Individual Book Library Database

Project Creator: Mariam Ben-Neticha
Database Project Document
Date Created: 5.11.16

OUTLINE:

The database will act as a library of sorts. It is meant to be personalized by an individual. The library will hold books that the individual has read in completion. The data stored in the database will include the individual's information, book information, authors, genres, and the purchase/acquired date of the book.

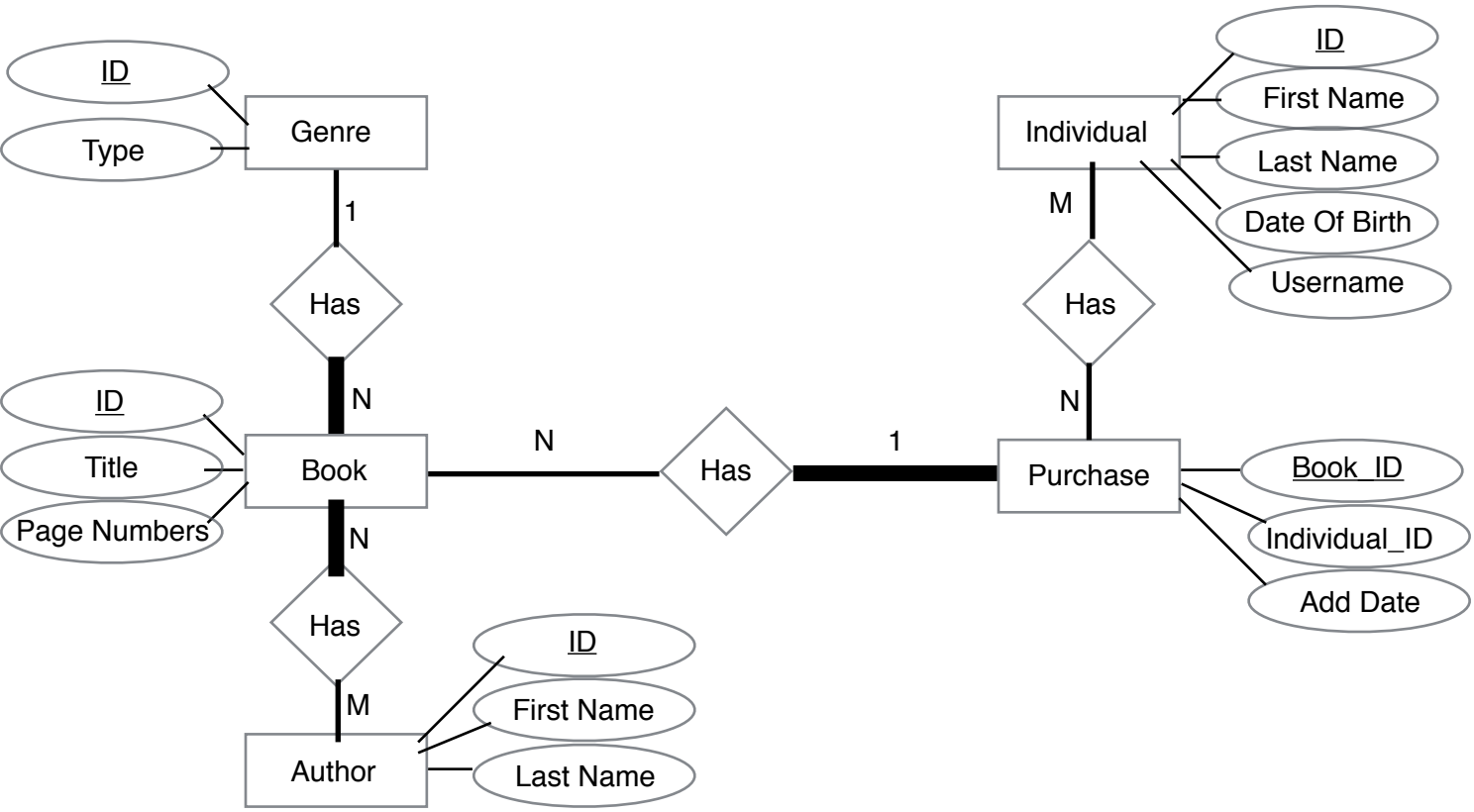
The reason why this information is important to keep track of is that it provides a place in which an individual can search for a book they read during a certain year whose title and/or author slips their mind. Finally, knowing how many novels you've read in your lifetime can be a very interesting piece of information, especially to avid readers.

DATABASE OUTLINE IN WORDS

The following is the description of this Bookcase Library:
The library(database) stores books. A Book has an id, title, author, genre, and page number.
Books have different genres. Each Genre has an id and a type.
A Book belongs to exactly one Genre. A Genre can consist of zero or more Books.
A Book has at least one author. An Author can have zero or more Books.
Each Author has an id, first_name, and last_name.
Books have different purchase dates. Each Purchase has a Book_id, an Individual_id and an add_date. A Book can only have one purchase date.

There are Readers/Individuals. An individual has an id, first_name, last_name, username, and birth_date.
An Individual can read/Purchase zero or more Books.
A Book can be read/Purchased by zero or more Individuals.
Many Individuals can read/Purchase many Books.

ER DIAGRAM OF DATABASE



DATABASE SCHEMA

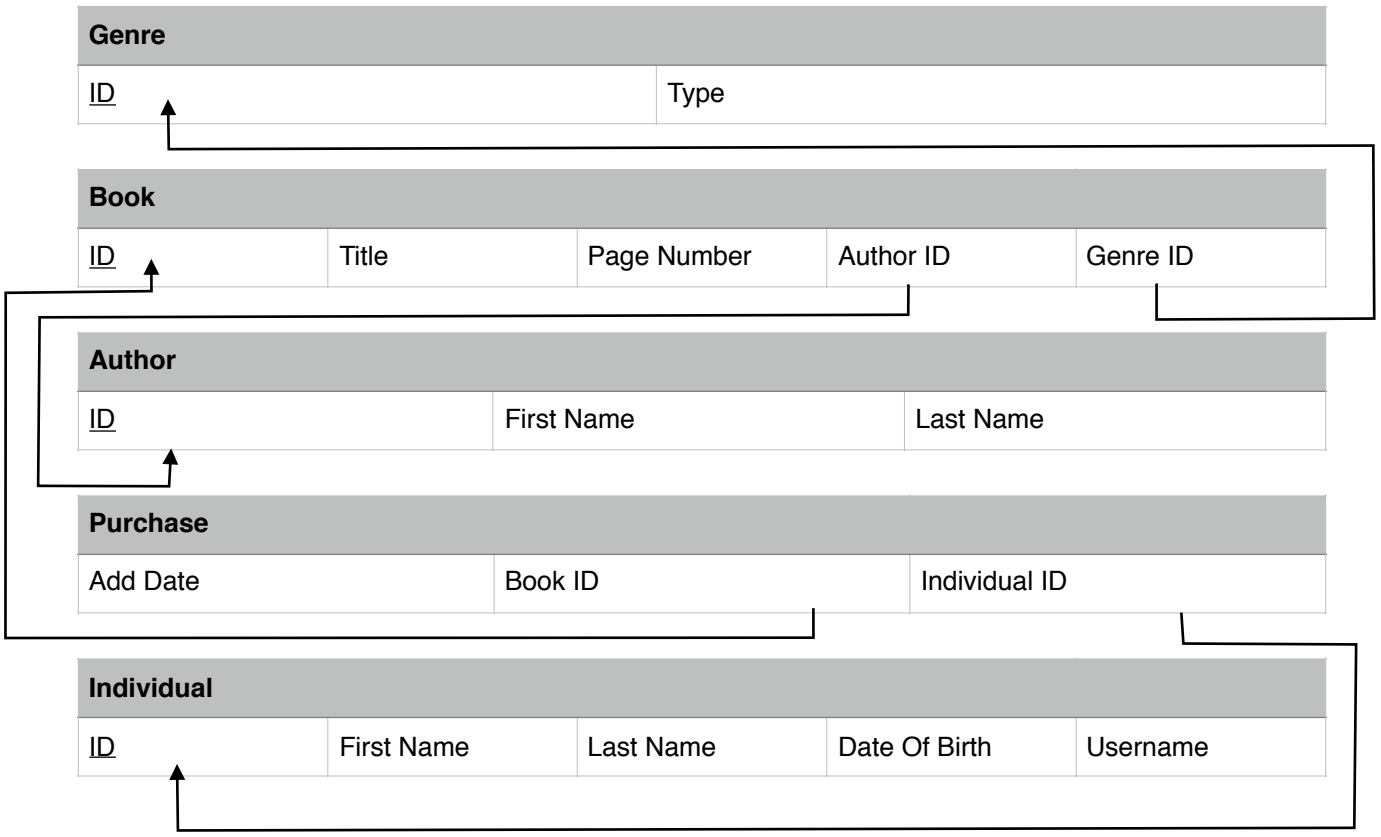


TABLE CREATION QUERIES:

-- Create a table called genre with the following properties:
-- id - an auto incrementing integer which is the primary key
-- type - a varchar of maximum length 255, cannot be null

```
CREATE TABLE `genre` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `type` varchar(255) NOT NULL,
    PRIMARY KEY (`id`),
    UNIQUE (`type`)
) ENGINE=InnoDB;
```

-- Create a table called author with the following properties:
-- id - an auto incrementing integer which is the primary key
-- first_name - a varchar of maximum length 255, cannot be null
-- last_name - a varchar of maximum length 255, cannot be null
-- UNIQUE Constraint uc_AuthorName of first_name and last_name

```
CREATE TABLE `author` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `first_name` varchar(255) NOT NULL,
    `last_name` varchar(255) NOT NULL,
    PRIMARY KEY (`id`),
    CONSTRAINT uc_AuthorName UNIQUE (`first_name`,`last_name`)
) ENGINE=InnoDB;
```

-- Create a table called book with the following properties:
-- id - an auto incrementing integer which is the primary key
-- aid - an integer which is a foreign key reference to the author table and a UNIQUE index
-- gid - an integer which is a foreign key reference to the genre table and a UNIQUE index
-- title - a varchar of maximum length 255, cannot be null
-- page_number - an integer value

```
CREATE TABLE `book` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `aid` int(11),
    `gid` int(11),
    `title` varchar(255) NOT NULL,
    `page_number` int(11),
    PRIMARY KEY (`id`),
    CREATE UNIQUE INDEX aid ON `book` (`aid`),
    CREATE UNIQUE INDEX gid ON `book` (`gid`),
    FOREIGN KEY (`aid`) REFERENCES `author` (`id`),
    FOREIGN KEY (`gid`) REFERENCES `genre` (`id`)
) ENGINE=InnoDB;
```

-- Create a table called individual with the following properties:
-- id - an auto incrementing integer which is the primary key
-- first_name - a varchar of maximum length 255, cannot be null
-- last_name - a varchar of maximum length 255, cannot be null
-- dob - a date type
-- username - a UNIQUE varchar of maximum length 255, cannot be null

```
CREATE TABLE `individual` (
    `id` int(11) NOT NULL AUTO_INCREMENT,
    `first_name` varchar(255) NOT NULL,
    `last_name` varchar(255) NOT NULL,
    `username` varchar(255) NOT NULL,
    `dob` date,
    PRIMARY KEY (`id`),
    UNIQUE (`username`)
) ENGINE=InnoDB;
```

-- Create a table called purchase with the following properties, this is a table representing a many-to-many relationship
-- between books and individuals:
-- bid - an integer which is a foreign key reference to book
-- iid - an integer which is a foreign key reference to individual and a UNIQUE index
-- add_date - a date type
-- The primary key is a combination of bid and iid

```
CREATE TABLE `purchase` (
    `bid` int(11),
    `iid` int(11),
    `add_date` date,
    PRIMARY KEY (`bid`, `iid`),
    CREATE UNIQUE INDEX iid ON `purchase` (`iid`),
    FOREIGN KEY (`bid`) REFERENCES `book` (`id`),
    FOREIGN KEY (`iid`) REFERENCES `individual` (`id`)
) ENGINE=InnoDB;
```

GENERAL USE QUERIES:

(bookHome.php)

SELECT:

```
SELECT id, first_name, last_name FROM author;

SELECT id, type FROM genre;

SELECT book.id, book.title FROM book;
```

(addreader.php)

INSERT:

```
INSERT INTO individual (first_name, last_name, dob, username)
VALUES (['first_name'], ['last_name'],['dob'], ['username']);
```

SELECT:

```
SELECT i.first_name, i.last_name, i.username FROM individual i ORDER BY
i.last_name ASC;
```

(addAuthor.php)

INSERT:

```
INSERT INTO author (first_name, last_name)
VALUES (['a_first_name'],['a_last_name']);
```

SELECT:

```
SELECT a.first_name, a.last_name FROM author a ORDER BY a.last_name ASC;
```

(addGenre.php)

INSERT:

```
INSERT INTO genre (type)
VALUES (['genre_type']);
```

SELECT:

```
SELECT g.type FROM genre g ORDER BY g.type ASC;
```

(addBook.php)

INSERT:

```
INSERT INTO book (aid, gid, title, page_number)
VALUES (['author'], ['genre'], ['title'], ['numBookPage']);
```

SELECT:

```
SELECT b.title, a.first_name, a.last_name, g.type, b.page_number FROM book b
INNER JOIN author a ON b.aid =a.id
INNER JOIN genre g ON b.gid=g.id
ORDER BY b.title ASC;
```

(addToBookshelf.php)

INSERT:

```
INSERT INTO purchase (bid, iid, add_date)
VALUES (['i_book'], (
        SELECT individual.id FROM individual WHERE individual.username =
        ['i_username']
        ), ['purchase_date']);
```

SELECT:

```
SELECT i.first_name, i.last_name, a.first_name, a.last_name, b.title, p.add_date
FROM individual i
INNER JOIN purchase p ON i.id = p.iid
INNER JOIN book b ON p.bid = b.id
```

INNER JOIN author a ON b.aid = a.id
WHERE i.username = [‘i_username’]
GROUP BY b.title ORDER BY p.add_date DESC;

(searchUsername.php) — uses aggregate function COUNT()

SELECT:

SELECT i.first_name, i.last_name, b.title, a.first_name, a.last_name, p.add_date,
COUNT(b.title) AS booksRead
FROM individual i
INNER JOIN purchase p ON i.id = p.iid
INNER JOIN book b ON p.bid = b.id
INNER JOIN author a ON b.aid = a.id
WHERE i.username = [’s_username’]
GROUP BY b.title ORDER BY p.add_date DESC;

(searchYear.php) — uses aggregate function COUNT()

SELECT:

SELECT i.first_name, i.last_name, b.title, a.first_name, a.last_name, p.add_date,
YEAR(p.add_date), COUNT(b.title) AS booksRead
FROM purchase p
INNER JOIN individual i ON p.iid = i.id
INNER JOIN book b ON p.bid = b.id
INNER JOIN author a ON b.aid = a.id
WHERE i.username = [‘y_username’] AND YEAR(p.add_date) = [’s_year’]
GROUP BY b.title;

HTML:

All forms and buttons are present and allow user to preform all actions required of the database. Site is fully functional.

PHP:

The mysqli class was used and no error are produced.

STYLE:

The site is designed in a simple and clear but fun fashion to increase user enjoyment.