Ejercicio: Pokedex Extendida

Descripción General

Desarrolla una aplicación web en Java utilizando Spring Boot, Spring Data JPA y MySQL para gestionar una pokedex de Pokémon. La aplicación deberá exponer múltiples endpoints REST que permitan realizar operaciones CRUD, filtrar información y consultar estadísticas, manteniendo la funcionalidad de actualizar el nivel de un Pokémon.

Requisitos Funcionales

1. **Listado Completo de Pokémon**
   - **Endpoint:** `GET /api/pokemons`
   - Devuelve la lista completa de Pokémon registrados.
2. **Obtener un Pokémon por ID**
   - **Endpoint:** `GET /api/pokemons/{id}`
   - Devuelve la información detallada de un Pokémon específico.
3. **Crear un Nuevo Pokémon**
   - **Endpoint:** `POST /api/pokemons`
   - Permite añadir un nuevo Pokémon a la base de datos. Se espera un objeto JSON con los datos del Pokémon.
4. **Eliminar un Pokémon**
   - **Endpoint:** `DELETE /api/pokemons/{id}`
   - Elimina un Pokémon específico identificándolo por su ID.
5. **Actualizar el Nivel de un Pokémon**
   - **Endpoint:** `PUT /api/pokemons/{id}/level`
   - Permite actualizar el campo `level` de un Pokémon. Se espera un objeto JSON que contenga el nuevo valor para el nivel.
6. **Filtrar Pokémon por Tipo**
   - **Endpoint:** `GET /api/pokemons/filter`

- **Parámetro:** `type`
- Devuelve la lista de Pokémon cuyo tipo coincide con el especificado.

7. **Filtrar Pokémon por Rango de HitPoints**
   - **Endpoint:** `GET /api/pokemons/hitpoints`
   - **Parámetros:** `min` y `max`
   - Devuelve la lista de Pokémon cuyo valor de hitPoints se encuentre entre el valor mínimo y máximo indicados.
8. **Consultar Estadísticas de la Pokedex**
   - **Endpoint:** `GET /api/pokemons/stats`
   - Devuelve estadísticas de la colección de Pokémon, que incluyen:
     - Promedio del nivel.
     - Promedio de hitPoints.
     - Cantidad de Pokémon agrupados por tipo.

Requisitos Técnicos

- **Entidad Pokémon:**

  La entidad debe incluir los siguientes atributos:
  - `id` (Long, autogenerado)
  - `name` (String, no nulo)
  - `type` (String, no nulo)
  - `hitPoints` (Long, no nulo)
  - `level` (Integer, no nulo)
- **Persistencia:**
  - Utiliza Spring Data JPA para interactuar con una base de datos MySQL.
  - Configura la conexión en el archivo `application.properties` (URL, usuario, contraseña, driver, etc.) y utiliza `ddl-auto=create` para generar la estructura de la base de datos automáticamente.
- **Servicios y Lógica de Negocio:**

    - Implementa un servicio principal (`PokemonService`) que gestione las operaciones de:
        - Obtener, crear, actualizar y eliminar Pokémon.
        - Filtrar Pokémon por tipo y por rango de hitPoints.
        - Consultar estadísticas (promedios y conteo por tipo).
    - Se debe mantener la funcionalidad para actualizar el nivel de un Pokémon.
- **Controlador REST:**
    - Implementa un controlador (`PokemonController`) que exponga todos los endpoints descritos y coordine las llamadas a los servicios correspondientes.

Estructura Sugerida del Proyecto

- **controllers:**

  Contendrá la clase `PokemonController.java` para definir los endpoints REST.
- **models:**

  Definición de la entidad `Pokemon.java` con sus atributos (id, name, type, hitPoints y level).
- **repositories:**

  Interfaz `PokemonRepository.java` que extienda de JpaRepository y permita realizar operaciones CRUD y consultas personalizadas.
- **services:**
    - Interfaz y clase de implementación de `PokemonService.java` para la lógica de negocio (incluyendo filtrado, operaciones CRUD y estadísticas).
- **configuración:**

  Archivo `application.properties` para la configuración de la base de datos y otros parámetros de Spring Boot.
- **Clase Principal:**

  `PokedexApplication.java` para iniciar la aplicación.

Población de Datos

```
INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Bulbasaur', 'Planta', 45, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Ivysaur', 'Planta', 60, 16);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Venusaur', 'Planta', 80, 32);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Charmander', 'Fuego', 39, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Charmeleon', 'Fuego', 58, 16);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Charizard', 'Fuego', 78, 36);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Squirtle', 'Agua', 44, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Wartortle', 'Agua', 59, 16);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Blastoise', 'Agua', 79, 36);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Caterpie', 'Bicho', 45, 3);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Metapod', 'Bicho', 50, 7);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Butterfree', 'Bicho', 60, 12);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Weedle', 'Bicho/Veneno', 40, 3);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Kakuna', 'Bicho/Veneno', 45, 7);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Beedrill', 'Bicho/Veneno', 65, 15);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Pidgey', 'Normal/Volador', 40, 5);
```

```
INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Pidgeotto', 'Normal/Volador', 63, 18);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Pidgeot', 'Normal/Volador', 83, 36);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Rattata', 'Normal', 30, 2);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Raticate', 'Normal', 55, 20);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Spearow', 'Normal/Volador', 40, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Fearow', 'Normal/Volador', 65, 22);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Ekans', 'Veneno', 35, 8);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Arbok', 'Veneno', 60, 24);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Pikachu', 'Eléctrico', 35, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Raichu', 'Eléctrico', 60, 22);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Sandshrew', 'Tierra', 50, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Sandslash', 'Tierra', 75, 22);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Nidoran♀', 'Veneno', 55, 5);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Nidorina', 'Veneno', 70, 16);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Nidoqueen', 'Veneno/Tierra', 85, 30);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Nidoran♂', 'Veneno', 55, 5);
```

```
INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Nidorino', 'Veneno', 70, 16);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Nidoking', 'Veneno/Tierra', 85, 30);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Clefairy', 'Hada', 70, 15);

INSERT INTO pokemons (name, type, hit_points, level) VALUES ('Clefable', 'Hada', 95, 30);
```