7CCSMDDW

Normalization

1 Normalization Challenge

Imagine you have been fired by an online training company to help them to analyse their data about student performance. As the managers of the training company are not proficient in database systems, their first attempt at producing the database has yielded a somewhat flat data model (1NF) that is contained in the file 1NF.sql. Before you are asked to perform any data analysis, you have to obtain a normalised database that is not prone to update abnormalities.

1. Load the database into your database.
2. Analyse the data contained and explain, informally, why the table is not well-designed.
3. Obtain the 2NF and 3NF.
4. As the number of rows and columns in the table is quite big, it is not possible for you to split it into different tables manually. Thus, you need to prepare SQL statements that produce and populate a set of tables in 3NF. It is possible that you may also need to create tables to store the 2NF tables temporary. Here are some tips:
   - Statements like CREATE TABLE AS SELECT may be very useful
   - The clause DISTINCT of the SELECT statement may be very useful
   - The final tables need to have primary and foreign keys as needed. The statement ALTER TABLE may be very useful for this.
5. Try writing a query the reproduces the flat data table from your normalised tables.

2 A More Realistic Challenge

Having a flat table that is already in 1NF is is a good starting point for building a normalised database. Unfortunately, it is very frequent that you are given a badly-formed dataset that misses some values, uses some values as the NULL value etc. In this challenge assume that the online training company has provided you with a badly-formed dataset (0NF.sql) and you have to transform it into 1NF.

- This is a more complex challenge. Take your time. You may have to do some research about how you can fill the missing values using SQL commands. Note that:
  - Rows have a column named row order that contains the order in which the rows where inserted into the flat table
  - In cases in which a column has a NULL value you can assume that the the value is missing and it must be set to the value from the previous known value (the previous row that has a not null value in that column). TIPS: the function `COALESCE` may be useful.