractere, not null |  | Matematica, informatica, etc. |
| PROBE | Proba\_id | Un cod unic pentru fiecare proba | int | Not null, unic |  | 1-100000 |
| Data | Data desfasurarii probei | date |  | sysdate | Orice data |
| Durata | Cat dureaza proba in minute | int | Numar>0 |  | 60-400 |
| Clasa | Clasa pentru care se da proba | int | Not null, doar valori intre 5-12 |  | 5-12 |
| ORASE | Oras\_id | Un cod unic pentru fiecare oras | int | Not null, unic |  | 1-50 |
| Nume\_oras | Numele orasului | varchar | Max 25 caractere, not null |  | Orice string |
| LOTURI | Lot\_id | Un cod unic pentru fiecare lot | int | Not null, unic |  | 1-100000 |
| Denumire | Numele lotului compus din orasul pe care-l reprezinta, materia si anul | varchar | Max 25 caractere, not null |  | Orice string cu formatul `oras\_materie\_an’ |
| LOCATII | Locatie\_id | Un cod unic pentru fiecare locatie | int | Not null, unic |  | 1-100000 |
| Strada | Numele strazii | varchar | Max 50 caractere |  | Orice string |
| Nr\_strada | Numarul strazii | int | Numar>0 |  | 1-300 |
| Cod\_postal | Codul postal al locatiei | varchar | String de cifre |  | Orice sting de forma `010001’ |
| PROFESORI\_  INSOTITORI | Profesor\_id | Un cod unic pentru fiecare profesor insotitor | int | Not null, unic |  | 1-100000 |
| Nume | Numele profesorului insotitor | varchar | Max 25 caractere, not null |  | Orice string |
| Prenume | Prenumele profesorului insotitor | varchar | Max 25 caractere |  | Orice string |
| Nr\_telefon | Numarul de telefon al profesorului | char | 10 caractere, cifre |  | Orice nr in formatul ‘0749469685’ |
| CAZARI | Cazare\_id | Un cod unic pentru fiecare cazare | int | Not null, unic |  | 1-100000 |
| Nume\_cazare | Numele cazarii | varchar | Max 100 caractere |  | Orice string |
| Capacitate | Capacitatea totala a cazarii | int | Numar>0 |  | 1-1000 |
| CAMERE | Nr\_camera | Numarul camerei | int | Not null |  | 1-1000 |
| Cazare\_id | ID-ul cazarii in care se afla camera | int | Not null |  | 1-100000 |
| Capacitate | Capacitatea camerei | int | Numar>0 |  | 1-10 |

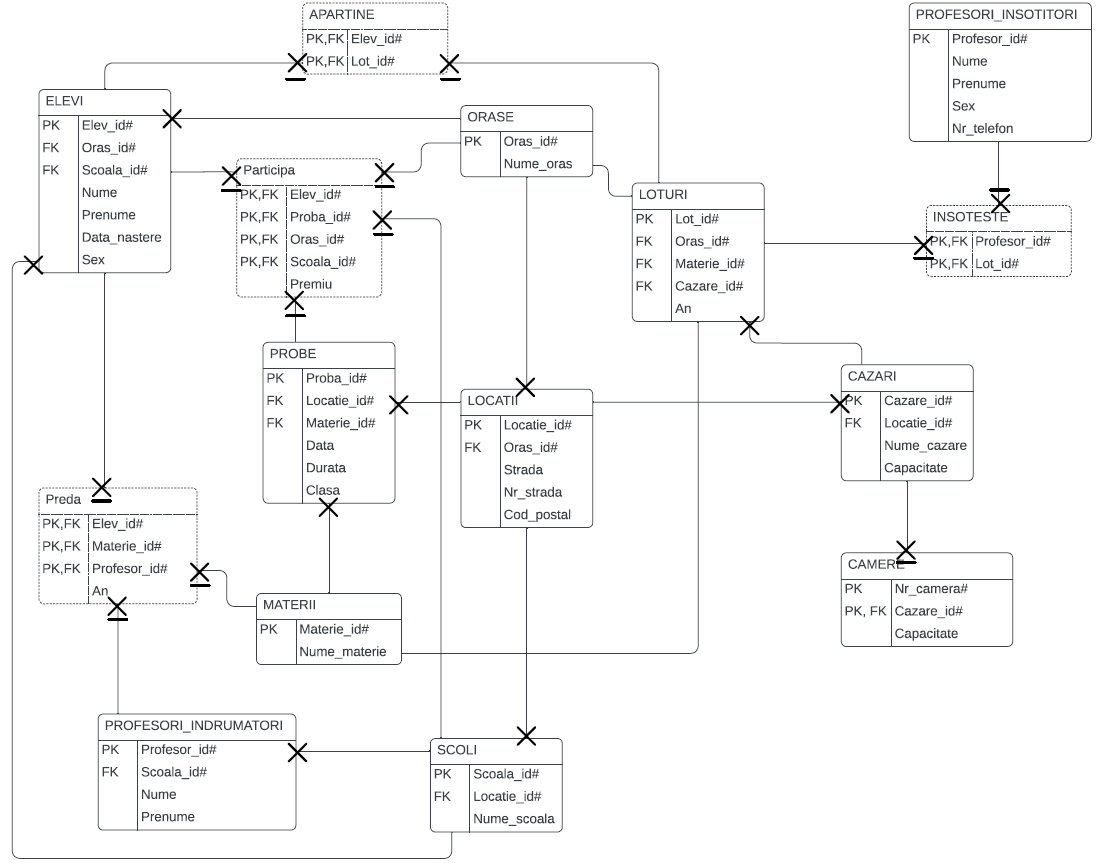


Diagrama conceptuală

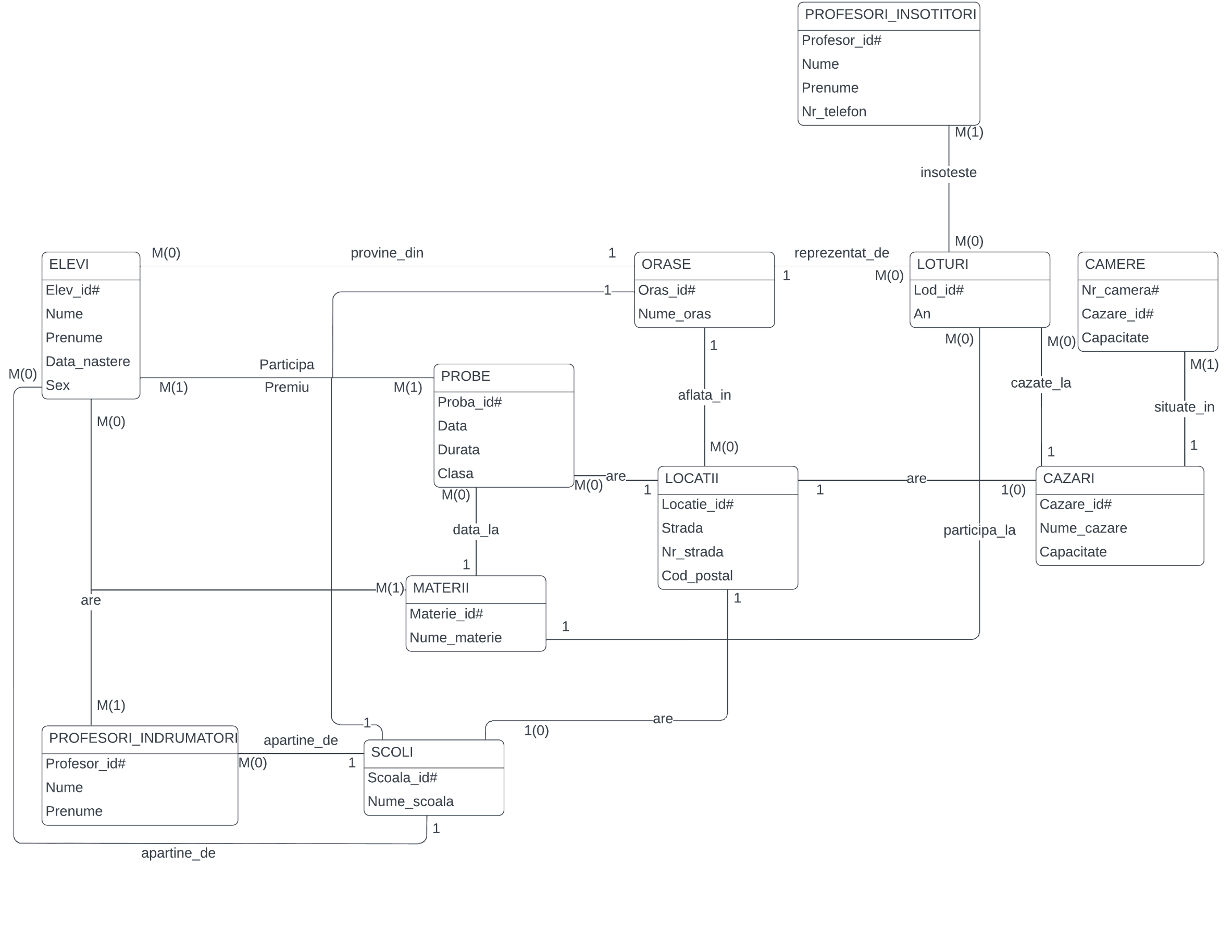


Diagrama E/R

1. Scheme relaționale
   * ELEVI (elev\_id#, oras\_id#, scoala\_id#, nume, prenume, data\_nastere, sex)
   * PREDA (elev\_id#, materie\_id#, profesor\_id#, an)
   * PROFESORI\_INDRUMATORI (profesor\_id#, scoala\_id#,nume, prenume)
   * SCOLI (scoala\_id#, locatie\_id#, nume\_scoala)
   * MATERII (materie\_id#, nume\_materie)
   * PROBE (proba\_id,# locatie\_id#, materie\_id#, data, durata, clasa)
   * PARTICIPA (elev\_id#, proba\_id#, oras\_id#, scoala\_id#, premiu)
   * APARTINE (elev\_id#, lot\_id#)
   * ORASE (oras\_id#, nume\_oras)
   * LOTURI (lot\_id,# oras\_id#, materie\_id#, cazare\_id#, denumire)
   * LOCATII (locatie\_id#, oras\_id#, strada, nr\_strada, cod\_postal)
   * PROFESORI\_INSOTITORI (profesor\_id#, nume, prenume, nr\_telefon)
   * INSOTESTE (profesor\_id#, lot\_id#)
   * CAZARI (cazare\_id#, locatie\_id#, nume\_cazare, capacitate)
   * CAMERE (nr\_camera#, cazare\_id#, capacitate)
2. Normalizare
   * non-FN1:

Diagrama concepută nu ar fi în FN1 daca ar avea un atribut divizibil, de exemplu, dacă în loc de 2 atribute, nume și prenume, am avea doar “nume\_complet” pentru Elevi;

ELEVI (elev\_id#, oras\_id#, scoala\_id#, ***nume\_complet***, data\_nastere, sex)

* + non-FN2:

Diagrama concepută nu ar fi în FN2 dacă am avea o dependență funcțională parțială, de ecemplu, dacă tabelul Participă am avea atributele an sau clasă, întrucât acestea sunt determinabile prin intermediul cheii “proba\_id”;

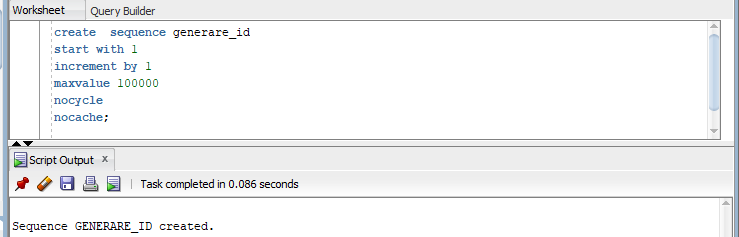
PARTICIPA (elev\_id#, proba\_id#, oras\_id#, scoala\_id#, premiu, ***an, clasa***)

* + non-FN3:

Diagrama concepută nu ar fi în FN3 dacă am avea o dependență tranzitivă, de exemplu, dacă în tabelul Elevi am avea și numele școlii, pe lângă ID-ul școlii.

* + ELEVI (elev\_id#, oras\_id#, ***scoala\_id#, nume\_scoala***, nume, prenume, data\_nastere, sex)

1. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).



create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

1. Crearea tabelelor în SQLși inserarea de date coerente în fiecare dintre acestea.
   * ELEVI

create table ELEVI (

elev\_id number(5),

nume varchar(25) constraint not\_null\_nume not null,

prenume varchar(25) constraint not\_null\_prenume not null,

data\_nastere date,

sex char(1),

scoala\_id number(5),

oras\_id number(2),

constraint pk\_elevi primary key(elev\_id)

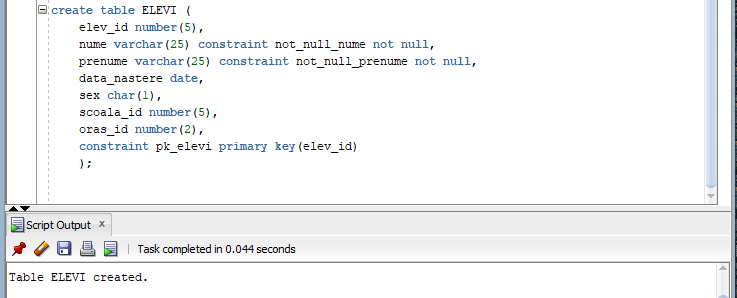
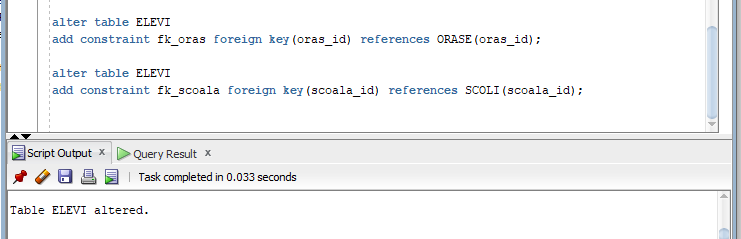
);

alter table ELEVI

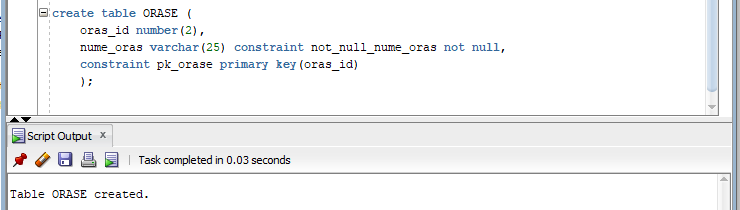
add constraint fk\_oras foreign key(oras\_id) references ORASE(oras\_id);

alter table ELEVI

add constraint fk\_scoala foreign key(scoala\_id) references SCOLI(scoala\_id);



* + ORASE



create table ORASE (

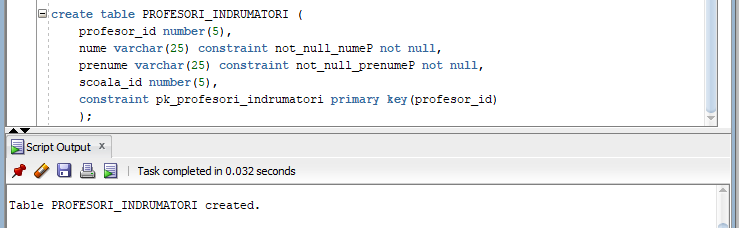
oras\_id number(2),

nume\_oras varchar(25) constraint not\_null\_nume\_oras not null,

constraint pk\_orase primary key(oras\_id)

);

* + PROFESORI\_INDRUMATORI



create table PROFESORI\_INDRUMATORI (

profesor\_id number(5),

nume varchar(25) constraint not\_null\_numeP not null,

prenume varchar(25) constraint not\_null\_prenumeP not null,

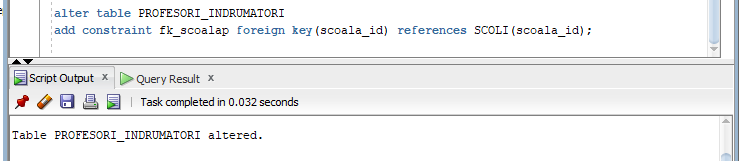
scoala\_id number(5),

constraint pk\_profesori\_indrumatori primary key(profesor\_id)

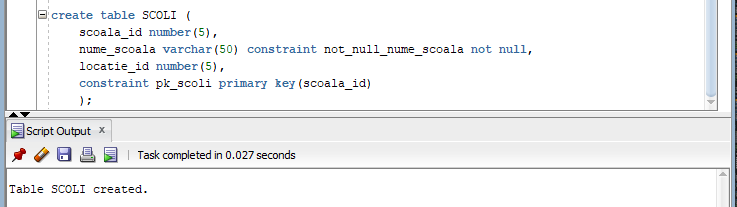
);

alter table PROFESORI\_INDRUMATORI

add constraint fk\_scoalap foreign key(scoala\_id) references SCOLI(scoala\_id);



* + SCOLI



create table SCOLI (

scoala\_id number(5),

nume\_scoala varchar(50) constraint not\_null\_nume\_scoala not null,

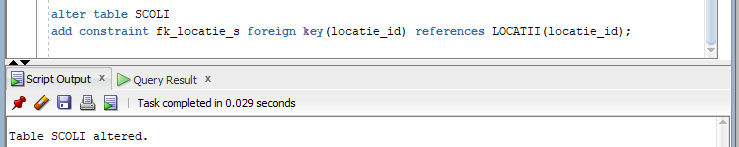
locatie\_id number(5),

constraint pk\_scoli primary key(scoala\_id)

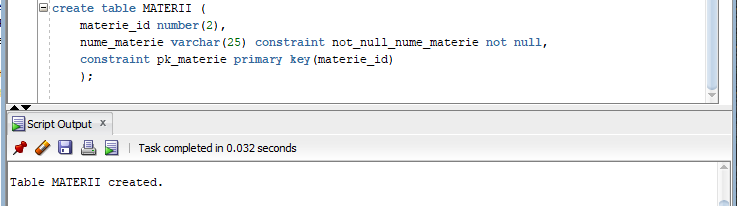
);

alter table SCOLI

add constraint fk\_locatie\_s foreign key(locatie\_id) references LOCATII(locatie\_id);



* + MATERII



create table MATERII (

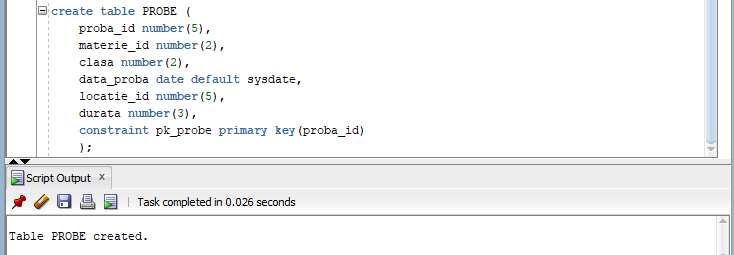
materie\_id number(2),

nume\_materie varchar(25) constraint not\_null\_nume\_materie not null,

constraint pk\_materie primary key(materie\_id)

);

* + PROBE



create table PROBE (

proba\_id number(5),

materie\_id number(2),

clasa number(2),

data\_proba date default sysdate,

locatie\_id number(5),

durata number(3),

constraint pk\_probe primary key(proba\_id)

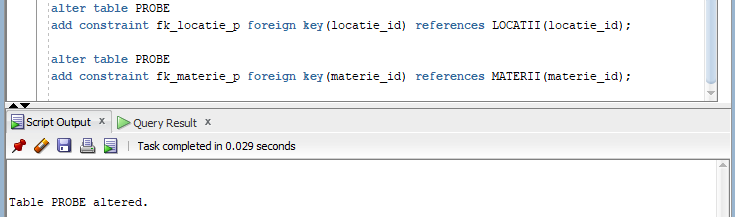
);

alter table PROBE

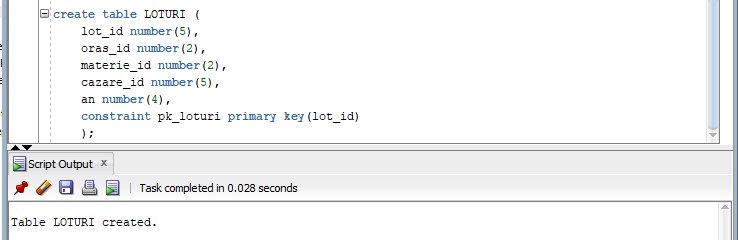
add constraint fk\_locatie\_p foreign key(locatie\_id) references LOCATII(locatie\_id);

alter table PROBE

add constraint fk\_materie\_p foreign key(materie\_id) references MATERII(materie\_id);



* + LOTURI



create table LOTURI (

lot\_id number(5),

oras\_id number(2),

materie\_id number(2),

cazare\_id number(5),

an number(4),

constraint pk\_loturi primary key(lot\_id)

);

alter table LOTURI

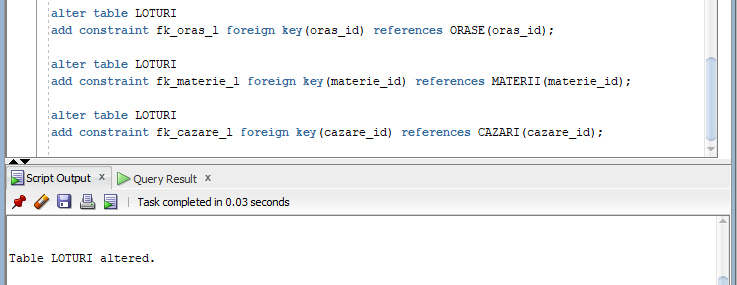
add constraint fk\_oras\_l foreign key(oras\_id) references ORASE(oras\_id);

alter table LOTURI

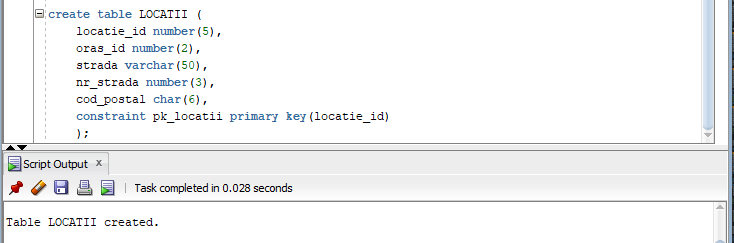
add constraint fk\_materie\_l foreign key(materie\_id) references MATERII(materie\_id);

alter table LOTURI

add constraint fk\_cazare\_l foreign key(cazare\_id) references CAZARI(cazare\_id);



* + LOCATII



create table LOCATII (

locatie\_id number(5),

oras\_id number(2),

strada varchar(50),

nr\_strada number(3),

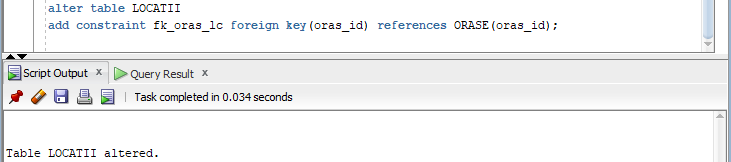
cod\_postal char(6),

constraint pk\_locatii primary key(locatie\_id)

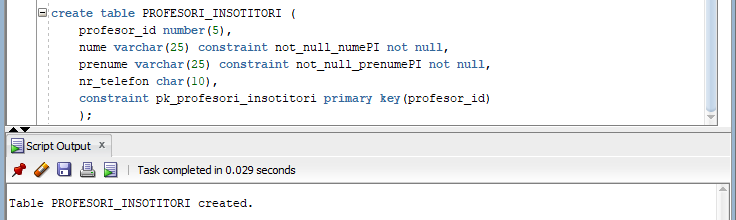
);

alter table LOCATII

add constraint fk\_oras\_lc foreign key(oras\_id) references ORASE(oras\_id);



* + PROFESORI\_INSOTITORI



create table PROFESORI\_INSOTITORI (

profesor\_id number(5),

nume varchar(25) constraint not\_null\_numePI not null,

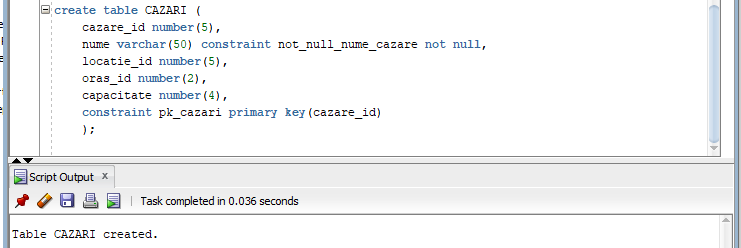
prenume varchar(25) constraint not\_null\_prenumePI not null,

nr\_telefon char(10),

constraint pk\_profesori\_insotitori primary key(profesor\_id)

);

* + CAZARI



create table CAZARI (

cazare\_id number(5),

nume varchar(50) constraint not\_null\_nume\_cazare not null,

locatie\_id number(5),

oras\_id number(2),

capacitate number(4),

constraint pk\_cazari primary key(cazare\_id)

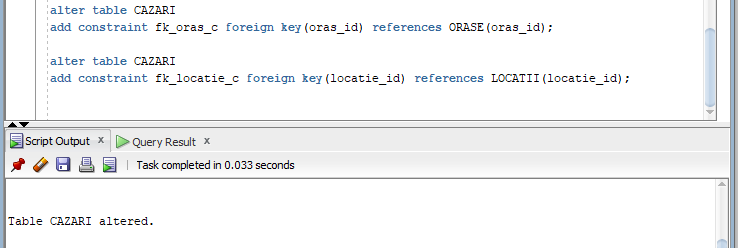
);

alter table CAZARI

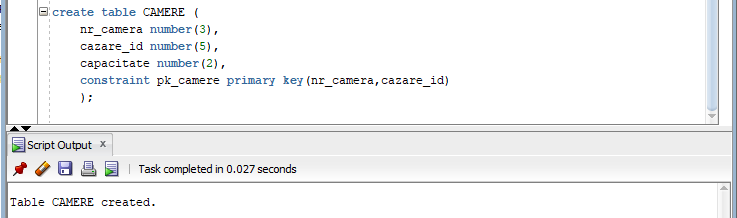
add constraint fk\_oras\_c foreign key(oras\_id) references ORASE(oras\_id);

alter table CAZARI

add constraint fk\_locatie\_c foreign key(locatie\_id) references LOCATII(locatie\_id);



* + CAMERE



create table CAMERE (

nr\_camera number(3),

cazare\_id number(5),

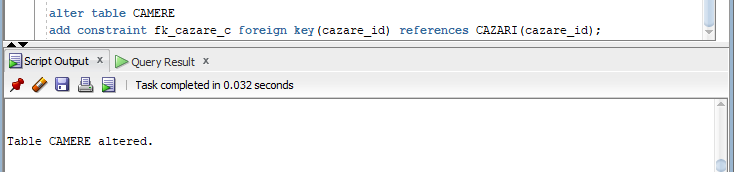
capacitate number(2),

constraint pk\_camere primary key(nr\_camera,cazare\_id)

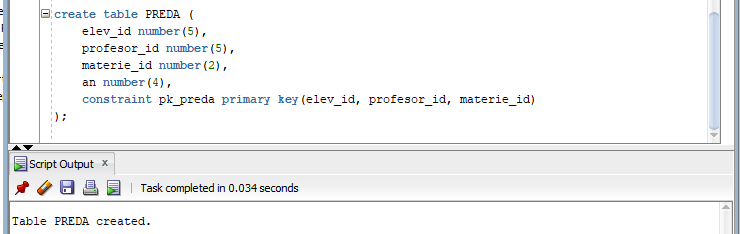
);

alter table CAMERE

add constraint fk\_cazare\_c foreign key(cazare\_id) references CAZARI(cazare\_id);



* + PREDA



create table PREDA (

elev\_id number(5),

profesor\_id number(5),

materie\_id number(2),

an number(4),

constraint pk\_preda primary key(elev\_id, profesor\_id, materie\_id)

);

alter table PREDA

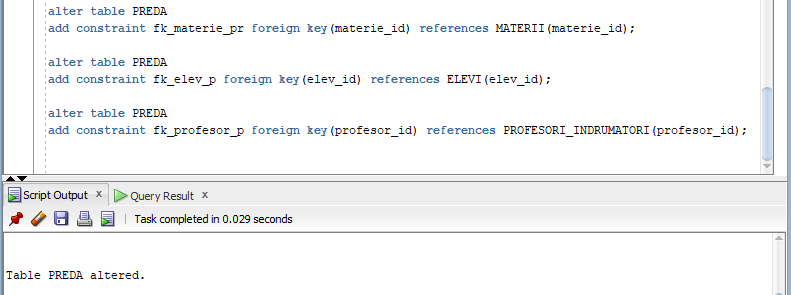
add constraint fk\_materie\_pr foreign key(materie\_id) references MATERII(materie\_id);

alter table PREDA

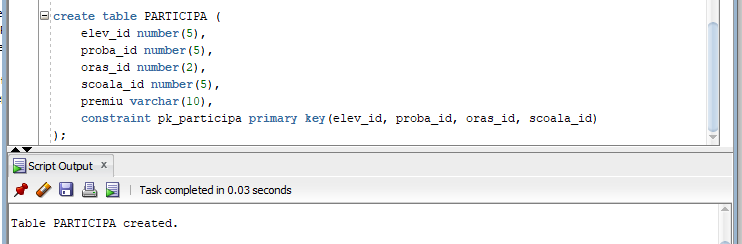
add constraint fk\_elev\_p foreign key(elev\_id) references ELEVI(elev\_id);

alter table PREDA

add constraint fk\_profesor\_p foreign key(profesor\_id) references PROFESORI\_INDRUMATORI(profesor\_id);



* + PARTICIPA



create table PARTICIPA (

elev\_id number(5),

proba\_id number(5),

oras\_id number(2),

scoala\_id number(5),

premiu varchar(10),

constraint pk\_participa primary key(elev\_id, proba\_id, oras\_id, scoala\_id)

);

alter table PARTICIPA

add constraint fk\_elev\_pp foreign key(elev\_id) references ELEVI(elev\_id);

alter table PARTICIPA

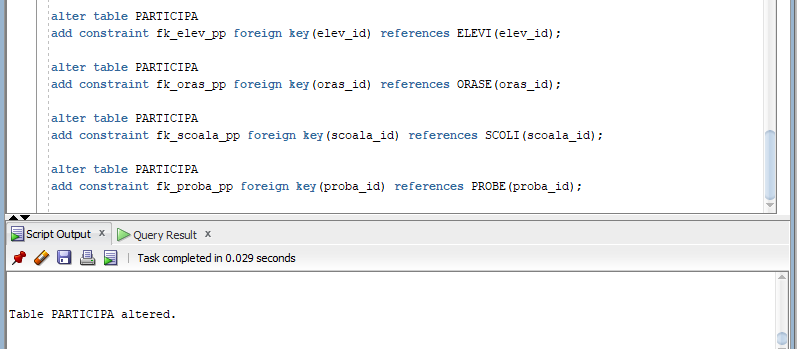
add constraint fk\_oras\_pp foreign key(oras\_id) references ORASE(oras\_id);

alter table PARTICIPA

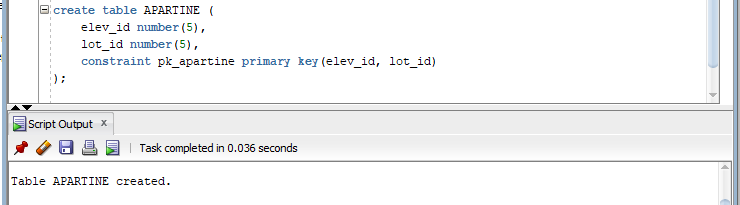
add constraint fk\_scoala\_pp foreign key(scoala\_id) references SCOLI(scoala\_id);

alter table PARTICIPA

add constraint fk\_proba\_pp foreign key(proba\_id) references PROBE(proba\_id);



* + APARTINE



create table APARTINE (

elev\_id number(5),

lot\_id number(5),

constraint pk\_apartine primary key(elev\_id, lot\_id)

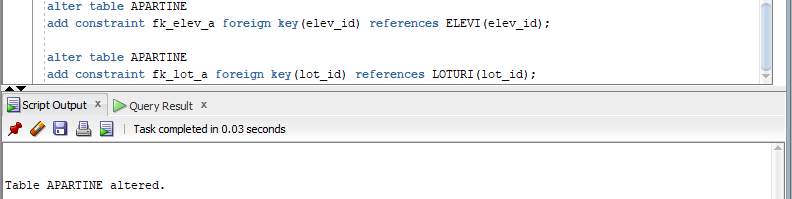
);

alter table APARTINE

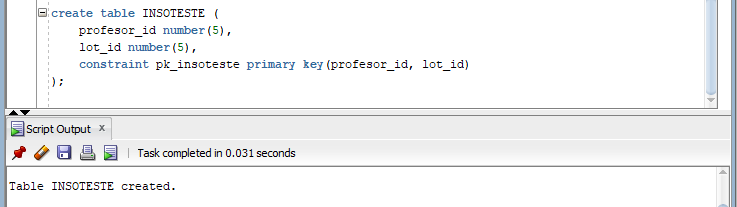
add constraint fk\_elev\_a foreign key(elev\_id) references ELEVI(elev\_id);

alter table APARTINE

add constraint fk\_lot\_a foreign key(lot\_id) references LOTURI(lot\_id);



* + INSOTESTE



create table INSOTESTE (

profesor\_id number(5),

lot\_id number(5),

constraint pk\_insoteste primary key(profesor\_id, lot\_id)

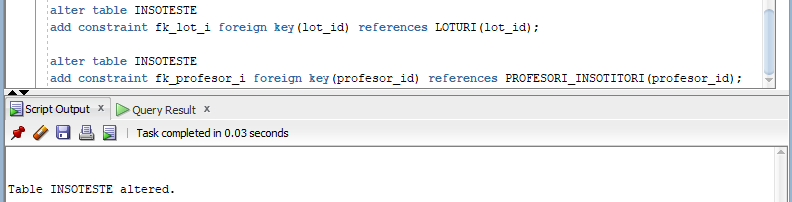
);

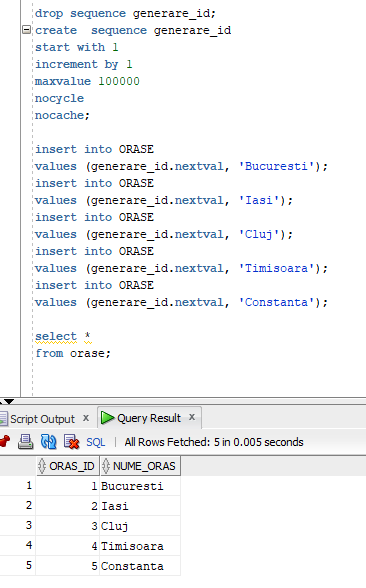
alter table INSOTESTE

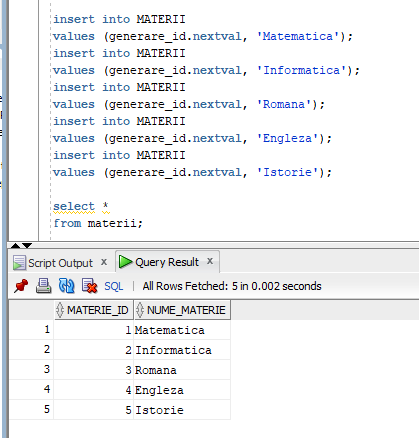
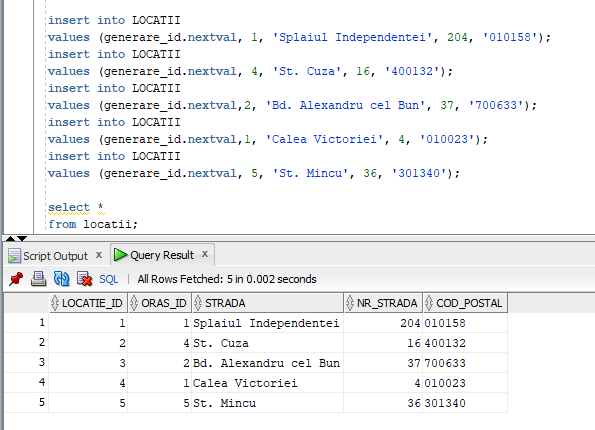
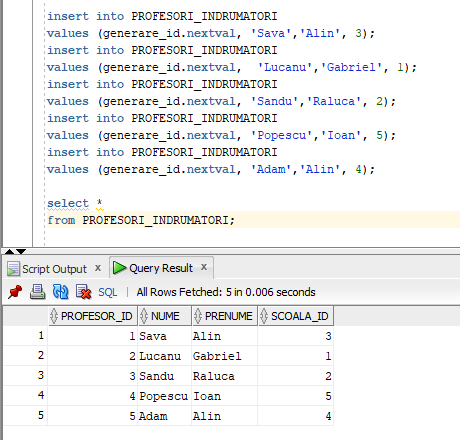
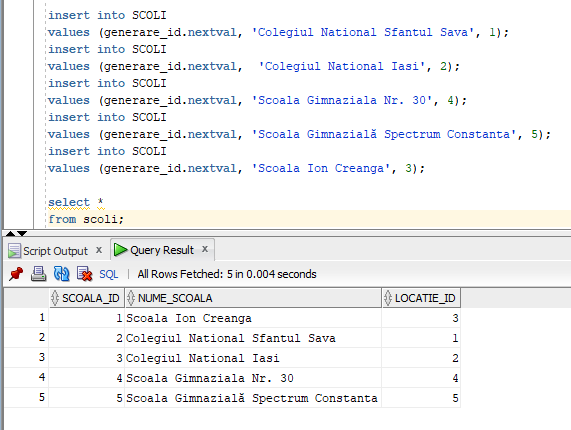
add constraint fk\_lot\_i foreign key(lot\_id) references LOTURI(lot\_id);

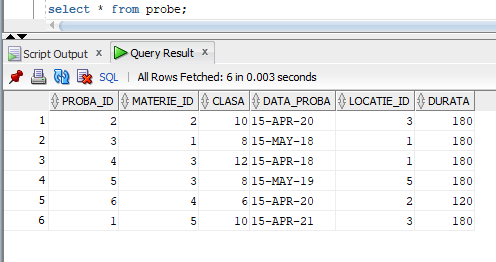
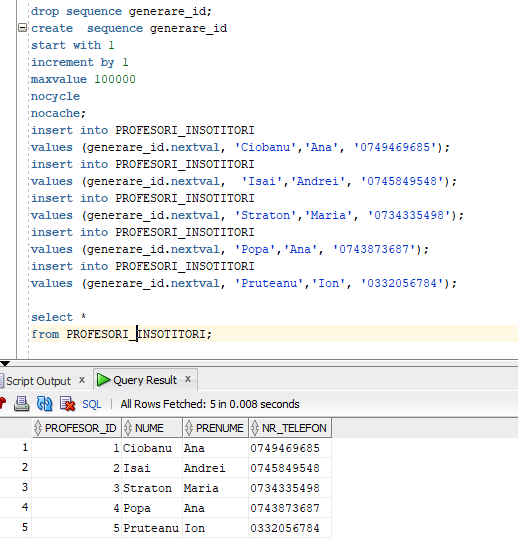
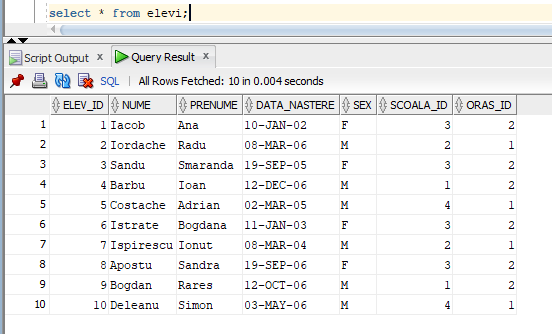
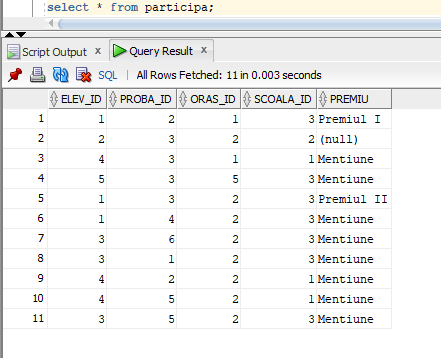
alter table INSOTESTE

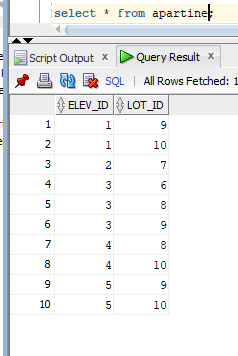
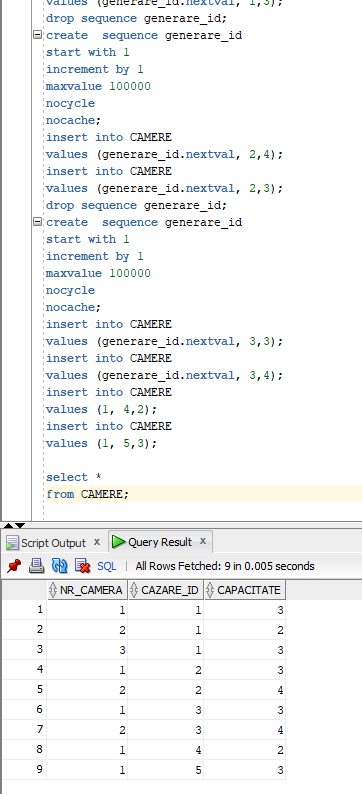
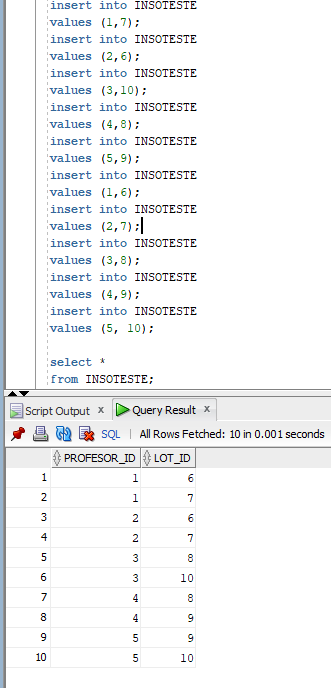
add constraint fk\_profesor\_i foreign key(profesor\_id) references PROFESORI\_INSOTITORI(profesor\_id);

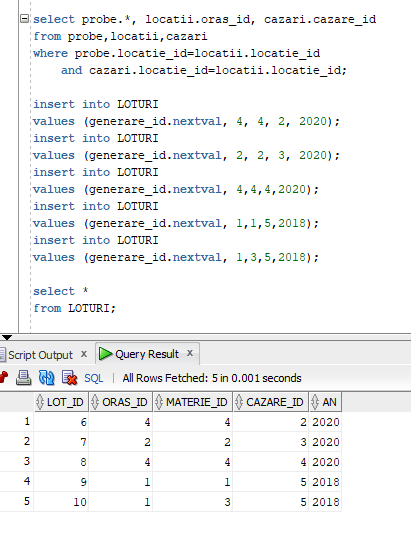
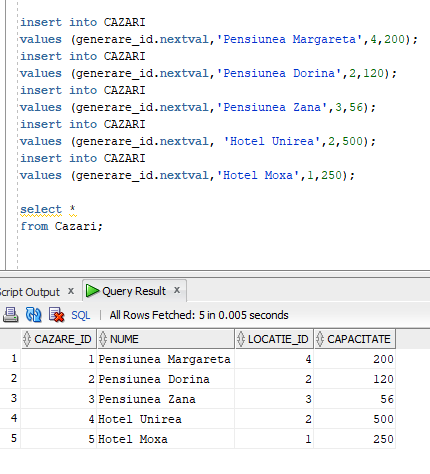


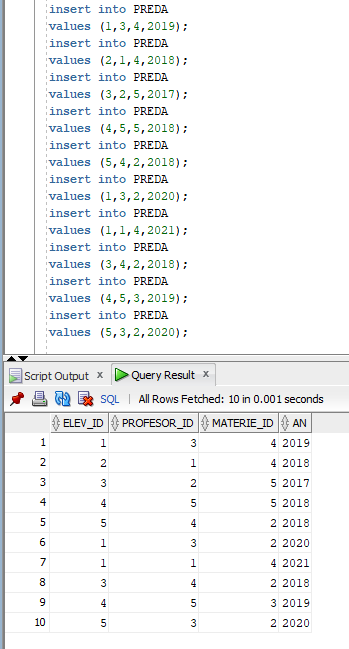
* + INSERĂRI











insert into MATERII

values (generare\_id.nextval, 'Matematica');

insert into MATERII

values (generare\_id.nextval, 'Informatica');

insert into MATERII

values (generare\_id.nextval, 'Romana');

insert into MATERII

values (generare\_id.nextval, 'Engleza');

insert into MATERII

values (generare\_id.nextval, 'Istorie');

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into ORASE

values (generare\_id.nextval, 'Bucuresti');

insert into ORASE

values (generare\_id.nextval, 'Iasi');

insert into ORASE

values (generare\_id.nextval, 'Cluj');

insert into ORASE

values (generare\_id.nextval, 'Timisoara');

insert into ORASE

values (generare\_id.nextval, 'Constanta');

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into LOCATII

values (generare\_id.nextval, 1, 'Splaiul Independentei', 204, '010158');

insert into LOCATII

values (generare\_id.nextval, 4, 'St. Cuza', 16, '400132');

insert into LOCATII

values (generare\_id.nextval,2, 'Bd. Alexandru cel Bun', 37, '700633');

insert into LOCATII

values (generare\_id.nextval,1, 'Calea Victoriei', 4, '010023');

insert into LOCATII

values (generare\_id.nextval, 5, 'St. Mincu', 36, '301340');

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into SCOLI

values (generare\_id.nextval, 'Colegiul National Sfantul Sava', 1);

insert into SCOLI

values (generare\_id.nextval, 'Colegiul National Iasi', 2);

insert into SCOLI

values (generare\_id.nextval, 'Scoala Gimnaziala Nr. 30', 4);

insert into SCOLI

values (generare\_id.nextval, 'Scoala Gimnazial? Spectrum Constanta', 5);

insert into SCOLI

values (generare\_id.nextval, 'Scoala Ion Creanga', 3);

select \* from locatii;

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into PROFESORI\_INDRUMATORI

values (generare\_id.nextval, 'Sava','Alin', 3);

insert into PROFESORI\_INDRUMATORI

values (generare\_id.nextval, 'Lucanu','Gabriel', 1);

insert into PROFESORI\_INDRUMATORI

values (generare\_id.nextval, 'Sandu','Raluca', 2);

insert into PROFESORI\_INDRUMATORI

values (generare\_id.nextval, 'Popescu','Ioan', 5);

insert into PROFESORI\_INDRUMATORI

values (generare\_id.nextval, 'Adam','Alin', 4);

select \* from profesori\_indrumatori;

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into PROBE

values (generare\_id.nextval, 2, 10, to\_date('15/04/2020', 'DD/MM/YYYY'), 3, 180);

insert into PROBE

values (generare\_id.nextval, 1, 9, to\_date('15/05/2018', 'DD/MM/YYYY'), 1, 360);

insert into PROBE

values (generare\_id.nextval, 3, 12, to\_date('15/04/2018', 'DD/MM/YYYY'), 1, 180);

insert into PROBE

values (generare\_id.nextval, 3, 7, to\_date('15/05/2019', 'DD/MM/YYYY'), 5, 120);

insert into PROBE

values (generare\_id.nextval, 4, 6, to\_date('15/04/2020', 'DD/MM/YYYY'), 2, 120);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into ELEVI

values (generare\_id.nextval, 'Istrate', 'Ana', to\_date('10/01/2002', 'DD/MM/YYYY'), 'F', 3, 1);

insert into ELEVI

values (generare\_id.nextval, 'Iordache', 'Radu', to\_date('08/03/2006', 'DD/MM/YYYY'), 'M', 2, 2);

insert into ELEVI

values (generare\_id.nextval, 'Sandu', 'Smaranda', to\_date('19/09/2005', 'DD/MM/YYYY'), 'F', 3, 4);

insert into ELEVI

values (generare\_id.nextval, 'Barbu', 'Ioan', to\_date('12/12/2006', 'DD/MM/YYYY'), 'M', 1, 1);

insert into ELEVI

values (generare\_id.nextval, 'Costache', 'Adrian', to\_date('02/03/2005', 'DD/MM/YYYY'), 'M', 4, 5);

insert into ELEVI

values (6, 'Istrate', 'Bogdana', to\_date('11/01/2003', 'DD/MM/YYYY'), 'F', 3, 2);

insert into ELEVI

values (7, 'Ispirescu', 'Ionut', to\_date('08/03/2004', 'DD/MM/YYYY'), 'M', 2, 1);

insert into ELEVI

values (8, 'Apostu', 'Sandra', to\_date('19/09/2006', 'DD/MM/YYYY'), 'F', 3, 2);

insert into ELEVI

values (9, 'Bogdan', 'Rares', to\_date('12/10/2006', 'DD/MM/YYYY'), 'M', 1, 2);

insert into ELEVI

values (10, 'Deleanu', 'Simon', to\_date('03/05/2006', 'DD/MM/YYYY'), 'M', 4, 1);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into PARTICIPA

values (1,2, 1,3,'Premiul I');

insert into PARTICIPA

values (2,3, 2, 2, Null);

insert into PARTICIPA

values (3,4, 4,3, 'Premiul II');

insert into PARTICIPA

values (4,3, 1, 1, 'Mentiune');

insert into PARTICIPA

values (5,3, 5,3, 'Mentiune');

insert into PREDA

values (1,3,4,2019);

insert into PREDA

values (2,1,4,2018);

insert into PREDA

values (3,2,5,2017);

insert into PREDA

values (4,5,5,2018);

insert into PREDA

values (5,4,2,2018);

insert into PREDA

values (1,3,2,2020);

insert into PREDA

values (1,1,4,2021);

insert into PREDA

values (3,4,2,2018);

insert into PREDA

values (4,5,3,2019);

insert into PREDA

values (5,3,2,2020);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into CAZARI

values (generare\_id.nextval,'Pensiunea Margareta',4,200);

insert into CAZARI

values (generare\_id.nextval,'Pensiunea Dorina',2,120);

insert into CAZARI

values (generare\_id.nextval,'Pensiunea Zana',3,56);

insert into CAZARI

values (generare\_id.nextval, 'Hotel Unirea',2,500);

insert into CAZARI

values (generare\_id.nextval,'Hotel Moxa',1,250);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into CAMERE

values (generare\_id.nextval, 1,3);

insert into CAMERE

values (generare\_id.nextval, 1,2);

insert into CAMERE

values (generare\_id.nextval, 1,3);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into CAMERE

values (generare\_id.nextval, 2,4);

insert into CAMERE

values (generare\_id.nextval, 2,3);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into CAMERE

values (generare\_id.nextval, 3,3);

insert into CAMERE

values (generare\_id.nextval, 3,4);

insert into CAMERE

values (1, 4,2);

insert into CAMERE

values (1, 5,3);

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

insert into PROFESORI\_INSOTITORI

values (generare\_id.nextval, 'Ciobanu','Ana', '0749469685');

insert into PROFESORI\_INSOTITORI

values (generare\_id.nextval, 'Isai','Andrei', '0745849548');

insert into PROFESORI\_INSOTITORI

values (generare\_id.nextval, 'Straton','Maria', '0734335498');

insert into PROFESORI\_INSOTITORI

values (generare\_id.nextval, 'Popa','Ana', '0743873687');

insert into PROFESORI\_INSOTITORI

values (generare\_id.nextval, 'Pruteanu','Ion', '0332056784');

drop sequence generare\_id;

create sequence generare\_id

start with 1

increment by 1

maxvalue 100000

nocycle

nocache;

select probe.\*, locatii.oras\_id, cazari.cazare\_id

from probe,locatii,cazari

where probe.locatie\_id=locatii.locatie\_id

and cazari.locatie\_id=locatii.locatie\_id;

insert into LOTURI

values (generare\_id.nextval, 4, 4, 2, 2020);

insert into LOTURI

values (generare\_id.nextval, 2, 2, 3, 2020);

insert into LOTURI

values (generare\_id.nextval, 4,4,4,2020);

insert into LOTURI

values (generare\_id.nextval, 1,1,5,2018);

insert into LOTURI

values (generare\_id.nextval, 1,3,5,2018);

select elevi.elev\_id, loturi.lot\_id

from loturi, elevi

where elevi.oras\_id=loturi.oras\_id;

insert into APARTINE

values (1,9);

insert into APARTINE

values (1,10);

insert into APARTINE

values (2,7);

insert into APARTINE

values (3,6);

insert into APARTINE

values (3, 8);

insert into APARTINE

values (4,8);

insert into APARTINE

values (4,10);

insert into INSOTESTE

values (1,7);

insert into INSOTESTE

values (2,6);

insert into INSOTESTE

values (3,10);

insert into INSOTESTE

values (4,8);

insert into INSOTESTE

values (5,9);

insert into INSOTESTE

values (1,6);

insert into INSOTESTE

values (2,7);

insert into INSOTESTE

values (3,8);

insert into INSOTESTE

values (4,9);

insert into INSOTESTE

values (5, 10);

CREATE OR REPLACE PROCEDURE Ex6

(nume\_elev elevi.nume%TYPE)

IS

TYPE tablou\_imbricat IS TABLE OF NUMBER;

locatii\_ids tablou\_imbricat := tablou\_imbricat();

TYPE tablou\_indexat IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

frecventa tablou\_indexat;

v\_oras\_id locatii.oras\_id%type;

v\_nume\_oras orase.nume\_oras%type;

BEGIN

select pr.locatie\_id bulk collect into locatii\_ids

from elevi e, participa p, probe pr

where e.nume=nume\_elev and e.elev\_id=p.elev\_id and p.proba\_id=pr.proba\_id;

for i in locatii\_ids.first..locatii\_ids.last loop

select oras\_id

into v\_oras\_id

from locatii l

where l.locatie\_id=locatii\_ids(i);

if frecventa.exists(v\_oras\_id) then

frecventa(v\_oras\_id):=frecventa(v\_oras\_id)+1;

else

frecventa(v\_oras\_id):=1;

end if;

end loop;

for i in frecventa.first..frecventa.last loop

if frecventa.exists(i) then

select nume\_oras into v\_nume\_oras

from orase o

where o.oras\_id=i;

DBMS\_OUTPUT.PUT\_LINE(v\_nume\_oras || ' ' || frecventa(i) );

end if;

end loop;

END Ex6;

/

Begin

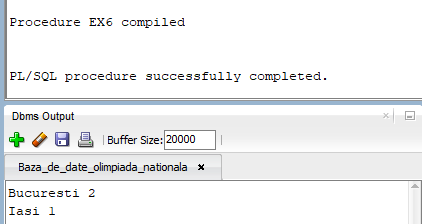
Ex6('Iacob');

End;

/

Ex. 6

Cerinta: Pentru un elev al carui nume este dat, identificati toate probele la care a participat acesta, respectiv locatiile probelor si afisati orasele in care se afla aceste locatii, precum si numarul de locatii vizitate de elev din fiecare oras.



Ex. 7

CREATE OR REPLACE PROCEDURE Ex7

(v\_nume\_materie materii.nume\_materie%TYPE)

IS

CURSOR cursor\_probe (parametru materii.materie\_id%type) IS

select proba\_id

from probe

where materie\_id=parametru;

v\_materie\_id number(4);

BEGIN

select materie\_id into v\_materie\_id

from materii

where initcap(nume\_materie)=v\_nume\_materie;

for i in cursor\_probe(v\_materie\_id) loop

DBMS\_OUTPUT.PUT\_LINE('Proba ID: ' || i.proba\_id );

for j in ( select nume,prenume

from participa p, elevi e

where p.proba\_id=i.proba\_id and p.elev\_id=e.elev\_id)

loop

DBMS\_OUTPUT.PUT\_LINE(j.nume ||' '||j.prenume);

end loop;

DBMS\_OUTPUT.PUT\_LINE('--------------');

end loop;

END Ex7;

/

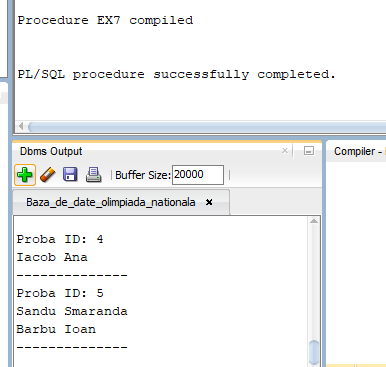
Begin

Ex7('Romana');

End;

/

Cerinta: Pentru fiecare dintre probele la o disciplina data, sa se afiseze elevii participanti la aceasta.



Ex. 8

Ex. 9