CSC540 TEAM D PROJECT REPORT 2 REGRADE REQUEST

Dear instructors:

Thank you for the insightful comments for our project report. We find they will help improve our design for the final report. However, we have a few specific concerns, which we discuss below. In accordance with the class regrade policy, we elaborate why we believe a higher grade is warranted for our solutions.

Page 3. Comment: "Do you really need this table if it's just one field in it?"

Our reasoning behind this approach was two-fold: (i) this allows for publications to have multiple topics associated with them. We found this to be trivial and did not explicitly mention it as an asssumption. This allows for fine-grained topics, rather than only supporting broad, general topics. (ii) We realize this could be realized via just the use of the *HasTopic* relation, by making both fields part of a primary key. However, as discussed earlier in class, we feel this design approach allows for future-proofing. Specifically, suppose we would like to associate with each topic, in the future, more information (as a concrete example, suppose we would like to say topics are unique, but are part of a broader subject). Apart from the future-proofing, we found it semantically odd to include the topic itself in the *HasTopic* relation; this allows for future developers to easily understand the semantics of each relation.

Page 12. Comment: "You made a number of things PK in Q2 but not translated here"

We cede that this is true. In Q2, we made the attributes part of a primary key to be correct in accordance with the rules of constructing formal relations. With a concrete SQL table, there are 3 cases with a potential issue (the remaining case is a set of pairs formed by unique, non-repeating values from each primary key domain):

- Multiple editors per publication
- Multiple publications assigned to an editor
- Multiple copies of a specific editor and publication

The first two are plausible cases, leaving only the last case as cause for concern. While this would not lead to erroneous results, it does result in a potential waste of space. We solve this (and other related security issues) using three-step validation: all inputs are validated at the client, server, and database end. For this specific instance, it is not possible to validate at the client end (indeed, it would be a major security flaw!); but it is certainly possible to validate at the server side. While we admit that our implementation lacked the database validation through a primary key or unique constraint, our argument for requesting more points is that it was not strictly necessary and does not affect the proper functioning of the application.

Regards,
Team D