CSE 3330 Database System & File Structures

11/02/2024

Project 2 library DB Part 1

Group 13, Section 004

Shaheen Nijamudheen, 1002101057

James Paul Nguyen, 1001983168

Ali Shirazi-Nejad, 1002062834

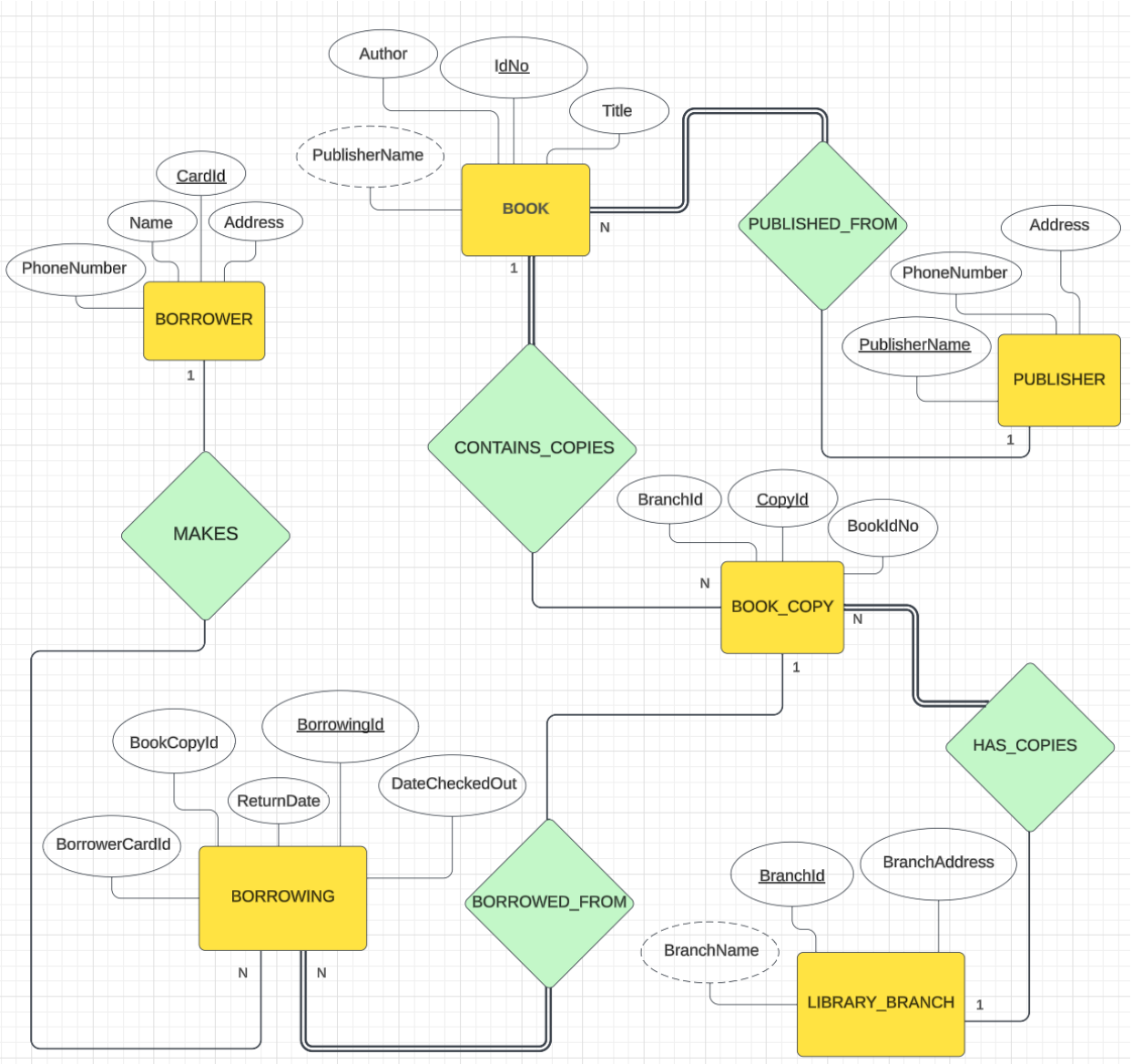
TABLE OF CONTENTS

INTRODUCTION

This document presents the design and development of a database for a Library Management System. The objective is to create an efficient system for tracking library resources, branches, and borrowing transactions.

MINI-WORLD DESCRIPTION

ER DIAGRAM:



ENTITIES AND ATTRIBUTES:

1. BOOK
    - IdNo (PK, integer)
    - Title (string)
    - Author (string)
    - PublisherName (FK from LIBRARY_BRANCH)

2. PUBLISHER
    - PublisherName (PK, string)
    - PhoneNumber (string)
    - Address (string)

3. LIBRARY_BRANCH
    - BranchId (PK, integer)
    - BranchName (string)
    - BranchAddress (string)

4. BOOK_COPY
    - CopyId (PK, integer)
    - BookIdNo (FK from BOOK)
    - BranchId (FK from LIBRARY_BRANCH)

5. BORROWER
    - CardId (PK, integer)
    - Name (string)
    - Address (string)
    - PhoneNumber (string)

6. BORROWING
    - BorrowingId (PK, integer)
    - BookCopyId (FK from BOOK_COPY)

- BorrowerCardId (FK from BORROWER)
- DateCheckedOut (date)
- ReturnDate (date, NULL if not returned)

MISSING OR INCOMPLETE REQUIREMENTS:

- Borrowing Limits:
  - In real life systems, there are usually limits in place for borrowing. A borrower cannot usually borrow more than N times (say 3 books at once).

- Due Dates and Penalties:
  - There is ambiguity surrounding what happens if a book is returned late. The requirements talk about borrowing and return dates but not regarding due dates and penalties.

- Multiple Authors:
  - In the ER diagram for the library management system, each book only has a single Author attribute; however, in real life, there can be multiple authors for a single book. A suggestion would be to use a many-to-many relationship between BOOK and AUTHOR. This will require an additional AUTHOR entity.

- Publisher Contact Details:
  - For PUBLISHER, the current design does not give ample contact information (PublisherName, PhoneNumber, and Address). An email would be beneficial for contacting a PUBLISHER.

- Book Genres or Categories:
  - In the ER diagram for the library management system, there is no way to specify the genre or category of books. Libraries often need to categorize books, so a new entity such as CATEGORY or GENRE would be beneficial.

- Library Employees:
    - There is currently no way for someone to manipulate the library system as a staff worker. There needs to be someone who handles borrowing, checking in/out, returns, or etc work that a librarian does. An EMPLOYEE entity would be beneficial.

- Book Condition:
    - There's currently no mention of the condition of book copies in a given library branch (e.g., New, Good, Damaged, etc.). This is crucial for tracking which books need maintenance (repairs, rebinding, tears, etc). A Condition attribute in BOOK_COPY could be beneficial.

ASSUMPTIONS:

- Unique Book Identification:
    - The system assumes each book is uniquely identified by an IdNo. Even if multiple copies of the same book exist, they are tracked separately as BOOK_COPY instances, each with a unique CopyId.

- Publishers are Predefined:
    - It is assumed that PUBLISHER records are predefined in the system. If new books are added, they must be linked to an existing PUBLISHER.

- Single Branch for Each Book Copy:
    - Each BOOK_COPY exists in only one LIBRARY_BRANCH at a time. There's no inter-branch transfer of books or sharing in the current design.

- Every Book is Published:
    - We assume every BOOK has a publisher. There's no need for books without publishers (e.g., self-published works) in this design.

- Return of Books:

    - The system assumes that borrowers always return books. No explicit mechanism is mentioned for permanently lost or unreturned books. If needed, you might track lost books separately.

- One-to-Many for Book to Copies:
    - The assumption is made that each BOOK will have multiple BOOK_COPIES, and at least one BOOK_COPY will exist for each book in the system.

- No Limit on Borrowing Time:
    - The system does not specify a maximum borrowing time, so it's assumed that ReturnDate can be set freely by the library without time restrictions.

EXPLANATION OF DESIGN CHOICES:

ENTITY CHOICES-

- BOOK: Represents each unique book with attributes like IdNo, Title, Author, and PublisherName to manage book-level details independently of copies.

- PUBLISHER: Stores publisher information (e.g., PublisherName, PhoneNumber, Address) for communication purposes.

- LIBRARY_BRANCH: Tracks branches by BranchId, BranchName, and BranchAddress, supporting multiple locations.

- BOOK_COPY: Represents individual copies of books, with CopyId, BranchId, and BookIdNo linking to the main BOOK entity.

- BORROWER: Identifies library users with CardId, Name, Address, and PhoneNumber.

- BORROWING: Records each transaction, including BorrowingId, BookCopyId, BorrowerCardId, DateCheckedOut, and ReturnDate to track borrowing history.

RELATIONSHIP CHOICES-

- CONTAINS_COPIES (BOOK to BOOK_COPY):
  - Type: One-to-Many. Each book can have multiple copies.
  - Total participation on BOOK side, ensuring every book has at least one copy.

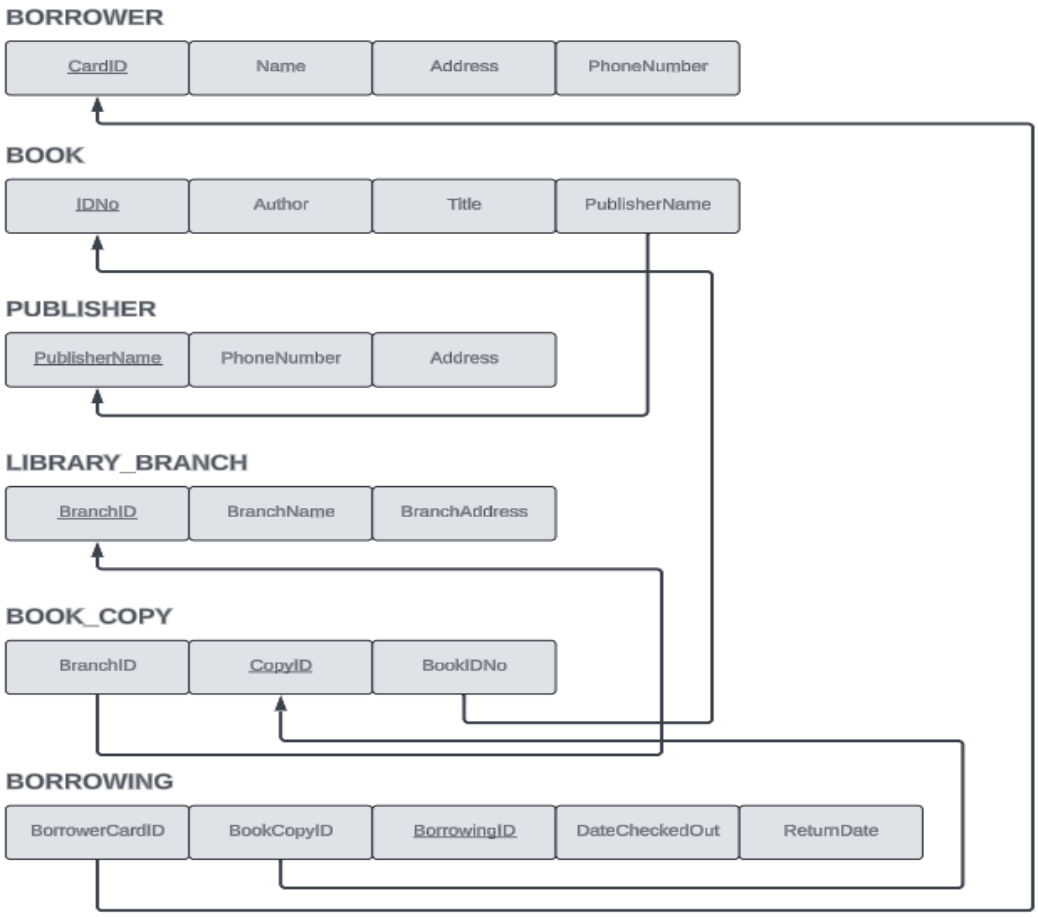
- PUBLISHED_FROM (BOOK to PUBLISHER):
  - Type: Many-to-One. Each book is published by one publisher, while a publisher can publish many books.
  - Total participation on BOOK side, as every book must have a publisher.

- HAS_COPIES (BOOK_COPY to LIBRARY_BRANCH):
  - Type: Many-to-One. Each book copy belongs to a specific branch.
  - Total participation on BOOK_COPY side, as every copy must be located in a branch.

- BORROWED_FROM (BOOK_COPY to BORROWING):
  - Type: One-to-Many. Each BOOK_COPY can be borrowed multiple times, tracking each borrowing instance.
  - Participation: Partial on BOOK_COPY side (not all copies are borrowed); Total on BORROWING side (each borrowing must link to a copy).

- MAKES (BORROWER to BORROWING):
  - Type: One-to-Many. A borrower can make multiple borrowing transactions.
  - Partial participation on BORROWER side, as not every borrower needs to borrow books.

RELATIONAL DATABASE SCHEMA



EXPLANATION OF DESIGN CHOICES:

Primary keys

- CardID(Borrower): To identify each individual separately.
- IDNo(Book): Each book will have a unique ID to separate books with the same titles.
- PublisherName(Publisher): Each publisher has a unique name.
- BranchID(Library_branch): To identify branches of the same name or address.
- CopyID(Book_copy): Book copies have their own ID separate to BookID for other branches.
- BorrowingID(Borrowing): A separate ID to check a specific borrowing of a book.

Foreign Keys

- PublisherName(Book): A single publisher can have many books.
- BranchID(Book_copy): Branches can hold many books.
- BookIDNo(Book_copy): There's one copy ID for each branch, so the same ID can show up again at a different branch.
- BookCopyID(Borrowing): Can identify what branch the borrowing took place.
- BorrowerCardID(Borrowing): To identify who's borrowing a book. They may borrow multiple.

CONTRIBUTION LIST

Shaheen Nijamudheen - ER diagram
James Paul Nguyen - Relational database schema
Ali Shirazi-Nejad - ER diagram, missing and incomplete requirements

HONOR CODE

We pledge, on our honor, to uphold UT Arlington's tradition of academic integrity, a tradition that values hard work and honest effort in the pursuit of academic excellence.

We promise that we will submit only work that we personally created or that we contribute to group collaborations, and we will appropriately reference any work from other sources. We will follow the highest standards of integrity and uphold the spirit of the Honor Code.