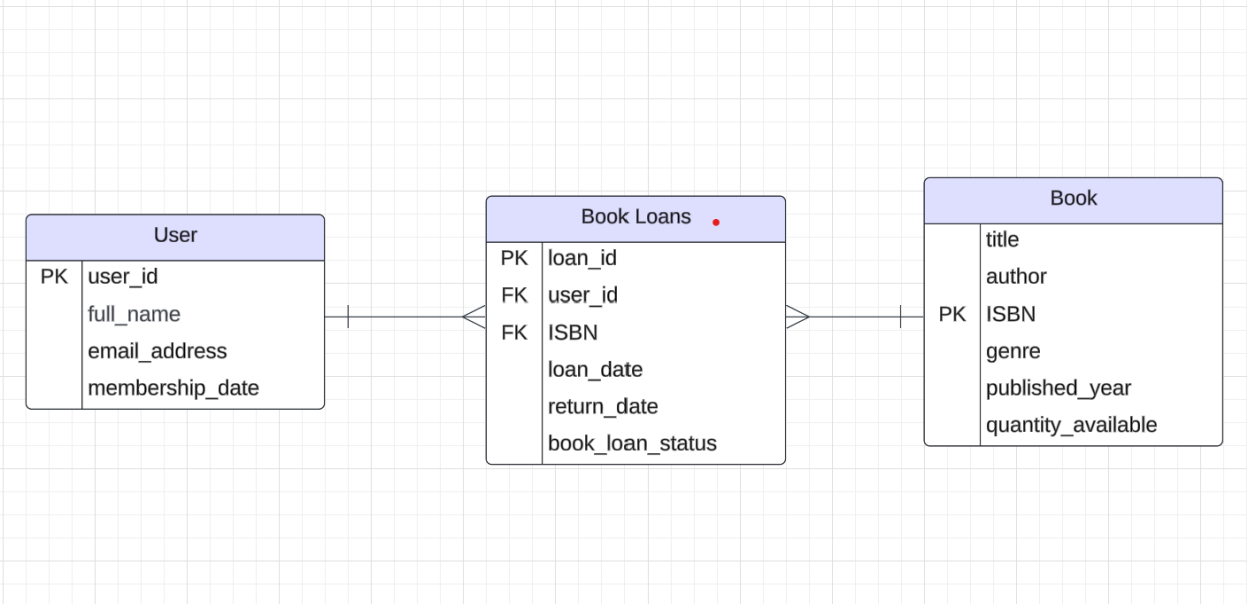
Sean Ashton V. Regalado

PART 1: Conceptual Design



PART 2: Logical Design

```sql
CREATE TABLE users (
    user_id SERIAL PRIMARY KEY,
    full_name VARCHAR(50) NOT NULL UNIQUE,
    email_address VARCHAR(50) NOT NULL UNIQUE,
    membership_date DATE NOT NULL
);

CREATE TABLE books (
  title VARCHAR(100) NOT NULL,
  author VARCHAR(100) NOT NULL,
  ISBN VARCHAR(14) PRIMARY KEY,
  genre VARCHAR(100) NOT NULL,
  published_year INT NOT NULL,
  quantity_available INT NOT NULL
);


CREATE TABLE book_loans (
  loan_id SERIAL PRIMARY KEY,
  user_id INT NOT NULL,
  ISBN VARCHAR(14),
  loan_date DATE NOT NULL,
  return_date DATE NOT NULL,
  book_loan_status VARCHAR(20) NOT NULL CHECK (book_loan_status IN ('borrowed', 'returned', 'overdue')),
  FOREIGN KEY (user_id) REFERENCES users(user_id),
  FOREIGN KEY (ISBN) REFERENCES books(ISBN)

);
```

PART 3: SQL Queries

```sql
INSERT INTO books (title, author, ISBN, genre, published_year, quantity_available)
VALUES
('To Kill a Mockingbird', 'Harper Lee', '978-0061120084', 'Fiction', 1960, 5),
('Pride and Prejudice', 'Jane Austen', '978-0141439507', 'Romance', 1813, 3),
('The Lord of the Rings', 'J.R.R. Tolkien', '978-0547991029', 'Fantasy', 1954, 2);

INSERT INTO users (full_name, email_address, membership_date)
VALUES
('Sean Ashton V. Regalado', 'sean@gmail.com', '2024-12-01'),
('Ash I. Regalado', 'ash@gmail.com', '2024-12-11');


INSERT INTO book_loans (user_id, ISBN, loan_date, return_date, book_loan_status)
VALUES
(1, '978-0061120084', '2024-12-09', '2024-12-20', 'borrowed'),
(1, '978-0141439507', '2024-12-11', '2024-12-16', 'overdue'),
(2, '978-0547991029', '2024-12-10', '2024-12-13', 'overdue');


SELECT b.title, b.author, bl.loan_date
FROM books b
JOIN book_loans bl ON b.ISBN = bl.ISBN
WHERE bl.user_id = 1;



SELECT u.full_name, b.Title, bl.loan_date, bl.return_date
FROM book_loans bl
JOIN users u ON bl.user_id = u.user_id
JOIN books b ON bl.ISBN = b.ISBN
WHERE bl.book_loan_status = 'overdue';
```

PART 4: Data Integrity and Optimization

Explain how you would ensure:

- The prevention of borrowing books when no copies are available.

I would display the quantity of the book available in the UI so that the user can get informed if that book is available or not. I would also disable the borrow button if the quantity stored in the database is zero.

- Fast retrieval of overdue loans. (20 pts- with CODE and actual screenshot of performance)

```sql
CREATE INDEX idx_status ON book_loans (book_loan_status);
```

After Adding

```sql
EXPLAIN ANALYZE
SELECT u.full_name, b.title, bl.loan_date, bl.return_date
FROM book_loans bl
JOIN users u ON bl.user_id = u.user_id
JOIN books b ON bl.ISBN = b.ISBN
WHERE bl.book_loan_status = 'overdue';
```

```
                                Buckets: 1024  Batches: 1  Memory Usage: 9k
                                ->  Seq Scan on book_loans bl  (cost=0.00..
                                      Filter: ((book_loan_status)::text = '
                                      Rows Removed by Filter: 1
 Planning Time: 0.146 ms
 Execution Time: 0.109 ms
(15 rows)
```

After Adding

```
EXPLAIN ANALYZE
SELECT u.full_name, b.title, bl.loan_date, bl.return_date
FROM book_loans bl
JOIN users u ON bl.user_id = u.user_id
JOIN books b ON bl.ISBN = b.ISBN
WHERE bl.book_loan_status = 'overdue';
```

```
                Index Cond: (user_id = bl
  ->  Index Scan using books_pkey on boo
        Index Cond: ((isbn)::text = (bl
Planning Time: 0.640 ms
Execution Time: 0.064 ms
(11 rows)
```

Output of Queries:

```
Output:

CREATE TABLE
CREATE TABLE
CREATE TABLE
INSERT 0 3
INSERT 0 2
INSERT 0 3
         title          |   author    | loan_date
------------------------+-------------+------------
 To Kill a Mockingbird | Harper Lee  | 2024-12-09
 Pride and Prejudice   | Jane Austen | 2024-12-11
(2 rows)

        full_name         |         title          | loan_date  | return_date
--------------------------+------------------------+------------+-------------
 Sean Ashton V. Regalado | Pride and Prejudice    | 2024-12-11 | 2024-12-16
 Ash I. Regalado         | The Lord of the Rings | 2024-12-10 | 2024-12-13
(2 rows)

CREATE INDEX
```

Part 5: Reflection

What challenges might arise when scaling this database to handle millions of users and books? Suggest one solution for each challenge.

1. Data Storage Limitations – A solution can be is upgrading the server hardware or distributing data across multiple servers.
2. Backup and recovery – A solution is to use incremental backups to save only the changes made since the last backup.
3. Increased query latency – A solution for this is to simplify queries by avoiding unnecessary joins and subqueries.