

Hand in a hardcopy of your answers (printed or handwritten). Include your name, and be sure to clearly identify which problems you are answering and which homework the problems are from since problems from multiple homeworks will be collected at the same time. These problems will be collected on Monday 9/21 and Monday 9/28.

You may use the MySQL database to develop and test your queries, but be sure to still think through what each query is doing for yourself in order to practice and build your skills at reading and understanding SQL. Being able to reason through SQL is important for building up complex queries, and is essential for verifying that your queries are correct — they need to work in any case and not just for a particular instance of the database. (The example database provided is not guaranteed to be sufficient to catch all edge cases or reveal common logic bugs.) Instructions for accessing MySQL can be found on Canvas.

Use the following schema for these problems.

```
BOOK(Book_id, Title, Publisher_name)
BOOK_AUTHORS(Book_id, Author_name)
BOOK_COPIES(Book_id, Branch_id, No_of_copies)
BOOK_LOANS(Book_id, Branch_id, Card_no, Date_out, Due_date)
BORROWER(Card_no, Name, Address, Phone)
LIBRARY_BRANCH(Branch_id, Branch_name, Address)
PUBLISHER(Name, Address, Phone)
```

The foreign keys: ($\rightarrow$ reads as “refers to”)

- BOOK.Publisher_name $\rightarrow$ PUBLISHER.Name
- BOOK_AUTHORS.Book_id $\rightarrow$ BOOK.Book_id
- BOOK_COPIES.Book_id $\rightarrow$ BOOK.Book_id
- BOOK_COPIES.Branch_id $\rightarrow$ LIBRARY_BRANCH.Branch_id
- BOOK_LOANS.Book_id $\rightarrow$ BOOK.Book_id
- BOOK_LOANS.Branch_id $\rightarrow$ LIBRARY_BRANCH.Branch_id
- BOOK_LOANS.Card_no $\rightarrow$ BORROWER.Card_no

Sample data for testing queries is available in the `ex_library` database in MySQL. Keep in mind, however, that what’s in the database not be sufficient to reveal every subtlety.

For each of the following, give a brief English description of the result of the query. Describe the result rather than the working of the query. For example, “names and salaries of employees working in department 5” rather than “first find the rows of EMPLOYEE where the department number is 5, then get the first name, last name, and salary”.

1.

```
SELECT Name
FROM BORROWER JOIN BOOK_LOANS USING (Card_no)
WHERE Date_out = ( SELECT MIN(Date_out) FROM BOOK_LOANS )
```
2.

```
SELECT Title
FROM BOOK
WHERE Book_id IN (
    SELECT Book_id
    FROM BOOK_COPIES JOIN LIBRARY_BRANCH USING (Branch_id)
    WHERE Branch_name = 'Geneva Public Library'
)
```
3.

```
SELECT Name
FROM PUBLISHER
WHERE NOT EXISTS (
    SELECT * FROM BOOK WHERE BOOK.Publisher_name = PUBLISHER.Name
)
```
4.

```
SELECT Name
FROM BORROWER
WHERE EXISTS (
    SELECT *
    FROM BOOK_LOANS JOIN BOOK USING (Book_id)
    WHERE BOOK_LOANS.Card_no = BORROWER.Card_no AND
        BOOK.Publisher_name = 'DAW'
)
```
5.

```
SELECT Name,
    ( SELECT COUNT(*)
      FROM BOOK_LOANS
      WHERE BOOK_LOANS.Card_no = BORROWER.Card_no ) AS num_loans
FROM BORROWER
```
6.

```
SELECT Title, Branch_id, No_of_copies
FROM BOOK JOIN BOOK_COPIES USING (Book_id)
WHERE No_of_copies >= ALL (SELECT No_of_copies FROM BOOK_COPIES)
```

```

7. SELECT Name, COUNT(*) AS num_loans
   FROM BORROWER JOIN BOOK_LOANS USING (Card_no)
   GROUP BY Card_no
   HAVING COUNT(*) > (
       SELECT AVG(loan_count)
       FROM (
           SELECT COUNT(*) AS loan_count
           FROM BOOK_LOANS
           GROUP BY Card_no
       ) AS PER_BORROWER
   )

```

Each problem in this section contains an English description of a query and one or two attempts at expressing it in SQL. For each problem:

- Give the actual results (the table) that would be returned by the SQL query or queries. **Use the sample data given on the next page for these exercises, not what is in ex.library!** If an error occurs, indicate that instead.
 - Explain the discrepancies: identify specifically what about the SQL causes each to fail to match the English description (or fail entirely).
 - Fix: give SQL for a query which correctly expresses the English description. Use subqueries even if you can think of another way to write the query. (It's fine to also give a solution without subqueries, just don't *only* give a solution without subqueries.)
8. Find the unique copy counts — the book-copy records whose number of copies doesn't match any other record.

```

SELECT Book_id, Branch_id, No_of_copies
FROM BOOK_COPIES
WHERE No_of_copies <> ALL ( SELECT No_of_copies FROM BOOK_COPIES )

```

9. Find the book-copy record(s) with the most copies.

```

SELECT Book_id, Branch_id, No_of_copies
FROM BOOK_COPIES
WHERE No_of_copies > ALL ( SELECT No_of_copies FROM BOOK_COPIES )

SELECT Book_id, Branch_id, No_of_copies
FROM BOOK_COPIES
WHERE No_of_copies > ANY ( SELECT No_of_copies FROM BOOK_COPIES )

```

BOOK_AUTHORS

Book_id	Author_name
101	C.J. Cherryh
103	C.J. Cherryh
103	Jane Fancher
105	Maureen F. McHugh

BOOK_COPIES

Book_id	Branch_id	No_of_copies
101	1	3
102	1	2
103	2	5
105	1	5

Borrower

Card_no	Name	Address	Phone
001	Bren Cameron	Shejidan	456-555-1111
002	Danny Fisher	Finisterre	789-555-2222
003	Arthur Dent	Islington	630-555-3333

BOOK

Book_id	Title	Publisher_name
101	Foreigner	DAW
102	Invader	DAW
103	Defiance	NULL
104	Hammerfall	Eos
105	Mission Child	Eos

BOOK_LOANS

Book_id	Branch_id	Card_no	Date_out	Due_date
101	1	001	2026-01-01	2026-01-15
102	1	001	2026-01-05	2026-01-19
103	2	002	2026-01-10	2026-01-24
103	2	001	2026-02-01	2026-02-15
105	1	001	2026-02-01	2026-02-15

Table 1: Data for questions 8–15.

10. Find the book(s) written by the same author as 'Foreigner'.

[illegible]

11. For each borrower, show the branch(es) of their most recent loan.

```
SELECT Name,
       ( SELECT Branch_id FROM BOOK_LOANS
         WHERE BOOK_LOANS.Card_no = BORROWER.Card_no AND
               Date_out = ( SELECT MAX(Date_out)
                           FROM BOOK_LOANS
                           WHERE Card_no = BORROWER.Card_no )
       ) AS last_branch
FROM BORROWER
```

12. Find every book whose publisher is different from book 103's.

```
SELECT Title
FROM BOOK
WHERE Publisher_name NOT IN ( SELECT Publisher_name
                              FROM BOOK
                              WHERE Book_id = 103 )

SELECT Title
FROM BOOK
WHERE Publisher_name <> ALL ( SELECT Publisher_name
                             FROM BOOK
                             WHERE Book_id = 103 )
```

13. For each borrower, whether or not they currently have any loan checked out.

```
SELECT Name,
       EXISTS ( SELECT * FROM BOOK_LOANS ) AS has_loan
FROM BORROWER
```

14. List borrowers who have never had a loan period longer than 20 days.

```
SELECT Name
FROM BORROWER
WHERE NOT ( SELECT MAX(DATEDIFF(Due_date, Date_out))
            FROM BOOK_LOANS
            WHERE BOOK_LOANS.Card_no = BORROWER.Card_no ) > 20
```

DATEDIFF(date1,date2) returns the number of days between two dates, computed as date1 - date2 (a positive value if the first date is later, negative if the first date is earlier).

15. What is the average number of loans per borrower, among borrowers who have at least one loan?

```
SELECT AVG(COUNT(*))
FROM BOOK_LOANS
GROUP BY Card_no
```

Write SQL for each of the following. Use subqueries even if you can think of another way to write the query. Additionally use only the constructs discussed in class 9/4-9/14 (SQL basics, joins, aggregation and grouping, subqueries) and follow best practices for the use of those constructs even if you could write the query another way. In all cases it is fine to also give additional solutions without subqueries or which use other constructs, just don't *only* give solutions without subqueries or that use other constructs.

16. List the titles of books that have never been checked out (have no loan record).
17. List the book title, branch name, and borrower name for the loan(s) due last.
18. List the branch(es) that have more different books than Wood Library does.
19. For each library branch, show its name and the title(s) of the book(s) stocked there in the greatest quantity.