League of Legends: Online Matchmaking Database System

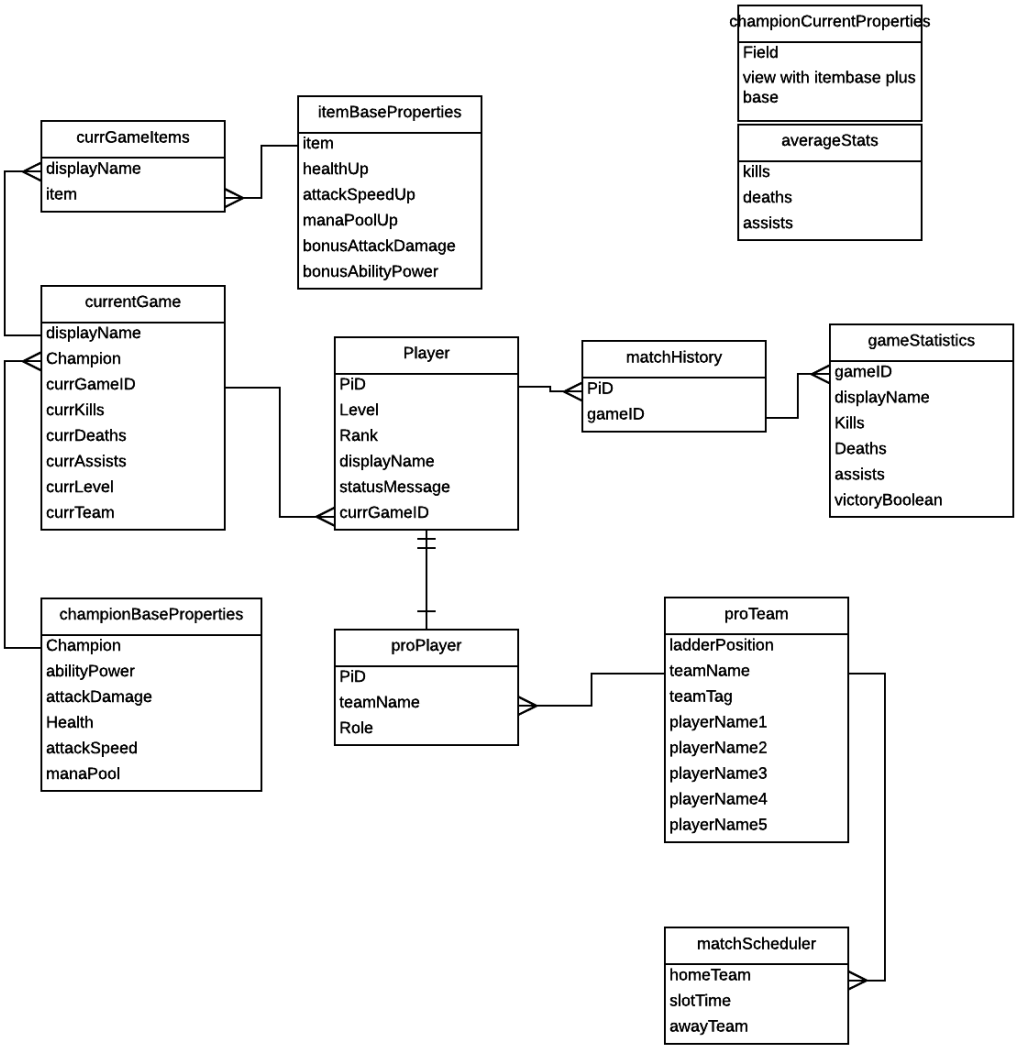
Designed By: John Kryspin

Table of Contents

Executive Summary

This entire document shows how League of Legends matchmaking in their online system works and how the data is stored in a way that makes sense and is easy to look through. The database shows players interactions with each other, their items, and their champions and has scheduling for pro teams. There are game statistics that are saved in the match history table and there are two views championCurrentProperties and averageStats

Entity Relationship Diagram



Tables

itemBaseProperties table

This table lists every item in the game and the properties attached to it. Each champion can buy items to use in the current game and it affects their champion's stats.

create table itemBaseProperties(healthUp int,

attackSpeedUp decimal,

manaPoolUp int,

bonusAttackDamage int,

bonusAbilityPower int,

item text primary key);

Functional Dependencies: item-> healthUp,attackSpeedUp,manaPoolUp,bonusAttackDamage,bonusAbilityPower

Sample Data:

Data Output | Explain | Messages | History

	healthup integer	attackspeedup numeric	manapoolup integer	bonusattackdamage integer	bonusabilitypower integer	item text
1	0	0	0	50	0	B.F. Sword
2	0	0	0	25	0	Pickaxe
3	0	0	0	10	0	Longsword
4	0	0	200	0	0	Sapphire Crystal
5	150	0	0	0	0	Ruby Crystal
6	0	1.3	0	0	0	Recurve Bow
7	0	0	0	0	80	Needlessly Large Rod
8	0	0	0	0	40	Blasting Wand
9	0	0	0	0	20	Amplifiying Tome

currGameItems table

This table uses each person's display name and item as a composite key. This is necessary because one displayName can have multiple items which is why having just displayName as the primary key would not work.

```
create table currGameItems(displayName text,
                item text,
                primary key (displayName, item)
                );
```

Functional Dependencies: displayName,item->

Sample Data:

	displayname text	item text
1	Jibbey	B.F. Sword
2	Jibbey	Pickaxe
3	SoberSoda	Ruby Crystal
4	SnatchyChicken	Pickaxe
5	Rushpack	Needlessly Large Rod
6	C Note	Recurve Bow
7	Bjergson	Longsword
8	Bjergson	Pickaxe
9	Dyrus	Pickaxe
10	Santorin	Amplifiying Tome
11	Wildturtle	B.F. Sword
12	Lustboy	Blasting Wand

OK.

currentGame table

This tracks the current data in an ongoing match. This table would be updated every time a champion buys an item gets a kill or dies or gets an assist or gains a level. The example is just a snapshot in time for one game that is ongoing.

create table currentGame(displayName text,

Champion text,

currGameID int,

currKills int,

currDeaths int,

currAssists int,

currLevel int,

currTeam int,

primary key(currGameID, displayName));

Functional Dependencies: currGameID,displayName ->Champion, currKills, currDeaths, currAssists, currLevel, currTeam

Sample Data:

	displayname text	champion text	currgameid integer	currkills integer	currdeaths integer	currassists integer	currlevel integer	currteam integer
1	Jibbey	Vi	0	4	0	2	11	1
2	SoberSoda	Shyvana	0	2	1	2	10	1
3	SnatchyChicken	Jinx	0	1	1	2	11	1
4	Rushpack	Zed	0	2	1	2	10	1
5	C Note	Aatrox	0	4	1	2	12	1
6	Bjergson	Akali	0	1	6	2	10	2
7	Dyrus	Singed	0	1	1	2	10	2
8	Santorin	Gragas	0	1	1	2	10	2
9	Wildturtle	Tristana	0	1	1	2	10	2
10	Lustboy	Lulu	0	0	4	2	10	2

OK.

championBaseProperties table

This table shows the base stats of each champion in the entire game. This table would be static and not change but rather a view would be created to show a champions current statistics with items and such.

create table championBaseProperties(Champion text primary key,

abilityPower int,

attackDamage int,

Health int,

attackSpeed decimal,

manaPool int);

Functional Dependencies: Champion -> abilityPower, attackDamage, Health, attackSpeed, manaPool

Sample Data:

Data Output | Explain | Messages | History

	champion text	abilitypower integer	attackdamage integer	health integer	attackspeed numeric	manapool integer
1	Vi	0	50	700	1.2	700
2	Shyvana	0	70	700	1.1	0
3	Jinx	0	50	500	1.4	500
4	Zed	0	90	500	1.0	0
5	Aatrox	10	50	700	1.4	0
6	Akali	10	20	700	1.2	700
7	Singed	0	50	700	1.2	770
8	Gragas	0	0	800	1.2	700
9	Tristana	0	50	700	1.5	700
10	Lulu	0	10	400	1.3	700
11	Corki	0	50	700	1.2	700
12	Rengar	0	50	700	1.2	700

OK.

Player table

This table tracks basic statistics about the player and shows if they are currently in a game.

create table Player(PiD varchar(10) primary key,

Level int,

Rank text,

displayName text,

statusMessage text,

currGameID int);

Functional Dependencies: PiD-> Level, Rank, displayName, statusMessage, currGameID

Sample Data:

	pid character varying(10)	level integer	rank text	displayname text	statusmessage text	currgameid integer
1	1	30	Gold 2	Jibbey	Stuck in gold :(	0
2	2	30	Silver 4	SoberSoda	Stuck in silver!	0
3	3	30	Bronze 1	SnatchyChicken	Stuck in bronze my life sucks	0
4	4	30	Diamond 5	Rushpack	Just got diamond!	0
5	5	30	Gold 4	C Note	Looking for RBGs	0
6	6	30	Challenger	Bjergson	Got a new cat	0
7	7	30	Challenger	Dyrus	I hate teamates	0
8	8	30	Challenger	Santorin	My chin is huge	0
9	9	30	Challenger	Wildturtle	I go deep everyday	0
10	10	30	Challenger	Lustboy	Support is so easy	0
11	11	30	Silver 3	KillerDoomSky	I hate silver	
12	12	30	Gold 3	SoPretty	online	
13	13	30	Gold 4	SaltySailor	online	
14	14	30	Challenger	Balls	Got a new cat	
15	15	30	Challenger	Meteos	I hate teamates	
16	16	30	Challenger	Hai	My chin is huge	
17	17	30	Challenger	Sneaky	I go deep everyday	
18	18	30	Challenger	Lemonation	Support is so easy	

OK.

proPlayer table

This table inherits data from player because every proPlayer is a Player but not vice versa. It also shows which role they play in the team.

create table proPlayer(PiD varchar(10) primary key,

teamName text,

Role text);

Functional Dependencies: PiD->teamName,Role

Sample Data:

	pid character varying(10)	teamname text	role text
1	6	Team Solo Mid	Mid
2	7	Team Solo Mid	Top
3	8	Team Solo Mid	Jungle
4	9	Team Solo Mid	ADC
5	10	Team Solo Mid	Support
6	14	Cloud 9	Top
7	15	Cloud 9	Jungle
8	16	Cloud 9	Mid
9	17	Cloud 9	ADC
10	18	Cloud 9	Support

proTeam table

This table lists each professional team and their roster in the League Championship Series. Currently there are only two teams but adding more would work fine too.

create table proTeam(teamName text primary key,

ladderPosition int,

teamTag varchar(4),

playerName1 text,

playerName2 text,

playerName3 text,

playerName4 text,

playerName5 text);

Functional Dependencies: teamName-> ladderPosition, teamTag, playerName1, playerName2, playerName3, playerName4, playerName5

Sample Data:

Data Output | Explain | Messages | History

	teamname text	ladderposition integer	teamtag character varying(4)	playername1 text	playername2 text	playername3 text	playername4 text	playername5 text
1	Team Solo Mid	1	TSM	Dyrus	Santorin	Bjergson	Wildturtle	Lustboy
2	Cloud 9	2	C9	Balls	Meteos	Hai	Sneaky	Lemonation

matchScheduler table

This table sets up matches between teams and users integer slots to determine the time they play. The sample data reflects one week.

create table matchScheduler(homeTeam text primary key,

slotTime int,

awayTeam text);

Functional Dependencies homeTeam->slotTime, awayTeam

Sample Data:

Data Output | Explain | Messages | History

	hometeam text	slottime integer	awayteam text
1	Team Solo Mid	1	Cloud 9

matchHistory table

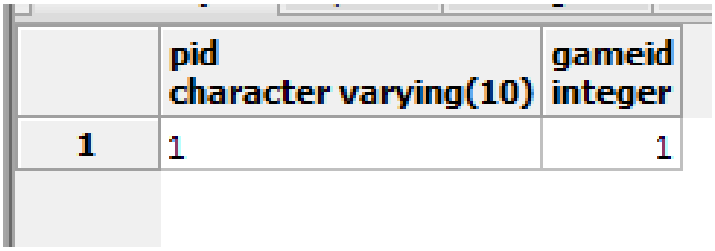
This table allows you to see how many games a player has played and links to another table which shows the stats for each game.

create table matchHistory(PiD varchar(10) primary key,

gameID int);

Functional Dependencies: PiD->gameID

Sample data:



	pid character varying(10)	gameid integer
1	1	1

gameStatistics

This table shows the details for each match a person has played and if they won or lost the match (boolean)

create table gameStatistics(gameId int primary key,

displayName text,

Kills int,

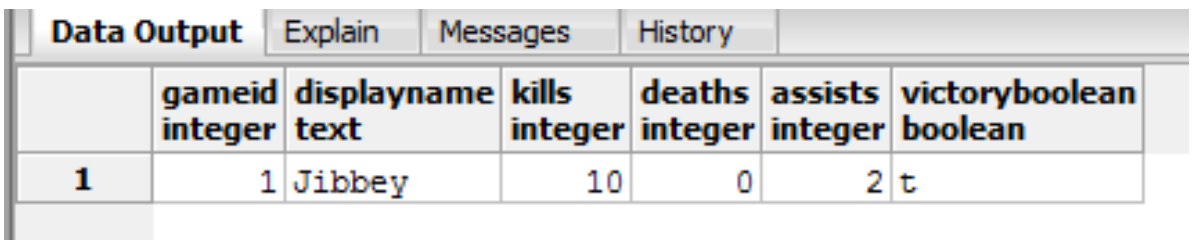
Deaths int,

Assists int,

victoryBoolean boolean);

Functional Dependencies: gameId-> diaplyName,Kills,Deaths,Assists,victoryBoolean

Sample Data:



Data Output | Explain | Messages | History

	gameid integer	displayname text	kills integer	deaths integer	assists integer	victoryboolean boolean
1	1	Jibbey	10	0	2	t

Views

PlayerAndChampion

Shows the players and the champion they are playing

```
create view PlayerAndChampion as
        select displayName, currentGame.Champion
        from currentGame, championBaseProperties
        where currentGame.Champion = championBaseProperties.Champion
```

	displayname text	champion text
1	Jibbey	Vi
2	SoberSoda	Shyvana
3	SnatchyChicken	Jinx
4	Rushpack	Zed
5	C Note	Aatrox
6	Bjergson	Akali
7	Dyrus	Singed
8	Santorin	Gragas
9	Wildturtle	Tristana
10	Lustboy	Lulu

PlayersInGame

Shows only players who are currently in a game

```
create view PlayersInGame as
        select *
        from Player
        where currGameID is not null
```

Data Output | Explain | Messages | History

	pid character varying(10)	level integer	rank text	displayname text	statusmessage text	currgameid integer
1	1	30	Gold 2	Jibbey	Stuck in gold :(	0
2	2	30	Silver 4	SoberSoda	Stuck in silver!	0
3	3	30	Bronze 1	SnatchyChicken	Stuck in bronze my life sucks	0
4	4	30	Diamond 5	Rushpack	Just got diamond!	0
5	5	30	Gold 4	C Note	Looking for RBGs	0
6	6	30	Challenger	Bjergson	Got a new cat	0
7	7	30	Challenger	Dyrus	I hate teamates	0
8	8	30	Challenger	Santorin	My chin is huge	0
9	9	30	Challenger	Wildturtle	I go deep everyday	0
10	10	30	Challenger	Lustboy	Support is so easy	0

Stored Procedures

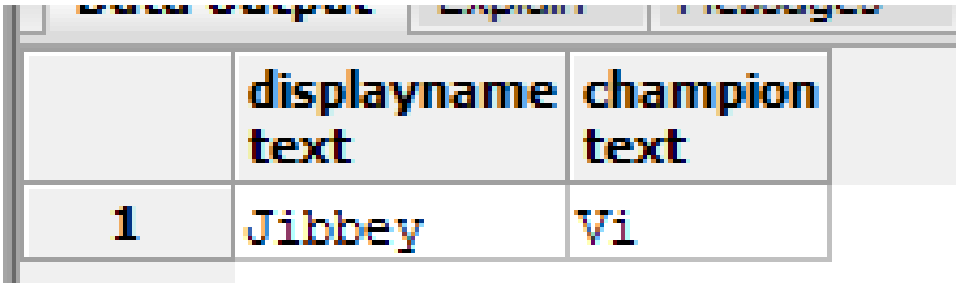
currChampion

Finds a players current champion, takes in the players name as a parameter.

```
create or replace function currChampion(text, REFCURSOR) returns refcursor as
$$
declare displayNameVar text :=$1;
resultset REFCURSOR := $2;
begin
        open resultset for
        select displayName,Champion
        from currentGame
        where displayName = displayNameVar;
        return resultset;
end;
$$ language plpgsql;
```

Sample Query:

```
select * from currChampion('Jibbey','results');
fetch all from results;
```



	displayname text	champion text
1	Jibbey	Vi

Security

We cannot trust players to report their wins or loses so everything is handled by the administrator. But, LCS has a separate league going on which we need a role for. I create an LCSadmin role and he can only edit the proTeam and matchScheduler table so he cannot accidentally mess up the users public data.

```
CREATE ROLE admin;
GRANT SELECT, INSERT, UPDATE
```

```
ON ALL TABLES IN SCHEMA PUBLIC
TO admin;

CREATE ROLE LCSadmin;
GRANT SELECT, INSERT, UPDATE
ON proTeam, matchScheduler
To LCSadmin;
```

Implementation Notes

This system took a long time to implement. I barely scratched the surface with sample data and there could be millions of rows in these tables like there probably is a RIOT HQ. The biggest difficulty was setting up the currentGame table. This was hard because the data gets dropped once the game ends and is then pushed to match history which I have no idea how to implement but I believe that's how it works. Putting in sample data took a long time to do, and I only used 12 champion of the 137 available that could all go into that table. I used my friends and I for the sample data and pro teams TSM and C9 for the pro team data. The stats are all realistic.

I don’t know of any know problems but I think that the matchScheduler could be more detailed and create time for certain teams and decide who is on what side (red or blue) per game. Also you could add best of threes into it somehow too.

There are many future enhancements that can be done. For one you could create a function that calculates the players current champion stats by adding the item data to the base stats data. I tried to do that but I could not figure it out after an hour of work on just that. I’m sure it is possible but I am not that proficient in SQL yet. There could be more matchHistory data I only have one game in there right now. This systems is only a starting point, the real system used by League of Legends is much more complex and detailed and I wish I could see there ER diagram to see how it actually works because it seems very complex after trying to make it simple in my project.