Steam: A deconstruction of a virtual video-game store

A Database Foundations Project

1st Juan Pablo Borja Espitia
Universidad Francisco José de Caldas
System engineering
Bogotá D.C, Colombia
20202020091

2nd Laura Marcela Santana Sánchez
Universidad Francisco José de Caldas
System engineering
Bogotá D.C, Colombia
20222020165

***Abstract*—This paper deconstructs Steam's database model, exploring its architecture and design. Steam, developed by Valve Corporation, is one of the most prominent platforms for video game distribution. The project investigates the structure behind Steam's database. The goal is to determine how the system is designed to manage the platform's vast game library and user interactions.**

***Index Terms*—Database, Model, ERM, Steam, Architecture**

I. Introduction

Steam, developed by Valve Corporation in 2003, originally designed to distribute Valve's games, Steam evolved into a platform that now hosts thousands of games from various developers worldwide.[1] Steam operates on a commission-based business model, where Steam takes a percentage cut from all the sales made on its platform. Steam and Amazon are considered to have utilized and brought this model to the forefront via each of their wildly successful online [distribution] stores. Steam also lead the way in the freemium model during its first few years of operations – it offered free-to-play games to boost its growth and reach a wider audience. This helped bring in more users to the platform who would eventually stick to Steam – since it was the only major service that had a lot of game titles on offer.[3]
The objective of this research is to analyze how Steam's database is structured, focusing on its core components. By studying its Entity Relationship Model (ERM), this project aims to provide a foundational understanding of how Steam organizes its vast user base and game library. Furthermore, insights gained from this analysis can guide future database design projects.

II. The Entity Relationship Model

A. What is ERM?

The Entity Relationship Model (ERM) also known as Entity Relationship Diagram (ERD), is a type of diagram for data modeling, which graphically illustrates the interrelationships of the entities of a database system. Said model, was proposed by the American-Taiwanese theoretical computer scientist, Peter Pin-Shan Chen in 1976.[2] The project follows a structured methodology involving ten steps for creating an ERM, which visualizes the interrelations between database entities. These steps include defining components, identifying entities and attributes, establishing relationships, and producing diagrams to illustrate the data structure.

Identify applicable funding agency here. If none, delete this.

III. The ten steps of the ERM

A. Step 1: Define Components

The first step in creating an ERM is identifying the major components or pillars of the system being modeled. These components serve as the foundational blocks for the database model, defining what the system is intended to manage.

B. Step 2: Define Entities

Once the key components were defined, the next step was to break these components down into smaller, more detailed entities. Entities represent distinct objects within the system that need to be tracked in the database. These entities will each be represented as a table in the database, with attributes representing their characteristics.

C. Step 3: Define Attributes per Entity

Each entity has specific properties, called attributes, which describe it in more detail. This step is crucial because attributes will later determine the fields (or columns) in each table of the database.

D. Step 4: Define Relationships

Entities do not exist in isolation; they are interconnected. In this step, the relationships between different entities are identified. These relationships indicate how the entities will interact with each other in the database.

E. Step 5: Define Relationship Types

Relationships between entities are further refined by defining their types. These include one-to-one, one-to-many, and many-to-many relationships, this will determine the model's complexity on how the data flows

F. Step 6: First ERM Diagram

At this stage, the first Entity-Relationship Diagram (ERD) is created. This diagram visually represents the entities, their attributes, and the relationships between them. This initial diagram provides a clear visualization of the database design and serves as a blueprint for future improvement.

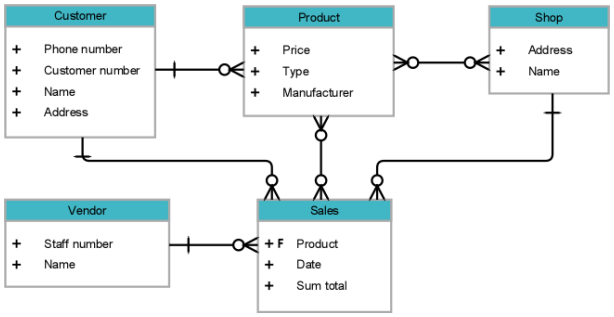

Fig. 1. Example of an ERD

G. Step 7: First Division of Many-to-Many Relationships

When dealing with many-to-many relationships, it becomes necessary to break these into smaller, more manageable pieces. This process involves creating junction tables that handle the many-to-many connections, at this will make sure that the database can efficiently handle complex relationships without redundancy.

H. Step 8: Second ERM Diagram

After resolving the many-to-many relationships, a second ERM diagram is created, incorporating the changes made in step 7. This second diagram provides a more accurate and refined view of the database model.

I. Step 9: Get ERM Data Structure

The ERM data structure refers to the set of rules that govern the relationships between entities, particularly how data will flow between them. In this step, decisions are made about cardinality (i.e., the number of times an entity can be related to another entity) and integrity constraints that ensure data consistency. This structure helps to enforce the rules for how entities and relationships are managed in the database, ensuring data is organized properly.

J. Step 10: Define Constraints and Properties of Data

Finally, constraints and properties of the data are defined. These include primary keys, foreign keys, data types, and other constraints like NOT NULL and UNIQUE that ensure the integrity of the database, these constraints will protect the database from redundancy and ensure accurate data retrieval.

IV. Results and Application

A. Step 1: Define Components

As the first step of an ERM, the main components that are very relevant into the Steam Platform are:

- Games
- Users
- Communities

B. Step 2: Define Entities

Each entity will be assigned with a code to index them without writing the name of the entities entirely

- User - E1
- Game - E2
- Genre - E3
- Category - E4
- Badge - E5
- Message - E6
- Community - E7
- Developer - E8
- Achievement - E9
- Forum - E10
- Review - E11
- DLC (Downloadable content) - E12

C. Step 3: Define Attributes per Entity

Note that this attributes only have the names of the attributes, which means that they're not ready to be any type of keys until step 7.

User (E1):
- Nickname
- Friends
- Shopping Cart
- Downloads
- Notifications
- Messages
- Wallet
- Wishlisted
- Email
- Password
- Level
- Country
- Creation Date
- SteamId
- Status
- Phone
- Subscriptions
- Name

Game (E2):
- ID
- Price
- Release Date
- Update Log
- Name
- Description
- Languages
- Achievements
- Category
- Genre
- Review
- DLCs
- System Requirements
- Community
- Developer
- Age Restriction
- Size

Genre (E3):
- Id
- Name

Category (E4):
- Id
- Name

Badge (E5):
- Id
- Name
- Experience
- Description
- Icon

Message (E6):
- Id
- Title
- Content
- Author
- Date

Community (E7):
- Id
- Subscribers
- Posts
- Associated Game

Developer (E8):
- Id
- Biography
- Associated Games

Achievement (E9):
- Name
- ID
- Description
- Icon

Forum (E10):
- Messages
- Title
- Author
- Date

Review (E11):
- Rating
- Comment
- User
- Date

DLC (E12):
- Name
- Price
- Associated Game
- Size

Fig. 2. Entities List of Attributes

D. Step 4: Define Relationships

The following table contains all the involved entities, where each cell represents the possible relationship with another entity, the red cell indicates a relationship between two entities.

	E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12
E1												
E2												
E3												
E4												
E5												
E6												
E7												
E8												
E9												
E10												
E11												
E12												

Fig. 3. Relationships Table

E. Step 5: Define Relationship Types

After the relations are set, the relationships must have a quantity of how many entities are related to another.

E1 Usuario	E1 Usuario
E1 Usuario	E2 Juego
E1 Usuario	E5 Insignia
E1 Usuario	E6 Mensaje
E1 Usuario	E7 Comunidad
E1 Usuario	E9 Logro
E1 Usuario	E10 Foro
E1 Usuario	E11 Reseña
E2 Juego	E3 Género
E2 Juego	E4 Categoria
E2 Juego	E7 Comunidad
E2 Juego	E8 Desarrollador
E2 Juego	E9 Logro
E2 Juego	E10 Foro
E2 Juego	E11 Reseña
E2 Juego	E12 DLC
E6 Mensaje	E7 Comunidad
E6 Mensaje	E10 Foro

Fig. 4. Relationships Types Table

F. Step 6: First ERM Diagram

This first ERM diagram shows the main entities like User, Game, Community, and their initial relationships, the relationship types as seen in the previous step are taken into this diagram.

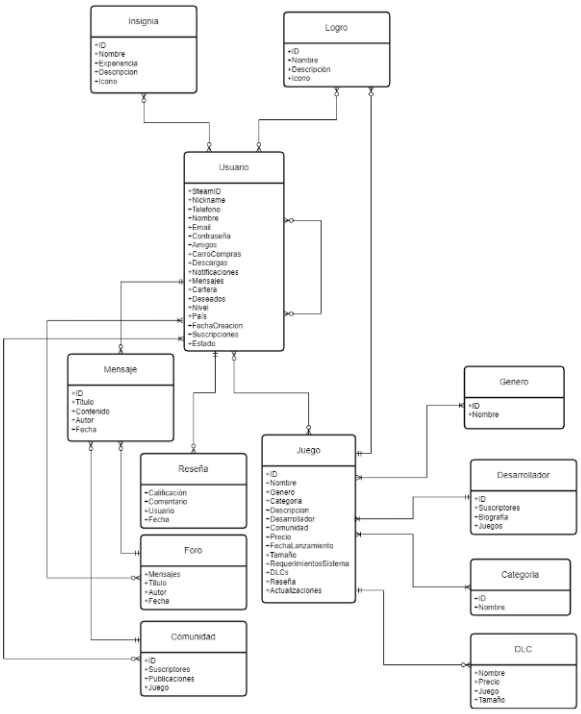

Fig. 5. First ERD of the model

G. Step 7-8: First Division of Many to Many Relationships and Second ERM Diagram

The red colored entities in the following ERD are the separations from many to many relationships, creating new entities each containing the foreign keys of the associated entities previously, and some attributes were slightly changed during the development of the second diagram.

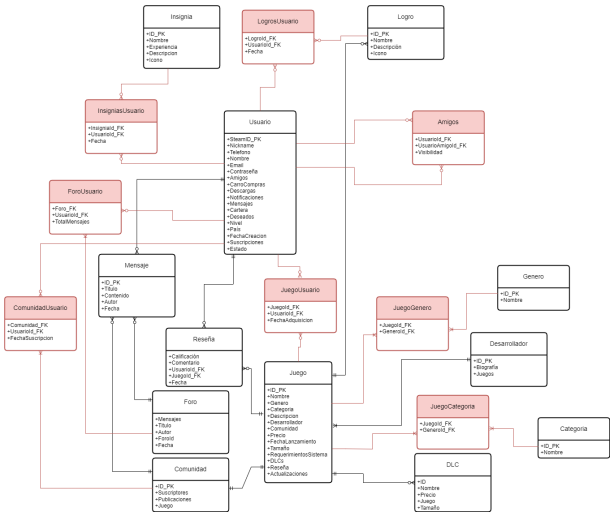

Fig. 6. Second ERD of the model

H. Step 9: Get ERM Data Structure

Some Foreign keys are named as the entity's name that is related to that attribute instead of an "int" attribute which means a number type data.

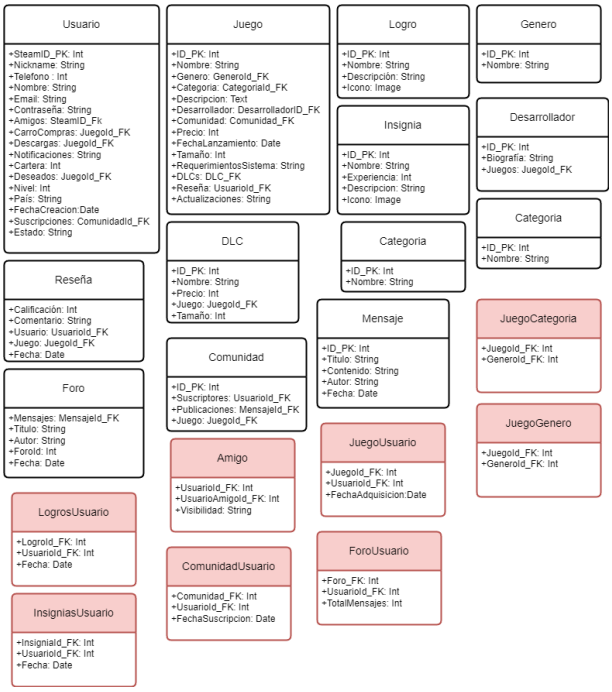

Fig. 7. Entity's Data Structure

I. Step 10: Define Constraints and Properties of Data

Constraints like a unique user ID and game ID ensure that each user and game is uniquely identifiable. Additionally, foreign keys in the UserGame junction table enforce relationships between users and games.

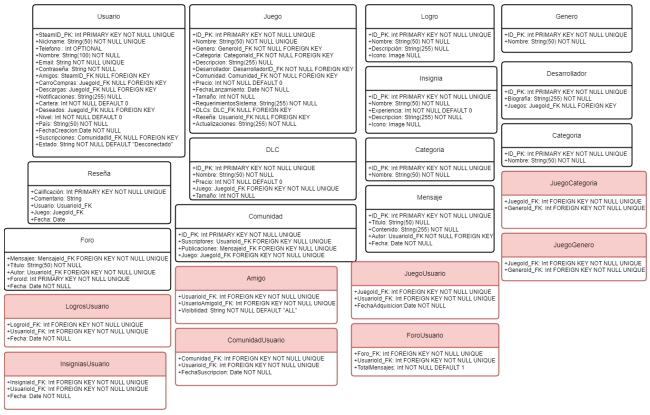

Fig. 8. Entity's Data Properties and Constraints

V. Conclusions

- It has been concluded that Steam relies heavily on games and users, meaning that the database model relationships usually aim to the users and the games.
- It was also concluded that the design of this model might be just a lite version of the actual database of steam due to it's complexity and size.

References

[1] Valve Corporation. Valve Corporation: About Us. https://www.valvesoftware.com/es/about, September 27th, 2021.

[2] SQLearning. Entity Relationship Model (ERM). https://sqlearning.com/sql-server-introduction/entity-relationship-model/, July 11th, 2022

[3] Sivakumar, B. (2023, August 7). How does Steam work — Steam Business Model. Feedough. https://www.feedough.com/how-does-steam-work-steam-business-model/