Data and Applications

Project Phase - 3:

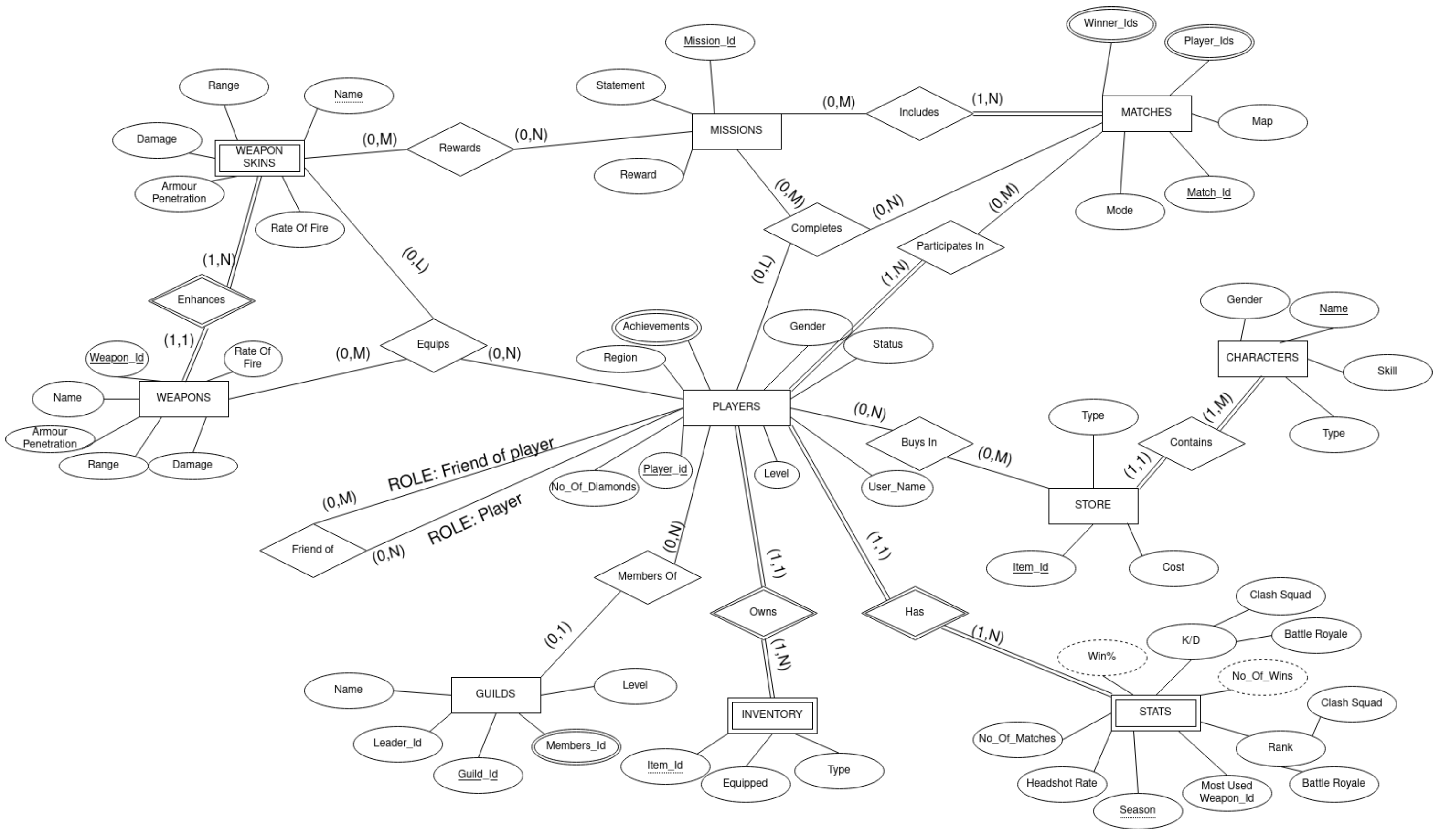
Data Architects (Team-37) :

- Sai Sudhan (2022101052)
- Krishna Koushik (2022101036)
- Raghava Nijesh (2022101056)
- Chaganti Venkata Karthikeya (2022101058)
- Vanarasi Vamseedhar (2022101073)

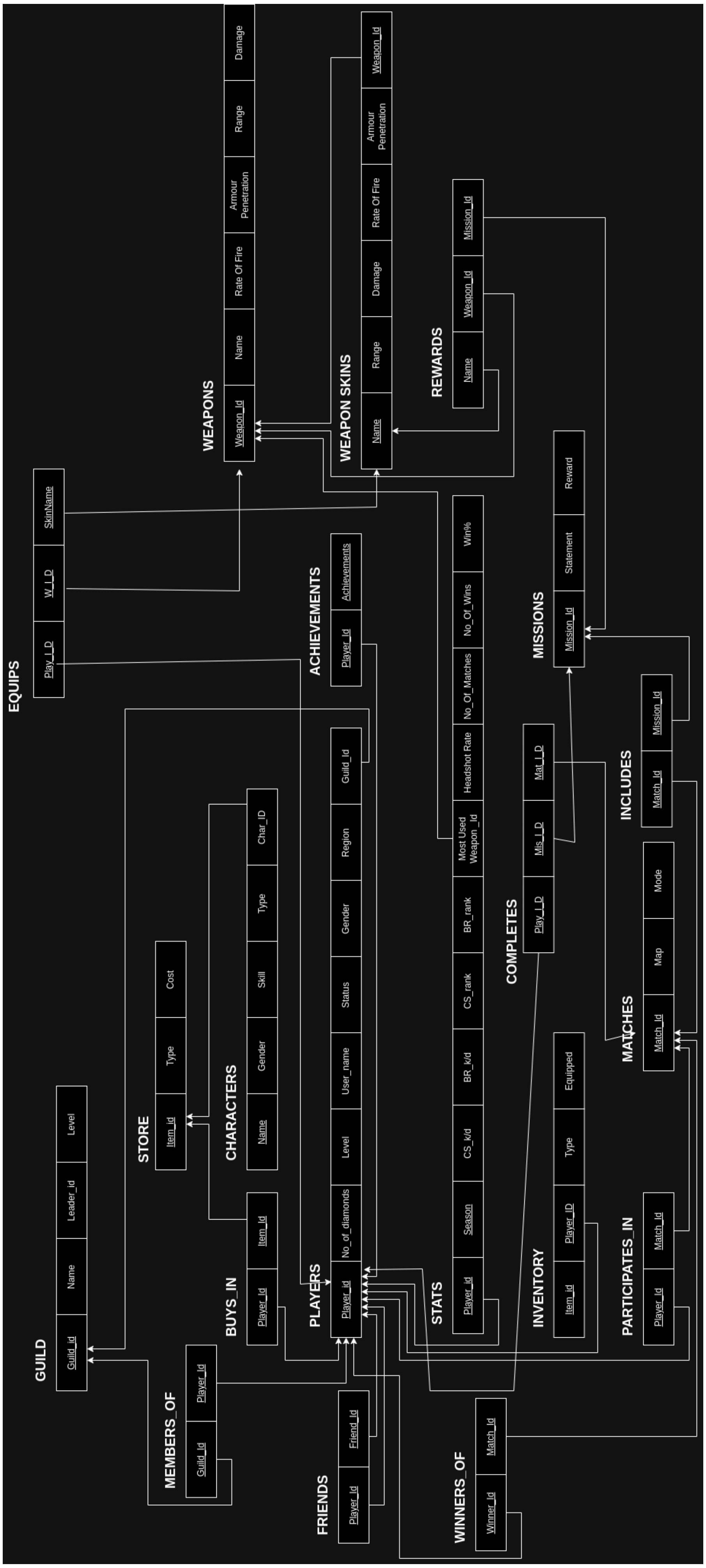
ER model to Relational Model :

Below steps describe the steps of an algorithm for ER-to-relational mapping. The FreeFire ER Schema is shown below, and the corresponding relational database schema followed by it to illustrate the mapping steps. Our mapping will create tables which includes simple single-valued attributes primary keys and referential integrity constraints on the relations which are specified in the mapping results.

Entity Relationship schema



Relational Database Schema



Step 1: Mapping of Regular Entity Types

For each regular (strong) entity type *E* in the ER schema, create a relation *R* that includes all the simple attributes of *E*. Include only the simple component attributes of a composite attribute. Choose one of the key attributes of *E* as the primary key for *R*. If the chosen key of *E* is a composite, then the set of simple attributes that form it will together form the primary key of *R*.

In Our Case,

- The composite attribute K/D is converted to CS_K/D and BR_K/D.
- The composite attribute Rank is converted to CS_rank and BR_rank.

Step 2: Mapping of Weak Entity Types

For each weak entity type W in the ER schema with owner entity type E, we created a relation R and included all its simple attributes (or simple components of composite attributes) of W as attributes of R. In addition, we included the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s) as foreign key attributes of R (this takes care of mapping the identifying relationship type of W).

In Our Case,

- The relation “Inventory” was created with Item_id as a partial key and the identifying relationship OWNS is included as OwnerP_ID which is a foreign key referencing Player_id of “Players”.
- The relation “Weapon Skins” was created with Name as a partial key and identifying relationship ENHANCES is included as W_ID which is a foreign key referencing Weapon_id of “Weapons”.
- The relation “Stats” was created with Season as a partial key and identifying relationship HAS included as P_ID which is a foreign key referencing Player_id of “Players”.

Step 3: Mapping of Binary 1:1 Relationship Types

No Binary 1:1 Relationship Types exists in our schema.

Step 4: Mapping of Binary 1:N Relationship Types:

We have employed the foreign key approach that is, For each regular binary 1:N relationship type *R*, we identified the relation *S* that represents the participating entity type at the *N-side* of the relationship type and included the primary key of the relation T that represents the other entity type participating in R as foreign key in S.

In Our Case,

- The “CONTAINS” relationship is mapped to the foreign key attribute Char_ID of CHARACTERS, which references the primary key Item_id of the STORE relation.
- The MEMBERS_OF relationship is mapped to the foreign key attribute Guild_ID of PLAYER, which references the primary key Guild_id of the “GUILD” relation.

Step 5: Mapping of Binary M:N Relationship Types

We have employed the Relationship relation / cross-reference approach. For each binary M:N relationship type R, we created a new relation S to represent R. We included the primary keys of the relations that represent the participating entity types as foreign key attributes in S; their combination will form the primary key of S. Also, we included any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S.

In Our Case,

- The relation “PARTICIPATES_IN” was created with Pla_ID as a foreign key referencing Player_id of “PLAYER” and Mat_ID as a foreign key referencing Match_id of “MATCHES”.
- The relation “INCLUDES” was created with Mis_ID as a foreign key referencing Mission_id of “MISSIONS” and Matc_ID as a foreign key referencing Match_id of “MATCHES”.
- The relation “FRIEND_OF” was created with PlayerID and FriendID as foreign keys referencing Player_id of “PLAYER”.
- The relation “BUYS_IN” was created with Player_I_D as a foreign key referencing Player_id of “PLAYER” and Item_I_D as a foreign key referencing Item_id of “STORE”.
- The relation “REWARDS” was created with Mission_I_D as a foreign key referencing Mission_id of “MISSIONS” ,Weapon_I_D as a foreign key referencing Weapon_id of “WEAPONS” and Skin_name as a foreign key referencing Name of “WEAPON_SKINS”.

Step 6: Mapping of Multivalued Attributes

For each multivalued attribute A, we created a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K as a foreign key in R of the relation that represents the entity type or relationship type that has A as a multivalued attribute. The primary key of R is the combination of A and K. If the multivalued attribute is composite, we included its simple components.

In Our Case,

- The relation “ACHIEVEMENTS” was created with Play_ID as a foreign key referencing Player_id of “PLAYER” and Misson_ID as a foreign key referencing Mission_id of “Missions”.
- The relation “WINNERS” was created with Winner_ID as a foreign key referencing Player_id of “PLAYER” and Match_I_D as a foreign key referencing Match_id of “MATCHES”.
- The relation for Multivalued attribute “Player_ids” of MATCHES was not created because we have already created a relation PARTICIPATES_IN in step 5 which already has same attributes player_ids and match_ids.
- The relation for Multivalued attribute “member_ids” of Guild was not created because we have already included attribute guild_id in PLAYER relation in step 4.

Step 7: Mapping of N-ary Relationship Types

We have employed the **Relationship relation / cross-reference approach.** For each n-ary relationship type R, where n > 2, we created a new relationship relation S to represent R. We included the primary keys of the relations that represent the participating entity types as foreign key attributes in S. Also, we included any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.

In Our Case,

- The relation “Equips” was created with attributes Play_I_D as a foreign key referencing Player_id of “PLAYER”, W_I_D as a foreign key referencing Weapon_id of “WEAPONS” and SkinName as a foreign key referencing Name of “WEAPON_SKINS”.

- The relation “Completes” was created with attributes Pla_I_D as a foreign key referencing Player_id of “PLAYER”, Mis_I_D as a foreign key referencing Missions_id of “MISSIONS” and Mat_I_D as a foreign key referencing Name of “MATCHES”.

Step 8: Options for Mapping Specialization or Generalization

No Subclasses exist in our Schema

Step 9: Mapping of Union Types (Categories):

No Union type exists in our Schema

Normalisation :

Conversion of Relational Model to 1NF:

Relation schema is in 1NF if the values in domain of each attribute are atomic. The relational model is already in 1NF as new relations for Multivalued attributes were created in Step 6 and Composite attributes were converted to Atomic (simple) attributes in Step 1.

Conversion of Relational Model to 2NF:

A relation schema is in 2NF if every non-prime attribute A in R is fully functionally dependent on every key of R.

In Our Case,

- The Relation “INVENTORY”{ item_id, Player_ID, type, Equipped} was broken down into :
  - INVENTORY { item_id, Player_ID, Equipped }
  - INVENTORY_ITEMS { item_id, Type }
- Reason : Type attribute is dependent only on item id but not on player_id i.e, partial dependency, on decomposing we can get full dependency.
- The Relation “WEAPON_SKIN”{ Weapon_id, Name, Range, Damage, Rate of fire, Armour Penetration} was broken down into :
  - WSKIN INFO { Name, Range, Damage, Rate of fire, Armour Penetration }
  - WEAPON_SKINS { Weapon_id, Name }
- Reason : { Range, Damage, Rate of fire, Armour Penetration } attributes is dependent only on Name but not on Weapon_id i.e, partial dependency, on decomposing we can get full dependency.

Conversion of Relational Model to 3NF:

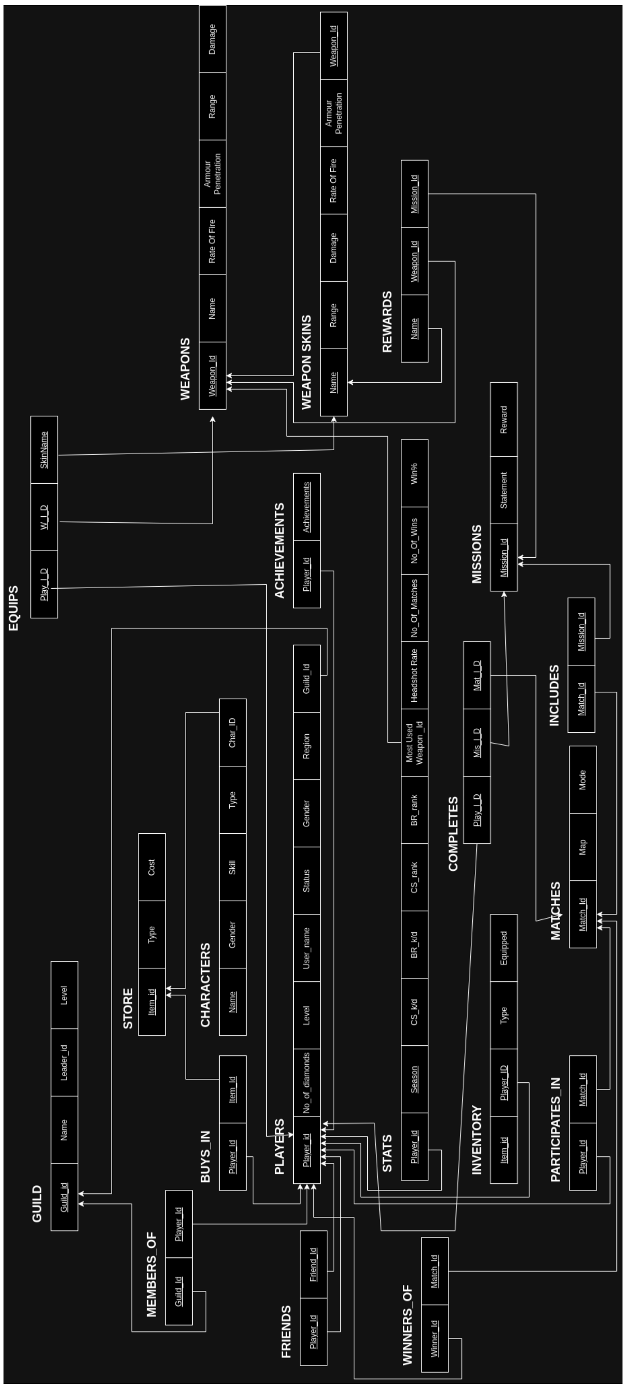
A relation schema is in 3NF if all non-trivial dependencies in F+ are of the form X->A with either

- X is a superkey
- A is a prime attribute

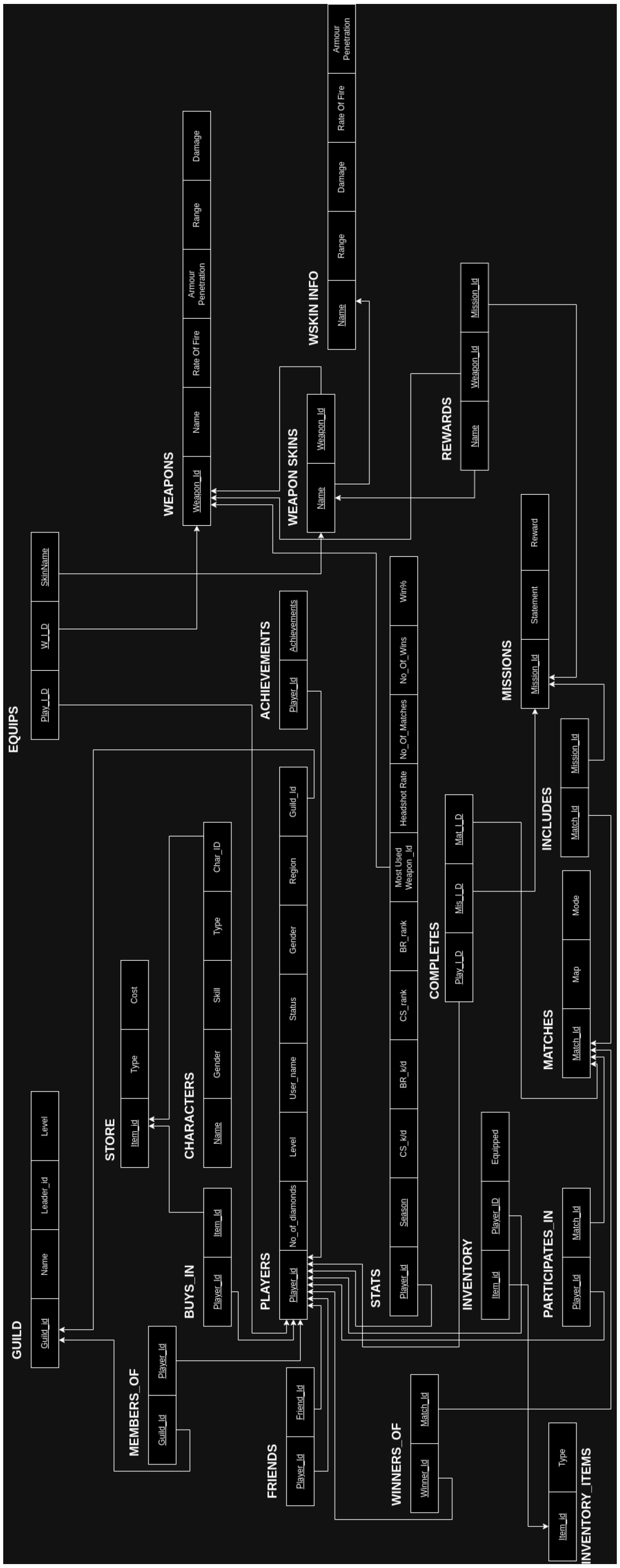
In Our Case,

- The Relation “STATS” { Player_id, Season, CS_k/d, BR_k/d, CS_rank, BR_rank, Most Used Weapon_id, Headshot Rate, No_of_Matches, No_of_Wins, Win% } was broken down into :
    - WIN% { No_of_Matches, No_of_Wins, Win% }
    - STATS { Player_id, Season, CS_k/d, BR_k/d, CS_rank, BR_rank, Most Used Weapon_id, Headshot Rate, No_of_Matches, No_of_Wins }
- Reason :
    - { Player_id, Season } -> { No_of_Matches, No_of_Wins }
    - { No_of_Matches, No_of_Wins } -> { Win% }
    - By Transitive Dependency { Player_id, Season } -> { Win% } i.e, Win% depends on Player_id and Season on decomposing we can remove transitive dependency.

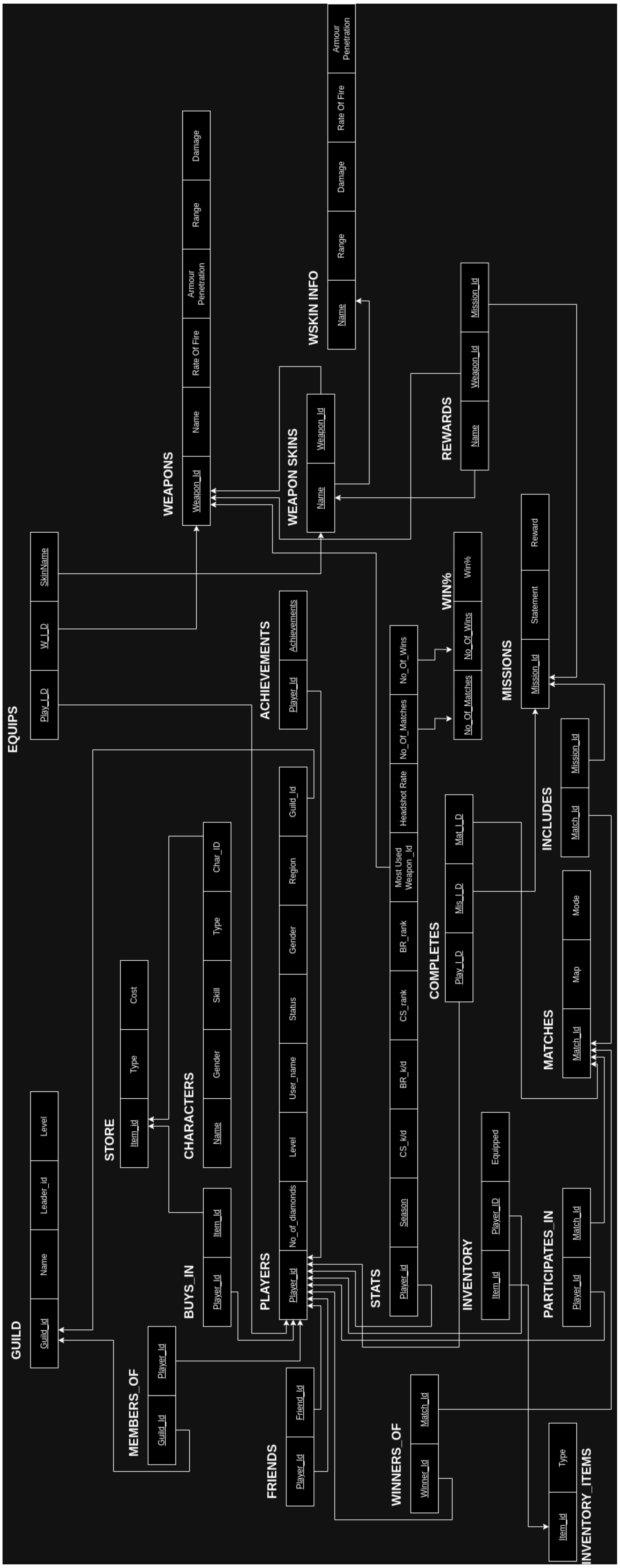
1-NF :



2-NF :


GUILD
Guild_id
Name
Leader_id
Level
EQUIPS
Play_I_D
W_I_D
SkinName
WEAPONS
Weapon_Id
Name
Rate Of Fire
Armour Penetration
Range
Damage
WEAPON SKINS
Name
Weapon_Id
WSKIN INFO
Name
Range
Damage
Rate Of Fire
Armour Penetration
REWARDS
Name
Weapon_Id
Mission_Id
MISSIONS
Mission_Id
Statement
Reward
ACHIEVEMENTS
Player_Id
Achievements
STORE
Item_id
Type
Cost
CHARACTERS
Name
Gender
Skill
Type
Char_ID
BUYS_IN
Player_Id
Item_Id
PLAYERS
Player_id
No_of_diamonds
Level
User_name
Status
Gender
Region
Guild_Id
MEMBERS_OF
Guild_Id
Player_Id
STATS
Player_id
Season
CS_k/d
BR_k/d
CS_rank
BR_rank
Most Used Weapon _Id
Headshot Rate
No_Of_Matches
No_Of_Wins
Win%
COMPLETES
Play_I_D
Mis_I_D
Mat_I_D
INCLUDES
Match_Id
Mission_Id
MATCHES
Match_Id
Map
Mode
INVENTORY
Item_id
Player_ID
Equipped
PARTICIPATES_IN
Player_Id
Match_Id
FRIENDS
Player_Id
Friend_Id
WINNERS_OF
Winner_Id
Match_Id
INVENTORY_ITEMS
Item_id
Type

3 - NF :


WEAPONS
Weapon_Id
Name
Rate Of Fire
Armour Penetration
Range
Damage
WSKIN INFO
Name
Range
Damage
Rate Of Fire
Armour Penetration
WEAPON SKINS
Name
Weapon_Id
REWARDS
Name
Weapon_Id
Mission_Id
EQUIPS
Play_I_D
W_I_D
SkinName
ACHIEVEMENTS
Player_Id
Achievements
WIN%
No_Of_Matches
No_Of_Wins
Win%
MISSIONS
Mission_Id
Statement
Reward
INCLUDES
Match_Id
Mission_Id
COMPLETES
Play_I_D
Mis_I_D
Mat_I_D
STATS
Player_Id
Season
CS_k/d
BR_k/d
CS_rank
BR_rank
Most Used Weapon_Id
Headshot Rate
No_Of_Matches
No_Of_Wins
MATCHES
Match_Id
Map
Mode
PLAYERS
Player_id
No_of_diamonds
Level
User_name
Status
Gender
Region
Guild_Id
CHARACTERS
Name
Gender
Skill
Type
Char_ID
STORE
Item_id
Type
Cost
BUYS_IN
Player_Id
Item_Id
GUILD
Guild_id
Name
Leader_id
Level
MEMBERS_OF
Guild_Id
Player_Id
FRIENDS
Player_Id
Friend_Id
WINNERS_OF
Winner_Id
Match_Id
INVENTORY
Item_id
Player_ID
Equipped
PARTICIPATES_IN
Player_Id
Match_Id
INVENTORY_ITEMS
Item_id
Type