Database Exercise I

TDA357/DIT60
SQL Exercises

2019-01-30

Exercise 1: Basics. Based on the *Departments* table below, create a **SQL** table called *Employees* that stores employee number, employee name, department and salary information. The `primary key` for the *Employees* table should be the employee number. Create a `foreign key` on the *Employees* table that references the *Departments* table based on the `department_id` field.

```
CREATE TABLE Departments (
  department_id INT NOT NULL,
  department_name TEXT NOT NULL,
  -- Here we use the explicit constraint form for primary keys
  CONSTRAINT department_pk PRIMARY KEY (department_id)
);
```

Exercise 2: Suppliers. Consider the following relational schema:

`Suppliers(`**sid:integer**`, sname:string, city:string, street:string)`
`Parts(`**pid:integer**`, pname:string,color:string)`
`Catalog(`**sid:integer**`,`**pid:integer**`,cost:real)`

Find the names of all suppliers who have supplied a non-blue part.

Exercise 3: Employees. Consider the following table:

`Employees(`**empId**`, name, department, salary)`

The columns `empId` and `name` are of type `text`, while `department` and `salary` are of type integer.

a) Write a query to find the employees who get a higher salary than anyone in department 5 (you may assume that department 5 has more than one employee).

b) Write a query to find the maximum salary from each department.

c) Find all employee records containing the word ,,Joe", regardless of whether it was stored as JOE, Joe or joe.

Exercise 4: Company. Consider following relational schema:

```
Employees(employeeId, employeeName, street, city)
Companies(companyId, companyName, city)
Works(employee, company, salary)
Manages(manager, employee)
```

The information on which company an employee works for and their current salary is stored in a relation *Works*. Assume that all people work for at most one company. The information on which employees have manager roles and who they manage is stored in the relation *Manages*.

Write a SQL query for the following tasks:

a) Find the names, street address, and cities for all employees who work for 'First Bank Corporation' and earn more than $10,000.

b) Find the names of all employees who:

   i) live in the same city as the company they work for.

   ii) live on the same street in the same city as their manager.

   iii) earn more than every employee of 'Small Bank Corporation'

c) Find the name of the company that has the smallest payroll (i.e. the smallest total sum of salaries).

d) Assuming that the `Employee` relation had an email column with `NULL` values, write a query to update the email values.

e) Find all employees who are not managers.

Exercise 5: Hospital. A database system used by a hospital to record information about patients and wards has the following relational schema:

```
Wards(number, numBeds)
Patients(pid, name, year, sex)
PatientInWard(pid, ward)
Tests(patient, testDate, testHour, temperature, heartRate)
```

A ward is identified by its number. The `numBeds` attribute is the number of beds in that ward. Patients are identified by their personal identification number. The name, year of birth and biological sex ('M' or 'F') of each patient is stored in the *Patients* relation. The ward to which each patient is assigned is stored in the *PatientInWard* relation.

During their stay in the hospital, patients will undergo routine tests. The date, hour, and measurements (temperature and heart rate) of each test is performed is recorded, and stored in the database. A patient will normally undergo multiple tests over the course of their hospitalization.

Implement the schema in SQL with the appropriate constraints and types, and do the following:

a) Write a SQL query that finds the temperature and heart rate measured in each test carried out on patients born before 1950.

b) Create a view, `FreeBeds(ward, numBeds)` where `ward` is a ward number, and `numBeds` is the number of available beds in that ward.

Exercise 6: Planets. Consider the relation:

```
Planets(star, name, distance, mass, atmosphere, oxygen, water)
```

We assume that all stars have different names, and that planet names are only unique within their star system. For simplicity, we also assume that each star system has exactly one star, and that all planets have circular orbits around their star at different distances. A planet's position indicates its order within the star system, e.g. Earth is the 3rd planet around the Sun, after Mercury and Venus. If a planet has oxygen or other gases, it has an atmosphere. If it does not have an atmosphere, it has no gases. The surface of a planet is either all water, all land, or somewhere in-between.

Do the following:

a) Implement the relation in a SQL table, with reasonable types and constraints. Store distances in millions of km, e.g. for Earth the value should be 149.6.

b) Assume that the relation contains the planet ,,Duna" in orbit around a star called ,,Kerbol". Write a SQL query to find how many planets are in orbits around their star that are larger than Duna's orbit around Kerbol.

c) We say that a planet is ,,habitable" if it satisfies the following conditions:

   i. it orbits at a distance between 100 and 200 million kilometers (inclusive) of its star (i.e. in the ,,Goldilocks zone"),

   ii. it has has an atmosphere, and an oxygen percentage between 15 and 25% (inclusive),

   iii. it has water on its surface.

   Create the view `Habitable(name, star, status)`, where `status` is 'habitable' if the planet `name` around `star` is habitable and 'uninhabitable' if the planet is uninhabitable.