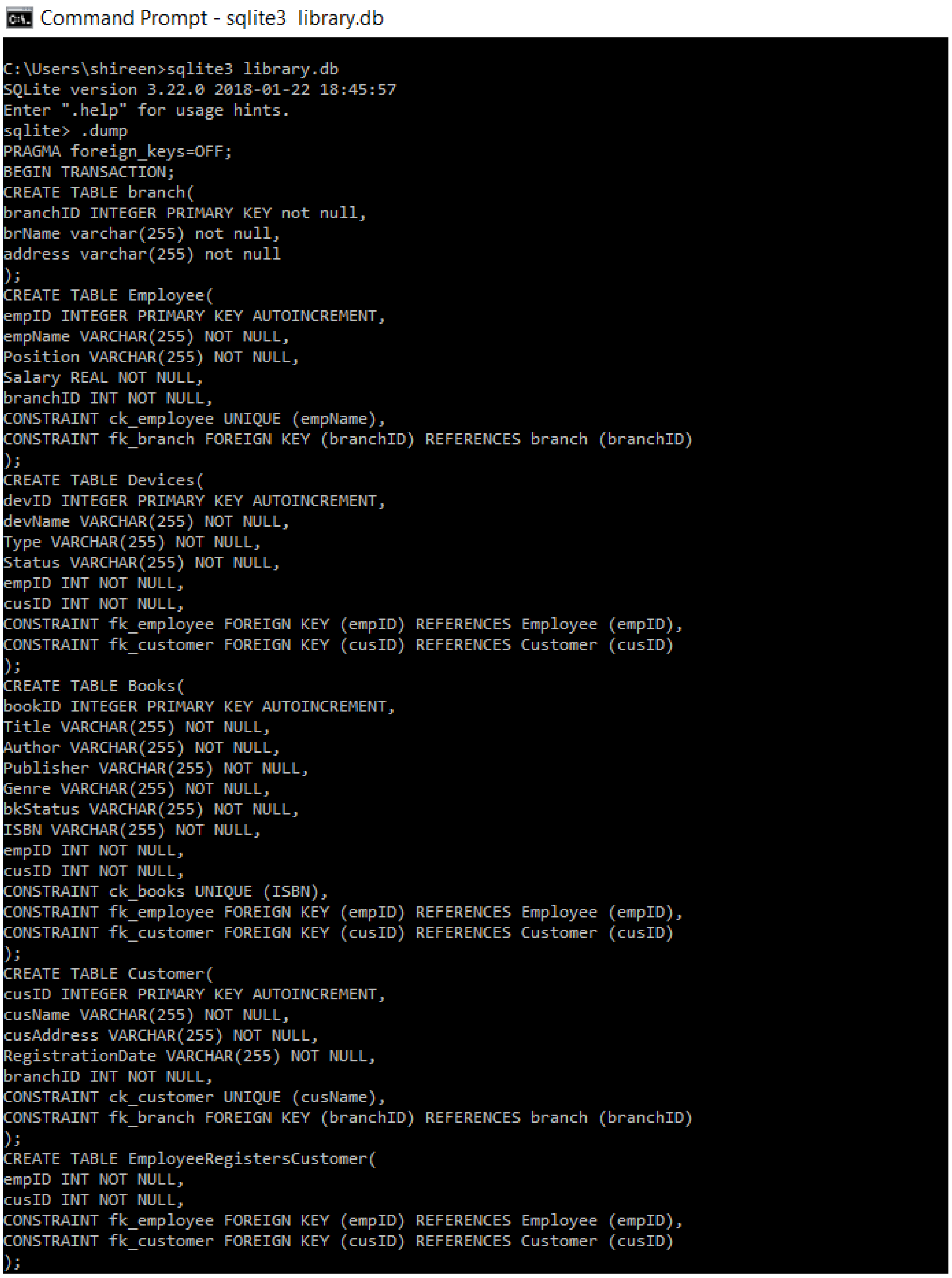
Shireen Taha
#500581584

Lab 3 – Database Design

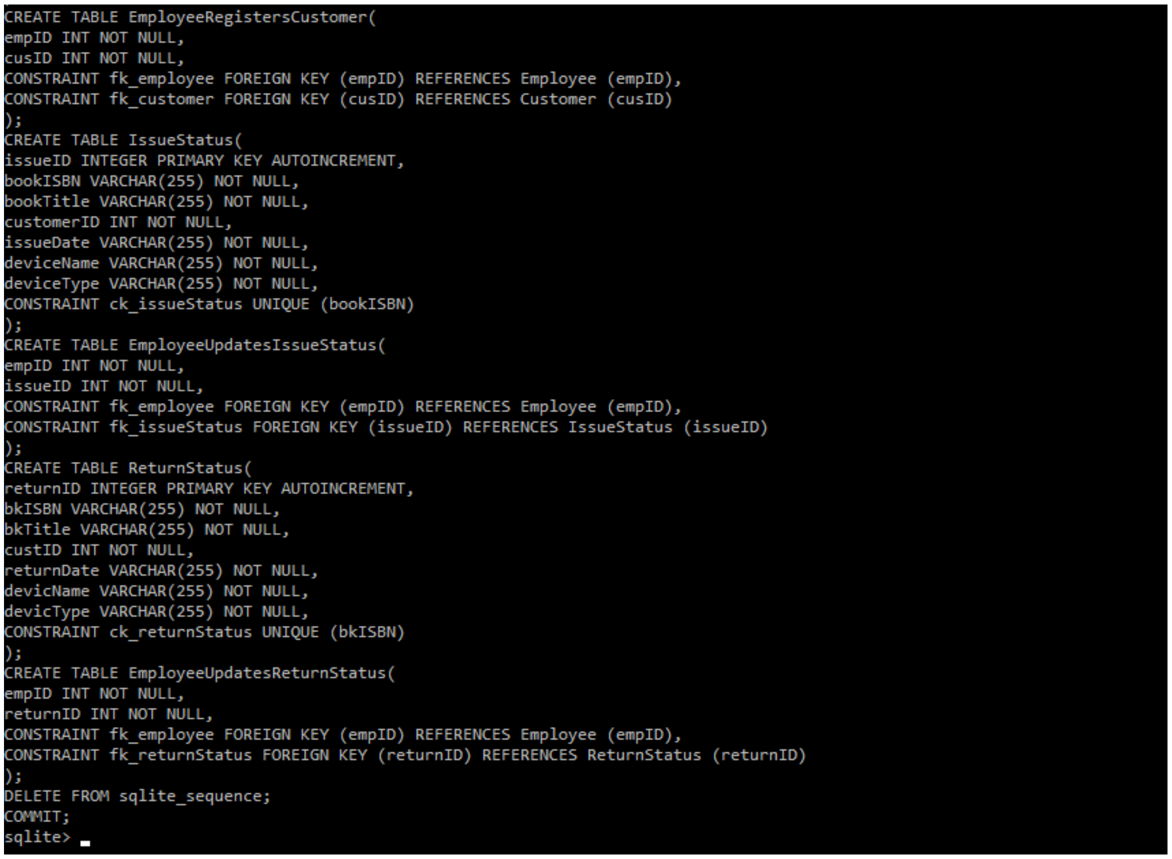
The screenshots shown below illustrate the structure of the tables.



Command Prompt - sqlite3 library.db

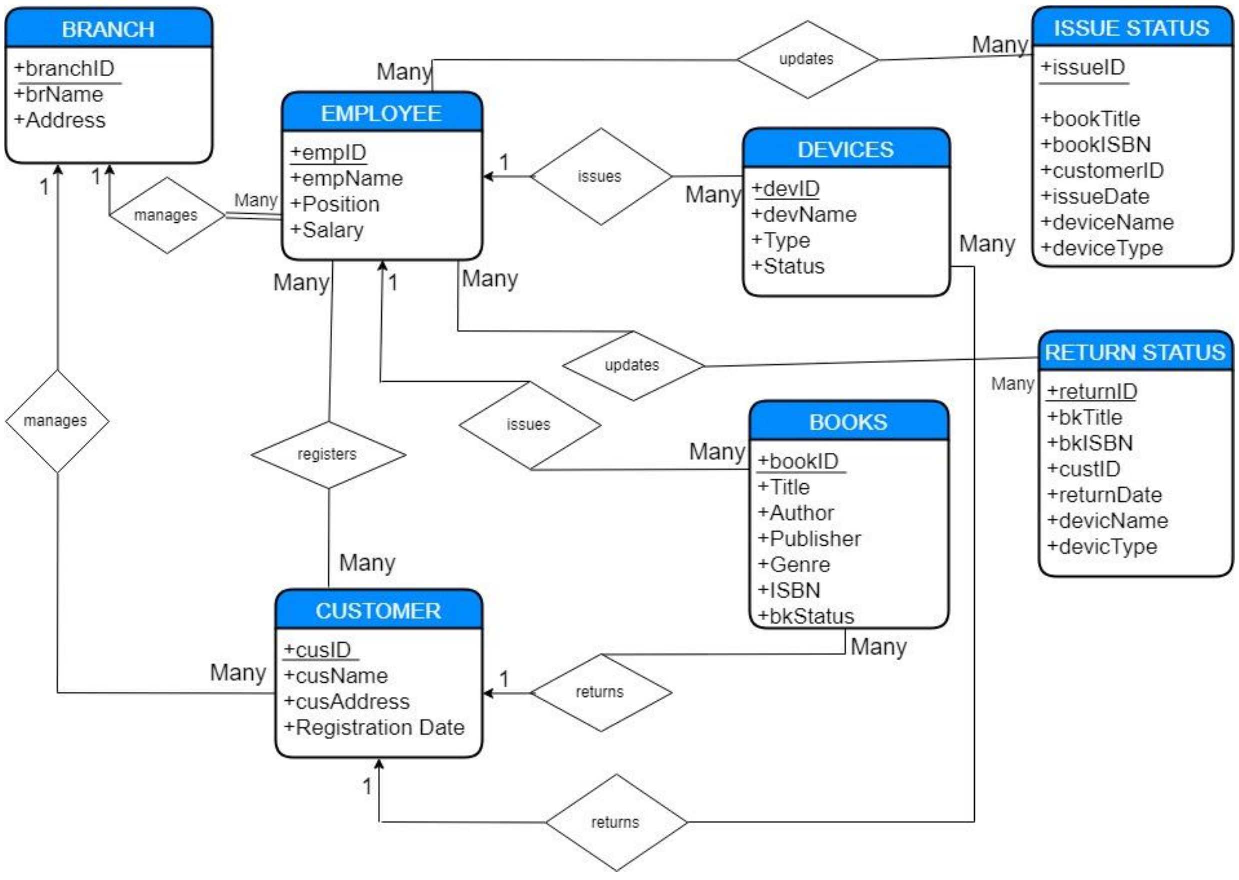
```
C:\Users\shireen>sqlite3 library.db
SQLite version 3.22.0 2018-01-22 18:45:57
Enter ".help" for usage hints.
sqlite> .dump
PRAGMA foreign_keys=OFF;
BEGIN TRANSACTION;
CREATE TABLE branch(
branchID INTEGER PRIMARY KEY not null,
brName varchar(255) not null,
address varchar(255) not null
);
CREATE TABLE Employee(
empID INTEGER PRIMARY KEY AUTOINCREMENT,
empName VARCHAR(255) NOT NULL,
Position VARCHAR(255) NOT NULL,
Salary REAL NOT NULL,
branchID INT NOT NULL,
CONSTRAINT ck_employee UNIQUE (empName),
CONSTRAINT fk_branch FOREIGN KEY (branchID) REFERENCES branch (branchID)
);
CREATE TABLE Devices(
devID INTEGER PRIMARY KEY AUTOINCREMENT,
devName VARCHAR(255) NOT NULL,
Type VARCHAR(255) NOT NULL,
Status VARCHAR(255) NOT NULL,
empID INT NOT NULL,
cusID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
CREATE TABLE Books(
bookID INTEGER PRIMARY KEY AUTOINCREMENT,
Title VARCHAR(255) NOT NULL,
Author VARCHAR(255) NOT NULL,
Publisher VARCHAR(255) NOT NULL,
Genre VARCHAR(255) NOT NULL,
bkStatus VARCHAR(255) NOT NULL,
ISBN VARCHAR(255) NOT NULL,
empID INT NOT NULL,
cusID INT NOT NULL,
CONSTRAINT ck_books UNIQUE (ISBN),
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
CREATE TABLE Customer(
cusID INTEGER PRIMARY KEY AUTOINCREMENT,
cusName VARCHAR(255) NOT NULL,
cusAddress VARCHAR(255) NOT NULL,
RegistrationDate VARCHAR(255) NOT NULL,
branchID INT NOT NULL,
CONSTRAINT ck_customer UNIQUE (cusName),
CONSTRAINT fk_branch FOREIGN KEY (branchID) REFERENCES branch (branchID)
);
CREATE TABLE EmployeeRegistersCustomer(
empID INT NOT NULL,
cusID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
```

Shireen Taha
#500581584



```
CREATE TABLE EmployeeRegistersCustomer(
empID INT NOT NULL,
cusID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_customer FOREIGN KEY (cusID) REFERENCES Customer (cusID)
);
CREATE TABLE IssueStatus(
issueID INTEGER PRIMARY KEY AUTOINCREMENT,
bookISBN VARCHAR(255) NOT NULL,
bookTitle VARCHAR(255) NOT NULL,
customerID INT NOT NULL,
issueDate VARCHAR(255) NOT NULL,
deviceName VARCHAR(255) NOT NULL,
deviceType VARCHAR(255) NOT NULL,
CONSTRAINT ck_issueStatus UNIQUE (bookISBN)
);
CREATE TABLE EmployeeUpdatesIssueStatus(
empID INT NOT NULL,
issueID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_issueStatus FOREIGN KEY (issueID) REFERENCES IssueStatus (issueID)
);
CREATE TABLE ReturnStatus(
returnID INTEGER PRIMARY KEY AUTOINCREMENT,
bkISBN VARCHAR(255) NOT NULL,
bkTitle VARCHAR(255) NOT NULL,
custID INT NOT NULL,
returnDate VARCHAR(255) NOT NULL,
devicName VARCHAR(255) NOT NULL,
devicType VARCHAR(255) NOT NULL,
CONSTRAINT ck_returnStatus UNIQUE (bkISBN)
);
CREATE TABLE EmployeeUpdatesReturnStatus(
empID INT NOT NULL,
returnID INT NOT NULL,
CONSTRAINT fk_employee FOREIGN KEY (empID) REFERENCES Employee (empID),
CONSTRAINT fk_returnStatus FOREIGN KEY (returnID) REFERENCES ReturnStatus (returnID)
);
DELETE FROM sqlite_sequence;
COMMIT;
sqlite> 
```

The entity relationship diagram was modified and the updated version is shown below, according to which the tables in the database were created.



Shireen Taha
#500581584

Table Descriptions

Branch: This table consists of a primary key 'branchID', the name of the branch 'brName', and the branch 'address'. It can be seen in the ER design above, that the entity 'Branch' has these attributes which are added to the table as columns. Branch has a one-to-many relationship with 'Employee' and 'Customer' entities, which are translated into tables and are discussed below.

Employee: This table consists of a primary key 'empID', the name of employee 'empName', 'Position' and 'Salary' of employee, and the foreign key 'branchID'. As per the ER design, this entity has a many-to-one relationship with Branch and accordingly, Employee table must contain a foreign key, which is the primary key of Branch, since Employee is on the many side. Therefore, this then establishes the relationship between Branch and Employee. Also, the unique constraint is applied on 'empName', thus making it a candidate key, in order to not have employees by the same name. Employee also has a many-to-many relationship with Customer as seen in the ER design above. In order to establish the relationship between these two entities, another table is created with the foreign keys that are primary keys of Employee and Customer. This table is called EmployeeRegistersCustomer and is described below.

Customer: This table consists of a primary key 'cusID', the name of customer 'cusName', address of customer 'cusAddress', the date on which the customer registered at the library 'RegistrationDate', and the foreign key 'branchID'. As per the ER design, this entity has a many-to-one relationship with Branch. As said above for Employee's many-to-one relationship with Branch, the same theory applies here as to why Customer table must contain a foreign key. Also, the unique constraint is applied on 'cusName', thus making it a candidate key, in order to not have customers by the same name.

Devices: This table consists of a primary key 'devID', name of device 'devName', the 'Type' of device, 'Status' of device as to whether it is available or borrowed, and foreign keys 'empID' and 'cusID'. As per the ER design, this entity has a many-to-one relationship with Employee and Customer. Therefore, to establish a relationship with both, the Devices table consists of foreign keys as such.

Books: This table consists of a primary key 'bookID', the 'Title', 'Author', 'Publisher', and 'Genre' of the book, the status of the book as to whether it is available of borrowed 'bkStatus', the 'ISBN' of the book, and foreign keys 'empID' and 'cusID'. The unique constraint is applied on the 'ISBN' of the book, since the each book is different from the other and the ISBN cannot be the same. As per the ER design, this entity has a many-to-one relationship with Employee and Customer. Therefore, to establish a relationship with both, the Books table consists of foreign keys as such.

Issue Status: This table consists of a primary key 'issueID', the ISBN of the book that is borrowed 'bookISBN', the title of the book 'bookTitle', the ID of the customer who issued the book or device 'customerID', the date that the book or device was issued 'issueDate', the name and type of the device that is borrowed, 'deviceName' and 'deviceType'. The unique constraint is again applied on the ISBN of the book 'bookISBN'. As per the ER design, Issue Status has a many-to-many relationship with Employee. In order to establish the relationship between these two entities, another table is created with the foreign keys that are primary keys of Issue Status and Employee. This table is called EmployeeUpdatesIssueStatus and is described below.

Return Status: This table consists of a primary key ‘returnID’, the ISBN of the book that has been returned ‘bkISBN’, the title of the book ‘bkTitle’, the ID of the customer who returned the book or device ‘custID’, the date that the book or device was returned ‘returnDate’, the name and type of the device that was returned, ‘devicName’ and ‘devicType’. The unique constraint is again applied on the ISBN of the book ‘bkISBN’. As per the ER design, Return Status has a many-to-many relationship with Employee as well. In order to establish the relationship between these two entities, another table is created with the foreign keys that are primary keys of Return Status and Employee. This table is called EmployeeUpdatesReturnStatus and is described below.

EmployeeRegistersCustomer: This table establishes the relationship of many-to-many between Employee and Customer, in which the employee registers a new customer in the library. The table consists of the foreign keys ‘empID’ and ‘cusID’ which are the primary keys of Employee and Customer respectively.

EmployeeUpdatesIssueStatus: This table establishes the relationship of many-to-many between Employee and Issue Status, in which the employee updates the issue status of the book or device that is borrowed. The table consists of the foreign keys ‘empID’ and ‘issueID’ which are the primary keys of Employee and Issue Status respectively.

EmployeeUpdatesReturnStatus: This table establishes the relationship of many-to-many between Employee and Return Status, in which the employee updates the return status of the book or device that has been returned. The table consists of the foreign keys ‘empID’ and ‘returnID’ which are the primary keys of Employee and Return Status respectively.