|  |  |  |  |
| --- | --- | --- | --- |
| **Campus** | **NPI** |  | Please Tick (✓) for  **Re-do Assignment** |
| **APU Foundation & Degree**  **Programmes** | |  |

**Coursework Submission and Feedback Form**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** |  | | | | **Name of Group Members (If Applicable)** |
| **Student No.** |  | | **Intake:** | **NPI1F1909IT** | **Nabin Chhetri**  **(NPI000032)** |
| **Module Code & Title** | **Introduction to Database (CT042-3-1)** | | | | **Sandesh Giri**  **(NPI000041)** |
| **Assignment Title** | **Design and document a database system** | | | | **Sandesh Subedi ‘A’**  **(NPI000040)** |
| **Name of Lecturer** | **Mr.Bidur Devkota** | | | | **Suraj Pandey**  **(NPI000051)** |
| **Date Due** | **24 February,2021** | **E-Mail: girisandy122@gmail.com** | | |  |

*I have read and understood the regulations on Plagiarism and Academic Dishonesty and declare that the work submitted does not breach those regulations.* **Signed: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**You must hand in to the designated APU Administrator – ensure that you receive your receipt**.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Received By** |  | **Signature** |  | **Date** |  | **Time** |  |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Criteria** | | | **Weighting** | | **Fail** | **Marginal**  **Fail** | **Pass** | | **Credit** | | **Distinction** |
|  | | |  | |  |  |  | |  | |  |
| **Additional Comments:** (These may be listed below or attached) | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Provisional Assessment Result: Grade Date Lecturer’s Initial**    Taking account of above factors, the overall provisional assessment of your work is: | | | | | | | | | | | |
|  | | | | | | | | | | | |
| **Distinction** | | **Credit** | | | | **Pass** | | | | | |
| **A+: 80-100%** | **A: 75-79%** | **B+: 70-74%** | | **B: 65-69%** | | **C+: 60-64%** | | **C: 55-59%** | | **C-: 50-54%** | |
| **Marginal Fail** | | **Fail** | | | | | | | | | |
| **D: 40-49%** | | **F+: 30-39%** | | | | **F: 20-29%** | | | **F-: 0-19%** | | |

*The comments and assessment result are subject to both internal and external moderation at the appropriate Examination Board.*

*Consequently, they may not reflect your final grade. You may not appeal against these result on ground of Academic Judgement.*



**ASSIGNMENT**

**INFOMAX COLLEGE OF IT AND MANAGEMENT**

***(CT042-3-1)***

***(INTRODUCTION TO DATABASES)***

**HAND OUT DATE: (10/12/2020)**

**HAND IN DATE: *(24/02/2021)***

**WEIGHTAGE: *(50) %***

**INSTRUCTIONS TO CANDIDATES:**

1. **Submit your assignment at the administrative counter.**
2. **Students are advised to underpin their answers with the use of references (cited using the Harvard Name System of Referencing)**
3. **Late submissions will be awarded zero(0) unless Extenuating Circumstances are upheld.**
4. **Cases of plagiarism will be penalized.**
5. **The assignment should be bound in appropriate style (comb bound or stapled)**
6. **Where the assignment should be submitted in hardcopy and softcopy, the softcopy of the written assignment and source code (where appropriate) should be on a CD in an envelope / CD cover and attached to the hard copy.**
7. **You must obtain 50% overall to pass this module.**

Team members:

| TEAM MEMEBERS | STUDENT NUMBER | Signature |
| --- | --- | --- |
| SURAJ PANDEY | NPI000051 |  |
| SANDESH SUBEDI | NPI000040 |  |
| NABIN CHHETRI | NPI000032 |  |
| SANDESH GIRI | NPI000041 |  |

1. Table of Contents

[Abstract 6](#_Toc67574596)

[Acknowledgement 6](#_Toc67574597)

[Disadvantages of file based system 6](#_Toc67574598)

[Advantages of database and DBMS 7](#_Toc67574599)

[Relation with case study 8](#_Toc67574600)

[Business rules and Normalization 8](#_Toc67574601)

[Business Rules: 8](#_Toc67574602)

[Normalization 9](#_Toc67574603)

[1NF (First Normal Form) 9](#_Toc67574604)

[2NF (Second Normal Form) 10](#_Toc67574605)

[3NF (Third Normal Form) 12](#_Toc67574606)

[Entity Relationship Diagram(ERD) 14](#_Toc67574607)

[Identification of entities 15](#_Toc67574608)

[Identification attributes of entities 16](#_Toc67574609)

[Description of attributes of each entity 18](#_Toc67574610)

[Simple ERD (Entity Relationship Diagram) 21](#_Toc67574611)

[Identification of relationship between entity 21](#_Toc67574612)

[One to one (1:1) 21](#_Toc67574613)

[One to many (1:M) 22](#_Toc67574614)

[Many to one (M:1) 22](#_Toc67574615)

[Many to many(M:M) 23](#_Toc67574616)

[Constraints 24](#_Toc67574617)

[Logical constraints 24](#_Toc67574618)

[Null constraints 24](#_Toc67574619)

[All attributes hold data which is important for an entity and should be ready to facilitate when required, so they cannot be null. Following expressions shows the importance attributes: 24](#_Toc67574620)

[Unique Constraints 24](#_Toc67574621)

[Primary Key Constraints 24](#_Toc67574622)

[Default Constraints 25](#_Toc67574623)

[Workload matrix 25](#_Toc67574624)

[Database Schema 26](#_Toc67574625)

[1. Finalized ERD (Entity Relationship Diagram) 26](#_Toc67574626)

[2. Conversion of ERD (Entity Relationship Diagram) into Relational Schema 27](#_Toc67574627)

[Normalization of all relation up to 3NF: 32](#_Toc67574628)

[Table formed after normalisation up to 3NF are: 38](#_Toc67574629)

[Data Dictionary: 41](#_Toc67574630)

[Database Diagram: 47](#_Toc67574631)

[SQL-Data Definition Language SQL-DDL 48](#_Toc67574632)

[Create database: 48](#_Toc67574633)

[Insert data in table: 52](#_Toc67574634)

[2. SQL-Data Manipulation Language (DML): 58](#_Toc67574635)

[Question **1:** 58](#_Toc67574636)

[Question 2: 58](#_Toc67574637)

[Question 3: 58](#_Toc67574638)

[Question 4: 59](#_Toc67574639)

[Question 5: 59](#_Toc67574640)

[Question 6: 59](#_Toc67574641)

[Question 7: 60](#_Toc67574642)

[Question 8: 60](#_Toc67574643)

[Question 9: 60](#_Toc67574644)

[Question 10: 61](#_Toc67574645)

[Question 11: 61](#_Toc67574646)

[Question 12: 61](#_Toc67574647)

[Workload Matrix 62](#_Toc67574648)

[References 62](#_Toc67574649)

# Abstract

The main purpose of this project is to design complete database system for the online mobile game called *BingBing* legend. The project starts with the identification of business rules of the game. After the identification of business rules, entities, attributes and relationship between the entities, ERD (Entity relationship diagram) is formed. Then the ERD is converted into its corresponding relational schema and normalization is being carried out to design a perfect database system. Finally, all the tables are created using MySQL and database schema is implemented.

# Acknowledgement

We would like to exhibit our genuine gratefulness to everyone who assisted us to complete this database design task. Firstly, we’d love to display extraordinary thankfulness to our subject teacher, Dr. Bidur Devkota. His counsel and encouragement lead us to complete our assignment task within proper time along with appropriate guidelines. Additionally, we’d love to applaud our mates who helped us in collecting required data. Without their cooperation, this paper would not be as good.

# Disadvantages of file based system

* **Data Redundancy and Instability**:

Though data are occupying in different data files, so there is possibility of redundancy and instability in data. For example, if the same customer has 2 different saving account as well as mortgage loan. So, here the customer details can be duplicated after all the programs for two different functions keep their similar data in two different data files. So, this may result to redundancy in customer data. If the same data is kept in 2 files, then it gives rise to instability.

* **Low Security**:

Security of data is major concern in database management system. Customers or users should be only accessible to required records rather than whole database. But this is not in practice in case of file based system, due to which data may get lost. Hence, data are not secured in file based system.

* **Unexpected Queries**:

Handling sudden/ad-hoc queries is so tough in file based system. It requires change in existing programs or application. This system is relatively uncomplicated but it cannot handle complicated queries and data recovery.

* **Chances of error and time consuming**:

In file based systems, there is likely to produce errors due to modifications. Thus, it requires more and extra time to debug the programs and correct those errors.

* **Integrity Problems**:

There can be definite data integrity rules which need to be maintained in any applications or programs. It may be noted that we are not claiming to say that we cannot handles all those above problems in file management system but the real issue is that each applications or programs had to handles all those problems of their own.

# Advantages of database and DBMS

1. **Improved data sharing:**

An advantage of the database management approach is that it helps to establish an environment in which more and better-managed information is allowed to user. It gives users the capacity to create more information from given data and also permit for data sharing among employees and also to other people who have access to the system.

1. **Improved data security:**

A DBMS produces a framework for better and greater fulfillment of data privacy and security policies. Each user can have a distinct set of access so data is secured from different negative issues like identity theft, leaks and misuse of data.

1. **Control of redundancy of data:**

Redundancy of data is removed by data normalization. If there is no duplication of data, then it may improve access time and also saves storage. Redundancy is controlled by application programming and kept to least by initiating as little redundancy as possible when making the database.

1. **Minimized Data Inconsistency:**

As we know, the root cause of data inconsistency is redundancy of data because of involvement in normalization of data. It exists when different version of identical data emerges in different place.

1. **Improved data access:**

Database management system makes possible to provide quick answer to ad-hoc queries. A query is a certain request provided to DBMS for data manipulation from a perspective of database. Database controls data in such a way that data can be easily

1. **Improved decision making**:

Better data management and improved data access allow quality information on the basis of better decisions. The quality of generated information associates to the quality of underlying data. Decision making results in a framework to ease data quality initiatives.

# Relation with case study

Overall, Database and database management systems are more effective to run, implement and design as per the requirement for “BingBing legend” mobile game than file-based systems. It permits the user to check the data by creating, controlling, defining, insertion, deletion and revising. It makes it possible for multiple authorized users to gain the same database in different ways from different places to complete different intents.

# Business rules and Normalization

## Business Rules:

Business rule helps us to identify the entities, attributes and relation. This further helps us to design entity relationship diagrams. So, the business rules according to the given scenario are listed below:

* A distinctive email address and username is required during player registration.
* At least one hero must be owned prior to any skin purchase.
* Only 5 (no more or less) active-players are permitted in a team.
* Heroes must be selected uniquely within a team during the battle.
* Skins and heroes can only be procured with diamonds or battle points.

Normalization

As per the requirement of the question let us take an example of UNF (un normal form) from given scenario which is given below:

| Player Name | Address | Clubs | Rank Name |
| --- | --- | --- | --- |
| Lionel Messi | Catalonia | FC Barcelona | Commander BE |
| Neymar Jr | Paris | Paris SG, FC Barcelona | Member BE |
| Alex Morgan | Madrid | Bayern CF, Real CF | Member BE |
| Neymar Jr | Manchester | Manchester Utd | Order BE |

Table1: Un Normal Form (UNF)

The table above is an example of un normal form. It consists of multivalued attributes in same tuple due to which it lacks efficiency and may face multiple issues like data redundancy or anomalies. Hence, it should be further normalized.

### 1NF (First Normal Form)

To perform first normal form, we need to follow certain conditions which are listed below:

* Domains of all its attributes must be atomic.
* The value of any attribute in a record is a single valued from its domain.

So, the 1NF of given UNF example after fulfilling all the conditions of 1NF table appears as bellow:

| Player Name | Address | Clubs | Rank Name |
| --- | --- | --- | --- |
|  |  |  |  |
| Lionel Messi | Catalonia | FC Barcelona | Commander BE |
| Neymar Jr | Paris | Paris SG | Member BE |
| Neymar Jr | Paris | FC Barcelona | Member BE |
| Alex Morgan | Madrid | Bayern CF | Member BE |
| Alex Morgan | Madrid | Real CF | Member BE |
| Neymar Jr | Manchester | Manchester Utd | Order BE |

Table 2: First Normalized Form (1NF)

After the table is normalized into first normal form by following required conditions, each tuple must be distinct. In table above, there are two players with same name, therefore we need to consider both name and address to identify a tuple distinctly. Doing that will create a composite key, which is a primary key composed of numerous columns to determine a tuple distinctively.

### 2NF (Second Normal Form)

Conditions of 2NF are:

* Table must be in 1NF.
* Relation must not contain any partial dependency.
* Table should only contain data of one type of thing.

**Partial dependency:** If non-key attributes are functionally dependent on part of a candidate key then it is called partial dependency.

**2NF**

| Player Registration No. | Player Name | Address | Rank Name |
| --- | --- | --- | --- |
|  |  |  |  |
| FIFA1001 | Lionel Messi | Catalonia | Commander BE |
| FIFA1002 | Neymar Jr | Paris | Member BE |
| FIFA1003 | Alex Morgan | Madrid | Member BE |
| FIFA1004 | Neymar Jr | Manchester | Order BE |

Table 3.1 : Player Information Table

| Player Registration No. | Clubs |
| --- | --- |
|  |  |
| FIFA1001 | FC Barcelona |
| FIFA1002 | Paris SG |
| FIFA1002 | FC Barcelona |
| FIFA1003 | Bayern CF |
| FIFA1003 | Real CF |
| FIFA1004 | Manchester Utd |

Table 3.2 : Club Information Table

As there was no other ways to simplify the table, we need to divide the table into two different tables. With this, The first table displays information about players while the second one displays club information. A new column named ‘Player registration No.’ is introduced as a primary key so that the data can be uniquely identified.

* **Player Registration No. Is the foreign Key in Club Information table**

In Club Information Table (Table 3.2), Player Registration Table is the foreign key.

A foreign key is basically a set of attributes that is accustomed as a reference of primary key in discrete table. It is used to connect two tables together. In Table 3.2, Player Registration No. Is referred as a foreign key which was primary key column in Table 3.1 (Player Information Table).

### 3NF (Third Normal Form)

Condition 3NF are:

* Table must be in 2NF
* Must not contain transitive dependency.

**Transitive dependency:** If any non-key attribute is functionally dependent up on another non key attribute then the relation is called transitive dependency.

| Player Registration No. | Player Name | Address | Rank Name |
| --- | --- | --- | --- |
|  |  |  |  |
| FIFA1001 | Lionel Messi | Catalonia | Commander BE |
| FIFA1002 | Neymar Jr | Paris | Member BE |
| FIFA1003 | Alex Morgan | Madrid | Member BE |
| FIFA1004 | Neymar Jr | Manchester | Order BE |

Table 4.1

There is transitive dependency between Player’s Name and Rank Name. The Rank Name is dependent upon Player Name. Modifying the name of player may alter the rank name of the player too. Hence, there is transitive dependency present in the table which does not fulfill the condition of Third Normal Form.

**Conversion into 3NF,**

| Player Registration No. | Player Name | Address | Rank Name ID |
| --- | --- | --- | --- |
|  |  |  |  |
| FIFA1001 | Lionel Messi | Catalonia | CBE-1 |
| FIFA1002 | Neymar Jr | Paris | MBE-1 |
| FIFA1003 | Alex Morgan | Madrid | MBE-1 |
| FIFA1004 | Neymar Jr | Manchester | OBE-1 |

Table 4.2

| Player Registration No. | Clubs |
| --- | --- |
|  |  |
| FIFA1001 | FC Barcelona |
| FIFA1002 | Paris SG |
| FIFA1002 | FC Barcelona |
| FIFA1003 | Bayern CF |
| FIFA1003 | Real CF |
| FIFA1004 | Manchester Utd |

Table 4.3

| Rank Name ID | Rank Name |
| --- | --- |
|  |  |
| CBE-1 | FC Barcelona |
| MBE-1 | Paris SG |
| MBE-1 | FC Barcelona |
| OBE-1 | Bayern CF |

Table 4.4

Following rules and conditions of Third Normal Form, the table in 2NF is divided and new table is created. The new table stores Rank Name of each player. The Rank Name ID in Table 4.3 while it is foreign key in Table 4.1.

In this way a table is normalized up to Third Normal Form in order to avoid redundancies and data anomalies. There is no transitive dependency I.e, the table is in Third Normal Form.

# Entity Relationship Diagram(ERD)

“Entity Relationship diagram is a pictorial representation that outlines relationships among people, objects, places, concepts or events within an information technology(IT) system” (B. Jacqueline,2019). Before drawing entity relationship diagrams for database management we need to go through certain steps which helps us in developing perfect ERD. The steps are as follows:

Step 1: Identify entities.

Step 2: Identify attributes of entities.

Step 3: Identify relationships between the entities.

If we are able to perform all the steps mentioned above, then it will be easier for us to design ERD of the system.

## Identification of entities

**Entities:(definition)**

First of all, we need to identify the entities present in a given case study. Entities are represented by rectangular shape in entity relationship diagram. The entities available in the given scenario are as follows:

**ENTITIES**

1. Player
2. account
3. Hero
4. Skin
5. Game-mode
6. Team
7. Personal statistics
8. Game

**Description of each entity:**

|  |  |
| --- | --- |
| Entities | description |
| player | Information of any user of this game will get through a unique player account. |
| Hero | Information about heroes can be owned by players and can be used as per role that user chooses in the game. |
| Skin | The player of this game can equip details of skin for each hero. |
| Game\_mode | All the game-mode is stored by an entity that can be picked and played by the team. |
| Game | This entity holds the record of result and number of participants that played in this game. |
| Team | This entity is related to creating a team consisting five players and the heroes which they picked with their respective levels. In this case, this entity creates game mode very simply and efficiently. |
| Personal statistics | This entity holds the details of player’s and accomplishments. |

## Identification attributes of entities

After identifying the entities present in the given case study we need to find out the attributes of those entity sets which describe the characteristics of each entity present in the entity sets. So the attributes of the above listed entity sets are as follows:

**ATTRIBUTES:**

1. **Player:**
   1. Player\_ID
   2. Player\_name
   3. Email
   4. Username
   5. Global\_level
   6. Diamonds collected
   7. Online\_status
   8. Hero\_lvl
   9. skin\_owned
2. **hero**
   1. Hero\_ID
   2. Hero name
   3. Hero\_Role
   4. Hero\_Speciality
   5. Hero\_price

1. **Skin**
   1. Skin\_ID
   2. Skin name
   3. Skin price
2. **Game\_mode**
   1. Game\_mode ID
   2. Game\_mode type
3. **Game**
4. Game ID
5. Game\_date
6. battle\_duration
7. Game result
8. Arena
9. **Team**
   1. Team\_ID
10. **Personal statistics**
    1. Result history
    2. Game won
    3. Additional detail
    4. nationality

### Description of attributes of each entity

**Entity : player**

|  |  |
| --- | --- |
| **Attributes** | **description** |
| Player\_ID | A distinctive ID given to each player which distinctly identify each information stored on that table of that particular player. |
| Player\_name | To make the message dynamic, the player's real name is used while sending the player emails and updates. |
| Email | Email address of player. |
| Username | Player’s name in game. |
| Global\_level | The global level of player’s viewed publicly. |
| Diamond Collected | This holds the record of collection of diamonds by players. |
| Online\_Status | It is  the boolean process where true indicates active or online and false indicates inactive or offline. |

**Entity : Hero**

|  |  |
| --- | --- |
| **Attributes** | **description** |
| Hero\_ID | A unique primary key indicates each hero and automatically increments. |
| Hero\_name | The names of hero’s like Fanny and Estes,  Clint, Cyclops, Rafaela, Dark Rose, etc. |
| Hero \_Role | Role description of hero’s like Assassin, Fighter, etc. |
| Hero\_Speciality | Specialty description of hero’s like Regen, Charge, etc. |
| Hero\_Price | It is the integer which represents the price of hero’s in diamonds and battle points. |
| Hero\_lvl | Specific played owned their current level of a specific hero. |

**Entity : Skin**

|  |  |
| --- | --- |
| **Attributes** | **description** |
| Skin\_ID | A unique primary key which indicates each skin and automatically increments. |
| Skin\_name | A name of skin is usually one to describe it. |
| Skin\_price | It is the integer which represents the price of skin’s in diamonds and battle points. |
|  |  |

**Entity : Game\_mode**

|  |  |
| --- | --- |
| **Attributes** | **description** |
| Game\_mode\_ID | A unique primary key which indicates each available game mode and automatically increments. |
| Game\_mode type | Classic, Brawl, or human vs AL mode are categorized as specific types of the mode. |

**Entity : Game**

|  |  |
| --- | --- |
| **Attributes** | **description** |
| Game\_ID | A unique primary key which indicates to each that has been played by players and automatically increments. |
| Game\_date | It indicates the date variable in which the game shows the date that has been started. |
| battle \_duration | It is an integer which represents the duration of game in which game can be change it into appropriate form if needed |
| Game\_result | A foreign key which represents the team\_ ID of the team that lost or won the game. |
| Arena | It is the cosmetics and environment of each game mode. |

**Entity : Team**

|  |  |
| --- | --- |
| **Attributes** | **description** |
| Team\_ID | A unique primary key which indicates each team created and automatically increments. |

**Entity : Personal Statistics**

|  |  |
| --- | --- |
| **Attributes** | **Description** |
| **Player\_ID** | It is a foreign key which indicates the Player\_ID that is located in the player in which the player has been linked to their statistics. |
| Most\_used\_hero | It is a foreign key which indicates the Hero\_ID that is located in the hero entity in which the player used the hero most. |
| pervious\_game\_result | It is represented by using boolean in which false indicates the player lost the game and true indicates the player one last game they played. |
| Additional\_result | Player description with their profiles. |

## Simple ERD (Entity Relationship Diagram)

## 

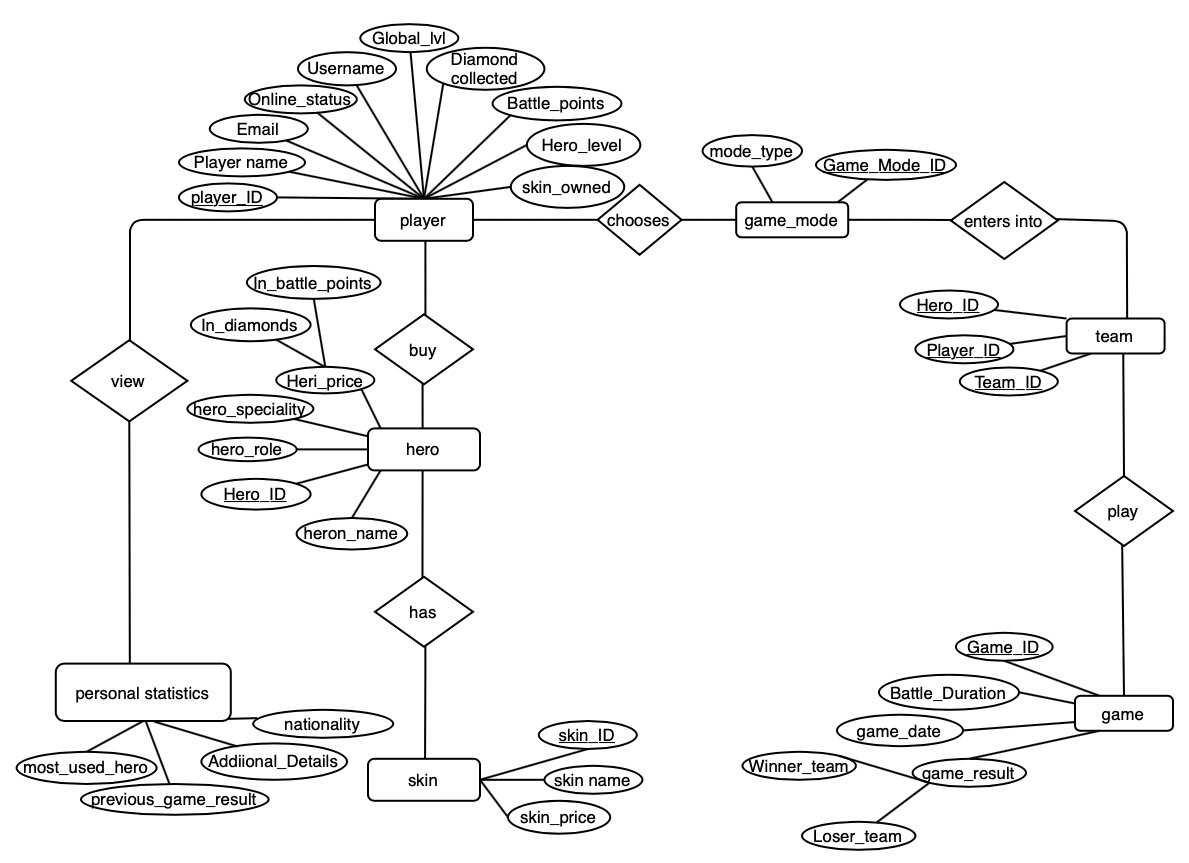


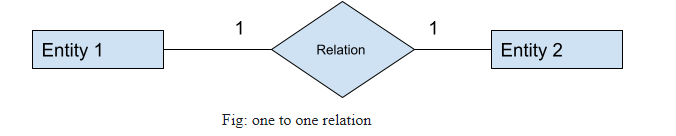
Figure: Simple ERD without cardinality

## Identification of relationship between entity

To draw an entity relationship diagram we need to find out the relation between two different entities which helps us in building ER diagrams easily. There are mainly four types of relation that associate entities with each other which are describe below:

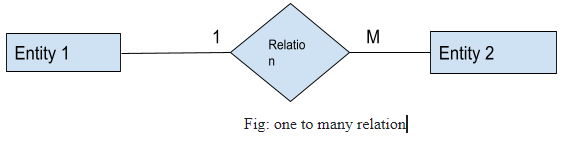
### One to one (1:1)

If one entity of the first entity set is associated with only one entity of another entity set then the relation is called one to one relation.



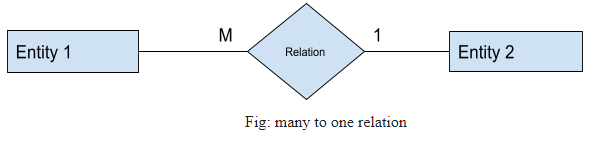
### One to many (1:M)

  If one entity of the first entity set is associated with more than one entity of another entity set is called one to many relation.



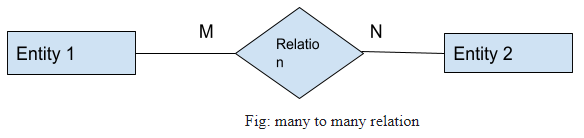
### Many to one (M:1)

Similarly, it is a type of relation in which many entities of the first entity set are linked with only one entity of another entity set .



### Many to many(M:M)

If many entities of the first entity are linked with more than one entity of another entity set, then the relation is said to be in many to many relations.



The relationship between entity according to the given scenario are listed below:

|  |  |
| --- | --- |
| **relationship** | **description** |
| Player M:M Hero | One player can buy many heroes; one hero can be bought by many players. There exists many to many relationships between player and hero. |
| Hero 1:M skin | One hero can have many skins but a skin is associated with only one hero. |
| Player 1:M Personal statistics | A player has more than one personal statistics of different games that he has played. |
| Team M:1 Game\_mode | Many teams can choose the same Game\_mode but only one Game\_mode can be selected by a team for a game. |
| Team M:1 game | A game can be played by only two teams and two teams are compulsory to play a game. |
| Player M:N Game\_mode | A player can choose many game modes and a Game\_mode can also be chosen by many players. |
| Game\_mode 1:M Game | One game is played in only one game mode and the same Game\_mode can be chosen to play different games. |

## Constraints

### Logical constraints

1. The player can buy the skin for a specific hero after getting at least one hero.
2. Using diamonds or points obtained from battle, both skin and heroes can be purchased.
3. Before the game has started, game mode must be chosen.
4. Member requirement is only five in a team.
5. Active players can get a chance in a team.
6. Each team requires a unique hero.

### Null constraints

### All attributes hold data which is important for an entity and should be ready to facilitate when required, so they cannot be null. Following expressions shows the importance attributes:

1. Personal\_statistics table has the details of attributes that can be Null at the beginning of the game when the player will not retain details.
2. Personal\_statistics table has the Most\_used\_hero can be Null at the starting of the game when the player will not have used any heroes.

### Unique Constraints

           Following are list of unique constraint present in given case study:

1. Player's name
2. Username
3. Email
4. Hero's
5. Skin
6. Both the game mode name and Arena
7. Requirement members in team.

### Primary Key Constraints

1. All the primary keys are integers.
2. To be an integer, the primary key can increment automatically with start at 1 and iterate by 1, which makes insertion process easier.

### Default Constraints

1. The default value of the global level attributes of the player entity is 1.
2. The default value of the Diamond attribute of the player entity is 250.
3. The defau;t value of the Battle\_points attribute of the player entity is 3000.
4. The default value of [v] attribute of the player entity is 1.

# Workload matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ASSIGNMENT COMPONENT | SURAJ PANDEY | SANDESH SUBEDI | NABIN CHHETRI | SANDESH GIRI |
| Advantages and disadvantages | 25% | 25% | 25% | 25% |
| Business rules | 25% | 25% | 25% | 25% |
| Normalisation | 25% | 25% | 25% | 25% |
| Entity Relationship Diagram | 25% | 25% | 25% | 25% |
| Signature | Suraj | Sandesh | Nabin | Sandesh |

# Database Schema

## Finalized ERD (Entity Relationship Diagram)

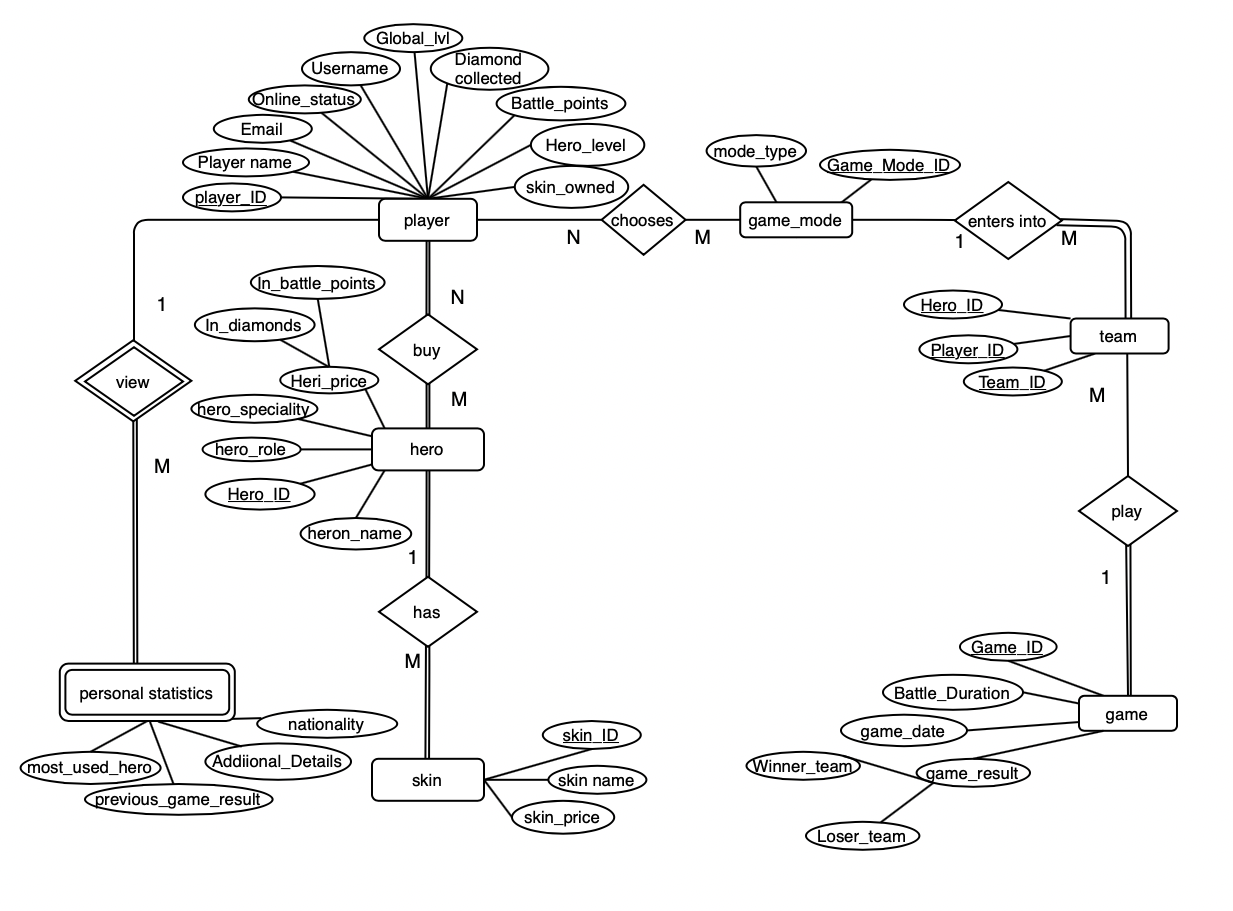
****

Figure: Finalized ERD (Entity Relationship Diagram)

## 2. Conversion of ERD (Entity Relationship Diagram) into Relational Schema

There are certain steps for the conversion of ERD (Entity relationship diagram) to relational schema. While converting ERD to its corresponding relational schema there may have formed more tables than the entity presents in the diagram. This statement gets cleared after reading the following steps. The steps are:

1. First of all, we need to convert each strong entity to its corresponding relation or table.
2. Secondly,convert attributes of the entity as columns for the table.
3. Convert multivalued attribute into new table and add primary key of the parent table as foreign key within the table.
4. Convert weak entities into tables and add each attribute to the table as column and primary key of the identifying entity.
5. Declare foreign key constraints for the table according to the relationship that exists between the entities.

* If a 1:1(one to one)  relationship exists between entities then add the primary key entity of one table as foreign key to another table or vice versa.
* If total and partial participation exist between two entities in 1:1(one to one) relationship then put the primary key of the entity of partial participation as foreign key to the table with total participation.
* If 1:M (one to many) or M:1(many to one)relationships exist between entities then add the primary key of the table with one cardinality as foreign key to the table with many cardinalities.
* If M: N (many to many) relation exists between the entities, then create an intermediary table and add the primary key of both the tables as columns for the intermediary table. The combination of both keys makes the primary key which distinctly recognize each tuple of the table.

In this way, the ERD is converted into a relational schema. So, from the above steps it is clear that there may appear more tables than the entity presents in the ERD (Entity Relationship Diagram).

**Conversion of strong Entity into Relation schema from ERD shown above:**

Entity is converted into table and attributes as fields of the table.

* **Player entity into Player table.**

Player table:

| Player\_ID | Player name | Email | Online \_status | Global\_level | Diamonds\_collected | Username | Hero\_level | Skin\_owned |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |

**Primary key:** Player\_ID

* **Hero entity into Hero table.**

Hero table:

| Hero\_ID | Hero\_name | Hero\_role | Hero\_speciality | In\_diamonds | In\_battle\_points |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

**Primary key:** Hero\_ID

Similarly, Hero\_price attribute has composite attributes that is price In\_diamonds and price In\_battle\_points, so the composite attributes are illustrated into the relational schema instead of Hero\_price attribute according to the rules.

* **Skin entity into Skin table.**

Skin table:

| Skin\_ID | Skin\_name | In\_diamonds | In\_battle\_points | Hero\_ID\* |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

**Primary key:** Skin\_ID

**Foreign key:** Hero\_ID

As there is one to many relations between hero table and skin table we need to add the primary key attribute of hero table as foreign key attribute to skin table which inter connect two tables with each other. Similarly, Skin\_price attribute has composite attributes that is price In\_diamonds and price In\_battle\_points, so the composite attributes are illustrated into the relational schema instead of Skin\_price attribute according to the rules.

* **Game\_mode entity into Game\_mode table**

Game\_mode table:

| Game\_mode ID | Mode\_type |
| --- | --- |
|  |  |

**Primary key:** Game\_mode\_ID

* **Team entity into Team table.**

Team table:

| Team\_ID | Game\_mode ID\* | Game\_ID\* |
| --- | --- | --- |
|  |  |  |

**Primary key:** Team\_ID

**Foreign key:** Game\_mode\_ID, Game\_ID

As shown in above figure there is one to many relations between Game\_mode entity and team entity respectively. So, according to rule, we added a column called Game\_mode ID which is also a primary key of Game\_mode table to team table as foreign key. Similarly, there is one to many relations between game entity and team entity so Game\_ID also became foreign key for team table.

* **Game entity into Game table**

Game table:

| Game\_ID | Battle\_Duration | Winner | Loser | Game\_date | Game\_mode\_ID\* |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |

**Primary key:** Game\_ID

**Foreign key:** Game\_mode\_ID

* **Table formed due to many to many relations between entities:**
* **Player hero table formed due to many to many relations between player and hero entity.**

**Player\_hero table:**

| PH\_ID | Player\_ID\* | Hero\_ID\* |
| --- | --- | --- |
|  |  |  |

**Primary key:** PH\_ID (combination of Player\_ID and Hero\_ID)

**Foreign key:** Player\_ID, Hero\_ID

* **Mode\_chosen table is formed due to many to many relations between Player and Game mode entity.**

Mode\_chosen table**:**

| Player\_ID\* | Game\_mode\_ID\* |
| --- | --- |
|  |  |

**Primary key:** combination of both Player\_ID and Game\_mode\_ID

**Foreign key:** Player\_ID, Game\_mode\_ID

* **Conversion of weak entity into its corresponding relation:**

In ERD (Entity Relationship Diagram) there is a weak entity called Personal\_statistics table which doesn’t contain its primary key. So, it borrows that primary key of player table as its own primary key which uniquely identify each row of the table.

* Personal\_statistics table:

| Player\_ID\* | most\_used\_hero | previous\_game\_result | Additional\_details | Nationality |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

**Primary key:** Player\_ID

**Foreign Key:** Player\_ID

## Normalization of all relation up to 3NF:

1. UNF player relation:

| Player\_ID | Player\_name | Email | Username | Online\_status | Global\_level | Diamonds\_collected | Battle\_points | Hero\_ID | Hero\_level | Skin\_owned |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | Suraj | suraj1@gmail.com | Suraj123 | 1 | 12 | 223 | 2223 | H001,H002 | 12,23 | Sk001,Sk002 |

**1NF:**

So the 1NF of above UNF player relation is :

| Player\_ID | Player\_name | Email | Username | Online\_status | Global\_level | Diamonds\_collected | Battle\_points | Hero\_ID | Hero\_level | Skin\_owned |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | Suraj | suraj1@gmail.com | Suraj123 | 1 | 12 | 223 | 2223 | H001 | 12 | Sk001 |
| **1** | Suraj | [suraj1@gmail.com](mailto:suraj1@gmail.com) | Suraj123 | 1 | 12 | 223 | 2223 | H002 | 23 | Sk002 |

**2NF:**

**Partial dependency:**

Player\_ID-> Player\_name, Email, Username, Online\_status, Global\_level, Diamonds\_collected, Battle\_points

Player\_ID, Hero\_ID ->Hero\_level

Player\_ID, Hero\_ID->Skin\_ID

Table formed after 2NF are:

Player\_table:

| Player\_ID | Player\_name | Email | Username | Online\_status | Global\_level | Diamonds\_collected | Battle\_points |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | Suraj | [suraj1@gmail.com](mailto:suraj1@gmail.com) | Suraj123 | 1 | 12 | 223 | 2223 |

Player\_hero table:

| Player\_ID\* | Hero\_ID\* | Hero\_level |
| --- | --- | --- |
| **1** | H001 | 12 |
| **1** | H002 | 23 |

Player\_skin table:

| Player\_ID\* | Hero\_ID | Skin\_ID\* |
| --- | --- | --- |
| **1** | H001 | Sk001 |
| **1** | H002 | Sk002 |

Since, there is not any transitive dependency present in the player table. Hence,3NF player table is :

| Player\_ID | Player\_name | Email | Username | Online\_status | Diamonds\_collected | Battle\_points |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | Suraj | [suraj1@gmail.com](mailto:suraj1@gmail.com) | Suraj123 | 1 | 223 | 2223 |

2. Hero table:

| Hero\_ID | Hero\_name | Hero\_role | Hero\_speciality | In\_diamonds | In\_battle\_points |
| --- | --- | --- | --- | --- | --- |
| H001 | Rafaela | Fighter | Charge | 240 | 1500 |
| H002 | Fanny | Assassin | Reap | 350 | 2000 |

Hero\_ID ->Hero\_name, Hero\_role, Hero\_speciality, In\_diamonds, In\_battle\_points. {full dependency}.

Hence, there is not any partial and transitive dependency present. So, the table is already in 3NF.

3.Skin table:

| Skin\_ID | Skin\_name | Skin\_price | Hero\_ID\* |
| --- | --- | --- | --- |
| Sk001 | Recon | 200 | H001 |
| Sk002 | Glacier | 250 | H002 |

Skin\_ID -> Skin\_name, Skin\_price Hero\_ID\*. {full dependency}.

Hence, there is not any partial as well as transitive dependency. So, the table is already in 3NF.

4. Relation: Game\_mode

| Game\_mode\_ID | Mode\_type |
| --- | --- |
| GM001 | Brawl |
| GM002 | Human vs AI |

Game\_mode\_ID -> Mode\_type {full dependency}

Hence, there is not any partial as well as transitive dependency. So, the table is already in 3NF.

5. Relation: Team

| Team\_ID | Player\_ID\* | Game\_ID\* | Game\_mode\_ID\* |
| --- | --- | --- | --- |
| **1** | 2,4,7,3,5 | 1 | GM002 |

Since, there are repeating data in player\_ID column. Above table is in UNF (Un Normal Form).

**1NF:**

| Team\_ID | Player\_ID\* | Hero\_ID\* | Game\_ID\* | Game\_mode\_ID\* |
| --- | --- | --- | --- | --- |
| **1** | 2 | H002 | 1 | GM002 |
| **1** | 3 | H004 | 1 | GM002 |
| **1** | 4 | H003 | 1 | GM002 |
| **1** | 5 | H008 | 1 | GM002 |
| **1** | 7 | H001 | 1 | GM002 |

**Candidate key:** Team\_ID, Player\_ID

**Partial dependency:**

Team\_ID -> Game\_ID, Game\_mode\_ID

Player\_ID ->Hero\_ID

**2NF:**

To make above table into 2NF, table need to be spilt into two table as below:

**Team table**

| Team\_ID | Game\_ID\* | Game\_mode\_ID\* |
| --- | --- | --- |
| **1** | 1 | GM002 |

**Team\_player table:**

| Team\_ID | Player\_ID\* | Hero\_ID\* |
| --- | --- | --- |
| **1** | 2 | H002 |
| **1** | 3 | H004 |
| **1** | 4 | H003 |
| **1** | 5 | H008 |
| **1** | 7 | H001 |

Hence, there is no transitive dependency present in above tables the tables are already in 3NF .

**6. Relation: Game**

| Game\_ID | Battle\_duration | Winner\_team | Loser\_team | Game\_date | Arena | Game\_mode\_ID\* |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | 00:30:12 | 1 | 2 | 2020-02-01 | TDM | GM002 |

Game\_ID -> Battle\_duration, Winner\_team, Loser\_team, Game\_date, Game\_mode\_ID {**full dependency**}

Since, there is no any partial as well as transitive dependency presence in above relation, the relation is already in 3NF.

7. Relation: Player\_hero

| PH\_ID | Player\_ID\* | Hero\_ID\* |
| --- | --- | --- |
| PH001 | 1 | H002 |
| PH002 | 1 | H003 |

Since, there are no any non-key attributes present in the table, table is already in 3NF.

8. Mode\_chosen

| Player\_ID\* | Game\_mode\_ID\* |
| --- | --- |
| **1** | GM002 |
| **2** | GM003 |
| **1** | GM003 |

since there are no any non key attributes present in the table . Hence, the table is already in 3NF(Third Normal Form).

9. Relation: Personal\_statistics

| Player\_ID\* | most\_used\_hero | Previous\_game\_result | Additional\_result |
| --- | --- | --- | --- |
| **1** | H001 | Win | score=25 |
| **2** | H002 | Lose | score=0 |

Since, there is no any partial as well transitive dependency in above table , Personal\_statistics table is already in 3NF.

## Table formed after normalisation up to 3NF are:

1. Player table:

| Player\_ID | Player\_name | Email | Username | Online\_status | Global\_level | Diamonds\_collected | Battle\_points |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | Suraj | [suraj1@gmail.com](mailto:suraj1@gmail.com) | Suraj123 | 1 | 12 | 223 | 2223 |

1. Hero table:

| Hero\_ID | Hero\_name | Hero\_rolle | Hero\_speciality | In\_diamonds | In\_battle\_points |
| --- | --- | --- | --- | --- | --- |
| H001 | Rafaela | Fighter | Charge | 240 | 1500 |
| H002 | Fanny | Assassin | Reap | 350 | 2000 |

1. Skin table:

| Skin\_ID | Skin\_name | Skin\_price | Hero\_ID\* |
| --- | --- | --- | --- |
| Sk001 | Recon | 200 | H001 |
| Sk002 | Glaciar | 250 | H002 |

1. Game\_mode table:

| Game\_mode\_ID | Mode\_type |
| --- | --- |
| GM001 | Brawl |
| GM002 | Human vs AI |

1. Team table:

| Team\_ID | Game\_ID\* | Game\_mode\_ID\* |
| --- | --- | --- |
| **1** | 1 | GM002 |

1. Game table:

| Game\_ID | Battle\_duration | Winner\_team | Loser\_team | Game\_date | Arena | Game\_mode\_ID\* |
| --- | --- | --- | --- | --- | --- | --- |
| **1** | 00:04:12 | 1 | 2 | 2020-02-01 | TDM | GM002 |

1. Team\_player table:

| Team\_ID | Player\_ID\* | Hero\_ID\* |
| --- | --- | --- |
| **1** | 2 | H002 |
| **1** | 3 | H004 |
| **1** | 4 | H003 |
| **1** | 5 | H008 |
| **1** | 7 | H001 |

1. mode\_chosen table

| Player\_ID\* | Game\_mode\_ID\* |
| --- | --- |
| **1** | GM002 |
| **2** | GM003 |
| **1** | GM003 |

1. Player\_hero table

| PH\_ID | Player\_ID\* | Hero\_ID\* | Hero\_level |
| --- | --- | --- | --- |
| PH001 | 1 | H001 | 12 |
| PH002 | 1 | H002 | 23 |

1. hero\_skin table:

| Player\_ID\* | Hero\_ID | Skin\_ID\* |
| --- | --- | --- |
| **1** | H001 | Sk001 |
| **1** | H002 | Sk002 |

Since, the combination of both Player\_ID and Hero\_ID makes PH\_ID, we can replace Player\_ID and Hero\_ID by PH\_ID. So, the hero\_skin table changes to:

| PH\_ID | Skin\_ID\* |
| --- | --- |
| PH001 | Sk001 |
| PH002 | Sk002 |

1. personal\_statistics table:

| Player\_ID\* | most\_used\_hero | Previous\_game\_result | Additional\_result |
| --- | --- | --- | --- |
| **1** | H002 | Win | score=25 |
| **2** | H003 | Lose | score=0 |

## Data Dictionary:

**Player table**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Player\_ID | Unique id given to each player to uniquely identify the each row of the table. | Int | Not null,  Unique,  Auto increment | Primary key |
| Player\_name | Each player’s name | Varchar | Not null |  |
| Username | Player’s game name | Varchar | Not null,  Unique |  |
| Email | Player’s email address | Varchar | Not null,  Unique |  |
| Online\_status | Stores the information regarding player’s status like whether they are online or offline for example if a player is online then shows 1 else shows 0. | BIT |  |  |
| Global\_level | player’s level in game | Int | Not null |  |
| Diamonds\_collected | Holds the information about the total number of diamonds that player has currently. | Int |  |  |

**Hero table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Hero\_ID | Unique ID of each hero that uniquely identify each hero. | Varchar | Not null, unique | Primary key |
| Hero\_name | Name given to each hero like Rafaela, fanny, Estes, etc. | Varchar | Not null, unique |  |
| Hero\_role | Role given to the hero like fighter, tank, assassin, etc. | Varchar | Not null |  |
| Hero\_speciality | Special power that hero own. | Varchar | Not null |  |
| In\_diamonds | Price of hero in diamonds. | Int |  |  |
| In\_battle\_points | Price of hero in battle points | Int |  |  |

**skin table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Skin\_ID | ID of each skin that uniquely distinguish each skin name. | Varchar | Not null, unique | Primary key |
| Skin name | Name given to the skin. | Varchar | Not null |  |
| Skin\_price | Price of each skin. | Int | Not null |  |
| Hero\_ID\* | ID of hero that own skin. | varchar | Not null | Foreign key |

**Player\_hero table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| PH\_ID | Unique ID that identify each tuple distinctly, which is the combination of Player\_ID and Hero\_ID. | Varchar | Not null, unique | Primary key |
| Player\_ID\* | ID of the player that owned hero. | Int | Not null | Foreign key |
| Hero\_ID\* | ID of the hero owned by the player. | Varchar | Not null | Foreign key |
| Hero\_level | Each player’s hero level. | Int | Not null |  |

**Hero\_skin table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| PH\_ID\* | ID that represent the specific hero of the specific player . | Varchar | Not null | Primary key,  Foreign key |
| Skin\_ID\* | ID of the skin that player’s hero owned. | Varchar | Not null | Primary key,  Foreign key |

**Game\_mode table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Game\_mode ID | Unique ID that identify each game\_mode uniquely. | Varchar | Not null, unique | Primary key |
| Mode\_type | Types of mode available in the game. | Varchar | Not null |  |

**Team table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Team\_ID | Unique ID of each team played in each game which is automatically created. | Int | Not null, unique | Primary key |
| game\_mode ID\* | The ID of the mode in which the team has played the game. | Varchar | Not null | Foreign key |
| Game\_ID\* | The ID of the game in which team has played. | Int | Int | Foreign key |

**Game\_table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Game\_ID | Unique ID formed automatically when game is played. | Int | Not null, unique | Primary key |
| Battle\_duration | Total time that the game has run. Holds the time in seconds. | Int | Not null |  |
| Winner\_team | The game winning team. | Int | Not null | Foreign key, |
| **Loser\_team** | The team that loses team. | Int | Not null | Foreign key |
| Game\_date | The date in which the game was started. | Int | Not null |  |
| Arena | The scene in which the game is played. | varchar | Not null |  |
| Game\_mode\_ID\* | The unique Id of the mode in which the game is being played. | Varchar | Not null | Foreign key |

**Team\_player table:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Team\_ID\* | ID of the team . | Int | Not null | Foreign key, primary key |
| Player\_ID\* | ID of the each player participation in the team. | Int | Not null | Foreign key, primary key |
| Hero\_ID\* | ID of the hero that team’s player had chosen. | Varchar | Not null | Foreign key, primary key |

**Mode\_chosen:**

| Attributes | Description | Datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Player\_ID\* | ID of the player that chooses Game\_mode. | Int | Not null | Foreign Key |
| Game\_mode\_ID\* | ID of the game mode choose buy the player. | Varchar | Not null | Foreign Key |

**Personal statistics:**

| Attributes | Description | datatype | Constraint | Key |
| --- | --- | --- | --- | --- |
| Player\_ID\* | ID of the player | Int | Not null | Primary key, foreign key |
| Most\_used\_hero | Holds the record about the hero that has been used most of the time in the game by player. | Varchar | Not null |  |
| Previous\_game\_result | Result of previous games whether the game is won or lose. | Varchar | Not null |  |
| Additional\_details | About player’s profile details. | Varchar | Not null |  |

## Database Diagram:

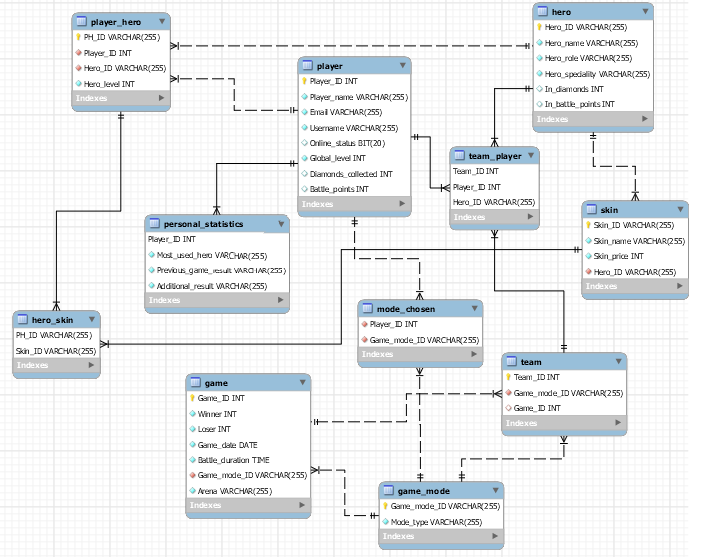
****

Figure: Database Diagram.

# SQL-Data Definition Language SQL-DDL

There are various syntaxes for the creating of database, creating of tables, insertion of data in the table, update data, delete data. All syntaxes are described below one by one.

### Create database:

In SQL, firstly we need to create a database. Without creating database, we cannot create tables. The syntax that is used for the creating of database in MySQL server is:

CREATE DATABASE database\_name;

**DATABASE bing\_bing:**



**Create tables:**

Same like database we need to create table where we can insert data. Tables is creating inside the database which you have previously created. So, to create a table you need to first execute the ‘use <database\_name in which you want to create a table>' command. Then the you need to create a table using following syntax:

CREATE TABLE table\_name(

Column\_1 datatypes,

Column\_2 datatypes,

column\_3 datatypes,

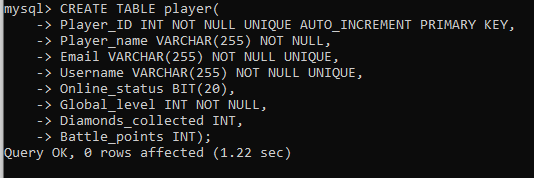
….

PRIMARY KEY(Column\_name),

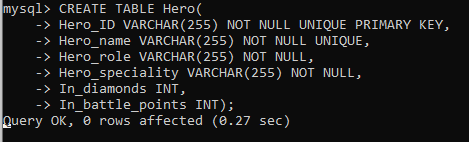
FOREIGN KEY(Column\_name) REFERENCES table\_name(Column\_name)

);

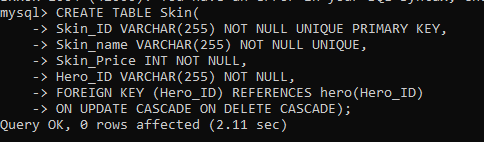
1. **Player\_table:**



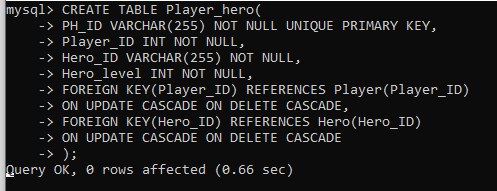
1. **Hero\_table:**



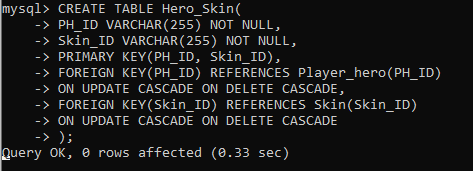
1. **Skin\_table**:



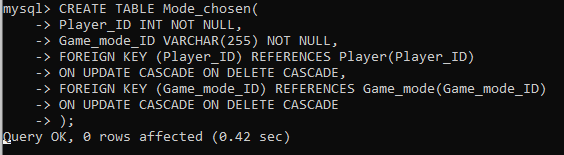
1. **Player\_hero table:**



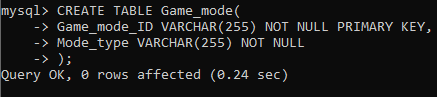
1. **Hero\_skin table:**



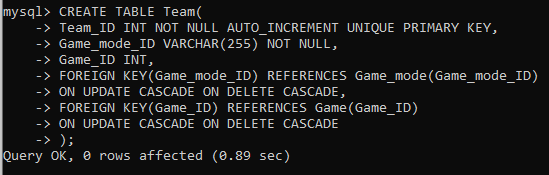
1. **Mode\_chosen table:**

****

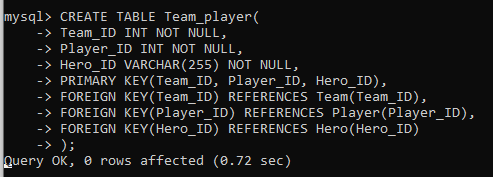
1. **Game\_mode table:**



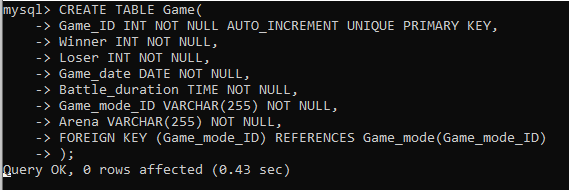
1. **Team table:**



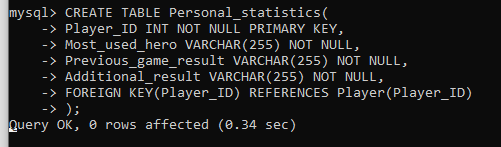
1. **Team\_player table:**



1. **Game table:**



1. **Personal\_statistics table:q**



Similarly, to update and delete data in child table automatically when the data are updated and deleted in parent table we need to add **ON UPDATE CASCADE** and **ON DELETE CASCADE** in the create\_table syntax, respectively.

**AUTO\_INCREMETN:** This command is used in order to generate unique number automatically when new data are inserted into the table.

### Insert data in table:

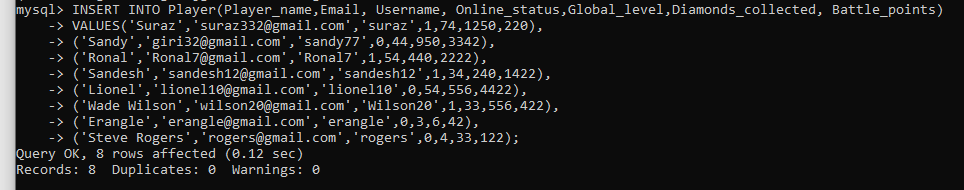
The syntax in order to insert the data in the table is:

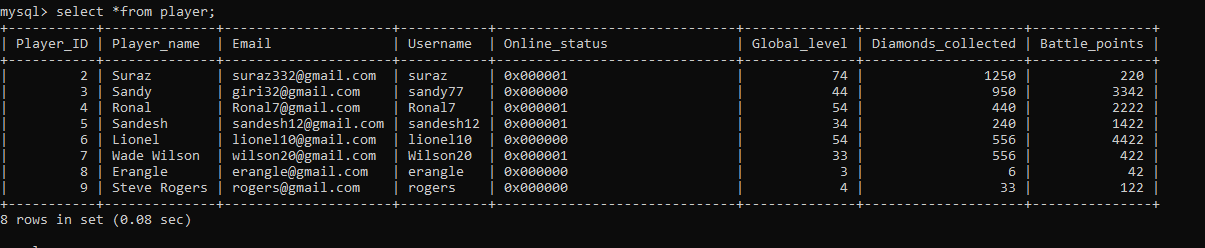
INSERT INTO table\_name(Column\_1, Column\_2, Column\_3,…)

VALUES (Value\_1, value\_2, value\_3, ….);

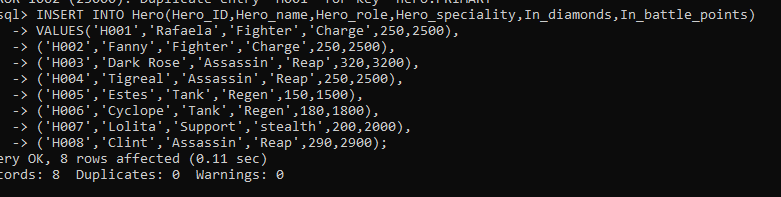
To insert the varchar or string values, we need to put the values inside inverted commas.

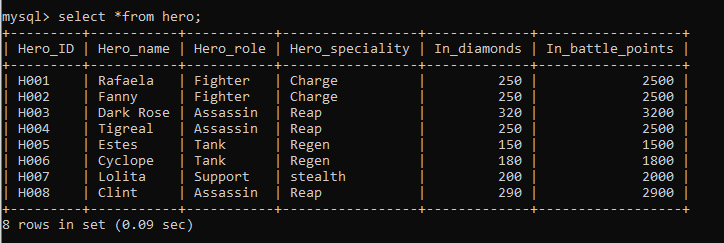
1. **Player\_table:**

****

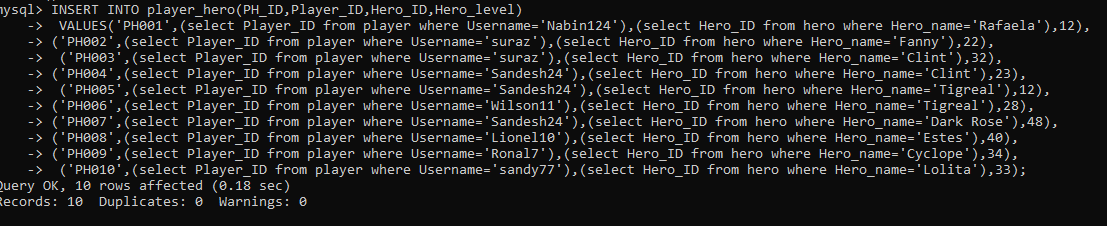
****

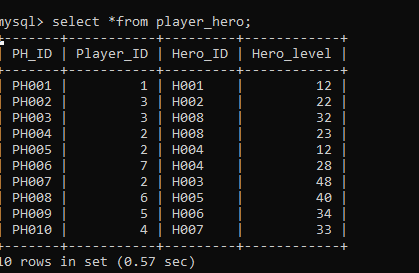
1. **Hero\_table:**

****

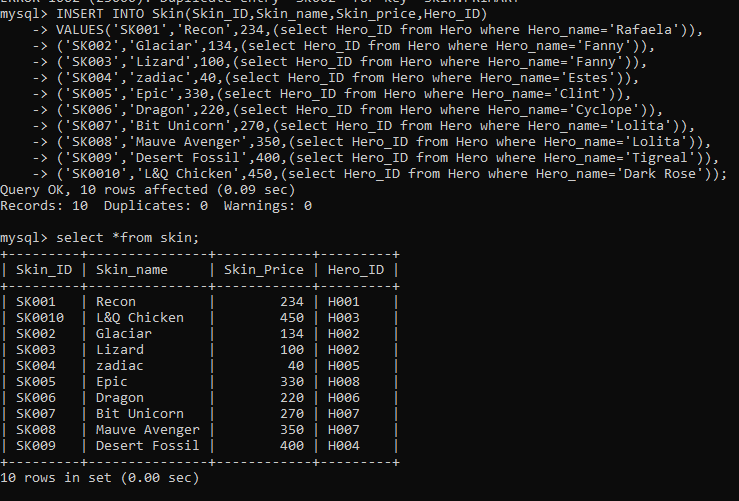
****

1. **Player\_hero table:**

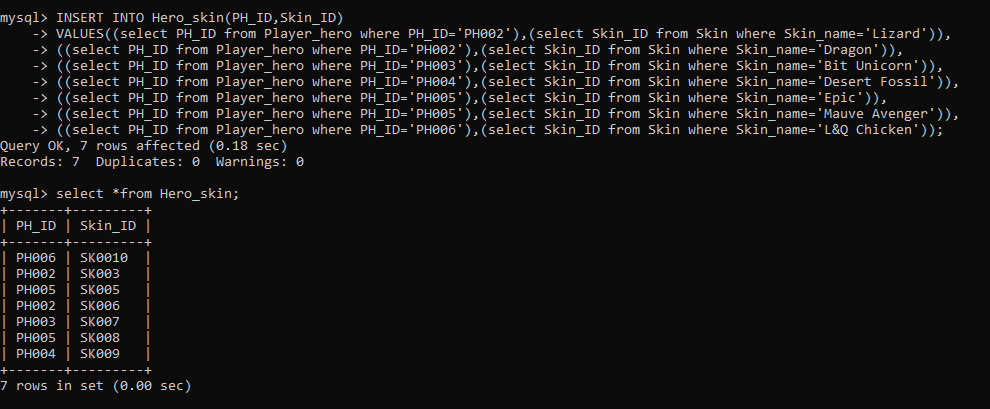
****

****

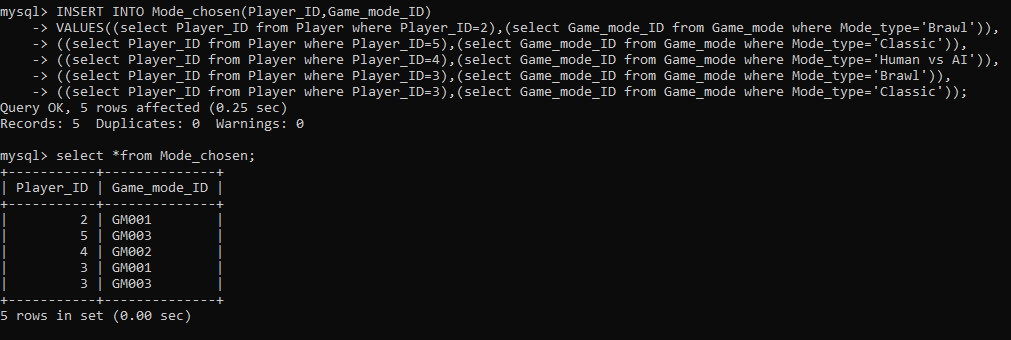
1. **Skin\_table:**

****

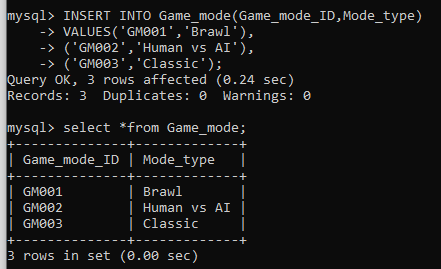
1. **Hero\_skin table:**

****

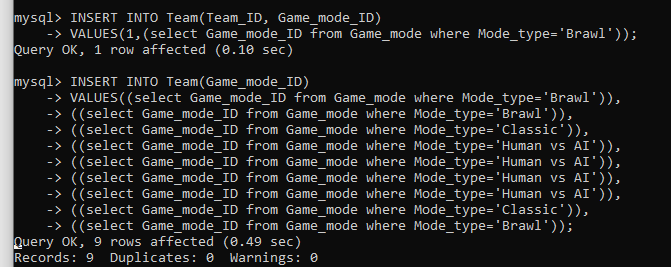
1. **Mode\_chosen table:**

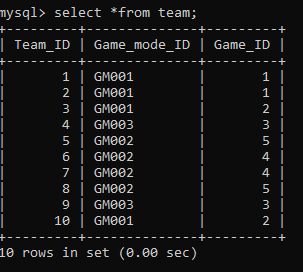
****

1. **Game\_mode table:**

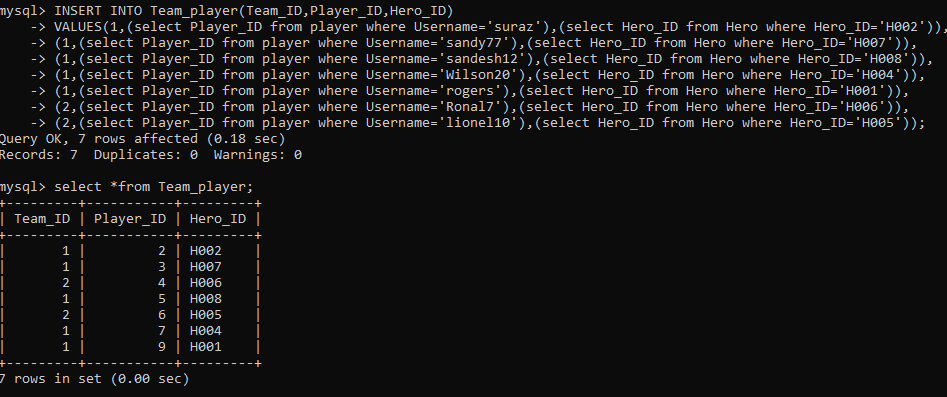
****

1. **Team\_table:**

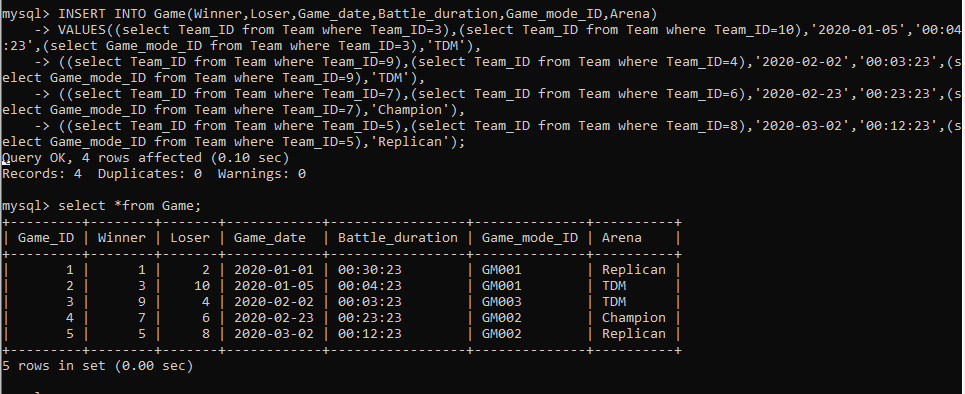
****

****

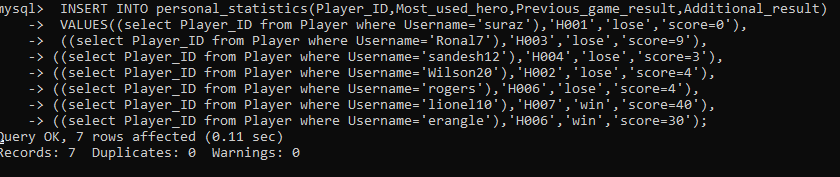
1. **Team\_player table:**

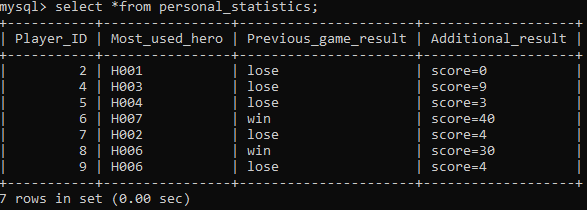
****

1. **Game table:**

****

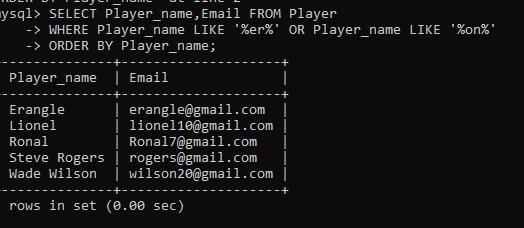
1. **Personal\_statistics table:**

****

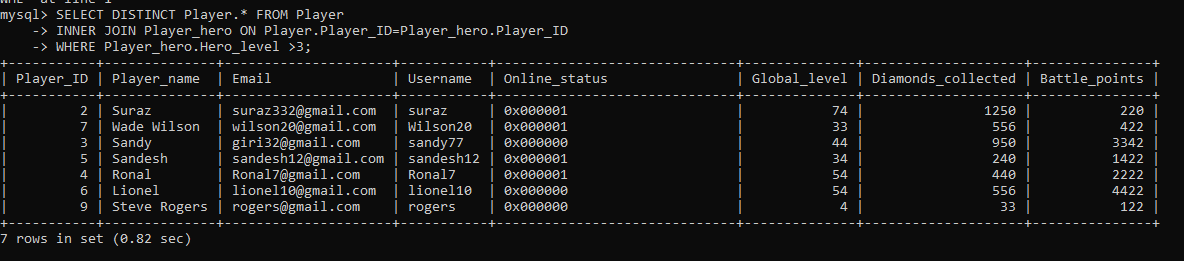
****

# 2. SQL-Data Manipulation Language (DML):

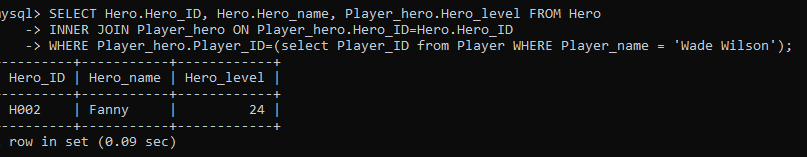
## Question **1:**

****

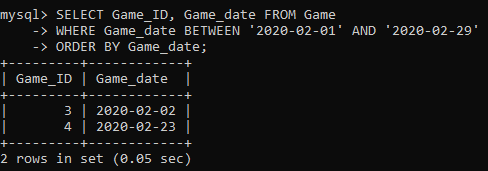
## Question 2:

****

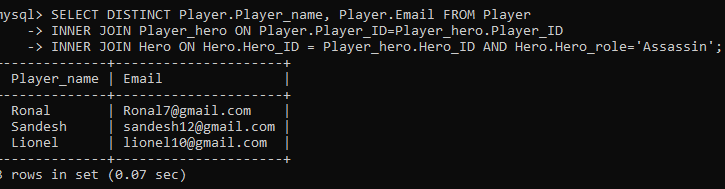
## Question 3:

****

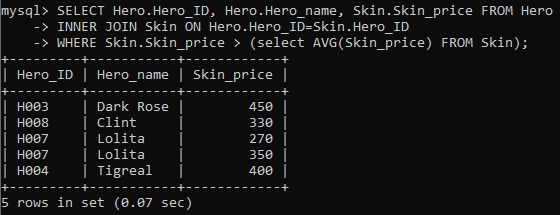
## Question 4:

****

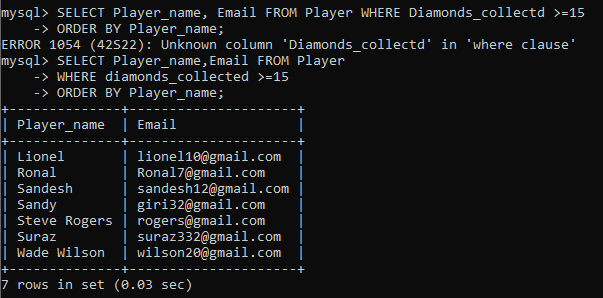
## Question 5:

****

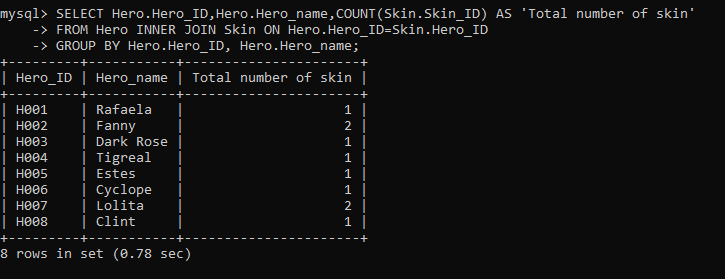
## Question 6:

****

## Question 7:

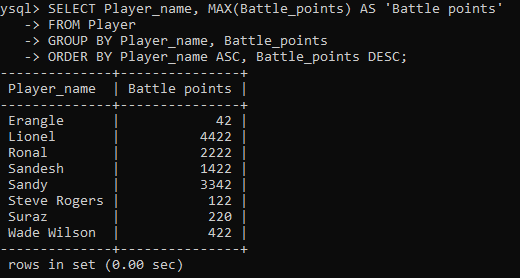
****

## Question 8:

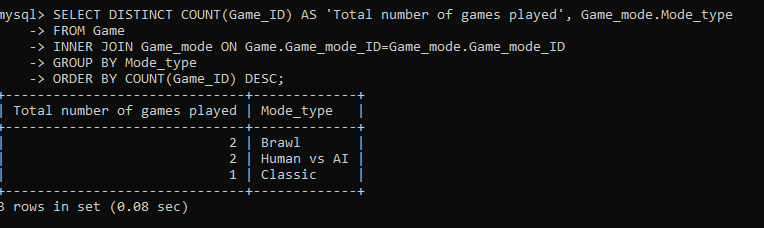
****

## Question 9:

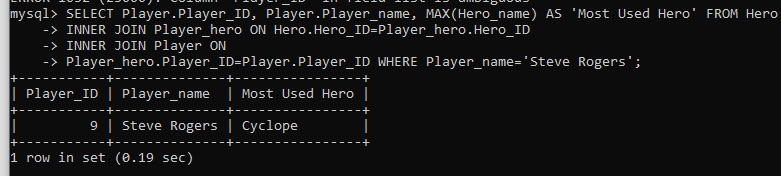
## Question 10:



## Question 11:

****

## Question 12:

****

# Workload Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ASSIGNEMNT TOPICS | SURAJ PANDEY | SANDESH SUBEDI | NABIN CHHETRI | SANDESHGIRI |
| ERD to Relational schema | 100% | - | - | - |
| Normalization of Relations | - | 100% | - | - |
| Database diagram | - | - | - | 100% |
| Data dictionary | - | - | 100% | - |
| Create tables | 25% | 25% | 25% | 25% |
| Data insertion | 25% | 25% | 25% | 25% |
| Queries | 25% | 25% | 25% | 25% |
| Signature | Suraj | Sandesh | Nabin | Sandesh |

# References

Anon, (n.d.). *Advanced Data Modeling | Relational DBMS with Network Model beneath*. [online] Available at: <https://raima.com/advanced-data-modeling/>. [Accessed 10 February 2021].

app.diagrams.net. (n.d.). *Flowchart Maker & Online Diagram Software*. [online] Available at: <https://app.diagrams.net>. [Accessed 10 February 2021].

beginnersbook.com. (2021). Advantages of DBMS over file system. [ONLINE] Available at: https://beginnersbook.com/2015/04/dbms-vs-file-system/. [Accessed 12 February 2021]

Castro, K. (2018). *Advantages of Database Management System*. [online] Tutorialspoint.com. [ONLINE] Available at: <https://www.tutorialspoint.com/Advantages-of-Database-Management-System>. [Accessed 13 February 2021].

Guru99.com. (2021). *What is DBMS? Application, Types, Example, Advantages*. [online] Available at: <https://www.guru99.com/what-is-dbms.html> [Accessed 14 February 2021].

patil, k., (2021). *ER Diagram to Relational Model Conversion*. [online] DEV Community. Available at: <https://dev.to/ketan\_patil/er-diagram-to-relational-model-conversion-49ip> [Accessed 10 February 2021].

Thiru. 2021. What is DBMS? Advantages and Disadvantages of Database Management System(DBMS). [ONLINE] Available at: http://www.myreadingroom.co.in/notes-and-studymaterial/65-dbms/462-advantages-and-disadvantages-of-dbms.html. [Accessed 14 February 2021].

www.guru99.com. (n.d.). *What is Normalization? 1NF, 2NF, 3NF, BCNF Database Example*. [online] Available at: https://www.guru99.com/database-normalization.html?fbclid=IwAR0iiwdeXOkg98