Library management database design and application

Jiahao Liang

D'Amore-McKim School of Business, Northeastern University, Boston, Massachusetts, 02120, United State

liang.jiaha@northeastern.edu

Abstract. Under the data-centric business environment, efficient database design is pivotal for efficient data management across various sectors. For libraries and bookstores, a well-tailored database is essential to categorize, index, and swiftly retrieve vast amounts of data, ensuring efficient operations. MySQL is an open-source database management system which facilitates efficient data manipulation. Based on MySQL, this essay delves into the realm of library database design and application, addressing the need for streamlined systems. This research addresses the need for adaptation in the evolving technology landscape and offers practical insights for librarians and researchers aiming to optimize information management in modern libraries. Employing the basic steps for database design, we explore one methodology for designing a library management database based on the design of database requirement, modeling, backend implementation, efficiency optimization and testing using MySQL. Through this approach, the study provides a potential solution that meets the basic requirements of libraries and enhances information management practices.

**Keywords:** Library Management, Database Design, Data Integrity.

1. Introduction

To maintain the competitive advantages under the dynamic change over the external technology environment, many businesses start to focus on the business digital transformation. The related technology such as cloud, mobile, big data analysis changed how organizations operate fundamentally and helped companies to maintain their competitive advantages [1]. For library or bookstore industry, a well-designed database that tailored to the unique needs of librarians, researchers, and patrons provides a structured framework to categorize, index, and retrieve the vast amount of data swiftly and accurately. Efficiently managing data within a library environment is paramount to its effective functioning. MySQL is an open-sourced database management system that enable insert, update, delete, query data efficiently [2]. To handle with the vast amount of data, create a relational database using MySQL is one of the most common solutions. Based on MySQL, this essay embarks on a journey into the realm of library database design and its practical applications, emphasizing the need for streamlined systems that enhance information management. By employing fundamental principles of database design, we delve into a specific methodology for crafting a comprehensive library management database. In light of the rapid advancements in digital technologies, the motivation behind this research stems from the pressing need to adapt and excel in an ever-changing technological landscape. Based on the methodology outlined in the reference book, this research navigates through the stages of database requirement

identification, conceptual modeling, backend implementation, optimization and offers some testing samples using MySQL[3]. By executing this comprehensive approach, the research provides a possible solution that addresses the motivational concerns, ensuring that the database design and implementation align with the basic library requirement for enhanced information management.

## 2. Database requirement collection

The first important step for the whole process of designing a database is collecting the requirement for the creation of metadata. This requirement can be based on the demand of our clients or the current data. At this step, determine the entities to hold the data enable represent the reality accurately. Figure 1 and figure 2 conceptualized some main business process based on Boston Public Library using flow chart [4]. Figure 1 illustrates a complete process of user completing the borrowing procedure or library card information management at the library. The right side of the flowchart shows the possible entities involved or information required on each process. A customer begins with the registration or login at a library using personal information. Subsequently, they have the option to either search for specific library collections using provided information or perform various library card management tasks, such as profile editing, and item returns within the library premises. If they want to borrow anything from the library, the system will check whether item status is available at the specific library branch. Finally, the customer finishes the reservation after any confirmation (Figure 1).
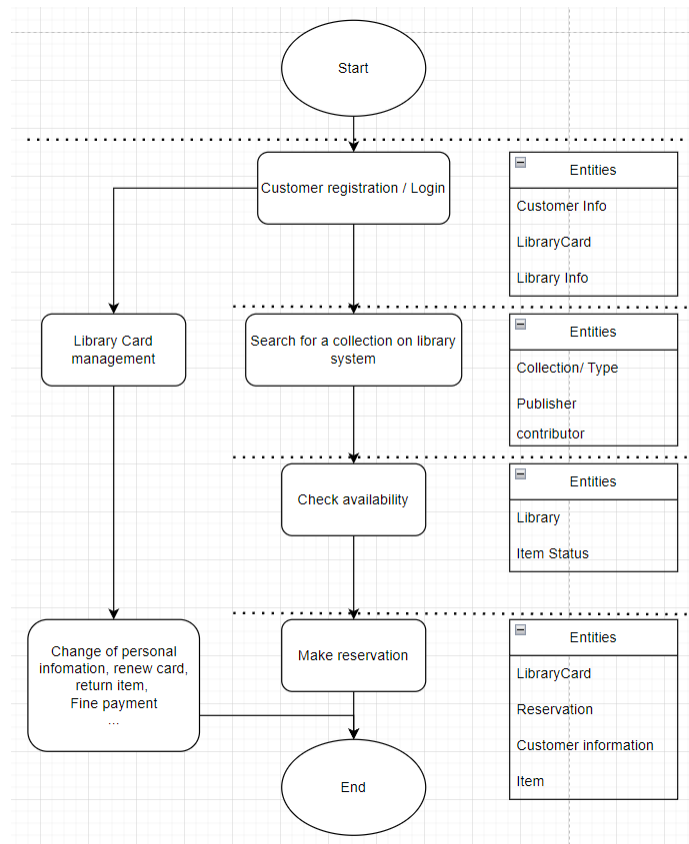
Figure 1. Reservation and information management procedure.

Figure 2 outlines an alternative pathway for customers seeking to engage with various services within the library. In a manner akin to the previous process, following customer registration or login, they proceed to choose a specific library branch. At this point, the webpage or application presents a range of available services, including participation in specialized library events and scheduling appointments

with staff members. They system then reply with the services details like schedules or help making an appointment.
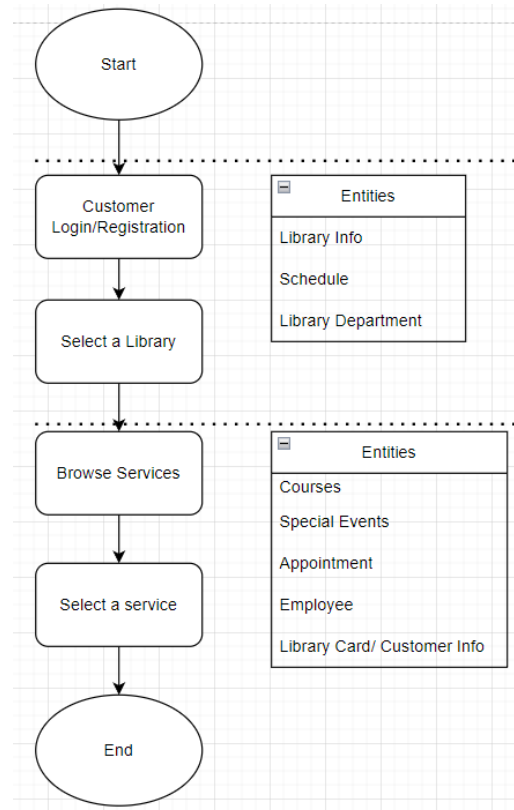
Figure 2. Process customer engages with library services.

Under those examples, the common entities include library information, library department, employee, customer, type of collections, collections, items, publishers, contributors, library events, library appointments, library courses, item reservation, and the datetime schedules for events. These data entities need to be flexibly adjusted according to practical requirements.

2.1. Database design for customers
The following section of the database requirement is to specify the relationships requirements in detail based on the different user groups. To facilitate the conceptual design process in the later part, the specific relationships and cardinalities between data entities need to be described as accurately as possible. Customers would be one of the main user groups using this database. Based on the flowchart above, three main roles that required to be supported include customer information management, library card management, reservation, and customer services.

The database should store essential customer details, including their name, Social Security Number (SSN), email, and phone number. A unique customer ID (CustomerID) will serve as the primary key. In term of library card issuance, the system must facilitate new customer registration and able to record the issuance of distinct library cards. Each library card is associated with a unique card ID (CardID) and is linked to a specific customer (CustomerID). A customer is allowed to possess only one library card at a time.

To achieve the role of reservation management, customers should be allowed to make reservations for library items like books, and CDs. The database needs to retain reservation particulars, comprising the reservation date, status, linked library card (CardID), and the item reserved (ItemID). The system should provide access to customers' reservation history and item details, including their status. Upon

item reservation, the database support frequent update toward item's status and record the reservation history. Customers can hold a maximum of 0 to 5 items concurrently.

For the service section, customers should have the capability to schedule appointments with library staff for assistance. The database should record appointment details, encompassing the time schedule, appointment type (AppointmentType), unique ID (AppointmentID), and status. A customer can have multiple appointments, and each appointment is distinct and reserved solely for that customer.

2.2. Database design for employees

The employee or librarian is the other main users of this database, whose information would be used in employee certification and customer service support. The database must store employee information, including their name, Social Security Number (SSN), gender, email, phone number, and employee status. Each employee should have a unique employee ID (EmployeeID) as the primary key. Employees should be linked with specific departments within a specific library branch. Each employee is exclusively affiliated with one department, ensuring a singular department assignment at any given time.

For appointment, the database should store information about the appointments they served. One appointment can be served by multiple employees depends on the appointment details. The employee can work for one department at the same time. An appointment can involve collaboration among multiple employees depending on the appointment's nature.

2.3. Database design for libraries

The database requirement design for the library component encompasses the intricate relationships and functionalities pertaining to library branches, departments, events, courses, collections, and associated entities. The database will encompass essential particulars for all connected library branches. This information comprises the library name (LibraryName), contact details, email address, physical address, and specific operating hours for each day. A customer can exclusively register at any one library branch, while enjoying service access across all Boston Public Library branches. Each library can cater to a diverse range of customers, adhering to a one-to-many relationship. Libraries house various departments, each with a unique opening schedule for the public. Each department have a unique id can belong to only one library.

Each library can host special events that free for publics to join. A library can organize multiple events concurrently, with uniqueness guaranteed across different libraries. The database will store the ID (EventID), name, description, and status of every events. Each course has one to many unique schedules. Library also provides many general courses to the publics. Similarly, the course name, schedule, status would be recorded. Each course has one to many unique schedules.

The database should also be able to manage its collections. The collection is virtual item in the library. Each collection possesses a unique ID, name, addition date, description, and ISBN. Collections are categorized into types like music, fiction, etc., identified by a TypeID. The publisher information will include the publisherID, publisher name, country, and the city. Collections can be published by multiple publishers, signifying various versions of an item. Each publisher maintains links to multiple collections, contributing to diverse library offerings. For every collection, the database also records the contributors. One collection can have one to many contributors. Each contributors' name, gender, age, education, major, job would also be recorded to facilitate user's searching.

Item entity represent the real hard copies of a collection that stored in different library. Every copy is unique with a status used to determine whether it can be reserved. Every collection can have multiple unique copies in libraries and each library can have multiple copies for the same collection.

3. Requirement Visualization

The conceptual design phase helps shaping the foundation of a robust and effective system. In the step of conceptual design, the objective is to map the relationships, attributes, and requirements into a standard database structure. This can be achieved by creating an Entity-Relationship (ER) diagram which serves as a visual representation that encapsulates the essence of the database's structure and the

connections between entities (Figure 3). The format and method referenced the book Database system [5].
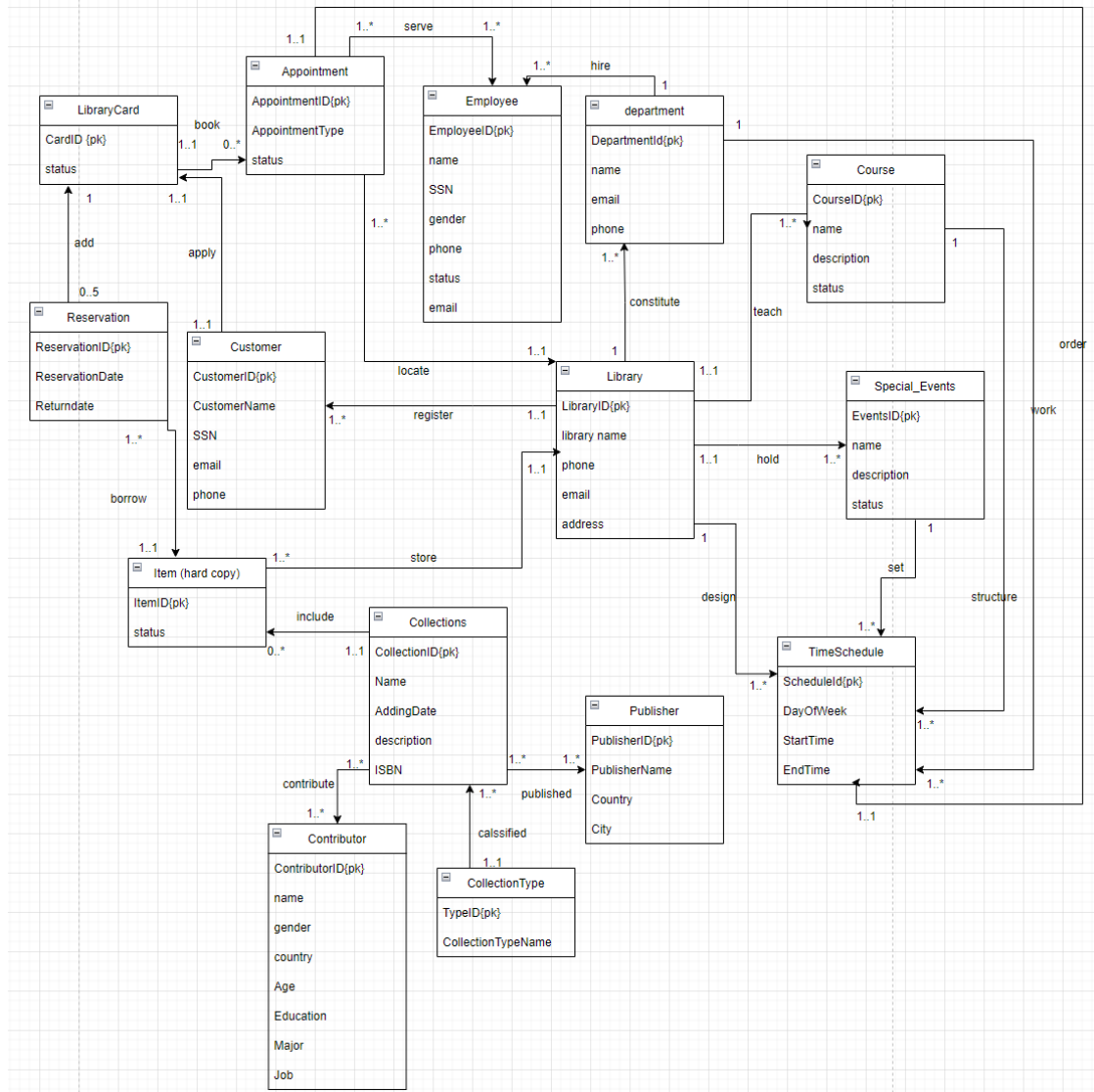
Figure 3. ER diagram for relational database.

To ensure the data integrity, the design here can directly adhere to the third normal form. This stage involves the consideration of two primary types of dependencies: partial functional dependency and transitive functional dependency.

4. Normalization

Second normal form (2NF) focusing primarily on addressing partial dependencies within the data model. In the context of our conceptual design, a key consideration lies in the avoidance of utilizing composite keys as the primary key for each entity. This step is crucial in reducing redundancy and ensuring that each primary key uniquely identifies the respective entity. Such dependencies can lead to inefficiencies, data anomalies, and increased complexity [6].

Based on the second normal form, to further fortify the database structure, we then examining transitive dependencies—a key step in reaching the third normal form (3NF). This involves ensuring that no non-key attribute depends on another non-key attribute which also help ensure the data integrate.

If a non-key attribute depends on another non-key attribute, we can simply separate the attributes into two separate tables.

5. Database Modeling

Before transitioning from the conceptual model to a practical implementation, it is necessary to convert the Entity-Relationship diagram into a relational model (Figure 4).
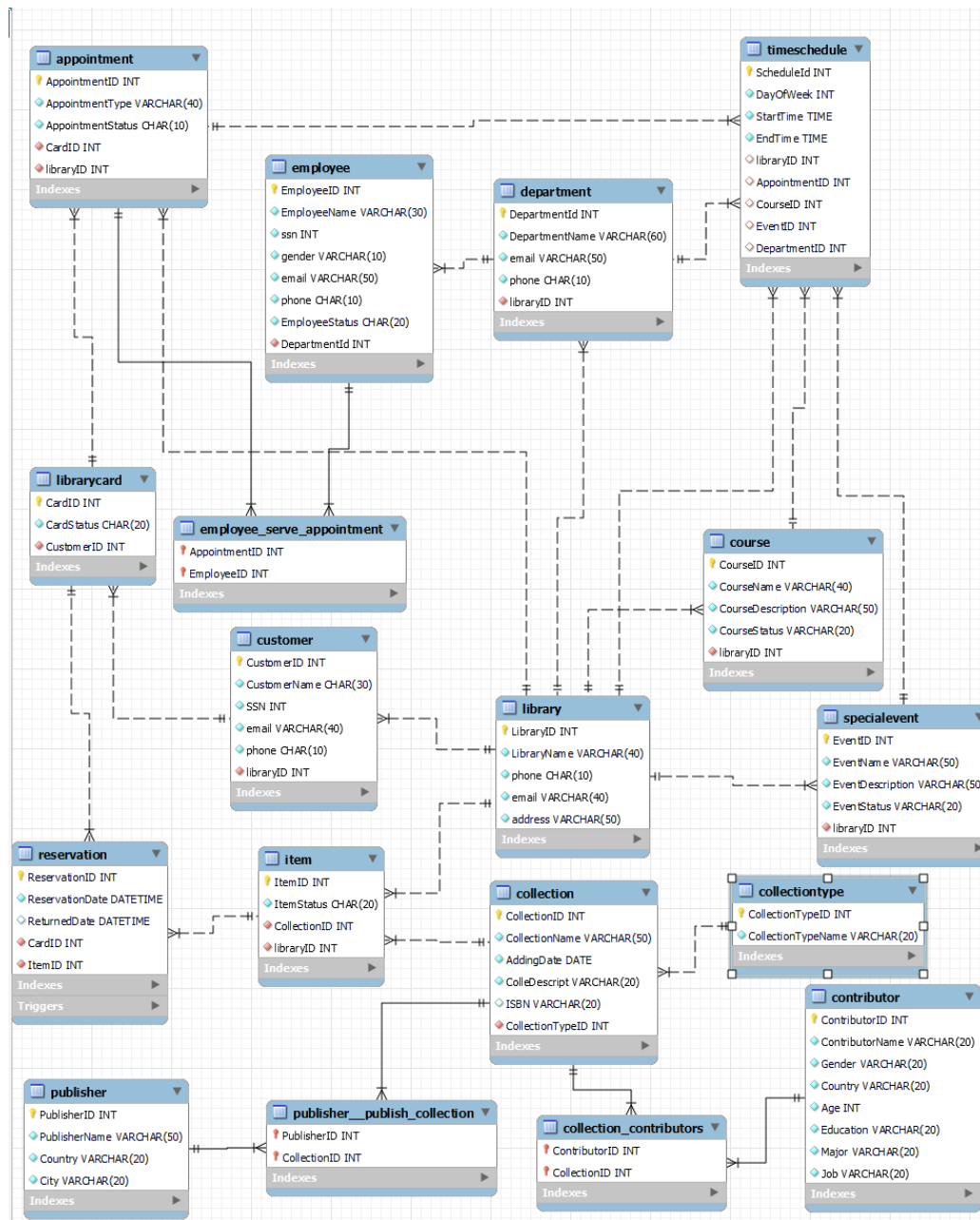

Figure 4. Relational Model.

The guiding principle of this transformation is based on the fundamental rule of mapping the relationships. In the table representing the "many" sides of the relationship, add a foreign key column that references the primary key of the table representing the "one" side.

In scenarios where many-to-many relationships are involved, we create a third table as the junction table represent the relationship between two entities. This table serves as an intermediary that captures the intricate associations between two entities. The junction table include a composite primary key of the two related entities and foreign key columns reference the primary keys of the two related tables.

6. Database optimization and implementation

In this research, all the database implementation is based on the MySQL database. With the relation model above, we can implement the database easily. Here's some optimization suggestions that can be considered. For primary key optimization, all primary keys used in the database should be an auto increment value for efficiency considerations. This is because MySQL add an index for every primary automatically with a B+ tree as the underlying structure [7]. In this case, any new insertion is a sequential insertion and will not change the structure of the B+ tree or split of the leaf node [8]. That can maintain the efficiency during insertion while the amount of the data growing.

For security concern, the one-to-one relationship customer and library card is separated instead of combined. The customer password would be stored in the library card entity so that we restrict customer's chance directly access to the reservation record and the password. On the other hand, the customer is free to change their address and other info in the customer table. This also helps avoid some potential privacy issues. It helps minimize data exposure by only showing necessary information to employees during interactions. For instance, if a customer wants to borrow a book, the employee doesn't need to see the customer's full personal details. On the other hand, this also helps fulfill some legal obligations such as European Union General Data Protection Regulation (EU GDPR) [9,10], when different levels level of protection is required between user personal information and user account.

Data integrity can be protected automatically by refining the triggers and constraints. An example will be presented using a trigger to automatically ensure data consistency. Regarding the reservation table, the reservation date and customer information are updated if an item is reserved from the library. Upon item return, the corresponding record is updated with the return date. When a customer aims to reserve an item online at a specific library, prevention measures are in place to limit excessive reservations or reserving transferred items. Following successful reservation, adjustment of the specific item's status becomes the responsibility of the librarian. Implementing a before-insert trigger on the reservation table involves assessing the item's status before executing the reservation action. A sample flowchart for this trigger is presented in Figure 5.
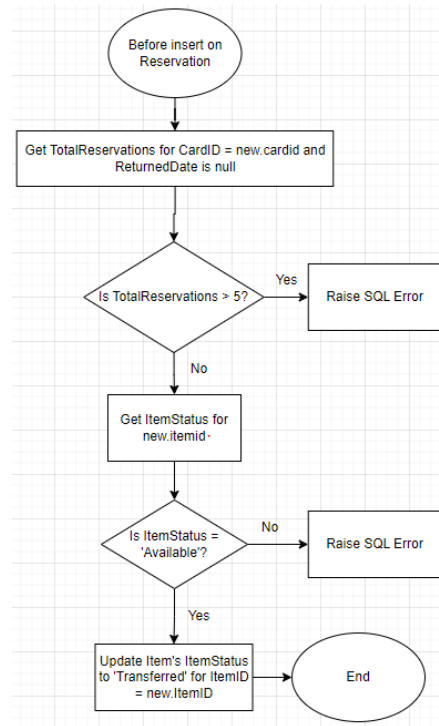
Figure 5. Flow chart for one sample trigger.

On the other hand, data integrity can also be ensured during front end application development. For example, data format consistency can be sustained by implementing a client-side validation to ensure that users enter data in the expected format. Dropdown selection, sanitization and escaping are some other commonly used methods to regulate the user input. Since this project forced on the backend database development, no further examples would be discussed in this context.

7. Application and testing

To ascertain the viability of the database design, a sample database and its associated triggers have been constructed in MySQL, adhering closely to the relational model. This following part provides some sample of data query, updating and insertion to test for the data integrity. Sample 1 shows when customer send a query for the status of a collection online: given a collection name, return the status and information of all such item under the collection. Based on the testing data (Appendix A), we write a query to select collection name, publisher name, isbn, contributors, item status, and library name from the joined table, we can get the information in Figure 6.

	CollectionName	PublisherName	ISBN	Contributors	ItemStatus	LibraryHoldingItem
▶	Harry Potter and the Sorcerer's Stone	Random House	978-1234567890	John Doe,Jane Smith	Available	Boston public Library
	Harry Potter and the Sorcerer's Stone	Random House	978-1234567890	John Doe,Jane Smith	Available	Branch Library 2
	Harry Potter and the Sorcerer's Stone	Random House	978-1234567890	John Doe,Jane Smith	Transferred	Boston public Library
	Harry Potter and the Sorcerer's Stone	Random House	978-1234567890	John Doe,Jane Smith	Transferred	Branch Library 1

Figure 6. The sample result from the query.

One of the trigger used in the database is tested in sample 2. The item table shows item 1, 3, 5 have been transferred (Figure 7).

	ItemID	ItemStatus	CollectionID	libraryID
▶	1	Transferred	1	1
	2	Available	2	1
	3	Transferred	3	1
	4	Available	4	1
	5	Transferred	1	2
	6	Available	2	2
	7	Available	3	2
	8	Available	4	2
	9	Available	1	3
	10	Available	2	3
	11	Available	1	1
✱	NULL	NULL	NULL	NULL

Figure 7. Item table before change in reservation table.

Assume one customer with card ID 1 want to reserve the item with ItemID 2 from the library. The system adds a new entry with reservation ID 4 into reservation table (Figure 8). Figure 9 shows the change of item status in the item table. The item status of item 2 is changed automatically into "transferred" as the reservation confirmed in the trigger.

	ReservationID	ReservationDate	ReturnedDate	CardID	ItemID
▶	1	2023-07-20 10:00:00	NULL	1	1
	2	2023-07-25 14:30:00	2023-07-25 14:30:00	1	3
	3	2023-07-05 09:45:00	NULL	4	5
	4	2023-08-01 10:00:00	NULL	1	2
✱	NULL	NULL	NULL	NULL	NULL

Figure 8. Add new reservation to table.

	ItemID	ItemStatus	CollectionID	libraryID
▶	1	Transferred	1	1
	2	Transferred	2	1
	3	Transferred	3	1
	4	Available	4	1
	5	Transferred	1	2
	6	Available	2	2
	7	Available	3	2
	8	Available	4	2
	9	Available	1	3
	10	Available	2	3
	11	Available	1	1
✱	NULL	NULL	NULL	NULL

item 38 ×

Output

Action Output ▼

#	Time	Action
✅ 140	01:44:37	select * from item LIMIT 0, 50
✅ 141	01:45:42	INSERT INTO Reservation (ReservationDate, ReturnedDate, CardID, ItemID) VALUES ('2023-08-01 10:00:00', NULL, 1, 2)
✅ 142	01:48:56	select * from item LIMIT 0, 50

Figure 9. Item table after change in reservation table.

Sample 3 shows a test for some self-defined reservation constraints: a user can reserve 5 items at maximum. In figure 8, customer with card ID 1 have two unreturned items. Then three more items have been reserved successfully by the same customers (Figure 10).

Figure 10. Reservation table after three new reservations from same customer.

If a librarian tried to approve the reservation of the sixth book, that action would be rejected by an error message "Error Code: 1644. reservation over maximum item: 5" (Figure 11).

Figure 11. New reservation rejected by the trigger.

8. Conclusion

This essay provides a sample of how to design a database for library based on the traditional database design steps. The core steps involve collecting the basic requirement based on the clients and main users, followed by visualizing and normalizing the database based on the requirements. After visualizing the database structure, the subsequent step includes completing the logical design of the database using a relational model and incorporating any essential optimizations. This design is then implemented and subjected to thorough testing. The sample provided is only a simple sample for the library management systems. In practical situations, the database design needs to be flexible based on different scenarios and requirements. Further research is needed on deep-level optimization, database integration with frontend applications, and other related issues.

References

[1] Binildas, C. A. (2019). Practical microservices architectural patterns: Event-based Java microservices with Spring Boot and Spring Cloud. *Apress* **29**.

[2] Schwertner, K. (2017). Digital transformation of business. Trakia Journal of Sciences, **15(1)**, 388-393.

[3] Jukic N. Vrbsky S. & Nestorov S. (2017). Database systems: introduction to databases and data warehouses. *Prospect Press.*

[4] Boston Public Library. (n.d.). https://www.bpl.org/

[5] Connolly, T. M., & Begg, C. E. (2015). Database systems: A practical approach to design, implementation and management. *Pearson* **34**.

[6]   Upadhyay, M. (2023,). Second normal form (2NF). *GeeksforGeeks*. https://www. geeksforgeeks.org/second-normal-form-2nf/

[7]   MySQL 8.0 Reference Manual: 8.3.2 primary key optimization. MySQL. (n.d.). https://dev.mysql.com/doc/refman/8.0/en/primary-key-optimization.html

[8]   P. Kieseberg, S. Schrittwieser, P. Frühwirt and E. Weippl, (2019) Analysis of the Internals of MySQL/InnoDB B+ Tree Index Navigation from a Forensic Perspective, *International Conference on Software Security and Assurance*, 46-51,

[9]   Prateek, T. L. (2023). Normalization in SQL: 1NF, 2NF, 3NF and BCNF in database. Edureka. https://www.edureka.co/blog/normalization-in-sql/

[10]   Axel., V., Paul. Von Dem. (2018) EU General Data Protection Regulation (GDPR): A practical guide. Springer international pu,1-10.

Appendix A:

Library:

▶	1	Boston public Library	1234567890	mainlibrary@example.com	123 Main Street
	2	Branch Library 1	9876543210	branch1@example.com	456 Branch Avenue
	3	Branch Library 2	5555555555	branch2@example.com	789 Branch Road
	4	Downtown Library	1112223333	downtown@example.com	1 Downtown Blvd
	5	Northside Library	4445556666	northside@example.com	50 North Ave
	6	West End Library	7778889999	westend@example.com	100 West End St
	7	Eastside Library	8889990000	eastside@example.com	75 Eastside Rd
	8	Southtown Library	2223334444	southtown@example.com	200 Southtown Ave
	9	Hilltop Library	6667778888	hilltop@example.com	15 Hilltop Way
	10	Riverside Library	9990001111	riverside@example.com	300 Riverside Dr
	NULL	NULL	NULL	NULL	NULL

Collection:

	CollectionID	CollectionName	AddingDate	ColleDescript	ISBN	CollectionTypeID
▶	1	Harry Potter and the Sorcerer's Stone	2023-07-01	Fantasy	978-1234567890	1
	2	Introduction to Python	2023-07-15	Programming	978-0987654321	1
	3	National Geographic	2023-07-10	Nature	NULL	3
	4	The Great Gatsby	2023-07-05	Classic Fiction	978-9876543210	1
	5	The Lord of the Rings	2023-07-12	Fantasy	978-5678912345	1
	6	Travel Photography	2023-07-20	Photography	NULL	2
	7	The New York Times	2023-07-08	News	NULL	3
	8	City Map	2023-07-03	Navigation	NULL	8
	9	Vinyl Classics	2023-07-17	Music	NULL	9
	10	Superhero Adventures	2023-07-25	Comic	NULL	10
	11	Sample book 1	2023-07-01	a	978-1234567891	1
	12	Sample book 2	2023-07-01	b	978-1234567892	1
	13	Sample book 3	2023-07-01	d	978-1234567893	1
	14	Sample book 4	2023-07-01	a	978-1234567894	1
	15	Sample book 5	2023-07-01	a	978-1234567895	1

Publisher:

	PublisherID	PublisherName	Country	City
▶	1	Random House	USA	New York
	2	HarperCollins	USA	San Francisco
	3	Penguin Books	USA	Boston
	4	Oxford University Press	UK	London
	5	Allen & Unwin	Australia	Sydney
	6	Simon & Schuster	USA	Chicago
	7	Macmillan Publishers	UK	Oxford
	8	Hachette Livre	France	Paris
	9	Wiley	USA	Hoboken
	10	Bloomsbury Publishing	UK	London
	NULL	NULL	NULL	NULL

Contributors

	ContributorID	ContributorName	Gender	Country	Age	Education	Major	Job
▶	1	John Doe	Male	USA	35	Master's	English Literature	Author
	2	Jane Smith	Female	USA	42	Ph.D.	History	Historian
	3	Michael Johnson	Male	USA	28	Bachelor's	Computer Science	Software Engineer
	4	Emily Williams	Female	UK	29	Bachelor's	Biology	Biologist
	5	Robert Lee	Male	Canada	40	Ph.D.	Physics	Physicist
	6	Laura Brown	Female	Australia	38	Master's	Psychology	Psychologist
	7	David Miller	Male	USA	25	Bachelor's	Chemistry	Chemist
	8	Emma Wilson	Female	UK	33	Ph.D.	Philosophy	Philosopher
	9	Christopher Clark	Male	Canada	37	Master's	Sociology	Sociologist
	10	Sophia Turner	Female	Australia	22	Bachelor's	Fine Arts	Artist
	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Publisher_publish_collection;

PublisherID	CollectionID
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
10	10
10	11
10	12
9	13
9	14
2	15
NULL	NULL

Item:

ItemID	ItemStatus	CollectionID	libraryID
1	Transferred	1	1
2	Available	2	1
3	Transferred	3	1
4	Available	4	1
5	Transferred	1	2
6	Available	2	2
7	Available	3	2
8	Available	4	2
9	Available	1	3
10	Available	2	3
11	Available	1	1
NULL	NULL	NULL	NULL