PROJECT PART 4: THE LOGICAL RELATIONAL MODEL

1, Introduction

Project Overview:

The Library Management System (LMS) is designed to facilitate the effective management of library assets, members, staff, and transactions. It allows for cataloging of physical and digital media, managing membership accounts, tracking item borrowings, handling reservations, and enforcing overdue fines. The system streamlines operations such as book borrowing and return, fine calculation, and report generation, enhancing both user experience and administrative efficiency.

Scope:

This LMS focuses on:

Cataloging books, magazines, and digital media.

Managing members and staff.

Processing borrowing and returning transactions.

Managing item reservations.

Handling fines for overdue returns.

Excluded features include e-book lending, external system integration, and recommendation systems.

Glossary:

ISBN: International Standard Book Number.

Member: A registered user who can borrow items.

Transaction: A borrowing or returning activity.

Reservation: A request for an item currently unavailable.

Availability: Boolean indicator of whether an item can be borrowed.

Fine: A penalty for overdue items.

1. Relational Schema Mapping

Entity/Relationship Set	Corresponding Table
`auth_user`	`auth_user`
`accounts_user`	`accounts_user`
`catalog_item`	`catalog_item`
`reports_report`	`reports_report`
`loans_reservation`	`loans_reservation`
`loans_borrowingtransaction`	`loans_borrowingtransaction`
`reservations_reservation`	`reservations_reservation`
`notifications_notification`	`notifications_notification`
`core_auditing`	`core_auditing`
`core_systemconfig`	`core_systemconfig`
`auth_group`	`auth_group`
`auth_permission`	`auth_permission`
`auth_user_groups`	`auth_user_groups`
`auth_user_user_permissions`	`auth_user_user_permissions`
`auth_group_permissions`	`auth_group_permissions`
`accounts_user_user_permissions`	`accounts_user_user_permissions`
`accounts_user_groups`	`accounts_user_groups`
`django_content_type`	`django_content_type`

---

2. Attributes and Domains (Data Dictionary)

Table: `auth_user`

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
password	VARCHAR(128)	Hashed password
last_login	TIMESTAMP	Last login time
is_superuser	BOOLEAN	Is user admin
first_name	VARCHAR(150)	First name
last_name	VARCHAR(150)	Last name
email	VARCHAR(254)	Email address
is_staff	BOOLEAN	Staff status
is_active	BOOLEAN	Active user status
date_joined	TIMESTAMP	Account creation timestamp

Table: `accounts_user`

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
user_id	INTEGER (FK)	Foreign key to `auth_user.id`
profile_pic	VARCHAR	URL/path to profile picture
bio	TEXT	User biography

Table: **`catalog_item`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
item_type	VARCHAR(50)	Type of item
isbn	VARCHAR(20)	ISBN number
issue_number	INTEGER	Issue number
publication_date	DATE	Date of publication
title	VARCHAR(200)	Title
publication_year	INTEGER	Year published
genre	VARCHAR(50)	Genre
creator	VARCHAR(100)	Creator/Author
item_format	VARCHAR(50)	Format (e.g. book, DVD)
availability	BOOLEAN	Availability status

Table: **`reports_report`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
created_by_id	INTEGER (FK)	Foreign key to `auth_user.id`
title	VARCHAR(100)	Report title
description	TEXT	Detailed report description
created_at	TIMESTAMP	Timestamp of creation

Table: **`loans_reservation`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
member_id	INTEGER (FK)	Foreign key to `auth_user.id`
item_id	INTEGER (FK)	Foreign key to `catalog_item.id`
reservation_id	INTEGER (FK)	Foreign key to another reservation
request_date	DATE	Date of request
due_date	DATE	Due date

Attribute	Domain	Description
status	VARCHAR(20)	Reservation status

Table: **`loans_borrowingtransaction`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
member_id	INTEGER (FK)	Foreign key to `auth_user.id`
reservation_id	INTEGER (FK)	Foreign key to `loans_reservation.id`
borrow_date	DATE	Date borrowed
return_date	DATE	Date returned

Table: **`reservations_reservation`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
content_type_id	INTEGER (FK)	Foreign key to `django_content_type.id`
object_id	INTEGER	ID of referenced object
timestamp	TIMESTAMP	Timestamp of reservation

Table: **`notifications_notification`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
recipient_id	INTEGER (FK)	Foreign key to `auth_user.id`
message	TEXT	Notification message
created_at	TIMESTAMP	Time sent
is_read	BOOLEAN	Read status

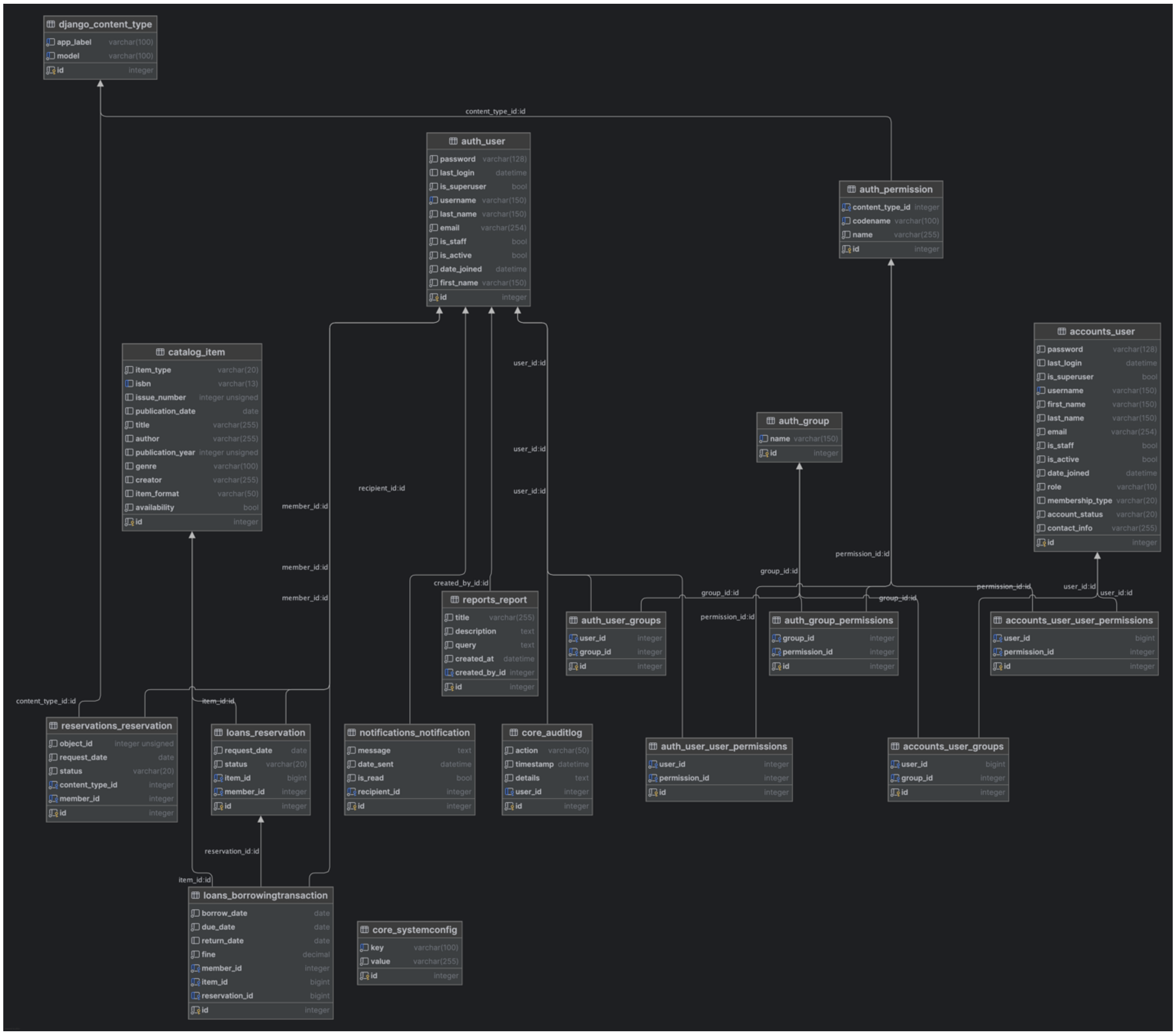
Table: **`core_auditing`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
user_id	INTEGER (FK)	Foreign key to `auth_user.id`
action	VARCHAR(100)	Performed action
timestamp	TIMESTAMP	Time of action

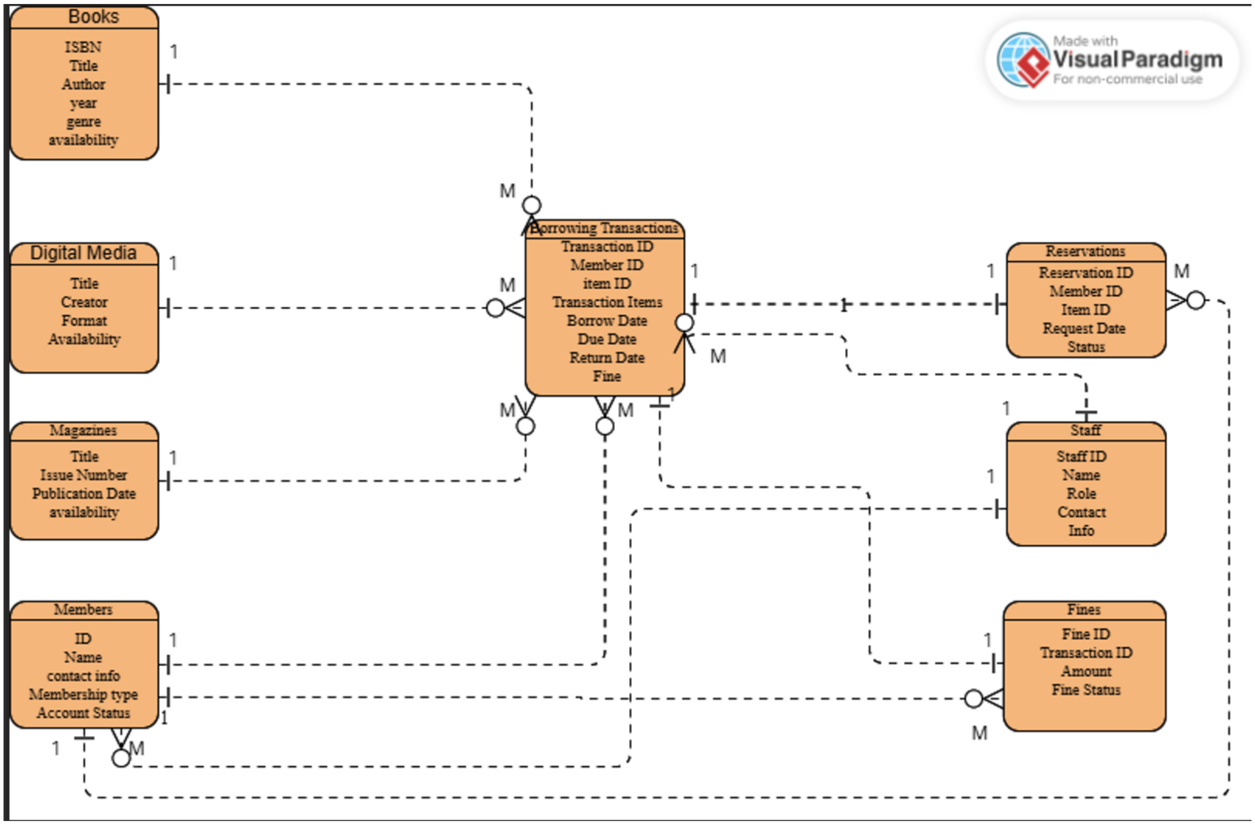
Table: **`core_systemconfig`**

Attribute	Domain	Description
id	INTEGER (PK)	Unique identifier
config_key	VARCHAR(100)	Config key
config_value	TEXT	Config value

RELATIONAL SCHEMA DIAGRAM:



ER DIAGRAM:



3. Primary Keys and Foreign Keys

Primary Keys

Table	Primary Key
auth_user	id
accounts_user	id
catalog_item	id
reports_report	id
loans_reservation	id
loans_borrowingtransaction	id
reservations_reservation	id
notifications_notification	id
core_auditing	id
core_systemconfig	id
auth_group	id
auth_permission	id
auth_user_groups	id
auth_user_user_permissions	id
auth_group_permissions	id
accounts_user_user_permissions	id
accounts_user_groups	id

Table	Primary Key
`django_content_type`	`id`

4. Foreign Keys

- `catalog_item.member_id` → `auth_user.id`
- `loans_reservation.member_id` → `auth_user.id`
- `loans_reservation.item_id` → `catalog_item.id`
- `loans_reservation.reservation_id` → `reservations_reservation.id`
- `loans_borrowingtransaction.member_id` → `auth_user.id`
- `loans_borrowingtransaction.reservation_id` → `loans_reservation.id`
- `reports_report.created_by_id` → `auth_user.id`
- `notifications_notification.recipient_id` → `auth_user.id`
- `core_auditing.user_id` → `auth_user.id`
- `auth_user_groups.user_id` → `auth_user.id`
- `auth_user_groups.group_id` → `auth_group.id`
- `auth_user_user_permissions.user_id` → `auth_user.id`
- `auth_user_user_permissions.permission_id` → `auth_permission.id`
- `auth_group_permissions.group_id` → `auth_group.id`
- `auth_group_permissions.permission_id` → `auth_permission.id`
- `auth_permission.content_type_id` → `django_content_type.id`
- `reservations_reservation.content_type_id` → `django_content_type.id`
- `reservations_reservation.object_id` → (polymorphic reference)

---

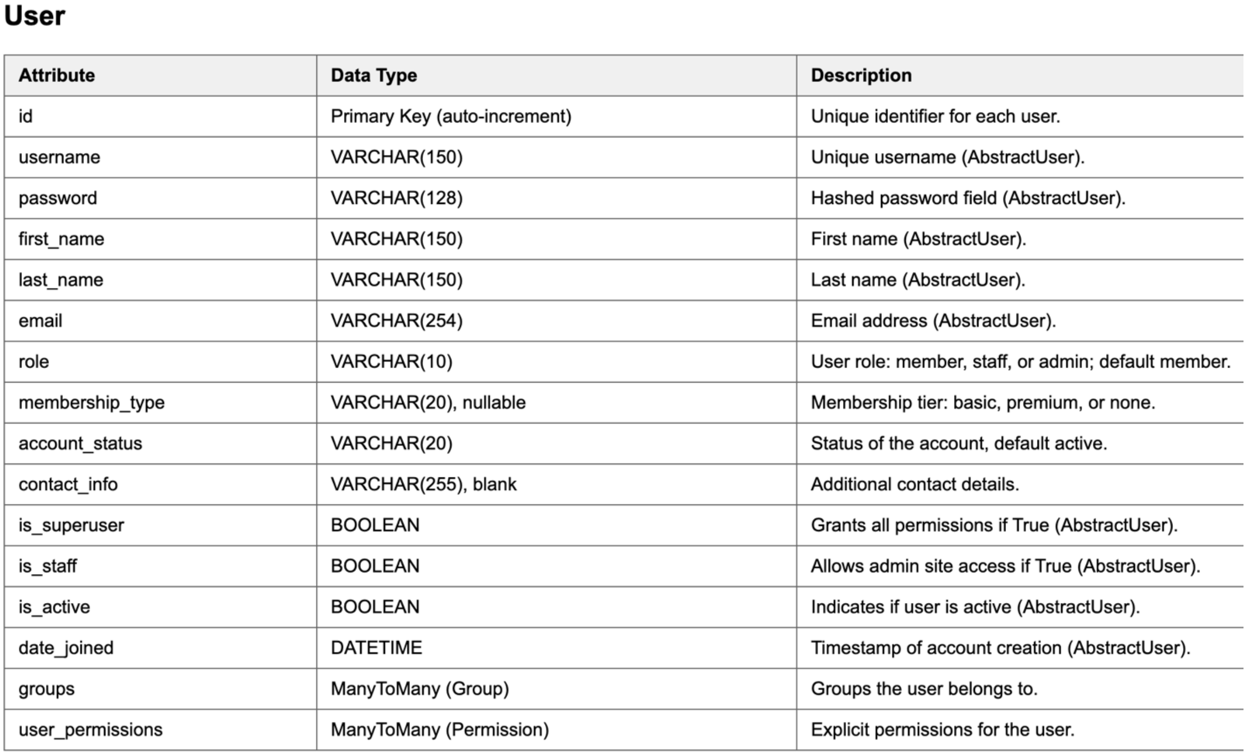
5. Functional Dependencies (FDs)

- In `auth_user`:
  `id → password, last_login, is_superuser, first_name, last_name, email, is_staff, is_active, date_joined`
- In `catalog_item`:
  `id → item_type, isbn, issue_number, publication_date, title, publication_year, genre, creator, item_format, availability`
- In `loans_reservation`:
  `id → request_date, due_date, status, item_id, member_id, reservation_id`
- In `auth_user_groups`:
  `{user_id, group_id} → ∅` (composite key)
- In `auth_permission`:
  `id → content_type_id, codename, name`
- In `auth_group_permissions`:
  `{group_id, permission_id} → ∅`
- Here's your **Schema Documentation with a Data Dictionary** formatted for easy copying into a Word document using table structure headings. You can paste each section into Word and use the “Insert Table” feature to make it visually clean.

-

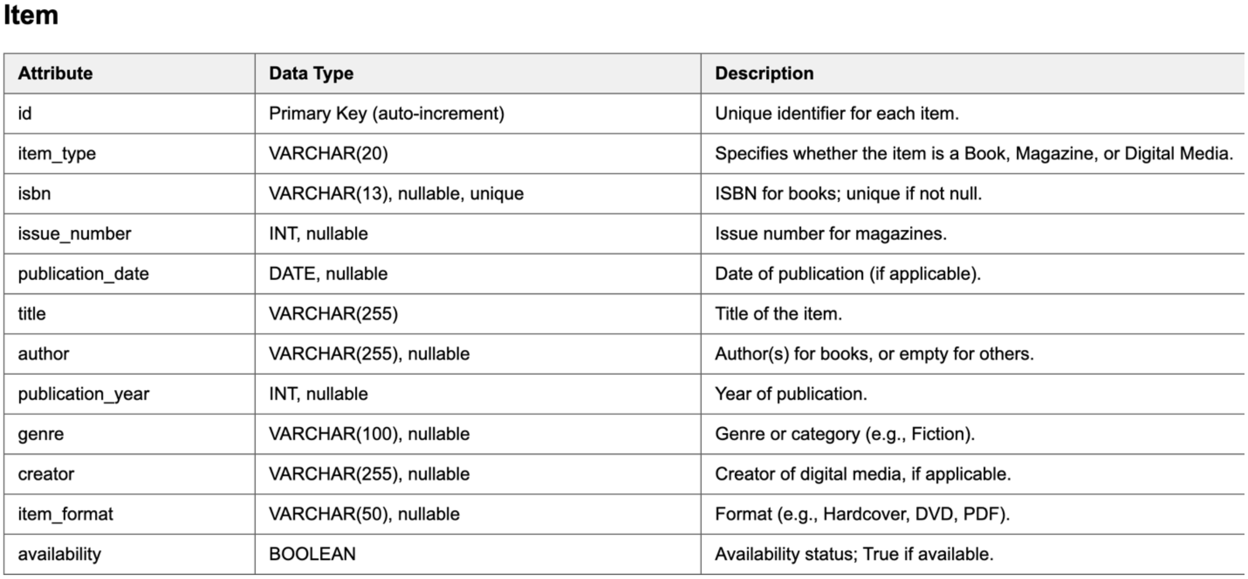
---

3. Schema Documentation with a Data Dictionary



User

Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each user.
username	VARCHAR(150)	Unique username (AbstractUser).
password	VARCHAR(128)	Hashed password field (AbstractUser).
first_name	VARCHAR(150)	First name (AbstractUser).
last_name	VARCHAR(150)	Last name (AbstractUser).
email	VARCHAR(254)	Email address (AbstractUser).
role	VARCHAR(10)	User role: member, staff, or admin; default member.
membership_type	VARCHAR(20), nullable	Membership tier: basic, premium, or none.
account_status	VARCHAR(20)	Status of the account, default active.
contact_info	VARCHAR(255), blank	Additional contact details.
is_superuser	BOOLEAN	Grants all permissions if True (AbstractUser).
is_staff	BOOLEAN	Allows admin site access if True (AbstractUser).
is_active	BOOLEAN	Indicates if user is active (AbstractUser).
date_joined	DATETIME	Timestamp of account creation (AbstractUser).
groups	ManyToMany (Group)	Groups the user belongs to.
user_permissions	ManyToMany (Permission)	Explicit permissions for the user.



Item

Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each item.
item_type	VARCHAR(20)	Specifies whether the item is a Book, Magazine, or Digital Media.
isbn	VARCHAR(13), nullable, unique	ISBN for books; unique if not null.
issue_number	INT, nullable	Issue number for magazines.
publication_date	DATE, nullable	Date of publication (if applicable).
title	VARCHAR(255)	Title of the item.
author	VARCHAR(255), nullable	Author(s) for books, or empty for others.
publication_year	INT, nullable	Year of publication.
genre	VARCHAR(100), nullable	Genre or category (e.g., Fiction).
creator	VARCHAR(255), nullable	Creator of digital media, if applicable.
item_format	VARCHAR(50), nullable	Format (e.g., Hardcover, DVD, PDF).
availability	BOOLEAN	Availability status; True if available.

SystemConfig

Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each config entry.
key	VARCHAR(100), unique	Unique configuration key.
value	VARCHAR(255)	Configuration value.

AuditLog

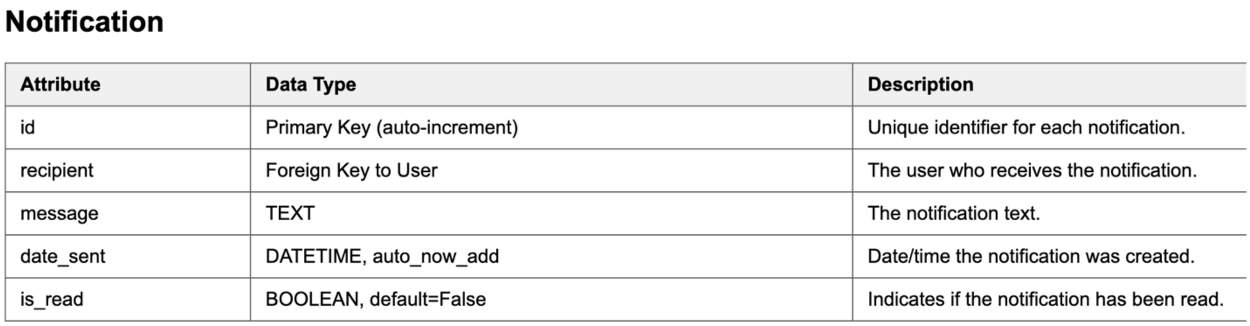
Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for the audit record.
action	VARCHAR(50)	Name or code for the action performed.
user	Foreign Key to User, nullable	Reference to the user who performed the action.
timestamp	DATETIME, auto_now_add	Date/time of the action.
details	TEXT, blank	Additional context about the event.

BorrowingTransaction

Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each borrowing transaction.
member	Foreign Key to User	Borrowing user.
item	Foreign Key to Item	Borrowed item.
borrow_date	DATE	Date the item was borrowed.
due_date	DATE	Date the item is due for return.
return_date	DATE, nullable	Date the item was returned.
fine	DECIMAL(6,2), default=0.00	Fine amount for late returns, etc.
reservation	OneToOne → Reservation, nullable	Links to an associated reservation if applicable.

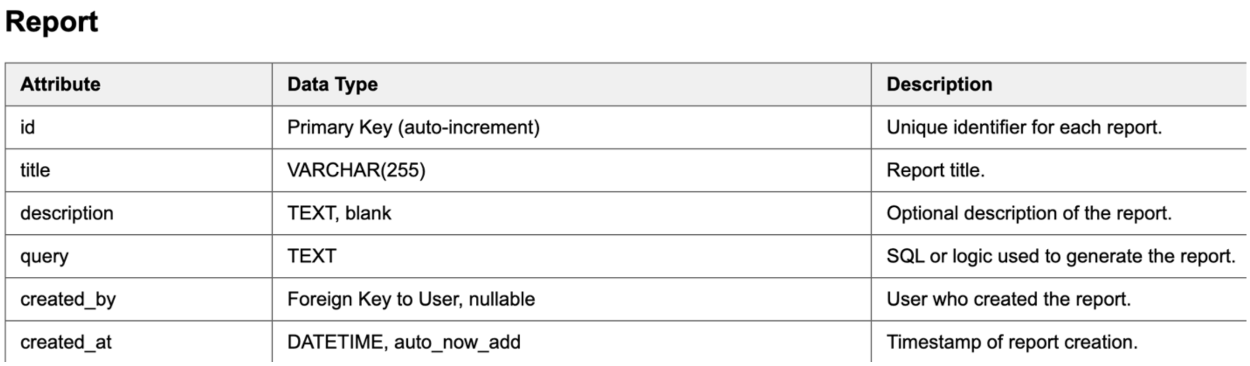
Reservation

Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each reservation.
member	Foreign Key to User	The user placing the reservation.
item	Foreign Key to Item	The reserved item.
request_date	DATE, auto_now_add	Date the reservation request was created.
status	VARCHAR(20), default='pending'	Current status (e.g., pending, fulfilled, canceled).



Notification

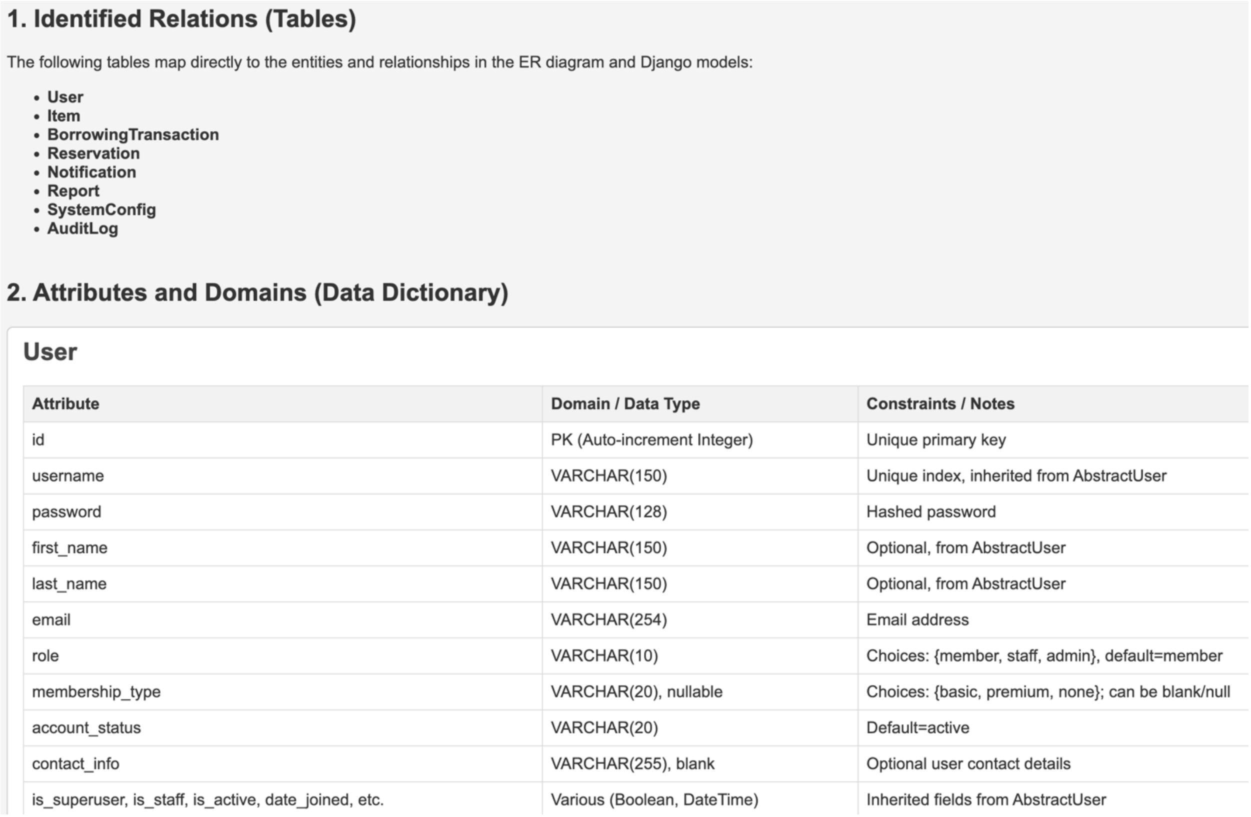
Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each notification.
recipient	Foreign Key to User	The user who receives the notification.
message	TEXT	The notification text.
date_sent	DATETIME, auto_now_add	Date/time the notification was created.
is_read	BOOLEAN, default=False	Indicates if the notification has been read.



Report

Attribute	Data Type	Description
id	Primary Key (auto-increment)	Unique identifier for each report.
title	VARCHAR(255)	Report title.
description	TEXT, blank	Optional description of the report.
query	TEXT	SQL or logic used to generate the report.
created_by	Foreign Key to User, nullable	User who created the report.
created_at	DATETIME, auto_now_add	Timestamp of report creation.

SCHEMA:



1. Identified Relations (Tables)

The following tables map directly to the entities and relationships in the ER diagram and Django models:

- **User**
- **Item**
- **BorrowingTransaction**
- **Reservation**
- **Notification**
- **Report**
- **SystemConfig**
- **AuditLog**

2. Attributes and Domains (Data Dictionary)

User

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
username	VARCHAR(150)	Unique index, inherited from AbstractUser
password	VARCHAR(128)	Hashed password
first_name	VARCHAR(150)	Optional, from AbstractUser
last_name	VARCHAR(150)	Optional, from AbstractUser
email	VARCHAR(254)	Email address
role	VARCHAR(10)	Choices: {member, staff, admin}, default=member
membership_type	VARCHAR(20), nullable	Choices: {basic, premium, none}; can be blank/null
account_status	VARCHAR(20)	Default=active
contact_info	VARCHAR(255), blank	Optional user contact details
is_superuser, is_staff, is_active, date_joined, etc.	Various (Boolean, DateTime)	Inherited fields from AbstractUser

BorrowingTransaction

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
member_id	FK → User(id)	On delete CASCADE
item_id	FK → Item(id)	On delete CASCADE
borrow_date	DATE	Date the item was borrowed
due_date	DATE	Date the item is due
return_date	DATE, nullable	Date the item was returned
fine	DECIMAL(6,2), default=0.00	Monetary fine if overdue
reservation_id	OneToOne FK → Reservation(id), nullable	On delete SET_NULL; links to a reservation if any

Reservation

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
member_id	FK → User(id)	On delete CASCADE
item_id	FK → Item(id)	On delete CASCADE
request_date	DATE, auto_now_add	Date reservation was requested
status	VARCHAR(20)	Default='pending' (e.g. pending, fulfilled, canceled)

Notification

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
recipient_id	FK → User(id)	On delete CASCADE
message	TEXT	The body of the notification
date_sent	DATETIME, auto_now_add	Timestamp of notification creation
is_read	BOOLEAN	False by default

Report

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
title	VARCHAR(255)	Report title
description	TEXT, blank	Optional additional details
query	TEXT	SQL or logic to generate report
created_by_id	FK → User(id), nullable	On delete SET_NULL
created_at	DATETIME, auto_now_add	Date/time the report was created

SystemConfig

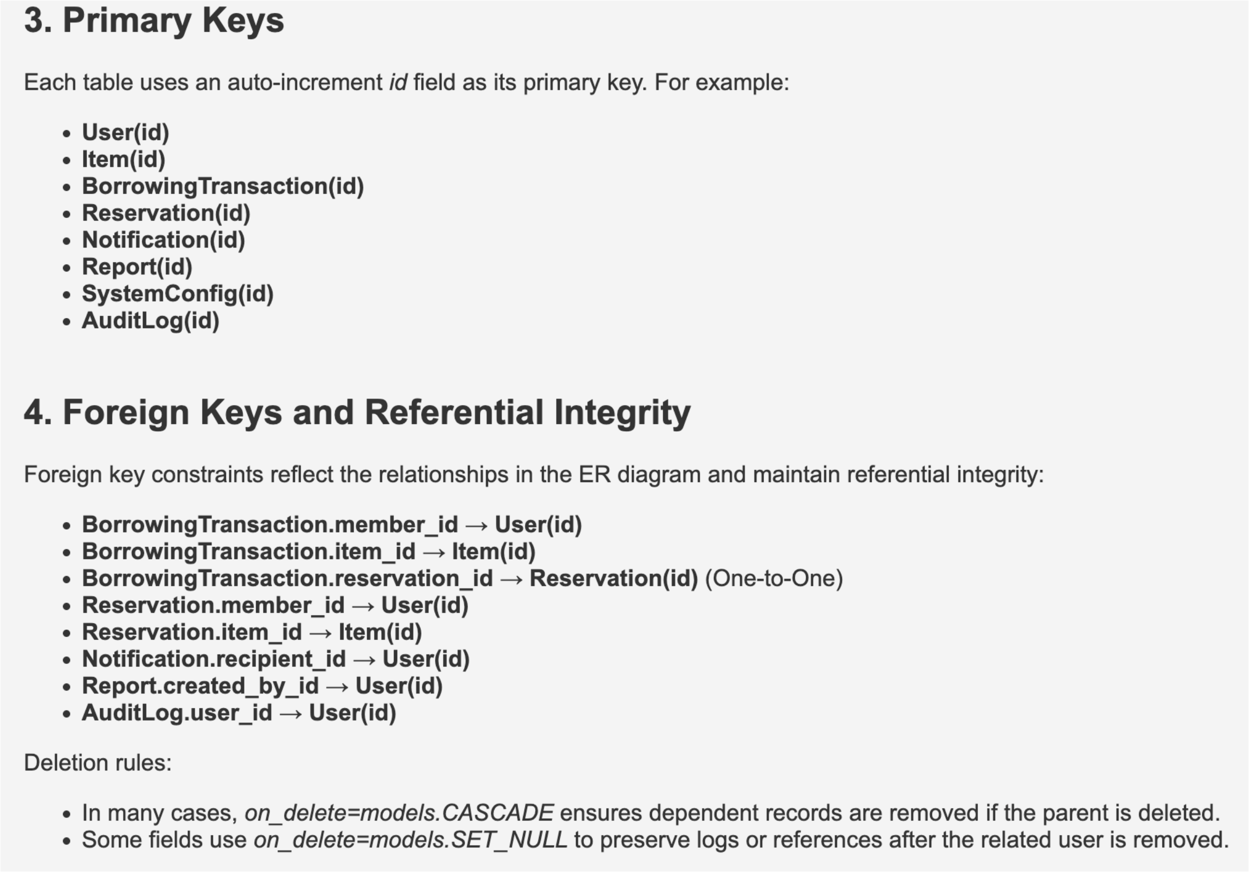
Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
key	VARCHAR(100)	Unique config key
value	VARCHAR(255)	Config value

AuditLog

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
action	VARCHAR(50)	Describes the logged action
user_id	FK → User(id), nullable	On delete SET_NULL
timestamp	DATETIME, auto_now_add	Log creation time
details	TEXT, blank	Additional context info

Item

Attribute	Domain / Data Type	Constraints / Notes
id	PK (Auto-increment Integer)	Unique primary key
item_type	VARCHAR(20)	Choices: {book, magazine, digital}
isbn	VARCHAR(13), nullable, unique	Optional unique ISBN for books
issue_number	INT, nullable	Issue # if magazine
publication_date	DATE, nullable	Date of publication
title	VARCHAR(255)	Required title
author	VARCHAR(255), nullable	Author(s) if applicable
publication_year	INT, nullable	Year of publication
genre	VARCHAR(100), nullable	Genre/category
creator	VARCHAR(255), nullable	Creator for digital media
item_format	VARCHAR(50), nullable	Format (e.g. Hardcover, PDF)
availability	BOOLEAN	True if item is available, default=True



3. Primary Keys

Each table uses an auto-increment *id* field as its primary key. For example:

- **User(id)**
- **Item(id)**
- **BorrowingTransaction(id)**
- **Reservation(id)**
- **Notification(id)**
- **Report(id)**
- **SystemConfig(id)**
- **AuditLog(id)**

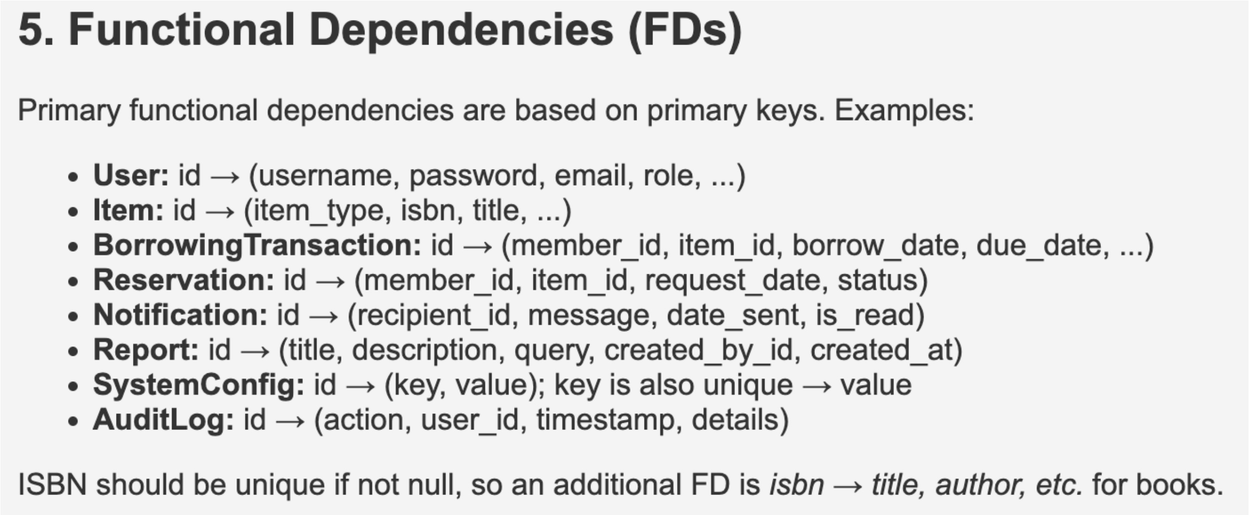
4. Foreign Keys and Referential Integrity

Foreign key constraints reflect the relationships in the ER diagram and maintain referential integrity:

- **BorrowingTransaction.member_id → User(id)**
- **BorrowingTransaction.item_id → Item(id)**
- **BorrowingTransaction.reservation_id → Reservation(id)** (One-to-One)
- **Reservation.member_id → User(id)**
- **Reservation.item_id → Item(id)**
- **Notification.recipient_id → User(id)**
- **Report.created_by_id → User(id)**
- **AuditLog.user_id → User(id)**

Deletion rules:

- In many cases, *on_delete=models.CASCADE* ensures dependent records are removed if the parent is deleted.
- Some fields use *on_delete=models.SET_NULL* to preserve logs or references after the related user is removed.



5. Functional Dependencies (FDs)

Primary functional dependencies are based on primary keys. Examples:

- **User:** id → (username, password, email, role, ...)
- **Item:** id → (item_type, isbn, title, ...)
- **BorrowingTransaction:** id → (member_id, item_id, borrow_date, due_date, ...)
- **Reservation:** id → (member_id, item_id, request_date, status)
- **Notification:** id → (recipient_id, message, date_sent, is_read)
- **Report:** id → (title, description, query, created_by_id, created_at)
- **SystemConfig:** id → (key, value); key is also unique → value
- **AuditLog:** id → (action, user_id, timestamp, details)

ISBN should be unique if not null, so an additional FD is *isbn → title, author, etc.* for books.