06-21980/21923 *Fundamentals/ICY: Databases*
Autumn Semester 2018

The University of Birmingham
School of Computer Science

Class Test 1 Repeated

Use a text editor of your choice to create a file "`test1r.sql`" and type in the following lines:

```
-- Start time: ...
-- This is my individual work. I understand there are penalties for plagiarism.
```

Add the start time, sign your name at the end, and **save the file**.

- You will write all your answers into this file: "`test1r.sql`".
- Make sure that you format your queries and use appropriate indentation.
- And **Relax**! This is no different from any other exercise.

Note on dates Dates in SQL are written in the format `date '16 October 2018'`. Dates can be compared using operators like "<". Adding or subtracting an integer to/from a date gives another date. Taking the difference between two dates gives an integer (the number of days). Today's date can be obtained via the expression `current_date`.

The problem domain you will work with is that of a library for a small town. The library has a number of books, some with multiple copies. The borrowers register with the library as members and can borrow up to 2 books at any given time. The book are issued for one month at a time, but if another member wants the book, the library "recalls" the book to be returned within a week. To request a book to be recalled, the other member would place a "reservation" for it. If multiple copies of a reserved book have been issued out, the copy that was the *earliest* to be issued is the one that will be recalled.

The requisite information for this application is stored in the following tables:

```
  Table "member"
 Attribute  |         Type
------------+----------------------
 mid        | integer
 firstname  | character varying(20)
 lastname   | character varying(20)
 address    | character varying(50)
```

```
  Table "book"
 Attribute  |         Type
------------+----------------------
 bid        | integer
 authorfirst| character varying(20)
 authorlast | character varying(20)
 title      | character varying(50)
 year       | integer
 numcopies  | integer
```

```
  Table "issues"
 Attribute |     Type
-----------+----------------------
 bid       | integer
 mid       | integer
 issuedate | date
 returndate| date
```

```
  Table "reservations"
 Attribute |     Type
-----------+-----------------
 mid       | integer
 bid       | integer
 rdate     | date
```

In the table `issues`, "`mid`" refers to the member that borrowed the book. In the table `reservartions`, "`mid`" refers to the member that placed the reservation. When a book is first issued, its "`returndate`" field would be `NULL`. After the book is returned, the field would get updated.

Assume that the four tables together cover several years of book issues and reservations made in this library. You may also assume that there are no nulls in the tables, except for the `returndates` mentioned above.

Develop SQL queries to answer the following questions:

(a) Find the total number of books (copies) currently issued out to members, i.e., book copies not yet returned. 15%

(b) List all the members, by their `mid`, and the total number of different books they have borrowed over the years. 20%

(c) List the names of all the members that ever borrowed books written by Jane Austen, along with the book titles and dates of borrowing. The list should be ordered by last name, firstname and the date of borrowing. 20%

(d) List all the outstanding reservations, i.e., members and the books they reserved. A reservation is "outstanding" if the book has not yet been issued to the member. (You can just list the id numbers of the members and books involved.) 20%

(e) List all the books that have reservations along with the members from whom the books should be recalled, i.e., members that borrowed those books the *earliest*. (For the purpose of this question, you can assume that all the records in the reservations table are outstanding, and that there is no more than one reservation for any particular book.) 25%

*Type your queries into the file `test1r.sql` and **save it**. Load it into `psql` using `\i` command, to make sure there are **no syntax errors**. If you are unable to correct syntax errors, then you should put that answer in comments so that the file will load properly. When you are done, upload the file to Canvas under the "Class test 1 Repeated" assignment.*

Double check to make sure that your file uploaded correctly.