League of Legends Tournament Database

Fall 2019

Project Phase 5

Team details:

Name : Jeyamaruthi Jayakumar

ID : 1001757737

Email : Jeyamaruthi.jayakumar@mavs.uta.edu

Title : League of legends – Tournament Database.

Team-ID : Team-21

Project Description:

Brief:

Project Name ---- League of Legends Tournament Database

The objective of the project is to build a database to monitor and compare the match, player, champion, item details which takes place in different tournaments.

(EER Diagram → Page 7)

Detailed (Phase 1 for reference):

The project holds all the data of what happens in different matches which are encapsulated into a specific entity known as the tournament, where the tournament entity holds multiple tournament id. Let's investigate a much more detailed point of view with the help of entities:

- Currently, the prototype of the project has 6 entities to maintain the database for the league of legends tournament detail database.
- The 6 entities are as follows:

1. **Tourny_details:** Consist of data values which describes tournament details. It contains 5 attributes such as:
   - **T_ID(PK)**: It represents Tournament ID, T_ID's are unique with respect to the tournament Name. T_ID is the primary key for table 'Tourny_details'. E.G: 1001 for LCS. Datatype – INT.
   - **T_Name:** It represents Tournament Name. The tournament Name has specific functionality. For example. Tournament conducted in North America is stated as "LCS", whereas Tournament conducted in Europe is stated as "LEC". Datatype – VARCHAR.
   - **No._of_Teams:** It represents No of teams participating in a specific tournament. It is integer datatype consists of numerical value for no of the team participating. Datatype – INT.

- **Winning_Prize:** It represents the prize pool for the specific tournament.
  Datatype – VARCHAR.
- **MVP:** It represents a player who has most valued points across the tournament. It is like the player of the tournament award received in soccer.
  Datatype – VARCHAR.

2. **Match:** Consist of data concerning the match which has taken place on a specific date. It contains 5 attributes as follows:
   - **Match_ID(PK):** It represents Match ID for a specific match. Match_ID is a primary key for the table 'Match_ID'.
     E.G: 1001-001 for any match taking place in LCS.
     Datatype – INT.
   - **TT_ID(FK)**: It is a foreign key from the table 'Tourny_Detail'. Which represents that each match falls under 1 specific tournament
   - **Team_1:** It represents a team participating in a specific match.
     Datatype – VARCHAR.
   - **Team_2:** It represents the second team participating in a specific match.
     Datatype – VARCHAR.
   - **Day:** It represents the day in which the match has been conducted.
     Datatype – DATE.

3. **Teams:** Consist of detailed data regarding the teams participating in the tournament/match. It contains 6 attributes as follows:
   - **M_ID(FK):** It is a foreign key from the table 'Match'.
   - **Team_Name(PK):** It represents the team name, this is the primary key for the table which holds all the information.
   - **No_of_Players:** It holds the total amount of players in the team.
     Datatype – INT.
   - **No_of_Matches:** It holds the data regarding the number of matches the current team has played.

Datatype – INT.

4. **Player_Details:** This table gives detailed information about the player who belongs to any of team in the table 'Teams'. It contains 10 attributes as follows:
   - **TT_Team_ID(FK):** It is a foreign key from the table 'Teams'.
   - **Player_Name:** It holds the data of the player's 'In-Game Name' (IGN)
     Datatype – VARCHAR.
   - **Real_Name:** It holds the data of player's real name i.e. both first & last name.
     Datatype – VARCHAR.
   - **Role:** It represents which role player is playing. E.G. if the player belongs to Top lane, it holds data such as Top.
     Datatype – VARCHAR.

5. **Champions:** This table gives detailed information about the champion which was played on the stage. It contains 8 attributes as follows:
   - **Champion_ID(PK):** It holds the unique data - ID to represents the champion.
     Datatype – INT.
   - **Champion_Name:** It holds the data of champions played in the stage.
     Datatype – VARCHAR.
   - **Items:** It contains the items built in a specific match for the champion played by the player.
     Datatype – VARCHAR.

6. **Champion_Stats:** This table holds states of the champion played by the player. It contains 6 attributes as follows:
   - **Stats_ID:** It is the primary key of the table. Holds a unique stats value for the champion played by a player.
     Datatype – INT.
   - **Kills:** Holds the value of No. of kills in a specific match.
     Datatype – INT.
   - **Deaths:** Holds the value of No. of deaths in a specific match.

Datatype – INT.

- **Assists:** Holds the value of No. of assists in a specific match.
  Datatype – INT.
- **Damage:** Holds the data of total damage done by a champion in a match.
  Datatype – INT.
- **Wards:** Holds the data of no. of wards placed.
  Datatype – INT.

Apart from the above description there are few update done in the database which is represented in the below ER diagram

EER Diagram (Desc):

The below EER diagram is a updated version of previous prototype which was submitted in **Project Phase 1** :

Few added entities are:

7. **Match_Overview:** This table hold the match overview of a particular match. The attributes are such as:
   - **Gold_Advantage:** Holds the total gold advantage of the winning team.
   - **Total_Kills_T:** Hold an integer value of total kills in entire game
   - **Total_Assist_T:** Hold an integer value of total Assist in entire game.

8. **Sponsor_Details:** This table hold the sponser's detail of a team. The attributes are such as:
   - **T_Team_ID(FK):** It is foreign key from table Teams.
   - **Sponsor_Name:** It is varchar attribute which holds the Name of the sponsor of a particular team
   - **Contract_Date:** Holds the date in which contract ends

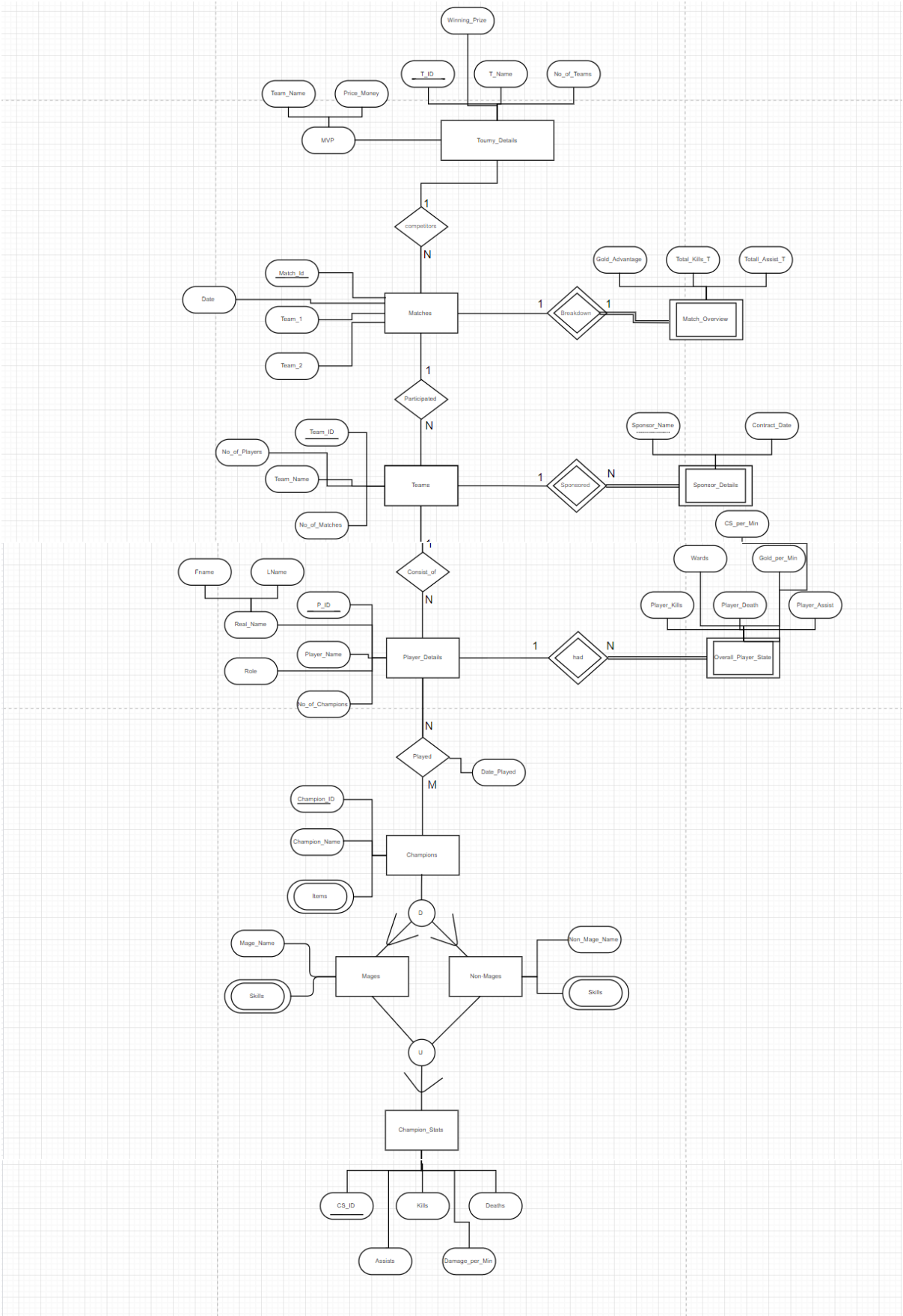
9. **Overall_Player_State:** This table holds the overall single player state upon entire tournament.
    - **PP_ID(FK):** It is foreign key from table Player_Details
    - **Player_Kills:** Hold an integer value of total kills in the entire tournament
    - **Player_Death:** Hold an integer value of total deaths in the entire tournament
    - **Player_Assist:** Hold an integer value of total assists in the entire tournament
    - **Wards:** Hold an integer value of total ward placed in the entire tournament
    - **Gold_Per_Min:** Hold an integer value of gold per min in the entire tournament
    - **CS_Per_Min:** Hold an integer value of minnion kills(creep score) per min in the entire tournament

10. **Mages:** This table holds the detail of Mage champions:
    - **C_ID(FK):** It is foreign key from table Champions
    - **Mage_Name:** It holds a varchar value of mage champion's Name
    - **CS_ID(surrogate key):** It is surrogate key from table Champion_Stats
11. **Mages:** This table holds the detail of Mage champions:
    - **C_ID(FK):** It is foreign key from table Champions
    - **Non_Mage_Name:** It holds a varchar value of non-mage champion's Name
    - **CS_ID(surrogate key):** It is surrogate key from table Champion_Stats

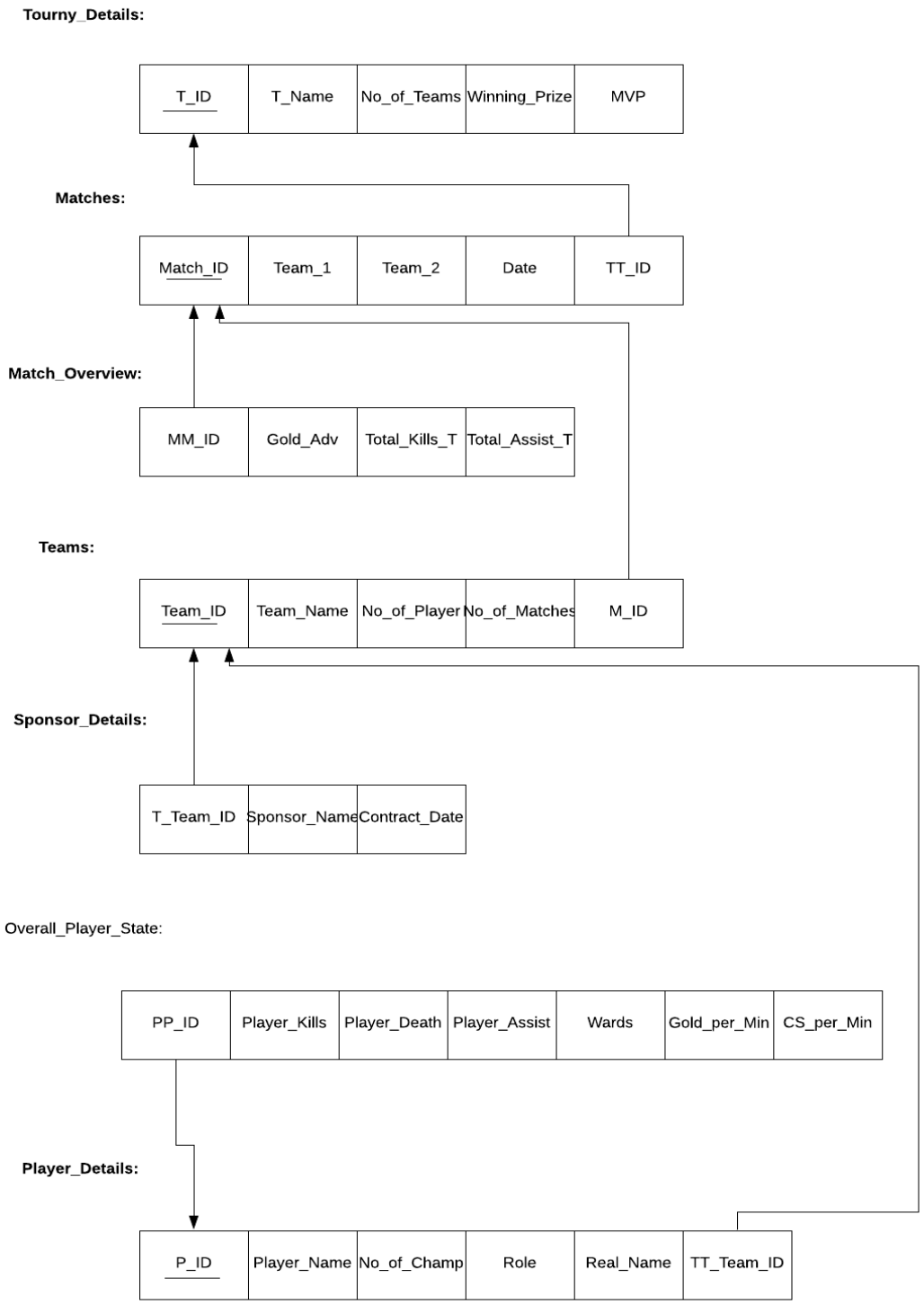
Generalization/Specialization:

- **There is partial disjoin for entity champion**
- **There is union class named "champion_stats"**

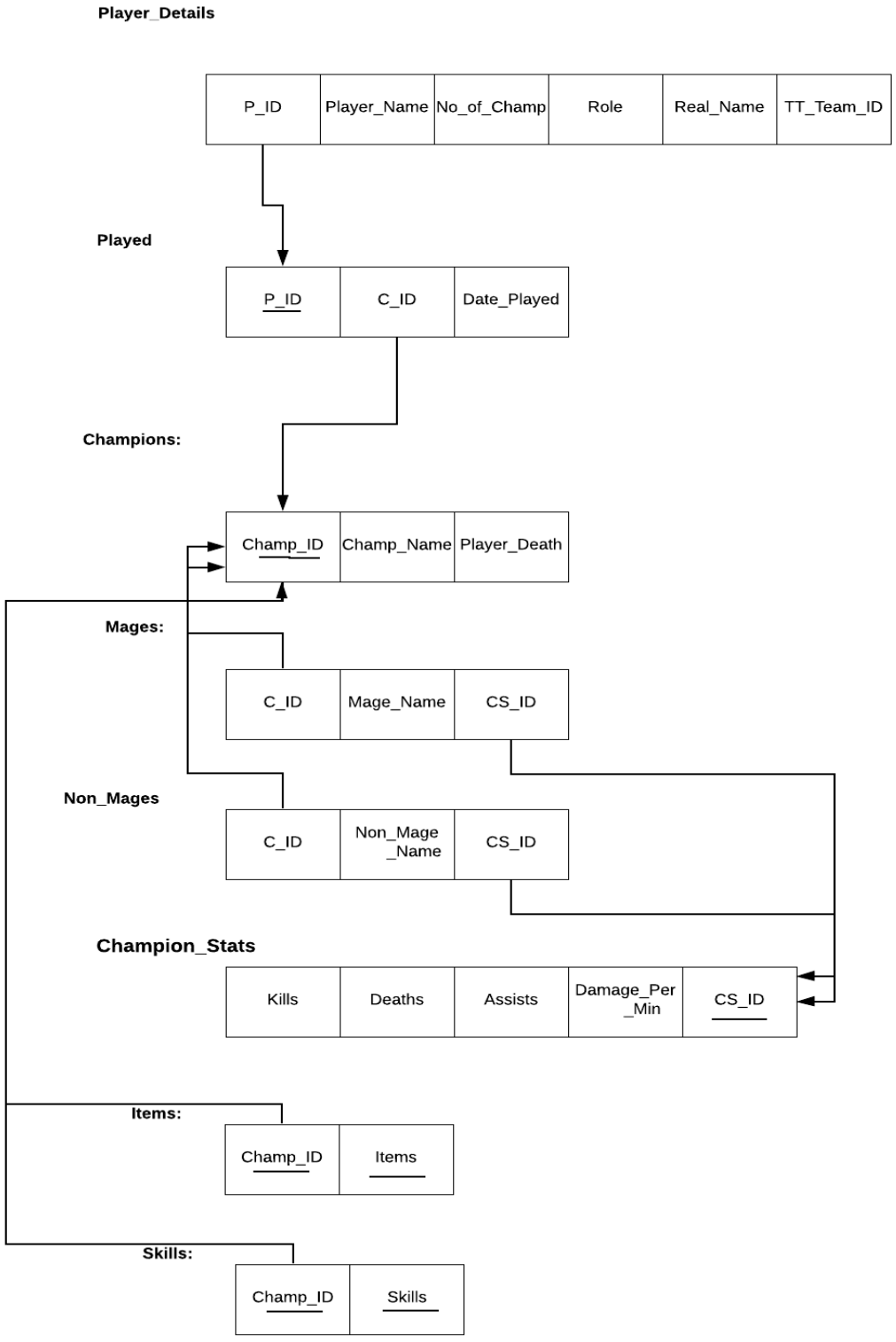
EER DIAGRAM:


Winning_Prize
T_ID
T_Name
No_of_Teams
Team_Name
Price_Money
MVP
Tourny_Details
1
competitors
N
Match_Id
Date
Team_1
Team_2
Matches
1
Breakdown
1
Gold_Advantage
Total_Kills_T
Totall_Assist_T
Match_Overview
1
Participated
N
Team_ID
No_of_Players
Team_Name
No_of_Matches
Teams
1
Sponsored
N
Sponsor_Name
Contract_Date
Sponsor_Details
CS_per_Min
Wards
Gold_per_Min
Player_Kills
Player_Death
Player_Assist
Overall_Player_State
Fname
LName
Real_Name
P_ID
Player_Name
Role
No_of_Champions
Consist_of
N
Player_Details
1
had
N
N
Played
Date_Played
M
Champion_ID
Champion_Name
Items
Champions
D
Mage_Name
Skills
Mages
Non-Mages
Non_Mage_Name
Skills
U
Champion_Stats
CS_ID
Kills
Deaths
Assists
Damage_per_Min

EER to a readable Relational DB schema (Part1):



EER to a readable Relational DB schema (Part2):



10 relational algebra expressions:

1. **Team with gold advantage > 4000 Gold:**

$$\sigma_{(gold\_advantage>4000)}(Teams)$$

2. **Player with role = “Mid” & number of champions played > 6:**

$$\sigma_{(role=”Mid”\wedge No\_of\_Matches>6)}(Player\_Details)$$

3. **Player with role = “Jungle” and has played champion named “Ekko”:**

$$\sigma_{(role=”Jungle”)}(Player\_Details) \wedge_{(Champion\_Name=”Ekko”)}(Champions)$$

4. **All the player’s kills & death from overall player stats:**

$$\Pi_{(Player\_Kills,Player\_Deaths)}(Overall\_Player\_State)$$

5. **Fetch player names & Role of a player whose number of champions played is greater than 10.**

$$\Pi_{(player\_Name,Role)}(\sigma_{(No\_of\_Champions>10)}(Player\_Details))$$

6. **TeamID & Name of the team whose sponsor name =”Red Bull”**

$$\Pi_{(Team\_ID,Team\_Name)}(\sigma_{(Teams.Team\_ID=Sponsor\_Details.Team\_ID\wedge sponsor\_Name =”Red\_Bull”\wedge No\_of\_Player=7)}(Teams \times Sponsor\_Details))$$

(OR)

$$\Pi_{(Team\_ID,Team\_Name)}(\sigma_{(sponsor\_Name =”Red\_Bull”\wedge No\_of\_Player=7)}(Teams \bowtie Sponsor\_Details))$$

7. **Fetch Champion Names & IDs who has played the role = “Top” & whose gold per min > 10.**

$$\Pi_{(champion\_Name,champion\_ID)}(\sigma_{(role=”TOP”\wedge Gold\_Per\_min>10)}(Champions \bowtie (Player\_Details \bowtie Overall\_Player\_State))$$

8. **LEFT OUTER JOIN: Whole player details (complete tuple) who have played with championID = “CI1001”:**

$$Player\_Details \bowtie_{player\_details.champion\_ID=Champion.champion\_ID} Champion$$

9. **UNION OPERATOR: Lists of Mage Champion Names Union Non-Mage Champion Name.**

$$\rho_{(CNAME)}(\Pi_{(Mage\_Name)}(Mages)) \cup \rho_{(CNAME)}(\Pi_{(NON\text{-}Mage\_Name)}(Mages))$$

10. **MINUS OPERATOR: Fetch Champion ID & Name which wasn’t played by any player**

$$\Pi_{(champion\_Name)}(\Pi_{(champion\_ID}(Champion) - \Pi_{(champion\_ID}(Player\_Details)) \bowtie Champions)$$

Primary Keys, Foreign Keys:

For detail description of the keys, check the detailed project description where keys are brief for understanding.

- **Tourny_Details:**
    - **Primary keys: T_ID**
    - **Foreign keys: NULL**
- **Matches:**
    - **Primary keys: Match_ID**
    - **Foreign keys: TT_ID**
- **Match_Overview:**
    - **Primary keys: NULL**
    - **Foreign keys: MM_ID**
- **Teams:**

    - **Primary keys: Team_ID**
    - **Foreign keys: M_ID**
- **Sponsor_Details:**
    - **Primary keys: NULL**
    - **Foreign keys: T_Team_ID**
- **Overall_Player_State:**
    - **Primary keys: NULL**
    - **Foreign keys: PP_ID**
- **Player_Details:**
    - **Primary keys: P_ID**
    - **Foreign keys: TT_Team_ID**
- **Champions:**
    - **Primary keys: Champ_ID**
    - **Foreign keys: None**
- **Mages:**
    - **Primary keys: NULL**
    - **Foreign keys: C_ID**
    - **Surrogate keys: CS_ID**
- **Non-Mages:**
    - **Primary keys: NULL**
    - **Foreign keys: C_ID**
    - **Surrogate keys: CS_ID**
- **Champion_States:**
    - **Primary keys: NULL**
    - **Foreign keys: C_ID**
    - **Surrogate keys: CS_ID**

Queries Used:

Query1:

Give Matches where Advantage is greater than "**Input Value" :**

```
SELECT * from match_overview where Gold_Advantage > '$gold';
```

Query2:

Display Team1 & Team2 participated on date = "**Input Value" :**

```
SELECT * from matches where Date = '$date';
```

Query3:

Count number of players who has CS per Min = "**Input Value" :**

```
SELECT CS_Per_Min, Count(*) from overall_player_state where CS_Per_Min = '10';
```

Query4:

Display the players who played number of champions between "**Input Value" to** "**Input Value" :**

```
SELECT * from player_details where No_Of_Champions between '$c1' and '$c2';
```

Query5:

Display Team Name and sponsor Name whose number of players = "**Input Value"**

```
SELECT Team_Name,Sponsor_Name from teams inner join sponsor_details on
Team_ID = Teams_Team_ID and No_Of_Players = '$c1';
```

Query6:

Fetch Player's Name and Roles whose kills are greater than "**Input Value"**

```
SELECT Player_Name,Role from player_details inner join overall_player_state on
P_ID = Player_Details_P_ID and Player_Kills > '$kills';
```

Query7:

Display team name and sponsor name whose number of player = **"Input Value"**

```
SELECT Team_Name,Sponsor_Name from teams inner join sponsor_details on
Team_ID = Teams_Team_ID and No_Of_Players = '$player';
```

Query8:

Display the player Name along with their number of mages played:

SELECT Player_Name,Count(*) from player_details inner join mages on P_ID = Champions_Player_Details_P_ID group by Champions_Player_Details_P_ID;

Query9:

Fetch sum of player kills along with CS per Min & No of wards whose number of wards = **“Input Value”:**

SELECT sum(Player_Kills), CS_Per_Min, Wards from overall_player_state

group by CS_Per_Min having wards = '$champ' order by Wards;

Query10:

Display Tournament Name, Match ID & Team Name which was conducted on date = **“Input Value”.**

SELECT T_Name,Match_ID,Team_Name from tourny_detials,matches,teams

where Matches_Match_ID = Match_ID and Tourny_Detials_T_ID = T_ID and Date = '$tid';

Conclusion/learned from the project (Phase1):
The advantage of learning the database system via creating this project will be useful for the future real world, where students can utilize the gathered knowledge while creating the database and implementing it along with queries which can be displayed.
The described project in this document – League of Legends Database helps me to understand the overall Database systems.

Phase II: Helps students understand a pre-requirement gathering in the form of EER diagram & Relational algebra, which helps in real world to reduce the overall defect & human activities.

Phase III: Helps student to understand the entire Physical Implementation of database along with different script such as DDL, DML, Drop & advance SQL queries.

Phase IV: Helps student to understand how a real time application will take place with back end process utilizing Database Systems.

Represented Demo successfully and got the desired result as per the user need.

Hence, the overall project is submitted successfully.

Thanks for this wonderful opportunity to do the former project.