Basics of database systems

Project – Database design

Lappeenranta-Lahti University of Technology LUT
Software Engineering

Basics of database systems
Spring 2023

Räkköläinen Aino, 000391612

TABLE OF CONTENTS

1 DEFINITION

The purpose of the project ‘Game project database’ is to develop a database for a customer who manages a medium sized game development company with many simultaneous game development projects. The database will store information about the projects the company has finished or is currently working on and also some other companies the same customer might own. Each project will have a project team which has at least one team member and a maximum of 20. Each team member can be part of multiple projects, so it is important to store in which projects specific member is participating and in which role(s) so one person does not get too much work. Additionally, it is important to know which project relates to which game so which game which team is developing. Important information about the project is the name, starting and ending time (if it is completed), duration, unique id for project, the budget and id for identifying the team who is responsible for the project.

Additionally, it is important to find information about the games so when thinking of a new game project, other project teams would not start working on the same or too similar game which already exists. It is possible to make a sequel to the game, but it is not always the case. The games have players, so it is useful to store some basic information about players such as the ages, usernames, played hours and possible feedback for the game.

The following database queries must be implemented:

1) List all projects and the roles in the project in which specific team member is participating in the company.

2) List all games in specific genre that are currently being developed meaning the launching date is not known.

3) List all feedback already launched games have received and list the budget and the duration of those projects so the end-user can compare whether it was waste of resources or efficient use based on the received user feedback.

4) List all projects (names, starting and ending days, duration of the projects) and who are the developer(s) of each project team, and the list is finally ordered by project duration. That way the user easily knows who to contact in development issues.

5) List all games and the descriptions in alphabetical order based on the game name and create view of those. The end user get easily an overview of all games stored in the database.

6) List the ages of all players of the specific game and the age limit of the game.

2 MODELING

2.1 Concept model

In the Figure 1 is the ER model of the designed database. There are six entities in the model: Company, Project, Project Team, Team member, Game and Player. There are six relationships of which two are N:M relationships, three 1:M and one 1:1 relationship. Project team and team member have N:M relationship because team can have many members and the same team member can be a part of many project teams. The game can have many players and the same player can play many different games with the same username. Project and Game has 1:1 relationship because it is expected that one project focuses only on one game and one game is product of one project. The same project team can have multiple projects and multiple games and one company can have multiple projects. For simplification, it is expected that the team member can have one role in each project even though in real life the same person might have multiple roles in the same project. The game can have multiple genres and multiple comments so for that reason, those are represented as multivalued attributes but those might need to be changed in relational model.

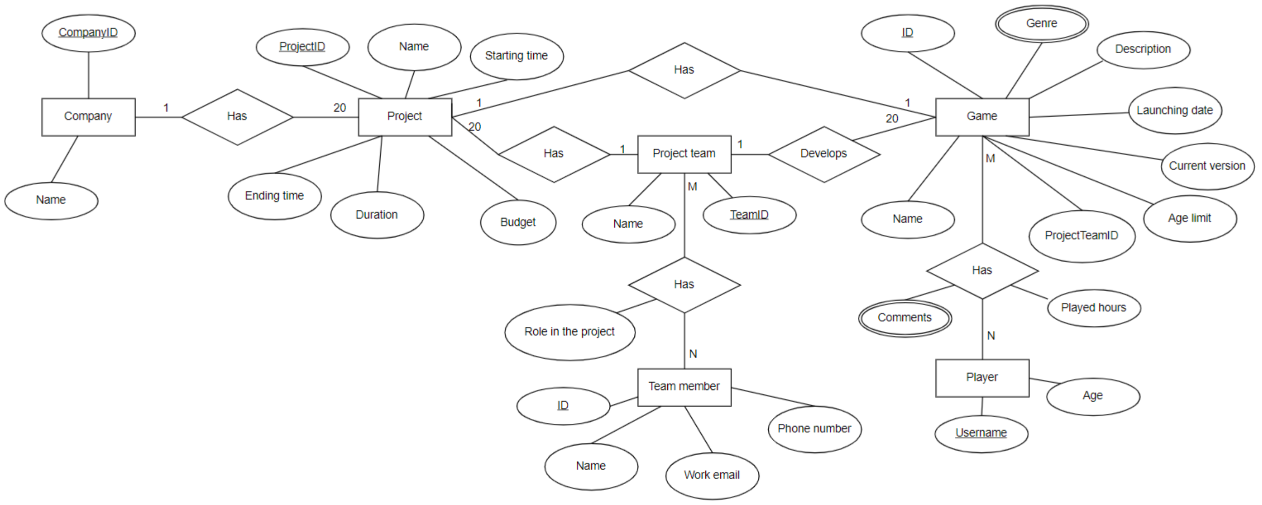

Figure 1: ER model

2.2 Relational model

Figure 2 shows the relational model that has been created based on the ER model and transformation rules given in the course material. In ER model there were two N:M relationships, so in relation model needed to create one intersection relation between Project team and Team member and other between entities Game and the Player. The multivalued attributes 'Genre' and 'Comments' were separated into additional entities which is supposed to done for multivalued attributes.

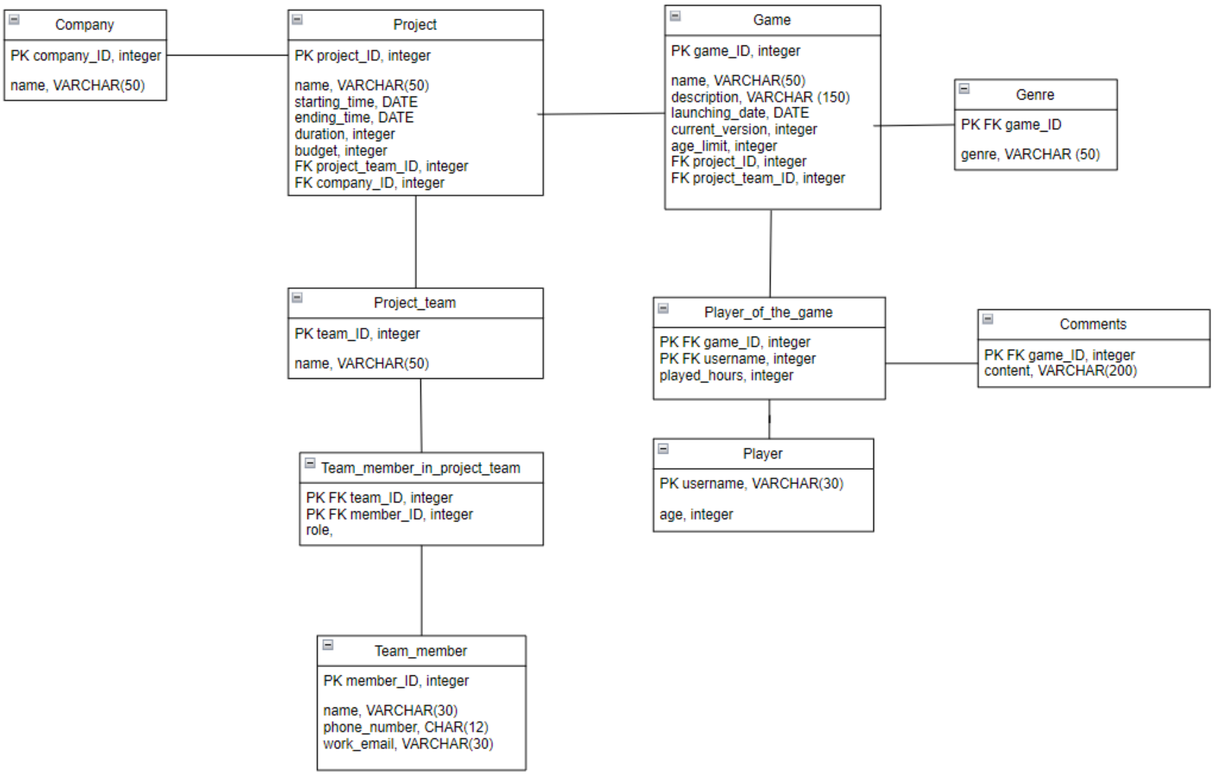

Figure 2: Relational model from the ER model

3 DATABASE IMPLEMENTATION

During the implementation, the following constraints are created for the relations:

- **Company:**
    - Name and company_ID cannot be null (NOT NULL)
- **Game:**
    - Game_ID, game_name, description, age_limit cannot be null (NOT NULL)
    - Foreign key reference to Project
    - Foreign key reference to Project team
- **Genre:**
    - Foreign key reference to game
    - The game has to have some genre (NOT NULL)
- **Player:**
    - Player has to have username and it has to be unique (NOT NULL, UNIQUE)
    - The age of the player must be at least 3 years. (CHECK)
- **Player of the game:**
    - Foreign key reference to Game
    - Foreign key reference to Player
    - It is expected that one player can have one username in each game so the combination of foreign key related to the game and the player's username must be inserted only once. Same combination cannot be inserted twice (UNIQUE)
    - If number of played hours is not given, it is 0 by default. (DEFAULT)
- **Project:**
    - Project_id, name, starting_time, duration and budget cannot be null (NOT NULL)
    - Foreign key reference to Company
    - Foreign key reference to Project team
    - Primary key project_ID is automatically increased (AUTOINCREMENT)
- **Project team**
    - Project team must have name and team id (NOT NULL)
    - Project team must have unique name (UNIQUE)
- **Team member**

    - Team member has to have name and id (NOT NULL)
    - Phone number and work email must be unique (UNIQUE)
- **Team member in the project team**
    - Team members role cannot be null (NOT NULL)
    - Foreign key reference to Project team
    - Foreign key reference to Team member
- **Comments**
    - Foreign key reference to Game

INDICES:

It is possible to create indices for foreign keys, for instance in query 3 I created indexes for foreign keys FK_project_ID on Game and FK_company_ID on the Project so it decreases a little the running time but not that much. For that reason, I didn't see it useful for creating other indices because most of my queries use ids for searching and those are primary keys, and it was not allowed to have indices based on the primary keys.

4 DISCUSSION

After implementing the database, I realized that it is not wise to have the duration of project inserted into the database as static value because then the database should be updated every day, or otherwise the data becomes outdated fast. T It should have been implemented in some different way, for instance only giving to duration to the ended projects or somehow make the database calculate it automatically.