**Projeto de Banco de Dados 1**

**Integrantes:**

* Andrew Matheus Barros da Silva
* Brenno Cezário de Oliveira Pereira
* Luisa Monteiro e Silva

**Tema:**

**Jogo de Pokémon**

Abaixo segue o mini mundo, modelo conceitual, modelo lógico, restrições semânticas, avaliação de qualidade, modelo físico (SQL), as consultas em SQL e em álgebra relacional, inserção de dados com e sem erro e, por fim, remoção de dado.

**Mini Mundo**

Em Pokémon, o jogador assume o papel de um treinador e embarca em uma jornada emocionante pelo mundo dos pokémon. O treinador possui uma equipe de pokémons chamada "party", que consiste em até seis pokémons ativos. Além disso, o jogador tem espaços de armazenamento adicionais chamados de "boxes", onde pode guardar pokémons extras. As boxes têm capacidade para até 150 pokémons, divididos em cinco blocos de 30 espaços cada.  
 Cada jogador é único e possui características pessoais. Eles têm um nome, sexo, número de identificação, insígnias conquistadas, dinheiro guardado e tempo de jogo. Todo jogador começa com pelo menos um pokémon em sua party. A party é composta pelos pokémons mais fortes e habilidosos do treinador, e são usados para batalhas contra outros treinadores e desafios.

Os pokémon são as principais criaturas do mundo Pokémon. Cada um deles possui seu próprio nome, sexo, número na Pokédex, número de identificação único, pontos de vida, pontos de experiência e um status de captura. Os pokémon também possuem ataques, podendo aprender até quatro ataques diferentes ao longo de sua jornada. Cada pokémon tem um nível que varia de 1 a 100, indicando seu poder e habilidades. Além disso, eles possuem uma tipagem, que determina seus pontos fortes e fracos em relação a outros tipos de pokémon. Um pokémon pode pertencer ou não a um jogador específico ou até mesmo a um NPC.

NPCs são personagens não controlados pelo jogador, que interagem com o treinador ao longo de sua jornada. Cada NPC tem um nome e um número de identificação único. Além disso, cada NPC possui sua própria party de pokémons, que podem ser desafiados em batalhas e usados como uma forma de elevar o nível dos pokémons do jogador.

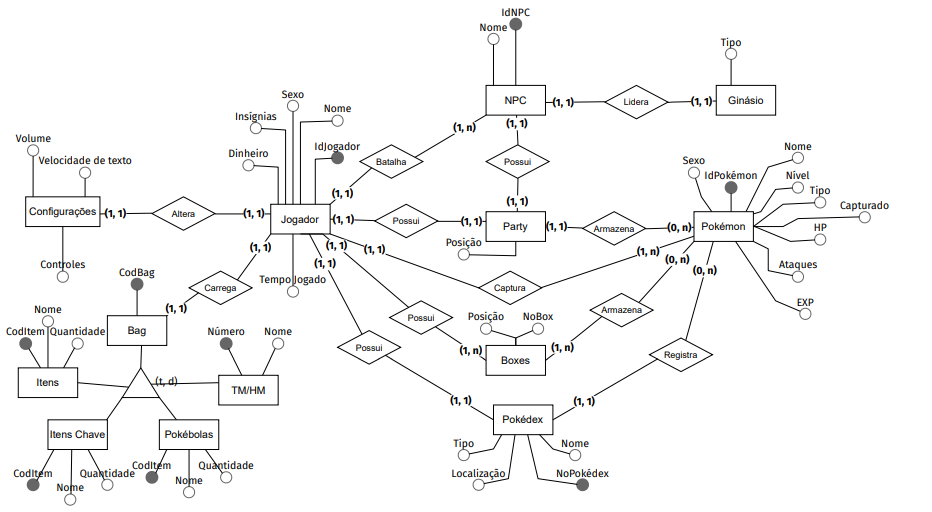
Existem também os Ginásios, locais especiais onde os treinadores podem testar suas habilidades e conquistar insígnias. Cada ginásio possui um tipo de pokémon predominante e é comandado por um NPC diferente. Para conquistar a insígnia de um ginásio, o treinador deve vencer o líder do ginásio em uma batalha.

Além disso tudo, o jogador possui uma mochila (bag) onde pode armazenar itens essenciais para sua jornada. A mochila é subdividida em várias categorias, incluindo itens, itens-chave, pokébolas e TMs/HMs. Os itens são recursos úteis, como poções de cura e pokébolas adicionais. Os itens-chave são objetos especiais para avançar na história do jogo. As pokébolas são usadas para capturar pokémon selvagens, enquanto as TMs/HMs são técnicas e habilidades que podem ser ensinadas aos pokémon.

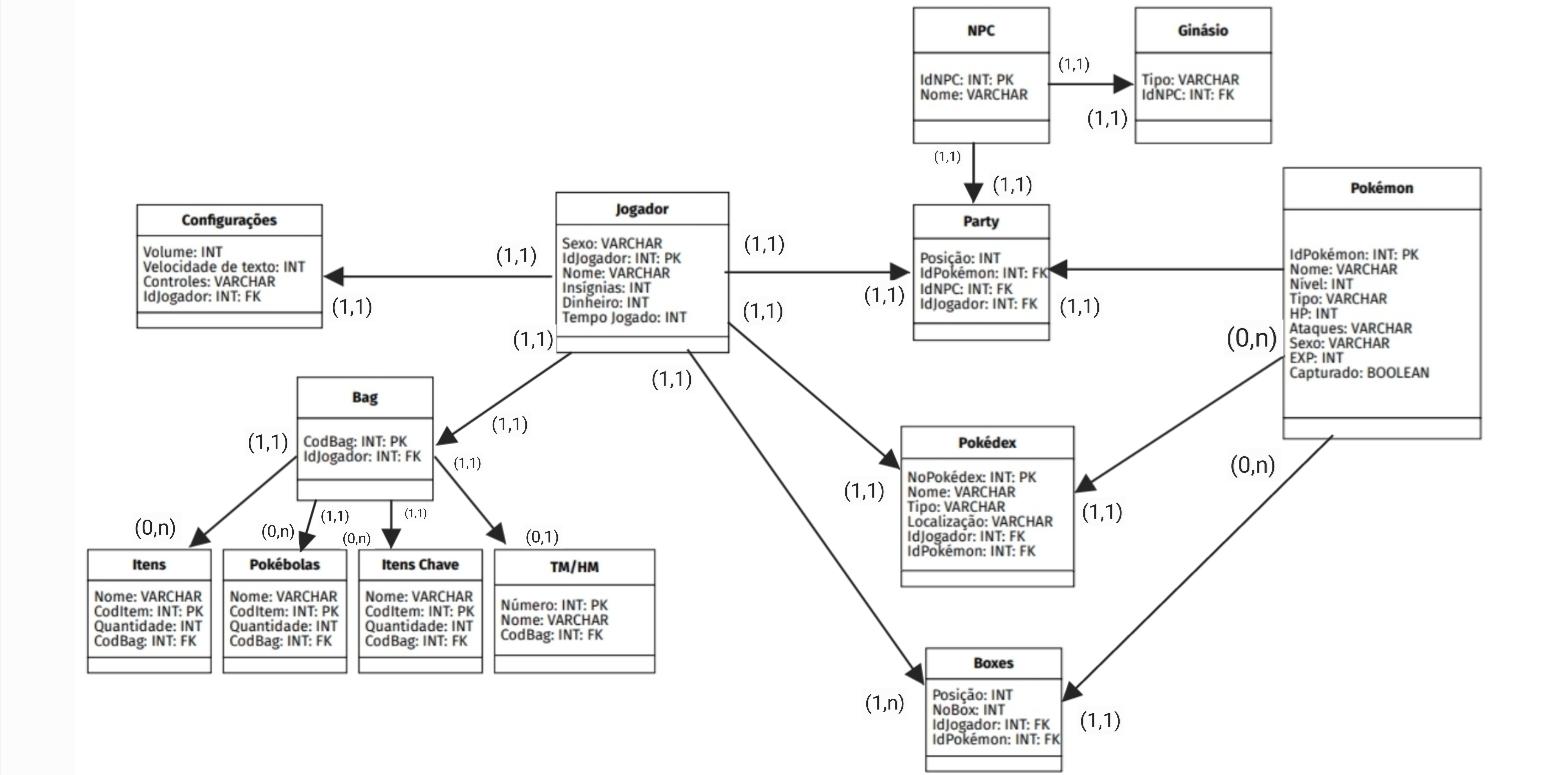
Uma parte essencial do jogo é a Pokédex, que é um dispositivo que registra e fornece informações sobre os pokémons encontrados durante a jornada. Cada entrada na Pokédex contém o nome do pokémon, seu número na Pokédex, tipo e a localização onde ele foi encontrado. A Pokédex permite ao jogador acompanhar sua progressão na captura e descobrir novos pokémons ao longo do caminho.

E, por fim, o jogador pode personalizar sua experiência de jogo ajustando as configurações. Isso inclui opções como volume de áudio, velocidade do texto e configurações de controles. Essas configurações permitem que o jogador adapte a jogabilidade de acordo com suas preferências pessoais.

**Modelo Conceitual**

**￼**

**Modelo Lógico**



**Restrições Semânticas**

**Tabela Bag:**  
 a) Restrição de Domínio:

i) As colunas CodBag e IdJogador precisam ser do tipo Inteiro porque aceitam apenas números

b) Restrição de integridade referencial:

i) A coluna IdJogador é chave estrangeira porque corresponde a um valor válido na coluna IdJogador da tabela Jogador. Isso garante que ela está vinculada a um Jogador já existente no sistema

**Tabela Boxes:**  
 a) Restrição de Domínio:

i) As colunas IdJogador, IdPokémon, NoBox e Posição precisam ser do tipo Inteiro porque aceitam apenas números

b) Restrição de integridade referencial:

i) As colunas IdJogador e IdPokemon precisam ser chaves estrangeiras porque corresponde a valores válidos na coluna IdJogador da tabela Jogador e idPokemon da tabela Pokemon.

**Tabela Configurações:**  
 a) Restrição de Domínio:

i) As colunas IdJogador, Velocidade de texto e Volume precisam ser do tipo Inteiro porque aceitam apenas números

b) Restrição de integridade referencial:

i) A coluna IdJogador precisa ser chave estrangeira porque corresponde a valor válido na coluna IdJogador da tabela Jogador.

**Tabela Ginásio:**  
 a) Restrição de Domínio:

i) A coluna Tipo precisa ser VarChar porque aceita número ou letra.

b) Restrição de integridade referencial:

i) A coluna IdNPC precisa ser chave estrangeira porque corresponde a valor válido na coluna IdNPC da tabela NPC.

**Tabela Itens:**  
 a) Restrição de Domínio:

i) As colunas CodBag, CodItem e Quantidade precisam ser do tipo Inteiro porque aceitam apenas números

b) Restrição de integridade referencial:

i) A coluna CodBag precisa ser chave estrangeira porque corresponde a valor válido na coluna CodBag da tabela Bag.

**Tabela Itens Chave:**  
 a) Restrição de Domínio:

i) As colunas CodBag, CodItem e Quantidade precisam ser do tipo Inteiro porque aceitam apenas números

b) Restrição de integridade referencial:

i) A coluna CodBag precisa ser chave estrangeira porque corresponde a valor válido na coluna CodBag da tabela Bag

**Tabela Jogador:**  
 a) Restrição de Domínio:

i) A colunas Dinheiro e TempoJogado precisam ser do tipo Inteiro porque aceitam apenas números.

**Tabela NPC:**  
 a) Restrição de Domínio:

i) A coluna Nome precisa ser VarChar porque aceita número ou letra.

**Tabela Party:**  
 a) Restrição de Domínio:

i) As colunas IdPokémon e Posição precisam ser do tipo Inteiro porque aceitam apenas números.

b) Restrição de integridade referencial:

i) A coluna IdPokémon precisa ser chave estrangeira porque corresponde a valor válido na coluna IdPokémon da tabela Pokémon.

**Tabela Pokebolas:**  
 a) Restrição de Domínio:

i) As colunas Nome precisa ser do tipo Inteiro porque aceitam números ou letras.

b) Restrição de integridade referencial:

i) A coluna CodBag precisa ser chave estrangeira porque corresponde a valor válido na coluna CodBag da tabela Bag.

**Tabela Pokedex:**  
 a) Restrição de Domínio:

i) Nome e Localização precisam ser VarChar porque aceitam número ou letra.

b) Restrição de integridade referencial:

i) A coluna IdJogador precisa ser chave estrangeira porque corresponde a valor válido na coluna IdJogador da tabela Jogador.

**Tabela Pokémon:**  
 a) Restrição de Domínio:

i) A coluna Capturado precisa ser do tipo BOOLEAN porque pode existir apenas um de dois valores possíveis.

**Tabela TM/HM:**  
 a) Restrição de Domínio:

i) As colunas Número e CodBag precisam ser do tipo Inteiro porque aceita apenas número

b) Restrição de integridade referencial:

i) A coluna CodBag precisa ser chave estrangeira porque corresponde a valor válido na coluna CodBag da tabela Bag

**Avaliação de Qualidade**

**Avaliando a tabela Jogador, temos:**  
  
Na tabela Configurações:

1. Dependências Funcionais:
   1. idJogador -> sexo, nome, insignias, dinheiro e tempo jogado
2. Formas normais:
   1. Terceira Forma Normal (3NF). Por não haver dependências parciais ou transitivas nas colunas da tabela (Jogador). A chave primária, ou Primary Key (PK), identifica de forma única cada registro, e os outros atributos são dependentes diretamente deste identificador.

Em síntese, a tabela Jogador não apresenta dependências indesejadas.

**Modelo Físico (SQL)**

| **CREATE TABLE Pokemon**  **(**  **IdPokémon int**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **Nivel int NOT NULL,**  **Tipo varchar(100) NOT NULL,**  **HP int,**  **EXP int,**  **Ataques varchar(100) NOT NULL,**  **Sexo varchar(100),**  **Capturado boolean NOT NULL**  **);**  **CREATE TABLE Jogador**  **(**  **IdJogador int**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **Sexo varchar(100) NOT NULL,**  **Dinheiro int,**  **TempoJogado int,**  **Insígnias int**  **);**  **CREATE TABLE Pokedex**  **(**  **NoPokedex int NOT NULL**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **Tipo varchar(100) NOT NULL,**  **IdPokémon int NOT NULL,**  **CONSTRAINT Pokedex\_Pokemon\_IdPokémon\_fk**  **FOREIGN KEY (IdPokémon) REFERENCES Pokemon (IdPokémon),**  **IdJogador int,**  **CONSTRAINT Pokedex\_Jogador\_IdJogador\_fk**  **FOREIGN KEY (IdJogador) REFERENCES Jogador (IdJogador),**  **Localização varchar(100)**  **);**  **CREATE TABLE Configurações**  **(**  **Volume int NOT NULL,**  **Controles varchar(100) NOT NULL,**  **VelocidadeTexto int NOT NULL,**  **IdJogador int NOT NULL,**  **CONSTRAINT Configurações\_Jogador\_IdJogador\_fk**  **FOREIGN KEY (IdJogador) REFERENCES Jogador (IdJogador)**  **);**  **CREATE TABLE Bag**  **(**  **CodBag int NOT NULL**  **PRIMARY KEY,**  **IdJogador int NOT NULL,**  **CONSTRAINT Bag\_Jogador\_IdJogador\_fk**  **FOREIGN KEY (IdJogador) REFERENCES Jogador (IdJogador)**  **);**  **CREATE TABLE Itens**  **(**  **CodItem int NOT NULL**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **Quantidade int NOT NULL ,**  **CodBag int NOT NULL,**  **CONSTRAINT Itens\_Bag\_CodBag\_fk**  **FOREIGN KEY (CodBag) REFERENCES Bag (CodBag)**  **);**  **CREATE TABLE Pokebolas**  **(**  **CodItem int NOT NULL**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **Quantidade int NOT NULL ,**  **CodBag int NOT NULL,**  **CONSTRAINT Itens\_Bag\_CodBag\_fk**  **FOREIGN KEY (CodBag) REFERENCES Bag (CodBag)**  **);**  **CREATE TABLE Itens\_Chave**  **(**  **CodItem int NOT NULL**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **Quantidade int NOT NULL ,**  **CodBag int NOT NULL,**  **CONSTRAINT Itens\_Bag\_CodBag\_fk**  **FOREIGN KEY (CodBag) REFERENCES Bag (CodBag)**  **);**  **CREATE TABLE TM\_HM**  **(**  **Numero int NOT NULL**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL,**  **CodBag int NOT NULL,**  **CONSTRAINT Itens\_Bag\_CodBag\_fk**  **FOREIGN KEY (CodBag) REFERENCES Bag (CodBag)**  **);**  **CREATE TABLE Boxes**  **(**  **IdJogador int,**  **CONSTRAINT Boxes\_Jogador\_IdJogador\_fk**  **FOREIGN KEY (IdJogador) REFERENCES Jogador (IdJogador),**  **NoBox int,**  **Posição int,**  **IdPokémon int NOT NULL,**  **CONSTRAINT Boxes\_Pokemon\_IdPokémon\_fk**  **FOREIGN KEY (IdPokémon) REFERENCES Pokemon (IdPokémon)**  **);**  **CREATE TABLE Party**  **(**  **Posição int NOT NULL,**  **IdPokémon int NOT NULL,**  **CONSTRAINT Party\_Pokemon\_IdPokemon\_fk**  **FOREIGN KEY (IdPokémon) REFERENCES Pokemon (IdPokémon)**  **IdJogador int,**  **CONSTRAINT Party\_Jogador\_IdJogador\_fk**  **FOREIGN KEY (IdJogador) REFERENCES Jogador (IdJogador)**  **IdNPC int,**  **CONSTRAINT Party\_NPC\_IdNPC\_fk**  **FOREIGN KEY (IdNPC) REFERENCES Pokemon (IdNPC)**  **);**  **CREATE TABLE NPC**  **(**  **IdNPC int NOT NULL**  **PRIMARY KEY,**  **Nome varchar(100) NOT NULL**  **);**  **CREATE TABLE Ginasio**  **(**  **Tipo varchar(100) NOT NULL**  **PRIMARY KEY,**  **IdNPC int,**  **CONSTRAINT Ginasio\_NPC\_IdNPC\_fk**  **FOREIGN KEY (IdNPC) REFERENCES NPC (IdNPC)**  **);** |
| --- |

**Consultas(SQL)**

**1. Todos os itens com quantidade menor ou igual a 10.**

| SELECT \*  FROM Bag  WHERE Quantidade <= 10; |
| --- |

**2. O nome e os níveis de todos os pokémon da equipe de batalha do jogador.**

| SELECT j.Nome AS Jogador, po.Nome AS Pokemon, po.Nivel  FROM Jogador j  JOIN Party pa ON j.IdJogador = pa.IdJogador  JOIN Pokemon po ON pa.IdPokémon = po.IdPokémon; |
| --- |

**3. Número de pokémon capturados conforme cada sexo.**

| SELECT Sexo, COUNT(\*) AS TotalCapturados  FROM Pokemon  WHERE Capturado = true  GROUP BY Sexo; |
| --- |

**4. As médias dos níveis dos pokémon de cada tipo, mas apenas para os grupos que possuem uma média de nível maior que 15.**

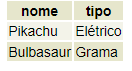
| SELECT Tipo, AVG(Nivel) AS MediaNivel  FROM Pokemon  GROUP BY Tipo  HAVING AVG(Nivel) > 15; |
| --- |

**Consultas (Álgebra Relacional)**

**Consulta 1:**

(**σ**(idjogador=1)**AND**(idnpc=NULL)(party))**⋃**(**∏**(nome, tipo)(pokemon))

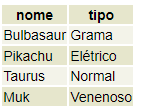
Essa consulta é realizada com o objetivo de projetar o nome e tipo do(s) pokémon que estão alocados na “party” do jogador com idjogador = 1.



**Consulta 2:**

(**∏**(nome, tipo) (pokemon)) - (**σ**(idjogador = 1)(pokedex))

Essa consulta é realizada com o objetivo de projetar o nome e tipo do(s) pokémon que o jogador com idjogador = 1 registrou em sua pokédex.



**Inserção de dados (com erro)**

Sendo as tabelas escolhidas NPC e Party, a tabela Party, possui uma coluna IdNPC que é chave estrangeira que referencia a coluna IdNPC da tabela NPC e a tabela Party possui uma coluna IdPokemon que é chave estrangeira que referencia a coluna IdPokemon da tabela Pokémon.

Utilizando o comando na linguagem SQL para inserir ou realizar atualização na tabela NPC, temos:

INSERT INTO Party (

Posição, idPokémon, idNPC

)

VALUES (

1, 1605, 3

);

Portanto, tenta-se inserir uma nova Party com sua posição igual a 1, idPokémon igual a 1605 e IdNPC igual a 3. Entretanto, o valor 3 para IdNPC não existe na tabela NPC e por conta disso, viola a Foreign Key (Chave estrangeira) e com isso, o comando além de falhar, aponta um erro indicando que a FK não foi encontrada na tabela de referência.

**Inserção de dados (com sucesso)**

Escolhendo a tabela Pokebolas, que possui as colunas Nome, CodItem, Quantidade e CodBag, essa última é a chave estrangeira que referencia a coluna CodBag da tabela Bag.  
  
Diante disso, temos o comando em SQL:

INSERT INTO Pokebolas

(

Nome, CodItem, Quantidade, CodBag

)

VALUES

(

'Ultraball', 7, 2, 10

);

O objetivo desse comando é inserir um registro na tabela `Bag`, garantindo que o valor fornecido para CodBag exista em alguma outra tabela referenciada (Bag). Neste exemplo, suponhamos que o valor 10 para CodBag já exista em uma tabela referenciada que é Bag.  
 Logo, ao executar o comando, ele tentará inserir um novo registro na tabela Pokebolas com CodBag igual a 10. Caso a chave estrangeira seja encontrada na tabela referenciada, o comando será executado com sucesso, inserindo os dados na tabela Bag.

**Remoção de dados**

Considerando a tabela Configurações para excluir registros dela com base em uma determinada condição, o comando WHERE será utilizado para especificar a condição que deverá ser atendida para que os registros sejam removidos - a condição pode ser baseada em uma ou mais colunas da tabela.  
  
DELETE FROM tabela

WHERE condição;

Para remover um jogador e suas configurações correspondentes na tabela Configurações, utiliza-se o comando em SQL:

DELETE FROM Configurações

WHERE idjogador = 1;

Diante disso, todos os registros da tabela Configurações onde o valor da coluna idjogador é igual a 1 estão sendo removidos.

Após executar o comando, os registros que atendem à condição serão removidos da tabela Configurações

**Link da pasta com todos os arquivos usados para a montagem do projeto, desde o esquema inicial à finalização do banco:**

[Banco de Dados 1](https://drive.google.com/drive/folders/1LzwJ2uze6jaEaRd201Gofj03kw8KAl-6?usp=drive_link)

**Link para o SGBD da UERJ:**

[SGBD - UERJ](http://labdb.ime.uerj.br/phppgadmin/redirect.php)