DigiBooks.com: The Online Book Megastore

COP5725 Group18: Database Design

Shantanu Kande, Sumeet Pande, Uday Manda, Wenyue Shi

University of Florida, Gainesville

The following document contains the database schema for our project "DigiBooks.com" that has been specified with the SQL DDL schema. The list of the tables in our project as well as the DDL SQL queries to create them are as follows:

1. First we create the data base. The name of our database, based on the name of our website will be DigiBooks.

CREATE DATABASE DigiBooks;

2. Now we create different tables that will be used in our databases. The name of the each table, its purpose explained in brief and also its primary key as well as foreign key dependencies will be explained as below.

Table Name: DigiBooks.mRegion
Purpose: Each record in this table will specify a region name and the corresponding ID for that region which will form the scope of the operation.
Keys: The primary key here is the RegionID. There are no foreign key dependencies.
DDL Query:

```
USE DigiBooks;
CREATE TABLE DigiBooks.mRegion (
   RegionID INTEGER NOT NULL,
   RegionName VARCHAR(30) NOT NULL,
   PRIMARY KEY (RegionID));
```

Table Name: DigiBooks.mGender
Purpose: Each record in this table will specify a gender and the corresponding ID for that gender which will be information that will be associated with the customer.
Keys: The primary key here is the GenderID. There are no foreign key dependencies.
DDL Query:

```
CREATE TABLE DigiBooks.mGender (
   GenderID BIT NOT NULL,
   Gender VARCHAR(10) NOT NULL,
   PRIMARY KEY (GenderID));
```

Table Name: DigiBooks.tCustomer
Purpose: Each record in this table will correspond to personal information for each of the customers who will be using the site for online books shopping. It will have personal information fields such as name, gender, address, phone number, zip code, state, email ID etc.
Keys: The primary key here is the CustomerID. There are two foreign key dependencies namely Gender which will be referenced by GenderID from the mGender table and the field State which will be referenced by the RegionID from the mRegion table.
DDL Query:

```
CREATE TABLE DigiBooks.tCustomer (
   CustomerID INTEGER NOT NULL,
   FName VARCHAR(30) NOT NULL,
   LName VARCHAR(30) NOT NULL,
   Gender BIT NOT NULL,
   AddressL1 VARCHAR(80) NOT NULL,
   AddresL2 VARCHAR(80),
   PhoneNumber VARCHAR(15),
   Zipcode INTEGER NOT NULL,
   State INTEGER NOT NULL,
   EmailID VARCHAR(30) NOT NULL,
   Password VARCHAR(30) NOT NULL,
   PRIMARY KEY (CustomerID),
   FOREIGN KEY (Gender) REFERENCES mGender(GenderID),
   FOREIGN KEY (State) REFERENCES mRegion(RegionID));
```

Table Name: DigiBooks.mPublicationHouse
Purpose: Each record in this table will contain information regarding each publication house that publishes the books to be retailed on our website and will contain information such as ID, name and the address.
Keys: The primary key here is the HouseID. There are no foreign key dependencies.
DDL Query:

```
CREATE TABLE DigiBooks.mPublicationHouse (
   HouseID INTEGER NOT NULL,
   HouseName VARCHAR(30) NOT NULL,
   Address VARCHAR(80),
   PRIMARY KEY (HouseID));
```

Table Name: DigiBooks.mGenre
Purpose: Each record in this table will specify a genre to which the books that are listed on our website will belong to. This will also contain information like ID and description.
Keys: The primary key here is the GenreID. There are no foreign key dependencies.
DDL Query:

```
CREATE TABLE DigiBooks.mGenre (
   GenreID INTEGER NOT NULL,
   GenreDesc VARCHAR(30) NOT NULL,
   PRIMARY KEY (GenreID));
```

Table Name: DigiBooks.mLanguage
Purpose: Each record in this table will specify a language to which the books that are listed on our website will belong to. This will also contain information like ID and description.
Keys: The primary key here is the LangID. There are no foreign key dependencies.
DDL Query:

```
CREATE TABLE DigiBooks.mLanguage (
   LangID INTEGER NOT NULL,
   LangName VARCHAR(30) NOT NULL,
   PRIMARY KEY (LangID));
```

Table Name: DigiBooks.tBook
Purpose: Each tuple in this table will correspond to the information regarding the individual books that will be listed on our website. It will have information such as name, author, genre, publication house, ISBN number, Publication Year, language etc.
Keys: The primary key here is the BookID. There are three foreign key dependencies namely PublicationHouse that will reference the HouseID field from mPublicationHouse table, Genre that will reference the GenreID field from the mGenre table, and the language field that will reference the LangID from the mLanguage table.
DDL Query:

The DDL for this table is as follows:

CREATE TABLE DigiBooks.tBook (
BookID INTEGER NOT NULL,
BookName VARCHAR(100) NOT NULL,
Author VARCHAR(50) NOT NULL,
PublicationHouse INTEGER NOT NULL,
Genre INTEGER NOT NULL,
ISBN LONG NOT NULL,
PublicationYear INTEGER NOT NULL,
Price FLOAT(6,2) NOT NULL,
Language INTEGER NOT NULL,
Inventory INTEGER NOT NULL DEFAULT 0,
PRIMARY KEY (BookID),
FOREIGN KEY (PublicationHouse) REFERENCES mPublicationHouse(HouseID),
FOREIGN KEY (Genre) REFERENCES mGenre(GenreID),
FOREIGN KEY (language) REFERENCES mLanguage(LangID));

Table Name: DigiBooks.mPaymentMethod
Purpose: Each record in this table will specify a payment method that will be used by the customer for payment options and the corresponding ID for that method.
Keys: The primary key here is the MethodID. There are no foreign key dependencies.
DDL Query:

CREATE TABLE DigiBooks.mPaymentMethod (
MethodID INTEGER NOT NULL,
PayMethod VARCHAR(30) NOT NULL,
PRIMARY KEY (MethodID));

Table Name: DigiBooks.tCustomerTransaction
Purpose: Each record in this table will correspond to a single customer transaction. Also it is very important to note here that in a single customer transaction the customer may choose to buy multiple books. It will consist of information such as ID, Transaction date, Transaction amount, CustomerID, payment method, Quantity (which is the number of books brought either same or different bought in that transaction).
Keys: The primary key here is the TransID. There are two foreign key dependencies. The field CustomerID references to the field CustomerID from the tCustomer table and the field Paymentmethod corresponds to the field MethodID from the mPaymentMethod table.
DDL Query:

```
CREATE TABLE DigiBooks.tCustomerTransaction (
   TransID INTEGER NOT NULL,
   TransDate DATE NOT NULL,
   TransAmount FLOAT(6,2) NOT NULL,
   CustomerID INTEGER NOT NULL,
   PaymentMethod INTEGER NOT NULL,
   Quantity INTEGER NOT NULL,
   PRIMARY KEY (TransID),
   FOREIGN KEY (CustomerID) REFERENCES tCustomer(CustomerID),
   FOREIGN KEY (PaymentMethod) REFERENCES mPaymentMethod(MethodID));
```

Table Name: DigiBooks.tTransactionDetails

Purpose: Each record in this table will correspond to details regarding a single transaction in the *tCustomerTransaction* table. Now it may happen that a single transaction may contain information regarding multiple books which may have different prices individually. But since the transaction will be represented by a single tuple in the above mentioned table it will not be possible to find out the exact information about the each book sold and its quantities. Hence we need to create this table so that information corresponding to single book sold as a part of transaction along with the price may be captured in the system. Thus a single transaction tuple from *tCustomerTransaction* table may have corresponding single or multiple tuple in this table depending upon the different types of the books which may be a part of the transaction. This information will be very helpful in implementing our proposed Sales Management Dashboard Module.

Keys: The primary key here is the DetailID. There are two foreign key dependencies. The field TransID references to the field TransID from the *tCustomerTransaction* table and the field BookID corresponds to the field BookID from the tBook table.

DDL Query:

```
CREATE TABLE DigiBooks.tTransactionDetails (
   DetailID INTEGER NOT NULL,
   TransID INTEGER NOT NULL,
   BookID INTEGER NOT NULL,
   BookUnitPrice FLOAT(6,2) NOT NULL,
   Quantity INTEGER NOT NULL,
   TotalPrice FLOAT(6,2) NOT NULL,
   TransDate DATE NOT NULL,
   PRIMARY KEY (DetailID),
   FOREIGN KEY (TransID) REFERENCES tCustomerTransaction(TransID),
   FOREIGN KEY (BookID) REFERENCES tBook(BookID));
```