CS 340 Project: Game Library
Nathan Perkins
Heather Godsell

Game Library Web App

Project Outline

At Nathan's work, there is an extensive library of video games that employees can check out for free and return whenever convenient. Currently, the library is managed by one person, manually on an Excel spreadsheet. We will work on a game library web app that can be used by employees to examine the current library inventory and to make requests. The heart of the web app will be a database that contains all the information about the users, games, game metadata, platforms and requests.

The project will be written in node.js, HTML and CSS. We will use hand-written MySQL queries to manage all database access and modifications.

Project Feedback and Changes

Feedback by the peer reviewer

Peer Review 1
The ER diagram was well designed with all the attributes and tables present that were defined in the project description. The primary keys were underlined and the relationships between the tables were accurate according to the outline.

The schema was also well designed with no error that I could see.

The SQL file ran with no issues, was syntactically correct, and possessed all the required attributes and tables that were described in the project description.

Peer Review 2

1. Are the attributes for each entity in the ERD same as that described in the database outline? Perhaps, There is a difficulty with the release id. I see release date; however, I am not sure if that is the release id. The descriptor for the release id is confusing as to what it connects to. There is a confusing id, called title id, I do not see it in the ERD. Platform_ID is stated as an attribute however I do not see it in the ERD. However, I believe there might be

some confusion in regars to what a foreign key would look like in an ERD. So I am aware of there existence through the schema and not ERD

2. Is the participation of entities in the relationships same as that described in the outline? Yes the relationships demonstrate what the outline had described.

3. Is the cardinality of entities in the relationships same as that described in the outline? Yes the cardinality matches the outline

4. Is there something that could be changed/improved in the E R Diagram and/or the overall database design? I would suggest that the relationships demonstrate which item is connecting them, and showing how the foreign key relates to other entities.

The best peer review for a Schema would answer all of the following questions:
1. Are the relationship tables present where required and correctly defined, when compared with the database outline? Yes the schema has all the information that the outline states, and also has demonstrates the relationships.

2. Are foreign keys present where required and correctly defined, when compared with the database outline? Yes the foreign keys are present and point to the accurate entity.

3. Do the entity attributes match those described in the outline? Yes the entities show all the attributes that were declared in the outline.

4. Is there something that could be changed/improved in the Schema and/or the overall database design? I think it looks great, it even has the lines that overlap to show a line is unique to an entity.

The ideal peer review for a DDQ file would answer all of the following questions:
1. Is the SQL file syntactically correct? This can be easily verified by importing/copy-pasting it in phpmyadmin. (Do not forget to take backup of your own database before you do this!) The SQL file is syntactically correct. It runs without any errors.

2. Are the data types appropriate considering the description of the attribute in the database outline? Yes the attributes are similar to how it has been defined in the outline.

3. Are the foreign keys correctly defined when compared to the Schema? Yes each foreign key is present and works correctly.

4. Are relationship tables present when compared to the ERD/Schema? Yes the tables are present and seem to work properly.

Peer Review 3

ERD
Your outline defines the cardinality of Game Release to Users a many to many but you used a cardinality of N. This should be M since the user could have 0 game releases.
Wrong notation for 1 to many relation between Game Requests and Game Copies. You used a P when it should be M.

Schema
Your foreign keys need to be underlined in Game Request, Game Release, and Game Copy.

DBQ
None of the tables are populated as required by the project. “Sample Data: And we also need you to submit INSERT queries to populate your Project database with sample data. All data types should be appropriate, foreign keys should exist and be correct and it should match the database outline, schema, and ERD that you submit. Again, these sample data statements should be easily run-able as stated above.”
Game Copies Table Library_tag is defined as being UNIQUE in your table but your outline does not define it as such. Outline does not define length of library_tag. Table has it set to 255. Dt_created and dt_upadated are not defined in your outline or ER Diagram for game copies.
Game Platforms Table Dt_created and dt_updated are not defined in your outline or ER Diagram for game platform Outline does not have defined sizes for varchar of length 255 for both Name and Manufacturer. Table has them set to 255.
Game Releases Table Rating is able to be set to NULL but is not defined as such in outline. Boxart_url varchar of length 255 length is 255 but is not defined as such in outline. Dt_created and dt_upadated are not defined in your outline or ER Diagram Game Request Table Dt_created and dt_upadated are not defined in your outline or ER Diagram Game Titles Table Name uses varchar of length 255 of size 255 but is not defined as such in outline. Name is specified as being unique but not defined as such in outline.
Genre uses varchar of length 255 of size 255 but is not defined as such in outline. Developer uses varchar of length 255 of size 255 but is not defined as such in outline.
Producer uses varchar of length 255 of size 255 but is not defined a such in outline.
Dt_created and dt_upadated are not defined in your outline or ER Diagram Users Table First_name, last_name, e-mail, password, password_salt all use varchar of length 255 of size 255 but this is not defined in the outline. Dt_created and dt_upadated are not defined in your outline or ER Diagram.

Actions based on the feedback

1. Regarding the foreign keys missing from the ERD, we decided not to take any action after doing additional research to confirm that foreign keys belong in the schema, but not the ERD.
2. Regarding the cardinality of relationships in the ERD, we decided not to take any action as it seems the reviewer is mistaken about notation practices. M, N, and P are all used to represent “many” relationships and signify that they can all be different numbers.

3. Regarding the suggestion to underline foreign keys in the schema, we decided not to take any action, citing the examples given in our class lectures.
4. Regarding the tables not being populated, we rectified the situation to meet the project specifications.
5. Regarding the many attribute parameters and constraints present in our DDQ but missing from our outline, we have added those parameters and constraints to the outline for consistency.
6. Regarding the dt_created and dt_updated attributes, while we feel that these are purely for maintenance and don't describe the entities, we've decided to err on the side of caution and include them in the schema and ERD.

Additional Fixes

1. Removed password_salt attribute from user entity since it is not needed when using bcrypt.
2. Changed game_request.game_id to game_request.copy_id for consistency.
3. Changed game_releases.rating to DECIMAL(5,2).

Project Change

We decided to change our project from a database representing the game **Stardew Valley** to this game library project because we felt that we wanted to complete a project that would demonstrate higher level skills and be more impressive on our resumes. The game library project is an improvement because:

1. It will be used by people in the real world, requiring us to make real decisions about which features that people will find useful. We will also be externally motivated to maintain it and fix bugs as necessary.
2. It has the potential to pull in data via real-world APIs (MobyGames) or through screen/DOM scraping (game review sites), demonstrating skills that are desirable in the job market.

Database Outline

Entities

- **All Entities:**
    - All entities will have these fields:
    - id: an auto-incremented int(11) identifying the user; cannot be NULL; used as the primary key

    - dt_created: current timestamp marking when the entity was created, cannot be NULL
    - dt_updated: current timestamp marking when the entity was last updated, cannot be NULL

- **Users:**
    - Describes a user of the web app and their role.
    - first_name: a varchar of length 255 string containing the user's first name; cannot be NULL
    - last_name: a varchar of length 255 string containing the user's last name; cannot be NULL
    - email: a varchar of length 255 string containing the user's email; cannot be NULL. Is Unique.
    - password: a varchar of length 255 string containing the user's hashed password; cannot be NULL
    - role: an enum describing the user's role. Cannot be NULL
        - 'User', 'admin', 'root'
- **Game_Copies:**
    - Describes an actual physical copy of a video game that is in the library.
    - status: an enum describing the status of this copy; cannot be NULL
        - Available, checked_out, lost
    - release_id: a foreign key linking a copy of a game to the metadata about a release of that game; cannot be NULL.
    - library_tag: a varchar of length 255 containing the library tag that is stuck to the game box; cannot be NULL
    - dt_procured: a date field containing the date this copy was purchased; can be NULL.
- **Game_Titles:**
    - Describes metadata related to a game title, but not a specific platform.
    - name: a varchar of length 255 string containing the game's name; cannot be NULL; must be unique
    - description: a TEXT field containing a description of the game; can be NULL
    - genre: a varchar of length 255 string containing the game's genre; can be NULL
    - developer: a varchar of length 255 string containing the game's developer; can be NULL
    - producer: a varchar of length 255 string containing the game's producer; can be NULL

- **Game_Platforms:**
    - Describes a game platform, like a specific console family or a PC.
    - Name: A varchar of length 255 string with the name of this platform; cannot be NULL. Is Unique.

- Manufacturer: A varchar of length 255 string with the manufacturer of this platform; can be NULL
- Release date: A date field with the release date of this platform; can be NULL

Relationships

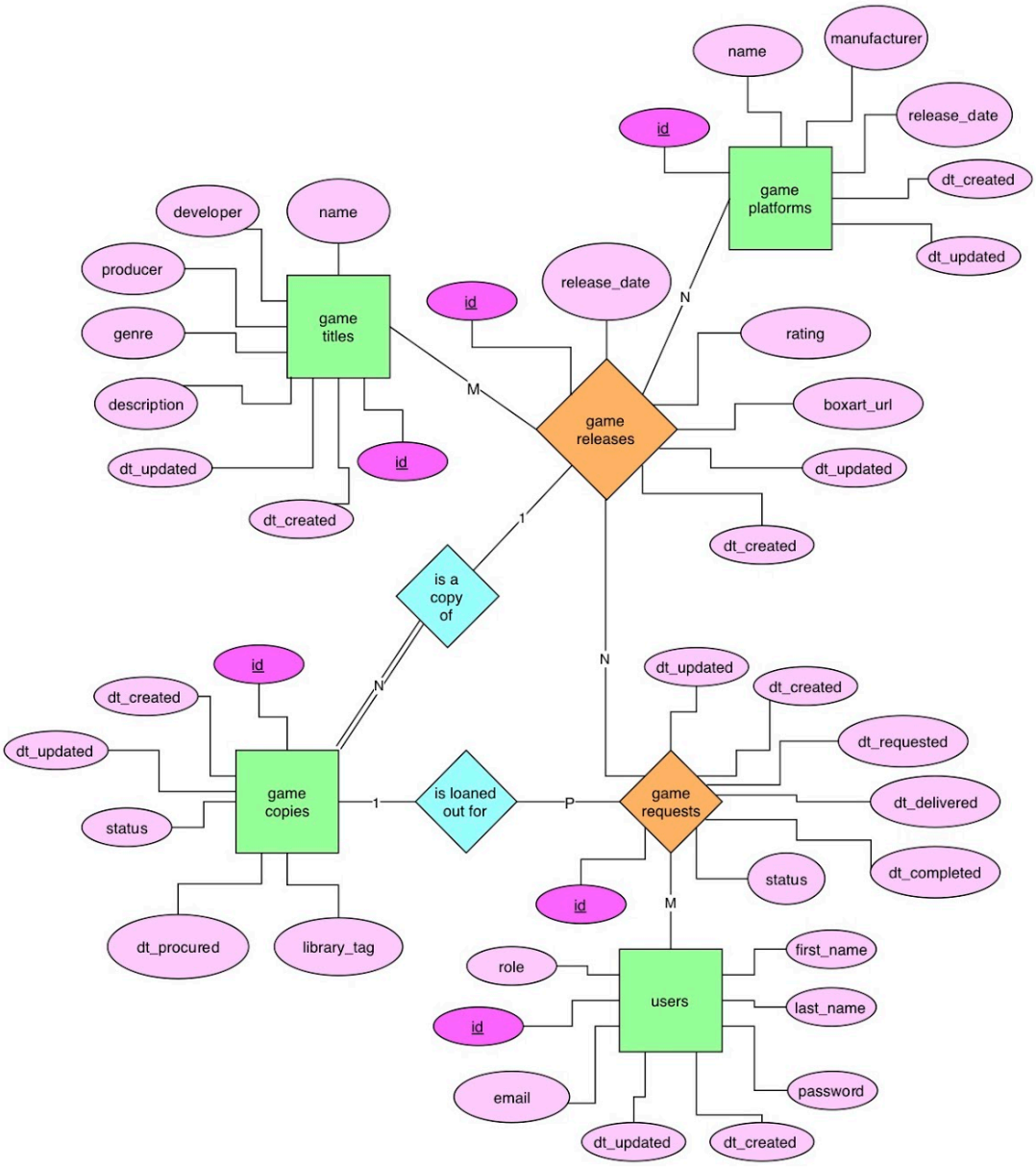
- **Game_Releases - game_titles, and game_platforms**
  - Many-to-many relationship between game_titles and game_platforms.
  - A game_Title can have many game_platforms through game_releases.
  - A game_platform can have many game_titles through game_releases.
  - Contains additional data about the release, with the following attributes:
    - title_ID: A foreign key linking a game release to GameTitle about that release; cannot be NULL
    - platform_ID: A foreign key linking a game release to the platform of that release; cannot be NULL
    - release_date: A date field representing the release date; cannot be NULL
    - rating: a decimal field representing the review score of this release; can be NULL.
    - boxart_url: A varchar of length 255 string with a URL link to boxart for this release; can be NULL

- **Game_Requests - users, game_releases**
  - Many-to-Many relationship involving users and game_releases, describing a request made by a user to borrow a game, and the status of that request until completion.
  - A user can make many requests for a game_release.
  - A game_release can have many requests by many users.
  - Contains additional data about the request, with the following attributes:
    - user_ID: foreign key linking a request to a specific user. Cannot be NULL
    - release_ID: foreign key linking a request to a release of a game. Used for the request initially, before an actual copy of the game is delivered. Cannot be NULL
    - copy_ID: foreign key linking a request to a specific copy of a game, once it has been delivered. Can be NULL
    - dt_requested: datetime field for when the game was requested by the user; can be NULL
    - dt_delivered: datetime field for when the game was delivered to the user; can be NULL
    - dt_completed: datetime field for when the game was returned to the library or the request cancelled; can be NULL
    - status: an enum describing the status of the request. Cannot be NULL
      - 'pending', 'checked_out', 'completed'

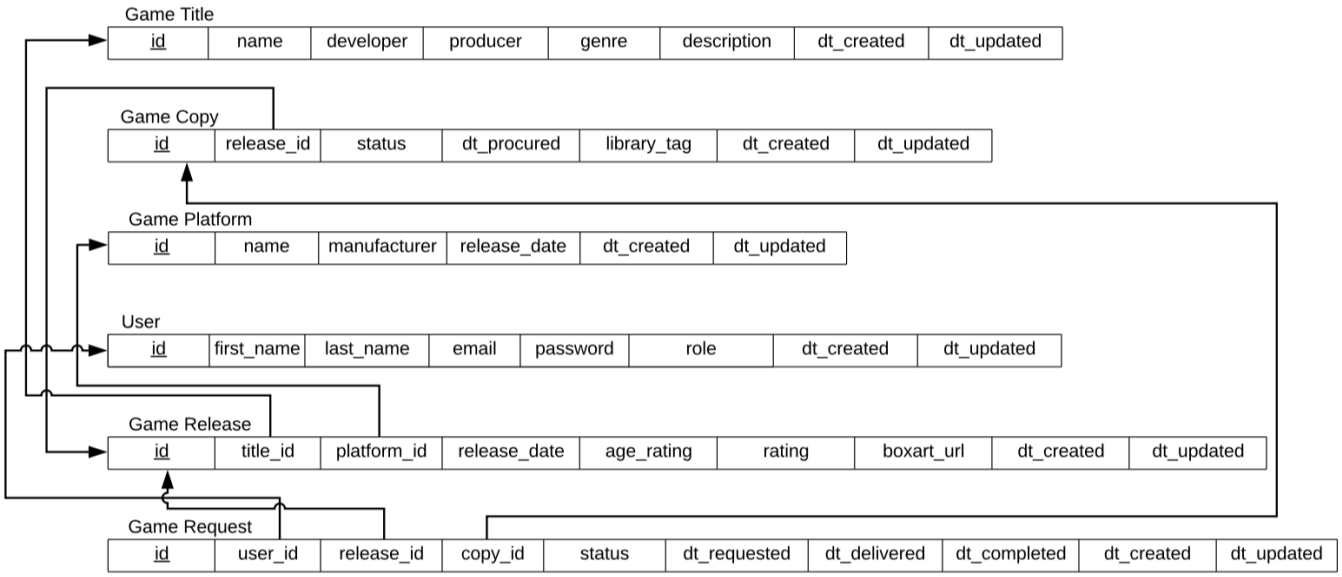
- **Adding a game to the library - game_copies and game_releases**
    - One to many relationship involving a game game_release and a game copy.
    - A game_copy must be a purchased physical copy (instance) of a game_release.
    - A game_release can have many purchased copies.

- **Loaning a game out - game_requests and game_copies**
    - One-to-many relationship involving a game_request and a game_copy.
    - A game_request can be fulfilled by loaning out one game_copy.
    - A game_copy can fulfill many requests (one at a time).

Entity-Relationship Diagram



Schema


Game Title
id
name
developer
producer
genre
description
dt_created
dt_updated
Game Copy
id
release_id
status
dt_procured
library_tag
dt_created
dt_updated
Game Platform
id
name
manufacturer
release_date
dt_created
dt_updated
User
id
first_name
last_name
email
password
role
dt_created
dt_updated
Game Release
id
title_id
platform_id
release_date
age_rating
rating
boxart_url
dt_created
dt_updated
Game Request
id
user_id
release_id
copy_id
status
dt_requested
dt_delivered
dt_completed
dt_created
dt_updated