EECS 447 Project

Logical Relational Model

Version 1.4

Revision History

Date	Version	Description	Author
2/22/2025	1.1	All team members completed their own individual parts of the document before the meeting. They worked on Parts 1 and 2, and then the group came together to work on the more collaborative sections.	Sophia Jacob, Kusuma Murthy, Anna Lin, Nimra Syed, Ella Nguyen, Nikka Vuong
3/22/2025	1.2	All team members worked on Section 1, 2.1, 2,2, and 2.3.	Sophia Jacob, Kusuma Murthy, Anna Lin, Nimra Syed, Ella Nguyen, Nikka Vuong
3/26/2025	1.3	All team members fixed Section 2.1, 2.2, 2.3, 2.5, and 3. We refined each part and proofread.	Sophia Jacob, Kusuma Murthy, Anna Lin, Nimra Syed, Nikka Vuong
3/28/2025	1.4	All team members refined most of the sections due to normalization considerations. We also refined the ER Diagram of Part 03 which is reflected in the changes made in Part 04.	Sophia Jacob, Kusuma Murthy, Anna Lin, Nimra Syed, Ella Nguyen, Nikka Vuong

Table of Contents

1 Introduction

1.1 Project Overview

As a Software as a Service (SaaS) company, ASKNrEceive (ASKNE) hopes to help streamline database creation and management. The purpose of this project is to provide local libraries with a Library Management System (LMS) that ensures efficiency for both types of end-users: library staff and library customers. This abstracted database will facilitate better documentation of the books, magazines, and other available items for ease of management and organization.

Through this project, ASKNE will develop a fully functional and scalable database software for libraries to handle, maintain, and analyze the popularity of their collections through detailed reports. Using a streamlined approach, the goal is to design a robust relational database with a well-defined conceptual schema and physical implementation.

**** Note: Refer to *01 - Project Plan/Vision* for any additional information on the project overview and definition of the project.**

1.2 Scope

ASKNE will use MySQL and other database management tools to deliver an extensive database backed in software to provide better insights and organization to libraries regarding books, magazines, and other content they house. This project entails creating a full-scale LMS, powered by a relational database. ASKNE aims to create a reporting style that helps the library determine popular books, members' favorite genres, and more quantifiable statistics to help maintain a growing library. This project will be responsible for managing the library's database operations such as adding new books as parts of entries, cleaning the database by removing and editing the borrow and return status of loanable items, and finally creating comprehensive reports for the Library admins and staff. Other features include tracking books and members based on a variety of attributes like author names, item IDs, publication dates, and more. Users can view the database for available books, while library staff can edit and maintain the database. As for the technical aspect of this project, ASKNE will be utilizing Structured Query Language (SQL) as the main tool used for searching the database and will create various tables that will be structured in a Relational database format.

**** Note: Refer to *01 - Project Plan/Vision* for any additional information on the scope and definition of the project.**

1.3 Introduction to Relational Schema Mapping

Relational Schema Mapping is a crucial step in designing the relational database. It ensures the integrity of data by providing consistent and accurate queries. The following section outlines ASKNE's processor mapping the Entity-Relationship (ER) model (which can be viewed in 03-Conceptual Modeling) to a relational schema. This is done by identifying relations/tables, defining attributes and domain types, assigning primary and foreign keys, and lastly, establishing functional dependencies.

When identifying the relations for the project ASKNE utilized the ER model created previously to use as a base structure for identifying the essential relationships. Next the team defined all the attributes for each of the relations and also determined the domain type for each attribute. After this step, the primary and foreign keys for each relation were identified to connect all of the separate relations, creating a cohesive database system. Lastly, functional dependencies of the database were identified and stated to represent the constraints of the real world to ensure data integrity.

**** Note: Refer to *03 - Conceptual Modeling* for any additional information on the Entity-Relation (ER) diagram and definition of the project.**

1.4 Glossary

**** Note: Our team has a dedicated Document for Abbreviations and Glossary. Refer to *07 - Glossary*.**

For the purpose of this document, it will also be put in this subsection.

- ASKNE - ASKNrEceive
- SaaS - Software as a Service
- LMS - Library Management System
- SQL - Structured Query Language
- TAs - Teaching Assistant
- ISBN - International Standard Book Number
- ISSN - International Standard Serial Number

2 Relational Schema Mapping

2.1 Identify Relations

Note: Underlined attributes are the primary keys.

Entity	**Attributes**
Copy	copy_id
	status

Entity	**Attributes**
Publisher	publisher_id
	name

Entity	**Attributes**
Author	author_id
	name

Entity	**Attributes**
Genre	genre_id
	genre_name

Entity	**Attributes**
Transaction	transaction_id
	checked_out_date
	due_date
	return_date

Entity	Attributes
Fine	fine_id
	fine_date
	fine_amount

Entity	Attributes
Library_Item (Generalization)	item_id
	current_inventory
	checked_out_status
	overdue_status

Entity	Attributes
Book (Specialization of Library_Item)	item_id (derived from generalization)
	ISBN (International Standard Book Number)
	title
	author
	publication_year
	genre
	availability_status
	book_rating

Entity	Attributes
Digital_Media_Item (Specialization of Library_Item)	item_id (derived from generalization)
	ISBN (International Standard Book Number)
	title
	author

	publication_year
	genre
	availability_status

Entity	**Attributes**
Magazines (Specialization of Library_Item)	item_id (derived from generalization)
	ISSN (International Standard Serial Number)
	author
	title
	issue_number
	publication_date
	genre
	availability_status

Entity	**Attributes**
Library_Report	report_id
	report_date
	total_amount_owed_per_day
	total_amount_paid_per_day
	number_of_checkouts

Entity	**Attributes**
Member_Account	account_id
	incurred_fees
	total_amount_paid
	overdue_balance

	recommended_items

Entity	**Attributes**
Staff	staff_id
	role

Entity	**Attributes**
Library_Members	member_id
	limit
	fee_type
	type_id
	name
	contact_information
	account_status

Entity	**Attributes**
Waitlist	waitlist_id
	request_date
	waitlist_status
	fulfilled_date

Entity	**Attributes**
Regular (Specialization of Library_Members)	member_id (derived from generalization)
	available_limit

Entity	**Attributes**

Student (Specialization of Library_Members)	member_id (derived from generalization)
	available_limit

Entity	**Attributes**
Senior_Citizen (Specialization of Library_Members)	member_id (derived from generalization)
	available_limit

Entity	**Attributes**
Recommendation	recommendation_id

Relationship	**Entity**	**Attribute**
Added	Waitlist → Copy	copy_id
		waitlist_id

Relationship	**Entity**	**Attribute**
Hold	Library_Member → Copy	{member_id, copy_id}

Relationship	**Entity**	**Attribute**
Pay	Library_Member → Fine	fine_id
		member_id

		amount_paid
		paid_date

Relationship	**Entity**	**Attribute**
Incur	Fine → Transaction	transaction_id
		fine_id

Relationship	**Entity**	**Attribute**
Process	Transaction → Staff	transaction_id
		staff_id

Relationship	**Entity**	**Attribute**
Loan	Transaction → Copy	copy_id
		transaction_id

Relationship	**Entity**	**Attribute**
Make	Library_Member → Transaction	transaction_id
		member_id

Relationship	**Entity**	**Attribute**
Receives	Recommendation → Member_Account	{recommendation_id, account_id}

Relationship	**Entity**	**Attribute**
Consists_Of	Recommendation → Library_Item	recommendation_id
		item_id

Relationship	**Entity**	**Attribute**
View	Library_Member → Member_Account	member_id
		account_id
		timestamp

Relationship	**Entity**	**Attribute**
Generates	Staff → Library_Report	report_id
		staff_id

Relationship	**Entity**	**Attribute**
Rating	Library_Member → Book	{member_id, item_id}
		stars_given

Relationship	**Entity**	**Attribute**
Oversees	Staff → Library_Member	member_id
		staff_id

Relationship	**Entity**	**Attribute**
Update	Staff → Copy	copy_id
		staff_id

Relationship	**Entity**	**Attribute**
Originate	Copy → Library_Item	copy_id
		item_id

Relationship	**Entity**	**Attribute**
Publishes	Publisher → Library_Item	item_id
		publisher_id

Relationship	**Entity**	**Attribute**
Writes	Author → Library_Item	item_id
		author_id

Relationship	**Entity**	**Attribute**
Categorized	Library_Item → Genre	item_id
		genre_id

Relationship	**Entity**	**Attribute**

Gives	Library_Member → Recommendation	recommendation_id
		member_id

2.2 Define Attributes and Domains

Refer to Section *4 Schema Documentation with a Data Dictionary*.

2.3 Determine Primary Keys

Relationships	**Entities**	**Primary Key**
Added	Waitlist → Copy	copy_id
Hold	Library_Member → Copy	{member_id, copy_id}
Pay	Library_Member → Fine	fine_id
Incur	Fine → Transaction	transaction_id
Process	Transaction → Staff	transaction_id
Loan	Transaction → Copy	copy_id
Make	Library_Member → Transaction	transaction_id
Receives	Recommendation → Member Account	{recommendation_id, account_id}
Consists of	Recommendation → Library_Item	recommendation_id
Views	Library_Member → Member Account	member_id
Generates	Staff → Library Report	report_id

Rating	Library Member → Book	{member_id, item_id}
Oversees	Staff → Library Member	member_id
Update	Staff → Copy	copy_id
Originate	Copy → Library Item	copy_id
Publishes	Publisher → Library_Item	item_id
Writes	Author → Library_Item	item_id
Categorized	Library_Item → Genre	item_id
Gives	Library_Member → Recommendation	recommendation_id

2.4 Establish Foreign Keys

Relationships	**Entities**	**Foreign Key**
Added	Waitlist → Copy	**copy_id** → Copy.copy_id **waitlist_id** → Waitlist.waitlist_id
Hold	Library_Member → Copy	**member_id** → Library_Member.member_id **copy_id** → Copy.copy_id
Pay	Library_Member → Fine	**fine_id** → Fine.fine_id **member_id** → Library_Member.member_id
Incur	Fine → Transaction	**transaction_id** → Transaction.transaction_id **fine_id** → Fine.fine_id
Process	Transaction → Staff	**transaction_id** → Transaction.transaction_id **staff_id** → Staff.staff_id
Loan	Transaction → Copy	**copy_id** → Copy.copy_id **transaction_id** → Transaction.transaction_id

Make	Library_Member → Transaction	**transaction_id** → Transaction.transaction_id **member_id** → Library_Member.member_id
Receives	Recommendation → Member Account	**recommendation_id** → Recommendation.recommendation_id **account_id** → Account.account_id
Consists_Of	Recommendation → Library_Item	**recommendation_id** → Recommendation.recommendation_id **item_id** → Library_Item.item_id
Views	Library_Member → Member Account	**member_id** → Library_Member.member_id **account_id** → Account.account_id
Generates	Staff → Library Report	**report_id** → Library_Report.report_id **staff_id** → Staff.staff_id
Rating	Library Member → Book	**member_id** → Library_Member.member_id **item_id** → Library_Item.item_id
Oversees	Staff → Library Member	**member_id** → Library_Member.member_id **staff_id** → Staff.staff_id
Update	Staff → Copy	**copy_id** → Copy.copy_id **staff_id** → Staff.staff_id
Originate	Copy → Library Item	**copy_id** → Copy.copy_id **item_id** → Library_Item.item_id
Publishes	Publisher → Library_Item	**item_id** → Library_Item.item_id **publisher_id** → Publisher.publisher_id
Writes	Author → Library_Item	**item_id** → Library_Item.item_id **author_id** → Author.author_id
Categorized	Library_Item → Genre	**item_id** → Library_Item.item_id **genre_id** → Genre.genre_id
Gives	Library_Member → Recommendation	**recomenndation_id** → Recommendation.recommendation_id **member_id** → Library_Member.member_id

2.5 Establish Functional Dependencies

Relationships	**Entities**	**Functional Dependencies**
Added	Waitlist → Copy	**TBD**

Hold	Library_Member → Copy	**TBD**
Pay	Library_Member → Fine	**TBD**
Incur	Fine → Transaction	**TBD**
Process	Transaction → Staff	**TBD**
Loan	Transaction → Copy	**TBD**
Make	Library_Member → Transaction	**TBD**
Receives	Recommendation → Member Account	**TBD**
Consists_Of	Recommendation → Library_Item	**TBD**
Views	Library_Member → Member Account	**TBD**
Generates	Staff → Library Report	**TBD**
Rating	Library Member → Book	**TBD**
Oversees	Staff → Library Member	**TBD**
Update	Staff → Copy	**TBD**
Originate	Copy → Library Item	**TBD**
Publishes	Publisher → Library_Item	**TBD**
Writes	Author → Library_Item	**TBD**
Categorized	Library_Item → Genre	**TBD**
Gives	Library_Member →	**TBD**

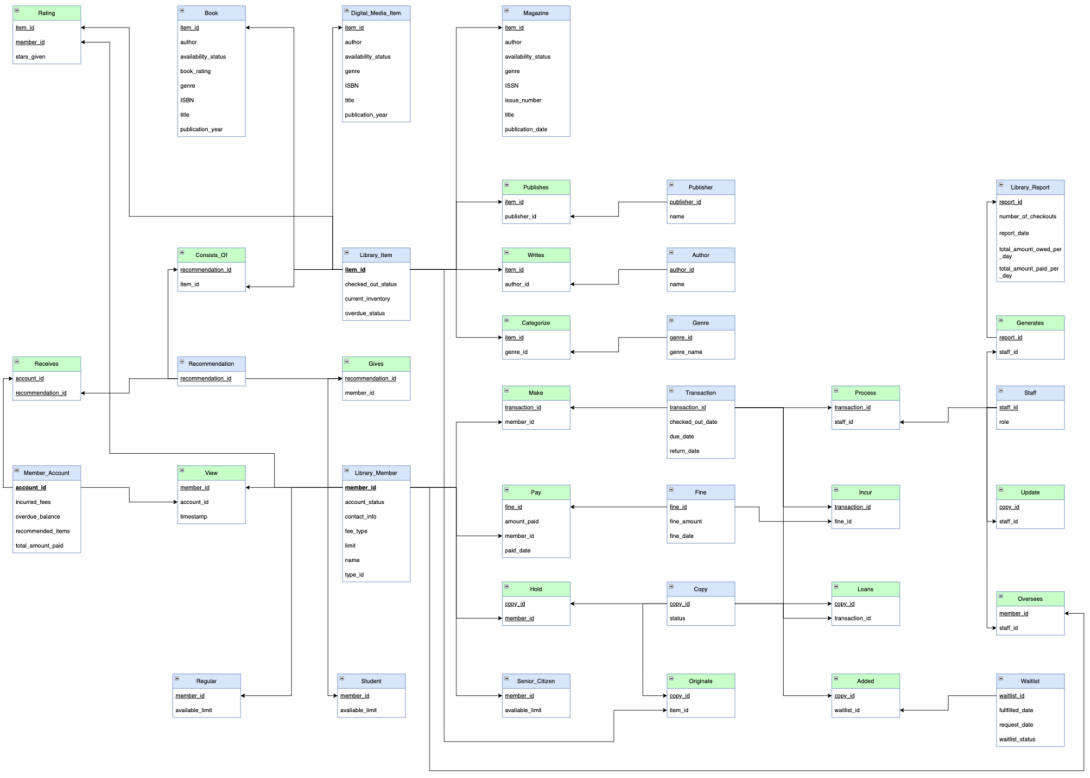
	Recommendation	

3 Relational Schema Diagram

The relational schema diagram was created from the previous Conceptual ER Model Diagram.

**** Note: To see the relational diagram better, click the link below for the PDF version.**

Relational Schema Diagram Link



4 Schema Documentation with a Data Dictionary

Attribute Name	Data Type	Description
copy_id	INT	The unique ID for the library

		item copies.
status	VARCHAR	The status indicated whether the library items are available. The types of status messages that are possible are: in_stock, checked_out.
publisher_id	INT	The is the unique id for each publisher.
Publisher.name	VARCHAR	This is the name of each publisher.
author_id	INT	This is the unique id for each author.
Author.name	VARCHAR	This is the name of each author.
genre_id	INT	This is the unique id used to distinguish each genre.
genre_name	VARCHAR	This is the name of the genre for the library items.
transaction_id	INT	This is the unique ID for each transaction of a library item.
checked_out_date	DATE_TYPE (YYYY-MM-DD)	This is the date when the members checked out the library items.
due_date	DATE_TYPE (YYYY-MM-DD)	This is the date when the members are expected to return the library items by.
return_date	DATE_TYPE (YYYY-MM-DD)	This is the date when the members are actually returned the library items. If they never returned it, this value will be NULL.
fine_id	INT	This is the unique attribute that indicates each time there is a fine charged to the person for a library item.

fine_date	DATE_TYPE (YYYY-MM-DD)	This is the date that the fine was charged to the library member.
fine_amount	DECIMAL(5,2)	This is the amount that was charged for a particular fine_id.
item_id	INT	This is the unique ID that is issued for each library member.
current_inventory	INT	This is the total current inventory of a particular current inventory.
checked_out_status	BOOLEAN	This is the status that indicates where the library item has been checked out or not. There are two types of messages that can be displayed: TRUE, FALSE.
overdue_status	BOOLEAN	This is the status that indicates that the library item has been overdue in the library database. There are two types of messages that can be displayed: TRUE, FALSE.
Book.ISBN	CHAR(16)	This is the unique ID that is used to identify each individual book in the library's database.
Book.title	VARCHAR	This is the book's title associated with a book.
Book.author	VARCHAR	This is the name of the author who wrote the book.
Book.publication_year	YEAR	This is the year that the book was published.
Book.genre	VARCHAR	This is the genre category that the book belongs to.
Book.availability_status	BOOLEAN	This is the status that indicates whether the book is in stock or not. There are two types of messages that can be displayed: TRUE, FALSE.

book_rating	DECIMAL(3,2)	This is the rating that is associated with each book.
Digital_Media.ISBN	CHAR(16)	This is the unique ID that is used to identify each individual digital media in the library's database.
Digital_Media.title	VARCHAR	This is the title associated with a digital_media.
Digital_Media.author	VARCHAR	This is the name of the author who created the digital media.
Digital_Media.publication_year	YEAR	This is the year that digital media was published.
Digital_Media.genre	VARCHAR	This is the genre category that the digital media item belongs to.
Digital_Media.availability_status	VARCHAR	This is the status that indicates whether the digital media is in stock or not. There are two types of messages that can be displayed: TRUE, FALSE.
ISSN	CHAR(9)	This is the unique ID that is used to identify each individual Magazine in the library's database.
Magazines.author	VARCHAR	This is the name of the author who created the Magazine.
Magazines.title	VARCHAR	This is the title associated with a Magazine.
Magazines.issue_number	INT	The issue number of the library item's library magazine.
Magazines.publication_date	DATE_TYPE (YYYY-MM-DD)	This is the date that the Magazine was published.
Magazines.genre	VARCHAR	This is the genre category that the Magazine item belongs to.
Magazines.availability_status	VARCHAR	This is the status that indicates

		whether the magazine is in stock or not. There are two types of messages that can be displayed: TRUE, FALSE.
report_id	INT	This is the unique ID that is associated with each of the reports that are generated.
report_date	DATE_TYPE (YYYY-MM-DD)	This is the date when the report was generated.
total_amount_owed_per_day	DECIMAL(5,2)	This is the total amount that is owed to the library, inclusive of all member accounts.
total_amount_paid_per_day	DECIMAL(5,2)	This is the total amount that is paid per day to the library inclusive of all member accounts.
number_of_checkouts	INT	This is the total number of library items that have a checked-out status.
account_id	INT	This is the unique ID that is associated with each member account.
incurred_fees	DECIMAL(5,2)	This is the total fees that the member account has incurred.
total_amount_paid	DECIMAL(5,2)	This is the total amount that has been paid by the member account.
overdue_balance	DECIMAL(5,2)	This is the balance that is overdue/remaining for the member account, indicating what needs to be paid.
recommended_items	VARCHAR	This indicates the library items that are recommended to each member accounts.
staff_id	INT	This is the unique identifier for the staff to determine their role and other relations.

role	VARCHAR	This indicates the employee title of the staff in the library.
member_id	INT	This is the unique identifier for the member to determine their specific attributes and transactions.
limit	INT	This is the limit on the number of items a library member can check out. The limit depends on the membership type: Regular members = 0 - 5 items Students = 0 -10 items Senior citizens = 0 - 5 items.
fee_type	INT	This is the amount of fee charged per overdue item. The type depends on the membership type: Regular = $0.30 Students = $0.20 Senior citizens = $0.15.
type_id	INT	This will identify which member type plan the library member is on. If it is 1, then it is Regular, if it is 2, then it is Student, if it is 3, then it is Senior Citizen.
Library_Member.name	VARCHAR	This is the name of the library member.
contact_information	VARCHAR	The contact information of the library member is stored.
account_status	VARCHAR	This is for determining if the client is Active or Inactive.
waitlist_id	INT	This is the unique identifier of the waitlist entity and will tell of each entry that has been added to the waitlist.
request_date	DATE_TYPE	This is the date that the copy

	(YYYY-MM-DD)	has been requested on the waitlist.
waitlist_status	VARCHAR	The types of status here would be: On_Hold and Available_For_CheckOut so that the client can determine if they can get off the waitlist for a library item.
fulfilled_date	DATE_TYPE (YYYY-MM-DD)	This is the date that the copy is taken of the waitlist - the request is being fulfilled.
Regular.available_limit	INT	The amount of items that are allowed to check out based on the regular member's current checked-out item count. The amount of items they have left to check out based on the limit.
Student.available_limit	INT	The amount of items that are allowed to check out based on the student member's current checked-out item count. The amount of items they have left to check out based on the limit.
Senior_Citizen.available_limit	INT	The amount of items that are allowed to check out based on the senior citizen's member's current checked-out item count. The amount of items they have left to check out based on the limit.
recommendation_id	INT	This is the unique ID that is used to identify each time a library member recommends a library item.
amount_paid	DECIMAL(5,2)	This is the amount that the library member has paid for their fees.
paid_date	DATE_TYPE (YYYY-MM-DD)	This is the date that the library member has paid their fees.

stars_given	INT(restrict range to 0-5).	This is the rating given by a member ranging from 0-5 stars for a book.
timestamp	DATE_TYPE (YYYY-MM-DD)	This is the timestamp of when the library member last viewed their member account.

5 Normalization Considerations

Normalization was considered in the diagrams. We performed normalization numerous times, and will do so as necessary.

6 Appendices

Refer to *01 - Project Plan/Vision* for any additional information on the scope and definition of the project.

Refer to *07 - Glossary* for any additional information regarding abbreviations and terms.

7 GitHub Repository Management

All members of the ASKNrEceive team will regularly manage, update, and commit to the GitHub Repository. The repository will be publicly available for view and accessing here: Library Database Management Project.

**** Note: All our Project Meeting Logs will be housed in the GitHub Repository on the Wiki Page. Please reference it as needed. Our Team Profile is also on the Wiki Page.**