Tâches à faire :

TÂCHE 1 : CRÉER UNE BASE DE DONNÉES POSTGRESQL A PARTIR DU FICHIER PRIZE.JSON

Lorsque vous regardez le fichier prize.json :

- Comment pouvez-vous représenter les données de manière structurée ?
- De combien de tables allons-nous avoir besoin pour ce fichier ?
- Quels sont ces tableaux et quelles colonnes contient chaque tableau ?

Dans la première tâche, vous allez créer un parseur JSON-SQL à l'aide de NodeJS. Après avoir analysé les données JSON, créez les tables qui représentent le mieux ces données de manière structurée tout en tenant compte des colonnes de clé primaire et de clé étrangère.

Après avoir créé les tables, vous devez les remplir avec des données dans le fichier JSON. Créez un parseur NodeJS qui itère dans le tableau JSON et insérez les données dans les tables correspondantes tout en respectant les associations. Assurez-vous de ne pas insérer d'éléments en double dans la base de données.

- Commencez par créer un script SQL **nobelprize.sql** dans lequel vous créez toutes les tables.
- Créez un fichier **parser.js** dans lequel vous implémentez votre parseur. Notez que le fichier **parser.js** ne contient pas de serveur Web et que vous pouvez l'exécuter dans votre terminal: **node parser.js**

TÂCHE 2 :REST API

- **Votre API doit offrir les fonctionnalités suivantes :**
    - **F1:** Lister tous les lauréats (id, prénom, nom).
    - **F2:** Étant donné un identifiant, affichez les informations du lauréat avec cet identifiant (prénom, nom, les prix remportés).
        - Ex: ID = 6
            - Marie Curie
                - 1911 chemistry in recognition of her services to the…
                - 1903 physics in recognition of the extraordinary services…
    - **F3:** Combien ont remporté plus d'un prix Nobel ?
        - (prénom, nom, nombre de prix gagnés)
        - Ex: Marie Curie, 2
    - **F4:** Lister toutes les catégories des prix nobel
        - Chemistry, economics, etc

- **F5**: Déterminez quelle catégorie a produit le plus grand nombre de lauréats du prix Nobel.
- **F6:** Pour chaque année, indiquez combien de lauréats avaient remporté un prix nobel.
  - Ex: 2021, 13
- **F7:** Afficher toutes les années au cours desquelles aucun prix Nobel n'a été décerné.
- **F8**: Afficher toutes les années de prix nobel triées par nombre de lauréats ascendant/descendant.
  - ?sort=asc_laureates ⇒ ascendant ⇒ commencer par les années avec le plus petit nombre de lauréats
  - ?sort=desc_laureates ⇒ descendant ⇒ commencer par les années avec le plus grand nombre de lauréats
  - Exclure les années où aucun prix Nobel n'a été décerné

- **F9:** Supprimer un lauréat avec un identifiant donné.
- **F10:** Mettre à jour la motivation d'un lauréat avec un identifiant donné dans une année donnée et une catégorie donnée.

Dans le projet R301, vous avez créé les routeurs, contrôleurs et services. Dans ce projet, la majorité des changements concerneront les services. Les routeurs et les contrôleurs n'ont besoin que de légères modifications pour certaines questions. Vous avez déjà répondu à toutes les questions du projet précédent, ce qui va changer dans celui-ci, c'est que les services liront désormais les données d'une base de données PostgreSQL au lieu d'un fichier JSON.
Les données envoyées aux contrôleurs par les services ne changeront généralement pas (vous pouvez avoir de très légers changements à partir de certaines questions dans lesquelles plus d'informations sont demandées).

Assurez-vous que vos routeurs et contrôleurs fonctionnent bien dans le projet R301.

Notes IMPORTANTES:

- Aucune documentation SWAGGER n'est requise (bien que si cela fonctionne dans le projet R301, rien ne changera dans celui-ci).
- Vous pouvez tester votre API en utilisant POSTMAN.
- Aucune vue n'est demandée dans ce projet. Uniquement une API qui renvoie JSON.
- Vous n'êtes pas autorisé à utiliser un ORM dans cette activité.

Comment présenter ce projet ?

Correction instantanée

Si vous avez terminé le projet avant la fin de la deuxième séance, vous pouvez appeler l'instructeur et il évaluera le projet en place.
Ce qui est recommandé :

- Présenter le script de création des tables et le parseur à la fin de la première séance.
- Présenter l'API à la fin de la deuxième séance.

L'instructeur vérifiera :

- Votre script sql pour la création de tables
    - Le nom des tables, les colonnes de chaque table et les associations doivent être clairs.
- Votre parseur JSON-SQL, vos tables postgreSQL doivent être remplies avec les données initiales.
- Routes d'API fonctionnelles.
- L'instructeur peut demander à vérifier les requêtes dans vos services/controlleurs NodeJS.
- L'instructeur peut vous demander d'expliquer la structure de votre projet et le rôle de chaque fichier/dossier.