Objectif

Ce test est conçu pour mettre en valeur votre compréhension des bases de données et du traitement des données, ainsi que votre aptitude dans un langage de programmation de votre choix.

Prérequis

- Connaissance des bases de données relationnelles. Pour les besoins de ce test, nous utilisons MySQL.
- Connaissance d'un langage de programmation : comment lire et écrire des fichiers, traiter des données et accéder à une base de données MySQL.
- Familiarité avec Docker pour la gestion des conteneurs, que nous utilisons à travers l'outil Docker Compose. Vous aurez besoin de Docker et de Docker Compose installés sur votre machine de développement.
- Familiarité avec Git pour le contrôle de source et un compte github.com qui sera utilisé pour partager votre code.

Nous avons inclus des données d'exemple et un code de programme. Le schéma d'exemple crée une table simple, avec un code d'exemple dans plusieurs langages de programmation courants pour charger des données à partir d'un fichier CSV et les exporter dans un fichier JSON. Il y a des instructions vers le bas de ce document expliquant comment utiliser les conteneurs Docker, démarrer la base de données et utiliser les exemples.

Contexte

Nous avons fourni un dépôt Github contenant :

- Un fichier docker compose.yml qui configure un conteneur pour la base de données MySQL et les conteneurs de scripts d'exemple.
- Un dossier images contenant des programmes d'exemple montrant comment accéder à la base de données à partir de C, Node, Python, R, Ruby et Swift.
- Un fichier example_schema.sql contenant un schéma de table utilisé par les scripts d'exemple.
- Un dossier data contenant quatre fichiers :
  - example.csv Un petit ensemble de données utilisé par les scripts d'exemple.
  - places.csv 113 lignes, où chaque ligne contient le nom d'une ville, d'un comté et d'un pays.
  - people.csv 10 000 lignes, où chaque ligne contient un prénom, un nom de famille, une date de naissance et une ville de naissance.
  - sample_output.json Fichier de sortie d'exemple, pour montrer à quoi votre sortie devrait ressembler.

Problème

Voici les étapes à accomplir :

- Concevez un schéma de base de données pour contenir les données des fichiers CSV people et places, et appliquez-le à la base de données MySQL. Vous pouvez appliquer ce schéma via un script, via le client de ligne de commande MySQL ou via un client GUI.
- Créez une image Docker pour charger les fichiers CSV places.csv et people.csv dans les tables que vous avez créées dans la base de données. Assurez-vous que la configuration appropriée est dans le fichier docker-compose. Votre processus d'ingestion de données peut être implémenté de n'importe quelle manière, tant qu'il s'exécute dans un conteneur Docker.

Vous pouvez implémenter cela via un code de programme dans une langue de votre choix ou via l'utilisation d'outils ETL.

- Créez une image Docker pour produire un résumé du contenu de la base de données. Vous pouvez l'implémenter en utilisant un langage de programmation de votre choix. La sortie doit être au format JSON et être écrite dans un fichier du dossier data appelé data/summary_output.json. Il devrait se composer d'une liste des pays et d'un compte du nombre de personnes nées dans ce pays. Nous avons fourni un fichier de sortie d'exemple data/sample_output.json pour comparer votre fichier.

Nous avons fourni un exemple de schéma et de code qui montre comment gérer une ingestion de données simple et une sortie. Les détails sur la façon de démarrer et de se connecter à la base de données sont ci-dessous, ainsi que sur la façon d'utiliser le schéma et le code d'exemple.

Notes

- Il n'y a pas de meilleur façon de faire. Nous nous intéressons aux choix que vous faites et à votre processus de développement.
- Lorsque vous créez un conteneur, assurez-vous d'ajouter la configuration du conteneur au fichier docker-compose.yml, et d'ajouter votre Dockerfile et votre code au dossier images.
- Assurez-vous que votre code est exécutable, et si vous travaillez dans un langage de script, assurez-vous que votre script contient une ligne "hash-bang" appropriée (comme dans nos exemples de scripts).
- La plupart du code d'exemple utilise des bibliothèques ORM pour se connecter à la base de données. Ce n'est pas essentiel pour l'objectif de ce test : votre code doit se connecter à la base de données et vos requêtes doivent être implémentées de la manière qui vous convient le mieux.
- Réfléchissez à quel type de gestion d'erreurs et de tests est approprié.
- Toutes les données d'entrée, de stockage et de sortie doivent être en UTF-8. Attendre des caractères multioctets dans les données.
- Le stockage de la base de données MySQL est éphémère ; il ne persistera pas, assurez-vous donc que toutes les requêtes de schéma et de données sont répétables.
- Vous pouvez trouver plus facile de travailler avec un sous-ensemble des données lors du développement de votre ingestion.

Exigences

Assurez-vous d'avoir des versions récentes de Docker et Docker Compose.

Construction des images

Cela construira toutes les images référencées dans le fichier Docker Compose. Vous devrez le relancer après avoir apporté des modifications au code. (Vous pouvez également spécifier des services individuels à construire si cela est plus pratique.)

docker compose build

Démarrage de MySQL

Cela prendra un court instant pour exécuter les scripts de démarrage de la base de données.

docker compose up database

Facultatif : si vous souhaitez vous connecter à la base de données MySQL via le client en ligne de commande. Cela peut être utile pour consulter le schéma ou les données de la base de données.

docker compose run database mysql --host=database --user=user_dteng --password=cirilgroupt db_dteng

Scripts d'exemple

Nous avons fourni du code d'exemple écrit Python qui montre comment utiliser un programme dans un conteneur Docker séparé pour se connecter à la base de données, en utilisant une bibliothèque ORM si nécessaire, pour charger des données à partir d'un fichier CSV et interroger les données pour les exporter sous forme de fichier JSON. Celles-ci doivent être considérées comme illustratives; il est possible d'utiliser l'un de ces exemples comme base pour votre propre solution, mais nous préférons que vous utilisiez les technologies avec lesquelles vous vous sentez à l'aise.

Assurez-vous que la base de données MySQL fonctionne, puis chargez le schéma d'exemple avec :

docker compose run --no-TTY database mysql --host=database --user=user_dteng --password=cirilgroupt db_dteng <example_schema.sql

Assurez-vous ensuite que les conteneurs ont été construits avec docker compose build et exécutez un ou plusieurs des programmes d'exemple avec :

docker compose run example-c
docker compose run example-node
docker compose run example-python
docker compose run example-r
docker compose run example-ruby
docker compose run example-swift

Dans chaque cas, le programme charge les données du fichier data/example.csv dans cette table et exporte les données de la table de la base de données sous forme de fichier JSON dans le dossier data. Notez que les scripts ne tronquent pas la table, donc chaque fois que vous en exécutez un, du contenu supplémentaire sera ajouté.

Nettoyage

Pour nettoyer en arrêtant tous les conteneurs et en les supprimant.

docker compose down