

Progetto Basi di Dati

PokéBase

Caccaro Sebastiano – Matricola 1142944

Zanatta Stefano – Matricola 1142897



1. Abstract

Pokémon è un media franchise giapponese creato nel 1996 da Satoshi Tajiri, e inizialmente nato come una coppia di videogiochi sviluppati da Game Freak e pubblicati da Nintendo. L'universo Pokémon si basa sugli omonimi mostriattoli immaginari che gli umani possono catturare, allenare, e far combattere tra loro per scopi ludici.

Il franchise videoludico, ad oggi (2017), ha venduto più di 300 milioni di copie, rendendolo secondo solo a Mario per volume di vendite. Il successo generato è stato tale da far nascere un anime (1997), un gioco di carte, e innumerevoli oggetti di merchandise.

Nel videogioco, come nel cartone animato, gli allenatori sono dotati di un Pokédex, ovvero un dispositivo dalle fattezze simili ad un telefono cellulare, che fornisce loro informazioni sui Pokémon che incontrano e/o ricercano.

Lo scopo di PokéBase sarà quello di emulare il suddetto device, fornendo inoltre funzioni di ricerca dettagliate e interrogazioni più precise e mirate. Verranno inoltre messe a disposizione di ogni allenatore informazioni non disponibili in un Pokédex standard, quali modalità di evoluzione, possibili abilità, e mosse apprendibili da ogni Pokémon.

Oltre a ciò, verranno fornite funzioni di inserimento necessarie ad aggiornare il database per l'arrivo di nuove generazioni di Pokémon.

2. Descrizione dei requisiti

Si vuole realizzare una base di dati che contenga informazioni relative al mondo di gioco dei pokémon, per ampliare il contenuto informativo del pokédex presente nel videogioco.

Una specie pokémon è caratterizzata da:

- ID (codice che la identifica univocamente);
- Nome
- Peso
- Altezza
- Descrizione testuale del pokémon

Ogni specie pokémon possiede un insieme di abilità e fino a due tipi.

Una abilità è una capacità passiva del pokémon.

Un tipo è una proprietà che descrive la natura e in parte le caratteristiche fisiche del pokémon.

Una mossa è un attacco che può essere imparato da un pokémon. Ogni mossa è caratterizzata da un tipo (una mossa di un certo tipo può essere appresa da un pokémon di un tipo diverso).

Ogni mossa è caratterizzata da:

- nome
- potenza: è il danno che la mossa infligge al nemico. Quando la potenza è negativa vengono applicate delle regole speciali: -1 significa che la mossa manderà sempre KO l'avversario, -2 significa che il danno è variabile e si può calcolare solo durante la battaglia
- categoria
- una breve descrizione che ne spiega il funzionamento e i tempi di ricarica;
- punti potenza (pp): indicano quante volte la mossa può essere usata in battaglia. I punti potenza possono aumentare fino al 60% dei punti potenza base.

Gran parte delle mosse possono essere imparate tramite MT o MN: la macchina tecnica (MT) è un oggetto utilizzabile da un pokémon una sola volta per imparare una nuova mossa (chiamata mossa MT); la macchina nascosta (MN) è come la MT ma utilizzabile all'infinito da più pokémon. Tutte le MN e alcune MT sono mosse utilitarie, cioè possono essere usate al di fuori della battaglia.

Alcuni pokémon possono evolversi. L'evoluzione è un processo irreversibile che trasforma un pokémon in un altro, mantenendo le sue mosse e migliorandone le caratteristiche fisiche.

Una specie pokémon può evolversi in più pokémon diversi, ma un pokémon "evoluto" può provenire da una sola "forma primitiva".

Alcune evoluzioni possono avvenire solo sotto condizioni particolari (tramite luogo, oggetto, scambio...).

Ogni pokémon appartiene ad un habitat, cioè una zona della mappa dove può essere trovato. Una zona è caratterizzata da:

- ID
- nome della zona
- morfologia
- regione di appartenenza
- descrizione

Ogni zona può avere tre fasi giornaliere: mattino, giorno, notte. Una zona può anche distinguersi a seconda delle sue caratteristiche morfologiche (per esempio acqua, grotta, edificio, percorso).

3. Progettazione Concettuale

Descrizione delle entità

Abilità

Una abilità è una capacità passiva che un pokémon possiede dalla nascita. Una specie pokémon può avere più abilità.

Attributi

- ID: smallint **PK** - *Id univoco che identifica ogni abilità*
- Nome: varchar(20) - *Nome dell'abilità, visualizzato nella descrizione di un pokemon*
- Descrizione: varchar(200) - *Spiegazione dell'utilità e delle caratteristiche dell'abilità*

Specie pokémon

Ogni pokémon appartiene a una specie basata sulle sue caratteristiche biologiche.

Attributi

- ID: smallint **PK** - *Id univoco che identifica ogni specie pokémon*
- Nome: varchar(15) - *Nome della specie pokémon*
- Peso: int - *Peso in grammi di una specie pokémon*
- Altezza: smallint - *Altezza in centimetri di una specie pokémon*

- Tipo1 varchar(10) FK(Tipo) - *Primo tipo una specie pokémon*
- Tipo2 varchar(10) FK(Tipo) - *Secondo tipo (opzionale) di una specie pokémon*
- Descrizione varchar(50)

Tipo

Un tipo è una proprietà che caratterizza un pokémon o una mossa. Ogni tipo può essere efficace, debole, neutrale o immune verso altri tipi.

Attributi

- Nome: varchar(10) **PK** - *Proprietà caratterizzante di una specie pokémon o di una mossa*
- Descrizione (50) - *Caratterizzazione e natura di un tipo*

Mossa

Una mossa è un attacco che un pokémon può usare in battaglia. Alcune mosse possono essere usate anche al di fuori della battaglia

Attributi

- Nome: varchar (20) **PK** - *Nome univoco di una mossa*
- Potenza: varchar(8) - *Fattore di calcolo del danno durante il combattimento*
- Categoria
- PPBase: tinyint - *quante volte una mossa può essere usata in battaglia. Questo valore può aumentare fino al 60% usando degli oggetti speciali.*
- Precisione: tinyint - *probabilità di colpire della mossa. Valore null significa che non può mai fallire*
- Descrizione: varchar(50)

Evoluzione

Alcuni pokémon possono evolversi in pokémon differenti. Una evoluzione può avvenire sotto alcune circostanze

Attributi

- Pokémon evoluto: varchar(15) **PK** - *specie pokémon evoluta*
- Evoluzione da: varchar() - *specie pokémon dalla quale si evolve 'pokémon evoluto'*
- Modalità evoluzione: varchar(50) - *come avviene l'evoluzione*
- Stato evoluzione: tinyint - *valore che indica quante pre-evoluzioni ha il pokémon evoluto*

Zona

Una zona è un luogo specifico dove è possibile trovare dei pokémon.

Attributi

- ID: smallint **PK** - *identificatore univoco della zona*
- Nome: varchar(40) - *Nome della zona*
- Regione: varchar(10) - *Regione dove si trova zona*
- descrizione: varchar(50)
- morfologia: varchar(10) - *caratteristiche fisiche della zona*

Relazioni

Apprendimento: specie pokémon - mossa

- una specie pokémon può apprendere almeno una mossa

Habitat: specie pokémon - zona

- Una specie pokémon può "abitare" in una o più zone
- Una zona esiste solo se è habitat di almeno un pokémon

Efficacia: tipo - tipo

- Un tipo può avere un modificatore di efficacia verso un altro tipo (x2, x0.5, x0)
- Il modificatore di default è x1

Appartenenza: mossa - tipo

- Ogni mossa ha un tipo
- un tipo definisce la natura di una mossa

Natura: specie pokémon - tipo

- la natura di un pokémon è definita da uno o due tipi
- la natura di un pokémon definisce le sue debolezze e resistenze

Linea evolutiva: specie pokémon - evoluzione

- alcuni pokémon (base) possono evolversi in altri pokémon (evoluzione)
- un pokémon può essere contemporaneamente sia base che evoluzione
- un pokémon evoluzione ha al più una base

Avvenimento: evoluzione - modalità evoluzione

- una evoluzione può avvenire tramite una specifica modalità

Caratterizzazione: specie pokémon - abilità

- una specie pokémon può avere più abilità

Generalizzazioni

Zona

Mattino - Giorno - Notte

L'orario di una zona determina quali pokémon si possono trovare (g. completa)

Grotta - Edificio - Acqua - Percorso

Una zona si può distinguere per le sue caratteristiche morfologiche (g. parziale)

Mossa

Mt - Mn

Una mossa può essere appresa tramite Mt o Mn (g. parziale)

Mossa Utilitaria

È una mossa che può essere utilizzata anche al di fuori della battaglia (g. parziale)

Descrizione dei vincoli di integrità aggiuntivi

Le Regioni (in Zona) possono essere selezionate solamente tra queste: Kanto, Hoenn, Johto, Sinnoh, Unima, Kalos o Alola.

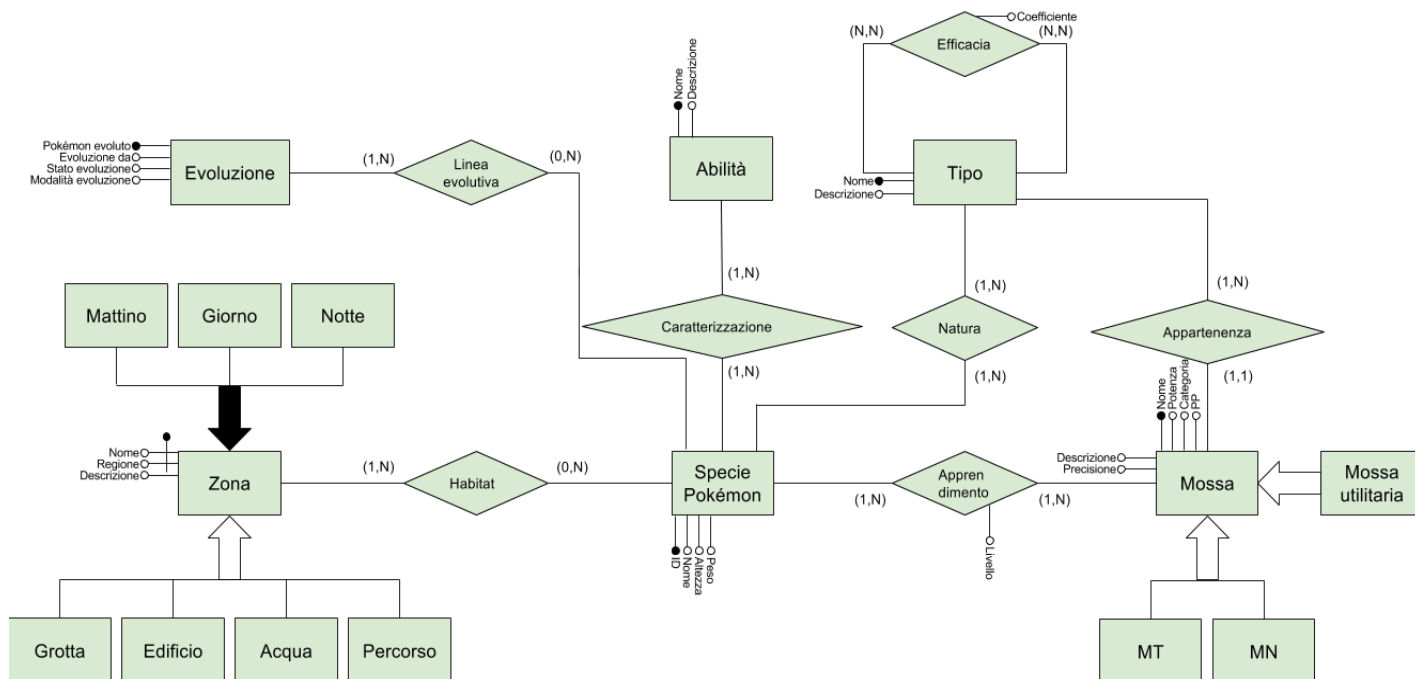
L'orario di un Habitat può essere solamente uguale a Mattino, Pomeriggio, Sera o Sempre*

Il livello di un pokémon è compreso tra [1,100].

Il coefficiente nella tabella efficacia può essere solamente compreso tra [0,2] ed è sempre un multiplo di 0.5.

Lo stato dell'evoluzione è un intero compreso fra [-1,4], (-1 = pokémon baby; 0 = pokémon base, 1°,2°,3°,4° stato evolutivo (nb: non esistono ancora pokémon con stati evolutivi > =3)

Schema Entità-Relazione



4. Progettazione logica

Scelte progettuali

L'orario Sempre*

In alcuni habitat l'orario non conta per quanto riguarda la presenza di pokémon.

Per evitare ridondanza nella base di dati abbiamo introdotto il nuovo orario "sempre" che comprende i tre orari originali.

Generalizzazione mattino giorno notte

La generalizzazione "mattino", "giorno", "notte" dell'entità "zona" è stata spostata nella tabella "Habitat" dello schema logico sotto forma di attributo ("orario"). In questo modo si evita ridondanza degli attributi di zona.

Generalizzazione morfologia

La generalizzazione "grotta", "edificio", "acqua", "percorso" dell'entità "zona" è rimasta nella tabella zona sotto forma di attributo "morfologia". Questa scelta è dovuta dal fatto che una zona possiede una sola caratteristica morfologica, quindi non c'è ridondanza.

Immutabilità di Tipo ed Efficacia

Le tabelle tipo ed efficacia sono bloccate tramite l'uso di foreign key tra i loro attributi. Questo perché l'aggiunta di un nuovo tipo comporterebbe anche l'aggiunta di [(Numero righe Tipo * 2) - 1] righe alla tabella efficacia. Siccome l'attributo coefficiente di tali righe è arbitrario (e quindi non calcolabile), non è possibile aggiungere automaticamente righe a Efficacia. Siccome i tipi pokémon non sono soggetti ad aggiornamenti frequenti, in caso di aggiunta di un nuovo tipo, sarà

Schema logico

SELECT DISTINCT Pokemon

FROM Habitat) AND e.StatoEvoluzione not in (0,-1);

Descrizione

Elenco di tutti i pokémon all'ultimo stadio di evoluzione e non base che si possono solo trovare in natura.

Output

Nome	arcanine
	alakazam
butterfree	tentacool
beedrill	golem
pidgeot	dewgong
raichu	muk
sandslash	cloyster
nidoking	gengar
ninetales	hypno
wigglytuff	kingler
golbat	seaking
vileplume	starmie
venomoth	jolteon
persian	flareon
golduck	omastar
primeape	kabutops

Query 3

SELECT e1.Difensore AS Difensore1, e2.Difensore AS Difensore2, e1.Attaccante AS AttaccanteForte

FROM Efficacia AS e1 JOIN Efficacia AS e2 ON e1.Attaccante = e2.Attaccante

WHERE e1.Coefficiente = 2.0 AND e2.Coefficiente = 2.0 AND e1.Difensore > e2.Difensore;

Descrizione

Per ogni coppia di tipi, elencare quali tipi sono forti con un moltiplicatore x4 verso la coppia di tipi

Output

Difensore1	Difensore2	AttaccanteForte	Fuoco	Coleottero	Roccia
Ghiaccio	Folletto	Acciaio	Ghiaccio	Coleottero	Roccia
Roccia	Folletto	Acciaio	Ghiaccio	Fuoco	Roccia
Roccia	Ghiaccio	Acciaio	Volante	Coleottero	Roccia
Roccia	Fuoco	Acqua	Volante	Fuoco	Roccia
Terra	Fuoco	Acqua	Elettro	Psico	Spettro
Terra	Roccia	Acqua	Fuoco	Acciaio	Terra
Spettro	Psico	Buio	Fuoco	Elettro	Terra
Erba	Buio	Coleottero	Roccia	Acciaio	Terra
Psico	Buio	Coleottero	Roccia	Elettro	Terra
Psico	Erba	Coleottero	Roccia	Fuoco	Terra
Volante	Acqua	Elettro	Veleno	Acciaio	Terra
Roccia	Acqua	Erba	Veleno	Elettro	Terra
Terra	Acqua	Erba	Veleno	Fuoco	Terra
Terra	Roccia	Erba	Veleno	Roccia	Terra
Drago	Buio	Folletto	Folletto	Erba	Veleno
Lotta	Buio	Folletto	Erba	Coleottero	Volante
Lotta	Drago	Folletto	Lotta	Coleottero	Volante
Coleottero	Acciaio	Fuoco	Lotta	Erba	Volante
Erba	Acciaio	Fuoco			
Erba	Coleottero	Fuoco			
Ghiaccio	Acciaio	Fuoco			
Ghiaccio	Coleottero	Fuoco			
Ghiaccio	Erba	Fuoco			
Erba	Drago	Ghiaccio			
Terra	Drago	Ghiaccio			
Terra	Erba	Ghiaccio			
Volante	Drago	Ghiaccio			
Volante	Erba	Ghiaccio			
Volante	Terra	Ghiaccio			
Buio	Acciaio	Lotta			
Ghiaccio	Acciaio	Lotta			
Ghiaccio	Buio	Lotta			
Normale	Acciaio	Lotta			
Normale	Buio	Lotta			
Normale	Ghiaccio	Lotta			
Roccia	Acciaio	Lotta			
Roccia	Buio	Lotta			
Roccia	Ghiaccio	Lotta			
Roccia	Normale	Lotta			

Query 4

```
SELECT Nome
FROM SpeciePokemon
WHERE ID not in (
    SELECT pokemonEvolvente
    FROM Evoluzione )
AND ID not in(
    SELECT pokemonEvoluto
    FROM Evoluzione
    WHERE StatoEvoluzione <> 0 );
```

Descrizione

Mostrare i Pokémon in forma base che non si evolvono

Output

Nome	mr-mime
	scyther
	jynx
zubat	electabuzz
tentacruel	magmar
ponyta	pinsir
rapidash	tauros
farfetchd	lapras
onix	ditto
hitmonlee	porygon
hitmonchan	aerodactyl
lickitung	snorlax
rhyhorn	articuno
rhydon	zapdos
chansey	moltres
tangela	mewtwo
kangaskhan	mew

Query 5

```
SELECT Nome, count(e1.Coefficiente) as deboleVerso
FROM SpeciePokemon JOIN Efficacia AS e1 ON e1.Difensore = Tipo1
JOIN Efficacia AS e2 on e2.Difensore = Tipo2
JOIN Evoluzione AS ev ON ev.PokemonEvoluto = SpeciePokemon.ID
WHERE e1.Coefficiente * e2.Coefficiente >1 AND ev.StatoEvoluzione = 2
GROUP BY Nome
order by count(e1.coefficiente) desc
LIMIT 1;
```

Descrizione

Pokémon all'ultimo stadio di evoluzione che ha il maggior numero di debolezze verso altri tipi

Output

Nome	deboleVerso
golem	102

Query 6

```
CREATE OR REPLACE VIEW TipiHabitat AS
SELECT distinct Zona, Pokemon, Tipo1 as Tipo
FROM Habitat JOIN SpeciePokemon ON Pokemon = ID
UNION
SELECT distinct Zona,Pokemon, Tipo2 as Tipo
FROM Habitat JOIN SpeciePokemon ON Pokemon = ID
WHERE Tipo2 IS NOT NULL;
```

```
CREATE OR REPLACE VIEW NumPokemonPerMorfologia AS
SELECT COUNT(Pokemon) as numPkm,Tipo,Morfologia
FROM TipiHabitat join Zona on Zona.ID = TipiHabitat.Zona
GROUP BY Tipo, Morfologia;
```

```
SELECT n1.Tipo, n1.Morfologia,n1.numPkm
FROM NumPokemonPerMorfologia as n1 where n1.numPkm =
(SELECT max(numPkm) from NumPokemonPerMorfologia where Tipo = n1.Tipo);
```

Descrizione

per ogni tipo mostrare in quale zona morfologica (acqua, grotta, percorso...) è presente il maggior numero di pokémon di quel tipo

Output

Tipo	Morfologia	numPkm
Acciaio	Grotta	1
Acciaio	Prateria	1
Acqua	Grotta	15
Coleottero	Acqua	5
Elettro	Prateria	5
Erba	Prateria	8
Fuoco	Prateria	6
Ghiaccio	Acqua	3
Lotta	Prateria	2
Normale	Prateria	23
Psico	Prateria	6
Roccia	Grotta	8
Spettro	Acqua	4
Terra	Acqua	7
Veleno	Grotta	16
Volante	Edificio	10

Query 7

```
SELECT S.Nome as PokemonEvoluto, S2.Nome as EvolveDa, E.ModalitaEvoluzione as Tramite
FROM Evoluzione as E JOIN SpeciePokemon as S JOIN SpeciePokemon as S2
ON E.PokemonEvoluto = S.ID
WHERE E.ModalitaEvoluzione like "Pietra%" AND
E.PokemonEvoluto = S.ID AND
E.PokemonEvolvente = S2.ID
ORDER BY S2.ID;
```

Descrizione

Mostra tutti i pokémon che si evolvono tramite Pietra;

Output

PokemonEvoluto	EvolveDa	Tramite
raichu	pikachu	PietraTuono
nidoqueen	nidorina	PietraLunare
nidoking	nidorino	PietraLunare
clefable	clefairy	PietraLunare
ninetales	vulpix	PietraFocaia
wigglytuff	jigglypuff	PietraLunare
vileplume	gloom	PietraFoglia
bellossom	gloom	PietraSolare
arcanine	growlithe	PietraFocaia
poliwrath	poliwhirl	PietraIdrica
victreebel	weepinbell	PietraFoglia
cloyster	shellder	PietraIdrica
exeggutor	exeggcute	PietraFoglia
starmie	staryu	PietraIdrica
vaporeon	eevee	PietraIdrica
jolteon	eevee	PietraTuono
flareon	eevee	PietraFocaia

6. Funzioni

Funzione 1

```
DELIMITER //
```

```
CREATE FUNCTION num_mosse(pkm varchar(15))
```

```
RETURNS tinyint
```

```
BEGIN
```

```
    DECLARE NumMosse tinyint;
```

```
    SELECT count(*) into NumMosse
```

```
    FROM SpeciePokemon as s JOIN Apprendimento as a on s.ID = a.Pokémon
```

```
    WHERE s.Nome = pkm;
```

```
    RETURN NumMosse;
```

```
END //
```

```
DELIMITER;
```

Descrizione

Elencare il numero di mosse conosciute da una Specie_Pokemon

Funzione 2

DELIMITER //

CREATE FUNCTION pokeName(pkID smallint)

RETURNS VARCHAR(15)

BEGIN

RETURN (SELECT Nome FROM SpeciePokemon AS s WHERE s.ID = pkID);

END //

DELIMITER ;

Descrizione

Ritorna il nome del pokémon dato il suo ID

Funzione 3

DROP FUNCTION IF EXISTS maxEv;

DELIMITER //

CREATE FUNCTION maxEv(pkm VARCHAR(15))

RETURNS VARCHAR(15)

BEGIN

DECLARE evPkm VARCHAR(15);

DECLARE bPkm VARCHAR(15);

SET evPkm = pkm;

SET bPkm = pkm;

WHILE evPkm IS NOT NULL DO

SET evPkm = NULL;

SELECT evoluto.Nome into evPkm

FROM Evoluzione AS e JOIN SpeciePokemon AS base ON

e.PokemonEvolvente = base.ID left outer JOIN SpeciePokemon AS evoluto ON

e.PokemonEvoluto = evoluto.ID

WHERE base.Nome = bPkm

LIMIT 1;

if evPkm IS NOT NULL then set bPkm = evPkm; end if;

end while;

return bPkm;

END//

DELIMITER ;

Descrizione

Ritorna la massima evoluzione del pokémon (esempio: maxEv(Bulbasaur) = Venusaur, maxEv(Charizard) = Charizard). Pokémon con più di una evoluzione ritornano una sola delle loro evoluzioni.

Funzione 4

DELIMITER //

CREATE FUNCTION zoneName(znID smallint)

RETURNS VARCHAR(15)

BEGIN

RETURN (SELECT Nome FROM Zona AS s WHERE s.ID = znID);

END //

DELIMITER ;

Descrizione

Ritorna il nome di una zona dato il suo ID

7. Trigger

Trigger 1a

CREATE TRIGGER Insert_validaRegione BEFORE INSERT ON Zona

FOR EACH ROW Begin

if new.Regione not in ('Kanto', 'Hoenn', 'Johto', 'Sinnoh', 'Unima', 'Kalos', 'Alola')

then SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT=' La regione non esiste';

end if;

END

Descrizione

Verifica l'esistenza della regione dopo l'inserimento di una Zona

Trigger 1b

CREATE TRIGGER Update_validaRegione AFTER UPDATE ON Zona

FOR EACH ROW Begin

if new.Regione not in ('Kanto', 'Hoenn', 'Johto', 'Sinnoh', 'Unima', 'Kalos', 'Alola')

THEN SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT=' La regione non esiste';

end if;

END

Descrizione

Verifica l'esistenza della regione dopo l'aggiornamento di una zona

Trigger 2a

```
CREATE TRIGGER Insert_ValidaLivelloEvol ON Evoluzione BEFORE INSERT
BEGIN
if(new.Livello > 100) new.Livello = 100
end if;
END
```

Descrizione

Verifica che il livello non sia superiore al 100 dopo un inserimento di una nuova evoluzione. In caso contrario imposta il livello a 100

Trigger 2b

```
CREATE TRIGGER Insert_ValidaLivelloEvol ON Evoluzione AFTER UPDATE
BEGIN
if(new.Livello > 100) new.Livello = 100
end if;
END
```

Descrizione

Verifica che il livello non sia superiore al 100 dopo l'aggiornamento di una evoluzione. In caso contrario imposta il livello a 100

8. VISTE

Vista 1

```
CREATE VIEW OR REPLACE VIEW ZonaPokemon AS
SELECT Pokemon as pkID, pokeName(Pokemon) as Pokémon, zoneName(Zona) as Zona, Orario
FROM Habitat
order by Pokemon;
```

Descrizione

Mostra la lista di tutti i Pokémon seguiti dalle zone dove, e in che periodo del giorno è possibile catturare ciascuna specie.

Vista2

CREATE OR REPLACE VIEW PokeMosse AS

```
SELECT p.Nome as Pokemon, m.Nome Mossa, m.Categoria as Categoria, m.Potenza as Potenza  
FROM SpeciePokemon AS p JOIN Apprendimento AS a ON p.ID = a.Pokemon JOIN Mossa as m  
ON a.Mossa = m.Nome;
```

Descrizione

Mostra la lista di tutti i pokémon con le mosse che possono imparare e le informazioni principali della mossa