CSCI 620/Section 01-03/Rivero, Introduction to Big Data, Spring 2205

Assignment 3 – Normalization

Description

- We are going to work with the schema described in Assignment 2.
- IMPORTANT: Everything that is output by your programs (functional dependencies, candidates keys, relations) must be lexicographically sorted; otherwise, the grading software will fail to match your output with the expected results.

Your tasks

1.- Provide a Gradle project named NaiveFDDiscovery that takes a database connection and a relation r as input, and outputs a text file in which each line is a functional dependency in r that must have the following format:

a1, a2, a3, …, ak -> aj

where each ai is an attribute in r. You must use the naïve approach to find functional dependencies, i.e., for all combinations of attributes in r, you need to find all functional dependencies that are minimal and non-trivial. Since you must generate all the possible combinations in a bottom up fashion, you should use the Sets.combinations method provided by the Guava library. You must create combinations of attributes from one to the number of attributes in r minus one. Note that you must directly discard trivial and non-minimal functional dependencies during your search. There are additional restrictions and hints. Use the project template and grading software. (15 points)

2.- Provide a Gradle project named CandidateKeyDiscovery that takes a relation rel and a set of functional dependencies as input. The relation rel is in the following format:

rel(a1, a2, …, an)

where ai are attributes that consist of one or more letters. The functional dependencies have the same format as in Question 1 but the right-hand side may contain multiple attributes separated by commas as well. The project must output a text file in which each line contains a candidate key as follows:

ai, aj, ak

where ai, aj and ak are attributes of rel. There are additional restrictions and hints. Use the project template and grading software. (30 points)

3.- Provide a Gradle project named CanonicalCoverComputation that takes a relation r and a set of functional dependencies as input having the same format as in Question 2, and outputs the canonical cover of the functional dependencies in a text file in which each line is a functional dependency. There are additional restrictions and hints. Use the project template and grading software. (30 points)

4.- Provide a Gradle project named ThreeNFDecomposition that takes a relation r, a canonical cover and a set of candidate keys as input having the same format as in Questions 1 and 2, and outputs the 3NF decomposition of relation r in a text file. Each line in the file is a new relation

that must have the same format as in Question 2. There are additional restrictions and hints. Use the project template and grading software. (25 points)

Submission instructions

- Use the software template provided in myCourses.
- Submit a single ZIP file to myCourses that must be named as your RIT user, e.g., crrvcs.zip. Do not include '@rit.edu.' The file must contain folders named 'NaiveFDDiscovery', 'CKDiscovery', 'CCComputation' and 'TNFDecomposition' containing your Gradle projects.
- Everything will be graded on a Linux machine, so you must always use the exact names provided in this document, software template and grading software.

Grading rubric

- Check the grading software.