SUMMER 2005 CSC 343 INTRODUCTION TO DATABASES

Test 2

Duration: 80 minutes

Aids allowed: a double-sided 8.5” x 11” sheet of paper with definitions, notations, etc.

Write your answers in the spaces provided. Write legibly. No remarking for exams written in pencil.

STUDENT NUMBER____

LAST NAME ________________________ FIRST NAME________________________

Q1_______/30

Q2_______/30

Q3_______/50

Q4_______/25

Q5_______/25

Q6_______/40

TOTAL_______/200

```
-- cID represents the ID of a company
-- city represents the city where the company is located
-- country represents the country where the company is located
-- assets represents the assets the company has
create table company ( \
   cID int not null, \
   cName varchar(15) not null, \
   city varchar(15) not null, \
   country varchar(15) not null, \
   assets double not null, \
   primary key (cID) )

-- eID represents the employer's ID
-- eName represents the employer's name
-- city represents the city where he/she lives
-- dob represents the employee's date of birth
create table employee ( \
   eID int not null, \
   eName varchar(20) not null, \
   city varchar(15) not null, \
   dob DATE not null, \
   primary key (eID) )

-- deptID represents the ID of a department
-- cID represents the ID of the company the dept belongs to
-- deptMngID represents the ID of the manager of the dept
-- emplNo represents the number of employees the dept has
create table department ( \
   deptID int not null, \
   cID int not null, \
   deptName varchar(20) not null, \
   deptMngID int not null, \
   emplNo int not null, \
   primary key (deptID), \
   foreign key (cID) references company on delete cascade, \
   foreign key (deptMngID) references employee (eID) on delete cascade )

-- eID represents the employee's ID
-- deptID represents the ID of the dept the employee works for
-- startDate represents the date when he/she started working for
-- endDate represents the date when he/she stopped working for
-- this department
-- if an employee is currently working, the endDate is NULL
-- salary represents his/hers monthly salary
create table worksFor ( \
   eID int not null, \
   deptID int not null, \
   startDate DATE not null, \
   endDate DATE default null, \
```

```
   salary int not null, \
   primary key (eID, deptID), \
   foreign key (eID) references employee on delete cascade, \
   foreign key (deptID) references department on delete cascade )

-- prID represents the ID of the project
-- deptID represents the ID of the dept developing the project
-- prMngID represents the ID of the project manager
-- startDate represents the date when the project started
-- deadline represents the deadline of the project
-- this department
-- completed indicates the status of the project ( 1 -completed,
-- 0 - under development)
-- budget represents an the  money allocated per month to the project
create table project ( \
   prID int not null, \
   deptID int not null, \
   prMngID int not null, \
   startDate DATE not null, \
   deadline DATE not null, \
   completed int not null default 0, \
   budget double not null, \
   primary key (prID), \
   foreign key (deptID) references department on delete cascade, \
   foreign key (prMngID) references employee (eID) on delete cascade )

create table baseProject ( \
   prID int not null, \
   cost double not null, \
   primary key (prID), \
   foreign key (prID) references project on delete cascade )

-- eID represents the ID of an employee
-- prID represents the ID of the project he/she works on
create table worksOn ( \
   eID int not null, \
   prID int not null, \
   primary key (eID, prID), \
   foreign key (eID) references employee on delete cascade, \
   foreign key (prID) references project on delete cascade )

--subprojectID represents a direct subproject of projectID
create table subProject ( \
   projectID int not null, \
   subprojectID int not null, \
   primary key (projectID, subprojectID), \
   foreign key (projectID) references project(prID) on delete cascade, \
   foreign key (subprojectID) references project(prID) on delete cascade
   check (projectID NOT IN (SELECT prID FROM baseProject))
```

Using the provided database schema, answer the following questions:

Part I SQL

Question 1 (30 points)

Find the id's of employees who are working on at least half of the projects still under development (uncompleted) in the department they are currently working for.

```
SELECT W.eid
FROM workFor W
WHERE (startDate < CURRENT DATE)
   AND ((endDate > CURRENT DATE) OR (endDate IS NULL)
GROUP BY W.eid
HAVING COUNT(*) > (SELECT COUNT(*)
                   FROM     Project P
                   WHERE  P.completed =  0 AND P.deptID = W.deptID)
```

Question 2 (30 points)

For each company, find the departments that TODAY have the highest number of employees. Only employees currently working in that department should be considered. The output should be a table with two columns cID and deptID.

```
SELECT  T1.cID, T1.deptID
FROM    (SELECT       COUNT(*)) AS nr_of_employees
         FROM         workFor W
          GROUP BY deptID, cID) T1,
        (SELECT       MAX(nr_of_employees) AS maximum
          FROM         (SELECT      COUNT(*)) AS nr_of_employees
                        FROM         workFor W
                        GROUP BY deptID, cID) T2
           GROUP BY cID) T3
WHERE T1.cID = T3.cID AND T1.COUNT(*) = T3.maximum
```

Question 3 (50 points)

Write a query that will compute the cost of all projects. The cost of a project is the sum of the costs of its direct subprojects. The output should be a table (prID, cost).

```
WITH RECURSIVE costProject(prID, cost) AS
                    (SELECT  *
                     FROM     baseProject)

                     UNION

                   (SELECT      S.prID, sum(C.cost)
                    FROM         subProject S, costProject C
                    WHERE       S.subproject = C.prID
                    GROUP BY S.prID)
SELECT *
FROM    costProject
```

Part II Assertions/Triggers

Question 4 (25 points)

Write an assertion that checks weather there are uncompleted projects whose monthly budget exceeds the sum of monthly salaries of all the people who listed as working on the project and are currently working for the department the project belongs to. The manager of a project is assumed to be listed as working on the project.

```
CREATE ASSERTION a1
CHECK ( NOT EXISTS (SELECT    P.prID
                    FROM      Project P, worksFor W1, worksON W2
                    WHERE     P.prID = W2.prPID AND
                              P.depID = W1.deptID AND
                              W1.eID = W2.eID AND
                              W.startDate < CURRENT DATE AND
                              W.endDate > CURRENT DATE AND
                              P.completed = 0
                   GROUP BY P.prID, P.budget
                   HAVING    P.budget > SUM(W.salary)
```

Question 5 (25 points)

Write a trigger that after an update of prMng on table **project** will insert a tuple on table **worksOn** such that the new manager will be listed as working on the project.

```
CREATE TRIGGER t1
AFTER UPDATE OF prMng ON project
REFERENCING NEW ROW AS new
FOR EACH ROW
INSERT ON TABLE worksON VALUES (new.prMNG, new. PrID)
```

Part III Datalog (40 points)

Query 6

Write a Datalog query that will list for all projects their direct and indirect subprojects. The output should be a relation (prID1, prID2) where prID2 is a direct or indirect subproject of prID1.

Relationship(prID1, prID2) :- subproject(prID1, prID2)
Relationship(prID1, prID20 :- Relationship(prID1, prID) AND subproject(prID, prID2)