CSE 532: Project 1 - Music Fan Club

Fall 2009

Deadline: October 12

In this project, you will be designing and implementing part of a knowledge base that keeps information about music fans, singers, and their albums. You will be using the Datalog language supported by the XSB system (`http://xsb.sourceforge.net`). The lecture slides provide a brief introduction to the use of XSB's Datalog. You are not required to build any GUI or store any data in a DBMS. Instead, you should keep the data in a file along with the rules for defining the intermediate predicates used in your queries.

1 General Description

The Music Fan Club knowledge base (MFCKB) is a social networking system that keeps information about music fans, singers, their albums, and the various friend-of-a-friend type of information. The system enables the users to ask analytical queries that can help them form various interest groups and discover new information.

2 System Users

The users of the system are the music fans who subscribe to the MFC services.

3 Required Data

The data items required by your system roughly fall into these categories:

- *Information about users.*
- *Information about singers.*
- *Information about albums and songs.*

Every user in the system is a person who has

- *Account.*
- *Name*
- *Address*
- *Friends* (i.e., other users in this user's network, as, for example, in Facebook).
- *Favorite singers.*
- *Favorite songs.*

A singer is a person with the usual attributes plus:

- *Albums* that the singer has created.

Note: an album may have several singers and it is possible for a singer to be a user of MFCKB. For simplicity, we will also assume that every singer associated with an album sings all the songs in that album.

For the various unstated assumptions, use common sense. For instance, a singer can also be a user, but does not have to. The same person may have several different accounts (and the information about songs and singers might differ in these accounts). Likewise, two different users can have the same name and/or address.
An album has the following information:

- *Title.*
- *The record company that produced the album.*
- *Singer(s).*
- *Songs in the album.*

A song has: a *title*, an *author* (or authors) of the lyrics, and a *composer* (or several composers). We assume that the song's title is its key.

The predicates/relations that will contain the above information should be designed according to the best practices of the database theory. The structure of the above information is such that naive ways to organize the data in relations will cause them not to be in 4NF. So, you should be careful that this does not happen. (There are not that many interesting functional dependencies here, so we are not talking about 3NF and BCNF here. You really should watch out for multivalued dependencies and thus focus on 4NF.)

4 Queries

You are to implement the following queries. You should not use function symbols, lists, *findall*, and the like: only pure Datalog with negation (tnot).

1. Given a user, find the friends who share this user's fondness for at least one singer and at least one song of that singer.
2. Find all users who indicated that they like some song, but did not specify that they like its album's singer (note: the same song can be in several albums and sung by several singers; in that case, it is enough to not specify that the user likes one of the singers in one of the albums).
3. Find all users who indicated that they like some song, but did not indicate that they like *any* of its singers.
4. Given a user, find all friends of that user who like all the singers that are liked by that user.

   Note: this query requires nontrivial use of double negation.
5. Given a user, find all friends-of-friends of that user who have similar interests. Two users have similar interests if there is a chain of friend relationships that connects the two users and all the users in the chain like the same song.

   Note: this query requires recursive rules.

5 Documentation and Submission Instructions

Your project should include a short document, which should appear at the top of the program file between the comment markers /* ... */. The document should detail the database schema used for the project (i.e., relation names, the meaning of the columns, and their types). Also, state what you expect to be the keys in each relation. You do not need to express them in Datalog (we did not discuss the representation of the schema in this language). Use some informal description (not SQL—this would be too verbose).

In order to express some queries you would need to include rules to define some intermediate relations. In some cases, the intent of those rules may not be obvious and you would have to explain their meaning with a comment line or two.

Pay attention to the aesthetics. Poorly formatted or designed works will be penalized. A typical layout of Datalog rules is:

- Single line rules, if they are shorter than 80 characters.
- Multiline rules. These have the layout

```
rulehead :-
    predicate_1,
    predicate_2,
    ...,
    predicate_n.
```

You will submit the project via the Blackboard system: go to the *Assignments* area and follow the instructions, which can be found at
`https://tlt.stonybrook.edu/StudentServices/BbStudents/Pages/SubmittingWork.aspx#assignments`
The data, the rules, and the queries should be in one **.txt** (not .doc) file, which should be uploaded as described at the above URL. The file must be directly loadable and executable in XSB without any editing, i.e., the TAs should be able to run your program and get all the answers by simply typing

```
..../xsb  -e "[yourProgram]."
```

6 Teaming

This project must be done **individually** — no partners.
At the top of the program file (in a comment block) include your name, student Id, email, and this statement:

I pledge my honor that all parts of this project were done by me individually and without collaboration with others.

7 Planning Your Work

This is not a hard project, but, since Datalog is likely to be new to you, you will encounter numerous problems trying to get things done. Therefore, start right away and do not delay. There will be no deadline extensions.

We will provide a test data set. However, you might need to add more information to ensure that your queries have enough data to work with.