# FINAL PROJECT PART – 2 FOR ADVANCED DATABASE TECHNOLOGIES

# FALL 2023 - GROUP 26

By team\_VAF

## TITLE: **Rate-My-Reads- A Book Rating Platform**

### *TEAM MEMBER DETAILS:*

* *Viswa Suhaas Penugonda (*[*vpenugon@iu.edu*](mailto:vpenugon@iu.edu)*)*
* *Abhijith Dameruppala (*[*adameru@iu.edu*](mailto:adameru@iu.edu)*)*
* *Fhariya Aseem Fathima (*[*fha@iu.edu*](mailto:fha@iu.edu)*)*

**1, Conceptual Schema for Database**

Our group created "Rate-My-Reads," a platform for rating books with the goal of improving how users discover, evaluate, and engage with books. The foundation of this platform is an intricately organized database that contains an array of information on books, authors, and ratings, giving users the basis to make knowledgeable picks. The crafted conceptual schema diagram of our database reflects a commitment to clarity and organization, acting as a blueprint that outlines the integral relationships between the various entities and their corresponding attributes, ensuring the platform's capabilities effectively meet the diverse needs of our user base.

**Conceptual Diagram:**

**A diagram of a book

Description automatically generated**

**Details:**

**Entities**:

Our database schema consists of the following entities, each representing essential aspects of the book data:

* **Books**: The cornerstone of our schema, this table contains crucial information such as the book ID, title, number of pages, and publisher. It forms the central point from which other entities are related.
* **Authors**: A dedicated table for authors, capturing their unique identifiers and names. This allows us to associate multiple authors with a single book if necessary.
* **Ratings**: This table holds the ratings information, with each record providing the average rating and total ratings count for a book, which is associated by the book ID.
* **ISBN**: This entity manages the ISBN details, ensuring that each book can be uniquely identified through standard international numbering.

**Relations:**

To ensure data consistency and integrity, our schema specifies clear relationships between entities:

* Books and Ratings: A one-to-many relationship exists between books and ratings, where a single book can have multiple ratings entries.
* Books and Authors: A many-to-many relationship is enabled by a linking mechanism (not explicitly depicted in the figure), which allows a book to have several authors and an author to contribute to multiple books.
* Books and ISBN: A one-to-one relationship is established between books and ISBN numbers, with each book having a unique ISBN for identification purposes.

**Indirect Relations:**

While the schema introduces direct relationships via foreign keys, the data organization also reveals implicit relationships:

* Book Details: Knowing the ID of a book allows readers to access detailed information about it, such as authorship and ratings.
* Author Contributions: The author ID can be used to identify all the books to which an author has contributed.
* Rating Insights: The ratings table provides information about how a book was received by the reading community, which can be directly linked to the book details.

The schema has been meticulously built to provide data accessibility and manipulation while conforming to best practices in database design and normalization.

**2. Database Constraints:**

PK – Primary Key – To enforce uniqueness amongst the data.

FK – Foreign Key – To enforce referential integrity amongst the data.

**Primary Key Constraints:** Primary key constraints ensure that each row in a table has a unique identifier.

**Foreign Key Constraints:** Maintains referential integrity by connecting tables and ensuring that foreign key values always refer to a valid record in the connected table.

**Not Null Constraints:** Not null constraints ensure that a column cannot have a null value.

**Check Constraints**: Check constraints ensure that data in a column meets certain conditions.

**Specific Table Constraints:**

Books Table:

* bookID: INT, PRIMARY KEY
* title: VARCHAR(255), NOT NULL
* authors: VARCHAR(255) — While this field is not explicitly constrained to NOT NULL, it's presumed that every book should have at least one author.
* num\_pages: INT
* publisher: VARCHAR(255)

Ratings Table:

* ratingID: INT, PRIMARY KEY, AUTO\_INCREMENT
* average\_rating: DECIMAL(3, 2) — Presumably must be greater than or equal to 0 and not exceed a certain maximum value.
* ratings\_count: INT
* bookID: INT, FOREIGN KEY, NOT NULL

Authors Table:

* authorName: VARCHAR(255), NOT NULL

ISBN Table:

* isbn: VARCHAR(50), PRIMARY KEY
* isbn13: VARCHAR(50)
* bookID: INT, FOREIGN KEY, NOT NULL

**Relational Dynamics:**

The relationships between Books and Ratings, and Books and ISBN are one-to-many and one-to-one, respectively, signifying that each book can have multiple ratings but only one ISBN.

**Views:**

To improve the browsing experience on the "Rate-My-Reads" platform, specialised views within the database have been designed to help users easily discover books based on their preferences. The BooksDesc view displays books ranked by average rating from highest to lowest, highlighting the most popular titles briefly. In contrast, the BooksAsc view organises books beginning with those that received lower ratings, providing an opportunity to discover less popular titles. The BookSizeDesc and BookSizeAsc views categorise books by their length, allowing readers to choose between lengthy stories and quick reads. Furthermore, books with high ratings are easily accessible with a filter applied to the BooksDesc view, ensuring that only the absolute best of the best are highlighted. These views are integral to the platform, streamlining the search process and aligning book suggestions with individual user preferences.

**3. Code to create a database and build queries**

Please refer the attached sql scripts (in canvas) – *project\_part2.sql*

We have created the database, tables, and inserted data into the tables, also created views and we are currently working on stored procedures and functions. We have documented all the queries in the uploaded sql file on Canvas/Github.

**4.Overall Contribution Summary**

|  |  |  |
| --- | --- | --- |
| **Name** | **Task** | **Contribution** |
| Fhariya Aseem Fathima | Building Schema | Aided in the creation of the database schema, focusing on structuring and normalizing data within the books and authors tables. |
| Table Creation & Designing Views | Creating a new table for isbn, reducing redundancy. Designed a view in sql containing frequently parsable data. |
| Constraint Implementation | Implemented foreign key constraints between the isbn and books tables, reinforcing the relational model and data consistency across the database |
| Documentation & Reporting | Responsible for a part of creating comprehensive documentation of the database schema, queries, and procedures, and reporting on database metrics. |
| Abhijith Dameruppala | Documentation & Reporting | Responsible for a part of creating comprehensive documentation of the database schema, queries, and procedures, and reporting on database metrics. |
| Normalization & table creation | Spearheaded the normalization process by creating a new table for Books, reducing redundancy in the books table. |
| Querying Data | Designed some sample queries to fetch data from across the tables. |
| Building Schema | Worked in tandem with the team to establish a robust database schema, ensuring that tables were properly normalized for optimal data management. |
| Viswa Suhaas Penugonda | Documentation & Reporting | Responsible for a part of creating comprehensive documentation of the database schema, queries, and procedures, and reporting on database metrics. |
| Building Schema | Collaboratively designed the foundational structure of the database, contributing to the overall logical workflow and normalization of the schema. |
| Database Ingestion & updating | Handled the loading of book and author data from CSV files into the database and updated book records to include rating IDs, enhancing relational data integrity​. |
| Table Creation & Designing Views | Creating a new table for ratings, reducing redundancy. Designed a view in sql containing frequently parsable data. |