COSC6340: Database Systems
Project: Database Normalization in SQL

Instructor: Carlos Ordonez

1 Introduction

You will develop a Java program that generates SQL code to certify normal forms (NF). The input is a set of tables (relations) and a potential "candidate key (CK)" for each table (unknown if the given CK is indeed a CK).

That is, initially the user is trying to explore if a given set of columns are a candidate key or not. If the answer is yes, then the program tells the user if the table is already normalized with respect to that key.This kind of tool is useful when the data modeler has no clear idea about the contents of columns in big tables, and requires to modify the tables to be in 3NF.

2 Program requirements

Your program will generate SQL code based on input tables whose basic schema is defined in a text file. Do not use STORED PROCEDURES. Basically, you have to check normal forms against a given candidate key and several nonkey (nonprime) attributes. You can assume the "candidate" key has at most two attributes and there are at most 10 nonkey (nonprime) attributes.

The main goal of your program is to certify (verify) normal forms from 1NF to 3NF. The maximum number of column in a table will be 10 and maximum number of tables in a input file will be 10.

1. For 1NF, you must check duplicates and a clean key. No atomic value checking since a table in the DBMS will be given as input. The given key may be composite. Therefore, every attribute must be clean. Bear in mind you are given a candidate key and the table itself already has another valid primary key. However, this "candidate key", proposed by the user, may have duplicates or nulls. In short, you are checking whether you have a potential new key.

2. For 2NF check partial functional dependency (FD) on the candidate key. If there is a simple key given then, the table is in 2NF. If you have a composite key, you must analyze $every$ subset of columns for partial FDs against every subset of the composite key. Rows with nulls should be skipped. You can assume the candidate key has at most 2 attributes.

3. For 3NF, as a simplification, we will only require you to check all the transitive dependencies between single nonkey attributes. That is, all transitive (if any) FDs of the form $A \rightarrow B$, where A and B are columns in your input table, should be analyzed.

2.1 Programming details

1. You need to use JDBC to submit queries, but your program should be able to generate an SQL script file for all the queries used in your program. The user must provide an input text file and output text file. If any of these parameters are missing, your program should return an error message and stop.

2.2 Theory

Remember a relation in 3NF is also in 2NF and 1NF and a relation in 2NF in also in 1NF. Therefore, if the program is only certifying, when a lower NF is not satisfied the program should not continue checking further NFs (since that would be unnecessary). Your program will get just one key without specifying other (secondary)

candidate keys. You are required to check normal forms based on that key only. SQL code generation: You can use any combination of SELECT statements . The only constraints that you have are not to modify the input table in any way and the output table should be called "NF_teamx". (x - team number) Drop the output table if exist else create a new one before you start writing into it.

3 Program call and Output

Here is a specification of input/output. There are just two input parameters: the input file and the output file.

- syntax for call: "java NF database=dbxyz.txt;output=result.txt".
- Example of input file (dbxyz.txt):

```
Employee(employeeid(k),name,salary)
Department(deptid(k),subdeptid(k),deptname,deptSalesAmt)
```

- If the connection to the DBMS fails, you have to send an error message. If the connection to the DBMS is successful, you should generate the "NF_teamx.sql" file and dump the SQL generated through your program.Add comments in the SQL script.
- The SQL of your generated script should be properly formated and indented.
- A summary result table, written into the output file", with 4 columns should be created in the DBMS. The first column should specify the table name, the second column should include the evaluated NF, the third column should say if the table was or not in the specified form, the following column should state the reason of why the form was violated, You should include all violations (define the column as char(200)), and evaluation of the derived tables if required. Call the output table "NF_teamx" and you can do a "SELECT * FROM NF_teamx;" as your last statement.
- Use JDBC to send SQL queries to certify each normal form.
- The output in simple text form is the only requirement. Your program will be tested with scripts. Therefore, a GUI is discouraged.

4 Examples of input and SQL

The input file contains a set of table schemas R ,$R1$,$R2$ and $R3$ .All tables has PK i (artificial) and a single/set of "candidate" key indicated with (k) notation in the schema . Notice i is NOT considered in the normalization certification, but it is used just to have a practical valid PK.

R

i	K1	A	B
1	1	1	0
2	2	1	0
3	3	2	0
4	4	2	0
5	1	2	0
6	3	1	0

R1

i	K1	K2	A	B
1	1	1	1	0
2	2	1	1	0
3	3	2	2	0
4	4	2	2	0
5	1	2	2	0
6	3	1	1	0

R2

i	K1	K2	A	B	C
1	1	2	0	0	1
2	1	3	1	1	0
3	2	1	0	0	1
4	2	2	1	1	0
5	2	3	0	0	1

R3

i	K1	A	B
1	1	1	0
2	2	0	1
3	4	1	0
4	8	1	1
5	16	0	0
6	32	1	0
7	64	0	0

4.1 Output: NF certification

If the program is called with "java NF database=dbxyz.txt;output=result.txt", then the input/output is:

Input file

dbxyz.txt

R(K1(k),A,B)
R1(K1(k),K2(k),A,B)
R2(K1(k),K2(k),A,B,C)
R3(K1(k),A,B)

Output File
result.txt (Insert the same in NF_teamx (Output table))

Table	Form	Y_N	Reason
R	1NF	N	$K1$ has duplicate values
R1	1NF	Y	
R1	2NF	N	$K2 \rightarrow A$, $K2 \rightarrow B$, $K1 \rightarrow A$
R2	1NF	Y	
R2	2NF	Y	
R2	3NF	N	$A \rightarrow B$, $B \rightarrow A$, non-keys should be mutu
R3	1NF	Y	
R3	2NF	Y	
R3	3NF	Y	

The following table S, which requires a NF certification, is in 3NF (input file:S(C(k),D)) and the result table should have YES for 1NF, 2NF and 3NF:

S

i	C	D
1	1	0
2	2	1
3	3	0