CHAPTER 8

Complex Data Types

Practice Exercises

8.1 Provide information about the student named Shankar in our sample university database, including information from the *student* tuple corresponding to Shankar, the *takes* tuples corresponding to Shankar and the *course* tuples corresponding to these *takes* tuples, in each of the following representations:

a. Using JSON, with an appropriate nested representation.

b. Using XML, with the same nested representation.

c. Using RDF triples.

d. As an RDF graph.

Answer:

a. FILL IN

b. FILL IN

c. FILL IN

d. FILL IN

8.2 Consider the RDF representation of information from the university schema as shown in Figure 8.3. Write the following queries in SPARQL.

a. Find the titles of all courses taken by any student named Zhang.

b. Find titles of all courses such that a student named Zhang takes a section of the course that is taught by an instructor named Srinivasan.

c. Find the attribute names and values of all attributes of the instructor named Srinivasan, without enumerating the attribute names in your query.

Answer:
FILL IN

8.3 A car-rental company maintains a database for all vehicles in its current fleet. For all vehicles, it includes the vehicle identification number, license number, manufacturer, model, date of purchase, and color. Special data are included for certain types of vehicles:

- Trucks: cargo capacity.
- Sports cars: horsepower, renter age requirement.
- Vans: number of passengers.
- Off-road vehicles: ground clearance, drivetrain (four- or two-wheel drive).

Construct an SQL schema definition for this database. Use inheritance where appropriate.

Answer:
For this problem, we use table inheritance. We assume that **MyDate**, **Color** and **DriveTrainType** are pre-defined types.

create type *Vehicle*
(*vehicle_id* **integer**,
license_number **char(15)**,
manufacturer **char(30)**,
model **char(30)**,
purchase_date **MyDate**,
color **Color**)

create table *vehicle* **of type** *Vehicle*

create table *truck*
(*cargo_capacity* **integer**)
under *vehicle*

create table *sportsCar*
(*horsepower* **integer**
renter_age_requirement **integer**)
under *vehicle*

create table *van*
(*num_passengers* **integer**)
under *vehicle*

```
create table offRoadVehicle
    (ground_clearance real
     driveTrain DriveTrainType)
    under vehicle
```

8.4 Consider a database schema with a relation *Emp* whose attributes are as shown below, with types specified for multivalued attributes.

Emp = (ename, ChildrenSet **multiset***(Children), SkillSet* **multiset***(Skills))*
Children = (name, birthday)
Skills = (type, ExamSet **setof***(Exams))*
Exams = (year, city)

Define the above schema in SQL, using the SQL Server table type syntax from Section 8.2.1.1 to declare multiset attributes.

Answer:

a. No answer.

b. Queries in SQL.

i. Program:

```
select ename
from emp as e, e.ChildrenSet as c
where 'March' in
        (select birthday.month
         from c
        )
```

ii. Program:

```
select e.ename
from emp as e, e.SkillSet as s, s.ExamSet as x
where s.type = 'typing' and x.city = 'Dayton'
```

iii. Program:

```
select distinct s.type
from emp as e, e.SkillSet as s
```

8.5 Consider the E-R diagram in Figure 8.7 showing entity set *instructor*. Give an SQL schema definition corresponding to the E-R diagram, treating *phone_number* as an array of 10 elements, using Oracle or PostgreSQL syntax.

Answer:

The corresponding SQL:1999 schema definition is given below. Note that the derived attribute *age* has been translated into a method.

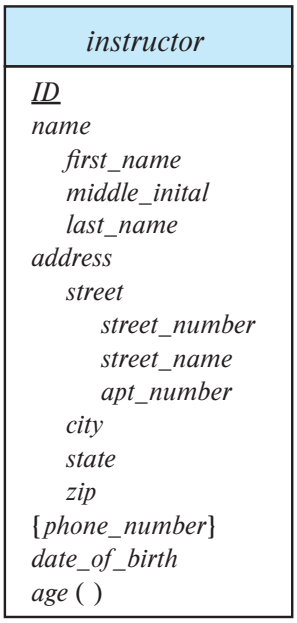

Figure 8.7 E-R diagram with composite, multivalued, and derived attributes.

```
create type Name
    (first_name varchar(15),
     middle_initial char,
     last_name varchar(15))
create type Street
    (street_name varchar(15),
     street_number varchar(4),
     apartment_number varchar(7))
create type Address
    (street Street,
     city varchar(15),
     state varchar(15),
     zip_code char(6))
create table customer
    (name Name,
     customer_id varchar(10),
     address Adress,
     phones  varray(10) of char(7) ,
     dob date)
method integer age( )
```

employee (*person_name*, *street*, *city*)
works (*person_name*, *company_name*, *salary*)
company (*company_name*, *city*)
manages (*person_name*, *manager_name*)

Figure 8.8 Relational database for Exercise 8.6.

The above array syntax is based on Oracle, in PostgreSQL *phones* would be declared to have type **char**(7)[].

8.6 Consider the relational schema shown in Figure 8.8.

a. Give a schema definition in SQL corresponding to the relational schema but using references to express foreign-key relationships.

b. Write each of the following queries on the schema, using SQL.

i. Find the company with the most employees.

ii. Find the company with the smallest payroll.

iii. Find those companies whose employees earn a higher salary, on average, than the average salary at First Bank Corporation.

Answer:

a. The schema definition is given below.

```
create type Employee
    (person_name varchar(30),
     street varchar(15),
     city varchar(15))
create type Company
    (company_name varchar(15),
    (city varchar(15))
create table employee of Employee
create table company of Company
create type Works
    (person ref(Employee) scope employee,
    comp ref(Company) scope company,
    salary int)
create table works of Works
create type Manages
    (person ref(Employee) scope employee,
    (manager ref(Employee) scope employee)
create table manages of Manages
```

b. i.
```
select comp->name
from works
group by comp
having count(person) ≥ all(select count(person)
                           from works
                           group by comp)
```

ii.
```
select comp->name
from works
group by comp
having sum(salary) ≤ all(select sum(salary)
                         from works
                         group by comp)
```

iii.
```
select comp->name
from works
group by comp
having avg(salary) > (select avg(salary)
                      from works
                      where comp->company_name="First Bank Corporation")
```

8.7 Compute the relevance (using appropriate definitions of term frequency and inverse document frequency) of each of the Practice Exercises in this chapter to the query "SQL relation".

Answer:

We do not consider the questions containing neither of the keywords because their relevance to the keywords is zero. The number of words in a question include stop words. We use the equations given in Section 31.2 to compute relevance; the log term in the equation is assumed to be to the base 2.

Q#	#wo--rds	# "SQL"	#"rela--tion"	"SQL" term freq.	"relation" term freq.	"SQL" relv.	"relation" relv.	Tota relv.
1	84	1	1	0.0170	0.0170	0.0002	0.0002	0.0004
4	22	0	1	0.0000	0.0641	0.0000	0.0029	0.0029
5	46	1	1	0.0310	0.0310	0.0006	0.0006	0.0013
6	22	1	0	0.0641	0.0000	0.0029	0.0000	0.0029
7	33	1	1	0.0430	0.0430	0.0013	0.0013	0.0026
8	32	1	3	0.0443	0.1292	0.0013	0.0040	0.0054
9	77	0	1	0.0000	0.0186	0.0000	0.0002	0.0002
14	30	1	0	0.0473	0.0000	0.0015	0.0000	0.0015
15	26	1	1	0.0544	0.0544	0.0020	0.0020	0.0041

8.8 Show how to represent the matrices used for computing PageRank as relations. Then write an SQL query that implements one iterative step of the iterative technique for finding PageRank; the entire algorithm can then be implemented as a loop containing the query.

Answer:
FILL

8.9 Suppose the *student* relation has an attribute named *location* of type point, and the *classroom* relation has an attribute *location* of type polygon. Write the following queries in SQL using the PostGIS spatial functions and predicates that we saw earlier:

a. Find the names of all students whose location is within the classroom Packard 101.

b. Find all classrooms that are within 100 meters or Packard 101; assume all distances are represented in units of meters.

c. Find the ID and name of student who is geographically nearest to the student with ID 12345.

d. Find the ID and names of all pairs of students whose locations are less than 200 meters apart.

Answer:
FILL