MET CS 669 Database Design and Implementation for Business
Term Project Iteration 4

Name: Megha Samala

Table of Contents

Project Direction Overview

I would like to create a web application designed to track a user's media consumption across several different mediums. The application, which I will call "TunedIn," will be a place where users can log the different media they're currently consuming, have consumed, or plan to consume. The media in question can be movies, TV shows, books, music (in album form), and video games. Additionally, the user will be able to track information about the vendor of the media and use this information to organize and revisit their media buys as well as give recommendations to others

I believe an application like this is important because an overall media tracking system does not appear to exist, leaving people to log their consumption on multiple different websites dedicated to only a specific medium (think Letterboxd or MyAnimeList), or create a Twitter thread or blog dedicated to tracking, or alternatively not track their watches/reads/listens at all. As someone who considers herself to be scatterbrained, I also often jump from media to media without finishing a series or completing a playthrough, and I forget what episode I left on or what chapter of a book I was on, and sometimes I completely forget that I was in the middle of watching or reading something. When people ask me what I've been watching or reading or playing, or what my favorite movie or show is, I often blank and fail to answer the question. I can safely presume that I am not the only one with this issue, and therefore an application dedicated to tracking the status of someone's varied media consumption is a convenient solution. I also believe that tracking media consumption encourages users to think more critically about what they are consuming and how they feel about it, rather than finishing a series or book or game and simply moving on to the next one.

TunedIn will offer a place for users to input information about a piece of media, the main focus being on a name, a rating (if completed/desired), and a status on the consumption. Additional information can be added based on relevance, such as author information for a book, or the URL for something consumed online so it can easily be provided to a friend when recommending the media to them, or a link to another existing database-esque website which may contain a blurb/critic ratings of the media (again, think Letterboxd or MyAnimeList or IMDB, etc), and more. Users will be able to log what chapter of a book they are on, or what episode of a show they are on, so that if they drop it and choose to resume it later, they can easily pick up from where they left off. Additionally, users can track when they started and finished a piece of media, to log past watches, and/or determine if it might be time to revisit something they previously consumed and enjoyed. Each kind of media has information that is specific to it, as well as information that can be applicable to multiple kinds of media, such as a genre and a review. Through my multiple project iterations, I will be able to refine my scope and determine what information is most necessary and relevant for the user to have logged, and create the most useful version of my database.

Use Cases and Fields

Use Case #1: Account Setup

Because TunedIn will be a web application, it will be necessary to register to keep track of each individual user's media.

Field	What it stores	Why it's needed
username	This stores a username	Users can share usernames to

	associated with each account	see what another user is consuming, and users can also have multiple accounts if they would prefer
first_name	This stores the user's first name.	This can be displayed on screen and shared with other users.
last_name	This stores the user's first name.	This can be displayed on screen and shared with other users.
account_created	This stores the date on which the user created the account.	Users may want to know how long they have been using the application to track their media consumption.

Use Case #2: Logging a Piece of Media
This is the supertype that users will log that contains common information across all media types before logging the specific subtype.

Field	What it stores	Why it's needed
media_name	This stores the name of the media.	This is necessary for the user to know what media they consumed.
date_released	This stores the date that the media was released.	This helps the user determine their viewing trends by date and also distinguish between potential media remakes or remasters with different release dates.

Use Case #3: Logging a Movie
A movie will be one of the media types that the application will encourage users to log, and has fields that are specific to it. The other media types will track similar information and will be able to be differentiated in the database using a flag.

Field	What it stores	Why it's needed
director	This stores the name of the director of the movie.	This is useful for the user if they want to track how much they like movies by a certain director

		or want to sort by a certain director.
production_company	This stores the name of the production company of the movie, such as A24 or Studio Ghibli.	This is useful for the user to track how much they like the works of a certain company or sort by them.

Use Case #4: Logging a Genre
Since genres are applicable to any media type, users will be able to log them for any media consumed.

Field	What it stores	Why it's needed
category	This is a catch-all for the different media types where the category/large genres can be stored, such as things like fiction or nonfiction for books, documentary or short film for movies, RPG or shooter for video games, etc.	This allows for users to sort and search for certain pieces of media in a broad way.
main_genre	This stores the primary genre of a piece of media, which can differ in definition based on the user, but is generally a broader category such as horror, action, etc.	This allows for users to sort media by genre when organizing and also looking for previously logged media.
subgenre	This stores the subgenre of a piece of a media, which can be a specifier, like historical fantasy vs urban fantasy, etc.	This is optional but allows for more specific searching and sorting for the user.

Use Case #5: Logging a Review
Since reviews are applicable to any media type, users will be able to log them for any media consumed.

Field	What it stores	Why it's needed
out_of_ten	This stores a number rating out of 10.	This is a typical method of rating not just media, but anything, and will let users reflect on what they enjoyed or disliked.

review	This stores a text review.	This lets the user give clarification on the number rating and offers more information about the media.
review_link	This stores an optional link that leads to an external blog review or a website like Letterbox or Goodreads that the user may have posted their review on.	This lets the user organize relevant info in one place by allowing them to keep track of different websites they may have used to review a piece of media.

Use Case #6: Logging a Status
Each piece of media will have a consumption status that can be logged by the user.

Field	What it stores	Why it's needed
overall_status	This stores a general status like Planned, In Progress, Completed, Dropped, or On Hold.	This helps users keep track of all the media they want to watch, are watching, and have watched, as well as media they chose to not complete and media they plan to return to, which is handy for not forgetting their progress.
section	This stores a larger indicator of progress, such as a season of a TV show, a chapter for a book, a section/act of a video game, etc	This is helpful for giving context to the part of the media the user is on, or is an indicator that they finished it.
subsection	This stores a smaller indicator of progress, such as an episode of a TV show, a page of a book, a level of a video game, etc.	This is helpful for tracking the exact part of a piece of media that the user left off on so they can jump back in.
date_started	This stores the date when the user started consuming the media.	This is helpful for the user to know when they first began a piece of media and for tracking how long it is taking them to complete it.
date_finished	This stores the last date the user interacted with/finished the media.	This is helpful for tracking the amount of time the user has spent with the media and for

		tracking how long ago they interacted with the media to see if it is worth a revisit.

Use Case #7: Logging a Vendor
Media is offered by multiple vendors and these vendors can be frequented by users, and therefore are logged in the database.

Field	What it stores	Why it's needed
vendor_name	This stores the name of the vendor.	This is needed for the user to know what vendors they buyt from.
vendor_type	This stores the type of vendor, such as website, streaming service, or physical storefront.	This is useful for the user to know where they acquired a piece of media.
vendor_link	This stores a link to the vendor website, if available.	This can help the user track past visits to an online vendor and allow them to revisit it or recommend it.

Use Case #8: Vendor-Media Relationship
This connects the vendor to the piece of media and provides useful information about the purchase.

Field	What it stores	Why it's needed
vendor_name	This stores the name of the vendor and will reference the vendor table.	This lets users know where they bought or can buy a piece of media.
date_purchased	This stores the date of the purchase of the media from the vendor.	This can assist with tracking the consumption of media by time.
media_link	This stores a direct link to the media on the vendor site, if available/applicable	This can help the user revisit their purchase easier or recommend the purchase to a friend.
price	This stores the price of the media purchase, if applicable	This can help users budget and track their money spent on media and also play into their

		review of the media by determining if it was worth the price they paid for it.

Use Case #9: Series, Anthology, Collection
Some pieces of media are part of a series or anthology or collection, such as a book series or short film anthology, and users can specify which series a piece of media is part of.

Field	What it stores	Why it's needed
SAC_name	This stores the name of the series, anthology, collection or other extended body of work that a piece of media is part of.	This helps users see how their media is connected as well as track their consumption of a series or the like.
sub_SAC_name	This is optional and stores the name of a subseries/sub-extended body of work within a larger series.	This further helps users organize and search for media.
SAC_part	This is optional and stores a number to indicate which part of a chronological series the piece of media is (such as the second book in a trilogy, etc)	This helps users track completion of a connected/chronological body of media.

Use Case #10: Rating
Most pieces of media have a rating, and users will be able to log this.

Field	What it stores	Why it's needed
rating	This stores the rating of a piece of media, such as G or PG-13 or R in the case of movies, E or T in the case of video games, etc.	This allows users to sort by the rating of the media they are looking for and keep track of their consumption trends.

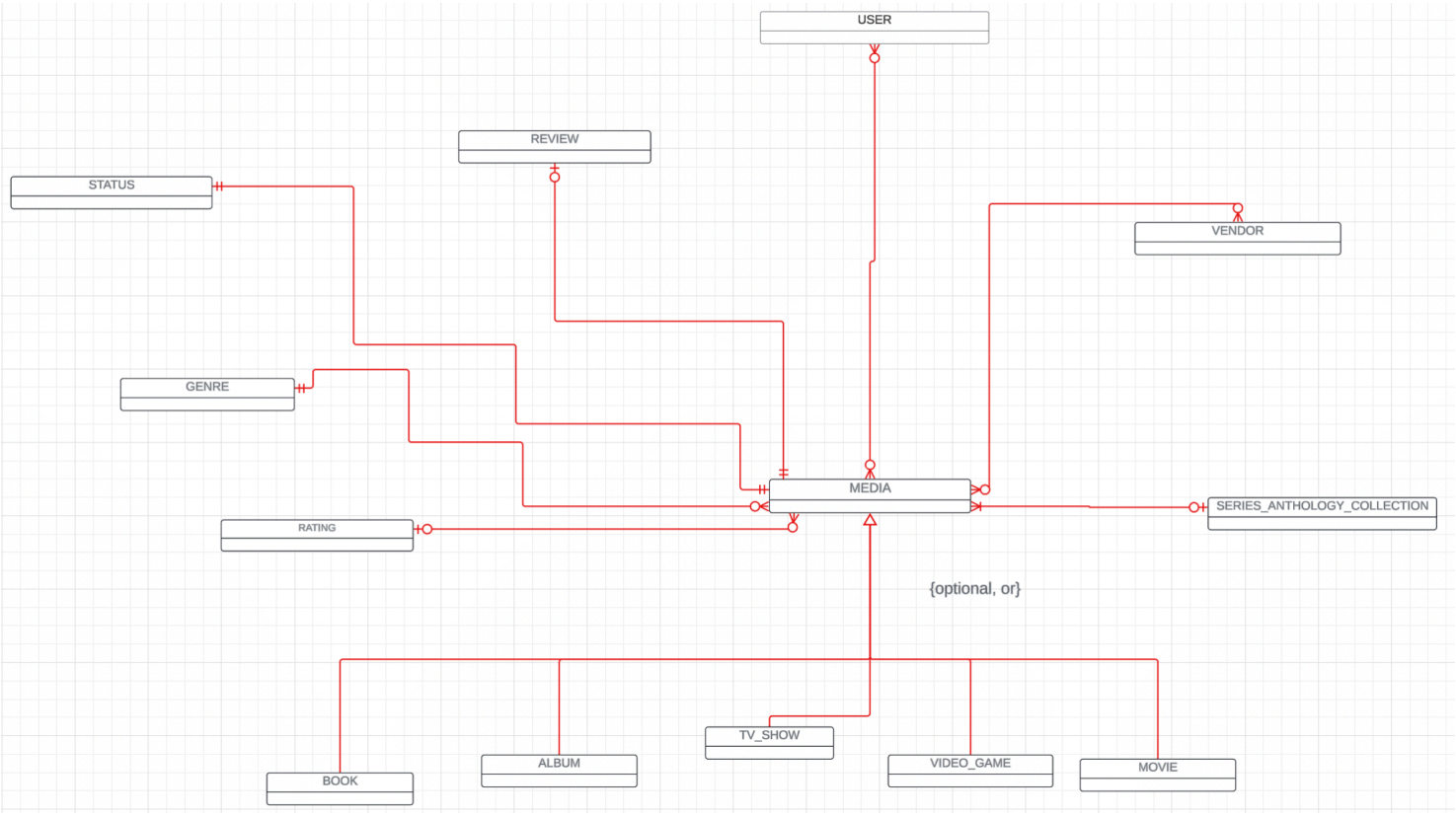
Structural Database Rules

1) Each user may log zero to many pieces of media, and each piece of media may be logged by zero to many users.
2) A piece of media is a book, album, TV show, movie, video game, or none of these.
3) Each piece of media may have one review, and each review will apply to one piece of media.

4) Each piece of media will have one status, and each status will apply to one piece of media.
5) Each piece of media will have one genre, and each genre can apply to zero to many pieces of media.
6) Each piece of media can be part of one series/anthologies/collections, and each series/anthology/collection can contain many pieces of media.
7) Each piece of media can be purchased/rented from zero to many vendors, and each vendor can offer zero to many pieces of media.
8) Each piece of media may have one rating, and each rating can apply to zero to many pieces of media.

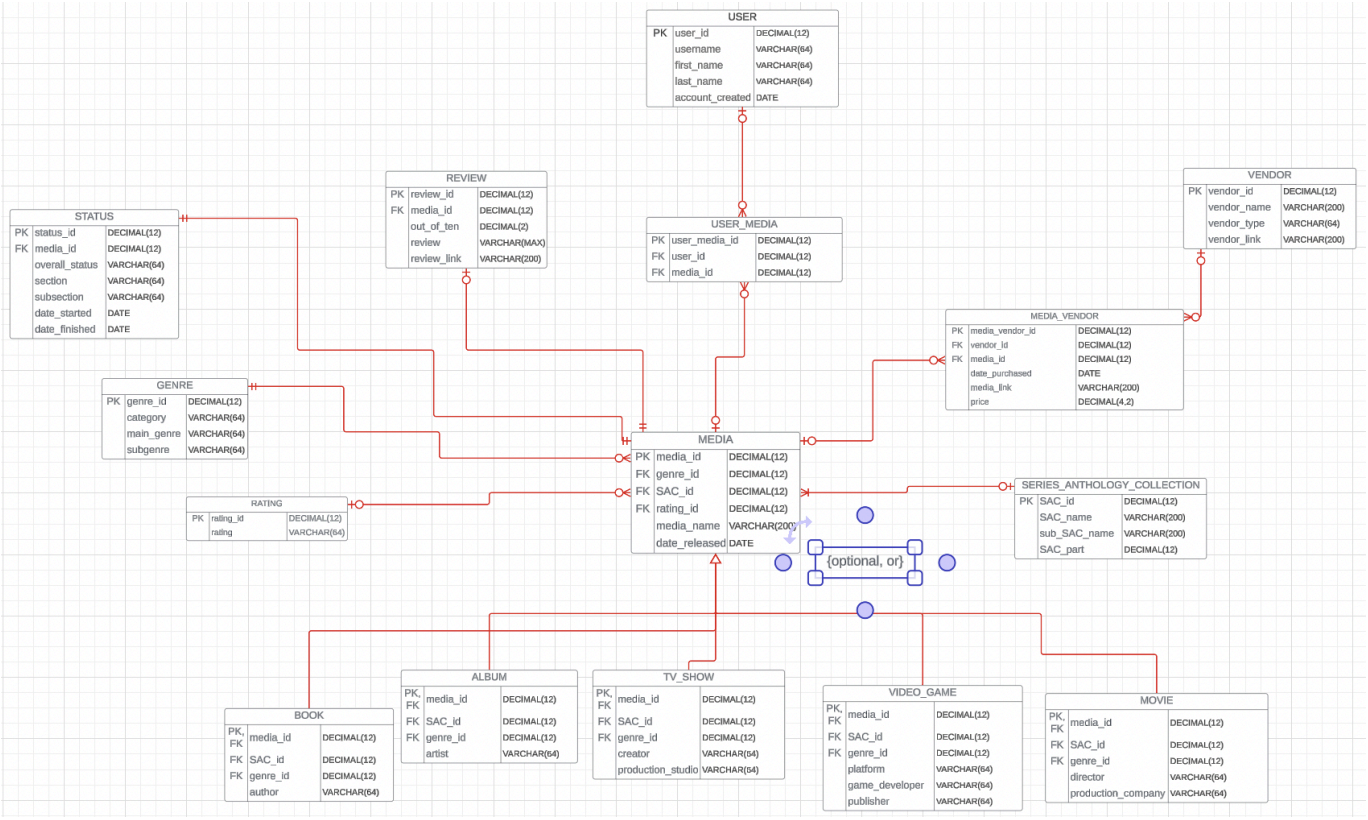
After editing and normalizing my DBMS ERD to reduce redundancy, I was able to reduce the number of structural database rules, and the existing rules are now broader to apply to all media types because the information logged in most of the entities besides the subtypes is universal.

Conceptual Entity-Relationship Diagram



This is my new conceptual ERD that has been considerably changed from the previous iteration in order to make it more legible and to move some of my entity relationships up from the subtypes to the supertype. The conceptual ERD now includes the new Rating entity and excludes a lot of the information I had in it before that would be more appropriate for the physical ERD, such as the bridging entities and attributes.

Full DBMS Physical ERD



The attributes I chose for each field in each entity are pretty self explanatory based on the title of the field. For fields that would contain a smaller number of characters, I used VARCHAR(64) because that seems like common practice, while for fields that might require more characters, such as titles and links, I increased the character limit to 200, and for the review field in the Review entity, I used VARCHAR(MAX) to accommodate a larger number of characters because I looked it up and this is best practice because the TEXT datatype is deprecated. For all of the primary and foreign keys I chose to use DECIMAL(12) because this can accommodate for a lot of entries in the database and is commonly used, and for prices I used DECIMAL(4,2) because I imagine most media will cost less than thousands of dollars. For all of the dates I used DATE, and for out_of_ten in the Review entity I used DECIMAL(2) because the max rating is 10.

Ultimately, I believe that most of my table is BCNF normalized because most determinants are candidate keys, and I have reduced redundancy and increased legibility by moving a lot of the entity relationships from the subtype level up to the supertype level. I would not normalize my ERD further because creating more entities would create unnecessary complexity and I do not believe that it would add anything meaningful to the project at this point.

Summary and Reflection

My database is for a web application called “TunedIn” in which users can track their media consumption across various mediums, as well as media purchases from various vendors. The application allows users to rate their media and track their progress in consuming it.

This week I made a lot of changes to my conceptual and physical ERDs because I changed a lot of the entity relationships and moved them from the subtype level up to the supertype. As a result, my ERDs became more legible and had fewer redundancies, and my structural database rules could be condensed into fewer and broader rules. Additionally, I changed some of the existing relationships between the entities, such as between Media and Vendor, to accommodate for more situations. I also added the Rating entity as one of my use cases and to the rules/ERDs, and I added a use case for the Media supertype entity and moved the date_released and media_name fields to this entity to reduce redundancy. This week I also started actually creating SQL queries for my database, and the beginning portion of my script file is shown below. The full script file will be attached to my submission.

```sql
DROP TABLE IF EXISTS Review;
DROP TABLE IF EXISTS Status;
DROP TABLE IF EXISTS User_media;
DROP TABLE IF EXISTS Media_vendor;
DROP TABLE IF EXISTS Book;
DROP TABLE IF EXISTS Album;
DROP TABLE IF EXISTS TV_show;
DROP TABLE IF EXISTS Video_game;
DROP TABLE IF EXISTS Movie;
DROP TABLE IF EXISTS Vendor;
DROP TABLE IF EXISTS Users;
DROP TABLE IF EXISTS Media;
DROP TABLE IF EXISTS Genre;
DROP TABLE IF EXISTS Series_anthology_collection;
DROP TABLE IF EXISTS Rating;

DROP SEQUENCE IF EXISTS user_seq;
DROP SEQUENCE IF EXISTS user_media_seq;
DROP SEQUENCE IF EXISTS media_seq;
DROP SEQUENCE IF EXISTS media_vendor_seq;
DROP SEQUENCE IF EXISTS vendor_seq;
DROP SEQUENCE IF EXISTS SAC_seq;
DROP SEQUENCE IF EXISTS review_seq;
DROP SEQUENCE IF EXISTS status_seq;
DROP SEQUENCE IF EXISTS genre_seq;
DROP SEQUENCE IF EXISTS rating_seq;

CREATE TABLE Users (
    user_id DECIMAL(12) PRIMARY KEY NOT NULL,
    username VARCHAR(64) NOT NULL,
    first_name VARCHAR(64) NOT NULL,
    last_name VARCHAR(64) NOT NULL,
    account_created DATE NOT NULL
);
CREATE TABLE Vendor (
    vendor_id DECIMAL(12) PRIMARY KEY NOT NULL,
    vendor_name VARCHAR(200) NOT NULL
```

I believe that my physical ERD is pretty solid and I am ready to flesh out my database in SQL, but I am open to any feedback that would help improve my project at this point.