RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Mechanical and Industrial Engineering

Course Number	**COE 848**
Course Title	**Fundamentals of Data Engineering**
Semester/Year	2022 Winter
Instructor	**Dr. Faezeh Ensan**
Teaching Assistant	**Shirin Seyedsalehi**

Lab Report No.	**3**

Report Title	**Lab 3 Report**

Section No.	**2**
Submission Date	**February 27th, 2022**
Due Date	**February 27th, 2022**

Name	Student ID	Signature*
Yasar Zaman	*xxxx13595*	Y.Z

(Note: remove the first 4 digits from your student ID)

ERD Diagram

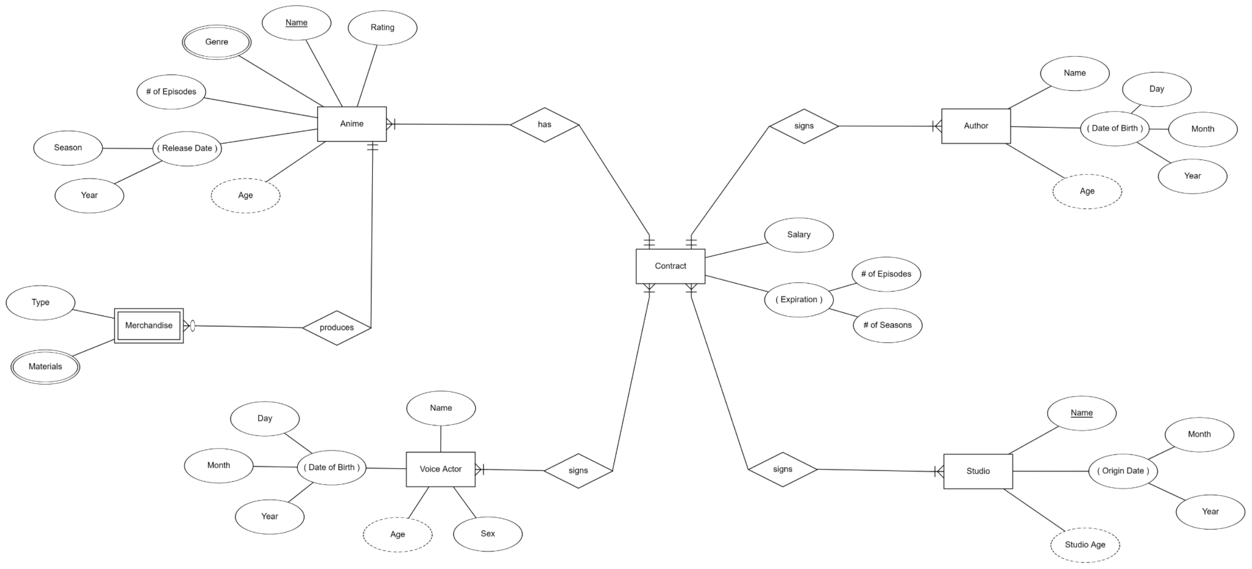
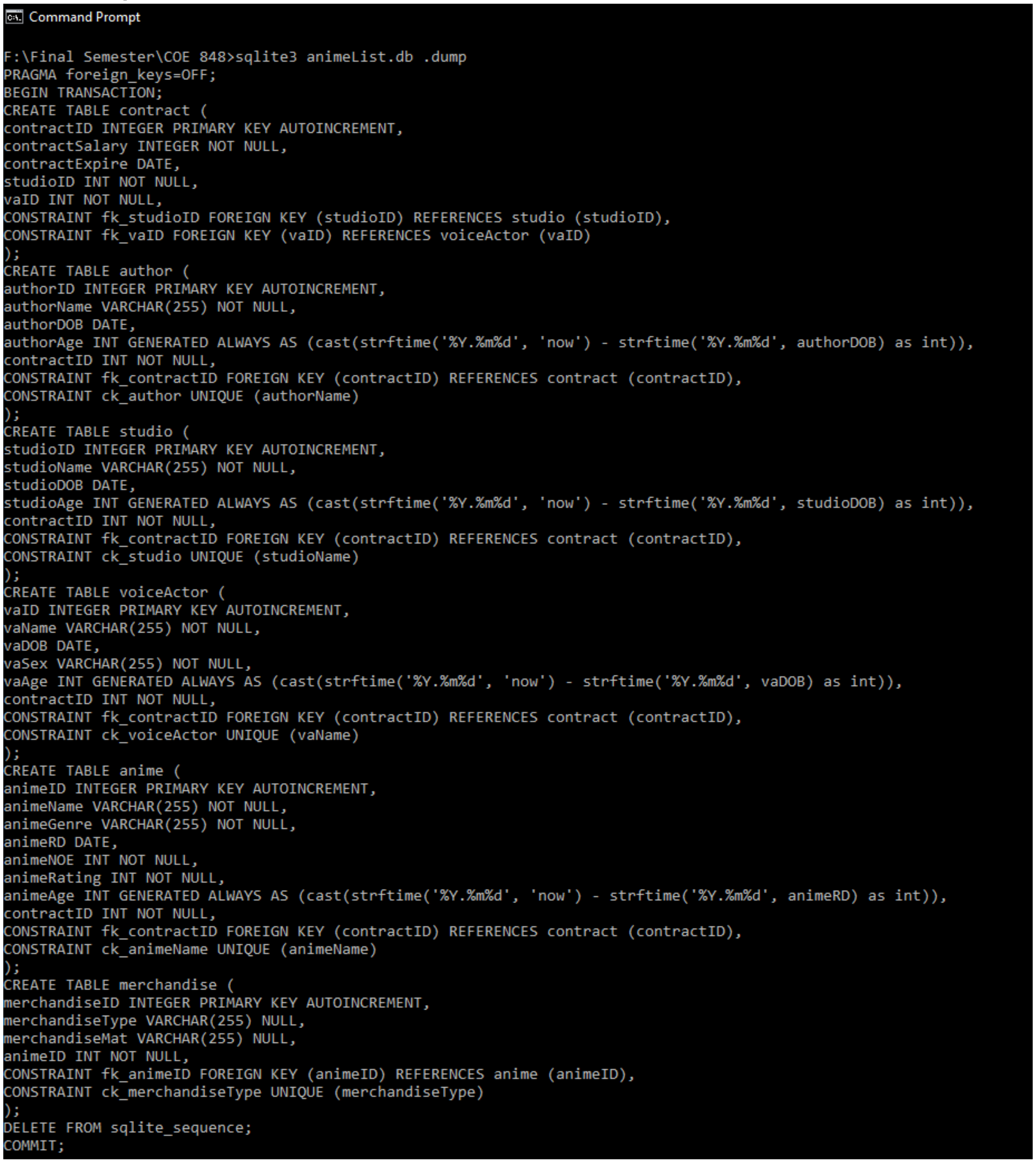


Figure 1: ERD of Database

Table Dump



```
Command Prompt

F:\Final Semester\COE 848>sqlite3 animeList.db .dump
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE contract (
contractID INTEGER PRIMARY KEY AUTOINCREMENT,
contractSalary INTEGER NOT NULL,
contractExpire DATE,
studioID INT NOT NULL,
vaID INT NOT NULL,
CONSTRAINT fk_studioID FOREIGN KEY (studioID) REFERENCES studio (studioID),
CONSTRAINT fk_vaID FOREIGN KEY (vaID) REFERENCES voiceActor (vaID)
);
CREATE TABLE author (
authorID INTEGER PRIMARY KEY AUTOINCREMENT,
authorName VARCHAR(255) NOT NULL,
authorDOB DATE,
authorAge INT GENERATED ALWAYS AS (cast(strftime('%Y.%m%d', 'now') - strftime('%Y.%m%d', authorDOB) as int)),
contractID INT NOT NULL,
CONSTRAINT fk_contractID FOREIGN KEY (contractID) REFERENCES contract (contractID),
CONSTRAINT ck_author UNIQUE (authorName)
);
CREATE TABLE studio (
studioID INTEGER PRIMARY KEY AUTOINCREMENT,
studioName VARCHAR(255) NOT NULL,
studioDOB DATE,
studioAge INT GENERATED ALWAYS AS (cast(strftime('%Y.%m%d', 'now') - strftime('%Y.%m%d', studioDOB) as int)),
contractID INT NOT NULL,
CONSTRAINT fk_contractID FOREIGN KEY (contractID) REFERENCES contract (contractID),
CONSTRAINT ck_studio UNIQUE (studioName)
);
CREATE TABLE voiceActor (
vaID INTEGER PRIMARY KEY AUTOINCREMENT,
vaName VARCHAR(255) NOT NULL,
vaDOB DATE,
vaSex VARCHAR(255) NOT NULL,
vaAge INT GENERATED ALWAYS AS (cast(strftime('%Y.%m%d', 'now') - strftime('%Y.%m%d', vaDOB) as int)),
contractID INT NOT NULL,
CONSTRAINT fk_contractID FOREIGN KEY (contractID) REFERENCES contract (contractID),
CONSTRAINT ck_voiceActor UNIQUE (vaName)
);
CREATE TABLE anime (
animeID INTEGER PRIMARY KEY AUTOINCREMENT,
animeName VARCHAR(255) NOT NULL,
animeGenre VARCHAR(255) NOT NULL,
animeRD DATE,
animeNOE INT NOT NULL,
animeRating INT NOT NULL,
animeAge INT GENERATED ALWAYS AS (cast(strftime('%Y.%m%d', 'now') - strftime('%Y.%m%d', animeRD) as int)),
contractID INT NOT NULL,
CONSTRAINT fk_contractID FOREIGN KEY (contractID) REFERENCES contract (contractID),
CONSTRAINT ck_animeName UNIQUE (animeName)
);
CREATE TABLE merchandise (
merchandiseID INTEGER PRIMARY KEY AUTOINCREMENT,
merchandiseType VARCHAR(255) NULL,
merchandiseMat VARCHAR(255) NULL,
animeID INT NOT NULL,
CONSTRAINT fk_animeID FOREIGN KEY (animeID) REFERENCES anime (animeID),
CONSTRAINT ck_merchandiseType UNIQUE (merchandiseType)
);
DELETE FROM sqlite_sequence;
COMMIT;
```

Figure 2: .dump of Database on Command Prompt

<u>Explanation</u>
Contract:
As shown above in **Figure 2** there are a total of six tables for each entity. The very first table that I created was the '*contract'* table for the entity *Contract*. From **Figure 1** we can see that the *Contract* Entity is in relation with four other entities; that is there can be at least one contract per entity. This means that we will have to reference the contract id in each of those four entities. The attributes that make up this table will be the contractID which will be the primary key and the contractSalary and contractExpire; holding the salary of the contract and the date of expiration respectively. The foreign keys will be the studioID and vaID (voice actor ID). This way if there is a new entry for the studio or voice actor, since according to **Figure 1**, they can have at least 1 contract, they will have another ID.

Table Name: contract
Primary Key: contractID
Foreign Key: studioID, vaID

Author:
In this table, we will be working with the *author* entity. Since there can only be one author per contract as shown in the relationship between *Author* and *Contract* in **Figure 1**, we need to create a foreign keys so that we can reference the contract that the author is signed to. The primary key for this table will be the authorID, and the foreign key will be the contractID that is referenced from the table 'contract.' The attributes in this table will be the authorName, authorDOB, authorAge which is meant to represent the author's name, date of birth, and age respectively. To generate the Age, we can use the cast function and subtracted the current date by the date of birth. Though in the ERD in **Figure 1** it does not show that the name for the author and voice actor entities are unique, to make this database more easy to handle in the beginning, we can change it so that they are unique. This means that the author's name will be a candidate key.

Table Name: author
Primary Key: authorID
Foreign Key: contractID references contract(contractID)
Candidate Key: authorName

Studio:
In this table, we will be working with the *studio* entity. There is at least one studio per contract, this would mean that we would need to reference the studioID in the contract table. The primary key will be the studioID and will have the foreign key being the contractID; this will allow us to know when the studio will finish it's contract with regards to that specific contract. The attributes for this table will be the studioName, studioDOB, studioAge which is meant to represent the studio's name, date of birth, and the age respectively. As previously mentioned, the age can be found by using the cast function and subtracting the current date by the date of birth. The candidate key will be the studio's name.

Table Name: studio
Primary Key: studioID
Foreign Key: contractID references contract(contractID)
Candidate Key: studioName

Voice Actor:
In this table, we will be working with the *voiceActor* entity. There is at least one voice actor per contract, which means that we need to reference the voice actor ID (vaID) in the contract table. The primary key will be the vaID and the foreign key will be the contractID for the reasons mentioned earlier. The attributes for this table will include the vaName, vaDOB, and vaAge which is meant to represent the voice actor's name, date of birth and the age respectively. The candidate key will be the voice actor's name.

Table Name: voiceActor
Primary Key: vaID
Foreign Key: contractID references contract(contractID)
Candidate Key: vaName

Anime:
In this table, we will be working with the *anime* entity. There is only one anime per contract so we need to reference the contractID. The primary key will be the animeID and the foreign key will be the contractID. The attributes for this table will consist of animeName, animeGenre, animeRD, animeNOE, animeRating, and animeAge which is meant to represent the anime name, anime genre, anime release date, anime number of episodes, anime rating and the age of the anime respectively. The candidate key will be the anime's name.

Table Name: anime
Primary Key: animeID
Foreign Key: contractID references contract(contractID)
Candidate Key: animeName

Merchandise:
In this table, we will be working with the *merchandise* entity. Since this is an optional (at least 0) relationship with anime, it will reference the animeID as the foreign key. The attributes for this table will be the merchandiseType and merchandiseMat which is meant to represent the merchandise Type (toy car, doll, etc) and the merchandise material that was used (cloth, wool, plastic).

Table Name: merchandise
Primary Key: merchandiseID
Foreign Key: animeID references anime (animeID)