CS 4611 – Spring 2017 – Homework 1
Assigned: 09/07/2017
Due: 09/19/2017
Submit both a hard copy of your work in class, and a soft copy to canvas.

Objectives: The objectives of this homework are the following:

- **You will practice how to write queries in SQL that perform simple selections, updates, deletions, projections, renaming, aggregation and joins.**
- **You will learn how to write sub-queries in SQL.**
- **You will learn how to interpret a given set of relational schemas.**
- **You will review the concept of foreign keys.**
- **You will learn how to use set membership operators (IN) and some of the set comparison operators in SQL (SOME, ALL, EXISTS), depending on how you write your queries.**

Important Note: In Questions 3 and 4 of this homework you will write queries in paper using SQL (i.e., not using a DBMS). I think that some of these queries can be challenging, especially the ones marked with asterisks (*). Don't get discouraged if you can't make progress with some of them. My advice is for you to start as early as possible with Questions 3 and 4 and don't leave it until the last minute. If you get stuck with any query, no matter how simple, don't hesitate to send me an email asking for help, or to come to office hours. We will also practice more on how to write these queries in SQL in the next couple of classes and in the next lab.

We have the following scenario. The Duluth Public Library wishes to update its old manual loan system by implementing its database in a relational DBMS. Assume that after talking to the people in charge of the loan system at the library, you come up with the following table schemas. Note that the primary keys of each schema are underlined.

book(book_id, title, publisher_name, no_of_copies)
book_authors(book_id, author_name)
publisher(name, address, phone)
book_loans(book_id, card_no, date_out, due_date)
borrower(card_no, name, address, phone)

1. For each table write its foreign keys, if it has them.
2. Write down the SQL code to create each of these tables. Include all types of constraints that you see fit. Note that you don't have to code them in Oracle, just write them down in a document.
3. Now you'll see a set of queries in natural language. *You are asked to write down these same queries using SQL*. Note that you don't have to code these queries in Oracle, just write them down in a document. Some queries could be a bit more difficult than others, so I have tried to indicate their relative order of difficulty with asterisks.

a. (Simple select) Retrieve the id, title and publisher of all the books stored in the database.
b. (Simple update) Update the name of publisher "Mc Graw Hill" to "McGrawHill."
c. (Simple delete) Delete all those books authored by "J.K. Rowling."
d. (Projection) Find the name and address of every borrower. In this query, rename the names of the attributes of the query result set to "Borrower name" and "Borrower address."
e. (Select with a complex condition) Find all the attributes of books with titles that start with an 'A' or a 'B' and such that there are at least 5 copies of them at the library. Note: For this query write two versions: one using *IN* for checking of values are contained in a set, and another one with *OR*.
f. (Aggregation) Retrieve for each publisher its name and the number of different books it has published (that is, not counting copies because copies are the same book).
g. (Join of 2 tables) For every book, retrieve its title and authors.
h. [*] (Join of 3 tables) Retrieve for each borrower the name of the borrower and the titles of the books that he/she has borrowed.
i. [*] (Grouping and Aggregation) For each publisher that has published at least 10 different books, retrieve the publisher name and the number of books it has published.
j. [**] Find the names of the "borrowers" that have never borrowed a book.
k. [**] (Aggregation) Retrieve the average number of books that each borrower has borrowed.

4. BONUS: The following queries are a bit more challenging. If you struggle with these, send me an email, or come to office hours! Now, try to write down these queries using SQL:
    a. [***] (Aggregation with sub-queries) Find the names of the borrowers that have borrowed more books than the average. (Hint: Query h can be useful here)
    b. [***] Retrieve all pairs of authors that have been coauthors in at least one book.
    c. [***] Find a borrower who has borrowed all books written by author "Patterson."