**Jean Van Wittenberge, Brassica Selvaratnam**

**MCD :**

Fait sur Mocodo.net :

est de type, 1N pokemons, 0N types

pokemons: id, nom, niveau, exp\_en\_cours

:

:

types: id, nom

connaître, 1N pokemons, 0N attaques

:

:

être de, 11 attaques, 0N types

attaques: id, nom, puissance, precision, description

:

:

:

:

:

dresseurs: id, nom, age, genre

appartenir, 0N dresseurs, 11 pokemons

Une image contenant texte, capture d’écran, diagramme, Police

Description générée automatiquement

**MLD :**

dresseurs(**id**,nom,age, genre)

pokemons(**id**,nom,niveau,exp\_en\_cours, #id\_dresseur)

types(**id**,nom)

attaques(**id**,nom,puissance,precision,description, #id\_type)

est\_de\_type(**#id\_pokemon**, **#id\_type**)

connait\_attaque(**#id\_pokemon**, **#id\_attaque**)

**Script pour créer la BDD :**

CREATE DATABASE projet\_sql\_pokemon;

USE projet\_sql\_pokemon;

CREATE TABLE dresseur(

id INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR(50) NOT NULL,

age INT,

genre VARCHAR(1)

);

CREATE TABLE pokemons(

id INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR(50) NOT NULL,

niveau INT DEFAULT 1,

exp\_en\_cours INT DEFAULT 0,

id\_dresseur INT REFERENCES dresseur(id)

);

CREATE TABLE types(

id INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR(50) NOT NULL

);

CREATE TABLE attaques(

id INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR (50) NOT NULL,

puissance INT NOT NULL,

`precision` INT NOT NULL,

description VARCHAR(100),

id\_type INT NOT NULL REFERENCES types(id)

);

CREATE TABLE est\_de\_type(

id\_pokemon INT REFERENCES pokemons(id),

id\_type INT REFERENCES type(id),

PRIMARY KEY(id\_pokemon,id\_type)

);

CREATE TABLE connait\_attaque(

id\_pokemon INT REFERENCES pokemons(id),

id\_attaque INT REFERENCES attaques(id),

PRIMARY KEY(id\_pokemon,id\_attaque)

);

**Remplissage de la base de donnée :**

[INSERT](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/insert.html) INTO types(nom) [VALUES](http://localhost/phpmyadmin/url.php?url=https://dev.mysql.com/doc/refman/8.0/en/miscellaneous-functions.html%23function_values)('Feu'), ('Eau'), ('Plante'), ('Electrique'), ('Roche'), ('Vol');

INSERT INTO dresseur(nom, age, genre) VALUES('Brassica', 21, 'F'), ('Bob', 25, 'H'), ('Charlie', 18, 'N'), ('Jean', 21, 'H'), ('Eva', 30, 'F');

INSERT INTO pokemons (nom, niveau, exp\_en\_cours, id\_dresseur) VALUES

('Bulbizarre', 5, 20, 1),

('Salameche', 5, 45, 2),

('Carapuce', 5, 10, 3),

('Pikachu', 6, 80, 3),

('Salamèche', 7, 60, 5),

('Rattata', 3, 30, 1),

('Dracaufeu', 8, 75, 2),

('Magicarpe', 4, 5, 3),

('Roucool', 5, 40, 4),

('Herbizarre', 6, 70, 5),

('Aspicot', 3, 15, 1),

('Miaouss', 4, 90, 2),

('Raichu', 7, 25, 3),

('Rapasdepic', 5, 50, 4),

('Abo', 3, 95, 5),

('Tentacool', 4, 85, 1),

('Chenipan', 2, 55, 2),

('Papilusion', 6, 35, 3),

('Rattatac', 4, 65, 4),

('Nosferapti', 3, 32, 5),

('Mewtwo', 10, 99, 1),

('Leveinard', 8, 78, 2),

('Goupix', 5, 12, 3),

('Mew', 9, 60, 4),

('Ronflex', 10, 42, 5),

('Grodoudou', 6, 88, 1),

('Piafabec', 4, 22, 2),

('Ramoloss', 7, 77, 3),

('Taupiqueur', 3, 18, 4),

('Tadmorv', 5, 48, 5),

-- Ajoutez d'autres pokémons avec des dresseurs différents...

('Pikachu', 3, 80, 1), -- Exemple de plusieurs dresseurs possédant le même Pokémon

('Pikachu', 4, 85, 4); -- Exemple de plusieurs dresseurs possédant le même Pokémon avec des stats différentes

INSERT INTO est\_de\_type (id\_pokemon, id\_type) VALUES

-- Pokémon avec un seul type

(1, 3), -- Bulbizarre est de type Plante

(2, 1), -- Salameche est de type Feu

(3, 2), -- Carapuce est de type Eau

(4, 4), -- Pikachu est de type Electrique

(5, 1), -- Salamèche est de type Feu

(6, 5), -- Rattata est de type Roche

(7, 1), -- Dracaufeu est de type Feu

(8, 2), -- Magicarpe est de type Eau

(9, 6), -- Roucool est de type Vol

(10, 3), -- Herbizarre est de type Plante

(11, 3), -- Aspicot est de type Plante

(12, 5), -- Miaouss est de type Roche

(13, 4), -- Raichu est de type Electrique

(14, 6), -- Rapasdepic est de type Vol

(15, 4), -- Abo est de type Electrique

(16, 2), -- Tentacool est de type Eau

(17, 3), -- Chenipan est de type Plante

(18, 6), -- Papilusion est de type Vol

(19, 5), -- Rattatac est de type Roche

(20, 4), -- Nosferapti est de type Electrique

-- Pokémon avec deux types

(21, 1), -- Mewtwo est de types Feu et Eau

(21, 2), -- Mewtwo est de types Feu et Eau

(22, 3), -- Leveinard est de types Eau et Plante

(22, 6), -- Leveinard est de types Eau et Plante

(23, 5), -- Goupix est de types Roche et Vol

(23, 6), -- Goupix est de types Roche et Vol

(24, 4), -- Mew est de types Vol et Electrique

(24, 6), -- Mew est de types Vol et Electrique

(25, 5), -- Ronflex est de types Electrique et Roche

(25, 4), -- Ronflex est de types Electrique et Roche

(26, 2), -- Grodoudou est de types Plante et Eau

(26, 6), -- Grodoudou est de types Plante et Eau

(27, 1), -- Piafabec est de types Feu et Eau

(27, 2), -- Piafabec est de types Feu et Eau

(28, 3), -- Ramoloss est de types Eau et Plante

(28, 6), -- Ramoloss est de types Eau et Plante

(29, 5), -- Taupiqueur est de types Roche et Vol

(29, 6), -- Taupiqueur est de types Roche et Vol

(30, 4), -- Tadmorv est de types Electrique et Roche

(30, 5), -- Tadmorv est de types Electrique et Roche

-- Nouveaux Pokémon avec des dresseurs différents

(31, 4), -- Pikachu est de type Feu

(32, 4); -- Pikachu est de type Eau

INSERT INTO attaques (nom, puissance, `precision`, description, id\_type) VALUES

('Flammeche', 40, 100, 'Une boule de feu s\'échappe de la gueule du Pokémon.', 1), -- Type: Feu

('Lance-Flammes', 90, 85, 'Un puissant jet de flammes incinère l\'ennemi.', 1), -- Type: Feu

('Bulles d\'O', 35, 95, 'Le Pokémon crache des bulles d\'eau sur son adversaire.', 2), -- Type: Eau

('Hydrocanon', 110, 80, 'Un torrent d\'eau surpuissant balaye tout sur son passage.', 2), -- Type: Eau

('Fouet Lianes', 45, 90, 'Le Pokémon utilise ses lianes pour frapper l\'ennemi.', 3), -- Type: Plante

('Tempête Verte', 100, 75, 'Une violente tempête de feuilles vertes assaille l\'ennemi.', 3), -- Type: Plante

('Éclair', 50, 100, 'Une décharge électrique frappe l\'ennemi.', 4), -- Type: Electrique

('Tonnerre', 110, 70, 'Un coup de foudre dévastateur s\'abat sur l\'ennemi.', 4), -- Type: Electrique

('Jet de Pierres', 60, 95, 'Le Pokémon lance des pierres avec force sur son adversaire.', 5), -- Type: Roche

('Lance-Roches', 100, 80, 'Une pluie de rochers s\'abat violemment sur l\'ennemi.', 5), -- Type: Roche

('Coup d\'Aile', 35, 100, 'Le Pokémon frappe son adversaire avec ses ailes.', 6), -- Type: Vol

('Hurricane', 110, 70, 'Un puissant ouragan engloutit l\'ennemi.', 6), -- Type: Vol

('Poison', 30, 90, 'L\'ennemi est empoisonné par le Pokémon.', 7), -- Type: Poison

('Laser Glace', 95, 75, 'Un rayon de glace d\'une intensité extrême frappe l\'ennemi.', 8), -- Type: Glace

('Lame de Roc', 70, 95, 'Le Pokémon frappe l\'ennemi avec une lame tranchante de roche.', 5), -- Type: Roche

('Boule Roc', 25, 95, 'Le Pokémon lance une boule de roche sur son adversaire.', 5), -- Type: Roche

('Tranch\'Herbe', 55, 95, 'Le Pokémon coupe l\'ennemi avec ses lames d\'herbe.', 3), -- Type: Plante

('Lance-Boue', 55, 90, 'Une lance de boue visqueuse frappe l\'ennemi.', 2), -- Type: Eau

('Ébullition', 80, 100, 'Le Pokémon chauffe l\'eau à ébullition pour attaquer.', 2), -- Type: Eau

('Vibraqua', 60, 100, 'Des vibrations d\'eau puissantes frappent l\'ennemi.', 2), -- Type: Eau

('Fulmifer', 70, 90, 'Une décharge électrique puissante frappe l\'ennemi.', 4), -- Type: Electrique

('Change Éclair', 70, 100, 'Le Pokémon se déplace rapidement en frappant l\'ennemi.', 4), -- Type: Electrique

('Lance Flamme', 90, 85, 'Une lance enflammée est projetée vers l\'ennemi.', 1), -- Type: Feu

('Tempête Florale', 90, 100, 'Une tempête de fleurs attaque l\'ennemi avec grâce.', 3); -- Type: Plante

-- Pikachu connaît plusieurs attaques

INSERT INTO connait\_attaque (id\_pokemon, id\_attaque) VALUES

(4, 1), -- Pikachu connaît l'attaque Flamèche (de type Feu)

(4, 7), -- Pikachu connaît l'attaque Éclair (de type Electrique)

(4, 17), -- Pikachu connaît l'attaque Change Éclair (de type Electrique)

(4, 25), -- Pikachu connaît l'attaque Lance Flamme (de type Feu);

-- Salamèche connaît plusieurs attaques

(5, 1), -- Salamèche connaît l'attaque Flamèche (de type Feu)

(5, 12), -- Salamèche connaît l'attaque Coup d'Aile (de type Vol);

-- Mew connaît plusieurs attaques

(24, 2), -- Mew connaît l'attaque Lance-Flammes (de type Feu)

(24, 4), -- Mew connaît l'attaque Hydrocanon (de type Eau)

(24, 8), -- Mew connaît l'attaque Tonnerre (de type Electrique)

(24, 19), -- Mew connaît l'attaque Vibraqua (de type Eau);

-- Autres pokémons connaissant plusieurs attaques

(1, 3), -- Bulbizarre connaît l'attaque Fouet Lianes (de type Plante)

(2, 12), -- Salameche connaît l'attaque Coup d'Aile (de type Vol)

(3, 18), -- Carapuce connaît l'attaque Tempête Florale (de type Plante)

(6, 11), -- Rattata connaît l'attaque Hurricane (de type Vol)

(7, 9), -- Dracaufeu connaît l'attaque Lance-Roches (de type Roche)

(8, 15), -- Magicarpe connaît l'attaque Lame de Roc (de type Roche)

(9, 10), -- Roucool connaît l'attaque Tranch'Herbe (de type Plante)

(10, 8), -- Herbizarre connaît l'attaque Hydrocanon (de type Eau)

(11, 7), -- Aspicot connaît l'attaque Éclair (de type Electrique)

(12, 22), -- Miaouss connaît l'attaque Change Éclair (de type Electrique)

(13, 5), -- Raichu connaît l'attaque Tempête Verte (de type Plante)

(14, 16), -- Rapasdepic connaît l'attaque Boule Roc (de type Roche)

(15, 13), -- Abo connaît l'attaque Poison (de type Poison)

(16, 14), -- Tentacool connaît l'attaque Laser Glace (de type Glace)

(17, 21), -- Chenipan connaît l'attaque Fulmifer (de type Electrique)

(18, 20), -- Papilusion connaît l'attaque Ébullition (de type Eau)

(19, 23), -- Rattatac connaît l'attaque Tranch'Herbe (de type Plante)

(20, 24), -- Nosferapti connaît l'attaque Lance-Boue (de type Eau)

(21, 26), -- Mewtwo connaît l'attaque Change Éclair (de type Electrique)

(22, 27), -- Leveinard connaît l'attaque Piafabec (de type Feu)

(23, 28), -- Goupix connaît l'attaque Ramoloss (de type Eau)

(26, 30), -- Grodoudou connaît l'attaque Tadmorv (de type Electrique)

(27, 29), -- Piafabec connaît l'attaque Taupiqueur (de type Roche)

(28, 31), -- Ramoloss connaît l'attaque Pikachu (de type Electrique)

(29, 32), -- Taupiqueur connaît l'attaque Pikachu (de type Electrique)

(30, 33); -- Tadmorv connaît

**Requêtes SQL :**

**1)Sélectionnez les dresseurs ayant au moins 7 pokemons (JOIN, COUNT, GROUP BY, HAVING)**

SELECT d.id, d.nom, COUNT(p.id) AS nombre\_pokemons

FROM dresseur d

JOIN pokemons p ON d.id = p.id\_dresseur

GROUP BY d.id, d.nom

HAVING COUNT(p.id) >= 7;

**2) Selectionne les 5 pokemons les plus haut niveau (LIMIT, ORDER BY)**

SELECT \* FROM pokemons

ORDER BY niveau DESC

LIMIT 5;

**3)Afficher le nombres de pokemons pour chaque types ( COUNT, JOIN, GROUP BY)**

SELECT t.nom AS type, COUNT(p.id) AS nombre\_de\_pokemons

FROM types t

JOIN est\_de\_type edt ON t.id = edt.id\_type

JOIN pokemons p ON edt.id\_pokemon = p.id

GROUP BY t.nom;

**4) On veut les noms des dresseurs commençant par la lettre « P » (LIKE)**

SELECT \* FROM dresseur

WHERE nom LIKE 'B%';

**5) Afficher les noms des Pokémons appartenant aux types 'Plante' ou 'Eau' (IN, JOIN)**

SELECT p.nom AS nom\_du\_pokemon, t.nom AS type

FROM pokemons p

JOIN est\_de\_type edt ON p.id = edt.id\_pokemon

JOIN types t ON edt.id\_type = t.id

WHERE t.nom IN ('Plante', 'Eau');

**6) Afficher les pokemons dont les noms commencent par "R" et dont le niveau est compris entre 20 et 30 (LIKE,BETWEEN)**

SELECT \* FROM pokemons

WHERE nom LIKE 'R%' AND niveau BETWEEN 7 AND 10

**7) Augmenter les niveaux des pokémons dont le nom commence par "S" de 10. (UPDATE, LIKE)**

UPDATE pokemons

SET niveau = niveau + 10

WHERE nom LIKE 'S%';

**8) Afficher les noms des dresseurs qui ont des Pokémon dont le niveau est supérieur à la moyenne des pokemons. Utilisez Union. (DINSTINCT,JOIN,AVG, requete imbriqué, UNION)**

SELECT DISTINCT d.nom

FROM dresseur d

JOIN pokemons p ON d.id = p.id\_dresseur

WHERE p.niveau > (SELECT AVG(niveau) FROM pokemons)

UNION

SELECT DISTINCT d.nom

FROM dresseur d

JOIN pokemons p ON d.id = p.id\_dresseur

WHERE d.id IN (

SELECT DISTINCT d.id

FROM dresseur d

JOIN pokemons p ON d.id = p.id\_dresseur

WHERE p.niveau > (SELECT AVG(niveau) FROM pokemons)

);

**9) Une erreur a été fait lors de la création de la base de donnée. En effet, nous avons crée une table « dresseur » au lieu de « dresseurs ». Afin de rester cohérent avec les noms des autres tables, nous aimerions palier à cela. (CREATE TABLE , INSERT INTO, SELECT, DROP TABLE). Pour assurer l’intégrité des données, nous allons utiliser les transactions.**

CREATE TABLE dresseurs(

id INT PRIMARY KEY AUTO\_INCREMENT,

nom VARCHAR(50) NOT NULL,

age INT,

genre VARCHAR(1)

);

INSERT INTO dresseurs(nom, age, genre)

SELECT nom, age, genre FROM dresseur;

DROP TABLE dresseur;

**10) Nous avons 3 pikachu, nous aimerions supprimé celui appartenant au dresseur Charlie. (DELETE, SELECT imbriqué)**

DELETE FROM pokemons

WHERE nom = "pikachu" AND id\_dresseur = (SELECT id FROM dresseur WHERE nom = "Charlie");

**11) Sélectionner les noms des dresseurs en fonction du genre en utilisant CASE. (DISTINCT,JOIN,CASE)**

SELECT DISTINCT d.nom

FROM dresseur d

JOIN pokemons p ON d.id = p.id\_dresseur

WHERE CASE

WHEN d.genre = 'F' THEN 1

ELSE 0

END = 1;

**12) On aimerait ajouter une table qui permet d’enregistrer les resultats des combats entre les dresseurs. Pour ce faire, on crée une table combat avec l’id du dresseur\_vainqueur et du dresseur\_perdant et le match\_nul.**

CREATE TABLE combat (

id INT PRIMARY KEY AUTO\_INCREMENT,

id\_dresseur\_vainqueur INT NOT NULL REFERENCES dresseurs(id),

id\_dresseur\_perdant INT NOT NULL REFERENCES dresseurs(id),

match\_nul INT NOT NULL

);

**13) Finalement on voudrait aussi avoir la date du combat dans la table. (ALTER TABLE)**

ALTER TABLE combat

ADD COLUMN date\_combat DATE NOT NULL;

**14) On veut inserer quelques lignes de données pour tester la table. (INSERT INTO)**

INSERT INTO combat (id\_dresseur\_vainqueur, id\_dresseur\_perdant, match\_nul, date\_combat)

VALUES

(1, 2, FALSE, '2024-01-20'), -- Victoire du dresseur 1 sur le dresseur 2

(3, 4, FALSE, '2024-02-05'), -- Victoire du dresseur 3 sur le dresseur 4

(2, 5, TRUE, '2024-03-10'); -- Match nul entre le dresseur 2 et le dresseur 5

**15) Maintenant qu’on a verifié que la table fonctionnait correctement, on aimerait vider la table. (TRUNCATE TABLE)**

TRUNCATE TABLE combat;

**16) Affiche moi le plan d’exécution de la requête SQL « SELECT \* FROM dresseurs » (EXPLAIN)**

EXPLAIN SELECT \* FROM dresseurs;

**17) Rajouter un dresseur s’appelant Charlie, ayant 18 ans qui est de genre N. On souhaite maintenant supprimer le dresseur Charlie ayant 18 ans que l’on vient d’ajouter. (SELECT, DELETE, WHERE, LIMIT, ORDER BY, TRANSACTION)**

INSERT INTO dresseurs(nom, age, genre) VALUES(‘Charlie’, 18, ‘N’) ;

SET autocommit=0 ;

SELECT \* FROM dresseurs ;

DELETE FROM dresseurs WHERE nom = ‘Charlie’ AND age = 18 AND genre = ‘N’ ;

SELECT \* FROM dresseurs ;

ROLLBACK ;

SELECT \* FROM dresseurs ;

DELETE FROM dresseurs WHERE id = (SELECT id FROM dresseurs WHERE nom = ‘Charlie’ AND age = 18 AND genre = ‘N’ ORDER BY id DESC LIMIT 1) ;

SELECT \* FROM dresseurs ;

COMMIT ;

SET autocommit=1 ;

**18) Créer une vue où l’on veut uniquement les dresseurs de genre masculin dans la table des dresseurs.**

CREATE VIEW vue\_dresseurs AS SELECT \* FROM dresseurs WHERE genre = 'H';

**Pour afficher la table on aura juste à mettre :**

SELECT \* FROM vue\_dresseurs;

**19) Créons une vue qui rassemble des informations sur les Pokémon, les dresseurs, les types et les attaques, pour qu’on puisse y acceder plus facilement.**

CREATE VIEW Infopokemons AS

SELECT

p.id AS id\_pokemon,

p.nom AS nom\_pokemon,

p.niveau,

p.exp\_en\_cours,

d.id AS id\_dresseur,

d.nom AS nom\_dresseur,

t.nom AS type,

a.nom AS attaque

FROM pokemons p JOIN dresseurs d ON p.id\_dresseur = d.id

JOIN est\_de\_type et ON p.id = et.id\_pokemon

JOIN types t ON et.id\_type = t.id

JOIN connait\_attaque ca ON p.id = ca.id\_pokemon

JOIN attaques a ON ca.id\_attaque = a.id;

SELECT \* FROM Infopokemons;

**Documentations sur les formes normales et comment on les a appliqués :**

Dans ce TP, nous avons dû respecter les 3 premières règles de normalisation.

Donc dans un premier temps, nous avons chercher à respecter la 1ère règle de normalisation qui consiste à seulement avoir des données atomiques (c’est-à-dire, qui ne contient qu’une seule information). Pour cela, on a éviter de créer une colonne dans la table attaques qui contient les statistiques de l’attaque car elle aurait contenu la puissance de l’attaque ainsi que la précision.

Par la suite, il nous a fallu réfléchir à la 2ème règle de normalisation qui stipule que chaques colonnes doit dépendre de l’entièreté de la clé composé. Dans notre cas, à chaque fois qu’on avait une clé composé de plusieurs informations, nous n’avions pas d’autres colonnes dans la table ce qui nous a permis de ne pas nous soucier de cette règles normales.

Finalement, la 3ème règle de normalisation veut que chaque attribut non clé dépendent directement de la clé primaire. Et dans notre cas, c’est respecté car nous avons séparé dans des tables différentes chaques éléments qui compose la base (les dresseurs, les pokémons, les attaques, etc). Ainsi, par exemple dans la table dresseurs, tous les attributs dépendent directement de la clé primaire et cela est respectée dans chacunes des tables de la base de donnée.

**Choix des 2 notions supplémentaires :**

Pour ce qu’il s’agit des notions supplémentaires, nous avons dans un premier choisit d’apprendre à utiliser les transactions. C’est très utile lorsqu’on a besoin d’exécuter plusieurs requêtes dont certaines dépendront du résultat des requêtes précédentes car il permet d’annuler toutes les requêtes précédentes si seulement une échoue. Par exemple vous avez sûrement déjà copier coller un script de base de donnée dans phpmyadmin mais qu’il y avait une erreur à un moment donnée dans le script. Une fois cette erreur corrigée, vous ne pouvez pas rééxécuter le même script car il vous dira que les premières commandes ont déjà été éxécuté. Avec les transactions, cela peut être évité. On peut s’assurer après chaque commandes qu’elles ont bien eu l’effet esconté puis une fois satisfait il nous suffit de commit ou si l’on ne l’est pas de faire un rollback.

Nous utilisons les transactions dans la requête SQL 17.

Et pour la deuxième notion, nous avons choisit la création de vues. Elle consiste a stocker une requête SELECT en lui donnant un nom, et on peut ensuite appeler directement la vue par son nom Les vues sont des représentations virtuelles de données construites à partir d'une requête SELECT spécifique. Elles ne stockent pas les données qu'elles contiennent mais conservent juste la requête permettant de les créer. En créant une vue, on attribue un nom à cette requête, ce qui permet d'accéder aux résultats de la requête en se référant simplement au nom de la vue. En d'autres termes, une vue offre une perspective structurée et prédéfinie sur les données existantes, sans nécessiter de stockage supplémentaire. Donc on stocke une requête SELECT en lui donnant un nom, et on peut ensuite appeler directement la vue par son nom.

Nous utilisons les vues dans les requêtes SQL 18 et 19.