CSE3421 fall 2008

Assignment #3: Due NEVER

0%

(This assignment is for practice only and not to be handed in or marked)

In this assignment you are requested to convert a given set of FDs into 3NF, with preserved dependencies and the lossless join property.

Input: A set of functional dependencies (FDs)
Output: set of functional dependencies, in 3NF, with preserved dependencies and the lossless join property.

How
First, using 3NF Synthesis, turn the FDs into ones that satisfy

1. 3NF,
2. preservation of dependencies
3. have lossless join property

Whenever you calculate the minimal cover, scan the FDs top-to-bottom (for eliminating redundant FDs); and within each FD, scan the attributes left-to-right (for eliminating extraneous attributes from the left hand side). Also, when you convert the FDs into standard form (step 1 of minimal cover computation), scan the RHS left-to-right, and list the FDs in the order scanned. For example, if the FD set is

```
...
ABC  ->  DEG
...
```

Then converting the FD `ABC  ->  DEG` into standard form should produce

```
...
ABC  ->  D
ABC  ->  E
ABC  ->  G
...
```

Instead of, for example,

```
...
ABC  ->  E
ABC  ->  G
ABC  ->  D
...
```

Details
Your program should feature a GUI that allows for input of a set of FDs. You should

provide 2 options:
Option I: the user specifies the FDs (see details below)
Option II: the program generates a set of FDs to serve as input (see details below)

Option I details (user specifies FDs)
Your application should read the FDs from a file named `F.txt`. In this file, each FD occupies several lines, as follows:

- each attribute is in a single line, starting from the beginning of the line
- the left hand side (LHS) attributes of an FD are listed first
- the "determines" arrow is denoted by '::' (two colons)
- the right hand side (RHS) attributes of an FD follow the '::' symbol; each RHS attribute occupies one line (as in the LHS).
- The end of an FD is denoted by the word 'END', in a single line by itself.

An example F.txt file is shown in appendix A.

Each attribute is a string. Note, an attribute can be more than a single letter (even though in the sample F.txt file all attributes are single letters).

Options II details (program generated FDs)
For this option, your application should provide for accepting input of certain parameters and based on those parameters it should generate a set of FDs. (then those FDs should be used for the 3NF synthesis). The FDs should be generated as follows:

- First, generate a specified number of attributes. This number should be provided as input by the user, via a textbox in your GUI. Given this number, your application should generate a specified number of attributes.
- After creating the attributes, create FDs using those attributes, as follows:
    - Read input the number of FDs to be generated. This number should be provided as input by the user, via a textbox in your GUI, similar to the number of attributes.
    - Upon knowing how many FDs are required, generate each FD as follows:
        - Number of attributes per FD: minimum 2 attributes (one in LHS and one in RHS), maximum 50% of the total number of attributes. In order to decide the number of attributes for each FD, separately generate a random number between 2 and N, where N is 50% of the total number of attributes.
        - Once the number of attributes for an FD has been decided (by your program), then your program will have to decide how the attributes would be distributed in the LHS and RHS of the FD. For this task, generate a random number that will correspond to a percentage and make the lengths of the LHS and RHS proportional to that percentage. For example, if the random number generated is 40%, then the LHS would have 40% of the attributes of that FD and the RHS would have the remaining 60%.
    - When creating your FDs, you will probably need to use some of the

attributes more than once. In selecting which of the generated attributes you will choose to use in an FD, use the following process: start using attributes from the attribute list, top-to-bottom. If you reach the bottom and you need to use more attributes, continue selcting attributes from the top again, and so on.

Your GUI application

Your GUI application should

- Provide textboxes for entering the inputs as described above.
- Provide a button labeled "**run F.txt**". Upon clicking this button, your program should run without any further interference, read the FDs from file F.txt and generate the output into a file named `Fout.txt`; also, display the resulting FDs in a textArea on your GUI.
- Provide a button labeled "**run INPUTS**". Upon clicking this button, your program should read the user's inputs and then run and generate two files, named `Finput.txt` and `FinputOut.txt`. File `Finput.txt` should contain the FDs generated by your program from the inputs (and prior to converting them through the 3NF Synthesis process). File `FinputOut.txt` should contain the resulting FDs, i.e. the ones that are the result of the 3NF Synthesis process. The FDs of both of those files should also be displayed in 2 separate textAreas on your GUI (with appropriate scrollbars).
- Your application should be named `A31.java`. If it consists of several files, A31.java should be the one containing main().
- All .txt files should reside in the same directory as your java application.

What to submit

- Nothing!

How to submit your assignment.

- This assignment is for practice only and not to be submitted or marked.

What happens if I miss the deadline:

This can not happen with this assignment!

Appendix A – a sample F.txt file

```
A
B
C
::
D
E
F
END
B
::
G
```

The above file contains two FDs:

ABC -> DEF

B -> G