University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: M2

Date: 10/20/23

Group Number: 81

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Bob Pham	44606424	z4j2v	bobpham@student.ubc.ca
Jason Wang	52783859	x5y6z	jason.wang014@gmail.com
Stevan Zhuang	57167090	m4y2u	stevan.zhuang@gmail.com

By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

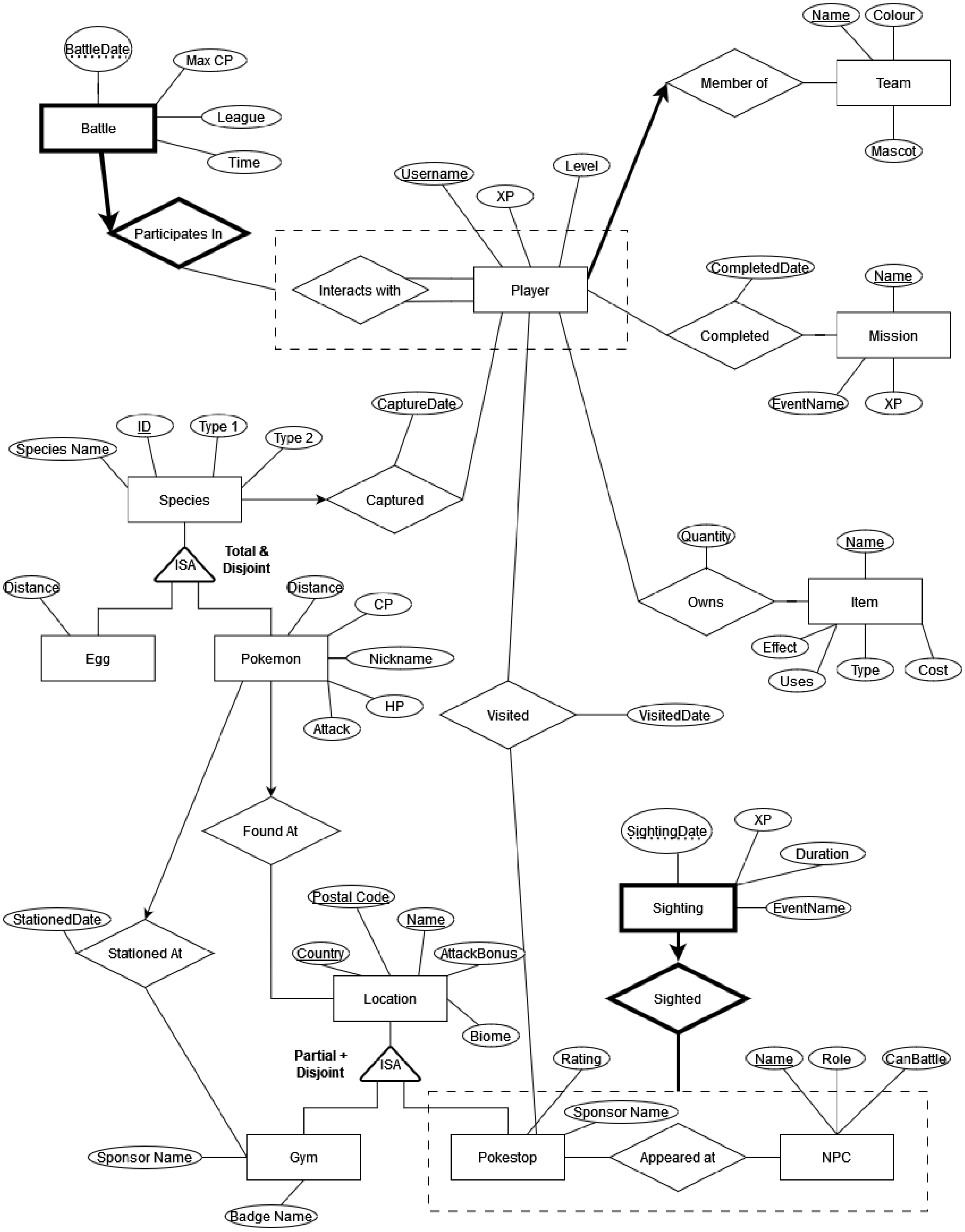
In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

Project Summary:

Our project is a Pokemon Go journal that aims to allow players to keep track of their activities and interactions with both in-game and outside of game elements. Our database models different in-game aspects that players may want to keep track of, in addition to actions that players may do in-game. As such, players can keep track of what they have accomplished within the game, as well as compare with other players.

ER Diagram:

See next page. Notes regarding any changes based on feedback from M1 is noted on the page after.


BattleDate
Max CP
Battle
League
Time
Participates In
Name
Colour
Member of
Team
Mascot
Username
XP
Level
Interacts with
Player
CompletedDate
Name
Completed
Mission
EventName
XP
CaptureDate
ID
Type 1
Type 2
Species Name
Species
Captured
Total &
Disjoint
ISA
Distance
Distance
CP
Egg
Pokemon
Nickname
HP
Attack
Quantity
Name
Owns
Item
Effect
Uses
Type
Cost
Visited
VisitedDate
Found At
SightingDate
XP
Duration
Sighting
EventName
StationedDate
Stationed At
Postal Code
Name
Country
AttackBonus
Location
Biome
Sighted
Partial +
Disjoint
ISA
Rating
Name
Role
CanBattle
Sponsor Name
Sponsor Name
Gym
Pokestop
Appeared at
NPC
Badge Name

Changes From Feedback:

- Corrected arrow in “Found at” and “Participates in” relationships, was incorrect based on the class diagram conventions and notations
- Updated “Battle” weak entity to include additional attributes outside of primary key
    - “Max CP” that is an integer that denotes the max CP a pokemon can have that is a player can use in the battle
    - “League” the name of the league that the battle is a part of
    - “Time” How long the battle lasted
- Updated “Team” entity to include additional attributes outside of primary key
    - “Colour” denoting the primary color that represents the team
    - “Mascot” the name of the mascot that represents the team
- Updated “Mission” entity to include additional attributes outside of primary key
    - “XP” the amount of XP that the mission rewards
    - “EventName” the name of the event that this mission is a part of
- Updated “Item” entity to include additional attributes outside of primary key
    - “Effect” denotes the effect of an item
    - “Type” what type of item it is
    - “Cost” the cost of the item in the in-game shop
    - “Use” the number of times the item can be used
- Updated “Sighting” weak entity to include additional attributes outside of primary key
    - “XP” the amount of XP gained from participating in the sighting
    - “EventName” the name of the event this sighting was a part of
    - “Duration” how long the npc was sighted at the location for
- Updated “NPC” entity to include additional attributes outside of primary key
    - “Role” the role of the NPC in the game, (ex. professor, opponent, etc)
    - “Can Battle”, whether or not this entity is able to battle with the player

Additional Notes:

- We decided that “EventName” should not be an entity because the goal of our application is to be a diary/journal/log for players, where they insert values themselves, so we give players freedom to call events what they want and/or create their own custom events.

Other Changes:

- Updated “Pokemon” entity to include additional attributes
    - “CP” current combat power score
    - “HP” current health points
    - “Attack” Current attack score
    - “Distance” Distance walked with as buddy
        - Different from egg distance since it can be null
- Removed “Level” Attribute from “Pokemon” Entity
    - Overlaps with CP, CP is the better term to use
- Updated “Location” entity to include additional attributes, and primary key
    - “Name” updated to also be part of the primary key
    - “Biome” the biome that the location is in (ex. desert, water)
    - “AttackBonus” the type that gets an attack bonus from being in the area

- Updated all attributes + keys named “Date” to something more meaningful and to avoid conflict with the SQL keyword

Schema:

Legend: Underline is primary key, bold is **foreign key**	
1.	Team(Name: char[8], Colour: char[10], Mascot: char[20]) - Mascot is Unique and not null - Colour is Unique and not null
2.	Player(Username: char[15], XP: integer, Level: integer, **TeamName**: char[8]) - XP is not null - Level is not null - TeamName is not null
3.	Item(Name: char[30], Effect: char[100], Type: char[20], Uses: integer, Cost: integer) - Type is not null
4.	Mission(Name: char[50], EventName: char[50], XP: integer) - XP is not null
5.	Location(Country: char[50], PostalCode: char[10], Name: char[50], Biome: char[20], AttackBonus: char[20]) - Biome is not null - Attack Bonus is not null - Name default is “Unknown”
6.	Gym(**Country**: char[50], **PostalCode**: char[10], **Name**: char[50], BadgeName: char[30], SponsorName: char[50]) - BadgeName is unique and is a Candidate Key, not null
7.	Pokestop(**Country**: char[50], **PostalCode**: char[10], **Name**: char[50], Rating: integer, SponsorName: char[50])
8.	Egg(ID: integer, SpeciesName: char[20], Type1: char[10], Type2: char[10], Distance: integer) - SpeciesName is not null - Type1 is not null - Distance is not null
9.	Pokemon(ID: integer, SpeciesName: char[20], Type1: char[10], Type2: char[10], Distance: integer, Nickname: char[15], HP: integer, Attack: integer, CP: integer, **GymCountry**: char[50], **GymPostalCode**: char[10], **GymName**: char[50],

	StationedAtDate: date, **FoundCountry**: char[50], **FoundPostalCode**: char[10], **FoundName**:[50]) - Species name is not null - Type 1 is not null - HP is not null - Attack is not null - CP is not null
10.	NPC(Name: char[20], Role: char[20], CanBattle: bool) - Role is not null - CanBattle is not null, default is false
11.	PlayerOwnsItem(**PlayerUsername**: char[15], **ItemName**: char[30], Quantity: integer) - quantity is not null
12.	PlayerCompletedMission(**PlayerUsername**: char[15], **MissionName**: char[50], CompletedDate: date) - CompletedDate is not null
13.	Battle(BattleDate, **PlayerUsername1**: char[15], **PlayerUsername2**: char[15], MaxCP: integer, League: char[20], Time: integer) - Time is not null - Represents elapsed time in minutes
14.	PlayerCapturedSpecies(**PlayerUsername**: char[15], **SpeciesID**: integer, CapturedDate: date) - CapturedDate is not null **Note**: Since Eggs and Pokemon are part of an ISA, decided to have a single table for player + species rather than keeping the relation as part of eggs or pokemon respectively
15.	PlayerVistedPokestop(**PlayerUsername**: char[15], **PokestopCountry**: char[50], **PokestopPostalCode**: char[10], **PokestopName**: char[50], VisitedDate: date) - VisitedDate is not null
16.	NPCAppearedAtPokestop(**NPCName***:* char[15], **PokestopCountry***:* char[50], **PokestopPostalCode***:* char[10]*,* **PokestopName**: char[50]) - Relation must appear in NPCSighting (Will enforce with assertions once taught)
17.	NPCSighting(**NPCName**: char[15], **PokestopCountry**: char[50], **PokestopPostalCode**: char[10], **PokestopName**: char[50], SightingDate: date, XP: integer, EventName: char[50], duration: integer) - XP is not null

	- Duration is the time NPC was there in hours

Functional Dependencies

1.	Team(Name, Colour, Mascot) - Name → Colour, Mascot - Mascot → Colour
2.	Player(Username, XP, Level, **TeamName**) - Username → XP, Level, TeamName - XP →Level
3.	Item(Name, Effect, Type, Cost, Uses) - Name → Effect, Type, Cost, Uses - Effect → Type - Type → Uses
4.	Mission(**Name**, EventName, XP) - Name → EventName, XP - EventName → XP
5.	Location(Country, PostalCode, Name, Biome, AttackBonus) - Country, PostalCode Name → Biome, AttackBonus - Biome → AttackBonus
6.	Gym(**Country**, **PostalCode**, **Name**, BadgeName, SponsorName) - Country, PostalCode, Name → BadgeName, SponsorName
7.	Pokestop(**Country**, **PostalCode**, **Name**, Rating, SponsorName) - Country, PostalCode, Name → Rating, SponsorName
8.	Egg(ID, SpeciesName, Type1, Type2, Distance)

	- ID → SpeciesName, Type1, Type2, Distance - SpeciesName → Type1, Type2, Distance
9.	Pokemon(ID, SpeciesName, Type1, Type2, Distance, Nickname, HP, Attack, CP, **GymCountry**, **GymPostalCode**, **GymName,** StationedAtDate, **FoundCountry**, **FoundPostalCode**, **FoundName**) - ID → SpeciesName, Type1, Type2, Distance, Nickname, HP, Attack, CP, GymCountry, GymPostalCode, GymName, StantionedAtDate, FoundAtCountry, FoundAtPostalCode, FoundAtName - SpeciesName → Type1, Type2 - SpeciesName, CP → Attack, HP - ID, GymCountry, GymPostalCode, GymName → StationedAtDate
10.	NPC(Name, Role, CanBattle) - Name → Role, CanBattle - Role → CanBattle
11.	PlayerOwnsItem(**PlayerUsername**, **ItemName**, Quantity) - PlayerUsername, ItemName → Quantity
12.	PlayerCompletedMission(**PlayerUsername**, **MissionName**, Date) - PlayerUsername, MissionName → Date
13.	Battle(Date, **PlayerUsername1**, **PlayerUsername2**, MaxCP, League, Time) - Date, PlayerUsername1, PlayerUsername2 → MaxCP, League, Time - League →MaxCP
14.	PlayerCapturedSpecies(**PlayerUsername**, **SpeciesID**, CapturedDate) - PlayerUsername, SpeciesID → CapturedDate
15.	PlayerVistedPokestop(**PlayerUsername**, **PokestopCountry**, **PokestopPostalCode**, **PokestopName**, VisitedDate) - PlayerUsername, PokestopCountry, PokestopPostalCode, PokestopName → VisitedDate
16.	NPCAppearedAtPokestop(**NPCName**, **PokestopCountry**, **PokestopPostalCode**, **PokestopName**)

<table>
<tr><td></td><td>- No non-trivial dependencies</td></tr>
<tr><td>17.</td><td>NPCSighting(<b>NPCName</b>, <b>PokestopCountry</b>, <b>PokestopPostalCode</b>, <b>PokestopName</b>, SightingDate, XP, EventName, Duration)

- NPCName, PokestopCountry, PokestopPostalCode, PokestopName → Date, Xp, EventName
- EventName → XP, Duration</td></tr>
</table>

Normalization

The following are normalized using 3NF (Either Decomposition or Synthesis, specified)

<table>
<tr><th></th><th></th></tr>
<tr><td>1.</td><td>Team(Name, Colour, Mascot)
FD’s:
- Name → Colour, Mascot
- Mascot → Colour

Normalization Steps:
Closures:
- Name$^+$ = {Name, Colour, Mascot}
- Mascot$^+$ = {Colour}
Table:
<table>
<tr><td>Left</td><td>Middle</td><td>Right</td></tr>
<tr><td>Name</td><td>Mascot</td><td>Colour</td></tr>
</table>

Keys:
- Name

Minimal Covers:
Standard Form:
- Name → Colour
- Name → Mascot
- Mascot → Colour
Reduce LHS:
- Already reduced
Remove Redundant:
- Name → Colour is redundant

Minimal Cover:</td></tr>
</table>

- Name ⟶ Mascot
- Mascot ⟶ Colour

3NF (Synthesis):

- R1(Name, Mascot) R2(Mascot, Colour)
- Done, since we have the key in a relation

Final Relation:

Team(Name, **Mascot**), MascotColour(Mascot, Colour)

2.

Player(Username, XP, Level, **TeamName**)

FD’s:

- Username ⟶ XP, Level, TeamName
- XP ⟶Level

Normalization Steps:

Closures:

- Username$^+$ = {Username, XP, Level, TeamName}
- XP$^+$ = {Level}

Table:

Left	Middle	Right
Username	XP	Level, TeamName

Keys:

- Username

Minimal Covers:

Standard Form:

- Username ⟶ XP
- Username ⟶ Level
- Username ⟶ TeamName
- XP ⟶ Level

Reduce LHS:

- Already reduced

Remove Redundant:

- Username ⟶ Level is redundant

Minimal Cover:

- Username ⟶ XP
- Username ⟶ TeamName
- XP ⟶ Level

3NF (Synthesis):

- R1(Username, XP), R2(Username, TeamName), R3(XP, Level)
- Done, since we have a key in the relation

3NF (Decomposition)

- R(Username, XP, TeamName, Level)
- Looking at the FD XP -> Level; XP is not a superkey, so we decompose:
    - R1(XP, Level), R2(Username, XP, TeamName)
- Looking at the remaining FD Username ⟶ XP, Level, TeamName; Username is a superkey so we are in BCNF and also 3NF
- All functional dependencies are preserved, so we’re done

We choose to move forward with the 3NF normalization done with decomposition, since having less tables will make it easier to work with.

Final Relation:

Player(Username, **XP,** TeamName), PlayerXPLevel(XP, Level)

3.

Item(Name, Effect, Type, Cost, Uses)

FD’s:

- Name ⟶ Effect, Type, Cost, Uses
- Effect ⟶ Type
- Type ⟶ Uses

Normalization Steps:

Closures:

- Name$^+$ = {Name, Effect, Type, Cost, Uses}
- Effect$^+$ = {Effect, Type}
- Type$^+$ = {Type, Uses}

Table:

Left	Middle	Right
Name	Effect, Type	Cost, Uses

Keys:

- Name

Minimal Covers:

Standard Form:

- Name ⟶ Effect
- Name ⟶ Type
- Name ⟶ Cost
- Name ⟶ Uses
- Effect ⟶ Type

- Type → Uses

Reduce LHS:

- Already reduced

Remove Redundant:

- Name → Type is redundant
- Name → Uses is redundant

Minimal Cover:

- Name → Effect
- Name → Cost
- Effect → Type
- Type → Uses

3NF (Synthesis):

- R1(Name, Effect), R2(Name, Cost), R3(Effect, Type), R4(Type, Uses)
- Done, since we have a key in the relation

3NF(Decomposition):

- R(Name, Effect, Type, Cost, Uses)
- Looking at the FD Type → Uses; Type is not a superkey, so we decompose:
    - R1(Type, Uses), R2(Name, Effect, Type Cost)
- Looking at the FD Effect → Type, Effect is not a superkey for R2, so we decompose R2:
    - R3(Effect, Type), R4(Name, Cost, Effect)
- Looking at the final FD Name → Effect, Type, Cost, Uses; Name is superkey so we are in BCNF and 3NF
- We are left with R1(Type, Uses), R3(Effect, Type) R4(Name, Cost, Effect)
- All functional dependencies are preserved, so we're done

We choose to move forward with the 3NF normalization done with decomposition, since having less tables will make it easier to work with.

Final Relation:

Item(Name, Cost, **Effect**), ItemEffectType(Effect, **Type**), ItemTypeUses(Type, Uses)

4. Mission(**Name**, EventName, XP)

FD's:

- Name → EventName, XP
- EventName → XP

Normalization Steps:

Closures:

- Name$^+$ = {Name, EventName, XP}
- EventName$^+$ = {EventName, XP}

Table:

Left	Middle	Right
Name	EventName	XP

Keys:

- Name

Minimal Covers:

Standard Form:

- Name ⟶ EventName
- Name ⟶ XP
- EventName ⟶ XP

Reduce LHS:

- Already Reduced

Remove Redundant:

- Name ⟶ XP is redundant

Minimal Cover:

- Name ⟶ EventName
- EventName ⟶ XP

3NF (Synthesis):

- R1(Name, EventName), R2(EventName, XP)
- Done, since we have a key in the relation

Final Relation:

Mission(Name, **EventName**), MissionEventNameXP(EventName, XP)

5. Location(Country, PostalCode, Name, Biome, AttackBonus)

FD's:

- Country, PostalCode, Name ⟶ Biome, AttackBonus
- Biome ⟶ AttackBonus

Normalization Steps:

Closures:

- (Country, PostalCode, Name)$^{+}$ = {Country, PostalCode, Name, Biome, AttackBonus}
- Biome^{+} = {Biome, AttackBonus}

Table:

Left	Middle	Right
Country, PostalCode, Name	Biome	AttackBonus

Keys:

- Country, PostalCode, Name

Minimal Covers:
Standard Form:

- Country, PostalCode, Name → Biome
- Country, PostalCode, Name → AttackBonus
- Biome → AttackBonus

Reduce LHS:

- Already Reduced

Remove Redundant:

- Country, PostalCode, Name → AttackBonus is redundant

Minimal Cover:

- Country, PostalCode, Name → Biome
- Biome → AttackBonus

3NF (Synthesis):

- R1(Country, PostalCode, Name, Biome), R2(Biome, AttackBonus)
- Done, since we have a key in the relation

Final Relation:

Location(Country, PostalCode, Name, **Biome**), BiomeAttackBonus(Biome, AttackBonus)

6. Gym(**Country**, **PostalCode**, **Name**, BadgeName, SponsorName)

FD's:

- Country, PostalCode, Name → BadgeName, SponsorName

Normalization Steps:
Closures:

- (Country, PostalCode, Name)$^+$ = {Country, PostalCode, Name, BadgeName, SponsorName}

Table:

Left	Middle	Right
Country, PostalCode, Name		BadgeName, SponsorName

Keys:

- Country, PostalCode, Name
-

Minimal Covers:
Standard Form:

- Country, PostalCode, Name → BadgeName

- Country, PostalCode, Name → SponsorName
-

Reduce LHS:

- Already reduced

Remove Redundant:

- N/A

Minimal Cover:

- Already Minimal

3NF (Synthesis):

- R1(Country, PostalCode, Name, GymBadgeName, SponsorName)
- Done, since we have a key in relation

Final Relation:

Gym(**Country**, **PostalCode**, **Name**, GymBadgeName, SponsorName)

7.

Pokestop(**Country**, **PostalCode**, **Name**, Rating, SponsorName)

FD’s:

- Country, PostalCode, Name → Rating, SponsorName

Normalization Steps:

Closures:

- (Country, PostalCode, Name)$^+$ = {Country, PostalCode, Name, Rating, SponsorName}

Table:

Left	Middle	Right
Country, PostalCode, Name		Rating, SponsorName

Keys:

- Country, PostalCode, Name

Minimal Covers:

Standard Form:

- Country, PostalCode, Name → Rating
- Country, PostalCode, Name → SponsorName

Reduce LHS:

- Already reduced

Remove Redundant:

- Nothing redundant

Minimal Cover:

- Country, PostalCode, Name → Rating

- Country, PostalCode, Name ⟶ SponsorName

3NF (Synthesis):

- R1(Country, PostalCode, Name, Rating), R2(Country, PostalCode, Name, SponsorName)
- Done, since we have a key in the relation

3NF(Decomposition)

- R(Country, PostalCode, Name, Rating, SponsorName) is already in BCNF because Country, PostalCode, Name, is a superkey of R, thus it is already in 3NF as well

We choose to move forward with the 3NF normalization done with decomposition, since having less tables will make it easier to work with.

Final Relation:

Pokestop(**Country**, **PostalCode**, **Name**, Rating, SponsorName)

8.

Egg(ID, SpeciesName, Type1, Type2, Distance)

FD's:

- ID ⟶ SpeciesName, Type1, Type2, Distance
- SpeciesName ⟶ Type1, Type2, Distance

Normalization Steps:

Closures:

- ID$^+$ = {ID, SpeciesName, Type1, Type2, Distance}
- SpeciesName$^+$ = {SpeciesName, Type1, Type2, Distance}

Table:

Left	Middle	Right
ID	SpeciesName	Type1, Type2, Distance

Keys:

- ID

Minimal Covers:

Standard Form:

- ID ⟶ SpeciesName
- ID ⟶ Type1
- ID ⟶ Type2
- ID ⟶ Distance
- SpeciesName ⟶ Type1
- SpeciesName ⟶ Type2
- SpeciesName ⟶ Distance

Reduce LHS:

- Already reduced

Remove Redundant:

- ID ⟶ Type1 is redundant
- ID ⟶ Type2 is redundant
- ID ⟶ Distance is redundant

Minimal Cover:

- ID ⟶ SpeciesName
- SpeciesName ⟶ Type1
- SpeciesName ⟶ Type2
- SpeciesName ⟶ Distance

3NF (Synthesis):

- R1(ID, SpeciesName), R2(SpeciesName, Type1), R3(SpeciesName, Type2), R4(SpeciesName, Distance)
- Done, since we have a key in relation

3NF (Decomposition):

- R(ID, SpeciesName, Type1, Type2, Distance)
- Start by looking at the FD SpeciesName ⟶ Type1, Type2, Distance; SpeciesName is not a superkey of the relation, so we decompose:
    - R1(SpeciesName, Type1, Type2, Distance), R2(ID, SpeciesName)
- Looking at the remaining FD ID ⟶ SpeciesName, Type1, Type2, Distance; ID is a superkey so we are in BCNF and also 3NF
- All functional dependencies are preserved, so we're done

We choose to move forward with the 3NF normalization done with decomposition, since having less tables will make it easier to work with.

Final Relation:

Egg(ID, **SpeciesName**), EggSpecies(SpeciesName, Type1, Type2, Distance)

9. Pokemon(ID, SpeciesName, Type1, Type2, Distance, Nickname, HP, Attack, CP, **GymCountry**, **GymPostalCode**, **GymName,** StationedAtDate, **FoundAtCountry**, **FoundAtPostalCode**, **FoundAtName**)

FD's:

- ID ⟶ SpeciesName, Type1, Type2, Distance, Nickname, HP, Attack, CP, GymCountry, GymPostalCode, StantionedAtDate, FoundAtCountry, FoundAtPostalCode
- SpeciesName ⟶ Type1, Type2
- SpeciesName, CP ⟶ Attack, HP
- ID, GymCountry, GymPostalCode, GymName ⟶ StationedAtDate

Normalization Steps:

Closures:

- ID^{+} = {ID, SpeciesName, Type1, Type2, Distance, Nickname, HP, Attack, CP, GymCountry, GymPostalCode, GymName, StantionedAtDate, FoundAtCountry, FoundAtPostalCode, FoundAtName}
- SpeciesName^{+} = {SpeciesName, Type1, Type2}
- SpeciesName, CP^{+} = {SpeciesName, CP, Attack, HP, Type1, Type2}
- (ID, GymCountry, GymPostalCode, GymName)$^{+}$ = {ID, SpeciesName, Type1, Type2, Distance, Nickname, HP, Attack, CP, GymCountry, GymPostalCode, GymName, StantionedAtDate, FoundAtCountry, FoundAtPostalCode, FoundAtName}

Table:

Left	Middle	Right
ID	SpeciesName, CP, GymCountry, GymPostalCode, GymName	Type1, Type2, HP, Attack, Nickname, StationedAtDate, FoundAtCountry, FoundAtPostalCode, FoundAtName

Keys:

- ID

Minimal Covers:

Standard Form:

- ID ⟶ SpeciesName
- ID ⟶ Type1
- ID ⟶ Type2
- ID ⟶ Distance
- ID ⟶ Nickname
- ID ⟶ HP
- ID ⟶ Attack
- ID ⟶ CP
- ID ⟶ GymCountry
- ID ⟶ GymPostalCode
- ID ⟶ GymName
- ID ⟶ StationedAtDate
- ID ⟶ FoundCountry
- ID ⟶ FoundPostalCode
- ID ⟶ FoundAtDate
- SpeciesName ⟶ Type1

- SpeciesName → Type2
- SpeciesName, CP →HP
- SpeciesName, CP → Attack
- ID, GymPostalCode, GymCountry, GymName → SpeciesName
- ID, GymPostalCode, GymCountry, GymName → Type1
- ID, GymPostalCode, GymCountry, GymName → Type2
- ID, GymPostalCode, GymCountry, GymName → Distance
- ID, GymPostalCode, GymCountry, GymName → Nickname
- ID, GymPostalCode, GymCountry, GymName → HP
- ID, GymPostalCode, GymCountry, GymName → Attack
- ID, GymPostalCode, GymCountry, GymName → CP
- ID, GymPostalCode, GymCountry, GymName → GymCountry
- ID, GymPostalCode, GymCountry, GymName → GymPostalCode
- ID, GymPostalCode, GymCountry, GymName → GymName
- ID, GymPostalCode, GymCountry, GymName → StationedAtDate
- ID, GymPostalCode, GymCountry, GymName → FoundCountry
- ID, GymPostalCode, GymCountry, GymName → FoundPostalCode
- ID, GymPostalCode, GymCountry, GymName → FoundAtDate
- ID, FoundPostalCode, FoundName, FoundCountry → SpeciesName
- ID, FoundPostalCode, FoundName, FoundCountry → Type1
- ID, FoundPostalCode, FoundName, FoundCountry → Type2
- ID, FoundPostalCode, FoundName, FoundCountry → Distance
- ID, FoundPostalCode, FoundName, FoundCountry → Nickname
- ID, FoundPostalCode, FoundName, FoundCountry → HP
- ID, FoundPostalCode, FoundName, FoundCountry → Attack
- ID, FoundPostalCode, FoundName, FoundCountry → CP
- ID, FoundPostalCode, FoundName, FoundCountry → GymCountry
- ID, FoundPostalCode, FoundName, FoundCountry → GymPostalCode
- ID, FoundPostalCode, FoundName, FoundCountry → GymName
- ID, FoundPostalCode, FoundName, FoundCountry → StationedAtDate
- ID, FoundPostalCode, FoundName, FoundCountry → FoundCountry
- ID, FoundPostalCode, FoundName, FoundCountry → FoundPostalCode
- ID, FoundPostalCode, FoundName, FoundCountry → FoundAtDate

Reduce LHS:

- ID → SpeciesName
- ID → Type1
- ID → Type2
- ID → Distance
- ID → Nickname
- ID → HP
- ID → Attack
- ID → CP

	- ID → GymCountry - ID → GymPostalCode - ID → GymName - ID → StationedAtDate - ID → FoundCountry - ID → FoundPostalCode - ID → FoundName - ID → FoundAtDate - SpeciesName → Type1 - SpeciesName → Type2 - SpeciesName, CP → Attack - SpeciesName, CP → HP - ID → SpeciesName - ID → Type1 - ID → Type2 - ID → Distance - ID → Nickname - ID → HP - ID → Attack - ID → CP - ID → GymCountry - ID → GymPostalCode - ID → GymName - ID → StationedAtDate - ID → FoundCountry - ID → FoundPostalCode - ID → FoundAtDate - ID → SpeciesName - ID → Type1 - ID → Type2 - ID → Distance - ID → Nickname - ID → HP - ID → Attack - ID → CP - ID → GymCountry - ID → GymPostalCode - ID → GymName - ID → StationedAtDate - ID → FoundCountry - ID → FoundPostalCode - ID → FoundName - ID → FoundAtDate

Remove Redundant:

- Duplicates after minimizing LHS of ID + FoundPostalCode,FoundCountry, FoundName can be removed
- Duplicates after minimizing LHS of ID + FoundPostalCode,FoundCountry, FoundName can be removed
- Since ID determines SpeciesName, we can remove ID functional dependencies that are also determined by the SpeciesName functional Dependencies

Minimal Cover:

- ID ⟶ SpeciesName
- ID ⟶ CP
- ID ⟶ Distance
- ID ⟶ Nickname
- ID ⟶ GymCountry
- ID ⟶ GymPostalCode
- ID ⟶ GymName
- ID ⟶ StationedAtDate
- ID ⟶ FoundCountry
- ID ⟶ FoundPostalCode
- ID ⟶ FoundName
- ID ⟶ FoundAtDate
- SpeciesName ⟶ Type1
- SpeciesName ⟶ Type2
- SpeciesName, CP ⟶ HP
- SpeciesName, CP ⟶ Attack
-

3NF (Decomposition):

- R1(ID, SpeciesName)
- R2(ID, CP)
- R3(ID, Distance)
- R4(ID, Nickname)
- R5(ID, GymCountry)
- R6(ID, GymPostalCode)
- R7(ID, GymName)
- R8(ID, StationedAtDate)
- R9(ID, FoundCountry)
- R10(ID, FoundPostalCode)
- R11(ID, FoundName)
- R13(SpeciesName, Type1)
- R14(SpeciesName, Type2)
- R15(SpeciesName, CP, HP)
- R16(SpeciesName, CP, Attack)

We optimize it this to the following:

- R1(ID, SpeciesName, CP, Distance, Nickname, GymCountry, GymPostalCode, GymName, StationedAtDate, FoundCountry, FoundPostalCode, FoundName)
    - Since ID is the superkey, this is in BCNF
- R2(SpeciesName, Type1, Type2)
    - Since SpeciesName is the superkey, this is in BCNF
- R3(SpeciesName, CP, HP, Attack)
    - Since SpeciesName and CP is the superkey, this is in BCNF

Since one of the relations has the key, we're done

Final Relation:

Pokemon(ID, **SpeciesName**, **CP**, Distance, Nickname, **GymCountry**, **GymPostalCode**, **GymName**, StationedAtDate, **FoundCountry**, **FoundPostalCode**, **FoundName**)

PokemonSpeciesTypes(SpeciesName, Type1, Type2)
PokemonSpeciesCP(**SpeciesName**, CP, HP, Attack)

10. NPC(Name, Role, CanBattle)

FD's:

- Name → Role, CanBattle
- Role → CanBattle

Normalization Steps:
Closures:

- Name$^+$ = {Name, Role, CanBattle}
- Role$^+$ = {Role, CanBattle}

Table:

Left	Middle	Right
Name	Role	CanBattle

Keys:

- Name

Minimal Covers:
Standard Form:

- Name → Role
- Name → CanBattle
- Role → CanBattle

Reduce LHS:

- Already reduced

Remove Redundant:

- Name → CanBattle is redundant

Minimal Cover:

- Name → Role
- Role → CanBattle

3NF (Synthesis):

- R1(Name, Role), R2(Role, CanBattle)
- Done, since we have a key in relation

Final Relation:

NPC(Name, **Role**), RoleCanBattle(Role, CanBattle)

11. PlayerOwnsItem(**PlayerUsername**, **ItemName**, Quantity)

FD’s:

- PlayerUsername, ItemName → Quantity

Normalization steps:

Closures:

- (PlayerUsername, ItemName)$^+$ = {PlayerUsername, ItemName, Quantity}

Table:

Left	Middle	Right
PlayerUsername, ItemName		Quantity

Keys:

- PlayerUsername, ItemName

Minimal Covers:

- Already Minimal

3NF (Synthesis)

- Already in 3NF

Final Relation:

PlayerOwnsItem(**PlayerUsername**, **ItemName**, Quantity)

12. PlayerCompletedMission(**PlayerUsername**, **MissionName**, Date)

FD’s:

PlayerUsername, MissionName → Date

Normalization Steps:

Closures:

- (PlayerUsername, MissionName)$^+$ = {PlayerUsername, MissionName, Date}

Table:

Left	Middle	Right
PlayerUsername, MissionName		Date

Keys:

- PlayerUsername, MissionName

Minimal Covers:

- Already minimal

3NF (Synthesis):

- Already in 3NF

Final Relation:

PlayerCompletedMission(**PlayerUsername**, **MissionName**, Date)

13. Battle(Date, **PlayerUsername1**, **PlayerUsername2**, MaxCP, League, Time)

FD’s:

- Date, PlayerUsername1, PlayerUsername2 → MaxCP, League, Time
- League →MaxCP

Normalization Steps:

Closures:

- (Date, PlayerUsername1, PlayerUsername2)$^+$ = {Date, PlayerUsername1, PlayerUsername2, MaxCP, League, Time}
- League$^+$ = {MaxCP}

Table:

Left	Middle	Right
Date, PlayerUsername1, PlayerUsername2	League	MaxCP, Time

Keys:

	- Date, PlayerUsername1, PlayerUsername2 Minimal Covers: Standard Form: - Date, PlayerUsername1, PlayerUsername2 ⟶ MaxCP - Date, PlayerUsername1, PlayerUsername2 ⟶ League - Date, PlayerUsername1, PlayerUsername2 ⟶ Time - League ⟶ MaxCP Reduce LHS: - Already reduced Remove Redundant: - Date, PlayerUsername1, PlayerUsername2 ⟶ MaxCP is redundant Minimal Cover: - Date, PlayerUsername1, PlayerUsername2 ⟶ League - Date, PlayerUsername1, PlayerUsername2 ⟶ Time - League ⟶ MaxCP 3NF (Synthesis): - R1(Date, PlayerUsername1, PlayerUsername2, League), R2(Date, PlayerUsername1, PlayerUsername2, Time), R3(League, MaxCP) - Done, since we have a key in relation 3NF (Decomposition): - R(Date, PlayerUsername1, PlayerUsername2, League, Time, MaxCP) - Start by looking at FD League ⟶ MaxCP; League is not a superkey so we decompose, - R1(League, MaxCP), R2(Date, PlayerUsername1, PlayerUsername2, League, Time) - Looking at the remaining FD: Date, PlayerUsername1, PlayerUsername2 ⟶ MaxCP, League, Time; Date, PlayerUsername1, PlayerUsername2 is a superkey so we are in BCNF and also 3NF - All functional dependencies are preserved, so we're done We choose to move forward with the 3NF normalization done with decomposition, since having less tables will make it easier to work with. **Final Relation:** Battle(**Date**, **PlayerUsername1**, **PlayerUsername2**, **League**, Time), BattleLeagueCP(League, MaxCP)
14.	PlayerCapturedSpecies(**PlayerUsername**, **SpeciesID**, CapturedDate) FD's: - PlayerUsername, SpeciesID ⟶ CapturedDate

Normalization Steps:

Closures:

- (PlayerUsername, SpeciesID)$^+$ = {PlayerUsername, SpeciesID, Date}

Table:

Left	Middle	Right
PlayerUsername, SpeciesID		Date

Keys:

- PlayerUsername, SpeciesID

Minimal Covers:

- Already minimal

3NF (Synthesis):

- Already in 3NF

Final Relation:

PlayerCapturedSpecies(**PlayerUsername**, **SpeciesID**, CapturedDate)

15. PlayerVistedPokestop(**PlayerUsername**, **PokestopCountry**, **PokestopPostalCode**, VisitedDate)

FD’s:

- PlayerUsername, PokestopCountry, PokestopPostalCode → VisitedDate

Normalization Steps:

Closures:

- (PlayerUsername, PokestopCountry, PokestopPostalCode)$^+$ = {PlayerUsername, PokestopCountry, PokestopPostalCode, VisitedDate}

Table:

Left	Middle	Right
PlayerUsername, PokestopCountry, PokestopPostalCode		VisitedDate

Keys:

- PlayerUsername, PokestopCountry, PokestopPostalCode

Minimal Covers:

- Already minimal

3NF (Synthesis):

- Already in 3NF

Final Relation:

PlayerVistedPokestop(**PlayerUsername**, **PokestopCountry**, **PokestopPostalCode**, VisitedDate)

16. NPCAppearedAtPokestop(**NPCName**, **PokestopCountry**, **PokestopPostalCode**, **PokestopName**)

No non-trivial dependencies, no normalization needed

17. NPCSighting(**NPCName**, **PokestopCountry**, **PokestopPostalCode**, **PokestopName**, SightingDate, XP, EventName, Duration)

FD’s:

- NPCName, PokestopCountry, PokestopPostalCode, PokestopName ⟶ SightingDate, XP, EventName
- EventName ⟶ XP, Duration

Normalization Steps:

Closures:

- (NPCName, PokestopCountry, PokestopPostalCode, PokestopName)$^+$ = {NPCName, PokestopCountry, PokestopPostalCode, PokestopName, SightingDate, XP, EventName, Duration}
- EventName$^+$ = {EventName, XP, Duration}

Table:

Left	Middle	Right
NPCName, PokestopCountry, PokestopPostalCode, PokestopName	EventName	SightingDate, XP, Duration

Keys:

- NPCName, PokestopCountry, PokestopPostalCode, PokestopName

Minimal Covers:

Standard Form:

- NPCName, PokestopCountry, PokestopPostalCode, PokestopName, SightingDate ⟶ XP
- NPCName, PokestopCountry, PokestopPostalCode, PokestopName, SightingDate ⟶ EventName
- EventName ⟶ XP
- EventName ⟶ Duration

Reduce LHS:

- Already reduced

Remove Redundant:

- NPCName, PokestopCountry, PokestopPostalCode, PokestopName, SightingDate → XP is redundant

Minimal Cover:

- NPCName, PokestopCountry, PokestopPostalCode, SightingDate → EventName
- EventName → XP
- EventName → Duration

3NF (Synthesis):

- R1(NPCName, PokestopCountry, PokestopPostalCode, EventName), R2(EventName, XP), R3(EventName, Duration)
- Done, since we have a key in relation

3NF (Decomposition):

- R(NPCName, PokestopCountry, PokestopPostalCode, EventName, SightingDate)
- First we look at the FD EventName → XP, Duration; EventName is not a superkey so we decompose:
    - R1(EventName, XP, Duration), R2(NPCName, PokestopCountry, PokestopPostalCode, EventName, SightingDate)
- Looking at the remaining FD NPCName, PokestopCountry, PokestopPostalCode, PokestopName → SightingDate, XP, EventName, the LHS is already a superkey, so we are in BCNF and also 3NF
- All functional dependencies are preserved, so we're done

We choose to move forward with the 3NF normalization done with decomposition, since having less tables will make it easier to work with.

Final Relation:

NPCSighting(**NPCName**, **PokestopCountry**, **PokestopPostalCode**, **EventName,** SightingDate), NPCSightingEventName(EventName, XP, Duration)

SQL DDL - Tables

1.

```
CREATE TABLE Team(
    Name CHAR(8) PRIMARY KEY,
    Mascot CHAR(20) NOT NULL UNIQUE,
    FOREIGN KEY (Mascot)
        REFERENCES MascotColour(Mascot)
        ON DELETE CASCADE
)

CREATE TABLE MascotColour (
    Mascot CHAR(20) PRIMARY KEY,
    Colour CHAR(20) NOT NULL UNIQUE
)
```

2.

```
CREATE TABLE Player(
   Username CHAR(15) PRIMARY KEY,
   XP INTEGER NOT NULL,
   TeamName CHAR(8) NOT NULL,
   FOREIGN KEY (TeamName),
      REFERENCES Team(Name)
      ON DELETE CASCADE,
   FOREIGN KEY (XP)
      REFERENCES PlayerXPLevel(XP)
      ON DELETE CASCADE
)


CREATE TABLE PlayerXPLevel(
   XP INTEGER PRIMARY KEY,
   Level INTEGER NOT NULL
)
```

3.

```
CREATE TABLE Item(
   Name CHAR(30) PRIMARY KEY,
   Cost INTEGER,
   Effect CHAR(100),
   FOREIGN KEY (Effect)
      REFERENCES ItemEffectType(Effect)
      ON DELETE CASCADE
)

CREATE TABLE ItemEffectType(
   Effect CHAR(100) PRIMARY KEY,
   Type CHAR(20) NOT NULL,
   FOREIGN KEY (Type)
      REFERENCES ItemTypeUses(Type)
      ON DELETE CASCADE
)
```

```
CREATE TABLE ItemTypeUses(
   Type CHAR(20) PRIMARY KEY,
   Uses INTEGER
)
```

4.

```
CREATE TABLE Mission(
   Name CHAR(50) PRIMARY KEY,
   EventName CHAR(50),
   FOREIGN KEY (EventName)
      REFERENCES MissionEventNameXP(EventName)
      ON DELETE CASCADE
)

CREATE TABLE MissionEventNameXP(
   EventName CHAR(50) PRIMARY KEY,
   XP INTEGER NOT NULL
)
```

5.

```
CREATE TABLE Location(
   Country CHAR(50),
   PostalCode CHAR(10),
   Name CHAR(50) DEFAULT ‘Unknown’,
   Biome CHAR(20) NOT NULL,
   PRIMARY KEY (Country, PostalCode, Name),
   FOREIGN KEY (Biome)
      REFERENCES BiomeAttackBonus(Biome)
      ON DELETE CASCADE
)

CREATE TABLE BiomeAttackBonus(
   Biome CHAR(20) PRIMARY KEY,
   AttackBonus CHAR(20) NOT NULL
)
```

6.

```
CREATE TABLE Gym(
   Country CHAR(50),
   PostalCode CHAR(10),
   Name CHAR(50),
   BadgeName CHAR(30) NOT NULL UNIQUE,
   SponsorName CHAR(50),
   PRIMARY KEY (Country, PostalCode, Name),
   FOREIGN KEY (Country, PostalCode, Name)
      REFERENCES Location(Country, PostalCode, Name)
      ON DELETE CASCADE
```

	)

7.	CREATE TABLE Pokestop(
	    Country CHAR(50),
	    PostalCode CHAR(10),
	    Name CHAR(50),
	    Rating INTEGER,
	    SponsorName CHAR(50),
	    PRIMARY KEY (Country, PostalCode, Name),
	    FOREIGN KEY (Country, PostalCode, Name)
	        REFERENCES Location(Country, PostalCode, Name)
	        ON DELETE CASCADE
	)

8.	CREATE TABLE Egg(
	    ID INTEGER PRIMARY KEY,
	    SpeciesName CHAR(20) NOT NULL,
	    FOREIGN KEY (SpeciesName)
	        REFERENCES EggSpecies(SpeciesName)
	        ON DELETE CASCADE
	)
	
	CREATE TABLE EggSpecies(
	    SpeciesName CHAR(20) PRIMARY KEY,
	    Type1 CHAR(10) NOT NULL,
	    Type2 CHAR(10),
	    Distance INTEGER NOT NULL
	)

9.	CREATE TABLE Pokemon(
	    ID INTEGER PRIMARY KEY,
	    SpeciesName CHAR(20) NOT NULL,
	    CP INTEGER NOT NULL,
	    Distance INTEGER,
	    Nickname CHAR(15),
	    GymCountry CHAR(50),
	    GymPostalCode CHAR(10),
	    GymName CHAR(50),
	    StationedAtDate DATE,
	    FoundCountry CHAR(50),
	    FoundPostalCode CHAR(10),
	    FoundName CHAR(50),
	    FOREIGN KEY (GymCountry, GymPostalCode, GymName)
	        REFERENCES Gym(Country, PostalCode, Name)

```
      ON DELETE CASCADE,
   FOREIGN KEY (FoundCountry, FoundPostalCode, FoundName)
      REFERENCES Location(Country, PostalCode, Name)
      ON DELETE CASCADE,
   FOREIGN KEY (SpeciesName)
      REFERENCES PokemonSpeciesTypes(SpeciesName)
      ON DELETE CASCADE,
   FOREIGN KEY (CP)
      REFERENCES PokemonXP(CP)
      ON DELETE CASCADE
)

CREATE TABLE PokemonSpeciesTypes(
   SpeciesName CHAR(20) PRIMARY KEY,
   Type1 CHAR(10) NOT NULL,
   Type2 CHAR(10)
)

CREATE TABLE PokemonSpeciesCP(
   SpeciesName CHAR(20),
   CP INTEGER,
   HP INTEGER NOT NULL,
   Attack INTEGER NOT NULL,
   PRIMARY KEY (SpeciesName, CP),
   FOREIGN KEY (SpeciesName)
      REFERENCES PokemonSpeciesType(SpeciesName)
      ON DELETE CASCADE
)
```

10.

```
CREATE TABLE NPC(
   Name CHAR(20) PRIMARY KEY,
   Role CHAR(20) NOT NULL,
   FOREIGN KEY (Role)
      REFERENCES RoleCanBattle(Role)
      ON DELETE CASCADE
)

CREATE TABLE RoleCanBattle(
   RoleCHAR(20) PRIMARY KEY,
   CanBattle BOOL NOT NULL DEFAULT FALSE
)
```

11.

```
CREATE TABLE PlayerOwnsItem(
    PlayerUsername CHAR(15),
    ItemName CHAR(30),
    Quantity INTEGER NOT NULL,
    PRIMARY KEY (PlayerUsername, ItemName),
    FOREIGN KEY (PlayerUsername)
        REFERENCES Player(Username)
        ON DELETE CASCADE,
    FOREIGN KEY (ItemName),
        REFERENCES Item(ItemName)
        ON DELETE CASCADE
)
```

12.

```
CREATE TABLE PlayerCompletedMission(
    PlayerUsername CHAR(15),
    MissionName CHAR(50),
    CompletedDate DATE NOT NULL,
    PRIMARY KEY (PlayerUsername, MissionName),
    FOREIGN KEY (PlayerUsername)
        REFERENCES Player(Username)
        ON DELETE CASCADE,
    FOREIGN KEY (MissionName)
        REFERENCES Mission(Name)
        ON DELETE CASCADE
)
```

13.

```
CREATE TABLE Battle(
    DateOccurred DATE,
    PlayerUsername1 CHAR(15),
    PlayerUsername2 CHAR(15),
    League CHAR(20),
    Time INTEGER NOT NULL,
    PRIMARY KEY (DateOccurred, PlayerUsername1, PlayerUsername2),
    FOREIGN KEY (PlayerUsername1)
        REFERENCES Player(Username)
        ON DELETE CASCADE,
   FOREIGN KEY (PlayerUsername2)
        REFERENCES Player(Username)
        ON DELETE CASCADE,
    FOREIGN KEY (League)
        REFERENCES LeagueMaxCP(League)
        ON DELETE CASCADE
)
```

```
CREATE TABLE LeagueMaxCP(
    League CHAR(20) PRIMARY KEY,
    MaxCP INTEGER
)
```

14.

```
CREATE TABLE PlayerCapturedSpecies(
    PlayerUsername CHAR(15),
    SpeciesID INTEGER,
    CapturedDate DATE NOT NULL,
    PRIMARY KEY (PlayerUsername, SpeciesID),
    FOREIGN KEY (PlayerUsername)
        REFERENCES Player(Username)
        ON DELETE CASCADE,
    FOREIGN KEY (SpeciesID)
        REFERENCES Pokemon(ID)
        ON DELETE CASCADE
)
```

15.

```
CREATE TABLE PlayerVisitedPokestop(
    PlayerUsername CHAR(15),
    PokestopCountry CHAR(50),
    PokestopPostalCode CHAR(10),
    PokestopName CHAR(50),
    VisitedDate INTEGER NOT NULL,
    PRIMARY KEY (PlayerUsername, PokestopCountry, PokestopPostalCode,
PokestopName),
    FOREIGN KEY (PlayerUsername)
       REFERENCES PlayerXP(Name)
        ON DELETE CASCADE,
    FOREIGN KEY (PokestopCountry, PokestopPostalCode, PokestopName)
        REFERENCES Pokestop(Country, PostalCode, Name)
        ON DELETE CASCADE
)
```

16.

```
CREATE TABLE NPCAppearedAtPokestop(
    NPCName CHAR(20),
    PokestopCountry CHAR(50),
    PokestopPostalCode CHAR(10),
    PokestopName CHAR(50),
    PRIMARY KEY (NPCName, PokestopCountry, PokestopPostalCode, PokestopName),
    FOREIGN KEY (NPCName)
       REFERENCES NPC(Name)
        ON DELETE CASCADE,
```

	FOREIGN KEY (PokestopCountry, PokestopPostalCode, PokestopName) REFERENCES Pokestop(Country, PostalCode, Name) ON DELETE CASCADE) **We will need to enforce that entries here must appear in appear in NPCSighting, which we will do when we are taught assertions**
17.	CREATE TABLE NPCSighting(NPCName CHAR(20), PokestopCountry CHAR(50), PokestopPostalCode CHAR(10), PokestopName CHAR(50), SightingDate DATE, EventName CHAR(50), PRIMARY KEY (NPCName, PokestopCountry, PokestopPostalCode, PokestopName, SightingDate), FOREIGN KEY (NPCName, PokestopCountry, PokestopPostalCode, PokestopName) REFERENCES NPCAppearedAtPokestop(NPCName, Country, PostalCode, Name) ON DELETE CASCADE, FOREIGN KEY (EventName) REFERENCES NPCSightingEventNameXP(EventName) ON DELETE CASCADE) CREATE TABLE NPCSightingEventName(EventName CHAR(50) PRIMARY KEY, XP INTEGER NOT NULL, Duration INTEGER)

SQL DDL - Insert

1.	INSERT INTO Team(Name, Mascot)
	VALUES (‘Valor’, ‘Moltres’),
	(‘Mystic’, ‘Articuno’),
	(‘Instinct’, ‘Zapdos’),

	(‘Aqua’, ‘Kyogre’),
	(‘Magma’, ‘Groudon’);
	
	INSERT INTO MascotColour(Mascot, Colour)
	VALUES (‘Moltres’, ‘Red’),
	(‘Articuno’, ‘Blue’),
	(‘Zapdos’, ‘Yellow’),
	(‘Kyogre’, ‘Sapphire’),
	(‘Groudon’, ‘Crimson’);

2.	INSERT INTO Player(Username, XP, TeamName) VALUES (‘Steph4n’, 6000, ‘Valor’), (‘J@son’, 30000, ‘Mystic’), (‘B0b’, 40000, ‘Instinct’), (‘Greg0r’, 40000, ‘Instinct’), (‘N0rm’, 40000, ‘Mystic’), (‘Go4t’, 6000, ‘Valor’), (‘J3ssica’, 304, ‘Aqua’), (‘R4ch3l’, 404, ‘Magma’); INSERT INTO PlayerXPLevel(XP, Level) VALUES (6000, 6), (30000, 30), (40000, 40), (300, 1), (400, 1);

3.	INSERT INTO Item(Name, Cost, Effect)
	VALUES (‘PokeBall’, 100, ‘Catches Pokemon’),
	(‘Incense’, 40, ‘Attracts Pokemon’),
	(‘Incubator’, 150, ‘Hatches eggs’),
	(‘Raid Pass’, 100, ‘Raid Entry Ticket’),
	(‘Lure Module’, 100, ‘Lures Pokemon’);
	
	INSERT INTO ItemEffectType(Effect, Type)
	VALUES (‘Catches Pokemon’, ‘Ball’),
	(‘Attracts Pokemon’, ‘Buff’),
	(‘Hatches Eggs’, ‘Egg Incubator’),
	(‘Raid Entry Ticket’, ‘Raid Items’),
	(‘Lures Pokemon’, ‘Lure’);
	
	INSERT INTO ItemTypeUses(Type, Uses)
	VALUES (‘Ball’, 20),

	(‘Buff’, 1), (‘Egg Incubator’, 5), (‘Raid Items’, 1), (‘Lure’, 1);

4.	INSERT INTO Mission(Name, EventName)
	VALUES (‘Catch 10 Pokemon’, ‘Default’),
	(‘A Spooky Message 2018’, ‘Halloween 2018’),
	(‘Go Fest 1st Part’, ‘GO Fest 2023 Fascinating Facets’),
	(‘All-in-One 151 1st Part’, ‘All-in-One’),
	(‘City Safari:Seoul 2023’, ‘City Safari 2023’);
	
	INSERT INTO MissionEventNameXP(EventName, XP)
	VALUES (‘Default’, 600),
	(‘Halloween 2018’, 1080),
	(‘Go Fest 2023 Fascinating Facets’, 2023),
	(‘All-in-one’, 5100),
	(‘City Safari 2023’ 2023);

5.	INSERT INTO Location(Country, PostalCode, Name, Biome) VALUES (‘Canada’, ‘V6T 1Z4’, ‘UBC Science’, ‘Nature’), (‘Canada’, ‘K1A 0A6’, ‘House of Commons’, ‘Water’), (‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’, ‘Snow’), (‘USA’, ‘NM 87111’, ‘White Residence’, ‘Toxic’), ('France', '75001', 'Louvre Museum', 'Nature'), ('Australia', '2000', 'Sydney Opera House', 'Water'), ('Brazil', '71020-970', 'Christ the Redeemer', 'Mountain'), ('Japan', '100-0001', 'Shibuya Crossing', 'Nature'), (‘UK’, ‘SW1A 1BQ’, ‘Buckingham Palace’, ‘Enchanted’); INSERT INTO BiomeAttackBonus(Biome, AttackBonus) VALUES (‘Nature’, ‘Grass’), (‘Water’, ‘Water’), (‘Snow’, ‘Ice’), (‘Toxic’, ‘Poison’), (‘Mountain’, ‘Ground’), (‘Enchanted’, ‘Fairy’);

6.	INSERT INTO Gym(Country, PostalCode, Name, BadgeName, SponsorName)
	VALUES (‘Canada’, ‘V6T 1Z4’, ‘UBC Science’, ‘ICICS Building’, ‘UBC’),
	(‘Canada’, ‘K1A 0A6’, ‘House of Commons’, ‘House of Commons CAN’, ‘Gov Of Canada’),
	(‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’, ‘GuestServicesBlckcmb’ ,‘Whistler’),

	(‘USA’, ‘NM 87111’, ‘White Residence, ‘TheOneWhoKnocks’, ‘Heisenberg’), ('Brazil', '71020-970', 'Christ the Redeemer', ‘The Redeemer’, 'Church'), ('Japan', '100-0001', 'Shibuya Crossing', ‘Shibuya’, 'Shibuya'), (‘UK’, ‘SW1A 1BQ’, ‘Buckingham Palace’, ‘BuckinghamPalace’, ‘Royal Family’);
7.	INSERT INTO Pokestop(Country, PostalCode, Name, Rating, SponsorName) VALUES (‘Canada’, ‘V6T 1Z4’, ‘UBC Science’, 0, ‘Starbucks’), (‘Canada’, ‘K1A 0A6’, ‘House of Commons’, 7, ‘Gov Of Canada’), (‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’, 7, ‘Whistler’), (‘USA’, ‘NM 87111’, ‘White Residence’, 10, ‘Heisenberg’), ('France', '75001', 'Louvre Museum', 8, 'Louvre Staff'), ('Australia', '2000', 'Sydney Opera House', 9, 'Kangaroos'), (‘UK’, ‘SW1A 1BQ’, ‘Buckingham Palace’, 9, ‘Royal Family’);

8.	INSERT INTO Egg(ID, SpeciesName)
	VALUES (0001, ‘MagiKarp’)
	(0002, ‘Machop’),
	(0003, ‘Meowth’),
	(0004, ‘Deino’),
	(0005, ‘Larvitar’);
	
	INSERT INTO EggSpecies(SpeciesName, Type1, Type2, Distance)
	VALUES (‘MagiKarp’, ‘Water’, NULL, 2),
	(‘Machop’, ‘Fighting’, NULL, 5),
	(‘Meowth’, ‘Normal’, NULL, 7),
	(‘Deino’, ‘Dark’, ‘Dragon’, 10),
	(‘Larvitar’, ‘Rock’, ‘Ground’, 12);

9.	INSERT INTO Pokemon(ID, SpeciesName, CP, Distance, Nickname, GymCountry, GymPostalCode, GymName, StationedAtDate, FoundCountry, FoundPostalCode, FoundName)
	VALUES (0006, ‘Slaking’, 3804, 114, NULL, ‘Canada’, ‘V6T 1Z4’, ‘UBC Science’, ‘2023-10-19’, ‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’),
	(0007, ‘Vaporeon’, 2616, 0, ‘Squidward’, NULL, NULL, NULL, ‘Canada’, ‘V6T 1Z4’, ‘UBC Science’),
	(0008, ‘Dialga’, 2242, 0, NULL, NULL, NULL, NULL, ‘Canada’, ‘K1A 0A6’, ‘House of Commons’),
	(0009, ‘Abomasnow’, 1803, 1, ‘ObamaSnow’, ‘Canada’, ‘K1A 0A6’, ‘House of Commons’, ‘Canada’, ‘K1A 0A6’, ‘House of Commons’),
	(0010, ‘Regirock’, 1319, 3, ‘Dwayne’, ‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’, ‘USA’, ‘NM 87111’, ‘White Residence’);
	
	INSERT INTO PokemonSpeciesTypes(SpeciesName, Type1, Type2)
	VALUES (‘Slacking’, ‘Normal’, NULL),

	(‘Vaporeon’, ‘Water’, NULL), (‘Dialga’, ‘Steel’, ‘Dragon’), (‘Abomasnow’, ‘Grass’, ‘Ice’), (‘Regirock’, ‘Rock’, NULL); INSERT INTO PokemonSpeciesCP(SpeciesName, CP, HP, Attack) VALUES (‘Slaking’, 3804, 218, 3), (‘Vaporeon’, 2616, 215, 2), (‘Dialga’, 2242, 131, 3), (‘Abomasnow’, 1803, 154, 2), (‘Regirock’, 1319, 105, 1);

10.	INSERT INTO NPC(Name, Role)
	VALUES (‘Cadela’, ‘Team Leader’),
	(‘Professor Oak’, ‘Professor’),
	(‘Arlo’, ‘Team Rocket Leader’),
	(‘Male Grunt’, ‘Team Rocket Grunt’),
	(‘Balloon Grunt’, ‘Team Rocket Balloon’);
	
	INSERT INTO RoleCanBattle(Role, CanBattle)
	VALUES (‘TeamLeader’, FALSE),
	(‘PROFESSOR’, FALSE),
	(‘Team Rocket Leader’, TRUE),
	(‘Team Rocket Grunt’, TRUE),
	(‘Team Rocket Balloon’, ‘TRUE’);

11.	INSERT INTO PlayerOwnsItem(PlayerUsername, ItemName, Quantity) VALUES (‘Steph4n’, ‘LureModule’, 1), (‘J@son’, ‘Pokeball’, 20), (‘B0b’, ‘Pokeball’, 10), (‘B0b’, ‘LureModule’, 5), (‘Greg0r’, ‘Pokeball’, 100), (‘N0rm’, ‘Pokeball’, 420), (‘Go4t’, ‘LureModule’, 23), (‘B0b’, ‘Raid Pass’, 2);

12.	INSERT INTO PlayerCompletedMission(PlayerUsername, MissionName, CompletedDate)
	VALUES (‘B0b’, ‘Catch 10 Pokemon’, ‘2018-09-11’),
	(‘B0b’, ‘All-in-One 151 1st Part’, ‘2021-02-20’),
	(‘J@son’, ‘Catch 10 Pokemon’, ‘2018-10-01’),
	(‘Steph4n’, ‘Catch 10 Pokemon’, ‘2023-10-19’),
	(‘J3ssica’, ‘Go Fest 1st Part’, ‘2023-08-22’);

13.

```
INSERT INTO BattleLeague(DateOccurred, PlayerUsername1, PlayerUsername2,
League, Time)
VALUES (‘2023-10-19’, ‘B0b’, ‘J@son’, ‘Great League’, 5),
           (‘2023-10-18’, ‘B0b’, ‘J3ssica’, ‘Ultra League’, 4),
           (‘2023-10-18’, ‘J@son’, ‘Steph4n’, ‘Master League’, 5),
           (‘2022-01-05’, ‘J3ssica’, ‘R4chel’, ‘Training’, 10),
           (‘2022-10-05’, ‘N0rm’, ‘J3ssica’, ‘Ultra League’, 3),
           (‘2023-01-18’, ‘J@son’, ‘Greg0r’, ‘Master League’, 1),
           (‘2021-01-05’, ‘Go4t’, ‘R4chel’, ‘Training’, 10),
           (‘2018-05-10’, ‘B0b’, ‘J@son’, ‘Friendly’, 1);

INSERT INTO LeagueMaxCP(League, CP)
VALUES (‘Great League’, 1500),
           (‘Ultra League’, 2500),
           (‘Master League’, 9999),
           (‘Training’, 1500),
           (‘Friendly’, 2500);
```

14.

```
INSERT INTO PlayerCapturedSpecies(PlayerUsername, SpeciesID, CapturedDate)
VALUES (‘B0b’, 0008, ‘2019-03-23’),
           (‘J@son’, 0010, ‘2019-04-04’),
           (‘St4phan’, 0007, ‘2021-11-14’),
           (‘B0b’, 0001, ‘2020-04-23’),
           (‘J@son’, 0002, ‘2021-05-05’),
           (‘St4phan’, 0003, ‘2021-12-14’),
           (‘J3ssica’, 0009, ‘2019-02-26’),
           (‘R4chel’,  0006, ‘2019-06-18’);
```

15.

```
INSERT INTO PlayerVisitedPokestop(PlayerUsername, PokestopCountry,
PokestopPostalCode, PokestopName, VisitedDate)
VALUES (‘B0b’, ‘USA’, ‘NM 87111’, ‘White Residence’, ‘2022-12-02’),
           (‘J@son’, ‘USA’, ‘NM 87111’, ‘White Residence’, ‘2023-09-06’),
           (‘St4phan’, ‘Canada’, ‘K1A 0A6’, ‘House of Commons’, ‘2020-04-05’),
           (‘J3ssica’, ‘Canada’, ‘V6T 1Z4’, ‘UBC Science’, ‘2023-10-19’),
           (‘R4chel’, ‘UK’, ‘SW1A 1BQ’, ‘Buckingham Palace’, ‘BuckinghamPalace’,
‘2019-01-01’);
```

16.

```
INSERT INTO NPCAppearedAtPokestop(NPCName, PokestopCountry,
PokestopPostalCode, PokestopName)
VALUES (‘Male Grunt’, ‘Canada’, ‘V6T 1Z4’, ‘UBC Science’),
           (‘Male Grunt’, ‘Canada’, ‘K1A 0A6’, ‘House of Commons’),
           (‘Arlo’, ‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’),
           (‘Balloon Grunt’, ‘USA’, ‘NM 87111’, ‘White Residence’),
           (‘Professor Oak’, ‘UK’, ‘SW1A 1BQ’, ‘Buckingham Palace’);
```

17.

```
INSERT INTO NPCSighting(NPCName, PokestopCountry, PokestopPostalCode,
PokestopName, SightingDate, EventName)
VALUES (‘Male Grunt’, ‘Canada’, ‘V6T 1Z4’, ‘UBC Science’, ‘2023-10-11’, ‘Default’),
            (‘Male Grunt’, ‘Canada’, ‘K1A 0A6’, ‘House of Commons’, ‘2018-10-30’,
‘Halloween 2018’),
            (‘Arlo’, ‘Canada’, ‘V0N 1B4’, ‘Blackcomb Guest Services’, ‘2023-07-22’, ‘Go Fest
2023 Fascinating Facets’),
            (‘Balloon Grunt’, ‘USA’, ‘NM 87111’, ‘White Residence’, ‘2021-02-20’,
‘All-in-One 151’),
            (‘Professor Oak’, ‘UK’, ‘SW1A 1BQ’, ‘Buckingham Palace’, ‘2023-11-04’, ‘City
Safari 2023’);

INSERT INTO NPCSightingEventName(EventName, XP, Duration)
VALUES (‘Default’, 100, 60),
            (‘Halloween 2018’, 1000, 48),
            (‘Go Fest 2023 Fascinating Facets’, 2023, 48),
            (‘All-in-one’, 250, 48),
            (‘City Safari 2023’, 2023, 48);
```