Video Game Inventory Management System

Suyog Nepal
922099890
GitHub: Sicarin

Milestone	Date Submitted
M1	09/18/2024
M2	10/02/2024
M3	10/16/2024

Table of Contents

Project Description

As someone who loves videogames and expanding my game collection, I set out to create a database system that implements features existing in other game inventories, while being able to manage videogames in one place without searching for extra tools across the internet. I want users to automatically find the best deals on games on their wishlist, get recommendations for games based on their favorite genres, and be able to exchange games that they recently purchased, but didn't like, all in one place.

The Video Game Inventory Database will allow hosts to stock video games of various genres, platforms, and pricepoints. Users will be able to maintain, manage, and sort their own game collection, receive deals for making purchases from the host, see the best price of the game from among multiple vendors, refund and exchange games, and get recommended games based on their purchase/playing habits. If the host allows communication between users, users will be able to match with other users who have similar collections/playing habits.

Aside from just refunding them, users will be able to exchange games within a specific time period after purchase, possibly getting a better deal than refunding and buying a new game. They can also exchange a game on one platform for the same game on another platform. Additionally, using past sale history, users will be able to view the next expected sales and prices of a game in the future.

Some existing software products on the market that would benefit from this database system are Steam, Epic Games Store, and IsThereAnyDeal. Steam would benefit by allowing users to find games similar to their favorite games quickly and in an organized way. They could find out how the game is trending and when it is likely to be on sale. IsThereAnyDeal could incentivize users with games that they might personally like based on other similar users' favorite games. This would create a reason besides the price for a user to pick a particular product. The Epic Games Store would benefit by having forums, which would drive engagement from users who want to discuss their games. This could foster a community of discussion or users sharing creations for their favorite games through modding.

Use Cases

Use Case Title:	Exchange games for more value
Actor(s):	Omar
Description:	Omar is a student who has just returned to his dorm on the last day of the Fall semester. After studying and working on final projects for the last several weeks, he is excited to play the newest game by his favorite game studio, so he purchases it on a popular game marketplace and plays for a few hours. He quickly realizes how rushed the game's development was, since it is filled with bugs and glitches, but he wants to give it a shot since it was so anticipated. The next day, when he opens the game, he finds that all his progress has been deleted, and looking around on the internet, he discovers that data loss is one of the critical problems with the new release. When he tries to refund the clearly unplayable game, the marketplace notifies him of an option to exchange it for an older game with similar features, which would cost more if he refunded the new game instead of exchanging. After accepting this exchange, he finds that many of the new game's concepts were likely derived from this older game, which is actually polished and fun. Omar ends up spending the Winter break playing through and enjoying the higher quality game he received through the marketplace's exchange system.

Use Case Title:	Predict Game Releases - Unsuccessful
Actor(s):	Jackie
Description:	Jackie infrequently plays video games, and likes revisiting her old games instead of buying new ones. One day, she logs on to a website that tracks sale histories of games and is notified that DLC(downloadable content), or additional content, for an older game she owns is going to be released in a few days, since other DLCs for the game were released on that week the previous two years, celebrating the anniversary of its launch. The next week, after work, she hurries to her local game store to purchase the DLC, but is dismayed to find that it isn't at the store. Two months later, the DLC is released for purchase, since it has more content than the DLCs of the previous years. The website only used previous trends to project when it would release, but was inaccurate because it didn't account for an unexpected break in the trend.

Use Case Title:	Exchange game platform
Actor(s):	Adrian
Description:	Adrian likes playing a certain multiplayer game on his computer whenever he has free time. When his cousin graduates from high school, Adrian decides to give his PC away to help his cousin study, since he also has an old laptop and game console left. Unfortunately, his laptop is not powerful enough to run his favorite game, so Adrian is unsure how he is going to play it without buying it again for the console. When he checks the game in his collection in the digital game store from which he bought it, he sees a particular option to exchange the game's platform. After looking at the option in detail, he decides to exchange it and receives a download code for the game on his console. Due to this feature from the digital gamefront, Adrian can play his favorite game again without buying it twice, by exchanging the platform from PC to console.

Functional Requirements

1. User
    - A user shall create at most one account
    - A user shall have a unique tracking id
    - A user, without an account, shall have zero games
    - A user shall be able to view news sections
    - A user shall be able to view news articles

2. Account
    - An account shall be owned by one and only one user
    - An account shall have an account id
    - An account shall have a creation date
    - An account shall have a state to define whether the account has been validated or not

3. Registered User
    - A registered user is also a user
    - A registered user has one account
    - A registered user shall have at least one payment method
    - A registered user shall own many games
    - A registered user shall have at most one favorite genre
    - A registered user shall be able to purchase many bundles
    - A registered user shall own many game collection subgroups
    - A registered user shall have one best friend, who is also a registered user
    - A registered user shall be a part of zero or one game party
    - A registered user shall be a part of zero or one game guild
    - A registered user shall be able to create many discussion posts
    - A registered user shall be able to create many discussion replies

4. Game
    - A game shall have a title
    - A game shall have a unique id
    - A game shall be owned by many registered users
    - A game shall be grouped into at least one genre
    - A game shall have many platforms
    - A game shall be developed by one developer
    - A game shall be published by one publisher
    - A game shall be a part of many bundles

- A game shall be contained in many game collection subgroups
- A game shall be the focus of many game parties
- A game shall be sold in many marketplaces

5. Genre
   - A genre shall be categorizing many games
   - A genre shall be the favorite of many registered users
   - A genre shall be the specialty of many developers
   - A genre shall be the specialty of many publishers
   - A genre shall be the focus of zero or many bundles
   - A genre shall be the focus of zero or many game collection subgroups

6. Developer
   - A developer shall focus on many genres
   - A developer shall focus on many platforms
   - A developer shall develop many games

7. Publisher
   - A publisher shall focus on many genres
   - A publisher shall focus on many platforms
   - A publisher shall publish many games

8. Platform
   - A platform shall have many games
   - A platform shall be the focus of many developers
   - A platform shall be the focus of many publishers
   - A platform shall be the focus of many game parties
   - A platform shall be the focus of many game guilds

9. Bundle
   - A bundle shall have at least one game
   - A bundle shall focus on zero or one genre
   - A bundle shall be bought by many registered users
   - A bundle shall be sold in one marketplace

10. Game Collection Subgroup
    - A game collection subgroup shall be owned by one and only one registered user
    - A game collection subgroup shall contain at least one game
    - A game collection subgroup shall focus on zero or one genres

11. Payment Method
    - A payment method shall belong to one registered user
    - A payment method shall have one unique payment method id
    - A payment method shall have one billing address

12. Game Party
    - A game party shall have at least one registered user
    - A game party shall be associated with zero or one game
    - A game party shall be associated with one platform
    - A game party shall be associated with zero or one game guild

13. Game Guild
    - A game guild shall have at least one registered user
    - A game guild shall be associated with one platform
    - A game guild shall have zero or many game parties

14. Forum
    - A forum shall have many discussion posts
    - A forum shall be focused on many topics
    - A forum shall have a name

15. Discussion Post
    - A discussion post shall be focused on many topics
    - A discussion post shall be a part of one and only one forum
    - A discussion post shall be the parent of many discussion replies
    - A discussion post shall be created by one and only one registered user
    - A discussion post shall have a date

16. Discussion Reply
    - A discussion reply shall be in the family of one and only one discussion post
    - A discussion reply shall be the parent of many discussion replies
    - A discussion reply shall be the child of one and only one discussion reply
    - A discussion reply shall be created by one and only one registered user
    - A discussion reply shall have a date

17. Marketplace
    - A marketplace shall have many games
    - A marketplace shall have a name
    - A marketplace shall have many bundles

18. News Section
    - A news section shall have many news articles
    - A news section shall be available for all users to view
    - A news section shall have a state to define whether the user is registered or not

19. News Article
    - A news article shall belong to one and only one news section
    - A news article shall be available for all users to view
    - A news article shall be focused on many topics
    - A news article shall have many authors

20. Credit Card Payment Method
    - A credit card payment method is also a payment method
    - A credit card payment method shall have one bank code
    - A credit card payment method shall have one unique card number
    - A credit card payment method shall have one expiration date
    - A credit card payment method shall have one verification value (CVV or CVC)

Non-Functional Requirements

1. Performance
    - On a desktop browser, the main page of a website using the database system shall render its first content to the screen within 2 seconds.
    - On a desktop browser, the main page of a website using the database system shall return the first byte of its response to a requested resource within 1.5 seconds.
    - On a desktop browser, the main page of a website using the database system shall render its largest visible image, text block, or video element within 2.5 seconds.

2. Security
    - Only encrypted passwords shall be supported by the database system.
    - All registered users shall be required to pass multi-factor authentication checks when logging in to access an application/website that uses the database system.
    - The database shall be automatically backed up every day at 11:59 pm.

3. Scalability
    - The database system shall be able to handle a 50% increase in average concurrent users every 2 years.
    - The database system shall be able to handle 1,000 more games than the previous year added each year to its marketplace.
    - The database system shall be able to handle an average of 17% increase in game transactions every year.

4. Capability
    - The database system shall have the capability to add games to a marketplace without disrupting the experience of current users.
    - The database system shall have the capability to list how many players are currently in a game if it is linked to a game launcher.
    - The database system shall have the capability to allow registered users to customize parts of their UI elements, and cater their marketplaces or news feeds through filters.

5. Environmental
    - The database system shall be able to display a registered user’s collection in an application without an internet connection.
    - A marketplace using the database system shall be updated even if a user is not using its application or website.

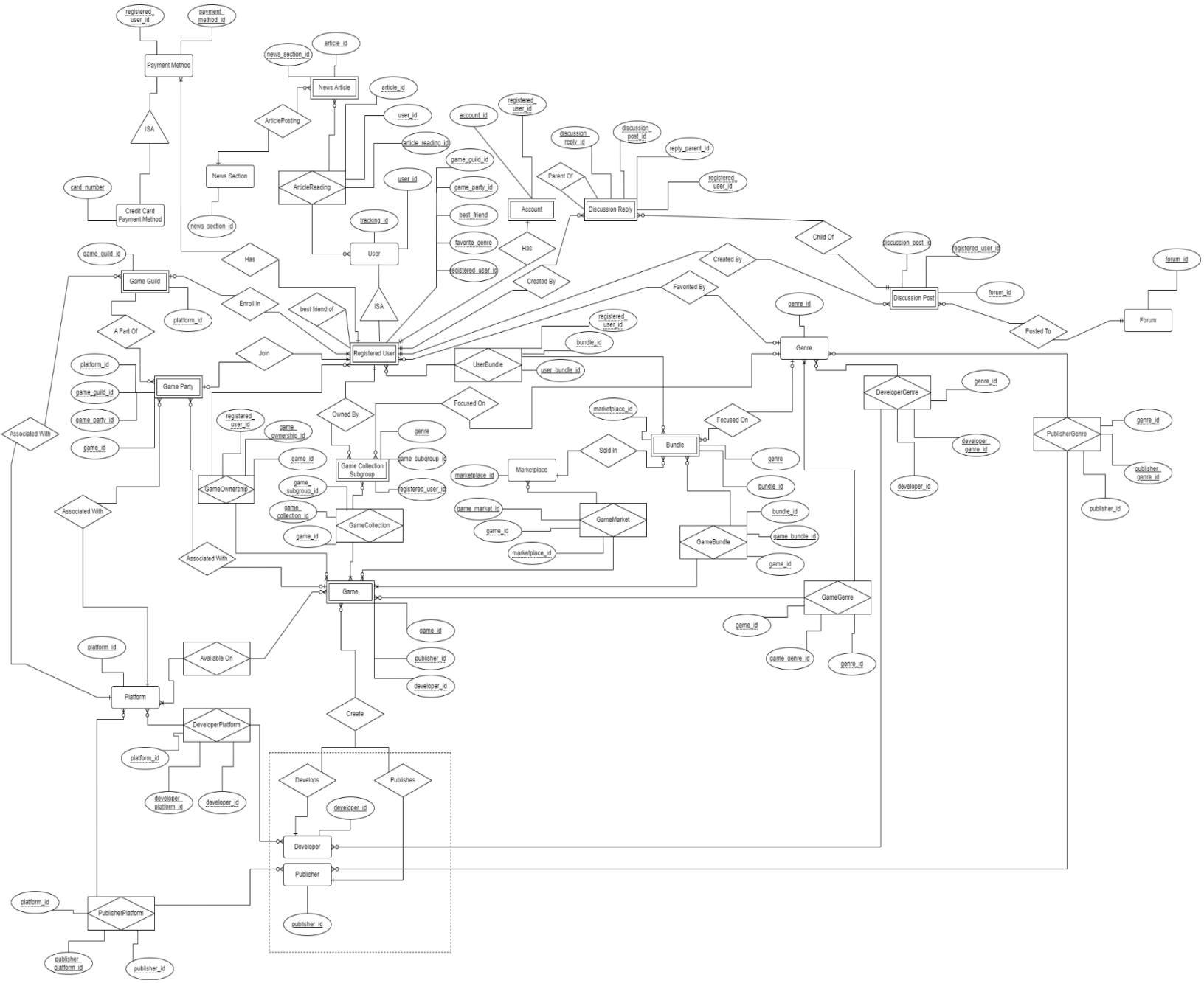
- The database system will not allow a user to browse a forum if there is no internet connection.

6. Coding Standard
   - The database system shall have descriptive or clearly understandable names and identifiers.
   - The database system shall not have abbreviations in names unless they are widely understood.
   - The database system shall have names in the singular, not plural.

7. Storage
   - The database system should support persistent storage.
   - The database shall hold space for 1000 different games by default in a marketplace utilizing it.
   - The database shall have space for 20 games in the collection of each registered user by default.

8. Privacy
   - The database system shall prevent unregistered users from viewing the profile of registered users.
   - The database system shall allow all users to view the prices of games between different vendors.
   - The database system shall prevent unregistered users from accessing forums.

Entity Relational Diagram



Entity Set Description

1. User (Strong)
    a. user_id: numeric, primary key
    b. tracking_id: numeric, unique key
    c. first_time_user: BOOLEAN
    d. has_access: BOOLEAN
2. Registered User (Weak)
    a. registered_user_id: numeric, primary key
    b. favorite_genre: numeric, foreign key (references Genre entity key)
    c. best_friend: numeric, foreign key (references Registered User entity set, since a Registered User is the best friend of another Registered User)
    d. game_party_id: numeric, foreign key (references Game Party entity set)
    e. game_guild_id: numeric, foreign key (references Game Guild entity set)
3. Account (Weak)
    a. account_id: numeric, primary key
    b. registered_user_id: numeric, foreign key (references Registered User entity set)
    c. creation_date: date
    d. is_validated: BOOLEAN
4. PaymentMethod (Weak)
    a. payment_method_id: numeric, primary key
    b. registered_user_id: numeric, foreign key (references Registered User entity set)
    c. billing_address: alphanumeric
5. Credit Card Payment Method (Weak)
    a. card_number: numeric, primary key
    b. bank_code: alphanumeric
    c. verification_value: numeric (CVC or CVV)
    d. expiration_date: composite, date
6. Game (Weak)
    a. game_id: numeric, primary key
    b. developer_id: numeric, foreign key (references Developer entity set)
    c. publisher_id: numeric, foreign key (references Publisher entity set)
    d. game_title: alphanumeric
7. GameOwnership (Associative, Weak)
    a. game_ownership_id: numeric, primary key
    b. game_id: numeric, composite primary key and foreign key (references the Game entity set)
    c. registered_user_id: numeric, composite primary key and foreign key (references the Registered User entity set)
    d. total_games_owned: numeric
8. Genre (Strong)
    a. genre_id: numeric, primary key

    b. genre_name: alphanumeric
    c. custom_genre: BOOLEAN (A custom genre created by the client, as opposed to standard ones like “action” or “adventure”)
9. GameGenre (Associative, Weak)
    a. game_genre_id: numeric, primary key
    b. game_id: numeric, composite primary key and foreign key (references the Game entity set)
    c. genre_id: numeric, composite primary key and foreign key (references the Genre entity set)
    d. subgenre: alphanumeric
10. Game Party (Weak)
    a. game_party_id: numeric, primary key
    b. game_guild_id: numeric, foreign key (references Game Guild entity set)
    c. platform_id: numeric, foreign key (references Platform entity set)
    d. game_id: numeric, foreign key (references Game entity set)
    e. created: date, composite
11. Platform (Strong)
    a. platform_id: numeric, primary key
    b. platform_name: alphanumeric
    c. release_date: date, composite
12. Game Guild (Weak)
    a. game_guild_id: numeric, primary key
    b. platform_id: numeric, foreign key (references Platform entity set)
    c. guild_name: alphanumeric
    d. guild_color: alphanumeric
13. Forum (Strong)
    a. forum_id: numeric, primary key
    b. forum_name: alphanumeric
    c. topic: multivalue, alphanumeric
14. Discussion Post (Weak)
    a. discussion_post_id: numeric, primary key
    b. forum_id: numeric, foreign key (references Forum entity set)
    c. registered_user_id: numeric, foreign key (references Registered User entity set)
    d. topic: multivalue, alphanumeric
    e. created: date, composite
15. Discussion Reply (Weak)
    a. discussion_reply_id: numeric, primary key
    b. discussion_post_id: numeric, foreign key (references Discussion Post entity set)
    c. reply_parent_id: numeric, foreign key (references Discussion Reply entity set, since a reply can be the child of another reply)
    d. registered_user_id: numeric, foreign key (references Registered User entity set)
    e. created: date, composite
16. Developer (Strong)
    a. developer_id: numeric, primary key

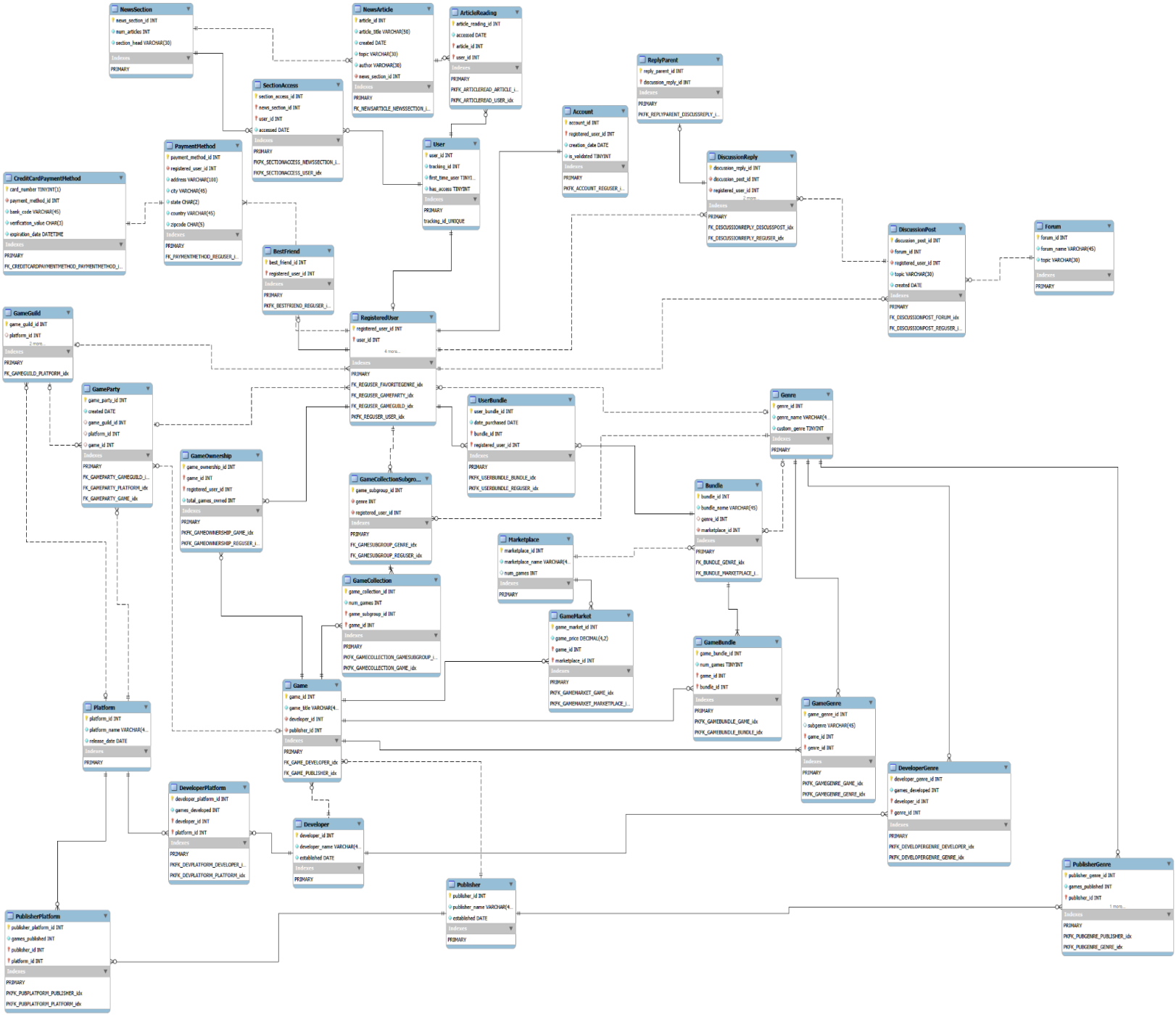
    b. developer_name: alphanumeric
    c. established: date, composite
17. DeveloperGenre (Associative, Weak)
    a. developer_genre_id: numeric, primary key
    b. developer_id: numeric, composite primary key and foreign key (references the Developer entity set)
    c. genre_id: numeric, composite primary key and foreign key (references the Genre entity set)
    d. games_developed: numeric (games of this Genre developed by this Developer)
18. DeveloperPlatform (Associative, Weak)
    a. developer_platform_id: numeric, primary key
    b. developer_id: numeric, composite primary key and foreign key (references the Developer entity set)
    c. platform_id: numeric, composite primary key and foreign key (references the Platform entity set)
    d. games_developed: numeric (games developed by this Developer on this Platform)
19. Publisher (Strong)
    a. publisher_id: numeric, primary key
    b. publisher_name: alphanumeric
    c. established: date, composite
20. PublisherGenre (Associative, Weak)
    a. publisher_genre_id: numeric, primary key
    b. publisher_id: numeric, composite primary key and foreign key (references the Publisher entity set)
    c. genre_id: numeric, composite primary key and foreign key (references the Genre entity set)
    d. games_published: numeric (games of this Genre published by this Publisher)
21. PublisherPlatform (Associative, Weak)
    a. publisher_platform_id: numeric, primary key
    b. publisher_id: numeric, composite primary key and foreign key (references the Publisher entity set)
    c. platform_id: numeric, composite primary key and foreign key (references the Platform entity set)
    d. games_published: numeric (games published by this Publisher on this Platform)
22. Bundle (Weak)
    a. bundle_id: numeric, primary key
    b. genre: numeric, foreign key (references Genre entity set)
    c. marketplace_id: numeric, foreign key (references Marketplace entity set)
    d. bundle_name: alphanumeric
23. UserBundle (Associative, Weak)
    a. user_bundle_id: numeric, primary key
    b. bundle_id: numeric, composite primary key and foreign key (references the Bundle entity set)

    c. registered_user_id: numeric, composite primary key and foreign key (references the Registered User entity set)
    d. date_purchased: date, composite
24. GameBundle (Associative, Weak)
    a. game_bundle_id: numeric, primary key
    b. game_id: numeric, composite primary key and foreign key (references the Game entity set)
    c. bundle_id: numeric, composite primary key and foreign key (references the Bundle entity set)
    d. num_games: numeric
25. Game Collection Subgroup (Weak)
    a. game_subgroup_id: numeric, primary key
    b. genre: numeric, foreign key (references Genre entity set)
    c. registered_user_id: numeric, foreign key (references Registered User entity set)
26. GameCollection (Associative, Weak)
    a. game_collection_id: numeric, primary key
    b. game_subgroup_id: numeric, composite primary key and foreign key (references the Game Collection Subgroup entity set)
    c. game_id: numeric, composite primary key and foreign key (references the Game entity set)
    d. num_games: numeric
27. Marketplace (Strong)
    a. marketplace_id: numeric, primary key
    b. marketplace_name: alphanumeric
    c. num_games: numeric
28. GameMarket (Associative, Weak)
    a. game_market_id: numeric, primary key
    b. game_id: numeric, composite primary key and foreign key (references the Game entity set)
    c. marketplace_id: numeric, composite primary key and foreign key (references the Marketplace entity set)
    d. game_price: numeric
29. News Section (Strong)
    a. news_section_id: numeric, primary key
    b. num_articles: numeric
    c. section_head: alphanumeric
30. News Article (Weak)
    a. article_id: numeric, primary key
    b. news_section_id: numeric, foreign key (references News Section entity set)
    c. article_title: alphanumeric
    d. created: composite, date
    e. topic: multivalue, alphanumeric
    f. author: composite, alphanumeric
31. ArticleReading (Associative, Weak)

a. article_reading_id: numeric, primary key
b. article_id: numeric, composite primary key and foreign key (references the News Article entity set)
c. user_id: numeric, composite primary key and foreign key (references the User entity set)
d. accessed: date, composite

32. SectionAccess (Associative, Weak)
    a. section_access_id: numeric, primary key
    b. news_section_id: numeric, composite primary key and foreign key (references the News Section entity set)
    c. user_id: numeric, composite primary key and foreign key (references the User entity set)
    d. accessed: date, composite

Enhanced Entity-Relationship (EER) Diagram



Normalization Techniques Used

1. PaymentMethod:

   In this table, I originally had a primary key “payment_method_id”, a foreign key “registered_user_id” referencing the RegisteredUser table, and a non-key attribute, “billing_address”, which had a VARCHAR(150) datatype. The table has unique records and no repeating groups, but it does not meet First Normal Form (1NF) because “billing_address” is not atomic. To make this attribute more atomic, I split it up into “address”(just the street and house number), “city”, “state”, “country”, and “zipcode”. With this, I believe PaymentMethod now meets 1NF. This will help organize my data to be more precise, as specific parts of the address can be found, and its integrity will be aided by the fact that these parts can be more easily modified in the future.

2. NewsSection:

   This table was already in 1NF because it had unique records, no repeating groups, and has all atomic attributes. However, it doesn’t meet Second Normal Form (2NF) because one of its non-key attributes, “user-registered” (BOOLEAN), is not completely dependent on the primary key, “news_section_id”. To make it reach 2NF, I replaced the “user-registered” attribute with a new one, “section_head”(VARCHAR(30)). Since section_head is completely dependent on each specific section, it helps NewsSection to meet 2NF. I also had to create a new entity, SectionAccess, to create a direct relationship between NewsSection and User, and more easily verify whether User is registered while accessing the news section.