Bookstore Sales Tracker

By Mo Hudeihed and Saurav Shrestha

Feedback by the peer reviewer

Review 1 (Zachary Maes - TA)

Great job on your first draft group 21. You appear to have a strong foundational understanding of the concepts learned so far. Please see my feedback for each of the key questions below. I think you are mostly on the right track but need to add a few more things prior to your final submission.

1.Does the overview describe what problem is to be solved by a website with DB back end?

Your overview is concise and describes the problem in a definitive way. You organize the information well and implement clear formatting methods like **bolding** and *italicization* that make entities stick out to the reader.

2.Does the overview list specific facts?

Your overview does list specific facts about the various entities like Customers and Stores. You also do a wonderful job painting the picture of how the business will operate using specific numerical data for revenue, sales volume, number of locations, and number of customers.

3.Are at least four entities described and does each one represent a single idea to be stored as a list?

You appear to have five entities that are all clearly defined and represent a single idea. I don't foresee any issues converting these entities into actual SQL.

(Note to students: If there was an actual issue, now would be a great time to describe what you see and how the group could go about fixing it.)

4.Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?

Your outline does a great job describing each entity's purpose, attributes, and relationships. The color and **bolding** go a long way in keeping the details organized. However, I do see a few things that are missing from the outline section:

·You are missing the **AuthorsBooks** intersection table in your outline section. Please add this table to the outline along with the two FK attributes, their details, and a relationships section.

·While you describe the foreign keys in the relationships sections (and your descriptions are correct!), you should also add these attributes to the outline and show the “FK” in the same way that you added “PK”. Correct this for **Invoices** and **AuthorsBooks.** Here is an example of how you might change the **Invoices** table to accomplish this:

Invoices: ...(description)...

·invoiceID: int, auto_increment, unique, not NULL, PK

·date: date, not NULL

·bookID: int, FK, not NULL

·storeID: int, FK, not NULL

·customerID: int, FK, not NULL

Relationships: (keep the current descriptions)

5**.Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?**

All one-to-many relationships have been accurately established, as demonstrated by the precise definitions of foreign keys (FK) and primary keys (PK) outlined in the relationship sections. There is also one intersection table **AuthorsBooks** that implements a many-to-many relationship between **Authors** and **Books**. The accuracy of these relationships is also reflected in the correct use of crow's foot notation within the Entity Relationship Diagram. **I recommend that you add the “FK” attributes inside of your ERD to make these relationships more clear.**

6.Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

Good work on your attention to details. Your entity names are consistently plural, capitalized, **bold,** and follow **PascalCase**. Your attributes are singular, colored blue for clarity, and follow camelCase.

Overall, great job on this project draft! If you implement the changes above, they will go a long way in keeping your project organized. This will help streamline your coding and SQL work during future project steps. I am happy to clarify any of my suggestions above.

Review 2 (Jerome Cagado)

1.Does the overview describe what problem is to be solved by a website with DB back end?

Your overview is well done. It explains the reason for why BooksRus needs a database and the overview explains the various data that will be tracked. I also like how you bolded the entities and provided color-coded to the attributes. It provides an easy ready so good on that.

2.Does the overview list specific facts?

The overview does list the specific facts about the various entities that you provided. You relate how the author's entity has a relationship with the book entity. In addition, you relate customers to the stores and the overall entities makes sense in a database.

3.Are at least four entities described and does each one represent a single idea to be stored as a list?

You appear to have five entities that display why they are there. You also have each entity represent a single idea to be stored as a list. The only thing that I would suggest is differentiating your attributes that have the same name. For instance, under stores you have phone and you also have phone under customers. Why not make your phone under the stores entities as storePhone and customers phone as customerPhone? This is also the case for firstName, lastName which is the same attribute name for authors. Just something to think about.

4.Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?

The outline does indeed outline the entity details and describes the purpose of each, while also listing the attribute datatypes, constraints, and relationships between entities. I feel overall, you guys did a great job.

5.Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?

The outline does indeed have a 1:M relationship and they are correctly formulated. There is also one M:M relationship as well. The ERD makes logical sense. The only issue I have is that you technically can make it a bit more complex if you like. The authorID and bookID could technically fall in either the store or invoices. Something to consider.

6.Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

There is consistency in the naming convention. The entities are plural and the attributes are singular. You also capitalize the entities. You guys also opted for proper naming conventions rather than brevity so good job on that.

Great job and good luck on the project!

Review 3 (Katalina Nakoa)

Great job, Group 21, here's my review of your draft:

1. Does the overview describe what problem is to be solved by a website with DB back end?

The overview is very well-written and describes the problem that the website with a DB backend aims to solve. It shows the need for better sales tracking capabilities and improved inventory management for BooksRus, a bookstore chain with lots of stores across the nation.

2. Does the overview list specific facts?

Yes, the overview lists specific facts about BooksRus, such as its annual book sales volume, the number of customers served, and the annual revenue generated. These specific facts provide a clear understanding of the scale and scope of the business.

3. Are at least four entities described, and does each one represent a single idea to be stored as a list?

There are five entities described: Authors, Books, Invoices, Stores, and Customers. Each entity represents a single idea and the details provided for each entity contribute to a good understanding of the data to be stored.

4. Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints, and describe relationships between entities?

The outline of entity details is thorough and well-organized. It effectively describes the purpose of each entity, lists attribute datatypes and constraints and describes relationships between entities. The use of relationships and the inclusion of intersection tables (like AuthorsBooks) for M:M relationships shows a good use of database design principles.

5. Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?

Yes, the 1:M relationships are correctly formulated, and there is one M:M relationship between Authors and Books, implemented with the AuthorsBooks intersection table. The Entity Relationship Diagram could be improved by adding FK attributes to make the relationships clearer, also suggested by other reviewers.

6. Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

Yea, there is consistency in naming conventions. Entities are consistently plural, attributes are singular, and capitalization follows a pattern. It seems like readability was prioritized and maintains a structure.

Overall, you did a good job in creating a draft for the bookstore sales tracker. H. Keep up the good work!

Review 4 (Alma Valenzuela)

Hi Team 21, overall good job on your draft! Here is my feedback:

Does the overview describe what problem is to be solved by a website with DB back end?

The overview does a good job describing the problem that is to be solved through the use of a database. The bookstore chain, BooksRus, is in need of a database to tracks sales to improve inventory management. I like that the entities and relationships of the database were included in the overview because it helps me better understand how the database will help BooksRus.

Does the overview list specific facts?

Yes, the overview states the name of a bookstore chain and mentions how many books they sell a year, the number of customers they have and the revenue generated. It also mentions the entities and relationships of the database.

Are at least four entities described and does each one represent a single idea to be stored as a list?

All of the entities are described. They each represent a single idea and the description explains the purpose of having them included in the database.

Does the outline of entity details describe the purpose of each, list attribute datatypes and constraints and describe relationships between entities?

Each entity has attributes and relationships. The attributes include the datatypes and constraints. The relationships between the entities are clearly outlined through the use of bullet points. The different font styles make it clear to understand what the entities vs attributes are.

Are 1:M relationships correctly formulated? Is there at least one M:M relationship? Does the ERD present a logical view of the database?

The 1:M relationships are properly formulated and explained and there is at least one M:M relationship. I noticed that the relationships described the foreign keys of the entities but the entities themselves did not have the foreign keys as attributes. For example, Invoices does not have the FK bookID in the outline. I am not sure the reasoning for this but it would be useful to include the foreign keys in the entity outline. The ERD is missing the AuthorsBooks intersection table but, other than this, the ERD represents the outline well and appropriately shows the relationships between the entities.

Is there consistency in a) naming between overview and entity/attributes b) entities plural, attributes singular c) use of capitalization for naming?

There is consistency in the naming between the entities and attributes. Entities are plural and the attributes are singular. The attributes are written in camelCase.

Canvas Draft Assignment Feedback (Dylan Majewski - ULA)

This is a great idea for a database and it seems like you have a firm understanding how of you want to implement it! The ERD provided gives more than it needs. For a better explanation, see the following canvas page:
https://canvas.oregonstate.edu/courses/1946034/pages/erd-vs-schema?module_item_id=23809319

A) **Fixes based on Feedback from Step 1:**

- Added "**AuthorsBooks**" intersection table to the outline, including the two FK attributes "authorID" and "bookID" with their data types and constraints, and a relationship section.

- Added FK to denote foreign keys to the attributes "bookID", "storeID" and "customerID" in the outline of "**Invoices**" table.

- Simplified the ER diagram by modeling it without any attributes. This allowed for basic understanding and logical view of the relationships between entities without going into the details of their properties.

One of the suggestion we decided not to act on was for the following:

"The only thing that I would suggest is differentiating your attributes that have the same name. For instance, under stores you have phone and you also have phone under customers. Why not make your phone under the stores entities as storePhone and customers phone as customerPhone? This is also the case for firstName, lastName which is the same attribute name for authors. Just something to think about."

This was because the attributes are implied by their tables, making their names more concise and their meanings more intuitive. Additionally, since these attributes serve the same purpose (name and contact information) in different tables, it is a good practice to be consistent with the attribute names to make the database easier to understand, manage, and maintain.

Another suggestion we decided not to act on was for the following:

"*The accuracy of these relationships is also reflected in the correct use of crow's foot notation within the Entity Relationship Diagram. I recommend that you add the "FK" attributes inside of your ERD to make these relationships more clear."*

This was because we decided to make our Entity Relationship Diagram much simpler, as described in "Actions based on feedback" section above. Since the diagram focused on just the entities without specifying any attributes, there was no need to add "FK" attributes inside the ERD. These details will be included in our schema diagram in the later part of the project.

Lastly,

"The only issue I have is that you technically can make it a bit more complex if you like. The authorID and bookID could technically fall in either the store or invoices. Something to consider."

We decided not to include authorID and bookID inside the stores table because authors and books are two separate entities with their own attributes. This would also lead to redundancy because author and book information could be repeated for each instance of a store. Similarly, authorID and booksID attributes are already in the invoices table as foreign keys. Our main goal of the database is to minimize data redundancy and dependency, which would not be achieved by following this suggestion.

Upgrades to the Draft version

Apart from the actions taken based on the feedback above, there were no design changes made to the project.

B) **Project Outline and Database Outline:**

Overview

BooksRus, a bookstore chain with an annual book sales volume of 2,000 books, serves approximately 1,500 customers, and generates $100,000 in revenue annually. An online sales tracker system, utilizing a relational database, is created to enhance the chain's sales tracking capabilities and improve inventory management. This system will efficiently record *Invoices*, capturing details of *Books* purchased by the *Customers* across its 5 *Stores* located throughout the United States. Additionally, the database schema accommodates scenarios of multiple *Authors* per book, multiple purchases for a single book, and the association of purchases with specific customers and stores.

Database Outline

- **Authors:** Records the details of authors of the books in the bookstore. An author can write more than one book.
  - `authorID`: int, auto_increment, unique, not NULL, PK
  - `firstName`: varchar(255), not NULL
  - `lastName`: varchar(255), not NULL
  - Relationship(s):
    - M:M relationship between **Authors** and **Books** is implemented with `authorID` as a FK inside **AuthorsBooks** (intersection table).

- **Books:** Records the details of the books in the bookstore. A book can have more than one author.
  - `bookID`: int, auto_increment, unique, not NULL, PK
  - `title`: varchar(255), not NULL
  - `yearOfPublication`: date, not NULL
  - `price`: decimal(12, 2), not NULL
  - Relationship(s):
    - M:M relationship between **Books** and **Authors** is implemented with `bookID` as a FK inside **AuthorsBooks** (intersection table).
    - 1:M relationship between **Books** and **Invoices** is implemented with `bookID` as a FK inside **Invoices**.

- **Invoices:** Records transaction details between the bookstores and customers. Only one book can be purchased in one transaction (invoiceID).
  - `invoiceID`: int, auto_increment, unique, not NULL, PK
  - `date`: date, not NULL
  - `bookID`: int, not NULL, FK
  - `storeID`: int, not NULL, FK
  - `customerID`: int, not NULL, FK
  - Relationship(s):
    - 1:M relationship between **Invoices** and **Books** is implemented with `bookID` as a FK inside **Invoices**.
    - 1:M relationship between **Invoices** and **Stores** is implemented with `storeID` as a FK inside **Invoices**.
    - 1:M relationship between **Invoices** and **Customers** is implemented with `customerID` as a FK inside **Invoices**.

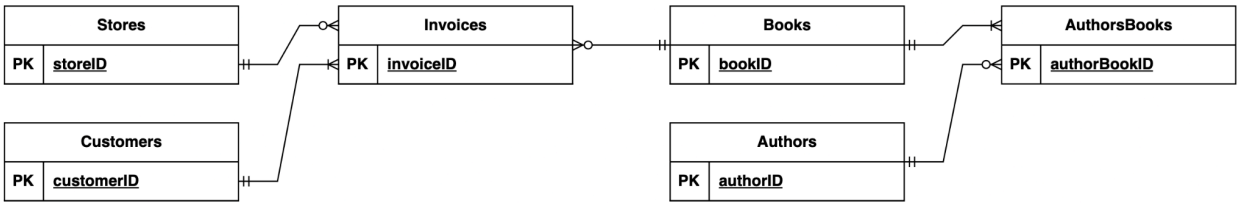
- **Stores:** Records the details of the bookstore locations.
  - `storeID`: int, auto_increment, unique, not NULL, PK
  - `name`: varchar(255), not NULL
  - `phone`: varchar(30), unique, not NULL
  - `address`: varchar(255), not NULL
  - Relationship(s):
    - 1:M relationship between **Stores** and **Invoices** is implemented with `storeID` as a FK inside **Invoices**.

- **Customers:** Records the details of the bookstores' customers information.
  - `customerID`: int, auto_increment, unique, not NULL, PK
  - `firstName`: varchar(255), not NULL
  - `lastName`: varchar(255), not NULL
  - `email`: varchar(100), not NULL
  - `phone`: varchar(30), unique, not NULL
  - Relationship(s):
    - 1:M relationship between **Customers** and **Invoices** is implemented with `customerID` as a FK inside **Invoices**.

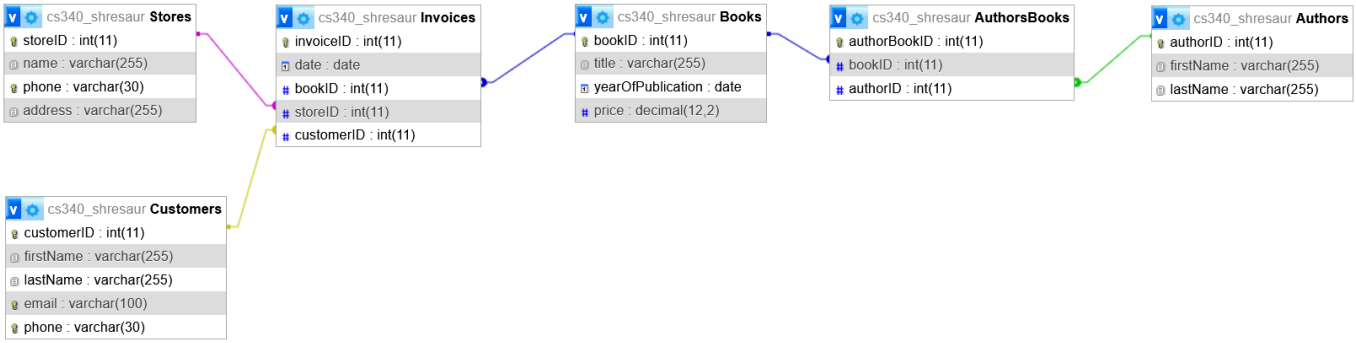
- **AuthorsBooks:** Records the details of each author from **Authors** and the books they've written from **Books**.
  - `authorBookID`: int, auto_increment, not NULL, PK
  - `authorID`: int, not NULL, FK

- `bookID`: int, not NULL, FK
- Relationship(s):
  - 1:M relationship between **AuthorsBooks** and **Authors** is implemented with `authorID` as a FK inside **AuthorsBooks** (intersection table).
  - 1:M relationship between **AuthorsBooks** and **Books** is implemented with `bookID` as a FK inside **AuthorsBooks** (intersection table).

C) ER Diagram:



D) SCHEMA:



E) EXAMPLE DATA:

Stores table:

storeID	name	phone	address
1	Metropolitan Stories	402-547-6787	1 Broad Court, Irmo, SC 29063
2	Literary Books	542-888-2167	121 Main Lane, Pataskala, OH 43062
3	Bookshelf Boutique	901-325-4321	9906 W. Union Street, Burke, VA 22015

Customers table:

customerID	firstName	lastName	email	phone
1	Patricia	Carlson	pcarlson@gmail.com	667-222-4521
2	Selena	Lozada	slozada@gmail.com	701-548-1944
3	Michael	Karlsson	mkarlsson@gmail.com	402-666-1234
4	Sergio	Hernandez	shernandez@gmail.com	801-951-3574

Authors table:

authorID	firstName	lastName
1	Edgar	Patterson
2	William	Fitzgerald
3	George	Wolf
4	Mike	Lang

Books table:

bookID	title	yearOfPublication	price
1	Linear Algebra	2005	80.00
2	Discrete Mathematics	1992	90.99
3	Organic Chemistry	2020	120.00
4	Statistics	2021	115.00

AuthorsBooks table:

authorBookID	bookID	authorID
1	1	1
2	1	2
3	2	2
4	3	3
5	4	NULL

M:M relationship between Authors and Books. A book can have one or many authors (the book with bookID = 1 has two authors (authorID = 1 and authorID = 2). An author can have zero books (authorID = 4 has no books in their name at the store) or many books (authorID = 2 has two books written by them, bookID = 1 and bookID = 2).

Similarly, if an author is deleted from Authors table, the authorID will be set to NULL in AuthorsBooks table. ON DELETE SET NULL operation was used to achieve this.

Invoices table:

invoiceID	date	bookID	storeID	customerID
1	2021-02-25	1	1	1
2	2021-03-19	2	1	1
3	2023-10-10	2	2	2
4	2022-02-05	1	1	4

1:M relationship between Customers and Invoice table. One customer can buy multiple books (customerID = 1 bought bookID = 1 and bookID = 2).

1:1 relationship between Invoice and Customers. Each invoice is associated with one customer.

1:M relationship between Books and Invoices. One book can be purchased by multiple customers (bookID = 2 is purchased by customerID = 1 and customerID = 2).

1:M relationship between Stores and Invoices. One store can sell multiple books (storeID = 1 sold bookID = 1 and bookID = 2).