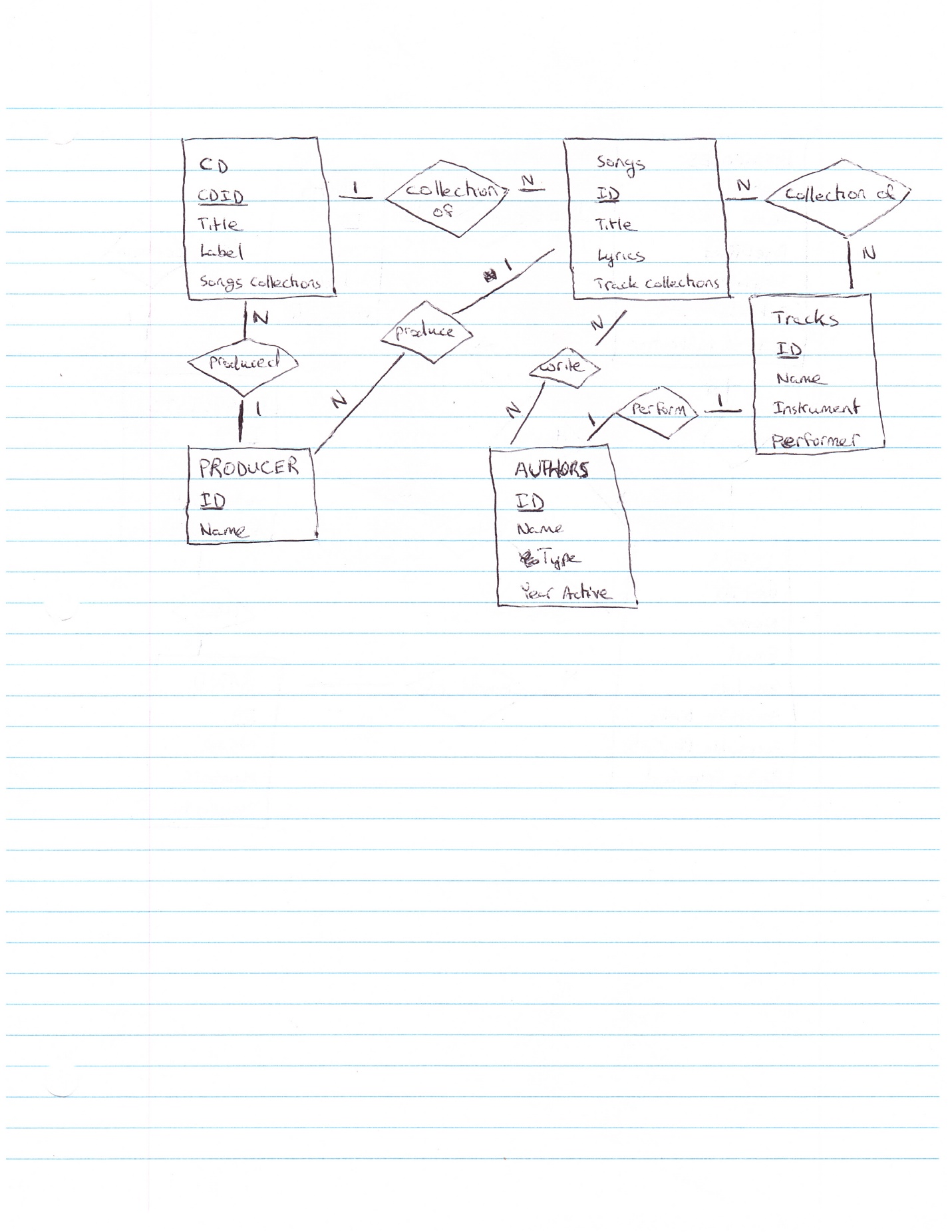
Assignment 2

## Problem 1

1. **Assumptions**: A CD contains CD title, artists, release date, label, producer and collection of songs. Therefore, a CD has a relationship with songs, authors, and producer. A song contains title, author, publisher and lyrics. A song is made up of different tracks. Therefore, a song has a relationship with author, producer, and tracks.
2. **ER Diagram**:



1. **Schema**:

**Table**: CD

**Primary Key**: CD ID (TEXT)

**Foreign Key**: Songs.ID, Authors.ID, Producers.ID

**Other Fields**: Title (TEXT), Label (TEXT), Songs Collections (LONG TEXT)

**Relationships**: CD –> Songs = 1: N, CD –> Producers = 1: N

**Table**: Songs

**Primary Key**: ID (TEXT)

**Foreign Key**: Tracks.ID, Authors.ID, Producers.ID

**Other Fields**: Title (TEXT), Lyrics (LONG TEXT), Track Collections (LONG TEXT)

**Relationships**: Songs –> Tracks = N: N, Songs –> Authors = N: N, Songs –> Producers = 1: N

**Table**: Authors

**Primary Key**: ID (TEXT)

**Foreign Key**: N/A

**Other Fields**: Name (TEXT), Type (TEXT) [Band, Musician, Various],

Year Active (DATE)

**Relationships**: Authors –> Songs = N: N, Authors –> Tracks = 1:1

**Table**: Producer

**Primary Key**: ID (TEXT)

**Foreign Key**: N/A

**Other Fields**: Name (TEXT)

**Relationships**: Producers –> CD = N: 1, Producers –> Songs = N: 1

**Table**: Tracks

**Primary Key**: ID (TEXT)

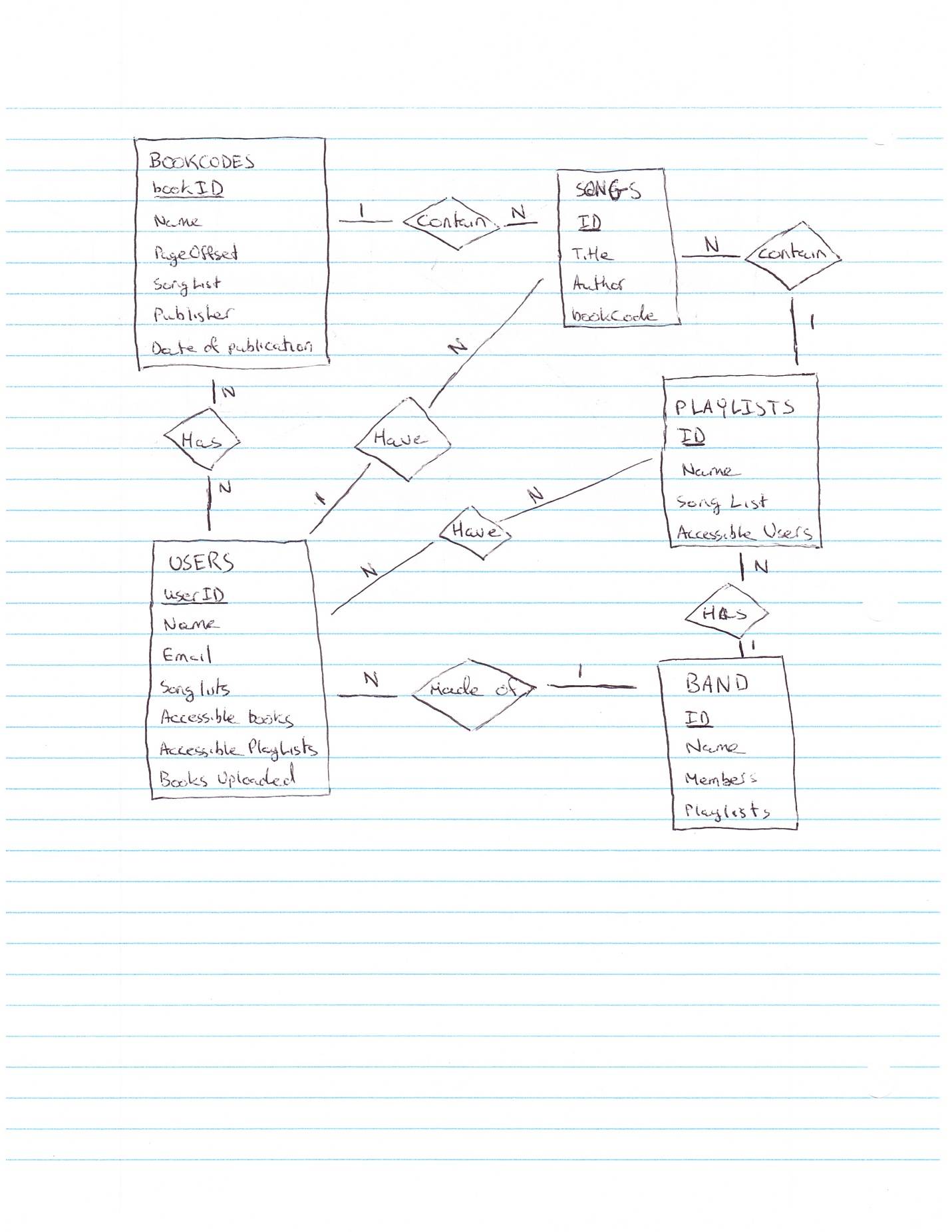
**Foreign Key**: Authors.ID

**Other Fields**: Name (TEXT), Instrument (TEXT), Performer (TEXT)

**Relationships**: Tracks –> Songs = N: N, Tracks –> Authors = 1:1

## Problem 2

1. **Assumptions**: Users can upload many books. There can only be one unique copy of the book. Each book should have an index of the offset information. Each book contains many songs. Each song contains title, book code, and page number. Users can have many playlists. Each playlist should have many songs in a particular order. Each playlist should be able to have non-indexed songs. Each playlist can have many users.
2. **ER Diagram**:



1. **Schema**:

**Table**: BookCodes

**Primary Key**: bookID (TEXT)

**Foreign Key**: Songs.ID

**Other Fields**: Name (TEXT), PageOffset (TEXT), Lists of Songs (TEXT), Publisher (TEXT), Date of publication (DATE)

**Relationships**: BookCodes –> Songs = 1: N, BookCodes –> Users = N: N

**Table**: Users

**Primary Key**: userID (TEXT)

**Foreign Key**: PlayLists.ID, Songs.ID, BookCodes.bookID

**Other Fields**: Name (TEXT), email (TEXT), password (TEXT), lists of songs (TEXT), Books Uploaded (TEXT), Accessible Books (TEXT), Accessible Playlists (TEXT)

**Relationships**: Users –> PlayLists = N: N, Users –> Songs = 1: N, Users –> BookCodes = N: N, Users –> Band = N: 1

**Table**: PlayLists

**Primary Key**: ID (TEXT)

**Foreign Key**: Users.userID, Songs.ID

**Other Fields**: Name (TEXT), List of songs (TEXT), Accessible Users (TEXT)

**Relationships**: PlayLists –> Users = N: N, PlayLists –> Songs: 1: N, PlayLists –> Band = N: 1

**Table**: Songs

**Primary Key**: ID (TEXT)

**Foreign Key**: PlayLists.ID, Users.userID, BookCodes.bookID

**Other Fields**: Title (TEXT), Author (TEXT), bookCode (TEXT)

**Relationships**: Songs –> BookCodes = N: 1, Songs –> Users = N: 1, Songs – > PlayLists = N: 1

**Table**: Band

**Primary Key**: ID (TEXT)

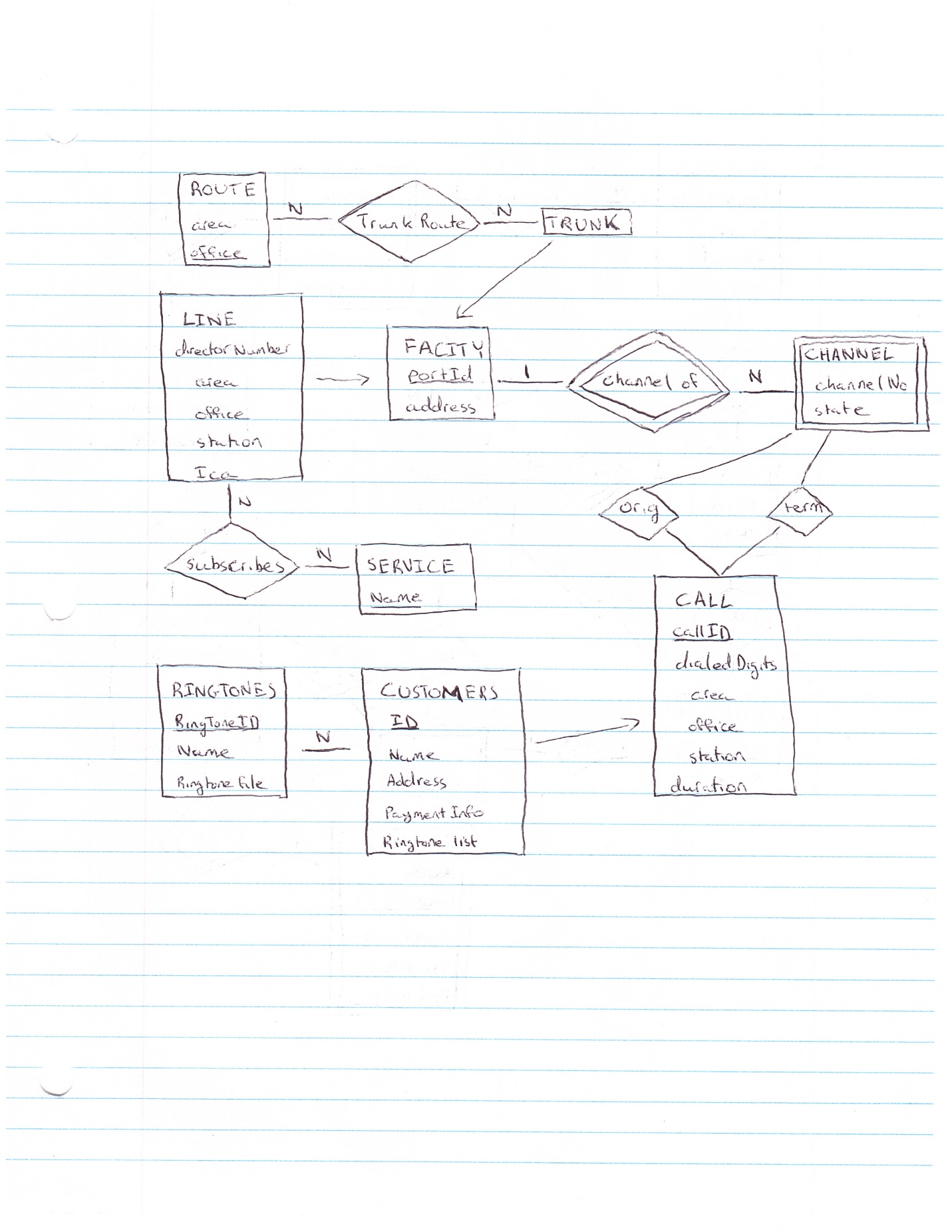
**Foreign Key**: Users.userID, PlayLists.ID

**Other Fields**: Name (TEXT), Members (TEXT), Playlists (TEXT)

**Relationships**: Band –> Users = 1: N, Band – > PlayLists = 1: N

## Problem 3

1. **Assumptions**: Each customer can have many distinctive ring tones. Each ring tone can have many customers. Customers will have to maintain a list of all the ring tones of their friends and family in order to properly distinguish between them.
2. **ER Diagram**:



1. **Schema**:

**Table**: Customers (Inherits from Call)

**Primary Key**: ID

**Foreign Key**:

**Other Fields**: Name (TEXT), Address (TEXT), Payment Info (TEXT), RingTone List (TEXT)

**Relationships**: Customers –> RingTones = N: N

**Table**: RingTones

**Primary Key**: Foreign Key + RingToneID (TEXT)

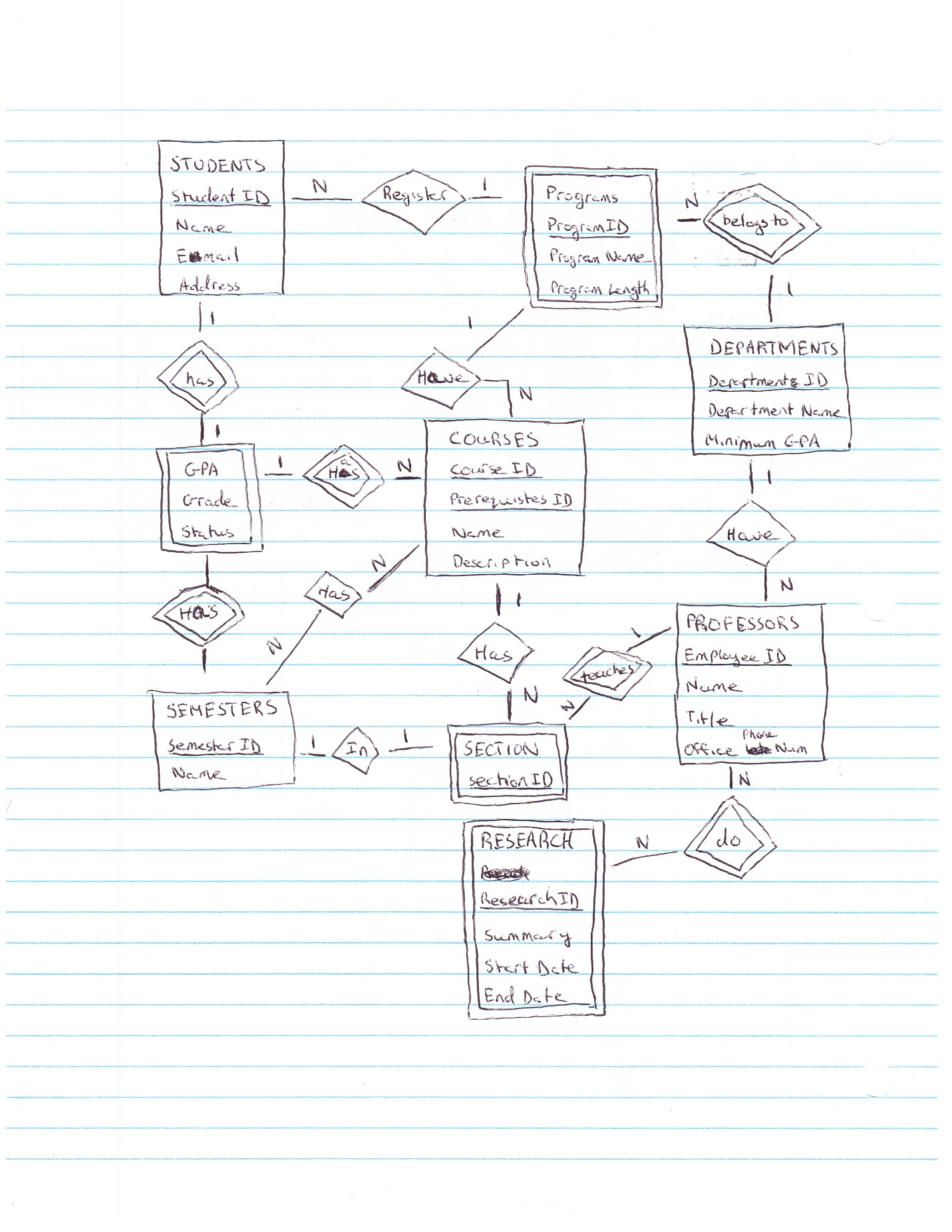
**Foreign Key**: Customers.ID

**Other Fields**: Name (TEXT), Ringtone file (TEXT)

**Relationships**: RingTones –> Customers = N: N

## Problem 4

1. **Assumptions**: Students are registered to a program, and have a GPA. A GPA has one student and many courses. A program belongs to a department and has many courses. A department has many programs, and many professors. Courses have many sections and many semesters. A section has a professor and a semester. A professors has zero or more researches.
2. **ER Diagram**:



1. **Schema**:

**Table:** Students

**Primary Key:** Student ID (text)

**Foreign Key:** Programs.Primary Key

**Other Fields:** Name (text), Email (text), Address (text)

**Relationship:** Students –> Programs = N: 1, Students –> GPA = 1: 1

**Table:** GPA

**Foreign Key:** Students.Student ID, Course.Course ID, Semester.Semester ID

**Primary Key:** Foreign Key

**Other Fields:** Grade (real), Status (text)

**Relationship:** GPA –> Students = 1: 1, GPA –> Courses = 1: N, GPA –> Semester = N: 1

**Table:** Departments

**Primary Key:** Department ID (text)

**Other Fields:** Department Name (text), Minimum GPA required (real)

**Relationship:** Departments –> Programs = 1: N, Departments –> Professors = 1: N

**Table:** Programs

**Foreign Key:** Departments.Department ID

**Primary Key:** Foreign Key + Program ID (text)

**Other Fields:** Program Name (text), Length of program (integer)

**Relationship:** Programs –> Students = 1: N, Programs –> Departments = N: 1, Programs –> Courses = 1: N

**Table:** Courses

**Primary Key:** Course ID (text), Prerequisites ID (text)

**Foreign Key:** Programs.Primary Key

**Other Fields**: Name (text), Description (text)

**Relationship:** Courses –> Programs = N: 1, Courses –> GPA = N: 1, Courses –> Section = 1: N, Course –> Semester = N: N

**Table:** Section

**Foreign Key:** Courses.Course ID, Professors.Emplyee ID, Semester.Semester ID

**Primary Key:** Courses.Course ID, Section ID (text)

**Relationship:** Section –> Courses = N: 1, Section –> Professors = 1: N, Section –> Semester = 1: 1

**Table:** Semester

**Primary Key:** Semester ID (text)

**Other Key:** Name (text)

**Relationship:** Semester –> GPA = 1: N, Semester –> Section = 1: 1, Semester –> Course = N: N

**Table:** Professors

**Primary Key:** Employee ID (text) [Not Null]

**Foreign Key:** Departments.Department ID

**Other Fields:** Name (text), Title (text) Office Phone Number (text)

**Relationship:** Professors –> Departments = N: 1, Professors –> Sections = 1: N, Professors –> Research = N: N

**Table:** Research

**Foreign Key:** Professors.Employee ID

**Primary Key:** Foreign Key + Research ID (text)

**Other Fields:** Name (text), Summary (text), Start Date (Date), End Date (Date)

**Relationship:** Research –> Professors = N: N

1. **Scenario**: The database that I have chosen is a representation of the university database. This database will provide the users with information about the students, departments, courses, and professors in that university. Users would be able to query the database and receive information about when a course is available, the different sections available if more than one, and the professor giving the lectures for each section. Users can also query information about the prerequisites for the course, the student’s GPA, the different departments at the university, and information about the professors.