Info - 2A - TP1

Base de données & DAO

Objectif du TP : Créer une première application qui communique avec la base de données

1. Configuration de VS Code

Nous allons utiliser Visual Studio Code.

Récupérer le squelette du TP1 (**squelette-tp1.zip**) sur moodle. Dézippez l'archive.

Lancez VS Code.

Allez dans Fichier > Préférences > Paramètres : dans la fenêtre d'édition de droite, ajoutez **entre les accolades** :

"telemetry.enableTelemetry": false

Ouvrez le répertoire obtenu (Fichier > Ouvrir le dossier …)

Dans le menu vertical à gauche, allez dans les extensions (icone carrée), recherchez Python (avec le moteur de recherche en haut) et installez la première extension de la liste (qui se nomme **python**)

Installez autopep8 et pylint : dans le menu en haut allez dans « Afficher » > « terminal intégré »

Tapez la ligne de commande suivante :

```
pip install autopep8 pylint --user --proxy http://pxcache-02.ensai.fr:3128
```

Pour pouvoir lancer un script python facilement, allez dans *Fichier* > *Préférences* > *Raccourcis Clavier*

Dans la fenêtre qui s'ouvre, cliquez sur *keybindings.json*

Dans la fenêtre de droite mettez le contenu suivant :

```
// Place your key bindings in this file to overwrite the defaults
[
    {"key": "ctrl+r", "command":"python.execInTerminal"}
]
```

Testez votre configuration :

Créer un fichier python (*test.py*) avec un contenu simple, par exemple :

print('Hello World')

Vérifier que vous avez bien Hello World qui s'affiche dans votre terminal

Puis supprimez *test.py*

2. Ecriture des données

a. Utilisation du DAO Book

Dans le squelette vous avez un exemple de DAO (*dao_book.py*) qui permet d'enregistrer en base des livres (méthode *create*) et de lister les livres enregistrés.

Vous avez un fichier *connection.py* où est configuré la connexion à la base de données et vous avez un fichier *main.py* qui contient un petit programme qui crée un livre et liste les livres créés.

Modifiez le fichier *connection.py* pour remplacer la valeur **MY_ID** par votre identifiant ENSAI.

Dans le repertoire sql vous avez un fichier *create_book.sql*. Sur l'ENT, allez dans mes applications > postgresql

Note: l'id d'un livre et de type *serial*. Le type serial intègre une séquence. L'identifiant est donc généré automatiquement lors de l'insertion.

Connectez vous avec votre identifiant (le mot de passe est aussi votre identifiant), et dans votre base (qui a pour nom votre identifiant), exécutez le contenu du fichier *create_book.sql*

Maintenant que vous avez créé votre table **book** vous pouvez y insérer des livres.

Installez **psycopg2** (*pip install pyscopg2-binary*) pour pouvoir se connecter et consulter votre base de donnée.
Vous pouvez retrouver la documentation de psycopg2 ici : http://initd.org/psycopg

```
pip install psycopg2-binary --user --proxy http://pxcache-02.ensai.fr:3128
```

Lancez le programme *main.py* (Cliquez dessus et faites Ctrl + R ou clic droit > « Exécuter le script python dans un terminal »)

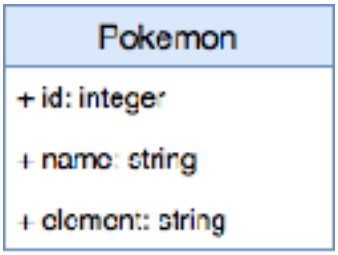
Dans la console, vous devez voir qu'un livre a été créé.

Depuis phpPgAdmin, consultez votre table book et vérifiez que le livre « A la recherche du temps perdu » est bien présent

b. Création des pokemons

En se basant sur l'exemple donné avec les livres, nous allons créé des pokemons.

Dans le répertoire business_object, créez un fichier pokemon.py et définissez une classe Pokemon. Un Pokemon aura un identifiant nom et un element (eau, feu …)



Dans le repertoire sql, créez un fichier *create_pokedex.sql* dans lequel vous écrivez la création de la table *Pokedex* qui vous permettra d'enregistrer vos pokemons. Exécutez le sur votre base.

Dans le répertoire dao, créez un fichier *dao_pokedex.py*

Dans ce fichier, de façon similaire à *dao_book*, créez une méthode ***create*** et une méthode ***get_all_pokemons*** qui vous permettent de créer des pokemons et de lister les pokemons créés.

Pour pouvoir communiquer avec le base de données, nous récupérons des *cursors* qui nous permettent d'exécuter des requêtes sql dans une transaction.
Lorsque l'on fait un *commit*, on valide la transaction. Si un problème survient on peut faire un *rollback*, on fait alors un retour arrière sur l'état au début de la transaction, ce qui permet d'avoir un état cohérent en base de données.

Avec psycopg2 lorsque l'on utilise with :

```
with connection.cursor() as cur:
```

Implicitement, commit() et close() sont exécutés (à la fin du block) et rollback si un exception est levée.

Créer un fichier main_pokemon.py dans lequel vous créez 3 pokemons, par exemple un *pikachu*, un *dracaufeu* et un *bulbizarre* et listez les pokemons créés avec la méthode *get_all_pokemons*

c. Suppression et mise à jour

Nous allons ajouter 2 méthodes dans notre dao pokedex.

La première :

- *update* qui prend en paramètre un pokemon (modifié) et qui met à jour le nom et l'élément de notre pokemon en base

Pour tester notre méthode, après avoir créer bulbizarre, modifiez le en changeant son nom par « Magicarpe » et son élément par « Eau » et appelez la méthode update de votre dao. Vérifiez que pour les 3 derniers pokemons, vous avez un magicarpe à la place du bulbizarre

La seconde :

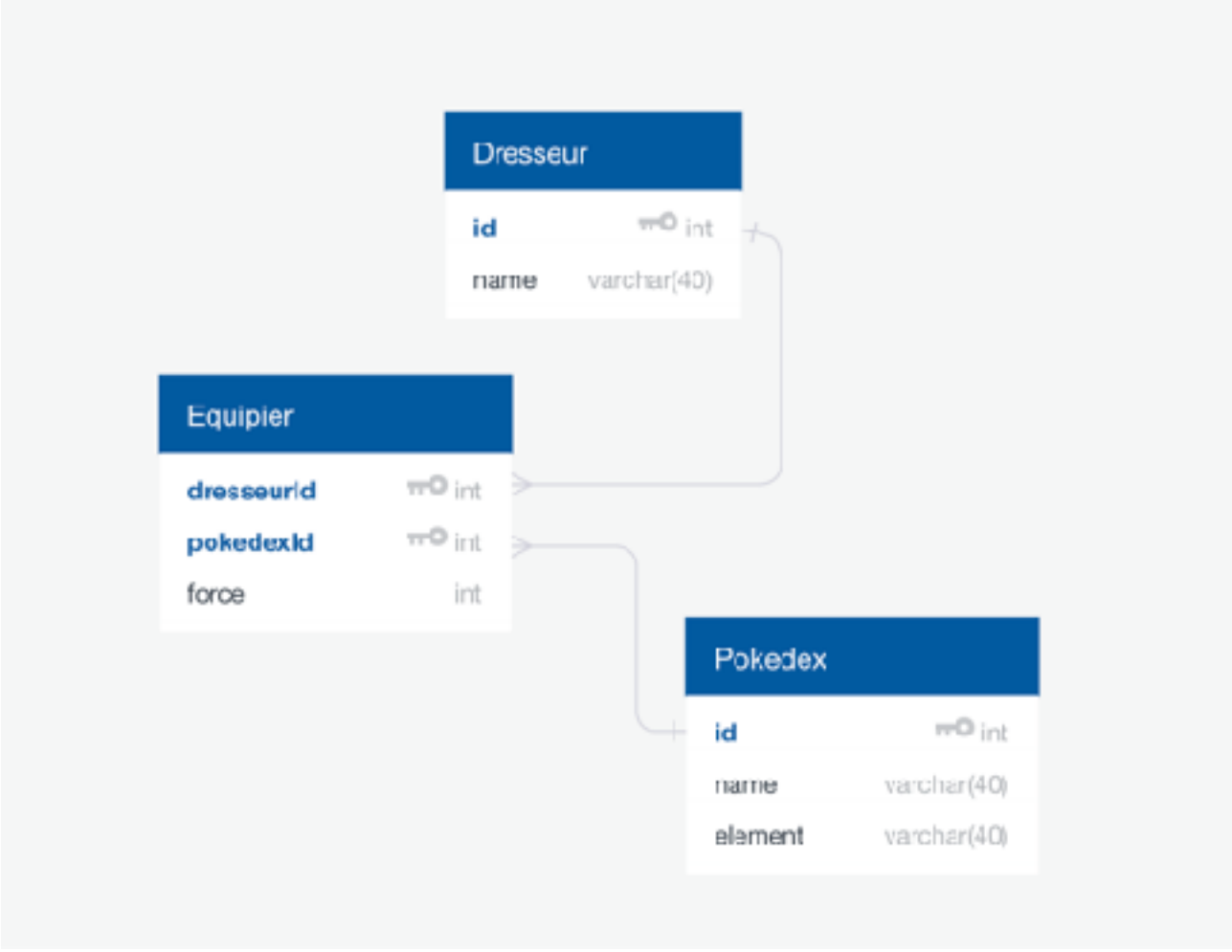
- d*elete* qui prend en paramètre un pokemon et qui le supprime de la base

Pour tester votre méthode, supprimez un des pokemons après l'avoir créé et vérifiez qu'il n'apparait pas dans votre liste

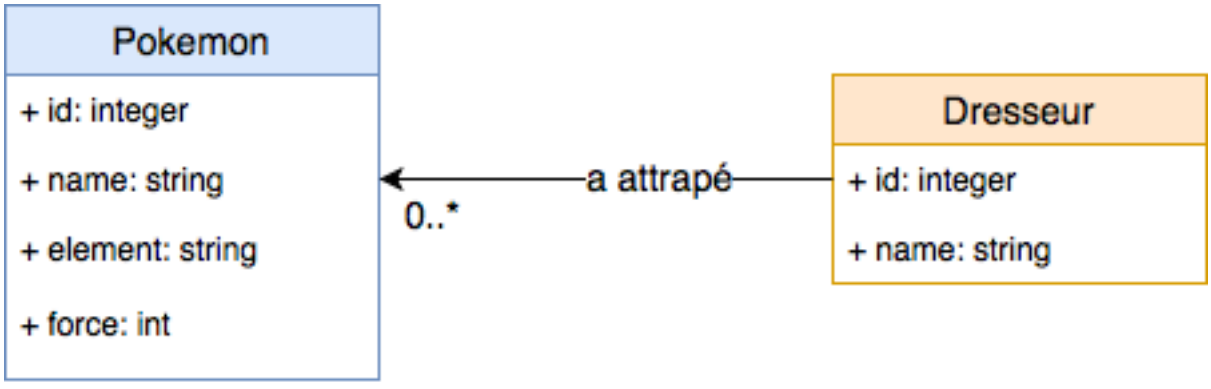
d. Dresseurs

Chaque dresseur de pokemons a une liste de pokemons. Leurs pokemons différent parce qu'ils ont une force (exprimée avec une valeur entière) différentes.

Nous allons donc créer 2 nouvelles tables *Dresseur* et *Equipier* pour enregistrer les dresseurs et lui associer des pokemons



Nous allons avoir une nouvelle classe Dresseur qui a une liste de pokemons et ajouter le champ force à notre classe Pokemon (qui vaudra par défaut 0)



Créez les 2 tables *Equipier* et *Dresseur*

Puis créer le classe *Dresseur* (*dresseur.py* dans le rep business_object)

Et enfin créez le dao *dao_dresseur.py* qui permet d'enregistrer un dresseur et ses équipiers (méthode create) et de lister les dresseurs avec leurs équipiers (méthode *get_all_dresseurs*)