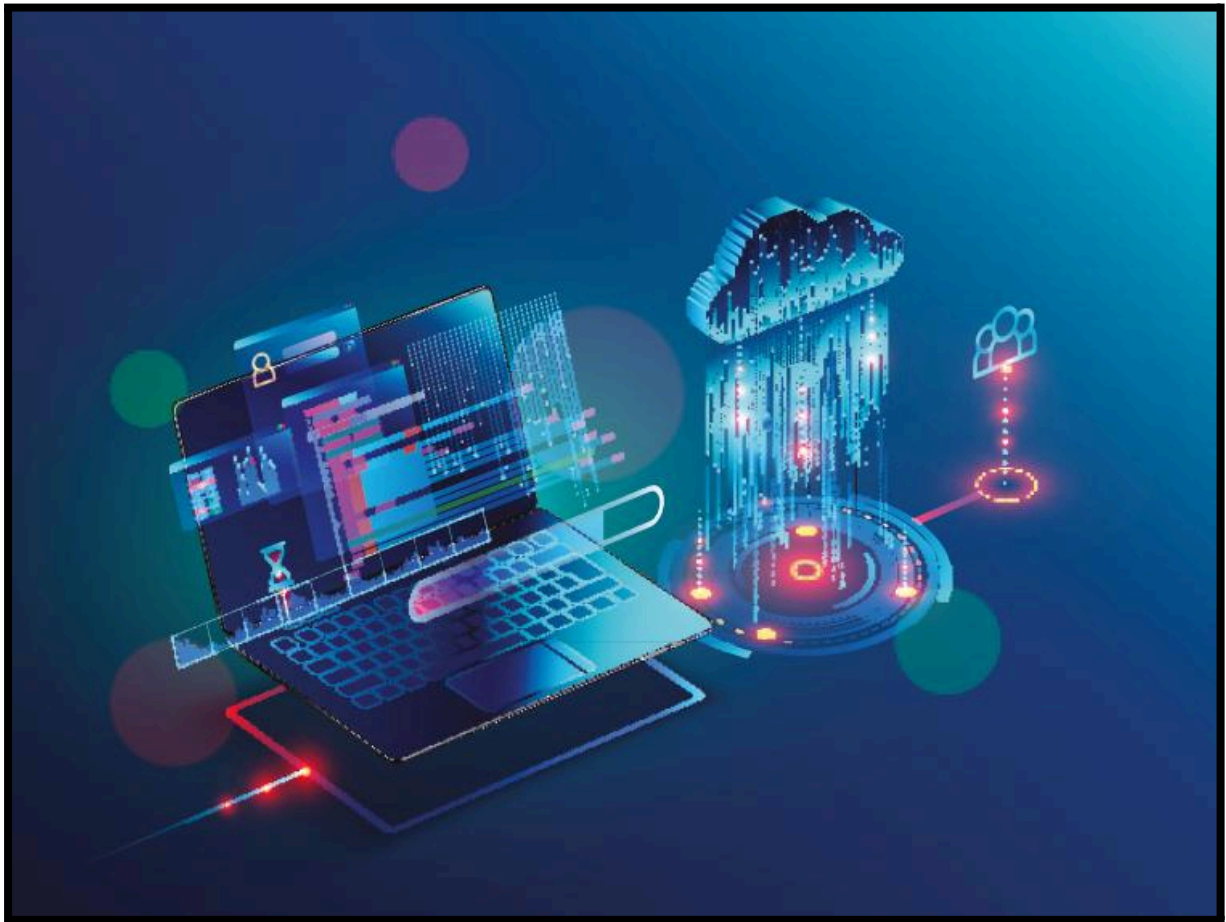


# Proyecto de Base de Datos: Pokemon



Francisco Rodríguez Pires  
12/01/2025  
1ºDAWm

# Índice

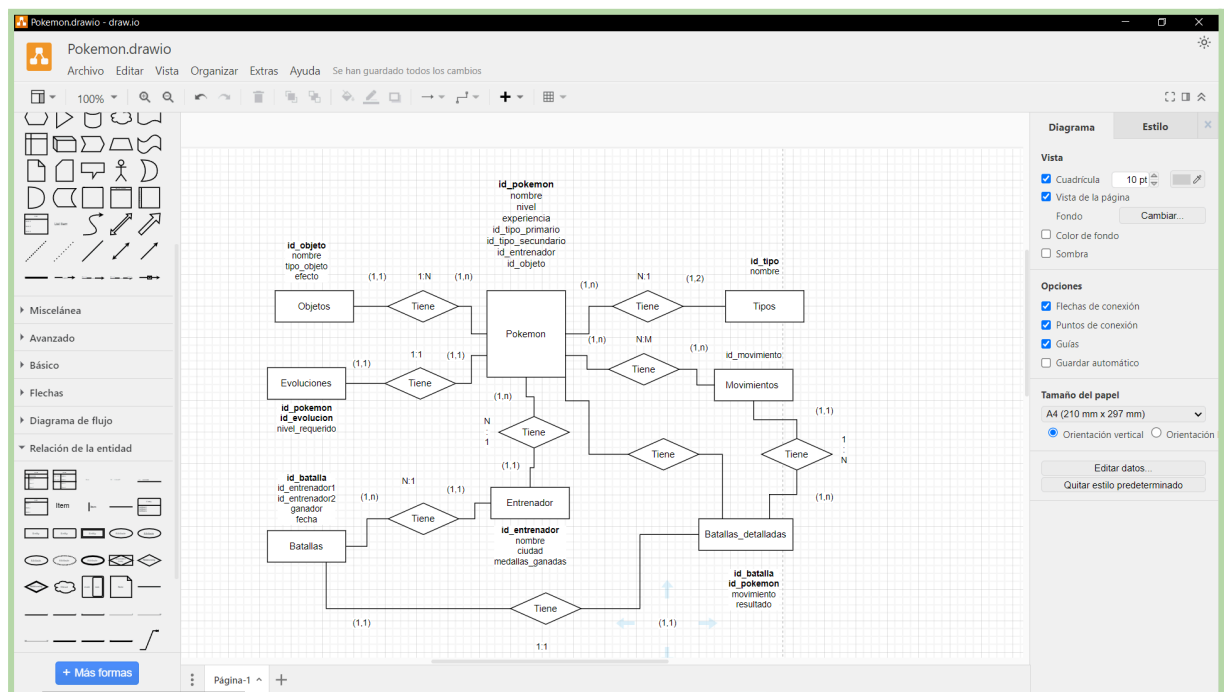
<b>1. Introducción</b>	<b>3</b>
<b>2. Modelo Entidad-Relación</b>	<b>3</b>
<b>3. Modelo Relacional</b>	<b>4</b>
<b>4. Carga Masiva de Datos</b>	<b>4</b>
<b>5. Consultas SQL</b>	<b>5</b>
<b>6. Conclusión</b>	<b>5</b>

# 1. Introducción

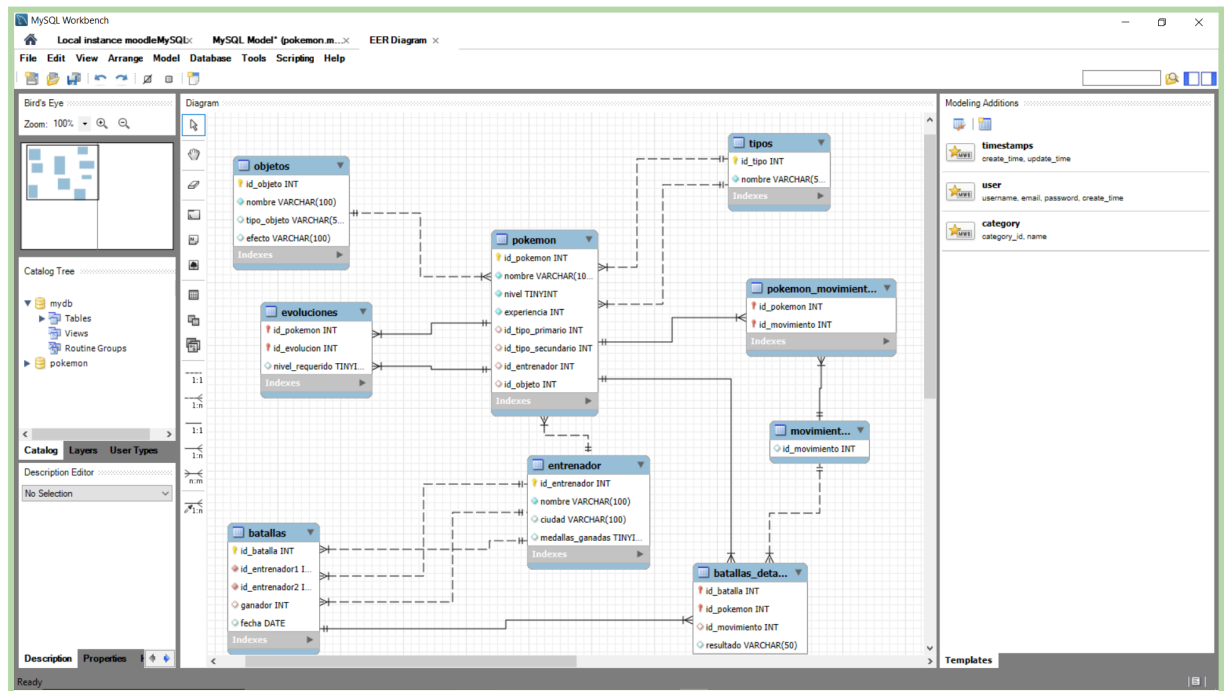
Este proyecto presenta el diseño e implementación de una base de datos para gestionar información relacionada con el mundo Pokémon. Incluye datos sobre entrenadores, Pokémon, tipos, movimientos, batallas, y objetos, brindando una solución robusta para futuras aplicaciones web de juegos, torneos, o investigaciones estadísticas sobre Pokémon.

El objetivo es proporcionar persistencia de datos eficiente, escalabilidad, y soporte para consultas complejas sobre batallas, evoluciones, y relaciones entre los diferentes elementos del sistema.

## 2. Modelo Entidad-Relación



### 3. Modelo Relacional



### 4. Carga Masiva de Datos

Generé un archivo con datos

```
-- Carga Masiva para Base de Datos Pokemon

-- Uso de la base de datos
USE pokemon;

-- Inserciones en la tabla tipos
INSERT INTO tipos (nombre) VALUES
('Fuego'), ('Agua'), ('Planta'), ('Eléctrico'), ('Roca'),
('Hielo'), ('Veneno'), ('Tierra'), ('Volador'), ('Psíquico'),
('Fantasma'), ('Siniestro'), ('Acero'), ('Dragón'), ('Hada');

-- Inserciones en la tabla objetos
INSERT INTO objetos (nombre, tipo_objeto, efecto) VALUES
('Poción', 'Curación', 'Restaura 20 PS'),
('Superpoción', 'Curación', 'Restaura 50 PS'),
('Hiperpoción', 'Curación', 'Restaura 120 PS'),
('Piedra Fuego', 'Evolución', 'Evoluciona Pokémon de tipo Fuego'),
('Antidoto', 'Curación', 'Cura el envenenamiento'),
('Revivir', 'Curación', 'Revive un Pokémon debilitado con mitad de sus PS'),
('Cuerda Huida', 'Objeto', 'Permite escapar de una cueva o mazmorra'),
('Carameloraro', 'Evolución', 'Sube un nivel a un Pokémon');

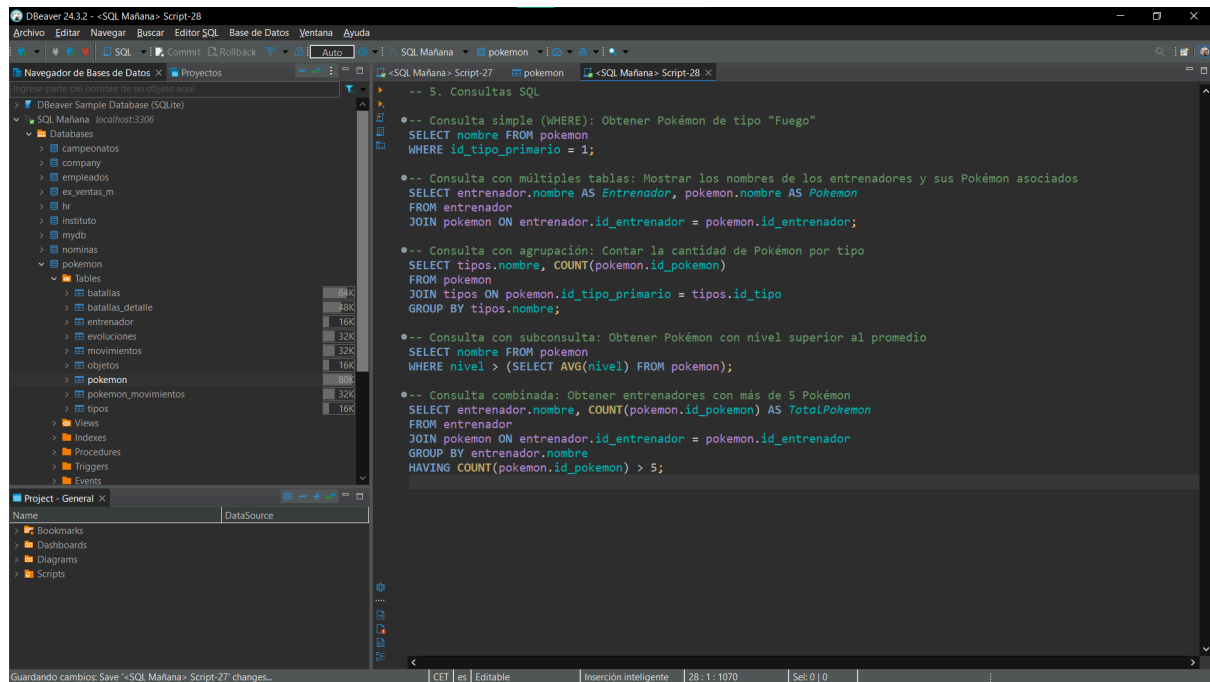
-- Inserciones en la tabla entrenadores
INSERT INTO entrenadores (nombre, ciudad, medallas_ganadas) VALUES
('Ash', 'Pueblo Paleta', 8),
('Brock', 'Ciudad Plateada', 5),
('Misty', 'Ciudad Celeste', 7),
('Gary', 'Ciudad Verde', 4),
('Sabrina', 'Ciudad Azafrán', 6),
('Erika', 'Ciudad Azulona', 3);
```

Estadísticas 1

Name	Value
Queries	9
Updated Rows	59
Execute time	0.031s
Fetch time	0.000s
Total time	0.031s
Start time	2025-01-12 19:13:16.686
Finish time	2025-01-12 19:13:16.737

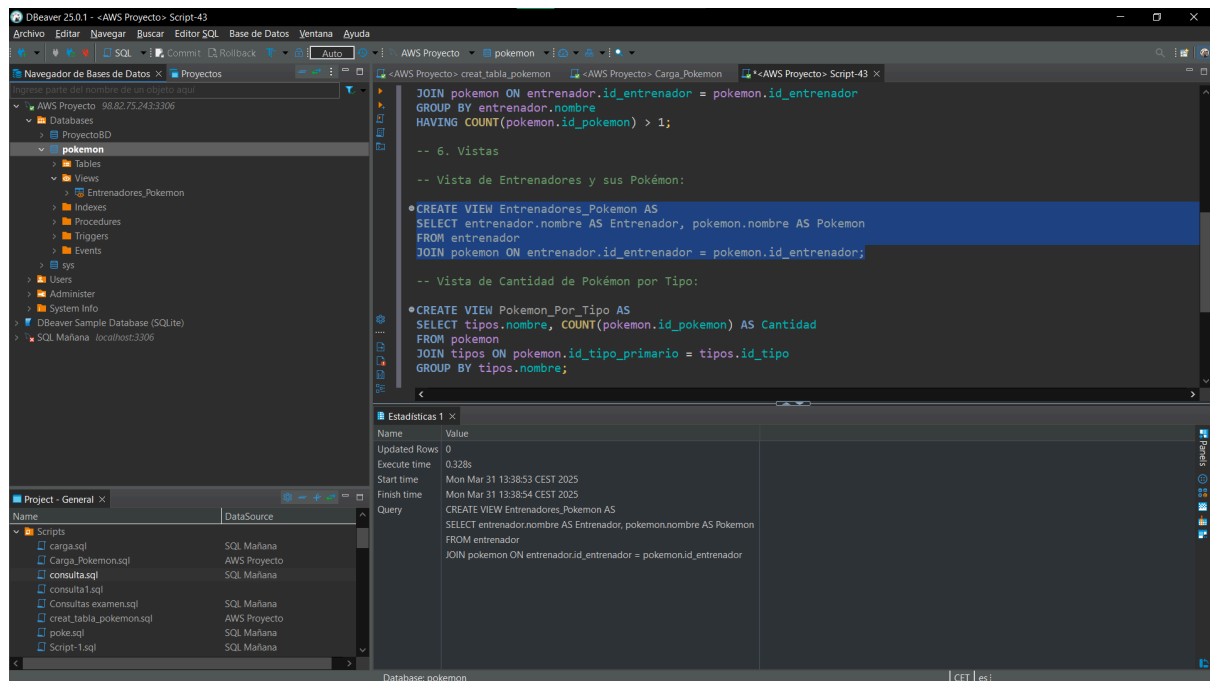
## 5. Consultas SQL

Creé otro fichero con las consultas



## 6. Vistas

La primera vista te muestra el nombre de cada entrenador junto con los nombres de los Pokémon que poseen



	Az Entrenador	Az Pokemon
1	Ash	Bulbasaur
2	Ash	Kricketune
3	Brock	Charmander
4	Brock	Shinx
5	Misty	Squirtle
6	Misty	Luxio
7	Gary	Pikachu
8	Gary	Luxray
9	Sabrina	Onix
10	Sabrina	Abra
11	Erika	Snorlax
12	Erika	Kadabra
13	Koga	Lapras
14	Koga	Alakazam
15	Blaine	Gengar
16	Blaine	Budew
17	Lance	Alakazam
18	Lance	Roselia

La segunda vista muestra la cantidad de Pokémon que hay de cada tipo

```

JOIN pokemon ON entrenador.id_entrenador = pokemon.id_entrenador
GROUP BY entrenador.nombre
HAVING COUNT(pokemon.id_pokemon) > 1;

-- 6. Vistas

-- Vista de Entrenadores y sus Pokémon:

CREATE VIEW Entrenadores_Pokemon AS
SELECT entrenador.nombre AS Entrenador, pokemon.nombre AS Pokemon
FROM entrenador
JOIN pokemon ON entrenador.id_entrenador = pokemon.id_entrenador;

-- Vista de Cantidad de Pokémon por Tipo:

CREATE VIEW Pokemon_Por_Tipo AS
SELECT tipos.nombre, COUNT(pokemon.id_pokemon) AS Cantidad
FROM pokemon
JOIN tipos ON pokemon.id_tipo_primario = tipos.id_tipo
GROUP BY tipos.nombre;

```

Estadísticas 1

Name	Value
Updated Rows	0
Execute time	0.313s
Start time	Mon Mar 31 13:42:29 CEST 2025
Finish time	Mon Mar 31 13:42:30 CEST 2025
Query	CREATE VIEW Pokemon_Por_Tipo AS

	Az nombre	123 Cantidad
1	Fuego	12
2	Agua	16
3	Planta	18
4	Eléctrico	12
5	Roca	12
6	Hielo	6
7	Veneno	14
8	Volador	6
9	Psíquico	8
10	Fantasma	3
11	Siniestro	5
12	Acero	4
13	Dragón	3
14	Hada	8
15	Lucha	24

## 7. Funciones y procedimientos

### 1. Función para Obtener el Nivel Promedio de Pokémon

The screenshot shows the DBeaver interface with the 'Navegador de Bases de Datos' on the left and the 'Pokemon\_queries' editor on the right. The editor contains the following SQL code:

```
CREATE VIEW Pokemon_Por_Tipo AS
SELECT tipos.nombre, COUNT(pokemon.id_pokemon) AS Cantidad
FROM pokemon
JOIN tipos ON pokemon.id_tipo_primario = tipos.id_tipo
GROUP BY tipos.nombre;

-- 7. Funciones y Procedimientos

-- Función para Obtener el Nivel Promedio de Pokémon:

DELIMITER //
CREATE FUNCTION NivelPromedioPokemon() RETURNS DECIMAL(10, 2)
DETERMINISTIC
BEGIN
    DECLARE promedio DECIMAL(10, 2);
    SELECT AVG(nivel) INTO promedio FROM pokemon;
    RETURN promedio;
END //
DELIMITER ;

SELECT NivelPromedioPokemon();
```

The results pane shows a single row with the value 22,08.

1	22,08

### 2. Función para Contar Pokémon por Tipo

The screenshot shows the DBeaver interface with the 'Navegador de Bases de Datos' on the left and the 'Pokemon\_queries' editor on the right. The editor contains the following SQL code:

```
END //
DELIMITER ;

SELECT NivelPromedioPokemon();

-- Función para Contar Pokémon por Tipo

DELIMITER //
CREATE FUNCTION CantidadPokemonPorTipo(nombre_tipo VARCHAR(50)) RETURNS INT
DETERMINISTIC
BEGIN
    DECLARE cantidad INT;
    SELECT COUNT(pokemon.id_pokemon) INTO cantidad
    FROM pokemon
    JOIN tipos ON pokemon.id_tipo_primario = tipos.id_tipo
    WHERE tipos.nombre = nombre_tipo;
    RETURN cantidad;
END //
DELIMITER ;

SELECT CantidadPokemonPorTipo("Hada");
```

The results pane shows a single row with the value 8.

1	8

### 3. Procedimiento para Mostrar Entrenadores con Más de 1 Pokémon

The screenshot shows a database management tool interface. On the left, a tree view displays the database structure, including tables like 'entrenador' and 'pokemon', and a procedure named 'EntrenadoresConMasDeUnPokemon'. The main window displays the following SQL code:

```
WHERE tipos.nombre = nombre_tipo;
RETURN cantidad;
END //
DELIMITER ;

SELECT CantidadPokemonPorTipo("Hada");

-- Procedimiento para Mostrar Entrenadores con Más de 1 Pokémon

DELIMITER //
CREATE PROCEDURE EntrenadoresConMasDeUnPokemon()
BEGIN
    SELECT entrenador.nombre, COUNT(pokemon.id_pokemon) AS TotalPokemon
    FROM entrenador
    JOIN pokemon ON entrenador.id_entrenador = pokemon.id_entrenador
    GROUP BY entrenador.nombre
    HAVING COUNT(pokemon.id_pokemon) > 1;
END //
DELIMITER ;

CALL EntrenadoresConMasDeUnPokemon();
```

Below the code, the results of the procedure are shown in a table:

entrenador 1	Estadísticas 1
nombre	TotalPokemon
Ash	2
Brock	2
Misty	2
Gary	2
Sabrina	2
Erika	2
Koga	2

### 4. Procedimiento para Mostrar Pokémon de un Tipo Específico

The screenshot shows the same database management tool interface. The SQL code in the main window is as follows:

```
SELECT entrenador.nombre, COUNT(pokemon.id_pokemon) AS TotalPokemon
FROM entrenador
JOIN pokemon ON entrenador.id_entrenador = pokemon.id_entrenador
GROUP BY entrenador.nombre
HAVING COUNT(pokemon.id_pokemon) > 1;
END //
DELIMITER ;

CALL EntrenadoresConMasDeUnPokemon();

-- Procedimiento para Mostrar Pokémon de un Tipo Específico

DELIMITER //
CREATE PROCEDURE MostrarPokemonPorTipo(IN nombre_tipo VARCHAR(50))
BEGIN
    SELECT nombre FROM pokemon
    WHERE id_tipo_primario = (SELECT id_tipo FROM tipos WHERE nombre = nombre_tipo);
END //
DELIMITER ;

CALL MostrarPokemonPorTipo("Dragon");
```

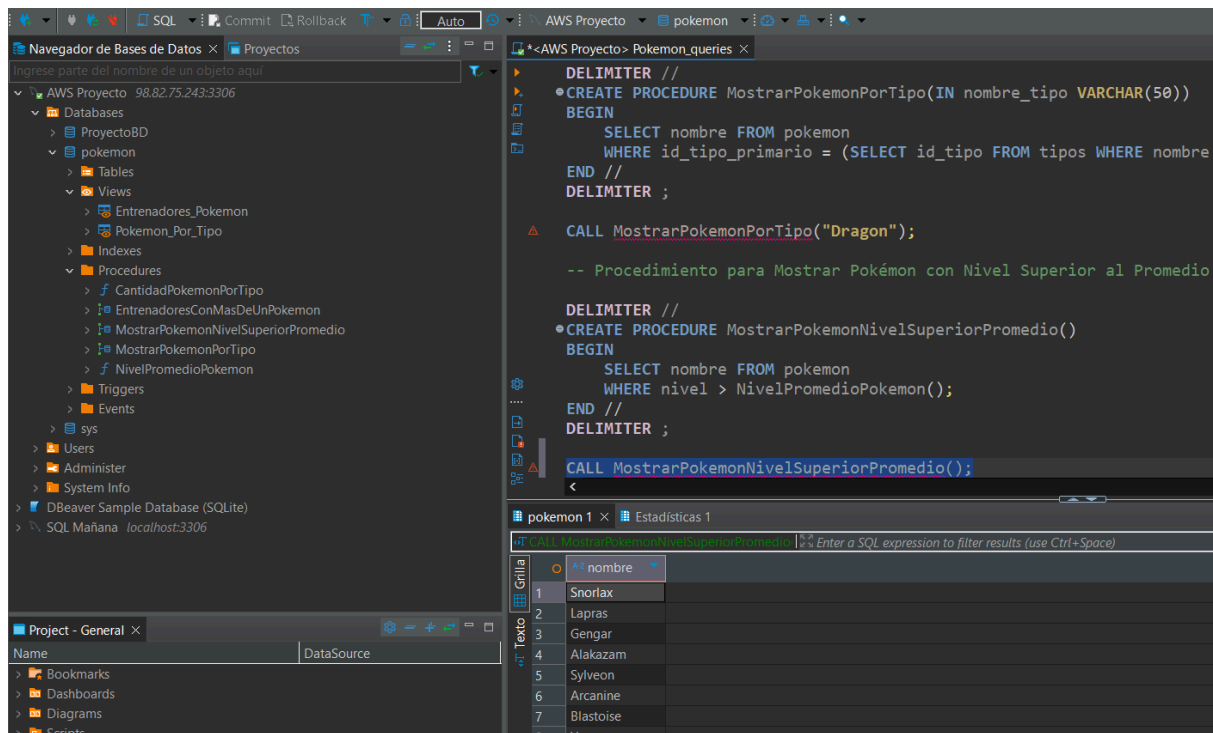
The results of the procedure are shown in a table:

pokemon 1	Estadísticas 1
nombre	
Dragonite	
Garchomp	
Trapinch	

### 5. Procedimiento para Mostrar Pokémon con Nivel Superior al Promedio

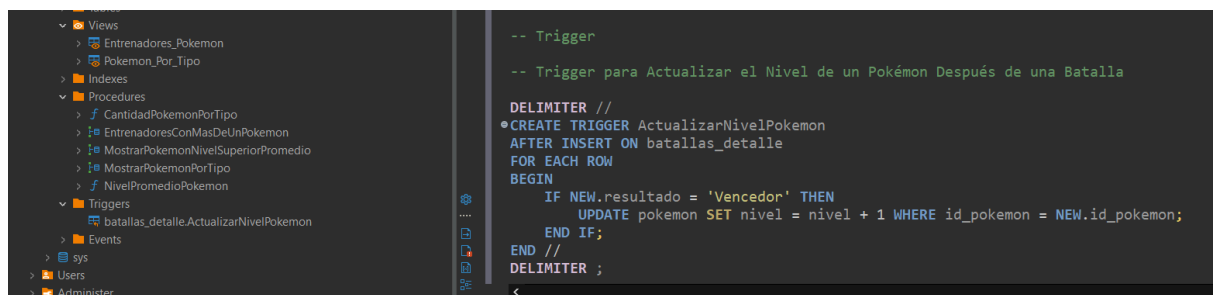
El procedimiento usa una función creada previamente





## 8. Triggers

- Trigger para Actualizar el Nivel de un Pokémon Después de una Batalla



prueba:

antes:

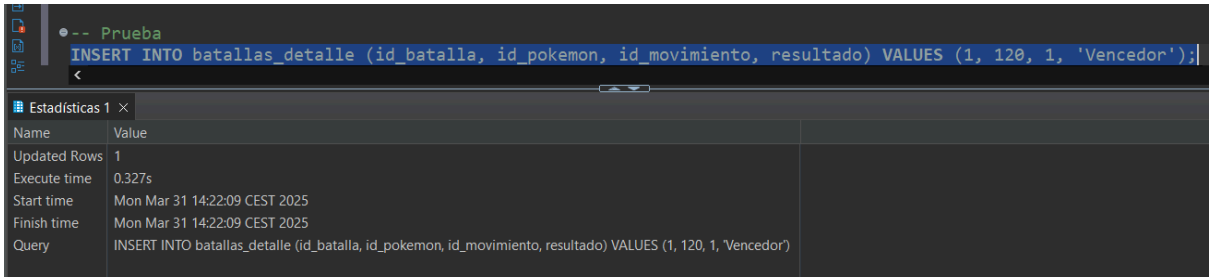
\*AWS Proyecto> Pokemon\_queries | pokemon x

Propiedades | Datos | Diagrama

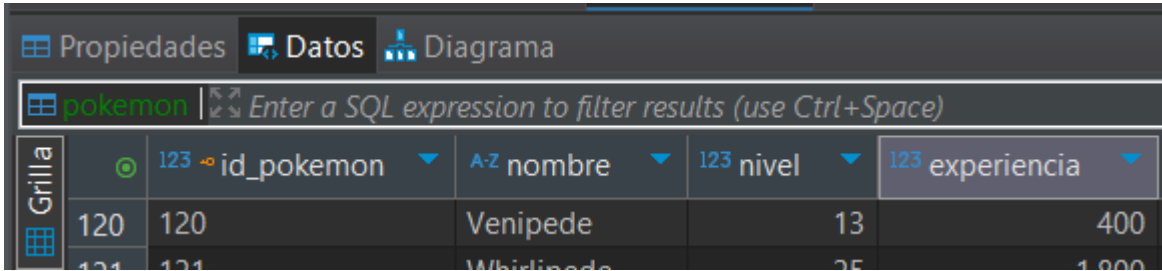
pokemon | Enter a SQL expression to filter results (use Ctrl+Space)

Grilla	id_pokemon	nombre	nivel	experiencia
120	120	Venipede	12	400
121	121	Whirlipede	25	1800

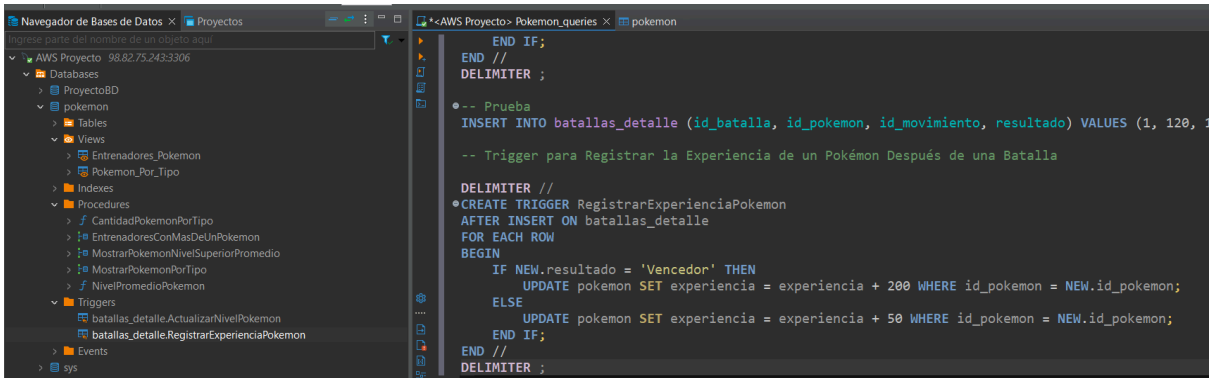
ejecuto el trigger:



después:

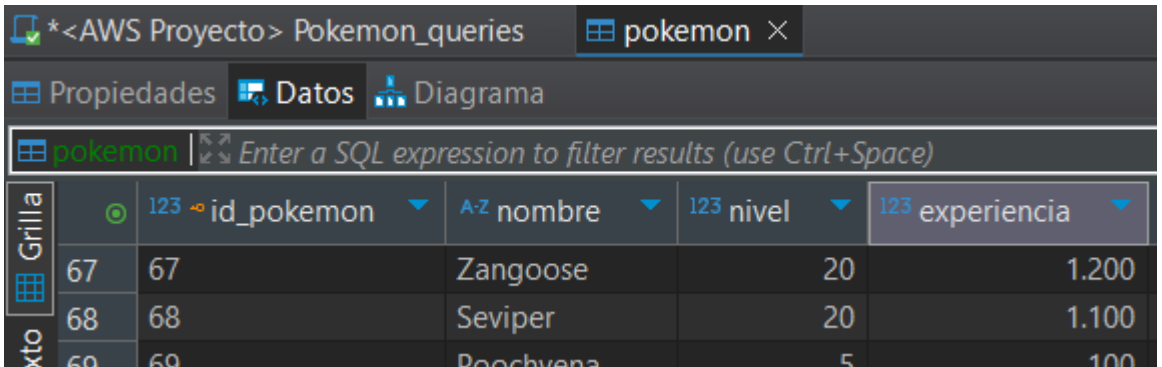


- Trigger para Registrar la Experiencia de un Pokémon Después de una Batalla



Prueba:

antes:



ejecuto el trigger:

```
-- Prueba
INSERT INTO batallas_detalle (id_batalla, id_pokemon, id_movimiento, resultado)
VALUES
(51, 67, 1, 'Vencedor'),
(51, 68, 3, 'Derrotado');
```

Name	Value
Updated Rows	2
Execute time	0.257s
Start time	Mon Mar 31 14:34:30 CEST 2025
Finish time	Mon Mar 31 14:34:30 CEST 2025
Query	INSERT INTO batallas_detalle (id_batalla, id_pokemon, id_movimiento, resultado) VALUES (51, 67, 1, 'Vencedor'), (51, 68, 3, 'Derrotado')

después:

* <AWS Proyecto> Pokemon_queries					
*pokemon					
Propiedades Datos Diagrama					
pokemon Enter a SQL expression to filter results (use Ctrl+Space)					
Grilla	123 id_pokemon	A-Z nombre	123 nivel	123 experiencia	
67	67	Zangoose	21	1.400	
68	68	Seviper	20	1.150	
69	69	Donphan	5	100	

## 9. Conclusión

Este proyecto ha permitido aplicar conocimientos sobre diseño de bases de datos, desde la planificación en el modelo entidad-relación hasta su implementación en MySQL y carga masiva de datos. He aprendido la importancia de las relaciones bien definidas, la normalización de datos y la optimización para consultas complejas.

Esta experiencia refuerza mis habilidades técnicas y me prepara para futuras aplicaciones prácticas, como el desarrollo de una aplicación web funcional para la comunidad de jugadores de Pokémon o analistas de datos de juegos.