Attachement for Relational Algebra Queries

Questions 7 to 10 are based on assignment 1âs **database schema** and the **syntax for valid relational algebra expressions** which are described in this attachment document.

1. Database Schema

The questions are based on the following application about a company's operation. Its ER data model is shown below with the following constraints.

ER Model

The company has at least one office. Each office (identified by oid with location specified by address) consists of one or more departments. Each department (identified by did with a budget dbudget) is located in one office and has one or more employees. Each employee (identified by eid) must belong to a department. The application focuses on two subclasses of employees: engineers and managers. An employee can be neither an engineer nor a manager, and no employee can be both an engineer and a manager. Each employee can specialize in 0 or more areas (identified by aid). Each department must be managed by exactly one manager, and each manager can manage 0 or more departments. Note that the manager of a department does not necessarily belong to that department. Each engineer can work in 0 or more projects, and there must be at least one engineer working in each project. For each project P that an engineer E works on, the number of hours per week that E spends on P is given by hours. Each project (identified by pid with a budget pbudget) must be supervised by exactly one manager. A manager can supervise 0 or more projects. The attributes hours, dbudget and pbudget have non-null values. The unit of dbudget is in millions of dollars (e.g., a value of 10 denote ten million dollars).

1.1. Relational Schema

The following is the relational schema for this application.

```
CREATE TABLE Offices (
        oid                     INTEGER,
        address                 TEXT,
        PRIMARY KEY (oid)
);



/* eid = eid of department's manager */
CREATE TABLE Departments (
        did                     INTEGER,
        dbudget                 INTEGER NOT NULL,
        oid                     INTEGER NOT NULL,
        eid                     INTEGER NOT NULL,
        PRIMARY KEY (did),
        FOREIGN KEY (oid) REFERENCES Offices
);

CREATE TABLE Employees (
        eid                     INTEGER,
```

```
        did                     INTEGER NOT NULL,
        PRIMARY KEY (eid),
        FOREIGN KEY (did) REFERENCES Departments
);

CREATE TABLE Engineers (
        eid                     INTEGER,
        PRIMARY KEY (eid),
        FOREIGN KEY (eid) REFERENCES Employees
);

CREATE TABLE Managers (
        eid                     INTEGER,
        PRIMARY KEY (eid),
        FOREIGN KEY (eid) REFERENCES Employees
);


/* eid = eid of project's supervisor  */
CREATE TABLE Projects (
        pid                     INTEGER,
        pbudget                 INTEGER NOT NULL,
        eid                     INTEGER NOT NULL,
        PRIMARY KEY (pid),
        FOREIGN KEY (eid) REFERENCES Managers
);


CREATE TABLE Works (
        pid                     INTEGER,
        eid                     INTEGER,
        hours                   INTEGER NOT NULL,
        PRIMARY KEY (pid,eid),
        FOREIGN KEY (eid) REFERENCES Engineers,
        FOREIGN KEY (pid) REFERENCES Projects
);

CREATE TABLE Areas (
        aid                     TEXT,
        PRIMARY KEY (aid)
);

CREATE TABLE Specializes (
        eid                     INTEGER,
        aid                     TEXT,
        PRIMARY KEY (eid,aid),
        FOREIGN KEY (eid) REFERENCES Employees,
        FOREIGN KEY (aid) REFERENCES Areas
);
```

2. Relational Algebra Expressions

2.1. Syntax

The following table explains the syntax of each relational operator, where R and S denote some relational algebra expressions.

SYNTAX	MEANING
select{c}(R)	selection operator on R with predicate c
project{l}(R)	projection operator on R with attribute list l
rename{a1:b1,,..,an:bn}(R)	rename attributes {a1,…,an} of R to {b1,…,bn}
R * S	cross product of R and S
R ~ S	natural join of R and S
R ~> S	natural left join of R and S
R <~ S	natural right join of R and S
R <~> S	natural full join of R and S
R ~{c} S	inner join of R and S with join predicate c
R ~>{c} S	left join of R and S with join predicate c
R <~{c} S	right join of R and S with join predicate c
R <~>{c} S	full join of R and S with join predicate c
R & S	set intersection of R and S
R | S	set union of R and S
R - S	set difference of R and S
R / S	division of R by S
T = R	define T to denote a relational expression R

- Keywords/relation/attribute names are case-insensitive
- String values must be double quoted in selection predicates (e.g., cname = "Moe") and must not contain the character } or ’
- Consecutive relational algebra expressions must be separated by a semicolon.
- A selection condition is a boolean combination of terms, where a term is one of the following forms:

- attribute **op** constant
  - Note that **op** is one of =, <>, <, >, <=, >=
- attribute1 **op** attribute2
- term1 **and** term2
- term1 **or** term2
- **not** term1
- **(** term1 **)**

2.2. Examples

```
project{rname}(
        select{area = "North" or area = "South"}(Restaurants)
)

project{pizza}(
        select{ingredient = "cheese"}(Contains)
)
&
project{pizza}(
        select{ingredient = "chilli"}(Contains)
)

project{cname,pizza}(
        Customers ~> Likes
)

R1 = project{pizza}(select{cname = "Maggie"}(Likes));

R2 = project{rname}(Sells) * R1;

R3 = project{rname}(
        R2 - project{rname,pizza}(Sells)
);

R4 = project{rname}(Sells) - R3;

R5 = rename{pizza:pizza5}(select{cname = "Ralph"}(Likes));

R6 = project{rname}(select{pizza5 = pizza}(Sells * R5));

R4 - R6
```

Last updated 2021-02-28 12:22:01 +0800