EECS 447

Library Database Management System
Logical Relational Schema Design

Version 1.0

Table of Contents

Logical Relational Schema Design

1. Introduction

1.1 Project Overview

The Library Database Management System project will deliver an efficient, user-friendly, and secure database system to support small library operations. The DBMS will enable seamless management of loanable items (books, digital media, magazines), enforce borrowing rules based on diverse membership categories, and generate insightful reports. By integrating modern database design principles, the system will streamline item tracking, membership management, and financial oversight.

1.2 Scope

The Library Database Management System (LMS) project will design and implement a relational database for a small library to manage loanable items, memberships, borrowing rules, and generate reports. It will model entities like books, digital media, magazines, and clients, and enforce constraints like borrowing limits and fees based on membership type. The system will include features for managing loans, returns, and client accounts, with user interfaces for both staff and clients. Advanced queries will generate financial and activity reports, while concurrency and transaction management will ensure seamless multi-user operations. The LMS will be developed through domain modeling, database design, and implementation phases, ensuring functionality and data integrity.

1.3 Glossary

LMS - Library Management System
DBMS - Database Management System

2. Relational Schema Mapping

2.1 Identify Relations

- User()
- Has_Membership()   (*Derived from the “Has” relationship)
- Member()
- Membership()
- Membership_Transaction()   (*Derived from the “Determines” relationship)
- Staff()
- Transaction()
- Performs()
- Involves()
- Purchase()
- Borrow()
- Item()
- Magazine()
- Music()
- DVD()
- Book()
- Digital_Media()
- Author()
- Writes()
- Genre()
- Belongs()
- Publisher()
- Publishes()

2.2 Define Attributes and Domains (Domains defined in 3.2

- User(
  - User_ID
  - Name
  - Address
  - Email
  - Phone
  -)
- Has_Membership(
  - User_ID
  - Card_Num
  - Current_Borrows
  -)
- Member(
  - User_ID
  - Membership_Type
  -)
- Membership(
  - Membership_Type
  - Borrow_Limit
  - Discount_Rate
  - Membership_Fee
  -)
- Membership_Transaction(
  - Transaction_ID
  - Membership_Type
  -)
- Staff(
  - User_ID
  - Position
  - Salary
  -)
- Transaction(
  - Transaction_ID

    - Date
    - Status
    -)
- Performs(
    - Transaction_ID
    - User_ID
    -)
- Involves(
    - Transaction_ID
    - Item_ID
    -)
- Purchase(
    - Transaction_ID
    - Amount
    - Purchase_Type
    - Method
    -)
- Borrow(
    - Transaction_ID
    - Borrow_Date
    - Due_Date
    - Return_Date
    - Late_Fee
    -)
- Item(
    - Item_ID
    - Price
    - Year
    - Availability
    - Item_Type
    -)
- Magazine(
    - Item_ID
    - ISSN
    - Name

    - Edition
    - Publish_Date
    -)
- Music(
    - Item_ID
    - Title
    - Artist
    - Album
    - Format
    -)
- DVD(
    - Item_ID
    - Name
    - Director
    - Duration
    -)
- Book(
    - Item_ID
    - ISBN
    - Title
    - Subject
    -)
- Digital_Media(
    - Item_ID
    - DOI
    - Title
    - Media_Type
    - Creator
    - Release_Date
    -)
- Author(
    - Author_ID
    - Name
    - Biography
    - Date_of_Birth

  - Date_of_Death
  -)
- Writes(
  - Author_ID
  - Item_ID
  -)
- Genre(
  - Genre_Name
  - Description
  - Location
  -)
- Belongs(
  - Item_ID
  - Genre_Name
  -)
- Publisher(
  - Publisher_ID
  - Name
  - Address
  - Phone
  - Email
  -)
- Publishes(
  - Item_ID
  - Publisher_ID
  -)

2.3 Define Primary Keys (PK)

- User PK: **User_ID**
- Has_Membership PK: **User_ID**
- Member PK: **User_ID**
- Staff PK: **User_ID**
- Membership PK: **Membership_Type**
- Membership_Transaction PK: **Transaction_ID**
- Transaction PK: **Transaction_ID**
- Performs PK: **Transaction_ID**
- Purchase PK: **Transaction_ID**
- Borrow PK: **Transaction_ID**
- Involves PK: **Transaction_ID**, **Item_ID**
- Item PK: **Item_ID**
- Magazine PK: **Item_ID**
- Music PK: **Item_ID**
- DVD PK: **Item_ID**
- Book PK: **Item_ID**
- Digital_Media PK: **Item_ID**
- Author PK: **Author_ID**
- Writes PK: **Item_ID**, **Author_ID**
- Genre PK: **Genre_Name**
- Belongs PK: **Item_ID**
- Publisher PK: **Publisher_ID**
- Publishes PK: **Item_ID**

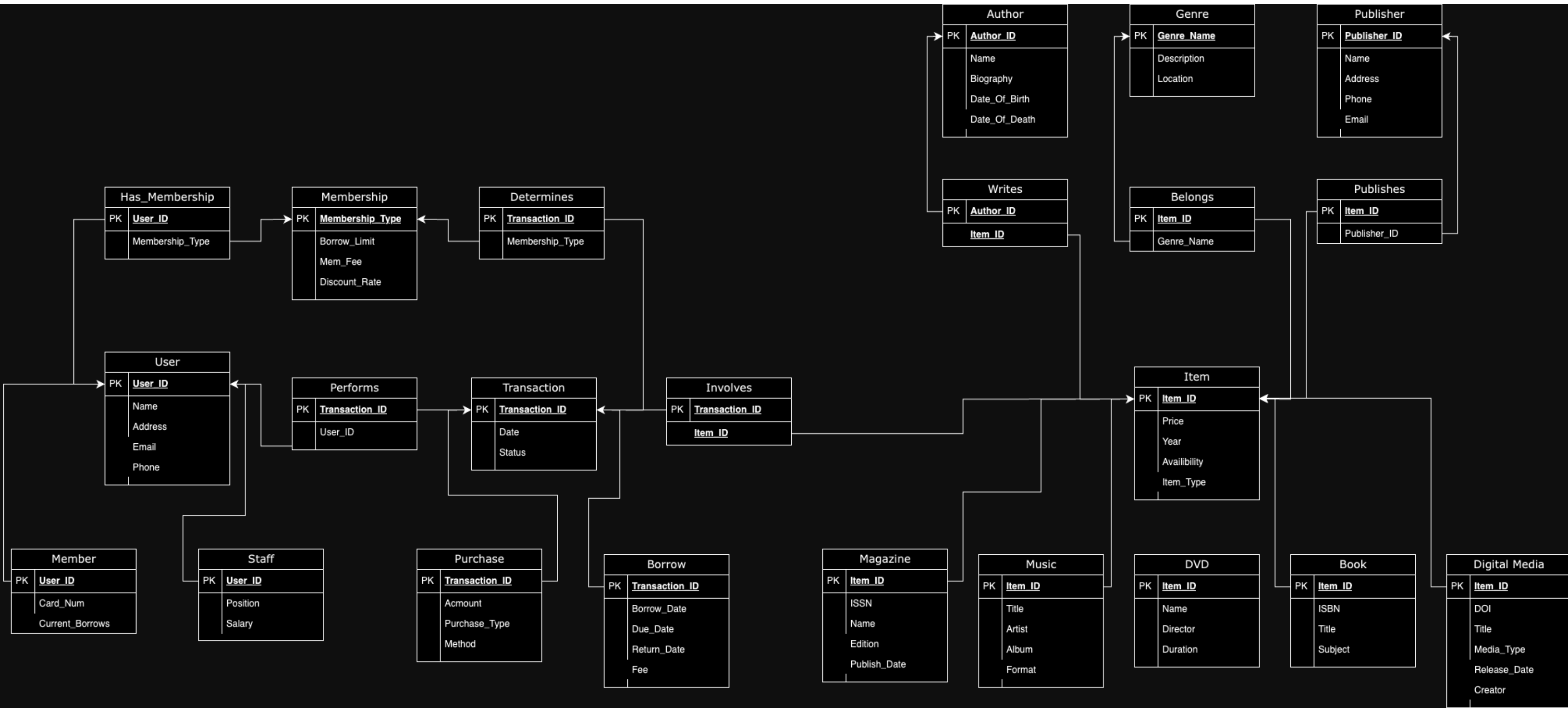
2.4 Establish Foreign Keys (FK)

- Has_Membership()
  - FK User_ID to User_ID in User
  - FK Membership_Type to Membership_Type in Membership
- *Member*
  - FK User_ID to User_ID in User
- *Staff*
  - FK User_ID to User_ID in User
- Membership_Transaction()

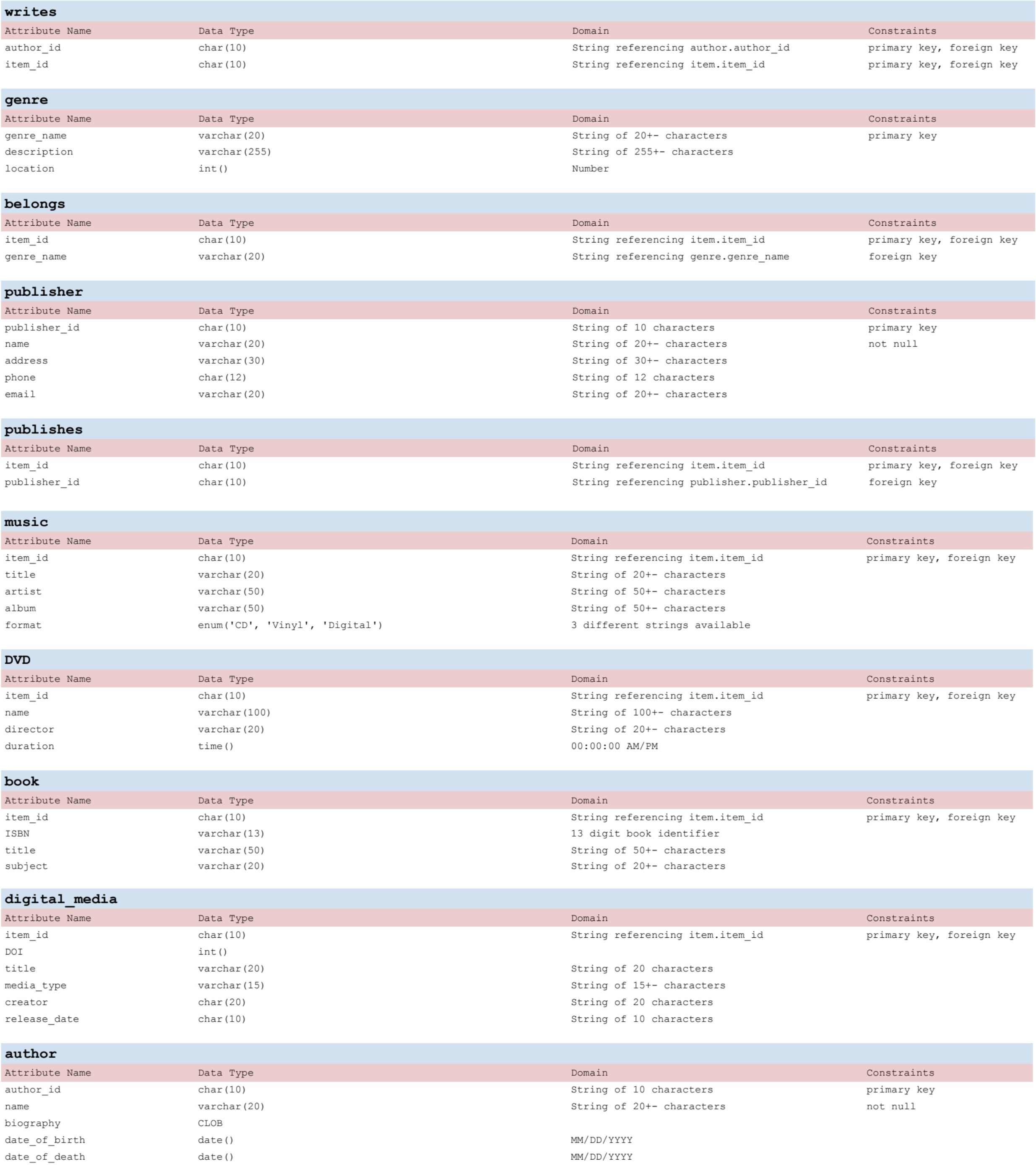
    - FK Transaction_ID to Transaction_ID in Transaction
    - FK Membership_Type to Membership_Type in Membership
- Performs()
    - FK Transaction_ID to Transaction_ID in Transaction
    - FK User_ID to User_ID in User
- *Purchase*
    - FK Transaction_ID to Transactin_ID in Transaction
- *Borrow*
    - FK Transaction_ID to Transactin_ID in Transaction
- Involves()
    - FK Transaction_ID to Transaction_ID in Transaction
    - FK Item_ID to Item_ID in Item
- *Magazine*
    - FK Item_ID to Item_ID in Item
- *Music*
    - FK Item_ID to Item_ID in Item
- *DVD*
    - FK Item_ID to Item_ID in Item
- *Book*
    - FK Item_ID to Item_ID in Item
- *Digital_Media*
    - FK Item_ID to Item_ID in Item
- Writes()
    - FK Item_ID to Item_ID in Item
    - FK Author_ID to Author_ID in Author
- Belongs()
    - FK Item_ID to Item_ID in Item
    - FK Genre_Name to Genre_Name in Genre
- Publishes()
    - FK Item_ID to Item_ID in Item
    - FK Pub_ID to Pub_ID in Publisher

3. Schema Documentation

3.1 Relational Schema Diagram



3.2 Data Dictionary.



writes

Attribute Name	Data Type	Domain	Constraints
author_id	char(10)	String referencing author.author_id	primary key, foreign key
item_id	char(10)	String referencing item.item_id	primary key, foreign key

genre

Attribute Name	Data Type	Domain	Constraints
genre_name	varchar(20)	String of 20+- characters	primary key
description	varchar(255)	String of 255+- characters	
location	int()	Number	

belongs

Attribute Name	Data Type	Domain	Constraints
item_id	char(10)	String referencing item.item_id	primary key, foreign key
genre_name	varchar(20)	String referencing genre.genre_name	foreign key

publisher

Attribute Name	Data Type	Domain	Constraints
publisher_id	char(10)	String of 10 characters	primary key
name	varchar(20)	String of 20+- characters	not null
address	varchar(30)	String of 30+- characters	
phone	char(12)	String of 12 characters	
email	varchar(20)	String of 20+- characters	

publishes

Attribute Name	Data Type	Domain	Constraints
item_id	char(10)	String referencing item.item_id	primary key, foreign key
publisher_id	char(10)	String referencing publisher.publisher_id	foreign key

music

Attribute Name	Data Type	Domain	Constraints
item_id	char(10)	String referencing item.item_id	primary key, foreign key
title	varchar(20)	String of 20+- characters	
artist	varchar(50)	String of 50+- characters	
album	varchar(50)	String of 50+- characters	
format	enum('CD', 'Vinyl', 'Digital')	3 different strings available	

DVD

Attribute Name	Data Type	Domain	Constraints
item_id	char(10)	String referencing item.item_id	primary key, foreign key
name	varchar(100)	String of 100+- characters	
director	varchar(20)	String of 20+- characters	
duration	time()	00:00:00 AM/PM	

book

Attribute Name	Data Type	Domain	Constraints
item_id	char(10)	String referencing item.item_id	primary key, foreign key
ISBN	varchar(13)	13 digit book identifier	
title	varchar(50)	String of 50+- characters	
subject	varchar(20)	String of 20+- characters	

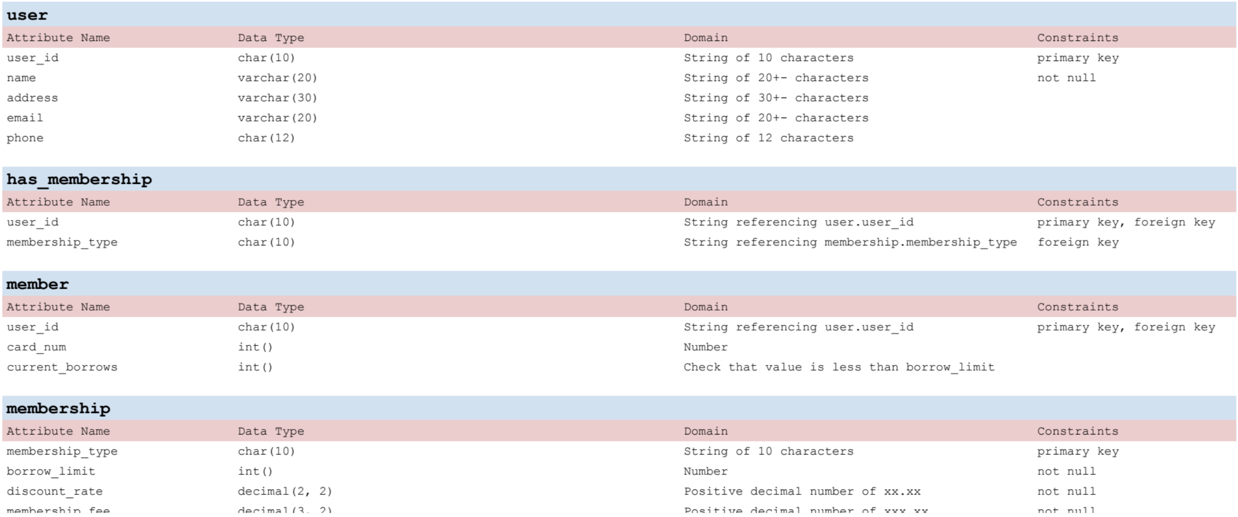
digital_media

Attribute Name	Data Type	Domain	Constraints
item_id	char(10)	String referencing item.item_id	primary key, foreign key
DOI	int()		
title	varchar(20)	String of 20 characters	
media_type	varchar(15)	String of 15+- characters	
creator	char(20)	String of 20 characters	
release_date	char(10)	String of 10 characters	

author

Attribute Name	Data Type	Domain	Constraints
author_id	char(10)	String of 10 characters	primary key
name	varchar(20)	String of 20+- characters	not null
biography	CLOB		
date_of_birth	date()	MM/DD/YYYY	
date_of_death	date()	MM/DD/YYYY	

Library DBMS	Version: 1.0
Logical Relational Schema Design	Date: 10/28/2024
EECS447_DBMS_LRS_Design	



user

Attribute Name	Data Type	Domain	Constraints
user_id	char(10)	String of 10 characters	primary key
name	varchar(20)	String of 20+- characters	not null
address	varchar(30)	String of 30+- characters	
email	varchar(20)	String of 20+- characters	
phone	char(12)	String of 12 characters	

has_membership

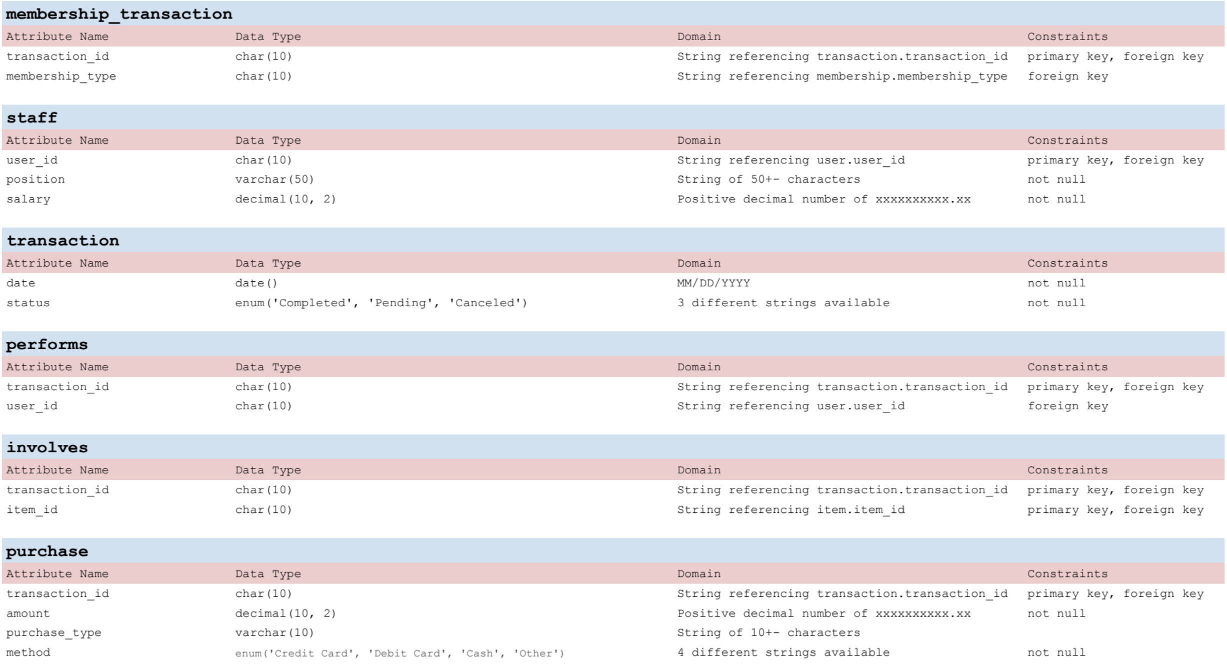
Attribute Name	Data Type	Domain	Constraints
user_id	char(10)	String referencing user.user_id	primary key, foreign key
membership_type	char(10)	String referencing membership.membership_type	foreign key

member

Attribute Name	Data Type	Domain	Constraints
user_id	char(10)	String referencing user.user_id	primary key, foreign key
card_num	int()	Number	
current_borrows	int()	Check that value is less than borrow_limit	

membership

Attribute Name	Data Type	Domain	Constraints
membership_type	char(10)	String of 10 characters	primary key
borrow_limit	int()	Number	not null
discount_rate	decimal(2, 2)	Positive decimal number of xx.xx	not null
membership_fee	decimal(3, 2)	Positive decimal number of xxx.xx	not null



membership_transaction

Attribute Name	Data Type	Domain	Constraints
transaction_id	char(10)	String referencing transaction.transaction_id	primary key, foreign key
membership_type	char(10)	String referencing membership.membership_type	foreign key

staff

Attribute Name	Data Type	Domain	Constraints
user_id	char(10)	String referencing user.user_id	primary key, foreign key
position	varchar(50)	String of 50+- characters	not null
salary	decimal(10, 2)	Positive decimal number of xxxxxxxxxx.xx	not null

transaction

Attribute Name	Data Type	Domain	Constraints
date	date()	MM/DD/YYYY	not null
status	enum('Completed', 'Pending', 'Canceled')	3 different strings available	not null

performs

Attribute Name	Data Type	Domain	Constraints
transaction_id	char(10)	String referencing transaction.transaction_id	primary key, foreign key
user_id	char(10)	String referencing user.user_id	foreign key

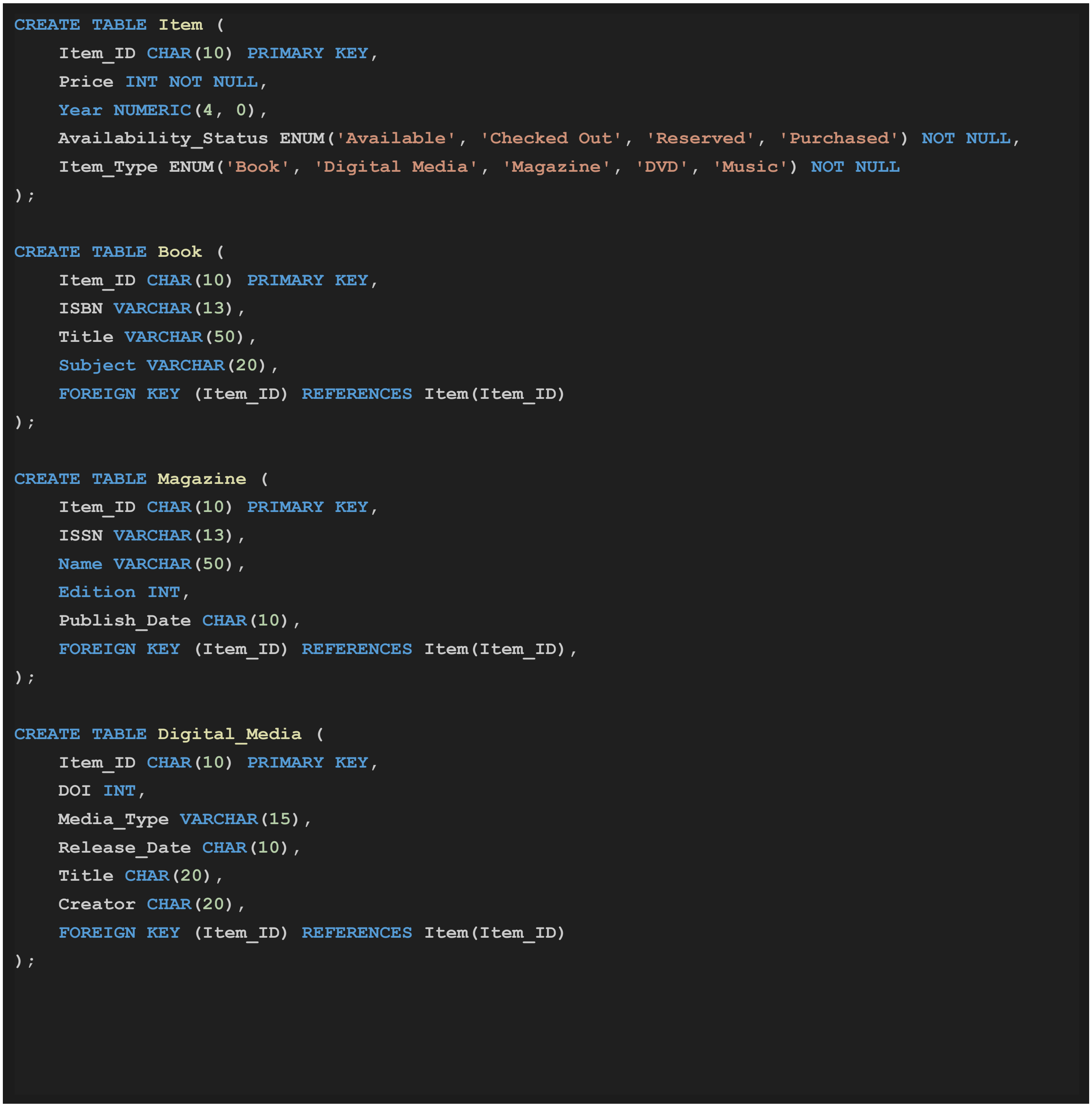
involves

Attribute Name	Data Type	Domain	Constraints
transaction_id	char(10)	String referencing transaction.transaction_id	primary key, foreign key
item_id	char(10)	String referencing item.item_id	primary key, foreign key

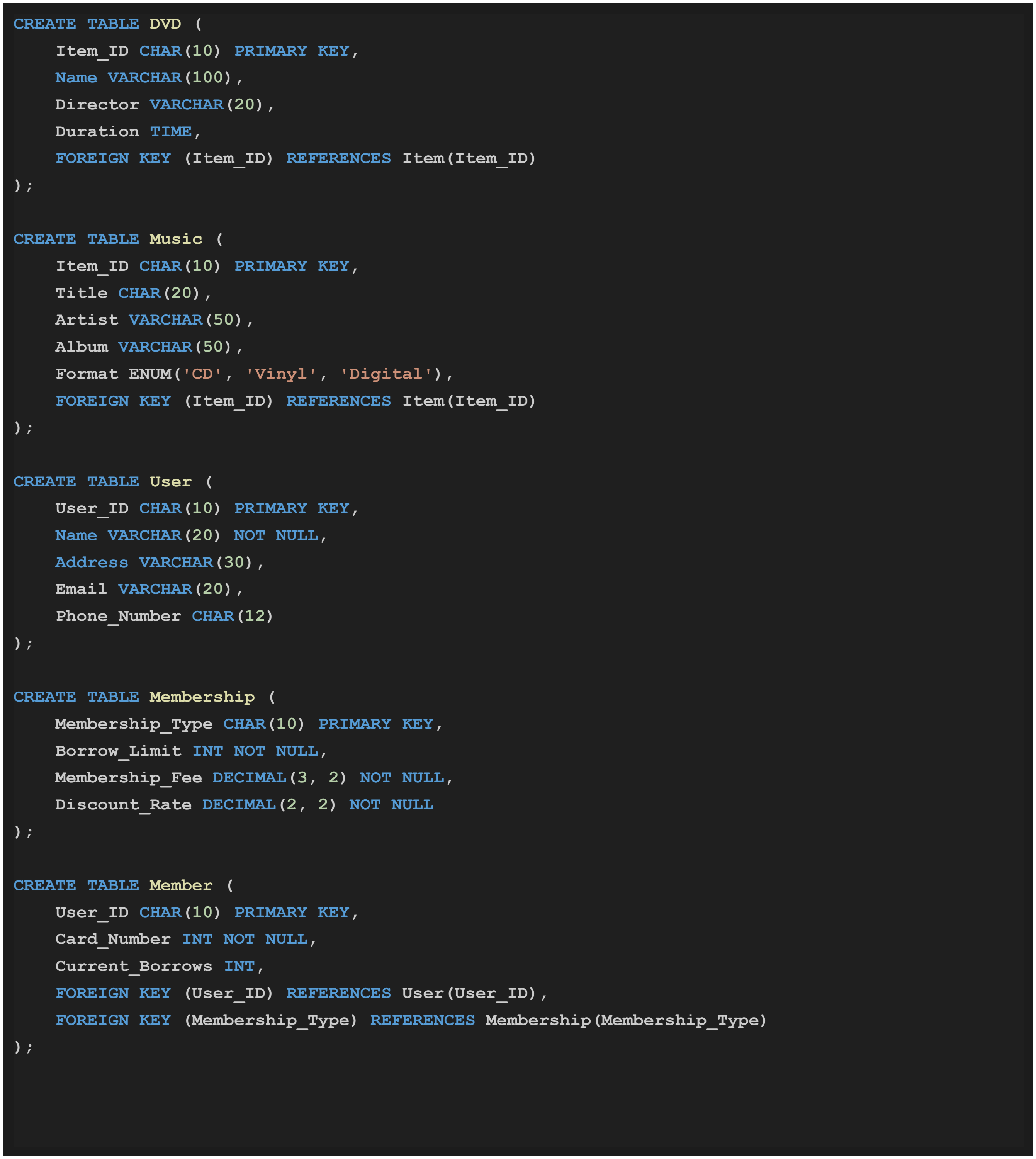
purchase

Attribute Name	Data Type	Domain	Constraints
transaction_id	char(10)	String referencing transaction.transaction_id	primary key, foreign key
amount	decimal(10, 2)	Positive decimal number of xxxxxxxxxx.xx	not null
purchase_type	varchar(10)	String of 10+- characters	
method	enum('Credit Card', 'Debit Card', 'Cash', 'Other')	4 different strings available	not null

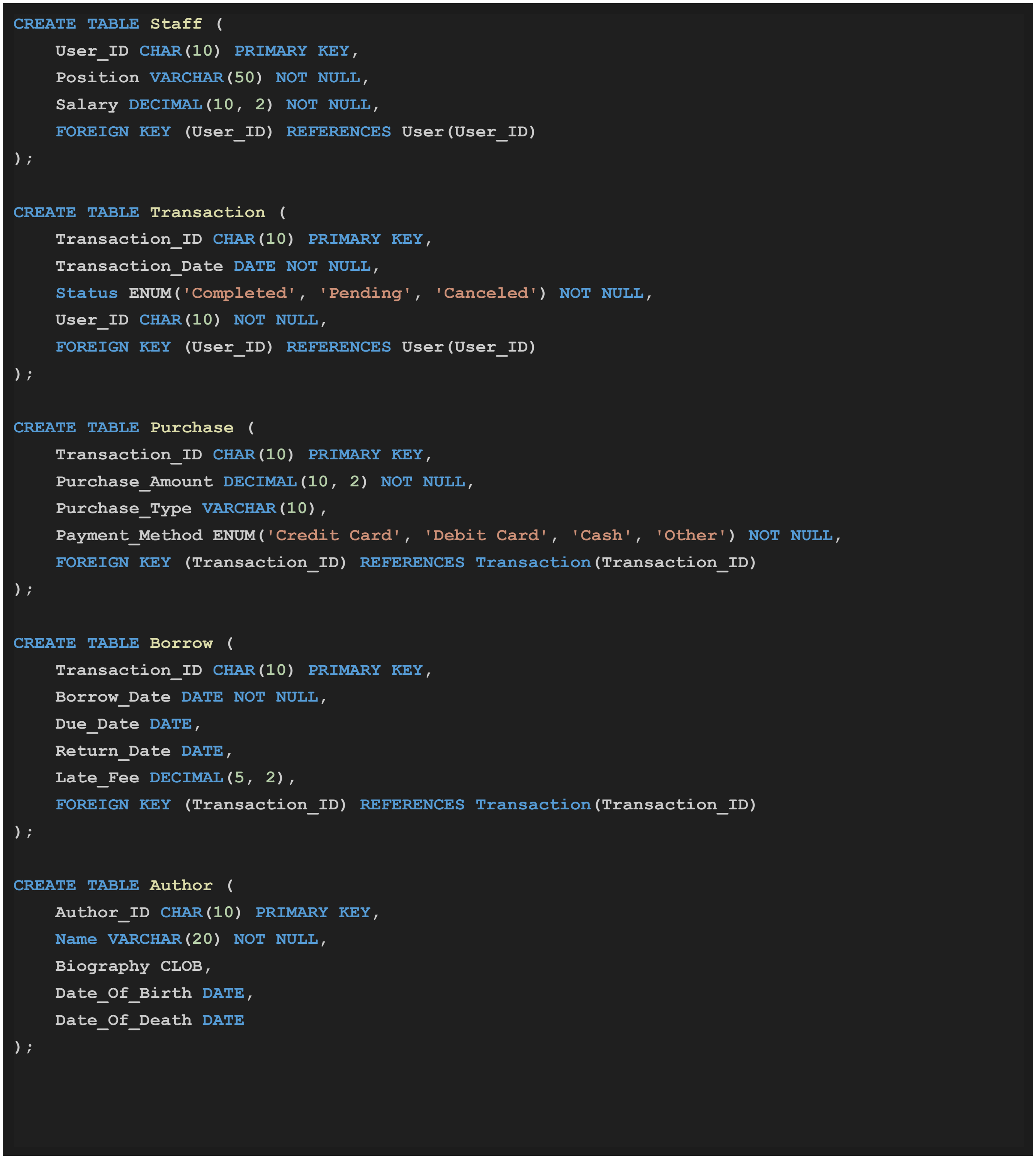
3.3 Generate DDL



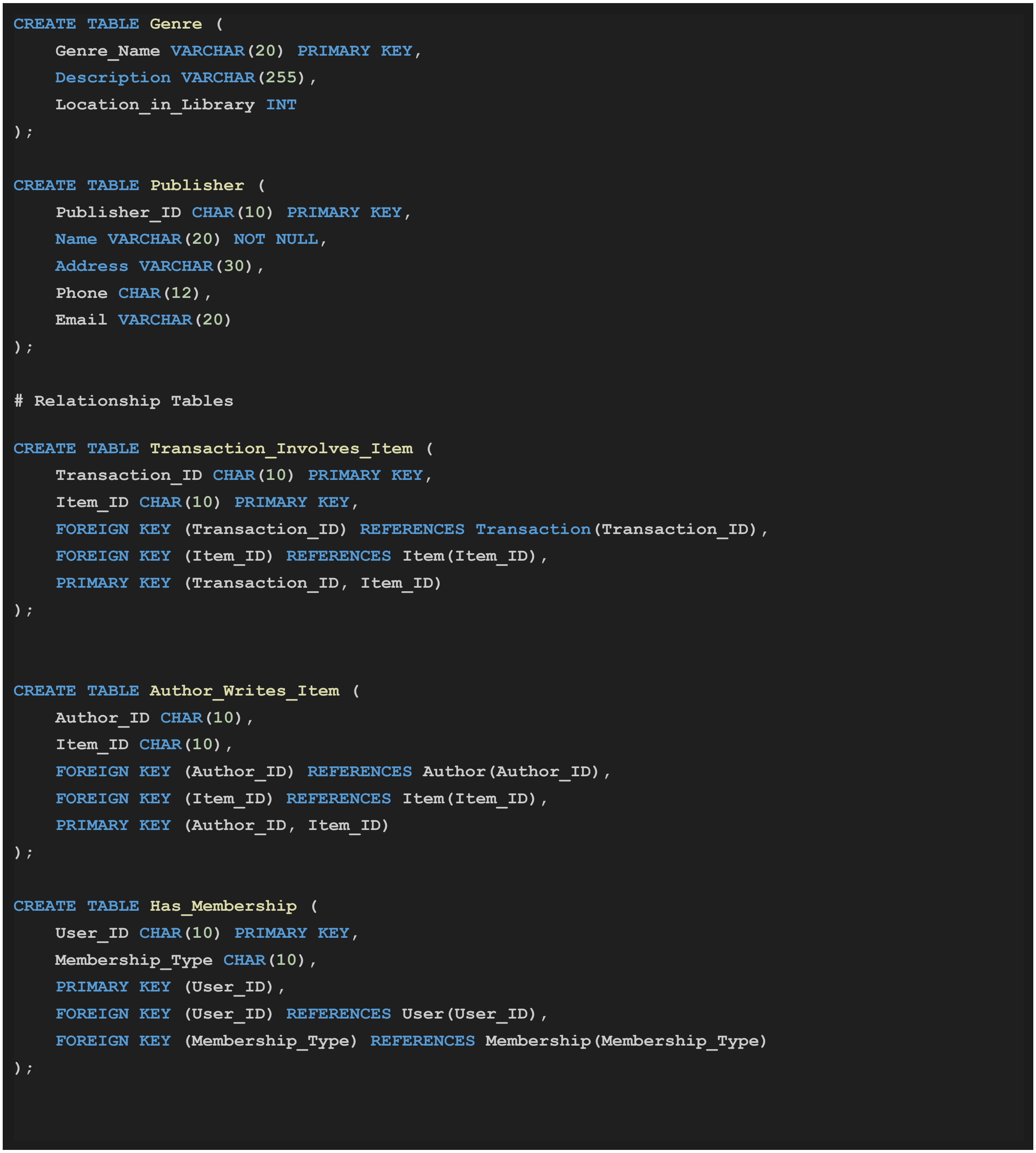
```
CREATE TABLE Item (
    Item_ID CHAR(10) PRIMARY KEY,
    Price INT NOT NULL,
    Year NUMERIC(4, 0),
    Availability_Status ENUM('Available', 'Checked Out', 'Reserved', 'Purchased') NOT NULL,
    Item_Type ENUM('Book', 'Digital Media', 'Magazine', 'DVD', 'Music') NOT NULL
);

CREATE TABLE Book (
    Item_ID CHAR(10) PRIMARY KEY,
    ISBN VARCHAR(13),
    Title VARCHAR(50),
    Subject VARCHAR(20),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID)
);

CREATE TABLE Magazine (
    Item_ID CHAR(10) PRIMARY KEY,
    ISSN VARCHAR(13),
    Name VARCHAR(50),
    Edition INT,
    Publish_Date CHAR(10),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID),
);

CREATE TABLE Digital_Media (
    Item_ID CHAR(10) PRIMARY KEY,
    DOI INT,
    Media_Type VARCHAR(15),
    Release_Date CHAR(10),
    Title CHAR(20),
    Creator CHAR(20),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID)
);
```



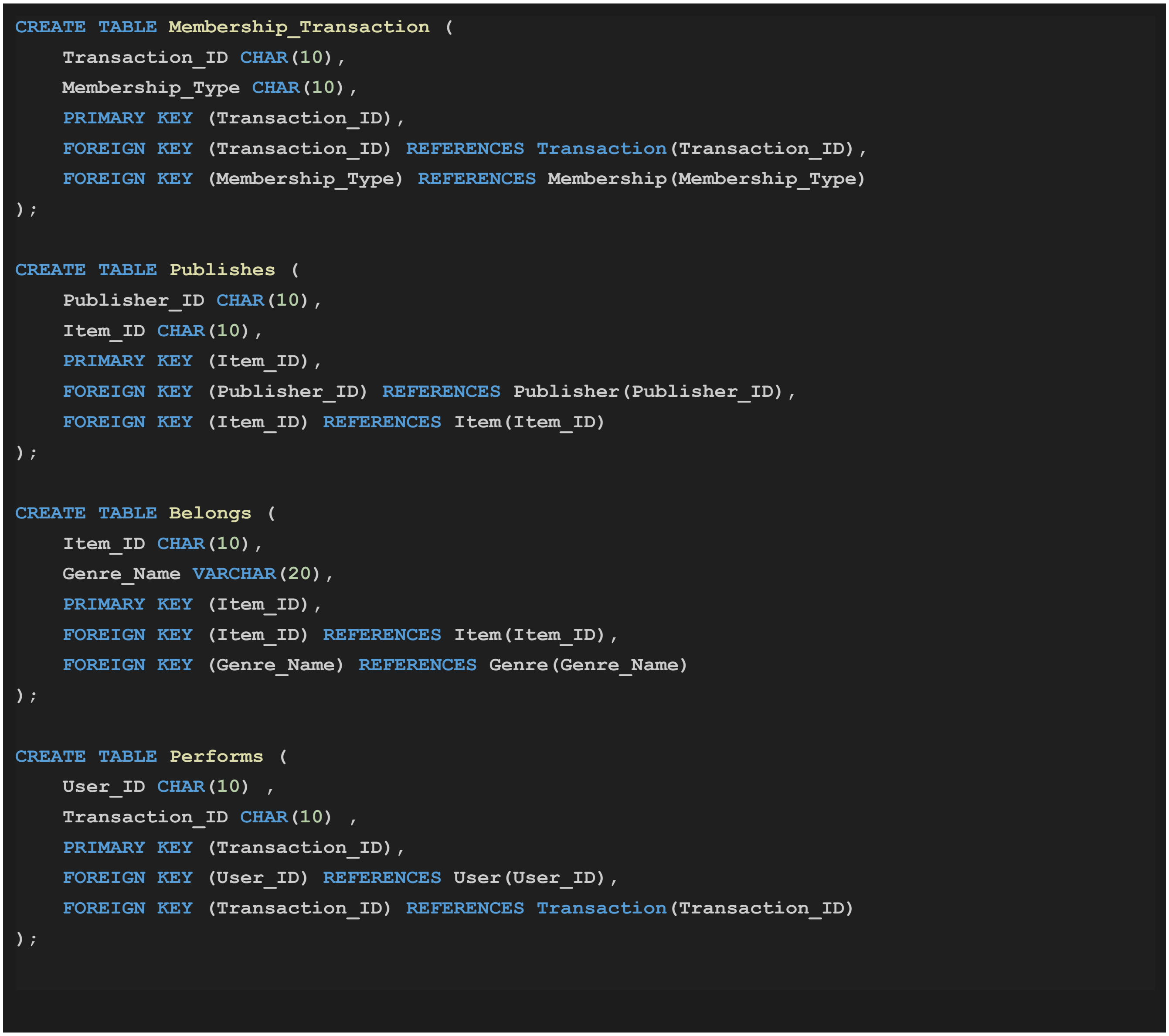
```
CREATE TABLE DVD (
    Item_ID CHAR(10) PRIMARY KEY,
    Name VARCHAR(100),
    Director VARCHAR(20),
    Duration TIME,
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID)
);

CREATE TABLE Music (
    Item_ID CHAR(10) PRIMARY KEY,
    Title CHAR(20),
    Artist VARCHAR(50),
    Album VARCHAR(50),
    Format ENUM('CD', 'Vinyl', 'Digital'),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID)
);

CREATE TABLE User (
    User_ID CHAR(10) PRIMARY KEY,
    Name VARCHAR(20) NOT NULL,
    Address VARCHAR(30),
    Email VARCHAR(20),
    Phone_Number CHAR(12)
);

CREATE TABLE Membership (
    Membership_Type CHAR(10) PRIMARY KEY,
    Borrow_Limit INT NOT NULL,
    Membership_Fee DECIMAL(3, 2) NOT NULL,
    Discount_Rate DECIMAL(2, 2) NOT NULL
);

CREATE TABLE Member (
    User_ID CHAR(10) PRIMARY KEY,
    Card_Number INT NOT NULL,
    Current_Borrows INT,
    FOREIGN KEY (User_ID) REFERENCES User(User_ID),
    FOREIGN KEY (Membership_Type) REFERENCES Membership(Membership_Type)
);
```



```
CREATE TABLE Staff (
    User_ID CHAR(10) PRIMARY KEY,
    Position VARCHAR(50) NOT NULL,
    Salary DECIMAL(10, 2) NOT NULL,
    FOREIGN KEY (User_ID) REFERENCES User(User_ID)
);

CREATE TABLE Transaction (
    Transaction_ID CHAR(10) PRIMARY KEY,
    Transaction_Date DATE NOT NULL,
    Status ENUM('Completed', 'Pending', 'Canceled') NOT NULL,
    User_ID CHAR(10) NOT NULL,
    FOREIGN KEY (User_ID) REFERENCES User(User_ID)
);

CREATE TABLE Purchase (
    Transaction_ID CHAR(10) PRIMARY KEY,
    Purchase_Amount DECIMAL(10, 2) NOT NULL,
    Purchase_Type VARCHAR(10),
    Payment_Method ENUM('Credit Card', 'Debit Card', 'Cash', 'Other') NOT NULL,
    FOREIGN KEY (Transaction_ID) REFERENCES Transaction(Transaction_ID)
);

CREATE TABLE Borrow (
    Transaction_ID CHAR(10) PRIMARY KEY,
    Borrow_Date DATE NOT NULL,
    Due_Date DATE,
    Return_Date DATE,
    Late_Fee DECIMAL(5, 2),
    FOREIGN KEY (Transaction_ID) REFERENCES Transaction(Transaction_ID)
);

CREATE TABLE Author (
    Author_ID CHAR(10) PRIMARY KEY,
    Name VARCHAR(20) NOT NULL,
    Biography CLOB,
    Date_Of_Birth DATE,
    Date_Of_Death DATE
);
```



```
CREATE TABLE Genre (
    Genre_Name VARCHAR(20) PRIMARY KEY,
    Description VARCHAR(255),
    Location_in_Library INT
);

CREATE TABLE Publisher (
    Publisher_ID CHAR(10) PRIMARY KEY,
    Name VARCHAR(20) NOT NULL,
    Address VARCHAR(30),
    Phone CHAR(12),
    Email VARCHAR(20)
);

# Relationship Tables

CREATE TABLE Transaction_Involves_Item (
    Transaction_ID CHAR(10) PRIMARY KEY,
    Item_ID CHAR(10) PRIMARY KEY,
    FOREIGN KEY (Transaction_ID) REFERENCES Transaction(Transaction_ID),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID),
    PRIMARY KEY (Transaction_ID, Item_ID)
);


CREATE TABLE Author_Writes_Item (
    Author_ID CHAR(10),
    Item_ID CHAR(10),
    FOREIGN KEY (Author_ID) REFERENCES Author(Author_ID),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID),
    PRIMARY KEY (Author_ID, Item_ID)
);

CREATE TABLE Has_Membership (
    User_ID CHAR(10) PRIMARY KEY,
    Membership_Type CHAR(10),
    PRIMARY KEY (User_ID),
    FOREIGN KEY (User_ID) REFERENCES User(User_ID),
    FOREIGN KEY (Membership_Type) REFERENCES Membership(Membership_Type)
);
```



```
CREATE TABLE Membership_Transaction (
    Transaction_ID CHAR(10),
    Membership_Type CHAR(10),
    PRIMARY KEY (Transaction_ID),
    FOREIGN KEY (Transaction_ID) REFERENCES Transaction(Transaction_ID),
    FOREIGN KEY (Membership_Type) REFERENCES Membership(Membership_Type)
);

CREATE TABLE Publishes (
    Publisher_ID CHAR(10),
    Item_ID CHAR(10),
    PRIMARY KEY (Item_ID),
    FOREIGN KEY (Publisher_ID) REFERENCES Publisher(Publisher_ID),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID)
);

CREATE TABLE Belongs (
    Item_ID CHAR(10),
    Genre_Name VARCHAR(20),
    PRIMARY KEY (Item_ID),
    FOREIGN KEY (Item_ID) REFERENCES Item(Item_ID),
    FOREIGN KEY (Genre_Name) REFERENCES Genre(Genre_Name)
);

CREATE TABLE Performs (
    User_ID CHAR(10) ,
    Transaction_ID CHAR(10) ,
    PRIMARY KEY (Transaction_ID),
    FOREIGN KEY (User_ID) REFERENCES User(User_ID),
    FOREIGN KEY (Transaction_ID) REFERENCES Transaction(Transaction_ID)
);
```