Carlos Carrillo

Date: 06/03/2016

LIBRARY DATABASE: FINAL PROJECT

Outline

This database project represents a library system. As any library system, it lends books and magazines to patrons, has an inventory, and tracks transactions. We believe it is crucial for any type of library to manage a database system since it is generally hard for humans to organize and keep track of large collections of items. Consequently, keeping track of vast collections of books, magazines, movies, photos, among others, is a task that would take too much time for a person without using a database system.

Thus, our database has been designed to categorize, structure and organize a big collection of books and magazines according to the attributes given by the user. Specifically, users will be able to perform the next actions through our database system:

- Create a brand new collection of books or magazines or enter the same items to an existing collection and give to each item attributes such as isbn or issue number, title, author, category, publisher, date of publication, number of pages, among few others.

- Create a list of patrons who are allowed to borrow books or magazines from the library. Each entry in this list will contain personal information about the patrons. The user can modify the patron's information as many times as needed.

- Create a transaction record for a unique patron, in which a new entry is inserted every single time a patron checks out a book or magazine. Each transaction entry will contain the data attributes such as transaction id, product id, patron id, creation date, transaction type, and amount (if a past-due fine is released). This information will allow the user to keep track of the status of every single item in the collection and charge patrons with fines in case they take too long in returning a particular item.

- Create an inventory to keep track of every single item within the library. In Each item will have and id and contain information about the product, such as ISBN number (if it is a book), issue number (if it is a magazine), acquisition date, and unit price. In order to permanently delete an item from the inventory, the user must use the unique id of the desired item and delete it directly from the inventory list, since deleting the item from the book or magazine collection will not delete that same item from the inventory.

- Create a record of all the book authors so both the user and patrons can get information about the books written by a particular author or the author(s) who wrote a particular book.

This way, the data will be entered into the system and accessed on a routine basis by the user. Each user may have an assigned password to gain access to the system; so multiple users can use the system at the same time in different ways.

Database Outline in Words

This database system has been designed for a books and magazines library. This is the description of the library system and the actions that need to be performed through our database system:

- The library lends books and/or magazines to patrons through transactions. The library has an inventory of book and magazine items.

- A patron has an id, a name, a DOB, an address, a join date, and a phone number.

- A book has an ISBN, a title, a publisher, a publish date, a category, a country, and a number of pages. A book author has an id, a name, and a country.

- A magazine has an issue number, a title, an issue date, a country, a category, and a number of pages.

- An inventory entry has an id, an ISBN or issue number corresponding to a particular book or magazine issue (foreign keys), an acquisition date, and a unit price.

- A transaction has an id, a product id corresponding to an inventory entry (foreign key), a patron id corresponding to a particular patron (foreign key), a type and a money amount. There are 3 types of transactions: item checkout, time extension for an item, and overdue item.

- A patron can have many transactions, but a transaction cannot have many patrons (1:N). Consequently, a patron can have many inventory items.

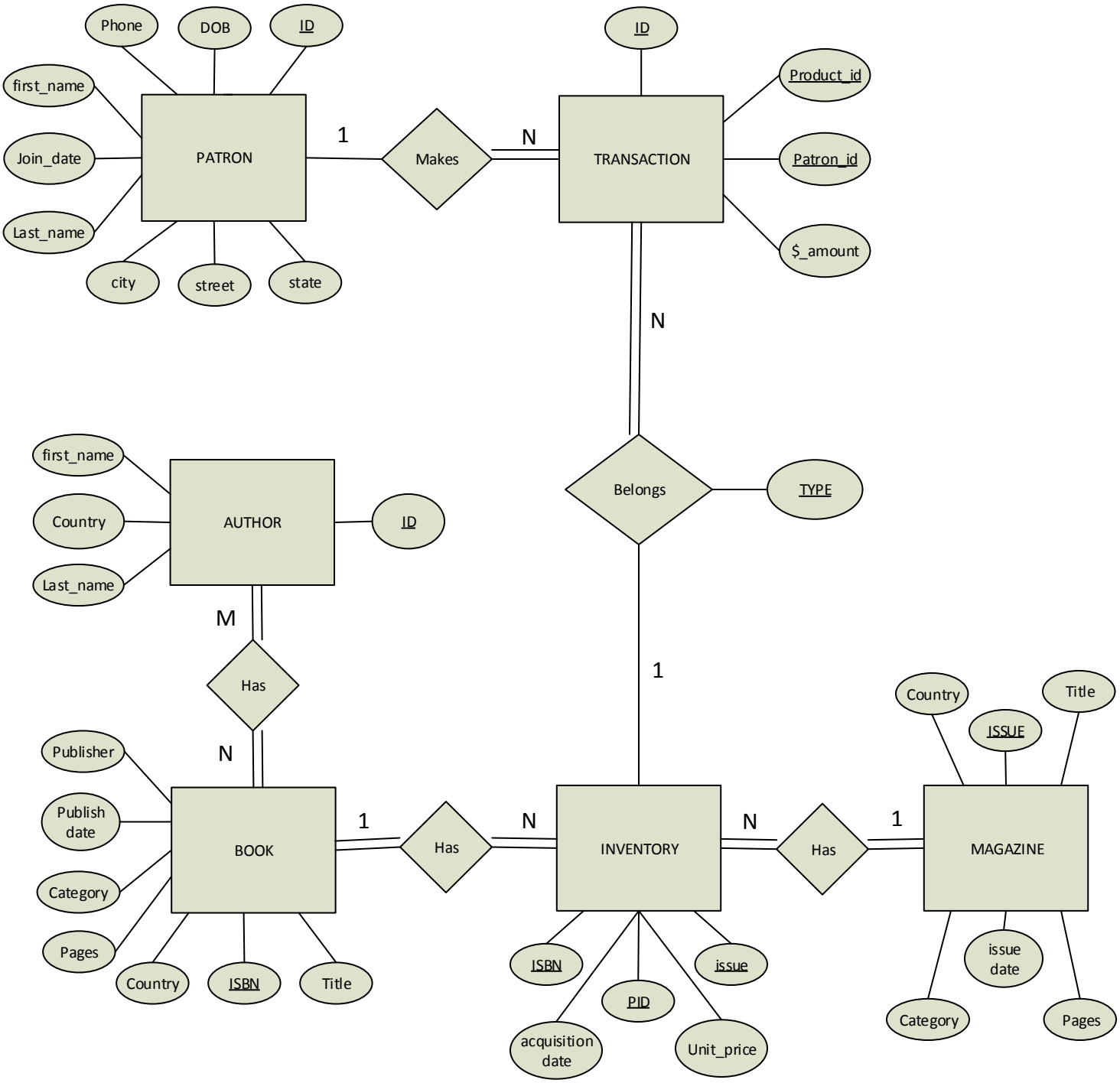
- An inventory item can belong to many transactions, but a transaction cannot point to multiple inventory items (1:N). Each transaction must indirectly correspond to one inventory item (either book or magazine). A new transaction must be open if the **transaction type** corresponding to a particular inventory item changes. For instance, if an inventory item is checked out and a time extension is given to the patron, a new checkout transaction must be opened over the same item and the old checkout transaction should be close/deleted (it would keep the database updated). The same thing happens if the patron is charged with an over-due fine.

- A book can have many inventory entries (1:N). There could be multiple copies of the same book in the inventory, but each copy must have a different id.

- Many books can have many authors (M:N). An author can write many books as well as many authors can write a single book. A separate table/list, titled “book_author”, must represent this relationship. book_author must have a foreign author id attribute corresponding to a particular author as well as a foreign ISBN attribute corresponding to a particular book.

- A magazine can have many inventory entries (1:N). There could be multiple copies of the same magazine issue in the inventory, but each copy must have a different id.

ER Diagram of Database



Database Schema

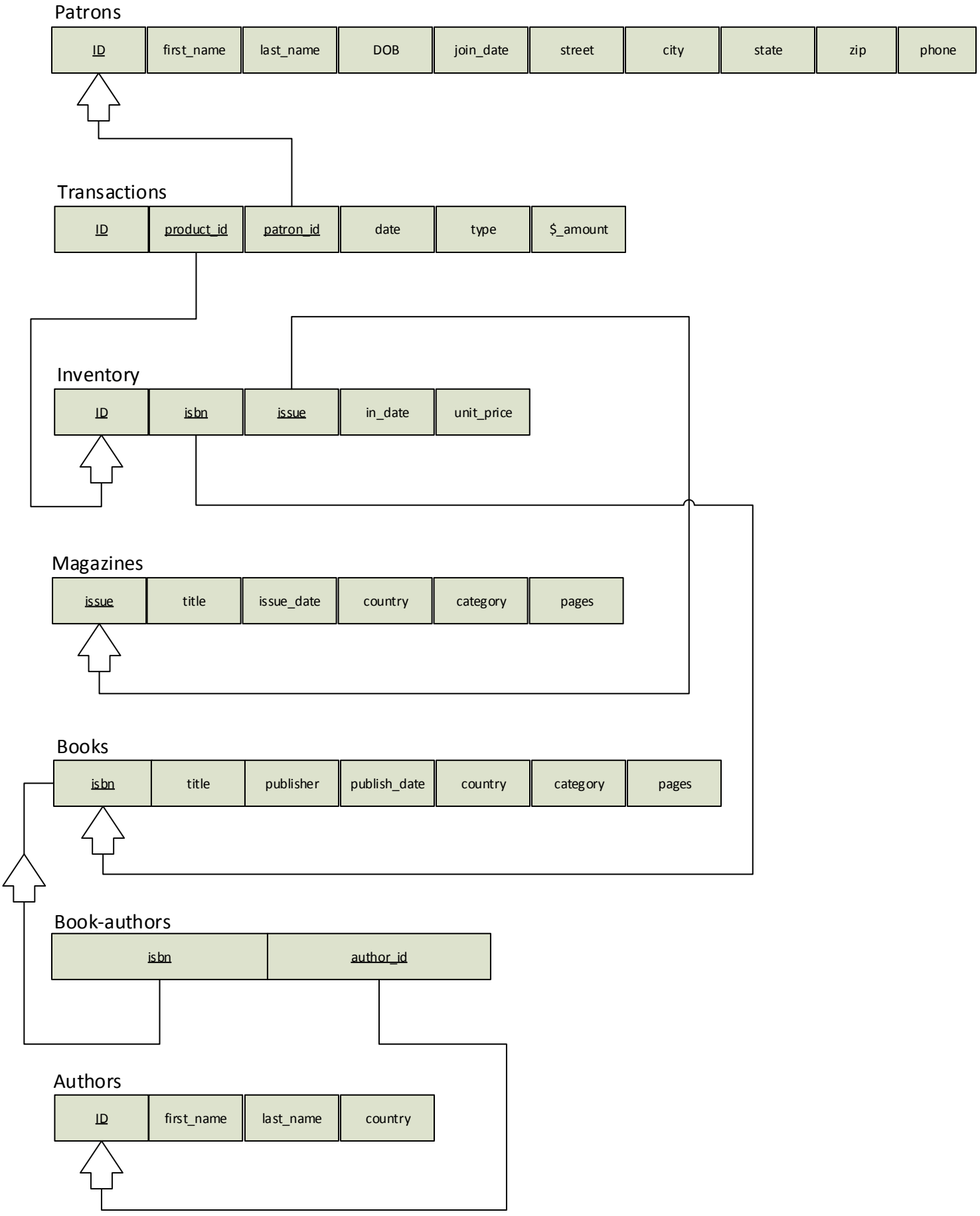


Table Creation Queries
These are the table creation queries for each entity in our library database. We also created pseudo data in order to fill up the database and perform our testing process:

```
-- DROP TABLES IN PROPER ORDER
DROP TABLE IF EXISTS `book_authors`;
DROP TABLE IF EXISTS `transactions`;
DROP TABLE IF EXISTS `patrons`;
DROP TABLE IF EXISTS `authors`;
DROP TABLE IF EXISTS `inventory`;
DROP TABLE IF EXISTS `books`;
DROP TABLE IF EXISTS `magazines`;
```

```
CREATE TABLE `patrons` (
    `id` INT(11) NOT NULL AUTO_INCREMENT,
    `first_name` VARCHAR(255) NOT NULL,
    `last_name` VARCHAR(255) NOT NULL,
    `DOB` date NOT NULL,
    `joinSince` date NOT NULL,
    `street` VARCHAR(255),
    `city` VARCHAR(255),
    `state` VARCHAR(255),
    `zip` INT(11),
    `phone` VARCHAR(255),
    PRIMARY KEY (`id`),
    UNIQUE (`first_name`, `last_name`, `DOB`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
INSERT INTO `patrons` (`first_name`, `last_name`, `DOB`, `joinSince`, `street`, `city`, `state`, `zip`, `phone`) VALUES
    ('Sara', 'Smith', '1970-1-2', '2014-6-2', '956 Jackson St.', 'Miami', 'FL', 60075, '315-288-4574'),
    ('David', 'Atkins', '1979-4-18', '2015-6-9', '2857 Clinton Ave.', 'Corvallis', 'OR', 40075, '415-223-4544'),
    ('Lauren', 'Jensen', '1985-3-2', '2012-7-2', '124 Clark St.', 'Milwaukee', 'WI', 50784, '615-743-4324'),
    ('Carlos', 'Carrillo', '1980-9-4', '2011-3-5', '9180 Castle Ave.', 'Chicago', 'IL', 89722, '564-269-7956'),
    ('Donald', 'Sanders', '1976-3-12', '2010-6-26', '762 Main St.', 'Tucson', 'AZ', 23075, '213-298-4784'),
    ('Sasha', 'Simpson', '1989-4-18', '2013-2-19', '2247 Madison Ave.', 'Seattle', 'WA', 76075, '465-213-4564'),
    ('Franz', 'Griffin', '1931-2-23', '2012-7-12', '734 Ralph St.', 'Dallas', 'TX', 98784, '785-773-4984'),
    ('Tanner', 'England', '1983-7-4', '2013-8-6', '247 Wilson St.', 'Iowa City', 'IA', 50003, '891-263-4716'),
    ('Katie', 'Thompson', '1972-9-24', '2011-5-15', '976 Wabash Ave.', 'Urbana', 'IL', 63722, '984-239-7236'),
    ('Thomas', 'Rodgers', '1987-7-16', '2013-9-16', '243 Central St.', 'Lawrence', 'KS', 89003, '341-223-3416');
```

```
CREATE TABLE `authors` (
    `id` INT(11) NOT NULL AUTO_INCREMENT,
    `first_name` VARCHAR(255) NOT NULL,
    `last_name` VARCHAR(255) NOT NULL,
    `country` VARCHAR(255),
    PRIMARY KEY (`id`),
    UNIQUE (`first_name`, `last_name`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
INSERT INTO `authors` (`first_name`, `last_name`, `country`) VALUES
  ('William', 'Shakespeare', 'England'),
  ('Ernest', 'Hemingway', 'USA'),
  ('Leo', 'Tolstoy', 'Russia'),
  ('Gabriel', 'Garcia', 'Colombia'),
  ('Edgar', 'Allan Poe', 'USA'),
  ('Charles', 'Dickens', 'England'),
  ('Miguel', 'De Cervantes', 'Spain'),
  ('George', 'Orwell', 'England'),
  ('Mark', 'Twain', 'USA'),
  ('Dante', 'Alighieri', 'Italy'),
  ('Honore', 'Balzac', 'France'),
  ('Franz', 'Kafka', 'Austria');

CREATE TABLE `books` (
    `isbn` BIGINT NOT NULL,
    `title` VARCHAR(255) NOT NULL,
    `publisher` VARCHAR(255),
    `publish_date` date,
    `country` VARCHAR(255),
    `category` VARCHAR(255),
    `pages` INT(11),
    PRIMARY KEY (`isbn`),
    UNIQUE (`title`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `books` (`isbn`, `title`, `publisher`, `publish_date`, `country`, `category`,
`pages`) VALUES
  (84536987452, 'British Classics', 'Classic Books', '2013-6-8', 'England', 'Collections', 825),
  (65123874956, 'Dickens and Orwell', 'Collect Editorial', '2008-3-9', 'Ireland', 'Poetry', 1258),
  (36514118799, 'Romeo and Juliet', 'Castell Inc.', '2012-3-19', 'Germany', 'Drama', 937),
  (65849756721, 'Our Mutual Friend', 'Door Books', '1935-8-21', 'Scotland', 'Classic Novels', 764),
  (98456231702, '1984', 'Jerry Sons', '1974-2-16', 'Belgium', 'Science Fiction', 527),
  (63254897265, 'Twain Versus Poe', 'Pinguin', '2002-7-11', 'USA', 'Thriller', 1587),
  (95123598740, 'American Collection', 'Pearson', '1989-4-26', 'USA', 'Collections', 1865),
  (24513698574, 'The Old Man and the Sea', 'Horizon Books', '1978-9-17', 'Brazil', 'Novels', 734),
  (65489721587, 'The Raven', 'Amazon Books', '2015-5-14', 'Canada', 'Poetry', 481),
  (12546987530, 'The Adventures of Tom Sawyer', 'Roggers Ed', '1976-4-15', 'Australia', 'Childhood Novel', 941),
  (35698456721, 'Hispanic Best-Sellers', 'Planeta-Anglo', '2006-2-9', 'Argentina', 'Collections', 2487),
  (12540025483, 'Don Quixote', 'Santillana', '1985-8-17', 'Spain', 'Classic Novels', 1684),
  (98742365154, 'One Hundred Years of Solitude', 'Oveja Negra', '1982-5-19', 'Colombia', 'Magic realism', 921),
  (58423697512, 'War and Peace', 'Ilyich', '1995-3-25', 'Russia', 'Political Novel', 1457),
  (85621479634, 'The Divine Comedy', 'Possetto Livre', '2009-9-15', 'Italy', 'Philosophical Novel', 1943),
  (32548962147, 'Pere Goriot', 'Le Contrat Autre', '2012-3-9', 'France', 'Drama', 846),
  (46587932154, 'The Trial', 'Verschollene', '1988-7-13', 'Germany', 'Philosophical Novel', 574),
  (25487566849, 'European Authors', 'Le Pettite Caza', '2004-2-27', 'France', 'Collections', 3743);
```

```
CREATE TABLE `magazines` (
    `issue` BIGINT NOT NULL,
    `title` VARCHAR(255) NOT NULL,
    `issue_date` date,
    `country` VARCHAR(255),
    `category` VARCHAR(255),
    `pages` INT(11),
    PRIMARY KEY (`issue`),
    UNIQUE (`title`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `magazines` (`issue`, `title`, `issue_date`, `country`, `category`, `pages`)
VALUES
  (36987, 'Time', '2016-6-28', 'USA', 'News', 78),
  (49756, 'Vogue', '2016-8-21', 'Australia', 'Fashion', 64),
  (23874, 'National Geographic', '2016-3-9', 'USA', 'Science', 58),
  (56231, 'Education Today', '2016-2-16', 'New Zealand', 'Education', 44),
  (54897, 'Crafts Beautiful', '2016-7-11', 'United Kingdom', 'Handcrafts', 87),
  (23598, 'Computer Games Magazine', '2016-4-26', 'USA', 'Technology', 65),
  (13698, '2beMAG', '2016-9-17', 'Spain', 'Fashion', 74),
  (14118, 'Forbes', '2016-8-19', 'USA', 'Economics', 42),
  (89721, 'Animal Fair', '2016-5-14', 'Canada', 'Pets', 81),
  (46987, 'Men Health Magazine', '2016-4-15', 'USA', 'Health', 41),
  (98456, 'Tech Today', '2016-2-9', 'Japan', 'Technology', 46),
  (40025, 'People en Español', '2016-8-17', 'Mexico', 'Entertainment', 68),
  (42365, 'Rolling Stone', '2016-5-17', 'USA', 'Pop Culture', 53);

CREATE TABLE `book_authors` (
    `author_id` INT(11) NOT NULL,
    `isbn` BIGINT NOT NULL,
    PRIMARY KEY (`isbn`, `author_id`),
    FOREIGN KEY (`isbn`) REFERENCES `books` (`isbn`)
       ON DELETE CASCADE
       ON UPDATE CASCADE,
    FOREIGN KEY (`author_id`) REFERENCES `authors` (`id`)
       ON DELETE CASCADE
       ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `book_authors` (`author_id`, `isbn`)
VALUES
  (1, 84536987452),(6, 84536987452),(8, 84536987452),(6, 65123874956),
  (8, 65123874956),(1, 36514118799),(6, 65849756721),(8, 98456231702),
  (9, 63254897265),(5, 63254897265),(2, 95123598740),(5, 95123598740),
  (9, 95123598740),(2, 24513698574),(5, 65489721587),(9, 12546987530),
  (7, 35698456721),(4, 35698456721),(7, 12540025483),(4, 98742365154),
  (3, 58423697512),(10, 85621479634),(11, 32548962147),(12, 46587932154),
  (10, 25487566849),(11, 25487566849),(12, 25487566849);
```

```
CREATE TABLE `inventory` (
    `pid` INT(11) NOT NULL AUTO_INCREMENT,
    `isbn` BIGINT,
    `issue` BIGINT,
    `acquisition_date` date,
    `unit_price` VARCHAR(255),
    PRIMARY KEY (`pid`),
    FOREIGN KEY (`isbn`) REFERENCES `books` (`isbn`)
       ON DELETE CASCADE
       ON UPDATE CASCADE,
    FOREIGN KEY (`issue`) REFERENCES `magazines` (`issue`)
       ON DELETE CASCADE
       ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `inventory` (`isbn`, `issue`, `acquisition_date`, `unit_price`)
VALUES
  (84536987452, NULL, '2013-7-8', '23.46'), (65123874956, NULL, '2010-4-9', '12.46'),
  (NULL, 98456, '2016-3-9', '11.29'), (NULL, 40025, '2016-9-17', '5.76'), (NULL, 42365, '2016-6-17', '16.65'),
  (36514118799, NULL, '2012-4-19', '46.52'), (65849756721, NULL, '2011-8-21', '62.76'),
  (98456231702, NULL, '2004-2-16', '37.24'), (NULL, 46987, '2016-5-15', '10.02'),
  (63254897265, NULL, '2012-7-11', '33.46'), (NULL, 49756, '2016-9-21', '4.78'),
  (NULL, 56231, '2016-3-16', '3.64'), (NULL, 54897, '2016-8-11', '5.32'), (NULL, 89721, '2016-6-14', '8.21'),
  (NULL, 23598, '2016-5-26', '9.23'), (95123598740, NULL, '2014-4-26', '9.12'),
  (24513698574, NULL, '2015-9-17', '5.76'), (65489721587, NULL, '2011-5-14', '13.73'),
  (12546987530, NULL, '2016-4-15', '23.21'), (35698456721, NULL, '2010-2-9', '6.98'),
  (NULL, 98456, '2016-3-9', '11.29'), (NULL, 40025, '2016-9-17', '5.76'),    (NULL, 13698, '2016-10-17', '7.21'),
  (NULL, 42365, '2016-6-17', '16.65'), (12540025483, NULL, '2015-8-17', '16.64'),
  (98742365154, NULL, '2012-5-19', '16.02'), (58423697512, NULL, '2015-3-25', '11.6'),
  (85621479634, NULL, '2011-9-15', '6.76'), (32548962147, NULL, '2013-6-19', '37.61'),
  (46587932154, NULL, '2010-7-15', '18.16'), (25487566849, NULL, '2014-9-27', '33.96'),
  (NULL, 36987, '2016-7-28', '8.16'), (NULL, 23874, '2016-4-9', '5.24'), (NULL, 23598, '2016-5-26', '9.23'),
  (NULL, 14118, '2016-9-19', '6.78'), (NULL, 49756, '2016-9-21', '4.78'), (NULL, 54897, '2016-8-11', '5.32'),
  (NULL, 56231, '2016-3-16', '3.64'), (84536987452, NULL, '2013-7-8', '23.46'),
  (65123874956, NULL, '2010-4-9', '12.46'), (36514118799, NULL, '2012-4-19', '46.52'),
  (65849756721, NULL, '2011-8-21', '62.76'), (NULL, 54897, '2016-8-11', '5.32'),
  (NULL, 23598, '2016-5-26', '9.23'), (NULL, 13698, '2016-10-17', '7.21'),
  (NULL, 89721, '2016-6-14', '8.21'), (NULL, 46987, '2016-5-15', '10.02'),
  (58423697512, NULL, '2015-3-25', '11.6'), (85621479634, NULL, '2011-9-15', '6.76'),
  (32548962147, NULL, '2013-6-19', '37.61'), (46587932154, NULL, '2010-7-15', '18.16'),
  (25487566849, NULL, '2014-9-27', '33.96'), (NULL, 98456, '2016-3-9', '11.29'),
  (NULL, 40025, '2016-9-17', '5.76'), (NULL, 42365, '2016-6-17', '16.65'),
  (95123598740, NULL, '2014-4-26', '9.12'), (24513698574, NULL, '2015-9-17', '5.76'),
  (65489721587, NULL, '2011-5-14', '13.73'), (12546987530, NULL, '2016-4-15', '23.21'),
  (35698456721, NULL, '2010-2-9', '6.98'), (NULL, 98456, '2016-3-9', '11.29'),
  (NULL, 40025, '2016-9-17', '5.76'), (NULL, 42365, '2016-6-17', '16.65'),
  (12540025483, NULL, '2015-8-17', '16.64'), (98742365154, NULL, '2012-5-19', '16.02'),
  (58423697512, NULL, '2015-3-25', '11.6'), (85621479634, NULL, '2011-9-15', '6.76'),
  (32548962147, NULL, '2013-6-19', '37.61'), (46587932154, NULL, '2010-7-15', '18.16'),
  (25487566849, NULL, '2014-9-27', '33.96'), (NULL, 36987, '2016-7-28', '8.16'),
  (NULL, 23874, '2016-4-9', '5.24'), (NULL, 14118, '2016-9-19', '6.78'),
  (NULL, 49756, '2016-9-21', '4.78'), (NULL, 56231, '2016-3-16', '3.64'),
  (84536987452, NULL, '2013-7-8', '23.46'), (65123874956, NULL, '2010-4-9', '12.46'),
  (36514118799, NULL, '2012-4-19', '46.52'), (65849756721, NULL, '2011-8-21', '62.76');
```

```
CREATE TABLE `transactions` (
    `id` INT(11) NOT NULL AUTO_INCREMENT,
    `product_id` INT(11) NOT NULL,
    `patron_id` INT(11) NOT NULL,
    `date` date NOT NULL,
    `type` VARCHAR(255) NOT NULL,
    `amount` VARCHAR(255),
    PRIMARY KEY (`id`),
    FOREIGN KEY (`product_id`) REFERENCES `inventory` (`pid`)
       ON DELETE CASCADE
       ON UPDATE CASCADE,
    FOREIGN KEY (`patron_id`) REFERENCES `patrons` (`id`)
       ON DELETE CASCADE
       ON UPDATE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `transactions` (`product_id`, `patron_id`, `date`, `type`, `amount`)
VALUES
   (10, 2, '2016-2-16', 'Past Due', '16.74'), (12, 2, '20016-7-11', 'Borrow', NULL),
   (14, 3, '2016-4-26', 'Past Due', '16.74'), (16, 3, '2016-9-17', 'Borrow', NULL),
   (18, 3, '2016-5-14', 'Past Due', '16.74'), (30, 4, '2016-4-15', 'Borrow', NULL),
   (56, 4, '2016-2-9', 'Past Due', '16.74'), (25, 4, '2016-8-17', 'Borrow', NULL),
   (54, 5, '2016-5-19', 'Past Due', '16.74'), (51, 5, '2016-3-25', 'Borrow', NULL),
   (47, 5, '2016-9-15', 'Past Due', '16.74'), (14, 6, '2016-3-9', 'Borrow', NULL),
   (53, 6, '2016-7-13', 'Past Due', '16.74'), (69, 6, '2016-2-27', 'Borrow', NULL),
   (18, 7, '2016-9-19', 'Past Due', '6.78'), (49, 8, '2016-9-21', 'Borrow', NULL),
   (32, 8, '2016-3-16', 'Past Due', '3.64'), (34, 8, '2016-8-11', 'Borrow', NULL),
   (36, 9, '2016-5-26', 'Past Due', '9.23'), (1, 9, '2016-10-17', 'Borrow', NULL),
   (20, 9, '2016-6-14', 'Past Due', '8.21'), (29, 10, '2016-5-15', 'Borrow', NULL),
   (39, 4, '2016-3-9', 'Past Due', '11.29'), (3, 10, '2016-9-17', 'Borrow', NULL),
   (9, 7, '2016-6-28', 'Past Due', '8.16'), (23, 7, '2016-4-9', 'Borrow', NULL),
   (2, 1, '2016-6-8', 'Past Due', '16.74'), (4, 1, '2016-3-9', 'Borrow', NULL),
   (6, 1, '2016-3-19', 'Past Due', '16.74'), (8, 2, '2016-8-21', 'Borrow', NULL),
   (5, 10, '2016-6-17', 'Past Due', '16.65');

-- PRINT TABLES IN PROPER ORDER
SELECT * FROM `patrons`;
SELECT * FROM `authors`;
SELECT * FROM `books`;
SELECT * FROM `magazines`;
SELECT * FROM `inventory`;
SELECT * FROM `transactions`;
SELECT * FROM `book_authors`;
```

General Use Queries
These are the general use queries for our library database. Some of the queries that are not going to actually be used in the project, but they were useful for testing purposes:

-- SEARCH BOOK BY AUTHOR
```
SELECT a.last_name, a.first_name, b.isbn, b.title, b.pages, b.category
FROM authors a
   INNER JOIN book_authors ba ON ba.author_id = a.id
   INNER JOIN books b ON b.isbn = ba.isbn
WHERE a.first_name=[firstNameInput] AND a.last_name=[lastNameInput];
```

-- SEARCH BOOK BY TITLE
```
SELECT b.title, b.isbn, a.last_name, a.first_name, b.pages, b.category
FROM authors a
   INNER JOIN book_authors ba ON ba.author_id = a.id
   INNER JOIN books b ON b.isbn = ba.isbn
WHERE b.title=[titleInput];
```

-- SEARCH BOOK BY ISBN
```
SELECT b.isbn, b.title, a.last_name, a.first_name, b.pages, b.category
FROM authors a
   INNER JOIN book_authors ba ON ba.author_id = a.id
   INNER JOIN books b ON b.isbn = ba.isbn
WHERE b.isbn=[isbnInput];

SELECT b.isbn, b.title, a.last_name, a.first_name, a.aCountry, b.publish_date, b.publisher,
b.country, b.category, b.pages, i.acquisition_date, i.unit_price, count(*) FROM authors a
        INNER JOIN book_authors ba ON ba.author_id=a.id
        INNER JOIN books b ON b.isbn = ba.isbn
        INNER JOIN inventory i ON i.isbn=b.isbn
WHERE b.isbn=[isbnInput]
GROUP BY b.isbn;
```

-- SEARCH BOOK BY CATEGORY
```
SELECT b.category, b.isbn, b.title, a.last_name, a.first_name, b.pages
FROM authors a
   INNER JOIN book_authors ba ON ba.author_id = a.id
   INNER JOIN books b ON b.isbn = ba.isbn
WHERE b.category=[categoryInput];
```

-- SEARCH FOR A BOOK IN THE INVENTORY
```
SELECT isbn, count(*) AS 'Number of Copies', acquisition_date,
   unit_price, SUM(unit_price) AS 'Total Price' FROM inventory i
WHERE isbn=[isbnInput]
GROUP BY isbn;
```

```
SELECT inventory.pid, books.title AS btitle, magazines.title AS mtitle
        FROM inventory
        LEFT JOIN books ON inventory.isbn = books.isbn
        LEFT JOIN magazines ON magazines.issue = inventory.issue
WHERE books.title =[titleInput] OR magazines.title =[titleInput];

-- SEARCH MAGAZINE BY TITLE
SELECT title, issue, issue_date, country, pages FROM magazines
WHERE title=[titleInput];

-- SEARCH MAGAZINE BY ISSUE
SELECT issue, title, issue_date, country, pages FROM magazines
WHERE issue=[issueInput];

SELECT m.issue, m.title, m.issue_m.date, m.country, m.pages, count(*) FROM magazines
        INNER JOIN inventory i ON i.issue = m.issue
WHERE m.issue =[isssueInput]
GROUP BY m.issue;

-- SEARCH FOR A MAGAZINE IN THE INVENTORY
SELECT issue, count(*) AS 'Number of Copies', acquisition_date,
   unit_price, SUM(unit_price) AS 'Total Price' FROM inventory i
WHERE issue=[issueInput]
GROUP BY issue;

-- SEARCH FOR ALL THE ITEMS CHECKED OUT
SELECT t.type AS 'STATUS', i.isbn, i.issue, p.last_name
   AS 'Patron last name', p.first_name AS 'Patron first name'
FROM patrons p
   INNER JOIN transactions t ON t.patron_id = p.id
   INNER JOIN inventory i ON i.pid = t.product_id
WHERE t.type='Checked out'
ORDER BY i.issue;

-- SEARCH FOR ALL TRANSACTIONS FROM A PATRON
SELECT p.last_name AS 'Patron last name', p.first_name AS 'Patron first name', i.isbn, i.issue,
        CASE
                when t.type = 1 THEN 'checked'
                when t.type = 2 THEN 'past due' END
AS 'STATUS' FROM patrons p
   INNER JOIN transactions t ON t.patron_id = p.id
   INNER JOIN inventory i ON i.pid = t.product_id
WHERE p.first_name=[nameInput]  AND p.last_name=[nameInput];

SELECT inventory.pid, books.title AS btitle, magazines.title AS mtitle
        FROM inventory
        LEFT JOIN books ON inventory.isbn = books.isbn
        LEFT JOIN magazines ON magazines.issue = inventory.issue
WHERE books.title =[Input]  OR magazines.title =[Input] ;
```

```
-- SEARCH FOR ALL BOOKS CHECKED OUT
SELECT t.type AS 'STATUS', i.isbn AS 'Book Checked', p.last_name
   AS 'Patron last name', p.first_name AS 'Patron first name'
FROM patrons p
   INNER JOIN transactions t ON t.patron_id = p.id
   INNER JOIN inventory i ON i.pid = t.product_id
WHERE t.type=[typeInput]  AND i.isbn != 'NULL';

-- SEARCH FOR TRANSACTIONS
SELECT id, date, type, first_name, last_name, btitle, title, author_fname, author_lname FROM
(SELECT t.id, t.date, t.type, p.first_name, p.last_name, t.product_id, i.isbn, i.issue AS missue
FROM transactions t
INNER JOIN patrons p ON t.patron_id = p.id
INNER JOIN inventory i ON i.pid = t.product_id
WHERE p.id =?) AS d
LEFT JOIN magazines m ON m.issue = d.missue LEFT JOIN (SELECT b.isbn AS bisbn, b.title
AS btitle, a.first_name AS author_fname, a.last_name AS author_lname FROM books b
INNER JOIN book_authors ba ON ba.isbn = b.isbn
INNER JOIN authors a ON a.id = ba.author_id) as e ON e.bisbn = d.isbn GROUP BY id
ORDER BY id;

SELECT id, date, type, first_name, last_name, btitle, title, author_fname, author_lname FROM
(SELECT t.id, t.date, t.type, p.first_name, p.last_name, t.product_id, i.isbn, i.issue AS missue
FROM transactions t
INNER JOIN patrons p ON t.patron_id = p.id
INNER JOIN inventory i ON i.pid = t.product_id) AS d
LEFT JOIN magazines m ON m.issue = d.missue LEFT JOIN (SELECT b.isbn AS bisbn, b.title
AS btitle, a.first_name AS author_fname, a.last_name AS author_lname FROM books b
INNER JOIN book_authors ba ON ba.isbn = b.isbn
INNER JOIN authors a ON a.id = ba.author_id) as e ON e.bisbn = d.isbn
GROUP BY id;

-- SEARCH FOR ALL MAGAZINES CHECKED OUT
SELECT t.type AS 'STATUS', i.issue AS 'Magazine Checked', p.last_name
   AS 'Patron last name', p.first_name AS 'Patron first name'
FROM patrons p
   INNER JOIN transactions t ON t.patron_id = p.id
   INNER JOIN inventory i ON i.pid = t.product_id
WHERE t.type=[typeInput] AND i.issue != 'NULL';

-- SEARCH FOR ALL THE PATRONS WITH DEBT BY PRODUCT
SELECT p.last_name AS 'Patron last name', p.first_name AS
   'Patron first name', t.amount AS 'FINE AMOUNT', i.isbn, i.issue
FROM patrons p
   INNER JOIN transactions t ON t.patron_id = p.id
   INNER JOIN inventory i ON i.pid = t.product_id
WHERE t.type=[idInput]
ORDER BY i.issue;
```

```
-- SEARCH FOR A PARTICULAR PATRON'S TOTAL DEBT
SELECT p.last_name AS 'Patron last name', p.first_name AS
   'Patron first name', SUM(t.amount) AS 'TOTAL FINE'
FROM patrons p
   INNER JOIN transactions t ON t.patron_id = p.id
   INNER JOIN inventory i ON i.pid = t.product_id
WHERE t.type=2 AND p.first_name=[firstName] AND p.last_name=[lastName];

-- DELETE A TRANSACTION BY patron_id AND product_id
-- Delete transaction
DELETE FROM transactions WHERE patron_id=[idInput] AND product_id=[idInput];

-- Check if the entry was deleted (you must see "Empty set")
SELECT id FROM transactions WHERE patron_id=[idInput]  AND product_id=[idInput];

-- DELETE A BOOK FROM THE COLLECTION BY ISBN
-- This will delete all copies of a particular isbn

-- First, delete book from inventory
DELETE FROM inventory WHERE isbn = [isbnInput];

-- Check if the book was deleted from inventory (you must see "Empty set")
SELECT isbn, title FROM books WHERE isbn = [isbnInput];

-- Second, delete book from collection
DELETE FROM books WHERE isbn = [isbnInput];

-- Check if the book was deleted from collection (you must see "Empty set")
SELECT isbn, title FROM books WHERE isbn = [isbnInput];

-- Third, delete book from Author-book
DELETE FROM book_authors WHERE isbn = [isbnInput];

-- Check if the book was deleted from collection (you must see "Empty set")
SELECT isbn, author_id FROM book_authors WHERE isbn = [isbnInput];

-- DELETE A MAGAZINE FROM THE COLLECTION BY ISSUE
-- This will delete all copies of a particular issue
-- First, delete magazine from inventory
DELETE FROM magazines WHERE issue = [issueInput];

-- Check if the magazine was deleted from inventory (you must see "Empty set")
SELECT issue, title FROM magazines WHERE issue = [issueInput];

-- Second, delete magazine from collection
DELETE FROM magazines WHERE issue = [issueInput];

-- Check if the magazine was deleted from the collection (you must see "Empty set")
SELECT issue, title FROM magazines WHERE issue = [issueInput];
```

-- DELETE A PATRON BY ID OR NAME
-- This will delete Carlos and Tanner from the list
-- Main delete query
DELETE FROM patrons WHERE id = =[idInput];
DELETE FROM patrons WHERE first_name=[nameInput] AND last_name=[nameInput];

-- Check if the patron was deleted (you must see "Empty set")
SELECT id, last_name FROM patrons
  WHERE id = [idInput];
SELECT id, last_name FROM patrons
  WHERE first_name=[nameInput] AND last_name=[nameInput];

-- DELETE A AUTHOR BY ID OR NAME
-- Main delete query
DELETE FROM authors WHERE id = [idInput];
DELETE FROM authors WHERE first_name=[nameInput] AND last_name=[nameInput];

-- Delete author also from book-authors ONLY BY ID IS POSSIBLE
DELETE FROM book_authors WHERE author_id = [idInput];

-- Check if the author was deleted (you must see "Empty set")
SELECT id, last_name FROM authors
  WHERE id = [idInput];
SELECT ba.author_id FROM book_authors ba INNER JOIN authors a ON a.id=ba.author_id
  WHERE a.id = [idInput];
SELECT id, last_name FROM authors
  WHERE first_name=[nameInput] AND last_name=[nameInput];
SELECT ba.author_id FROM book_authors ba INNER JOIN authors a ON a.id=ba.author_id
  WHERE a.first_name=[nameInput] AND a.last_name=[nameInput];

-- UPDATE PATRON DATA
UPDATE patrons SET id='?', first_name='?', last_name='?', DOB='?',
  joinSince='?', street='?', city='?', state='?', zip='?', phone='?'
WHERE id = [idInput];

-- UPDATE AUTHOR DATA
UPDATE authors SET id='?', first_name='?', last_name='?', country='?'
WHERE id = [idInput];

-- UPDATE BOOK DATA
UPDATE books SET isbn='?', title='?', publisher='?', publish_date='?',
  country='?', category='?', pages='?'
WHERE isbn = [isbnInput];

-- UPDATE MAGAZINE DATA
UPDATE magazines SET issue='?', title='?', issue_date='?', country='?',
  category='?', pages='?'
WHERE issue = [issueInput];

-- UPDATE INVENTORY DATA
UPDATE inventory SET pid='?', isbn='?', issue='?', acquisition_date='?', unit_price='?'
WHERE pid = [idInput];

-- UPDATE TRANSACTION DATA BY patron_id AND product_id
-- By the nature of this DB system ONLY date, type, and amount can be updated
UPDATE transactions SET date='?', type='?', amount='?'
WHERE patron_id=[idInput] AND product_id=[idInput];

-- UPDATE BOOK-AUTHOR DATA
-- Book-author will automatically be updated if an author or a book is DELETED
-- If an author or a book is UPDATED or ADDED, use this query with the NEW
-- id and isbn values AFTER they had been entered into the authors and books table

INSERT INTO `book_authors` (`author_id`, `isbn`) VALUES
((SELECT id FROM authors WHERE id=[idInput]), (SELECT isbn FROM books WHERE isbn=[isbnInput]));

-- PLEASE RUN THE DEFINITION.SQL FILE AGAIN IN ORDER TO REPOPULATE THE DATA BASE