TP : Pokemon

2026-2027

Thibaud Lemanissier

PC Lycée Carnot

On dispose d'une base de donnée contenant des informations sur le premier jeu Pokémon. Ce jeu ne tenait sur une carte qui ne pouvait contenir qu'un mégaoctet. Il faut donc faire en sorte que les données prenne le moins de place possible. Les données sont donc séparé en plusieurs tables. Voici le code Capytale pour accéder à la base de donnée : 1ea4-11541476

La table pokedex

On commence par se focaliser sur la table pokedex dont le schéma est le suivant :

pokedex(id : int, nom : str, categorie : str, taille : float, poids : float)

Plus précisément, cette table contient les attribues :

— id : l'identifiant d'une espèce de Pokémon sous la forme d'un entier ;
— nom : le nom d'une espèce de Pokémon sous la forme d'une chaîne de caractères ;
— categorie : le nom de la catégorie du Pokémon sous la forme d'une chaîne de caractères ;
— taille : la taille d'un Pokémon en mètre sous la forme d'un flottant ;
— poids : le poids d'un Pokémon en kilogramme sous la forme d'un flottant.

Voici les premières lignes :

id	nom	categorie	taille	poids
1	Bulbizarre	Graine	0.7	6.9
2	Herbizarre	Graine	1.0	13.0
3	Florizarre	Graine	2.0	100.0
4	Salameche	Lézard	0.6	8.5

Écrire les requêtes SQL permettant d'obtenir les données suivantes :

1. Donner une clé primaire de cette table.
2. Afficher toutes les noms de Pokémon avec leur catégorie.
3. Afficher les noms de Pokémon de la catégorie Dragon avec leurs tailles.
4. Afficher les noms de Pokémon de la catégorie Dragon avec une taille supérieure ou égale à 1.8.
5. Afficher les noms de Pokémon de la catégorie Dragon ou qui ont une taille supérieure ou égale à 1.8.
6. Afficher les noms de Pokémon dont le nom contient « flor ».
7. Afficher l'ensemble des catégorie de Pokémons sans répétition.
8. Afficher le nombre de catégories distinctes de Pokémons.
9. Quelles sont les noms de Pokémon ayant un IMC ($poids/taille^2$) compris entre 18.5 et 25 inclus ? On voudra récupérer deux colonnes : une colonne espece et une colonne nommée IMC rangé par IMC croissant.
10. Afficher la taille du plus grand Pokémon, avec le poids moyen de tous les Pokémons et le nombre de Pokémons.
11. Afficher le poids minimal des Pokémons qui ont une taille supérieur ou égal à 1.5 m.

Union, intersection et différence

On considère en plus la table suivante :

evolue_en(pokemon_base_id : int, pokemon_evol_id : int, niveau : int)

Plus précisément, cette table contient les attribues :

— pokemon_base_id : l'identifiant d'un Pokémon qui peut évoluer ;
— pokemon_evol_id : l'identifiant d'un Pokémon qui est une évolution ;
— niveau : le niveau de l'évolution (si l'évolution nécessite un objet, le niveau est à -1).

Voici les premières lignes :

pokemon_base_id	pokemon_evol_id	niveau
1	2	16
2	3	32
4	5	16
5	6	36

12. Donner une clé primaire de cette table. Attention ! Évoli peut évoluer en 3 Pokémons différents.
13. Afficher les identifiants des Pokémons qui sont des évolutions d'autres Pokémons et qui peuvent évoluer.

On considère en plus la table suivante :

detient_pokemons(dresseur_id : int, pokemon_id : int, niveau : int)

Plus précisément, cette table contient les attribues :

— dresseur_id : l'identifiant d'un dresseur ;
— pokemon_id : l'identifiant d'un Pokémon ;
— niveau : le niveau du Pokémon.

Voici les premières lignes :

dresseur_id	pokemon_id	niveau
2	74	12
2	95	14
3	120	18
3	121	21

14. Donner une clé primaire pour cette table.
15. Donner les identifiants des Pokémons qui ne sont possédés par aucun dresseur.
16. Donner les identifiants des Pokémons qui peuvent évoluer ou qui appartiennent à un dresseur.

Jointure et auto-jointure

Nous allons pouvoir commencer à faire des jointures entre plusieurs tables :

17. Afficher le nom de chaque pokémon avec son niveau d'évolution.

On considère en plus les deux tables suivantes :

detient_attaques(attaque_id : int, pokemon_id : int)

Plus précisément, cette table contient les attribues :

— attaque_id : l'identifiant d'une attaque ;
— pokemon_id : l'identifiant d'un Pokémon.

Voici les premières lignes :

attaque_id	pokemon_id
1	23
1	24

attaques(id : int, libelle : str, type_id : int, pp : int, puissance : int, precis : int)

Plus précisément, cette table contient les attribues :

— id : l'identifiant de l'attaque ;
— libelle : le nom de l'attaque ;
— type_id : l'identifiant de son type (que l'on utilisera pas dans ce TP) ;

— pp : le nombre de point de pouvoir dont dispose cette attaque (c'est-à-dire le nombre de fois que l'on peut l'utiliser) ;

— puissance : la puissance de l'attaque ;

— precis : la précision de l'attaque.

Voici les premières lignes :

id	libelle	type_id	pp	puissance	precis
1	Abîme	13	5	0	30
2	Acidarmure	10	40	0	100
3	Acide	10	30	40	100
4	Adaptation	8	40	0	100
5	Affûtage	8	30	0	100
6	Amnésie	11	20	0	100

18. Déterminer des clés primaires les deux nouvelles tables.
19. Afficher les noms des Pokémons avec le noms de leurs attaques qui ont une puissance non nulle.

On peut maintenant faire des auto-jointures ainsi que plus généralement des jointures faisant intervenir plusieurs fois une même table :

20. Affiche tous les couples d'attaques qui ont la même puissance sans qu'il y ai de répétition de couple.
21. Afficher chaque nom de pokémon avec le nom de son évolution et son niveau d'évolution.

Regroupement

On peut maintenant regrouper les enregistrements pour obtenir des informations plus globales.

22. Afficher chaque catégorie de Pokémon avec le nombre de Pokémon de cette catégorie.
23. Afficher le nom chaque dresseur avec le nombre de Pokémons distincts qu'il détient.
24. Afficher le nom chaque Pokémon avec la puissance de sont attaque la plus puissante parmi les Pokémons qui ont plus de 40 attaques.

Requêtes imbriquées

On peut encore faire quelques requêtes imbriquées.

25. Afficher les attaques qui ont une puissance supérieur à la moitié de l'attaque la plus puissante.
26. Afficher le nom des Pokémons avec le nombre de Pokémon de cette espèce faut-il pour qu'ils pèsent le même poids que le Pokémon le plus lourd.

Compléter le pokedex

Les tables restantes sont se trouvent dans le diagramme à la fin du document. Vous pouvez l'utiliser pour résoudre ces dernières questions de difficultés variables.

27. Afficher les pokémons détenus par Pierre.
28. Afficher l'identifiant des dresseurs qui ont un Pokémon en commun sans répétition. On affichera l'identifiant du premier dresseur, l'identifiant du deuxième, l'identifiant du pokémon en commun (si vous voulez faire apparaître le nom des dresseurs et Pokémon vous devez faire trois jointures de plus...).
29. Déterminer les pokémons qui perdent du poids en évoluant.
30. Déterminer le Pokémon qui possède le plus d'attaques possibles.

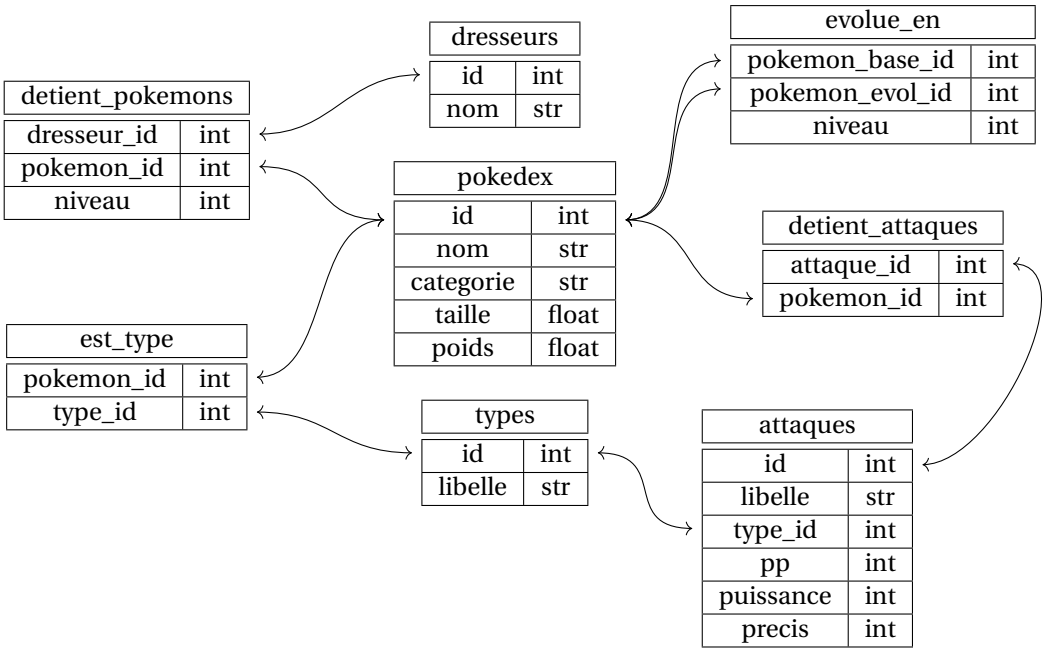

FIGURE 1 – Schéma des tables avec leurs relations