

## 1. Pokémon:

- Esta es la entidad central del tema. Cada Pokémon tiene características específicas que lo hacen único.
- **Atributos:**
  - Nombre
  - Número de Pokédex
  - Tipo (puede ser uno o dos)
  - Habilidades
  - Evolución (si aplica)
  - Estadísticas base (Ataque, Defensa, Velocidad, etc.)
  - Región de origen
  - Generación

## 2. Entrenadores:

- Los entrenadores son los personajes que capturan y entrenan a los Pokémon.
- **Atributos:**
  - Nombre
  - Edad
  - Ciudad/Región de origen
  - Medallas obtenidas
  - Equipo Pokémon (relación con la entidad Pokémon)

## 3. Tipos:

- Los **tipos** de Pokémon son fundamentales en sus características, pues definen sus fortalezas y debilidades.
- **Atributos:**
  - Nombre del tipo (por ejemplo, Fuego, Agua, Eléctrico)
  - Fortalezas contra otros tipos
  - Debilidades contra otros tipos

## 4. Habilidades:

- Las habilidades son características especiales que pueden tener los Pokémon.
- **Atributos:**
  - Nombre de la habilidad
  - Descripción del efecto
  - Pokémon que pueden tener esa habilidad (relación con Pokémon)

## 5. Movimientos:

- Los movimientos son las técnicas que un Pokémon puede aprender para combatir.
- **Atributos:**

- Nombre del movimiento
- Tipo de movimiento (físico, especial, estado)
- Potencia
- Precisión
- Puntos de Poder (PP)
- Tipo (relacionado con la entidad Tipos)
- Pokémon que pueden aprender ese movimiento (relación con Pokémon)

## 6. Regiones:

- Las regiones definen las áreas del mundo Pokémon. Cada región tiene su propia Pokédex y entrenadores.
- **Atributos:**
  - Nombre de la región
  - Lista de Pokémon nativos (relación con Pokémon)
  - Ciudades/Gimnasios

## 7. Gimnasios:

- Cada gimnasio es un lugar donde los entrenadores pueden luchar contra líderes de gimnasio para ganar medallas.
- **Atributos:**
  - Nombre del gimnasio
  - Ciudad donde está ubicado (relación con la entidad Regiones)
  - Líder del gimnasio (relación con entrenadores)
  - Tipo de gimnasio (relación con la entidad Tipos)
  - Medalla que otorga

## 8. Evoluciones:

- La evolución es el proceso por el cual un Pokémon cambia a otra forma más avanzada.
- **Atributos:**
  - Pokémon de origen
  - Pokémon evolucionado
  - Nivel o condición necesaria para la evolución (nivel, piedra evolutiva, amistad, etc.)

## 9. Objetos (Items):

- Los objetos son elementos que pueden utilizarse en combate o para mejorar a los Pokémon.
- **Atributos:**
  - Nombre del objeto
  - Tipo (poción, baya, piedra evolutiva, etc.)
  - Efecto
  - Pokémon que puede utilizar el objeto (si aplica)

## 10. Películas:

- Las películas hechas
- **Atributos:**
  - Nombre de la película
  - Genero
  - Duración
  - Personaje(s) Principal(es)

-----  
}por favor hazme la tabla en sql

Aquí tienes la estructura de la tabla **Entrenadores** en SQL, junto con la relación de **EquipoPokemon** que enlaza a la tabla de **Pokémon**:

sql

Copiar código

```
-- Tabla de Entrenadores
CREATE TABLE Entrenadores (
    entrenador_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para
cada entrenador
    nombre VARCHAR(100) NOT NULL, -- Nombre del entrenador
    edad INT NOT NULL, -- Edad del entrenador
    ciudad_origen VARCHAR(100), -- Ciudad o región de origen
    medallas_obtenidas INT DEFAULT 0 -- Número de medallas
obtenidas
);

-- Tabla de Pokémon
CREATE TABLE Pokemon (
    pokemon_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para
cada Pokémon
    nombre VARCHAR(100) NOT NULL, -- Nombre del Pokémon
    tipo VARCHAR(50) NOT NULL, -- Tipo de Pokémon (e.g., Fuego,
Agua)
    nivel INT NOT NULL, -- Nivel del Pokémon
    entrenador_id INT, -- Relación con el entrenador
FOREIGN KEY (entrenador_id) REFERENCES
Entrenadores(entrenador_id) -- Clave foránea que relaciona con
Entrenadores
);
```

## Explicación:

- **Entrenadores:** Esta tabla almacena la información básica del entrenador, como nombre, edad, ciudad de origen y medallas obtenidas.
- **Pokémon:** Aquí se almacenan los Pokémon capturados por los entrenadores. Cada Pokémon tiene un campo `entrenador_id` que indica qué entrenador lo posee. Esto crea una relación de uno a muchos entre entrenadores y Pokémon (un entrenador puede tener varios Pokémon).

Este diseño te permite hacer consultas para obtener el equipo Pokémon de un entrenador o detalles sobre un Pokémon específico.

40

### Tú dijiste:

y ahora este 3. Tipos: Los tipos de Pokémon son fundamentales en sus características, pues definen sus fortalezas y debilidades. Atributos: Nombre del tipo (por ejemplo, Fuego, Agua, Eléctrico) Fortalezas contra otros tipos Debilidades contra otros tipos

Aquí tienes cómo podrías estructurar la entidad **Tipos** en SQL, teniendo en cuenta las fortalezas y debilidades de cada tipo de Pokémon. El diseño incluye una tabla principal para los tipos y tablas relacionadas para las fortalezas y debilidades.

## Estructura de la tabla **Tipos** en SQL:

sql

Copiar código

```
-- Tabla de Tipos
CREATE TABLE Tipos (
    tipo_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada
    tipo
    nombre_tipo VARCHAR(50) NOT NULL -- Nombre del tipo (e.g.,
    Fuego, Agua, Eléctrico)
);

-- Tabla de Fortalezas entre tipos
CREATE TABLE Fortalezas (
    tipo_id INT, -- Tipo que tiene fortalezas
    tipo_fuerte_contra INT, -- Tipo contra el cual es fuerte
    PRIMARY KEY (tipo_id, tipo_fuerte_contra), -- Clave primaria
    compuesta
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id), -- Clave
    foránea que referencia a Tipos
    FOREIGN KEY (tipo_fuerte_contra) REFERENCES Tipos(tipo_id) --
    Clave foránea que referencia a Tipos (contra quién es fuerte)
```

```
);

-- Tabla de Debilidades entre tipos
CREATE TABLE Debilidades (
    tipo_id INT, -- Tipo que tiene debilidades
    tipo_debil_contra INT, -- Tipo contra el cual es débil
    PRIMARY KEY (tipo_id, tipo_debil_contra), -- Clave primaria
    compuesta
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id), -- Clave
    foránea que referencia a Tipos
    FOREIGN KEY (tipo_debil_contra) REFERENCES Tipos(tipo_id) --
    Clave foránea que referencia a Tipos (contra quién es débil)
);
```

### Explicación:

- **Tipos:** Esta tabla guarda el nombre de cada tipo de Pokémon (e.g., Fuego, Agua, Eléctrico).
- **Fortalezas:** Relaciona un tipo con otros tipos contra los que es fuerte. Por ejemplo, "Agua" sería fuerte contra "Fuego".
- **Debilidades:** Relaciona un tipo con otros tipos contra los que es débil. Por ejemplo, "Fuego" sería débil contra "Agua".

Esto te permitirá definir las fortalezas y debilidades de cada tipo en relación con otros tipos, utilizando relaciones entre los registros de la tabla **Tipos**.

4o

### Tú dijiste:

4. Habilidades: Las habilidades son características especiales que pueden tener los Pokémon. Atributos: Nombre de la habilidad Descripción del efecto Pokémon que pueden tener esa habilidad (relación con Pokémon)

Aquí te dejo la estructura de la entidad **Habilidades** en SQL, con una relación hacia los Pokémon que pueden tener cada habilidad:

### Estructura de la tabla **Habilidades** en SQL:

sql

Copiar código

```
-- Tabla de Habilidades
CREATE TABLE Habilidades (
    habilidad_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para
    cada habilidad
```

```

        nombre_habilidad VARCHAR(100) NOT NULL, -- Nombre de la
habilidad
        descripcion_efecto TEXT NOT NULL -- Descripción del efecto de
la habilidad
    );

-- Tabla de relación entre Habilidades y Pokémon
CREATE TABLE Pokemon_Habilidades (
    pokemon_id INT, -- ID del Pokémon que tiene la habilidad
    habilidad_id INT, -- ID de la habilidad
    PRIMARY KEY (pokemon_id, habilidad_id), -- Clave primaria
compuesta por el Pokémon y la habilidad
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(pokemon_id), --
Clave foránea que referencia a Pokémon
    FOREIGN KEY (habilidad_id) REFERENCES Habilidades(habilidad_id)
-- Clave foránea que referencia a Habilidades
);

```

### Explicación:

- **Habilidades:** Esta tabla guarda el nombre y la descripción de cada habilidad que puede tener un Pokémon. Ejemplo: "Intimidación", con una descripción que diga "Reduce el ataque del oponente".
- **Pokemon\_Habilidades:** Esta tabla crea la relación muchos a muchos entre los Pokémon y las habilidades, ya que un Pokémon puede tener múltiples habilidades y una habilidad puede pertenecer a varios Pokémon.

Este diseño te permitirá consultar qué habilidades tiene un Pokémon específico, o qué Pokémon pueden poseer una habilidad determinada.

40

### Tú dijiste:

5. Movimientos: Los movimientos son las técnicas que un Pokémon puede aprender para combatir. Atributos: Nombre del movimiento Tipo de movimiento (físico, especial, estado) Potencia Precisión Puntos de Poder (PP) Tipo (relacionado con la entidad Tipos) Pokémon que pueden aprender ese movimiento (relación con Pokémon)

Aquí tienes la estructura para la entidad **Movimientos** en SQL, con sus atributos y relaciones con las tablas de **Tipos** y **Pokémon**:

### Estructura de la tabla **Movimientos** en SQL:

sql

Copiar código

```
-- Tabla de Movimientos
CREATE TABLE Movimientos (
    movimiento_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para
cada movimiento
    nombre_movimiento VARCHAR(100) NOT NULL, -- Nombre del
movimiento
    tipo_movimiento VARCHAR(50) NOT NULL, -- Tipo de movimiento
(físico, especial, estado)
    potencia INT, -- Potencia del movimiento (puede ser NULL si no
aplica)
    precision INT, -- Precisión del movimiento (en porcentaje)
    puntos_poder INT NOT NULL, -- Puntos de poder (PP)
    tipo_id INT, -- Relación con la entidad Tipos (fuego, agua,
eléctrico, etc.)
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id) -- Clave
foránea que relaciona con Tipos
);

-- Tabla de relación entre Movimientos y Pokémon
CREATE TABLE Pokemon_Movimientos (
    pokemon_id INT, -- ID del Pokémon que puede aprender el
movimiento
    movimiento_id INT, -- ID del movimiento
    PRIMARY KEY (pokemon_id, movimiento_id), -- Clave primaria
compuesta por el Pokémon y el movimiento
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(pokemon_id), --
Clave foránea que referencia a Pokémon
    FOREIGN KEY (movimiento_id) REFERENCES
Movimientos(movimiento_id) -- Clave foránea que referencia a
Movimientos
);
```

## Explicación:

- **Movimientos:** Esta tabla almacena los detalles de los movimientos, incluyendo el nombre, tipo (físico, especial, estado), potencia (si aplica), precisión (en porcentaje), PP (puntos de poder), y el tipo de movimiento (relacionado con la tabla de **Tipos**).
- **Pokemon\_Movimientos:** Esta tabla establece la relación muchos a muchos entre Pokémon y movimientos, ya que un Pokémon puede aprender varios movimientos, y un movimiento puede ser aprendido por múltiples Pokémon.

Con este diseño puedes hacer consultas para encontrar qué movimientos puede aprender un Pokémon o cuáles son los Pokémon que pueden aprender un movimiento específico.

40

**Tú dijiste:**

6. Regiones: Las regiones definen las áreas del mundo Pokémon. Cada región tiene su propia Pokédex y entrenadores. Atributos: Nombre de la región Lista de Pokémon nativos (relación con Pokémon) Ciudades/Gimnasios

**ChatGPT Plus**

ChatGPT

Aquí tienes la estructura para la entidad **Regiones** en SQL, incluyendo la relación con **Pokémon** nativos y las **Ciudades/Gimnasios** que pertenecen a cada región.

### Estructura de la tabla **Regiones** en SQL:

sql

Copiar código

-- Tabla de Regiones

```
CREATE TABLE Regiones (  
    region_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada  
    región  
    nombre_region VARCHAR(100) NOT NULL -- Nombre de la región  
(e.g., Kanto, Johto, Hoenn)  
);
```

-- Tabla de relación entre Pokémon y Regiones (Pokémon nativos)

```
CREATE TABLE Pokemon_Regiones (  
    pokemon_id INT, -- ID del Pokémon nativo de la región  
    region_id INT, -- ID de la región  
    PRIMARY KEY (pokemon_id, region_id), -- Clave primaria  
    compuesta  
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(pokemon_id), --  
    Clave foránea que referencia a Pokémon  
    FOREIGN KEY (region_id) REFERENCES Regiones(region_id) -- Clave  
    foránea que referencia a Regiones  
);
```

-- Tabla de Ciudades/Gimnasios en cada región

```
CREATE TABLE Ciudades (  
    ciudad_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada  
    ciudad
```



```

    nombre_ciudad VARCHAR(100) NOT NULL, -- Nombre de la ciudad o
gimnasio
    region_id INT, -- Relación con la región
    FOREIGN KEY (region_id) REFERENCES Regiones(region_id) -- Clave
foránea que referencia a Regiones
);

```

## Explicación:

- **Regiones:** Esta tabla almacena el nombre de cada región en el mundo Pokémon.
- **Pokemon Regiones:** Relaciona a los Pokémon con las regiones donde son nativos. Un Pokémon puede estar en múltiples regiones, y una región puede tener varios Pokémon.
- **Ciudades:** Almacena las ciudades y gimnasios presentes en cada región, asociándose a la tabla **Regiones**.

Este diseño te permitirá hacer consultas sobre qué Pokémon son nativos de una región específica o cuáles son las ciudades y gimnasios de cada región.

4o

## Tú dijiste:

7. Gimnasios: Cada gimnasio es un lugar donde los entrenadores pueden luchar contra líderes de gimnasio para ganar medallas. Atributos: Nombre del gimnasio Ciudad donde está ubicado (relación con la entidad Regiones) Líder del gimnasio (relación con entrenadores) Tipo de gimnasio (relación con la entidad Tipos) Medalla que otorga

Aquí tienes la estructura para la entidad **Gimnasios** en SQL, que incluye las relaciones con **Regiones**, **Entrenadores** y **Tipos**.

## Estructura de la tabla **Gimnasios** en SQL:

sql

Copiar código

```

-- Tabla de Gimnasios
CREATE TABLE Gimnasios (
    gimnasio_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para
cada gimnasio
    nombre_gimnasio VARCHAR(100) NOT NULL, -- Nombre del gimnasio
    ciudad_id INT, -- ID de la ciudad donde está ubicado el
gimnasio
    lider_id INT, -- ID del líder del gimnasio
    tipo_id INT, -- ID del tipo de gimnasio (relacionado con Tipos)

```

```

    medalla_otorgada VARCHAR(100), -- Nombre de la medalla que
otorga
    FOREIGN KEY (ciudad_id) REFERENCES Ciudades(ciudad_id), --
Clave foránea que referencia a Ciudades
    FOREIGN KEY (lider_id) REFERENCES Entrenadores(entrenador_id),
-- Clave foránea que referencia a Entrenadores
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id) -- Clave
foránea que referencia a Tipos
);

```

### Explicación:

- **Gimnasios:** Esta tabla almacena la información sobre cada gimnasio, incluyendo su nombre, la ciudad donde se ubica (relacionada con la tabla de **Ciudades**), el líder del gimnasio (relacionado con la tabla de **Entrenadores**), el tipo de gimnasio (relacionado con la tabla de **Tipos**) y la medalla que otorga al ser derrotado.

Este diseño permite que cada gimnasio esté vinculado a un líder específico, a una ciudad determinada y a un tipo de Pokémon, lo que lo convierte en un lugar único para los entrenadores que buscan ganar medallas y mejorar sus habilidades. Además, puedes realizar consultas para saber qué gimnasio está en qué ciudad, quién es su líder, y qué medalla otorga.

4o mini

### Tú dijiste:

8. Evoluciones: La evolución es el proceso por el cual un Pokémon cambia a otra forma más avanzada. Atributos: Pokémon de origen Pokémon evolucionado Nivel o condición necesaria para la evolución (nivel, piedra evolutiva, amistad, etc.)

Aquí tienes la estructura para la entidad **Evoluciones** en SQL, que incluye la relación entre un Pokémon de origen y su forma evolucionada, así como las condiciones necesarias para la evolución.

### Estructura de la tabla **Evoluciones** en SQL:

sql

Copiar código

```

-- Tabla de Evoluciones
CREATE TABLE Evoluciones (
    evolucion_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para
cada evolución
    pokemon_origen_id INT, -- ID del Pokémon de origen
    pokemon_evolucionado_id INT, -- ID del Pokémon evolucionado

```

```

    condicion_evolucion VARCHAR(100) NOT NULL, -- Condición
necesaria para la evolución (nivel, piedra, amistad, etc.)
    FOREIGN KEY (pokemon_origen_id) REFERENCES Pokemon(pokemon_id),
-- Clave foránea que referencia al Pokémon de origen
    FOREIGN KEY (pokemon_evolucionado_id) REFERENCES
Pokemon(pokemon_id) -- Clave foránea que referencia al Pokémon
evolucionado
);

```

## Explicación:

- **Evoluciones:** Esta tabla almacena las relaciones entre un Pokémon de origen y su forma evolucionada.
  - **pokemon\_origen\_id:** ID del Pokémon que evoluciona.
  - **pokemon\_evolucionado\_id:** ID del Pokémon resultante de la evolución.
  - **condicion\_evolucion:** Describe la condición necesaria para que se produzca la evolución (puede ser un nivel específico, uso de una piedra evolutiva, nivel de amistad, etc.).

Este diseño permite que puedas hacer consultas sobre qué Pokémon evolucionan, en qué se convierten, y qué condiciones se requieren para que se produzca la evolución.

4o mini

### Tú dijiste:

9. Objetos (Items): Los objetos son elementos que pueden utilizarse en combate o para mejorar a los Pokémon. Atributos: Nombre del objeto Tipo (poción, baya, piedra evolutiva, etc.) Efecto Pokémon que puede utilizar el objeto (si aplica)

Aquí tienes la estructura para la entidad **Objetos** (o **Items**) en SQL, que incluye los atributos y las relaciones con los Pokémon que pueden utilizar estos objetos:

## Estructura de la tabla **Objetos** en SQL:

sql

Copiar código

-- Tabla de Objetos

CREATE TABLE Objetos (

objeto\_id INT AUTO\_INCREMENT PRIMARY KEY, -- ID único para cada  
objeto

nombre\_objeto VARCHAR(100) NOT NULL, -- Nombre del objeto  
(e.g., Poción, Baya, Piedra Evolutiva)

```

        tipo_objeto VARCHAR(50) NOT NULL, -- Tipo del objeto (e.g.,
Poción, Baya, Piedra Evolutiva)
        efecto TEXT NOT NULL -- Descripción del efecto del objeto
    );

-- Tabla de relación entre Objetos y Pokémon
CREATE TABLE Pokemon_Objetos (
    pokemon_id INT, -- ID del Pokémon que puede utilizar el objeto
    objeto_id INT, -- ID del objeto
    PRIMARY KEY (pokemon_id, objeto_id), -- Clave primaria
compuesta por el Pokémon y el objeto
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(pokemon_id), --
Clave foránea que referencia a Pokémon
    FOREIGN KEY (objeto_id) REFERENCES Objetos(objeto_id) -- Clave
foránea que referencia a Objetos
);

```

## Explicación:

- **Objetos:** Esta tabla almacena información sobre los diferentes objetos disponibles en el mundo Pokémon, incluyendo:
  - **nombre\_objeto:** Nombre del objeto (por ejemplo, "Poción", "Baya", "Piedra Evolutiva").
  - **tipo\_objeto:** Clasificación del objeto (por ejemplo, "Poción", "Baya", "Piedra Evolutiva").
  - **efecto:** Descripción detallada del efecto que tiene el objeto (por ejemplo, "Restaura 20 puntos de salud").
- **Pokemon\_Objetos:** Esta tabla establece la relación muchos a muchos entre Pokémon y objetos, permitiendo que un Pokémon pueda utilizar múltiples objetos y que un objeto pueda ser utilizado por varios Pokémon.

Con este diseño, puedes realizar consultas para saber qué objetos puede utilizar un Pokémon específico o qué efectos tienen los diferentes objetos disponibles en el juego.

10. Películas: Las películas hechas Atributos: Nombre de la película Genero Duración  
Personaje(s) Principal(es)

Aquí tienes la estructura para la entidad **Películas** en SQL, que incluye los atributos relacionados con las películas de Pokémon:

## Estructura de la tabla **Películas** en SQL:

sql

Copiar código

-- Tabla de Películas

```
CREATE TABLE Peliculas (  
    pelicula_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para  
    cada película  
    nombre_pelicula VARCHAR(100) NOT NULL, -- Nombre de la película  
    genero VARCHAR(50) NOT NULL, -- Género de la película (e.g.,  
    Aventura, Acción, Fantasía)  
    duracion INT NOT NULL, -- Duración de la película en minutos  
    personajes_principales TEXT NOT NULL -- Personaje(s)  
    principal(es) de la película  
);
```

### Explicación:

- **Películas:** Esta tabla almacena información sobre cada película de Pokémon, incluyendo:
  - **nombre\_pelicula:** Nombre de la película (por ejemplo, "Pokémon: La película - Mewtwo contraataca").
  - **género:** Género de la película (por ejemplo, "Aventura", "Acción", "Fantasía").
  - **duracion:** Duración de la película en minutos (por ejemplo, 90 minutos).
  - **personajes\_principales:** Una lista o descripción de los personajes principales que aparecen en la película (puede ser un texto que incluya varios nombres).

Este diseño te permitirá realizar consultas sobre las películas disponibles, sus géneros, duración y personajes principales. Si deseas establecer relaciones adicionales con otros aspectos del universo Pokémon (como personajes en específico), se podría considerar la creación de tablas adicionales.

---

```
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Paleta",1);
```

```

INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Verde",1);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Celeste",1);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Azafrán",1);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Fucsia",1);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Carmín",1);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Celadon",1);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Nuevo",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Chococor",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Olivo",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Ecruteak",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Mahogany",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Iris",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Caoba",2);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo raíz",3);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Mauville",3);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Toho",3);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Pastoria",3);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Sootopolis",3);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Toho",4);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Eterna",4);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Sunyshore",4);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Twinleaf",4);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Jubilife",4);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Nuvema",5);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Accumula",5);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Striaton",5);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Nimbasa",5);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Castelia",5);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Vaniville",6);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Lumiose",6);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Santalune",6);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Couriway",6);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Shalour",6);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Akala",7);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Hau'oli",7);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Akala",7);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Ula'ula",7);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Poni",7);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Wyndon",8);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Motostoke",8);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Hammerlocke",8);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Turffield",8);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Stow-on-Side",8);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Jubilife",9);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de la Luna",9);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de la Isla de la Corriente",9);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de las Sombras",9);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Cascada de las Almas",9);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Phenac",10);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad Pyrite",10);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Gateon",10);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo Agate",10);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de la Colonia",10);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de Kincaid",11);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de Ringtown",11);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de la Bahía",11);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de la Aurora",11);

```

```

INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Almia",11);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Isla Oblivia",12);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de la Isla",12);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Fuego",12);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de la Luz",12);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de las Aguas",12);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Pasio",13);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Isla de la Nieve",13);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Isla de la Luz ",13);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Isla de la Fuerza",13);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Isla de la Velocidad",13);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de Roto",14);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Playa de Roto",14);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Bosque de Roto",14);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Montañas de Roto",14);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Cueva de Roto",14);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de Fiore",15);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Pueblo de Florar",15);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de la Costa",15);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Ciudad de Relieve",15);
INSERT INTO Ciudades (ciudad_id,nombre_ciudad,region_id) VALUES(NULL,"Isla de Fiore",15);

```

---

INSERT INTO Entrenadores

```

(entrenador_id,nombre,edad,ciudad_origen,medallas_obtenidas) VALUES
(NULL,"Ash Ketchum",10,1,8),
(NULL,"Misty (Kasumi)",16,3,2),
(NULL,"Brock (Takeshi)",16,5,1),
(NULL,"Giovanni",30,2,8),
(NULL,"May (Haruka)",12,16,0),
(NULL,"Dawn (Hikari)",10,23,0),
(NULL,"Serena",10,30,0),
(NULL,"Cynthia",25,7,0),
(NULL,"Leon",21,40,1),
(NULL,"Roxie ",16,25,1),
(NULL,"Clair",20,9,1),
(NULL,"Korrina",15,34,1),
(NULL,"Alain",16,31,1),
(NULL,"N",16,28,0),
(NULL,"Lusamine",35,37,0);

```

Aquí tienes la tabla de entrenadores modificada, incluyendo la región de origen de cada uno:

Entrenador	Edad	Ciudad de origen	Medallas obtenidas	Región
Ash Ketchum	10 años	Pueblo Paleta	8	Kanto
Misty (Kasumi)	16 años	Ciudad Celeste	2	Kanto
Brock (Takeshi)	16 años	Ciudad Pewter	1	Kanto

Giovanni	30s	Ciudad Verde	8	Kanto	
May (Haruka)	12 años	Ciudad Mauville	0	Hoenn	
Dawn (Hikari)	10 años	Ciudad Twinleaf	0	Sinnoh	
Serena	10 años	Ciudad Vaniville	0	Kalos	
Cynthia	25 años	Ciudad Celadon	0	Kanto	
Leon	21 años	Ciudad Wyndon	0	Galar	
Roxie	16 años	Ciudad Virbank	1	Unova	
Clair	20 años	Ciudad Blackthorn	1	Johto	
Korrina	15 años	Ciudad Shalour	1	Kalos	
Alain	16 años	Ciudad Lumiose	0	Kalos	
N	16 años	Ciudad Driftveil	0	Unova	
Lusamine	35 años	Ciudad Aether Paradise	0	Alola	

Si necesitas más ajustes o información adicional, ¡dímelo!

```

INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Kanto");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Johto");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Hoenn");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Sinnoh");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Unova");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Kalos");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Alola");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Galar");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Hisui");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Orre");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Almia");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Oblivia");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Pasio");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Roto");
INSERT INTO Regiones (region_id,nombre_region)VALUES(NULL,"Fiore");

```

```

CREATE DATABASE IF NOT EXISTS pokemon;

```

```

USE pokemon;

```

```

-- Tabla de Regiones

```

```

CREATE TABLE IF NOT EXISTS Regiones (
    region_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada región
    nombre_region VARCHAR(100) NOT NULL -- Nombre de la región (e.g., Kanto, Johto, Hoenn)
);

```

```

-- Tabla de Ciudades

```

```

CREATE TABLE IF NOT EXISTS Ciudades (
    ciudad_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada ciudad
    nombre_ciudad VARCHAR(100) NOT NULL, -- Nombre de la ciudad o gimnasio

```



```

    region_id INT, -- Relación con la región
    FOREIGN KEY (region_id) REFERENCES Regiones(region_id) -- Clave foránea que
referencia a Regiones
);

```

-- Tabla de Tipos

```

CREATE TABLE IF NOT EXISTS Tipos (
    tipo_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada tipo
    nombre_tipo VARCHAR(50) NOT NULL -- Nombre del tipo (e.g., Fuego, Agua, Eléctrico)
);

```

-- Tabla de Entrenadores

```

CREATE TABLE IF NOT EXISTS Entrenadores (
    entrenador_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada entrenador
    nombre VARCHAR(100) NOT NULL, -- Nombre del entrenador
    edad INT NOT NULL, -- Edad del entrenador
    ciudad_origen INT, -- Ciudad o región de origen
    medallas_obtenidas INT DEFAULT 0, -- Número de medallas obtenidas
    FOREIGN KEY (ciudad_origen) REFERENCES Ciudades(ciudad_id)
);

```

-- Tabla de Pokemon

```

CREATE TABLE IF NOT EXISTS Pokemon(
    Id_pokemon INT PRIMARY KEY, -- Identificador
    Numero_podex INT NOT NULL, -- Numero en el que está ubicado el
Pokémon en la Pokédex
    ataque_base INT, -- Estadística base de ataque
    defensa_base INT, -- Estadística base de defensa
    velocidad_base INT, -- Estadística base de velocidad
    hp_base INT, -- Estadística base de puntos de salud (HP)
    generacion INT,
    Id_entrenador INT, -- ID del entrenador que posee al Pokémon
    FOREIGN KEY (Id_entrenador) REFERENCES Entrenadores(entrenador_id) -- Clave
foránea que relaciona con Entrenadores
);

```

-- POKEMONES Y TIPOS

```

CREATE TABLE IF NOT EXISTS Pokemon_Tipos(
    pokemon_id INT, -- ID del Pokémon que consume
    tipo_id INT,
    PRIMARY KEY (pokemon_id,tipo_id),
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(Id_pokemon),
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id)
);

```

-- Tabla de Fortalezas

```

CREATE TABLE IF NOT EXISTS Fortalezas (
    tipo_id INT, -- Tipo que tiene fortalezas

```

```

    tipo_fuerte_contra INT, -- Tipo contra el cual es fuerte
    PRIMARY KEY (tipo_id, tipo_fuerte_contra), -- Clave primaria compuesta
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id), -- Clave foránea que referencia a
Tipos
    FOREIGN KEY (tipo_fuerte_contra) REFERENCES Tipos(tipo_id) -- Clave foránea que
referencia a Tipos (contra quién es fuerte)
);

```

-- Tabla de Debilidades

```

CREATE TABLE IF NOT EXISTS Debilidades (
    tipo_id INT, -- Tipo que tiene debilidades
    tipo_debil_contra INT, -- Tipo contra el cual es débil
    PRIMARY KEY (tipo_id, tipo_debil_contra), -- Clave primaria compuesta
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id), -- Clave foránea que referencia a
Tipos
    FOREIGN KEY (tipo_debil_contra) REFERENCES Tipos(tipo_id) -- Clave foránea que
referencia a Tipos (contra quién es débil)
);

```

-- Tabla de Habilidades

```

CREATE TABLE IF NOT EXISTS Habilidades (
    habilidad_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada habilidad
    nombre_habilidad VARCHAR(100) NOT NULL, -- Nombre de la habilidad
    descripcion_efecto TEXT NOT NULL -- Descripción del efecto de la habilidad
);

```

-- Tabla de relación entre Habilidades y Pokémon

```

CREATE TABLE IF NOT EXISTS Pokemon_Habilidades (
    pokemon_id INT, -- ID del Pokémon que tiene la habilidad
    habilidad_id INT, -- ID de la habilidad
    PRIMARY KEY (pokemon_id, habilidad_id), -- Clave primaria compuesta por el Pokémon
y la habilidad
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(Id_pokemon), -- Clave foránea
que referencia a Pokémon
    FOREIGN KEY (habilidad_id) REFERENCES Habilidades(habilidad_id) -- Clave foránea
que referencia a Habilidades
);

```

-- Tabla de Movimientos

```

CREATE TABLE IF NOT EXISTS Movimientos (
    movimiento_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada
movimiento
    nombre_movimiento VARCHAR(100) NOT NULL, -- Nombre del movimiento
    tipo_movimiento VARCHAR(50) NOT NULL, -- Tipo de movimiento (físico, especial,
estado)
    potencia INT, -- Potencia del movimiento (puede ser NULL si no aplica)
    puntos_poder INT NOT NULL, -- Puntos de poder (PP)
    tipo_id INT, -- Relación con la entidad Tipos (fuego, agua, eléctrico, etc.)
);

```

```
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id) -- Clave foránea que relaciona  
con Tipos  
);
```

-- Tabla de relación entre Movimientos y Pokémon

```
CREATE TABLE IF NOT EXISTS Pokemon_Movimientos (  
    pokemon_id INT, -- ID del Pokémon que puede aprender el movimiento  
    movimiento_id INT, -- ID del movimiento  
    PRIMARY KEY (pokemon_id, movimiento_id), -- Clave primaria compuesta por el  
Pokémon y el movimiento  
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(Id_pokemon), -- Clave foránea  
que referencia a Pokémon  
    FOREIGN KEY (movimiento_id) REFERENCES Movimientos(movimiento_id) -- Clave  
foránea que referencia a Movimientos  
);
```

-- Tabla de relación entre Pokémon y Regiones (Pokémon nativos)

```
CREATE TABLE IF NOT EXISTS Pokemon_Regiones (  
    pokemon_id INT, -- ID del Pokémon nativo de la región  
    region_id INT, -- ID de la región  
    PRIMARY KEY (pokemon_id, region_id), -- Clave primaria compuesta  
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(Id_pokemon), -- Clave foránea  
que referencia a Pokémon  
    FOREIGN KEY (region_id) REFERENCES Regiones(region_id) -- Clave foránea que  
referencia a Regiones  
);
```

-- Tabla de Gimnasios

```
CREATE TABLE IF NOT EXISTS Gimnasios (  
    gimnasio_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada gimnasio  
    nombre_gimnasio VARCHAR(100) NOT NULL, -- Nombre del gimnasio  
    ciudad_id INT, -- ID de la ciudad donde está ubicado el gimnasio  
    lider_id INT, -- ID del líder del gimnasio  
    tipo_id INT, -- ID del tipo de gimnasio (relacionado con Tipos)  
    medalla_otorgada VARCHAR(100), -- Nombre de la medalla que otorga  
    FOREIGN KEY (ciudad_id) REFERENCES Ciudades(ciudad_id), -- Clave foránea que  
referencia a Ciudades  
    FOREIGN KEY (lider_id) REFERENCES Entrenadores(entrenador_id), -- Clave foránea  
que referencia a Entrenadores  
    FOREIGN KEY (tipo_id) REFERENCES Tipos(tipo_id) -- Clave foránea que referencia a  
Tipos  
);
```

-- Tabla de Evoluciones

```
CREATE TABLE IF NOT EXISTS Evoluciones (  
    evolucion_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada evolución
```

```

    pokemon_origen_id INT, -- ID del Pokémon de origen
    pokemon_evolucionado_id INT, -- ID del Pokémon evolucionado
    condicion_evolucion VARCHAR(100) NOT NULL, -- Condición necesaria para la
    evolución (nivel, piedra, amistad, etc.)
    FOREIGN KEY (pokemon_origen_id) REFERENCES Pokemon(Id_pokemon), -- Clave
    foránea que referencia al Pokémon de origen
    FOREIGN KEY (pokemon_evolucionado_id) REFERENCES Pokemon(Id_pokemon) --
    Clave foránea que referencia al Pokémon evolucionado
);

```

-- Tabla de Objetos

```

CREATE TABLE IF NOT EXISTS Objetos (
    objeto_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada objeto
    nombre_objeto VARCHAR(100) NOT NULL, -- Nombre del objeto (e.g., Poción, Baya,
    Piedra Evolutiva)
    tipo_objeto VARCHAR(50) NOT NULL, -- Tipo del objeto (e.g., Poción, Baya, Piedra
    Evolutiva)
    efecto TEXT NOT NULL -- Descripción del efecto del objeto
);

```

-- Tabla de relación entre Objetos y Pokémon

```

CREATE TABLE IF NOT EXISTS Pokemon_Objetos (
    pokemon_id INT, -- ID del Pokémon que puede utilizar el objeto
    objeto_id INT, -- ID del objeto
    PRIMARY KEY (pokemon_id, objeto_id), -- Clave primaria compuesta por el Pokémon y
    el objeto
    FOREIGN KEY (pokemon_id) REFERENCES Pokemon(Id_pokemon), -- Clave foránea
    que referencia a Pokémon
    FOREIGN KEY (objeto_id) REFERENCES Objetos(objeto_id) -- Clave foránea que
    referencia a Objetos
);

```

-- Tabla de Películas

```

CREATE TABLE IF NOT EXISTS Peliculas (
    pelicula_id INT AUTO_INCREMENT PRIMARY KEY, -- ID único para cada película
    nombre_pelicula VARCHAR(100) NOT NULL, -- Nombre de la película
    genero VARCHAR(50) NOT NULL, -- Género de la película (e.g., Aventura, Acción,
    Fantasía)
    duracion INT NOT NULL, -- Duración de la película en minutos
    personajes_principales TEXT NOT NULL -- Personaje(s) principal(es) de la película
);

```