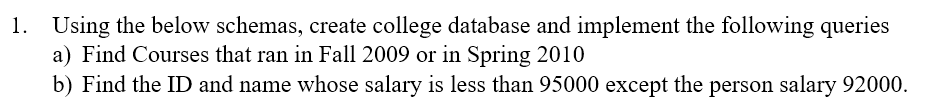


|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | |
|  | | CSLR 51 : DBMS LAB-EndSem | | | | |  | |
|  |  | | | | | | |  |
|  | | | |  |  | | | |
|  | | | | Roll no. : 106119100Name : Rajneesh PandeySection : CSE-B |  | | | |
|  | | |  | | |  | | |



create database collegeDB;

use collegeDB;

create table department(

    dept\_name varchar(20) not null,

    building varchar(20),

    budget int,

    primary key(dept\_name)

);

create table course(

    course\_id varchar(20) not null,

    title varchar(20) not null,

    dept\_name varchar(20) not null,

    credits int not null,

    primary key(course\_id),

    foreign key(dept\_name) references department(dept\_name)

);

 create table time\_slot(

     time\_slot\_id int not null,

     day varchar(20) not null,

     start\_time int not null,

     end\_time int not null,

     primary key(time\_slot\_id)

 );

create table section(

    section\_id int not null,

    course\_id varchar(20) not null,

    semester varchar(20) not null,

    year DATE not null,

    building int not null,

    room\_number int not null,

    primary key(section\_id, course\_id),

    foreign key(course\_id) references course(course\_id)

);

create table instructor(

    instructor\_id int not null,

    name varchar(20) not null,

    dept\_name varchar(20) not null,

    salary int not null,

    primary key(instructor\_id),

    foreign key(dept\_name) references department(dept\_name)

);

insert into department values

      ('Finance', 'Building', '100000'),

      ('Physics', 'Building', '100000'),

      ('CompSci', 'Building', '100000');

insert into course values

    ('CS-101', 'Title', 'CompSci', 10),

    ('CS-315', 'Title', 'CompSci', 10),

    ('CS-347', 'Title', 'CompSci', 10),

    ('FIN-201', 'Title', 'Finance', 10),

    ('MU-199', 'Title', 'Finance', 10),

    ('PHY-101', 'Title', 'Physics', 10);

insert into section values

    (1, 'CS-101', 'Fall', '2009-01-01', 1, 10101),

    (1, 'CS-315', 'Spring', '2010-01-01', 1, 10101),

    (1, 'CS-347', 'Fall', '2009-01-01', 1, 10101),

    (1, 'FIN-201', 'Spring', '2010-01-01', 1, 12121),

    (1, 'MU-199', 'Spring', '2010-01-01', 1, 15151),

    (1, 'PHY-101', 'Fall', '2009-01-01', 1, 22222);

insert into instructor values

    (12121, 'Wu', 'Finance', 90000),

    (222222, 'Einstein', 'Physics', 95000),

    (33456, 'Gold', 'Physics', 87000),

    (83821, 'Brnadt', 'CompSci', 92000);

-- a)   Find Courses that ran in Fall 2009 or in Spring 2010

select \* from course, section where course.course\_id = section.course\_id

and ((year(section.year) = year('2009-01-01') and section.semester = 'Fall') or

(year(section.year) = year('2010-01-01') and section.semester = 'Spring'));

Graphical user interface, table

Description automatically generated with medium confidence

-- b)   Find the ID and name whose salary is less than 95000 except the person salary 92000.

select instructor\_id, name from instructor where salary < 95000 and salary != 92000;

Graphical user interface, text, application

Description automatically generated

Text, letter

Description automatically generated

CREATE DATABASE schoolDB;

use schoolDB;

drop table if exists Highschooler;

drop table if exists Friend;

drop table if exists Likes;

create table Highschooler(

        ID int,

        name text,

        grade int);

create table Friend(

        ID1 int,

        ID2 int);

create table Likes(

        ID1 int,

        ID2 int);

/\* Populate the tables with our data \*/

insert into Highschooler

        values

            (1468, 'Kris', 10),

            (1510, 'Jordan', 9),

            (1689, 'Gabriel', 9),

            (1381, 'Tiffany', 9),

            (1709, 'Cassandra', 9),

            (1101, 'Haley', 10),

            (1782, 'Andrew', 10),

            (1641, 'Brittany', 10),

            (1247, 'Alexis', 11),

            (1316, 'Austin', 11),

            (1911, 'Gabriel', 11),

            (1501, 'Jessica', 11),

            (1304, 'Jordan', 12),

            (1025, 'John', 12),

            (1934, 'Kyle', 12),

            (1661, 'Logan', 12);

insert into Friend

                values

                    (1510, 1381),

                    (1510, 1689),

                    (1689, 1709),

                    (1381, 1247),

                    (1709, 1247),

                    (1689, 1782),

                    (1782, 1468),

                    (1782, 1316),

                    (1782, 1304),

                    (1468, 1101),

                    (1468, 1641),

                    (1101, 1641),

                    (1247, 1911),

                    (1247, 1501),

                    (1911, 1501),

                    (1501, 1934),

                    (1316, 1934),

                    (1934, 1304),

                    (1304, 1661),

                    (1661, 1025);

insert into Friend select ID2, ID1 from Friend;

insert into Likes values

                (1689, 1709),

                (1709, 1689),

                (1782, 1709),

                (1911, 1247),

                (1247, 1468),

                (1641, 1468),

                (1316, 1304),

                (1501, 1934),

                (1934, 1501),

                (1025, 1101);

-- Write two triggers to maintain symmetry in friend relationships.

-- “ If (A,B) is deleted from Friend, then (B,A) should be deleted too”

DELIMITER //

create trigger Del

after delete on Friend

        for each row

                        begin

                          delete from Friend

                          where (ID1 = Old.ID2 and ID2 = Old.ID1);

                        end //

DELIMITER ;

DELIMITER //

create trigger Ins

after insert on Friend

        for each row

                        begin

                          insert into Friend values (New.ID2, New.ID1);

                        end //

DELIMITER ;

select \* from Highschooler;

select \* from Friend;

select \* from Likes;

insert into Friend values (1510, 1782);

select \* from Friend;