School of Computer Science, McGill University

COMP-421 Database Systems, Winter 2024

Written Assignment 1: SQL

Due Date Feb 16, 12:00pm (noon)

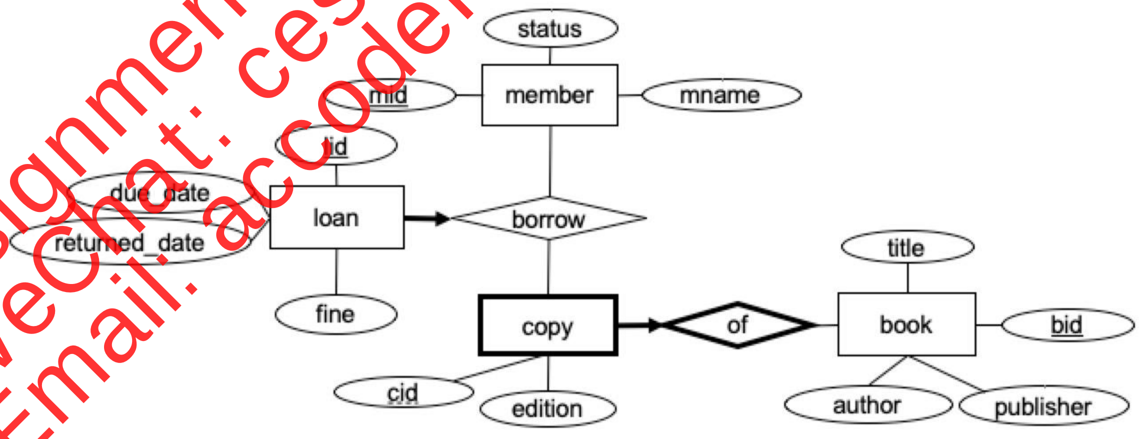
This is an individual assignment. You are required to work on your own to create the solution.

This assignment will be graded by an automated system. It is very important to read the instructions and follow them exactly. Not following those instructions can result in your assignment receiving 0 points.

Please pause and go over the included COMP421-A2-SQL-FormatGuidelines.pdf completely once before you continue reading this assignment description. It describes how to setup the sample data set that is provided to you and to ensure that you do not are not going down the wrong path on how to prepare your solutions.
Turn in one tar file in mycourses.

Ex. 1 — SQL (70 Points)

The following ER model and relational model are designed for a library. Each book has a unique book id with title, author and publisher information stored in the system. A book can have multiple copies or no copies at all. Each copy has a copy id which is unique only for a given book id. Each Member is labeled with a status (i.e. STUDENT, REGULAR, YOUTH). For the sake of simplicity we will assume that members do not change their status. Members can borrow books and must return the book before the due date. Overdue fines will be charged if books are returned late. Fine and returned date are initially NULL in the loan table and are updated only on return of the book. Fine is updated to 0 upon return within due date. A member can borrow many copies altogether and return them separately. That is why each loan record only associates with one copy.
A sample schema and few records have been provided as **setup.sh** Please add more records into this as you may need to test various scenarios. Use your individual database accounts to work on the assignment. **DO NOT** use your project groups database account as it is shared between all of your team members.



`book(bid, title, author, publisher)`
`copy(bid, cid, edition)`
bid is a foreign key to book.
`member(mid, mname, status)`
`loan(lid, bid, cid, mid, due_date, returned_date, fine)`
mid is a foreign key to member and (bid, cid) is a foreign key to copy.

Important !!

All the sql solutions will be evaluated by an automated system, which compares the output data produced by executing your query on our dataset with the expected output result for the correct query. So it is important that you include the correct column names, in the correct order, perform any ordering on output tuples as asked etc.

Double check your SQL for **typos**, for example if you spelt '*STUDNT*' instead of '*STUDENT*', a query might not return the correct records and you will not get any points.
While the columnn and table names are not case sensitive, **the data itself can be case sensitive**. So do not write '*Student*', where it was required to write '*STUDENT*' this can produce no results or wrong results.
For more details read the attached sql formatting guide. If you have questions about this post it in the discussion forum for assignment 2. **Remember you will either get 0 or all points for a given SQL question !!**

For this assignment you will not create views or tables in your solution. You can however use derived tables as we saw in class in your SQL. All your answers should be comprised of only a select query. **Output ONLY the attributes in the question, following the exact order mentioned in the question**. Adding attributes not mentioned can result in a 0 score !

Unless specified, your output query should not produce duplicate results in your output resultset.
Use the technique taught in class to eliminate duplicate records from the output.
Where an output ordering is asked for, remember to order the output records. The technique for this was also shown in class.

1. (2 Pts) List the book id, copy id and edition number of all 2nd or 3rd edition copies. Ordering the output by book id and copy id.
2. (2 Pts) List the book id of books with 3rd edition but do not have a 2nd edition. Order the output by book id.
3. (2 Pts) List the book id and title of the books ever borrowed by the member with member id 11111111. Order the output by book id.
4. (2 Pts) List the book id, title and overdue fine for loans that the member with member id 11111111 has ever been charged. Order the output by book id.
5. (3 Pts) List the book id and title of books that both member 11111114 and member 11111118 have borrowed it before. Order the output by book id.
6. (3 Pts) List the book id and title of books that have ever been borrowed by member 11111114 but not by member 11111118. Order the output by book id.
7. (2 Pts) List the book id and title of books published by Vintage and ever borrowed by the member with member id 11111111. WITHOUT using joins. Order the output by book id.
8. (3 Pts) List the book id and title of books published by Vintage and ever borrowed by the member with member id 11111111. Using joins. Order the output by book id.
9. (3 Pts) List the book id and title of books published by Vintage and ever borrowed by the member with member id 11111111. Using a correlated subquery. Order the output by book id.
10. (4 Pts) List the book id and title of books that have at least one copy but have not been borrowed yet. Using a correlated query. Order the output by book id.
11. (2 Pts) Find the total number of members, name the output column `nummembers`.
12. (3 Pts) Find the total number of members who have been charged at least a fine before, name the output column `nummembers`.

13. (3 Pts) List the id and name of members who have more than or equal to 3 overdue charged loan records. Order the output by member id.
14. (4 Pts) List the book id and the number of loans of each book (name this column `numloans`). Order the output by descending order of numloans and then ascending order of book id.
15. (5 Pts) List the average number of loans of a member. Name the average column `avgloans`.
16. (6 Pts) List `status` and average number of overdue loans of members each `status`. Name the average column `avgloans`. Include members with no overdue loans when computing the average. Order the output by status.
17. (5 Pts) List each book id and how many times the copies of that book have been returned by members (name this column `numloans`). If a book was never loaned, `numloans` must be 0. Ignore the books that do not have a copy in the library. Write this query without using any outer joins. Order the output by book id.
18. (6 Pts) Redo the above question with an outer join. **Hint:-** use COALESCE
19. (4 Pts) Find the book that has the highest number of loan records associated with STUDENT members. List the book id and the number (name this column `numloans`). Order the output by book id.
20. (6 Pts) List the member id and total overdue fine of REGULAR members whose total overdue fine is larger than average overdue among all members. Name this column `totalfine`. Order the output by decreasing order of overdue fine. When calculating the average, consider for the members with the total overdue fine of 0 (include) and ignore those with only NULL values in the table.

Submission

Review the instructions in the attached formatting guide to test your individual solution sqls and then follow the instructions on creating the **.tar.gz** file and submit it. Remember to verify the contents of the tar file to ensure you ran the tar command correctly and it captured the correct contents. You are responsible to verify the integrity of the file you submit to MyCourses. There will be no exceptions.

Please turn in your submission in mycourses under assignment 2. Submissions to wrong folders may not get graded. **You are responsible to download and verify that your submission is correct (not corrupted or incorrect file, etc).** There will be no accommodations if you decided to ignore doing due diligence (you will get a 0).

Resubmissions are allowed (please do not email me saying you do not know how to resubmit files). In such cases, the last submission will be graded. Please ensure that you are submitting a complete set.

Important Database Etiquette

Please remember that the server `winter2024-comp421.cs.mcgill.ca` and the databases installed them are meant to be used for the course work of COMP 421. Using these systems for other work (including other course work) is not allowed. You are sharing these resources with other classmates, so be mindful of its proper use.

Please go through the `README` file in MyCourses under "Database" → "Connecting to DB Servers" for comprehensive list of restrictions. Not following them may result in your id being disabled as stated in the course outline.

Questions ?

Please use **Ed** for any clarifications you need (`assignments` → `Assignment 2`). Do not email the instructor or TAs as this leads to a lot of duplicate questions and responses (not an efficient system). Such emails will not receive any replies.

Please check the pinned post "A2 general clarifications" in Ed before you post a new question. It might have been already addressed there, in which case we will not address it again.

Questions about general clarifications must be marked public (as other students will also benefit from this and may even have a valid response). TAs and Instructors upon their discretion may toggle any private posts into public mode for the benefit of the student population at large.

There will be specific office hours for the assignment that will be announced closer to the due date.

Extensions and Late submissions

- Remember, your submission is **due on Feb 16th 12:00 noon**. There is no place for excuses.
- A maximum of 3 days of late submission is allowed with a penalty of 10% of the achieved grade per day (rounded up, even for a minute).
- Penalty waivers are granted only for medically documented emergencies and under any circumstances **will not be granted unless requested 24 hours before the due**. I expect you to be better organized and get the job done ahead and leave the last 24 hours only for one last final check.