**HW5 SQL and Relational Algebra due on March 4 at 11:50 PM**

1. Give the SQL definitions of the tables:

CrossCountrySkier(Name, Country, Age)

Competes(SkierName,ContestName, Placement)

Contest(Name, Place, Country, Length)

Showing particularly the foreign key constraints of the Competes table.

**Solution:**

Create table CrossCountrySkier

(Name character (25) Primary key,

Country character (25),

Age smallint)

Create table Contest

(Name character (25) Primary key,

Place character (30),

Country character (25),

Length numeric(6))

Create table Competes

(SkierName character (25) references CrossCountrySkier (Name),

ContestName character (25),

Placement smallint,

Primary key (SkierName,ContestName),

foreign key(ContestName) references Competes(Name) )

1. Given the following schema:

Airport (City, Country, NumberOfRunways)

Flight (FlightID, Day, DepartCity, DepartTime, ArrCity, ArrTime, PlaneType)

Plane (PlaneType, NumberOfPassengers)

Write the SQL queries with A. using except operator, or with “Not IN” operator, then B. covert it to Relational Algebra way. The question is, we want to find out:

**The Belgian airport that have only domestic flights.**

1. Show this query in SQL using Except or “Not IN” (pick only one):

Solution:

select DepartCity

from Flights join Airport on DepartCity=City

where Country= ‘Belgium’

**except**

select DepartCity from Airport as A1 join Flight on A1.City=DepartCity

join Airport as A2 on ArrCity=A2.City

where (A1.Country=’Belgium’ and A2.Country<>’Belgium’ )

Or, with the not in operator:

select DepartCity

from Flights join Airport on DepartCity=City

where Country= ‘Belgium’ and DepartCity **not in**

(select DepartCity from Airport as A1 join Flight on A1.City=DepartCity

join Airport as A2 on ArrCity=A2.City where A1.Country=’Belgium’ and

A2.Country <> ‘Belgium’ )

1. Show this query in Relational Algebra:

**The Belgian airport that have only domestic flights.**

Solution:

**π** DepartCity **σ** Coutry=’Belgium’(Airport **⨝** City=DepartCity Flight)

1. Give a sequence of update commands that alter the attribute **Salary** in the **Employee** table,

increasing by 10% the salaries below 30 thousand and decreasing by 5% those above 30 thousand.

**Solution:**

update Employee set Salary=Salary/2

where Salary <= 30000

update Employee set Salary=Salary\*0.95

where Salary > 30000

update Employee set Salary=Salary\*2.2

where Salary<= 15000

1. Give the SQL definitions of the tables:

Author (FirstName, Surname, DateofBirth, Nationality)

Book (BookTitle, AuthorFirstName, authorSurname, Language)

For the *foreign key* constraint specify a cascade policy on deletion.

**Solution:**

Create table Author

(FirstName character (25),

Surname character (25),

DateofBirth date,

Nationality character (20),

primary key (FirstName, Surname))

Create table Book

(BookTitle character (30) primary key,

AuthorFirstName character (25),

AuthorSurname character (25),

Language character (20),

foreign key(AuthorfirstName,AuthorSurname) references Author(FirstName, Surname) on delete cascade)