# CS340 – Project Step 2 Draft

Group 101 | Cameron Brooks & Brayden Plumb

Winter 2026

## a) Fixes Based on Feedback from Step 1

We received four peer reviews on our Step 1 Draft submission. Below is a summary of the feedback and actions taken.

**Reviewers:** Kai Sherman, Olivia Choi, Bryant Aragon, Lucas Feldsien.

### Actions Based on Feedback

1. **Naming Consistency – Singular to Plural** (Kai Sherman): Changed all entity names from singular to plural (Game → Games, Player → Players, RunCategory → RunCategories, Platform → Platforms, RunSubmission → RunSubmissions).
2. **Intersection Table Naming Inconsistency** (Kai Sherman): Standardized the intersection table name to "GamesOnPlatforms" across both the ERD and all documentation.
3. **Game-RunCategory Relationship Wording** (Kai Sherman, Lucas Feldsien): Clarified that the relationship is 1:M from Games to RunCategories: one Game can have multiple RunCategories, but each RunCategory belongs to exactly one Game.
4. **Annual Run Estimates** (Kai Sherman, Olivia Choi): Added estimate of approximately 750–3,000 run submissions annually based on 25–100 players × 1–30 runs per player.
5. **Entity Highlighting in Overview** (Kai Sherman): Added colored highlighting to entity names in the Overview section.
6. **Verification Date Attribute** (Olivia Choi): Added `verifiedDate: date` attribute to RunSubmissions.
7. **User Roles Clarification** (Olivia Choi): Clarified that the website will be "editable by admins, and viewable by players."
8. **Run Category Edge Cases** (Olivia Choi): Confirmed each RunSubmission must belong to exactly one category (NOT NULL FK).

### Additional Design Changes for Step 2

- **Normalization Verified**: Applied the normalization process to all tables (details in Normalization section below). All tables satisfy 3NF.

- **CASCADE Constraints**: Added ON DELETE CASCADE and ON UPDATE CASCADE to all foreign key constraints, as recommended by the course materials.
- **Schema Diagram**: Created based on the DDL implementation.
- **Example Data**: Added 3–5 rows of sample data per table that demonstrate all relationship types in action.

---

## b) Project Overview and Database Outline – Updated Version

### Overview

*The following overview is a fictional scenario we are using as a framework for our project.*

**Northern Oregon Speedrunners Association** is a nonprofit organization that runs various video game speedrunning events in local communities in Northern Oregon. They would like to have a website where they can store the various speedrun submissions from their competitions, which typically consist of 25–100 players, depending on the competition. The amount of runs submitted per year will vary drastically, depending on activity in the local speedrunning community, but on average most players will submit 1–30 runs per year. Based on these figures, the system is expected to handle approximately 750–3,000 **run submissions** annually.

Due to the limited needs of this system, they don't need a way to categorize the runs by the event they are a part of, and storing them by date is enough information for them to infer the event they are from. This then allows users to submit runs to be verified that they may complete from home by sending the info about their run to an administrator of the association.

The association only wants **run submissions** to be able to be submitted by admin board members of the association. They would like to be able to see the name of the **players**, the runs they have successfully completed, what **game** they speedran, what **run categories** these runs are in, and what **platform** they were on when they completed the run. This information will serve as a record for all local speedrunners in the area to show off their previous times, and gives **Northern Oregon Speedrunners Association** a record to determine local speedrunning records. This website will be editable by admins, and viewable by players.

### Database Outline

**Players**

*Description: A speedrunner participant (via a displayName) that can speedrun various games.*

- **playerID**: int, auto_increment, unique, not NULL, PK

- **displayName**: varchar(100), not NULL, unique – The player's username/unique handle
- **country**: varchar(100) – The player's country of residence

**Relationship(s):**

- 1:M with RunSubmissions with playerID as a FK in RunSubmissions (one Player can have many RunSubmissions)

---

## Games

*Description: A video game that is able to be speedrun.*

- **gameID**: int, auto_increment, unique, not NULL, PK
- **title**: varchar(255), not NULL – The name of the video game
- **releaseYear**: YEAR – Year the game was released
- **developer**: varchar(255) – The company that developed the game

**Relationship(s):**

- 1:M with RunSubmissions with gameID as a FK in RunSubmissions (one Game can have many RunSubmissions)
- 1:M with RunCategories with gameID as a FK in RunCategories (one Game can have multiple RunCategories)
- M:N with Platforms via *GamesOnPlatforms* intersection table with both gameID and platformID as FKs (one Game can be on many Platforms; one Platform can have many Games)

---

## RunCategories

*Description: A speedrun submission type which different games provide.*

- **runCategoryID**: int, auto_increment, unique, not NULL, PK
- **name**: varchar(100), not NULL – The category name (e.g., "Any%", "Glitchless", "100%")
- **ruleset**: text – Description of rules for this category

- **gameID**: int, not NULL, FK – References Games.gameID

**Relationship(s):**

- 1:M with RunSubmissions with runCategoryID as a FK in RunSubmissions (one RunCategory can have many RunSubmissions)
- M:1 with Games with gameID as a FK in this RunCategories table (each RunCategory belongs to one and only one Game)

---

## Platforms

*Description: The hardware or environment a game can be played on.*

- **platformID**: int, auto_increment, unique, not NULL, PK
- **name**: varchar(100), not NULL, unique – The platform name (e.g., "PC", "Nintendo Switch", "PlayStation 5")

**Relationship(s):**

- 1:M with RunSubmissions with platformID as a FK in RunSubmissions (one Platform can have many RunSubmissions)
- M:N with Games via *GamesOnPlatforms* intersection table with both platformID and gameID as FKs (one Platform can have many Games; one Game can be on many Platforms)

---

## RunSubmissions

*Description: A specific speedrun record that has all the information about that speedrun attempt.*

- **runSubmissionID**: int, auto_increment, unique, not NULL, PK
- **runTime**: time(3), not NULL – The duration of the speedrun (HH:MM:SS.mmm)
- **submissionDate**: date, not NULL – The date the run was submitted
- **verified**: boolean, default FALSE – Whether the run has been verified by an admin
- **verifiedDate**: date – The date the run was verified (NULL if not yet verified)
- **playerID**: int, not NULL, FK – References Players.playerID

- **gameID**: int, not NULL, FK – References Games.gameID
- **platformID**: int, not NULL, FK – References Platforms.platformID
- **runCategoryID**: int, not NULL, FK – References RunCategories.runCategoryID
- **videoLink**: varchar(2048) – URL to video proof of the run

**Relationship(s):**

- M:1 with Players with playerID as a FK in this RunSubmissions table (many RunSubmissions belong to one Player)
- M:1 with Games with gameID as a FK in this RunSubmissions table (many RunSubmissions belong to one Game)
- M:1 with Platforms with platformID as a FK in this RunSubmissions table (many RunSubmissions were played on one Platform)
- M:1 with RunCategories with runCategoryID as a FK in this RunSubmissions table (many RunSubmissions are in one RunCategory)

---

**GamesOnPlatforms** *(Intersection Table)*

*Description: Tracks which games are available on which platforms (M:N relationship).*

- **gameOnPlatformID**: int, auto_increment, unique, not NULL, PK
- **gameID**: int, not NULL, FK – References Games.gameID
- **platformID**: int, not NULL, FK – References Platforms.platformID

**Constraint:** Unique combination of (gameID, platformID)

## Relationship Summary

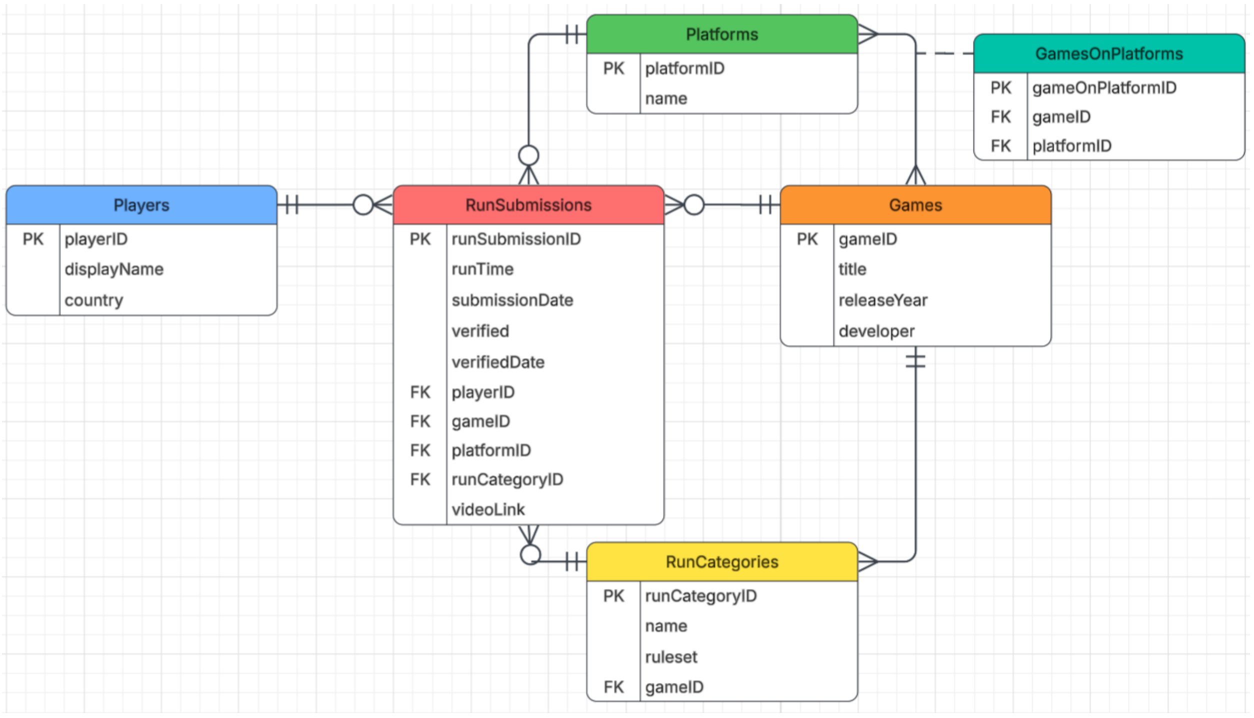
| **Relationship** | **Type** | **Implementation** |
|---|---|---|
| Players → RunSubmissions | 1:M | playerID FK in RunSubmissions |
| Games → RunSubmissions | 1:M | gameID FK in RunSubmissions |
| Platforms → RunSubmissions | 1:M | platformID FK in RunSubmissions |
| RunCategories → RunSubmissions | 1:M | runCategoryID FK in RunSubmissions |
| Games ↔ Platforms | M:N | GamesOnPlatforms intersection table |
| Games → RunCategories | 1:M | gameID FK in RunCategories |

**Total: 6 relationships (including 1 M:N) across 6 tables**

---

## c) Entity-Relationship Diagram

*The ER diagram below demonstrates the relationships between all entities. Only primary keys are shown in the ERD; full attributes are listed in the Database Outline above and the Schema Diagram below.*



*Note: The ERD matches the Database Outline and Schema Diagram. All entity names are plural, the intersection table is named GamesOnPlatforms, and the verifiedDate attribute has been added to RunSubmissions.*

---

## Normalization Analysis

We applied the normalization process as described in the Exploration – Normalization Steps. We started by examining a hypothetical sample report (similar to the construction company example from the exploration) and checked for redundancies and anomalies.

### Hypothetical Denormalized Report

Consider a flat report that an admin might want to generate – listing all run submissions with full details:

| Player | Country | Game | Developer | Category | Platform | RunTime | Verified |
|---|---|---|---|---|---|---|---|
| SpeedDemon42 | US | Super Mario 64 | Nintendo | Any% | N64 | 15:35 | Yes |
| SpeedDemon42 | US | Super Mario 64 | Nintendo | 16 Star | N64 | 49:12 | Yes |
| PixelRunner | CA | Zelda: OoT | Nintendo | Any% | Switch | 18:22 | No |
| GlitchHunter | US | Celeste | Maddy Makes | Any% | PC | 32:45 | Yes |
| NightOwlPlays | JP | Portal | Valve | Inbounds | PC | 14:08 | No |

## Redundancy and Anomaly Analysis

If all data were stored in a single flat table:

- **Redundant Data**: "SpeedDemon42" and "US" are repeated for each run by that player. "Nintendo" is repeated for each game developed by Nintendo. "Super Mario 64" is repeated for each run of that game.
- **Insert Anomaly**: To add a new game, we would need to create a dummy run submission. To add a new player, we would need a dummy run.
- **Update Anomaly**: Changing a player's country would require updating every row with that player. Changing a game's developer would require updating every run for that game.
- **Delete Anomaly**: Deleting the only run for a game would lose all information about that game. Deleting the only run for a player would lose that player's information.

## 1NF Verification

All tables are in 1NF:

- Every table has a defined primary key (no repeating groups).
- All attributes contain atomic values (single value per cell, no lists or sets).
- No table has repeating groups – each row is uniquely identifiable by its PK.

## 2NF Verification

All tables are in 2NF:

- No table has a composite primary key with partial dependencies. Players, Games, Platforms, RunCategories, and RunSubmissions all use single-column auto-increment primary keys, so partial dependency is impossible.
- The GamesOnPlatforms table has a surrogate PK (`gameOnPlatformID`) and a unique constraint on (`gameID`, `platformID`). The table contains no non-key attributes beyond the two foreign keys, so there are no partial dependencies.

### 3NF Verification

All tables are in 3NF:

- **Players**: `playerID` → `displayName`, `country`. No transitive dependency (country does not determine any other non-key attribute).
- **Games**: `gameID` → `title`, `releaseYear`, `developer`. No transitive dependency (developer is an independent non-key attribute and does not functionally determine `title` or `releaseYear`; multiple games can share the same developer while having different titles and release years, so there is no transitive dependency among non-key attributes).
- **Platforms**: `platformID` → `name`. Only one non-key attribute; no transitive dependency possible.
- **RunCategories**: `runCategoryID` → `name`, `ruleset`, `gameID`. The gameID is a FK, not a transitive determinant. The name and ruleset are specific to the category, not determined by gameID alone (one game has multiple categories with different names and rulesets).
- **RunSubmissions**: `runSubmissionID` → all other attributes. All non-key attributes (runTime, submissionDate, verified, verifiedDate, videoLink) are facts about the specific run submission, not about any other non-key attribute. The FKs (playerID, gameID, platformID, runCategoryID) reference other tables – they do not transitively determine each other.
- **GamesOnPlatforms**: Contains only the surrogate PK and two FKs. No non-key attributes to create transitive dependencies.

**Conclusion**: All six tables satisfy 3NF. No denormalization decisions were necessary, as the schema was already properly decomposed from the initial design in Step 1.

---

## d) Schema Diagram

The schema diagram below represents the final normalized database design. All tables, their attributes, primary keys (PK), and foreign keys (FK) are shown, along with the relationships between tables.

*Note: This schema matches the Database Outline and ERD exactly. All foreign keys use ON DELETE CASCADE and ON UPDATE CASCADE constraints.*

| **Players** | | |
|---|---|---|
| **Attribute** | **Type** | **Constraints** |
| playerID | int | PK, auto_increment, NOT NULL |
| displayName | varchar(100) | NOT NULL, UNIQUE |
| country | varchar(100) | |

| **Games** | | |
|---|---|---|
| **Attribute** | **Type** | **Constraints** |
| gameID | int | PK, auto_increment, NOT NULL |
| title | varchar(255) | NOT NULL |
| releaseYear | YEAR | |
| developer | varchar(255) | |

| **Platforms** | | |
|---|---|---|
| **Attribute** | **Type** | **Constraints** |
| platformID | int | PK, auto_increment, NOT NULL |
| name | varchar(100) | NOT NULL, UNIQUE |

| **RunCategories** | | |
|---|---|---|
| **Attribute** | **Type** | **Constraints** |
| runCategoryID | int | PK, auto_increment, NOT NULL |
| name | varchar(100) | NOT NULL |
| ruleset | text | |
| gameID | int | FK → Games.gameID, NOT NULL |

| **RunSubmissions** | | |
|---|---|---|
| **Attribute** | **Type** | **Constraints** |
| runSubmissionID | int | PK, auto_increment, NOT NULL |
| runTime | time(3) | NOT NULL |
| submissionDate | date | NOT NULL |
| verified | boolean | DEFAULT FALSE |
| verifiedDate | date | (NULL if unverified) |
| playerID | int | FK → Players.playerID, NOT NULL |
| gameID | int | FK → Games.gameID, NOT NULL |
| platformID | int | FK → Platforms.platformID, NOT NULL |
| runCategoryID | int | FK → RunCategories.runCategoryID, NOT NULL |
| videoLink | varchar(2048) | |

| **GamesOnPlatforms** *(Intersection Table)* | | |
|---|---|---|
| **Attribute** | **Type** | **Constraints** |
| gameOnPlatformID | int | PK, auto_increment, NOT NULL |
| gameID | int | FK → Games.gameID, NOT NULL |
| platformID | int | FK → Platforms.platformID, NOT NULL |
| *UNIQUE constraint on (gameID, platformID)* | | |

### Schema Relationships

| | | |
|---|---|---|
| Players | $1 : M$ | RunSubmissions |
| Games | $1 : M$ | RunSubmissions |
| Games | $1 : M$ | RunCategories |
| Platforms | $1 : M$ | RunSubmissions |
| RunCategories | $1 : M$ | RunSubmissions |
| Games | $M : N$ | Platforms *(via GamesOnPlatforms)* |

---

## e) Example Data

The following tables show the sample data inserted into each table. This data is included in the accompanying DDL.sql file via INSERT statements. The data demonstrates how foreign keys enforce 1:M relationships and how the intersection table shows the M:N relationship in action.

### Players (5 rows)

| playerID | displayName | country |
|---|---|---|
| 1 | SpeedDemon42 | United States |
| 2 | PixelRunner | Canada |
| 3 | GlitchHunter | United States |
| 4 | NightOwlPlays | Japan |
| 5 | TurboTina | United Kingdom |

*Note: TurboTina (playerID=5) has no RunSubmissions, demonstrating that a Player can exist without runs.*

### Games (4 rows)

| gameID | title | releaseYear | developer |
|---|---|---|---|
| 1 | Super Mario 64 | 1996 | Nintendo |
| 2 | The Legend of Zelda: Ocarina of Time | 1998 | Nintendo |
| 3 | Celeste | 2018 | Maddy Makes Games |
| 4 | Portal | 2007 | Valve |

### Platforms (4 rows)

| platformID | name |
|---|---|
| 1 | PC |
| 2 | Nintendo 64 |
| 3 | Nintendo Switch |
| 4 | PlayStation 5 |

*Note: PlayStation 5 (platformID=4) has no games in GamesOnPlatforms, demonstrating that a Platform can exist without associated games.*

### RunCategories (5 rows)

| runCategoryID | name | ruleset | gameID (FK) |
|---|---|---|---|
| 1 | Any% | Complete the game as fast as possible by any means. | 1 |
| 2 | 16 Star | Complete the game collecting exactly 16 stars. | 1 |
| 3 | Any% | Complete the game as fast as possible by any means. | 2 |
| 4 | Any% | Complete the game as fast as possible by any means. | 3 |
| 5 | Inbounds | Complete all chambers without leaving intended boundaries. | 4 |

*Note: Game 1 (Super Mario 64) has two categories (Any% and 16 Star), demonstrating the 1:M relationship from Games to RunCategories. Each category name like "Any%" can appear for different games with different rulesets, but each row belongs to exactly one game.*

### GamesOnPlatforms (7 rows) – Intersection Table

| gameOnPlatformID | gameID (FK) | platformID (FK) |
|---|---|---|
| 1 | 1 | 2 |
| 2 | 1 | 3 |
| 3 | 1 | 1 |
| 4 | 2 | 2 |
| 5 | 2 | 3 |
| 6 | 3 | 1 |
| 7 | 4 | 1 |

*M:N demonstration: Game 1 (Super Mario 64) appears on 3 platforms (N64, Switch, PC). Platform*

*1 (PC) hosts 3 games (Super Mario 64, Celeste, Portal). This shows the many-to-many relationship in action.*

### RunSubmissions (5 rows)

| ID | runTime | subDate | verified | verifiedDate | playerID | gameID | platID | catID |
|---|---|---|---|---|---|---|---|---|
| 1 | 00:15:35.230 | 2026-01-10 | TRUE | 2026-01-11 | 1 | 1 | 2 | 1 |
| 2 | 00:49:12.800 | 2026-01-15 | TRUE | 2026-01-16 | 1 | 1 | 2 | 2 |
| 3 | 00:18:22.450 | 2026-01-20 | FALSE | NULL | 2 | 2 | 3 | 3 |
| 4 | 00:32:45.110 | 2026-02-01 | TRUE | 2026-02-02 | 3 | 3 | 1 | 4 |
| 5 | 00:14:08.670 | 2026-02-03 | FALSE | NULL | 4 | 4 | 1 | 5 |

*Note: Column headers abbreviated for readability. Full column names: runSubmissionID, runTime, submissionDate, verified, verifiedDate, playerID, gameID, platformID, runCategoryID. The videoLink column is omitted from the table for space but is included in the DDL.sql file.*

**Relationship demonstrations:**

- **1:M Players→RunSubmissions**: Player 1 (SpeedDemon42) has 2 runs (IDs 1 and 2), while Player 5 (TurboTina) has 0 runs.
- **1:M Games→RunSubmissions**: Game 1 (Super Mario 64) has 2 runs (IDs 1 and 2).
- **1:M RunCategories→RunSubmissions**: Each run belongs to exactly one category.
- **NULL verifiedDate**: Runs 3 and 5 are unverified, so verifiedDate is NULL.
- **NULL videoLink**: Run 4 has no video link, showing that videoLink is optional.

---

## Citations

None. All content of this proposal is our original work.