# Software Requirements Specification

**for**

# Game Zone

**Version 1.0 approved**

**Prepared by:**
Thakkar Amit (201901038)
Makwana Jigar (201901428)
Prajapati Parth (201901429)
Gohil Arpit (201901471)

**DAIICT**

**26/10/2021**

## PROBLEM DESCRIPTION

The project will be an SQL database which will be created in a modern database management system. This will allow it to use the most modern features in DBMS. This will allow us to access the database through the provided GUI, pgadmin for postgres, or the CLI (command line interface), psql for postgres. So, the data can be conveniently modified or inserted into using these interfaces.

PostgreSQL, also known as Postgres, is a free and open-source relational database management system emphasizing extensibility and SQL compliance. PostgreSQL is a powerful, open source object-relational database system with over 30 years of active development that has earned it a strong reputation for reliability, feature robustness, and performance. PostgreSQL includes multiple features that are designed to help the developers in developing the applications, manage our data in the datasets, and managers can keep the data integrity, and create the Risk-tolerant environments.

The essential features of PostgreSQL are as follows:

- Compatible with Data Integrity :- It supports data integrity which includes the Primary Keys, UNIQUE, NOT NULL, Foreign Keys, Explicit Locks, Advisory Locks, Exclusion Constraints.
- Support multiple features of SQL :- PostgreSQL supports various features of SQL which include the MVCC (Multi-Version Concurrency Control), It supports multiple Indexing such as Multicolumn, Partial, B-tree, and expressions, Complex SQL queries, It supports transactions, Nested Transactions through Savepoints.
- Compatible with multiple data types :- Structured, Primitives, Geometry, Document (XML, JSON/JSONB)
- Highly extensible :- PostgreSQL is highly extensible because It supports procedural Languages such as Perl, PL/PGSQL, and Python, etc. and also has Stored procedures and functions.
- Secure :- It is safe because it provides a robust access control system and it supports Column and row-level security.
- Highly Reliable :- It is highly reliable and also provides disaster recovery as Active standbys, PITR (Point in time recovery), It supports WAL (Write-ahead Logging).

So using the PgAdmin which is one of the most famous and open-source management and development platforms for PostgreSQL , developers can develop and design many applications easily. Now, let’s talk about the features of the project.

The project will have the following features for the GameZone management system.

- **User** :-

  In our Gamezone database application, we found there are mainly three types of users, whose information we need to store in the database application: Player, Manager and Employee. All three entities have some common properties like name, age, contact no., Email address etc. For the purpose of authentication we will ask users to provide username and password. We will store all this information in the database. In the ER diagram we have used specialization to represent the User class and its three subclasses. Below we have given detailed information about all three types of users.

- **Player :-**

  We are making the game zone database management system to provide the best environment for players to play games. So, players are the first requirement for any game zone, without players there is no meaning of Game zone. Players are the main source of earning money for managers, employees and owners of Game Zone. To experience the players, the features of this application like leaderboard, multiplayer games, Payment records to provide more consistency of transactions, rewards etc we need to store the personal details of players.

  The Database will store the information of players like player id, player name, player age, height, weight, mobile number, email id, team id. Here we use player id as primary key in “player” relation. In the game zone, there are games which have some criteria like age, height and weight so the database stores the age, height, and weight of the player. In future if there are any updates or discounts or events in the game zone, they can contact their customer via their Email Id and mobile number. For the multiplayer games database stores the Team Id which is also unique for each team, so two or more Teams can’t take the same Team Id. Team ID will be referred from Team entity.

  The application will take care that the personal details of players do not leak by providing some restrictions like players can see their own profiles and also modify some allowed details. Players can see other player’s details like name, score, rank but Players are not allowed to see other player’s personal details like password, address, email id, contact number etc. Manager can also see the details of the player apart from the password field. Here the password can be reset by only the player.

- **Manager :-**

  Every organization or system has some management for their growth. Management is needed in order to coordinate the activities of a business and make sure all employees are working together toward the accomplishment of the organization's goals. For that we need managers to manage these things. In our game zone , to manage the whole system and to monitor employees work the system must need a manager.

  The manager database will have the information like ID, name, mobile no., email address, details about profit and loss of the game zone etc. ID is used to uniquely identify the manager. Profits and Losses have information about a game zone's sales and expenses over a specified period of time. Other information will be taken to contact the manager like using email ID, mobile no., address etc.

  Manager has to manage everything and for that he/she needs the whole control of the game zone. Means he/she must have the privilege of modifying the data, reading the data etc. Manager can read all information related to the game zone like employees data, players data, games data etc. Managers can modify the employees salary, delete any record of game history, remove the account of any player in case of cheating or damaging the property of the game zone and there are many other information managers can modify.

- **Employee :-**

  Gamezone can’t be managed by only the manager. Manager needs some workers who can work under his guidance. Employees will have work like registration of players, supervising, managing payment details etc. So to manage and everything will work smoothly under the guidance of the manager the employees are needed.

  The Employee database will specialize in two subclasses : Account manager and worker. Employee is also a specialized subclass of User class which will have information like ID, name, age, contact no., email address, role, salary etc. ID is used to uniquely identify each employee. Salary represents the amount paid by the owner per month to the employee. Role has the information of the role of each employee. Other information is used to contact the employees like using mobile no, email ID etc.

  Employees can read and update his/her personal information except its salary (it can only be modified by the manager). They don’t have any privilege to

modify or read the details of the manager. They can read and insert the players' details. Some of the employees who are granted by the manager can modify the details of the game zone like price of game, rewards, etc.

- **Account Manager :-**

  Account Manager is a subclass of an employee superclass. Account manager will have information of Game Zone account no. and balance. By using this information he/she can manage the transactions and rewards. He/She will credit the reward amount the player has got while playing any game in his(player's) account using account details. This transaction will be recorded in the transaction database.

  Account Manager has privilege of modification in account balance and can read the transaction details and account details of the particular player.

- **Worker :-**

  Worker is also a subclass of an employee superclass. It will have information about a worker's speciality related with the games like game designer, game artist, game developer, etc. Workers will be dealing with game related queries like removing games, adding new games, fixing bugs, game lag, security issues etc.

  Worker has the privilege of modification in the game database to solve the game related queries.

- **Game** :-

  Games are also the most important entity in our project. In a Gamezone there are various types of games available. Players will see the details of the game and after that they make a decision about which game they should play. So, to attract the players and manage all the games we need to store details of the games in our database.

  The database application will store the details of games like name of game, type, mode, price of game, maximum amount of prize that the player can win by playing games etc. To determine all game details uniquely, the database application will give a unique Game ID to all Games. The name, type and mode will provide an overview of the game to players. Some games have age restrictions like this game is only available for players who will satisfy the age criteria decided by the owner or manager. The default value of required age will

be 3 years. Game Zone can have both types of games, online games and offline games. Mode will specify whether the game is online or offline. The Game Zone can have different types of solo,duo or multiplayers games like racing games, adventure games, action games etc. Type of game will provide this information to players. In gamezone, generally all the games are premium, you need to pay money if you want to play a game and the player should know the price of the game. Players can win some rewards according to the rank in the leaderboard, the application will store the maximum amount of prize money that the player can win by playing the game. This will also encourage players to play games. Location attribute will show the game location if the game is online then it will show room id and if it's offline then it will show the location address.

The details of the Game will be inserted or modified or deleted by the manager or employee (granted by the manager) . Players can retrieve the details of any game and enjoy the various types of game and also can win some exciting rewards. Players are not allowed to modify or delete any game details.

- **Account :-**

  When Player visits GameZone, he will be asked to open his account in the GameZone system. For that he has to choose a membership package and pay some money and upload some required documents and details as decided by the owner. After this process, one debit card will be provided to the player by gamezone. Using this debit card, players can make payments and get prize money. So all details related to money will be stored in the account class.

  Account relation contains the field, player_id, account_id, debit_number, current_balance, date_of_joining, username and password. Field player_id is referenced from the "player" relation, account_id is autogenerated which is unique and assigned by application to each player, debit_number will be provided by game zone, current_balance shows the amount that in the account which will be updated when any transaction happen between player and game zone, date_of_joining is the date that first time player visited game zone, username is unique for all players which is selected by player and using this player can login to the website.

  Players are allowed to see their Account details, but they aren't able to see other player details. Players are not allowed to modify or delete any records. Manager can also see the account details of all players but can't modify it. Manager can delete the account of any player if the player is found to violate any term and condition, or his validity of the package is finished and the player has not renewed membership yet.

- **Transaction :-**

  As we have discussed above, we will give the debit card to all players when they open their account in Game Zone and we are storing the current balance of the player in the Account entity. Players will pay the price for playing the game and also may win or lose some rewards in game, and it will affect the current balance of the player. So, we need to store the all-transaction records in the database application as a proof of transaction . For that we make the class of Transaction, which has the information of transactions like Player ID, Game ID, Amount paid/win/loss, Date and Time of payment, method of transaction etc. To uniquely determine all records, we need to add one more attribute Payment No. for each player. The Combination of Player ID and Payment no. will become the primary key of Transaction. We will take the reference of Player ID from Player Entity and Game ID from Game Entity. The information of the amount transferred helps the application to update the balance of the player efficiently.

  If we talk about multiplayer games, then all the members of the team need to pay for playing the game. And the rewards and losses will be equally divided to all members of the team. So, the case of multiplayer is also handled as an individual player's payment.

  Players are allowed to retrieve transactions from their own account. But they are not allowed to modify or delete any records. Players are not allowed to see other player's transaction details. Manager and employees (Granted by Manager) are allowed to retrieve the information of transactions. But they can't modify or delete it. The updating in the Account will be done by application as any record will be inserted in the transaction.

- **Team :-**

  In GameZone, there are some multiplayer games and to play multiplayer games, players are required to make a team and register a team in gamezone.

  The database application will store the information about the team like team name, game id, team rank and the number of games won. All the members of the team should be members of the game zone. To uniquely determine all teams, team_id will be provided by the application, team_name can be decided by the team members, team rank shows their position on the leaderboard of the game, and no. of game wins shows how many game teams won.

Here players are allowed to view their own team's profile as well as the team member's profile. Some fields like rank, no. of games won will be automatically updated in the database.

- **Game history :-**

  Game History is one of the most important parts of our Game zone management system. The Game History database stores all the past information of players till now. The information which it stores is all the games details the individual player has played, game performance details like scores, how many levels has completed ?, achievements, no. of won and loss games, how much reward the player has got in different games ?, date and time of when the player has played that game ? etc.

  Now why do we need all this information ? This information helps the player to see their past gaming history about individual games. It has the game performance, scores and rank which helps the player to see their performance so that he/she can improve that. The owner/manager of the game zone can identify which player has the most amazing performance of all time?Which games are played most till now?Which player gets the highest rewards including all games he/she has played till now? etc. From all this information, owners and game developers get to know which games are underrated or have the least popularity among players so they can improve or can remove those games. So as a business perspective for the owner, game history is a very important entity of the game zone.

  For the Game history database, we need Record no., Player ID, Game ID, Player's Score, Level, Rank, the number of won and lost games, win rate, rewards, date and time etc. Record no. will uniquely identify all the records. Player ID and Game ID to get the details of player and the game details he/she has played and For the game statistics and performance, player's score , rank, level all these details are needed. Date and time fields are needed to store when the player has played all those games. This information will be taken from Player, Game and Leaderboard entities.

  Every player can only see their own Game history details. He/she has no privilege to modify the details. Employees and managers can see all players' gaming history details but employees can't make any change but the owner has the privilege to remove or delete the record in case of banning the account or deleting the account.

- **Lobby :-**

  The database application will store the details of players who are currently playing in Game Zone. Here, the application should have the information about which player is playing now? Which game is he playing? At which location is he playing? What is the current status of a player means the player is playing or has finished? This is all the information we need to maintain the players who are currently waiting to play a game.

  We are going to store the information like Player ID, Game ID, Location, number of players playing game (corresponding to game ID), date and time, and current status of player in Lobby. We need Player ID and Game ID to uniquely determine any tuple; we will take this information from the Player and Game entity. Location, date and time will give extra information to manage the list. In current status there are two possible status: Playing and Finished.

  Here, in Game details we have given the capacity of the game which denotes the maximum number of players that can play at a time. When a player will login into the game, we will compare the number of currently playing players with the capacity of the corresponding game, and if capacity has not been reached, we will add him into the currently playing players list and set its status playing, otherwise we will add the player to the waiting list. When a player logs out from the game, we will set the status of the player finished. After login or logout in game, we will update the number of currently playing players in the corresponding game. When any player logs out from the game, we will check in the lobby that any player is waiting for a corresponding game, if yes then we will add into this list and set his status playing and update the number of currently playing players. For multiplayer games, teams will be handled individually as above.

  Any insertion, modification and deletion will be done by the application automatically according to the information stored in This list, waiting list and game history. Players are not allowed to view or modify this list. Manager can give a role to any employee to manage this list. Manager can retrieve the information about currently playing players. But he can’t modify it.

## All Extracted Nouns & Verbs from Problem Description

| Nouns | Verbs |
|---|---|
| Above | Access |
| Account | Add |
| Account id | affect |
| Achievements | allow |
| Action | Alter |
| Address | ask |
| Administrator | Auto generate |
| Adventure | Automate |
| Age | Ban |
| Application | become |
| attribute | Change |
| Awareness | Check |
| balance | choose |
| Capacity | compare |
| class | Compete |
| Company | Connect |

| | |
|---|---|
| Competition | contact |
| Consistency | Contain |
| Constraints | Create |
| Contact number | Decide |
| criteria | Delete |
| Current balance | denote |
| customer | Design |
| Database | determine |
| Date | Develop |
| Date of joining | Differ |
| Debit Card | discuss |
| Debit card no. | Display |
| Desire | divide |
| detail | Editing |
| Developer | Entertain |
| discount | Find |
| documents | finish |

| | |
|---|---|
| Employee | Follow |
| Engagement | Foster |
| Engineers | get |
| Entity | Give |
| events | Grant |
| feature | Handle |
| field | Help |
| Foreign key | Identify |
| Game Interfaces | Improve |
| Game mode | Insert |
| Game Name | leave |
| Game session time | Log out |
| Game Type | login |
| Gameplay | Loss |
| GameZone | Maintain |
| Geometry | Make |
| Goal | ManageImprove |

| | |
|---|---|
| Height | modifyUnderrate |
| History | Pay |
| information | Play |
| Integrity | process |
| Interface | Provide |
| Keys | reach |
| Leaderboard | Read |
| Level | referenced |
| list | remove |
| Lobby | reset |
| location | Restrict |
| Locks | retrieve |
| Manager | Revoke |
| Member | Scroll |
| Modern | Secure |
| Modes | see |
| money | Seek |

| | |
|---|---|
| Motivator | Select |
| multiplayer | Serves |
| Notoriety | set |
| Owner | Show |
| password | Specify |
| Past | Spend |
| Payment | Store |
| Payment No. | take |
| Peer | Task |
| People | Travel |
| Performance | Type |
| permission | Underrate |
| PgAdmin | Update |
| Player | Upload |
| Player Age | use |
| Player ID | view |
| Player Name | visit |

| | |
|---|---|
| Player team | Wait |
| PostgreSQL | Win |
| Primary key | Work |
| Primitive | |
| Privilege | |
| Prize | |
| profile | |
| Queue | |
| Racing | |
| Rank | |
| Rating | |
| Record | |
| Recovery | |
| reference | |
| relation | |
| Rewards | |
| role | |

| | |
|---|---|
| Rules | |
| Salary | |
| Score | |
| Security | |
| Stand | |
| Statistics | |
| Status | |
| Storage | |
| Survival | |
| System | |
| Table | |
| Team | |
| Team ID | |
| Team Name | |
| Team rank | |
| Time | |
| Tournament | |

| | |
|---|---|
| Transaction | |
| tuple | |
| username | |
| vacancy | |
| Waiting list | |
| Website | |
| Weight | |
| Win rate | |
| Membership | |
| Type | |
| Validity | |
| Membership No. | |

## All Accepted Nouns & Verbs list

| Candidate Entity Set | Candidate attribute set | Candidate relationship set |
|---|---|---|
| Game | Game id | Play |
| Player | Game name | History |
| Game History | Type | Team history |
| Team | Mode | Member of |
| User | Price | Open |
| Manager | Reward | Manages |
| Employee | Age | Playing in |
| Account manager | Height | Transaction |
| Transaction | Weight | Deals |
| Account | Record number | |
| Lobby | Date and time | |
| Worker | Rank | |
| | Score | |
| | Level | |
| | Wins | |
| | Losses | |
| | Win rate | |
| | Team id | |
| | Team name | |
| | Number of players | |
| | User id | |
| | Name | |
| | First name | |
| | Second name | |

| | | |
|---|---|---|
| | Last name | |
| | Contact number | |
| | Email Address | |
| | Username | |
| | Password | |
| | Salary | |
| | Role | |
| | Speciality | |
| | Account number | |
| | Debit card number | |
| | Date of joining | |
| | Balance | |
| | Transaction id | |
| | Amount | |
| | Type | |
| | Method | |
| | Status | |
| | Location | |
| | Login time | |

## All Rejected Nouns list with reason of rejection

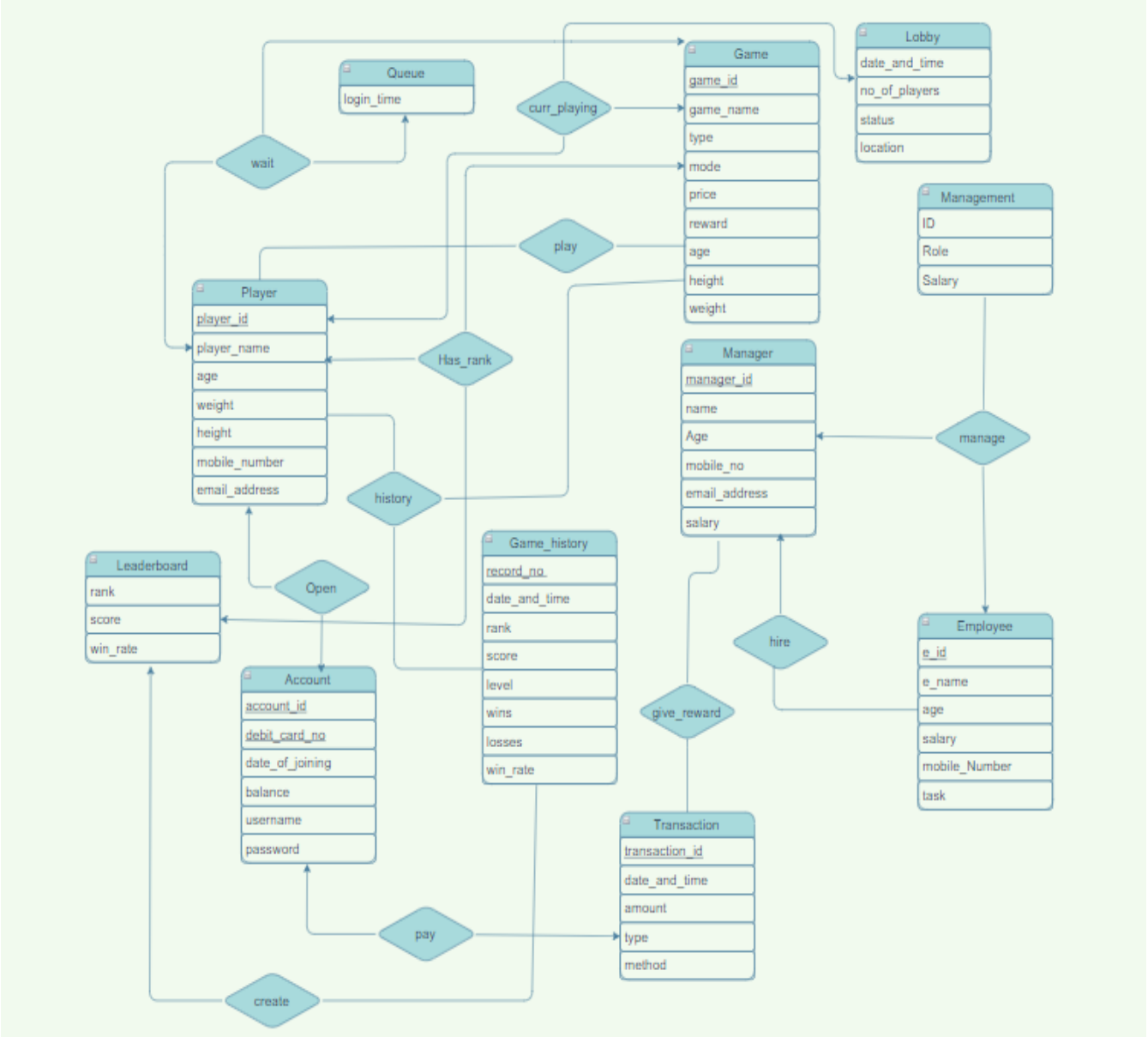
| NOUN | REJECT REASON | VERB | REJECT REASON |
|---|---|---|---|
| Above | General | Access | Vague |
| Achievements | Irrelevant | Add | Duplicates |
| Action | Attributes | Affect | General |
| Administrator | Irrelevant | Allow | General |
| Adventure | Attributes | Ask | Irrelevant |
| Application | General | Automate | Associations |
| Attribute | Duplicates | Ban | Attributes |
| Awareness | General | Become | Vague |
| Balance | Duplicates | Change | Associations |
| Capacity | Attributes | Check | General |
| Class | Duplicates | Choose | Vague |
| Company | General | Compare | Duplicates |
| Competition | Irrelevant | Compete | Attributes |
| Consistency | Vague | Connect | Irrelevant |
| Constraints | Associations | Contact | Attributes |
| Criteria | Duplicates | Contain | Irrelevant |
| Contact Number | Duplicates | Decide | General |
| Customer | Duplicates | Denote | Vague |
| Database | General | Design | General |
| Debit Card | Duplicates | Determine | Irrelevant |
| Desire | Irrelevant | Develop | General |
| Detail | General | Differ | Vague |
| Developer | Vague | Discuss | General |

| Discount | Irrelevant | Display | Duplicates |
|---|---|---|---|
| Documents | Vague | Divide | Duplicates |
| Engagements | Associations | Entertain | Irrelevant |
| Engineers | Irrelevant | Find | Vague |
| Entity | Duplicates | Finish | General |
| Events | Vague | Follow | Attributes |
| Feature | Irrelevant | Foster | Vague |
| Field | General | Get | Associations |
| Foreign Key | Associations | Grant | Associations |
| Game Interfaces | Duplicates | Handle | General |
| Game play | Irrelevant | Help | General |
| Game Zone | General | Identify | Associations |
| Geometry | Irrelevant | Improve | Associations |
| Goal | Attributes | Leave | General |
| History | General | Log out | General |
| Information | General | Log in | General |
| Integrity | Associations | Make | Duplicates |
| Interface | Irrelevant | ManageImprove | Associations |
| Keys | Vague | modifyUnderrate | Duplicates |
| Level | Attributes | Process | Vague |
| List | Irrelevant | Provide | Vague |
| Locks | Vague | Reach | Attributes |
| Modern | Irrelevant | Read | General |
| Money | Duplicates | Referenced | Associations |
| Motivator | Irrelevant | Remove | Duplicates |
| Multiplayer | Attributes | reset | Irrelevant |

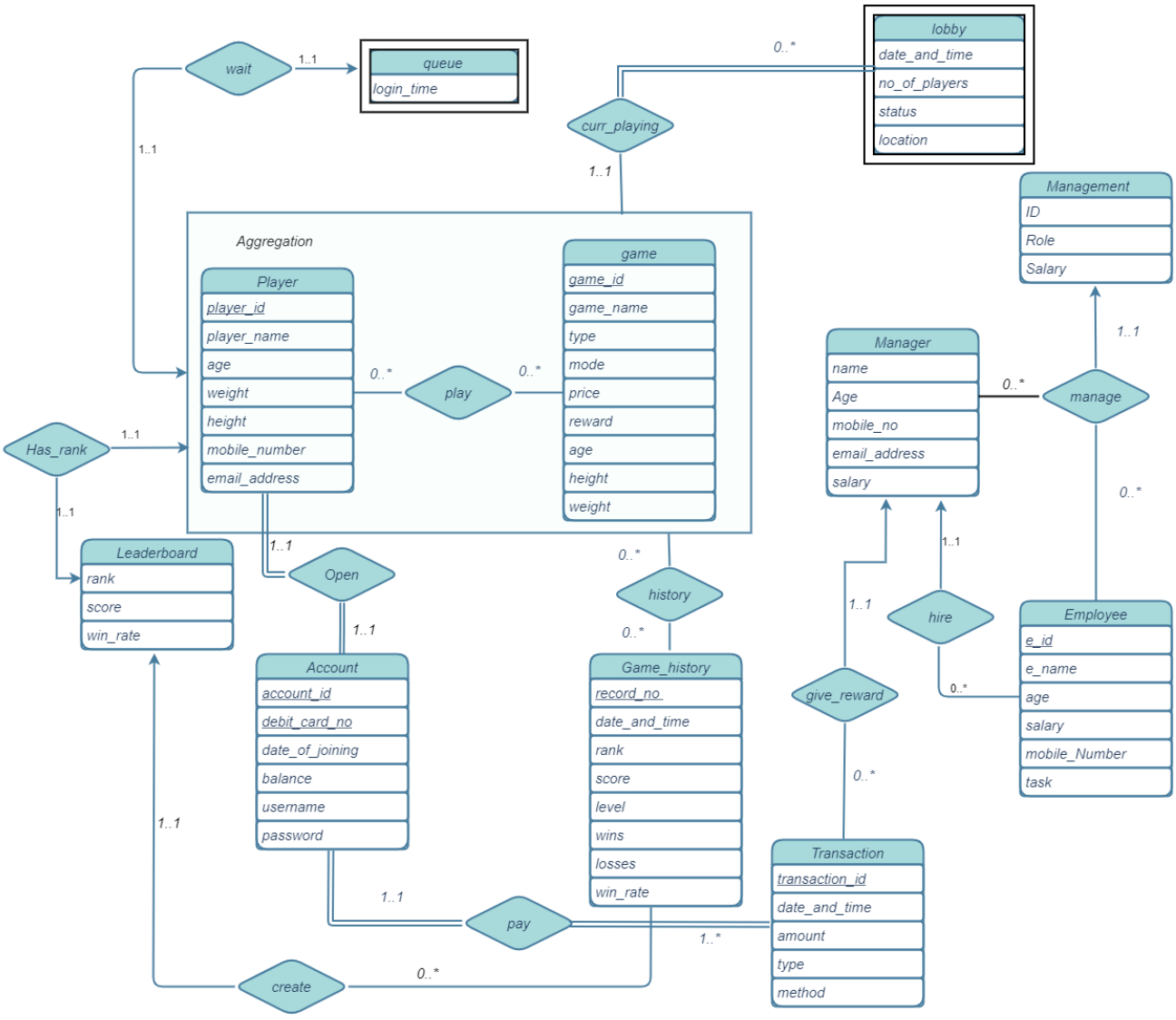
| Notoriety | General | Restrict | Attributes |
|---|---|---|---|
| Past | General | retrieve | Vague |
| Payment | Duplicates | Revoke | Vague |
| Peer | Duplicates | Scroll | Irrelevant |
| People | General | Secure | General |
| Performance | Duplicates | see | Vague |
| Permission | Associations | Seek | Attributes |
| pgAdmin | General | Serves | Associations |
| PostgreSQL | General | Show | Duplicates |
| Primary Key | Associations | Specify | Irrelevant |
| Primitive | Associations | Spend | Duplicates |
| Privilege | General | Store | General |
| Price | Attributes | Take | Duplicates |
| Profile | Irrelevant | Task | Duplicates |
| Queue | Irrelevant | Travel | Irrelevant |
| Racing | Attributes | Type | General |
| Rating | Duplicate | Underrate | Attributes |
| Record | General | Use | General |
| Recovery | General | View | General |
| Reference | Associations | Visit | Vague |
| Relation | Associations | Work | Duplicates |
| Reward | General | | |
| Role | Irrelevant | | |
| Rules | General | | |
| Score | Attributes | | |
| Security | General | | |

| | | | |
|---|---|---|---|
| Stand | Associations | | |
| Statistics | General | | |
| Status | Attributes | | |
| Storage | General | | |
| Survival | Irrelevant | | |
| System | General | | |
| Table | Duplicates | | |
| Tournaments | General | | |
| Tuple | General | | |
| Vacancy | Attributes | | |
| Website | General | | |

# E-R Diagram V1



# E-R Diagram V2



# E-R Diagram final