**DBST 651 Project Step #3**

**By: Matt Lester**

**15 July 2024**

-- Drop statements for all objects

DROP TABLE IF EXISTS Evolution;

DROP TABLE IF EXISTS Pokemon;

DROP TABLE IF EXISTS Type;

DROP TABLE IF EXISTS Ability;

DROP TABLE IF EXISTS Trainer;

DROP VIEW IF EXISTS PokemonOverview;

DROP VIEW IF EXISTS TrainerDetails;

DROP SEQUENCE IF EXISTS seq\_pokemon;

DROP SEQUENCE IF EXISTS seq\_trainer;

DROP TRIGGER IF EXISTS trg\_before\_insert\_pokemon;

DROP TRIGGER IF EXISTS trg\_before\_insert\_trainer;

-- Create statements for all tables and constraints

CREATE TABLE Type (

type\_id INT PRIMARY KEY,

type\_name VARCHAR(30) NOT NULL UNIQUE,

strength TEXT,

weakness TEXT,

description TEXT

);

CREATE TABLE Ability (

ability\_id INT PRIMARY KEY,

ability\_name VARCHAR(50) NOT NULL UNIQUE,

effect TEXT,

description TEXT,

category VARCHAR(50)

);

CREATE TABLE Trainer (

trainer\_id INT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

age INT CHECK (age >= 0),

region VARCHAR(50),

experience\_level VARCHAR(50)

);

CREATE TABLE Pokemon (

pokemon\_id INT PRIMARY KEY,

name VARCHAR(50) NOT NULL,

type\_id INT,

ability\_id INT,

hp INT CHECK (hp >= 0),

attack INT CHECK (attack >= 0),

defense INT CHECK (defense >= 0),

speed INT CHECK (speed >= 0),

description TEXT,

trainer\_id INT,

FOREIGN KEY (type\_id) REFERENCES Type(type\_id),

FOREIGN KEY (ability\_id) REFERENCES Ability(ability\_id),

FOREIGN KEY (trainer\_id) REFERENCES Trainer(trainer\_id)

);

CREATE TABLE Evolution (

evolution\_id INT PRIMARY KEY,

pokemon\_id INT,

evolves\_to\_id INT,

level INT CHECK (level >= 0),

method VARCHAR(50),

FOREIGN KEY (pokemon\_id) REFERENCES Pokemon(pokemon\_id),

FOREIGN KEY (evolves\_to\_id) REFERENCES Pokemon(pokemon\_id)

);

-- Create indexes for natural, foreign key, and frequently queried columns

CREATE INDEX idx\_pokemon\_name ON Pokemon(name);

CREATE INDEX idx\_type\_name ON Type(type\_name);

CREATE INDEX idx\_ability\_name ON Ability(ability\_name);

CREATE INDEX idx\_trainer\_name ON Trainer(name);

-- Create a minimum of two views

-- View for Pokemon overview

CREATE VIEW PokemonOverview AS

SELECT p.pokemon\_id, p.name, t.type\_name, a.ability\_name, p.hp, p.attack, p.defense, p.speed

FROM Pokemon p

JOIN Type t ON p.type\_id = t.type\_id

JOIN Ability a ON p.ability\_id = a.ability\_id;

-- View for Trainer details

CREATE VIEW TrainerDetails AS

SELECT tr.trainer\_id, tr.name AS trainer\_name, tr.region, tr.experience\_level, COUNT(p.pokemon\_id) AS pokemon\_count

FROM Trainer tr

LEFT JOIN Pokemon p ON tr.trainer\_id = p.trainer\_id

GROUP BY tr.trainer\_id, tr.name, tr.region, tr.experience\_level;

-- Create a minimum of two sequences

CREATE SEQUENCE seq\_pokemon START WITH 1 INCREMENT BY 1;

CREATE SEQUENCE seq\_trainer START WITH 1 INCREMENT BY 1;

-- Create a minimum of two triggers

-- Trigger for automatic ID generation for Pokemon

CREATE TRIGGER trg\_before\_insert\_pokemon

BEFORE INSERT ON Pokemon

FOR EACH ROW

BEGIN

SELECT seq\_pokemon.NEXTVAL INTO :NEW.pokemon\_id FROM dual;

END;

-- Trigger for automatic ID generation for Trainer

CREATE TRIGGER trg\_before\_insert\_trainer

BEFORE INSERT ON Trainer

FOR EACH ROW

BEGIN

SELECT seq\_trainer.NEXTVAL INTO :NEW.trainer\_id FROM dual;

END;

-- Demonstrate successful creation of all objects by querying the database catalog/data dictionary

SELECT object\_name, object\_type FROM user\_objects WHERE object\_type IN ('TABLE', 'VIEW', 'SEQUENCE', 'TRIGGER');

SELECT table\_name FROM user\_tables;

SELECT sequence\_name FROM user\_sequences;

SELECT trigger\_name FROM user\_triggers;

-- Describe the business purpose of views and triggers

-- The PokemonOverview view provides a comprehensive summary of each Pokemon, including its name, type, ability, and key attributes, facilitating quick access to important information.

-- The TrainerDetails view offers detailed insights into each trainer, including the number of Pokemon they train, helping to manage and evaluate trainer performance.

-- The trg\_before\_insert\_pokemon trigger ensures that each new Pokemon record gets a unique identifier automatically, simplifying data entry and maintaining consistency.

-- The trg\_before\_insert\_trainer trigger ensures that each new Trainer record gets a unique identifier automatically, simplifying data entry and maintaining consistency.

-- Select from views to demonstrate they work correctly

SELECT \* FROM PokemonOverview;

SELECT \* FROM TrainerDetails;