Travaux pratiques : ETL à l'aide de scripts shell

Temps estimé nécessaire : **30** minutes

Objectifs

Après avoir terminé ce laboratoire, vous serez capable de :

- Extraire des données d'un fichier délimité.
- Transformer les données textuelles.
- Charger des données dans une base de données à l'aide de commandes shell.

À propos de Skills Network Cloud IDE

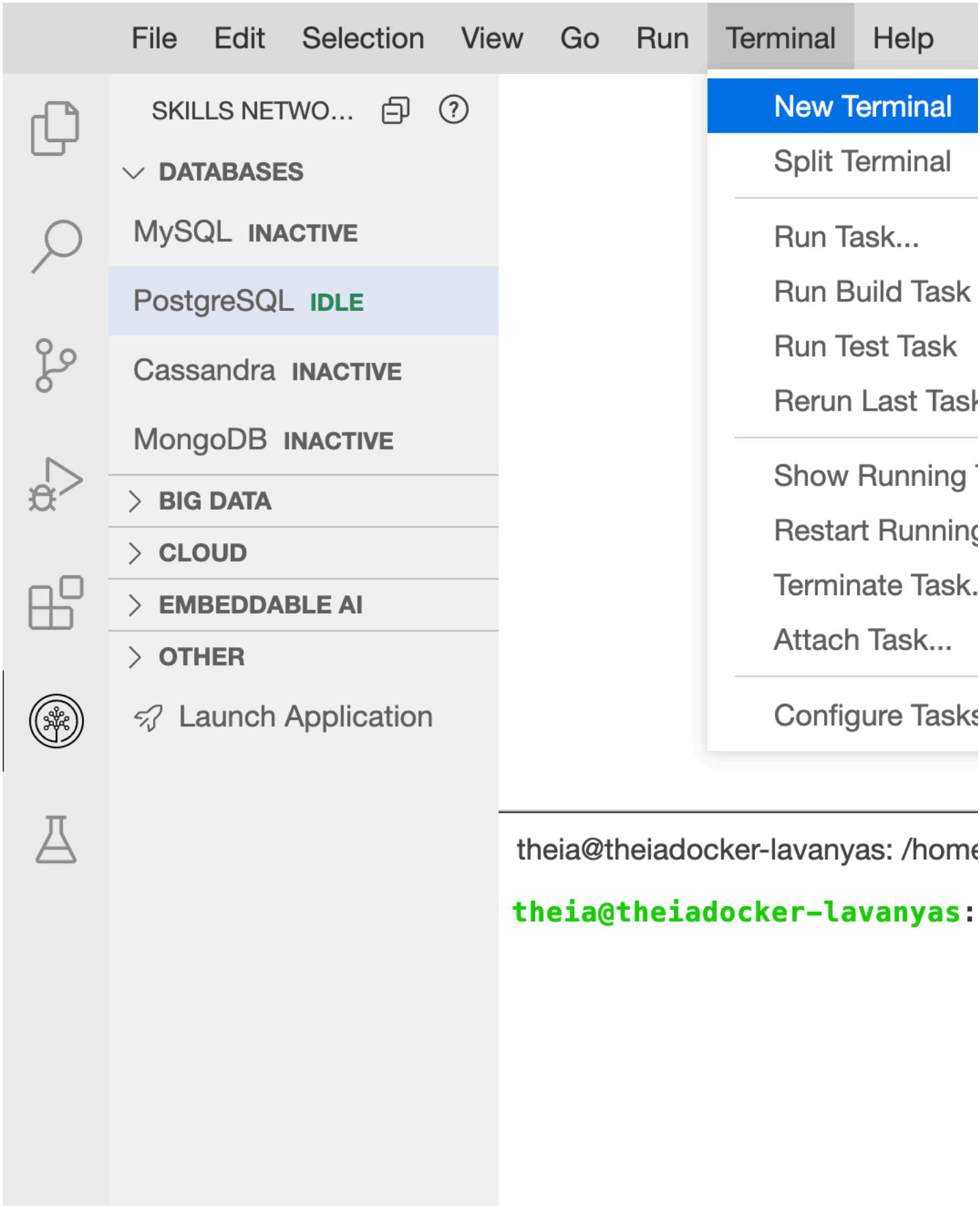
L'IDE Cloud de Skills Network (basé sur Theia et Docker) fournit un environnement pour les travaux pratiques liés aux cours et aux projets. Theia est un IDE open source (environnement de développement intégré), qui peut être exécuté sur un ordinateur de bureau ou sur le cloud. Pour réaliser ce laboratoire, nous utiliserons l'IDE Cloud basé sur Theia et Postgres exécuté dans un conteneur Docker.

Avis important concernant cet environnement de laboratoire

Veuillez noter que les sessions de cet environnement de laboratoire ne sont pas conservées. Chaque fois que vous vous connectez à ce laboratoire, un nouvel environnement est créé pour vous. Toutes les données que vous avez pu enregistrer lors de la session précédente seront perdues. Prévoyez de terminer ces laboratoires en une seule session, pour éviter de perdre vos données.

Préparer l'environnement

Ouvrez un nouveau terminal en cliquant sur la barre de menu et en sélectionnant **Terminal** -> **Nouveau terminal** , comme dans l'image ci-dessous. Cela ouvrira un nouveau terminal en bas de l'écran.



Exécutez toutes les commandes sur le terminal nouvellement ouvert. (Vous pouvez copier le code en cliquant sur le petit bouton de copie en bas à droite du bloc de code ci-dessous, puis le coller où vous le souhaitez.)

Exercice 1 – Extraction de données à l'aide de la commande « cut »

La commande de filtre `cut`nous aide à extraire les caractères ou les champs sélectionnés d'une ligne de texte.

1. Extraction de caractères.

La commande ci-dessous montre comment extraire les quatre premiers caractères.

```
1. 1
```

```
1. echo "database" | cut -c1-4
```

Copié! Exécuté!

Vous devriez obtenir la chaîne « data » en sortie.

La commande ci-dessous montre comment extraire les 5e à 8e caractères.

```
1. 1
```

```
1. echo "database" | cut -c5-8
```

Copié! Exécuté!

Vous devriez obtenir la chaîne « base » en sortie.

Les caractères non contigus peuvent être extraits à l'aide de la virgule.

La commande ci-dessous montre comment extraire les 1er et 5ème caractères.

```
1. 1
```

```
1. echo "database" | cut -c1,5
```

Copié! Exécuté!

Vous obtenez la sortie : 'db'

2. Extraction de champs/colonnes

Nous pouvons extraire une colonne/un champ spécifique d'un fichier texte délimité, en mentionnant

- le délimiteur utilisant l' `-d`option, ou
- le numéro de champ utilisant l' `-f`option.

Le /etc/passwd est un fichier délimité par « : ».

La commande ci-dessous extrait les noms d'utilisateur (le premier champ) de /etc/passwd.

```
1. 1
```

```
1. cut -d":" -f1 /etc/passwd
```

Copié! Exécuté!

La commande ci-dessous extrait plusieurs champs 1er, 3e et 6e (nom d'utilisateur, ID utilisateur et répertoire personnel) de /etc/passwd.

```
1. 1
```

```
1. cut -d":" -f1,3,6 /etc/passwd
```

Copié! Exécuté!

La commande ci-dessous extrait une plage de champs 3 à 6 (identifiant utilisateur, identifiant de groupe, description de l'utilisateur et répertoire personnel) de /etc/passwd.

```
1. 1
```

```
1. cut -d":" -f3-6 /etc/passwd
```

Copié! Exécuté!

Exercice 2 – Transformer des données à l'aide de « tr »

`tr`est une commande de filtre utilisée pour traduire, compresser et/ou supprimer des caractères.

1. Traduire d'un jeu de caractères à un autre

La commande ci-dessous traduit tous les alphabets minuscules en majuscules.

```
1. 1
```

```
1. echo "Shell Scripting" | tr "[a-z]" "[A-Z]"
```

Copié! Exécuté!

Vous pouvez également utiliser les jeux de caractères prédéfinis à cette fin :

```
1. 1
```

```
1. echo "Shell Scripting" | tr "[:lower:]" "[:upper:]"
```

Copié! Exécuté!

La commande ci-dessous traduit tous les alphabets majuscules en minuscules.

```
1. 1
```

```
1. echo "Shell Scripting" | tr  "[A-Z]" "[a-z]"
```

Copié! Exécuté!

2. Presser les occurrences répétées de caractères

L' -soption remplace une séquence de caractères répétés par une seule occurrence de ce caractère.

La commande ci-dessous remplace les occurrences répétées de « espace » dans la sortie de `ps`la commande par un « espace ».

```
1. 1
```

```
1. ps | tr -s " "
```

Copié! Exécuté!

Dans l'exemple ci-dessus, le caractère espace entre guillemets peut être remplacé par ce qui suit : `"[\:space\:]"`.

3. Supprimer des caractères

Nous pouvons supprimer des caractères spécifiés à l'aide de l' `-d` option.

La commande ci-dessous supprime tous les chiffres.

```
1. 1
```

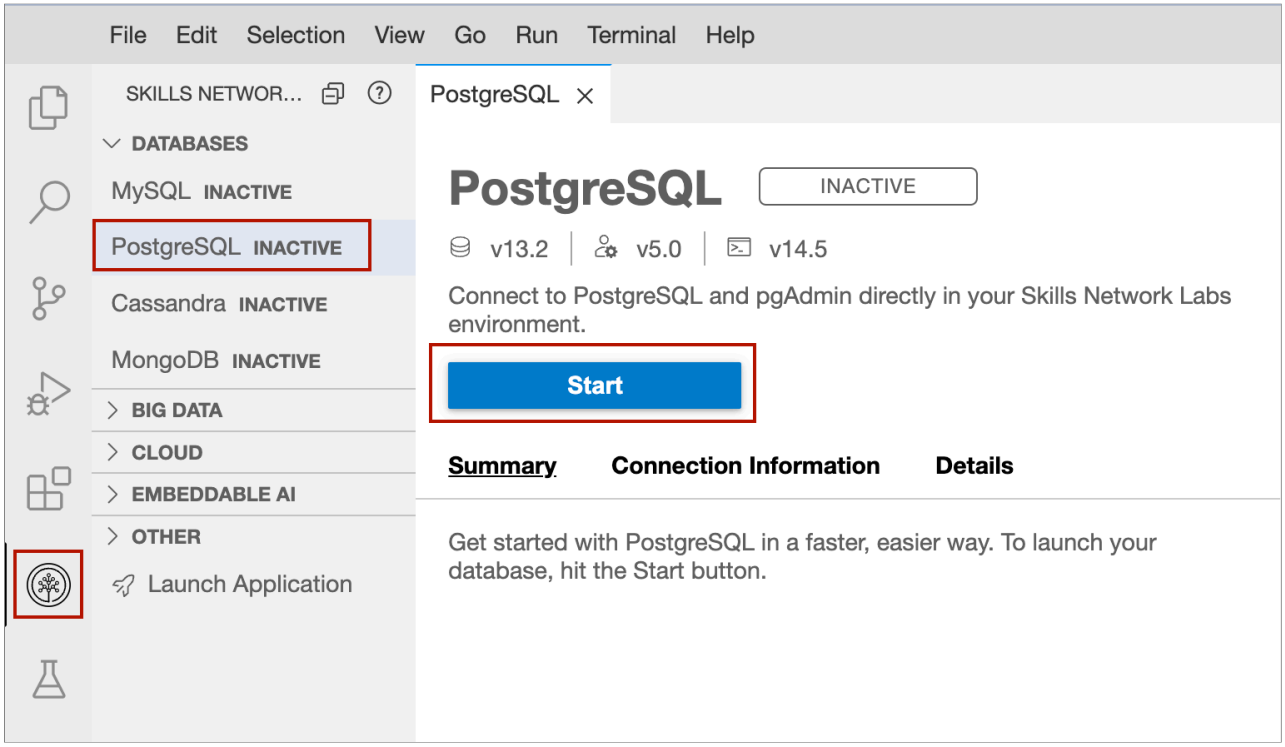
```
1. echo "My login pin is 5634" | tr -d "[:digit:]"
```

Copié! Exécuté!

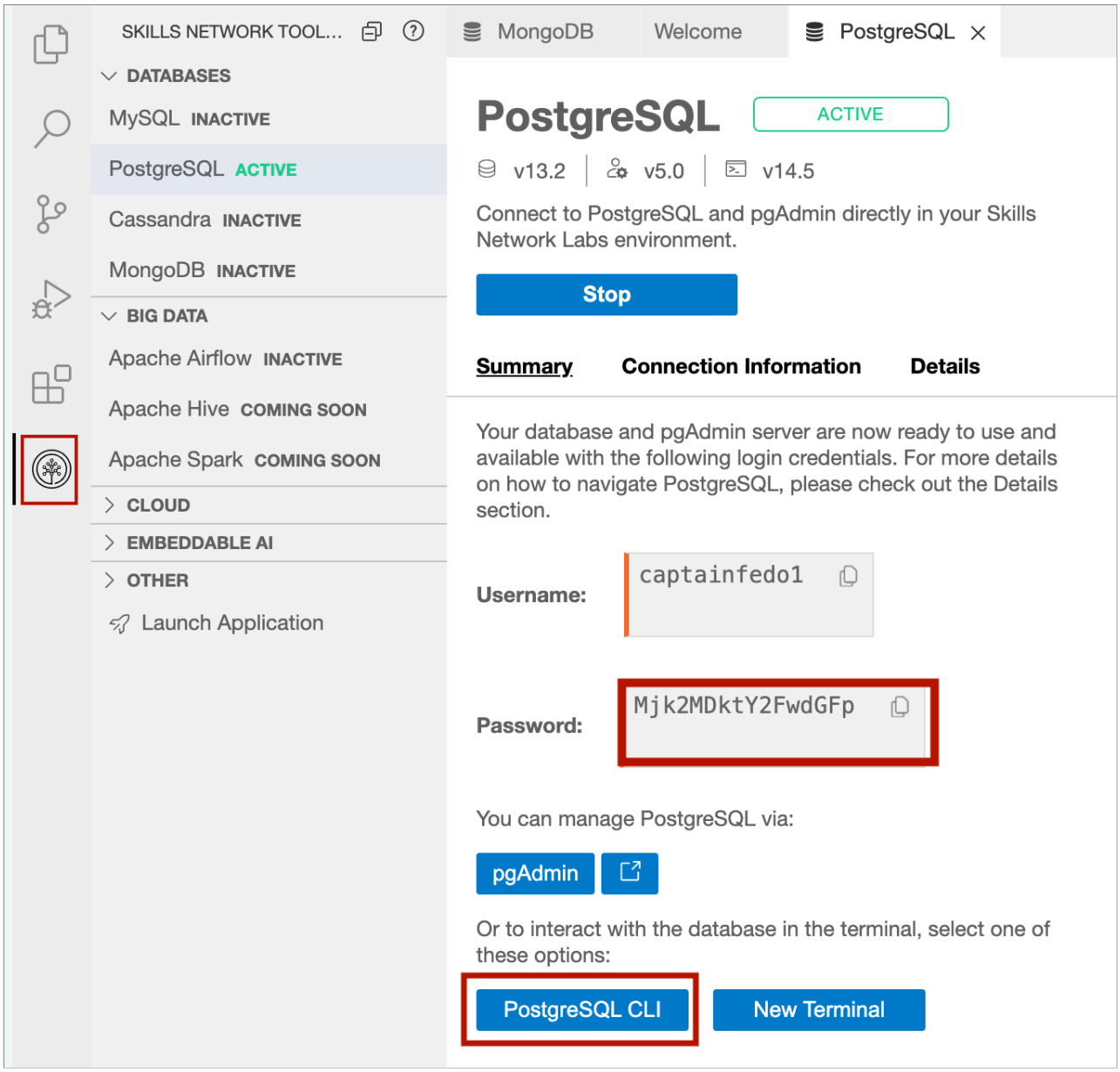
Le résultat sera : « Mon code PIN de connexion est »

Exercice 3 – Démarrer la base de données PostgreSQL.

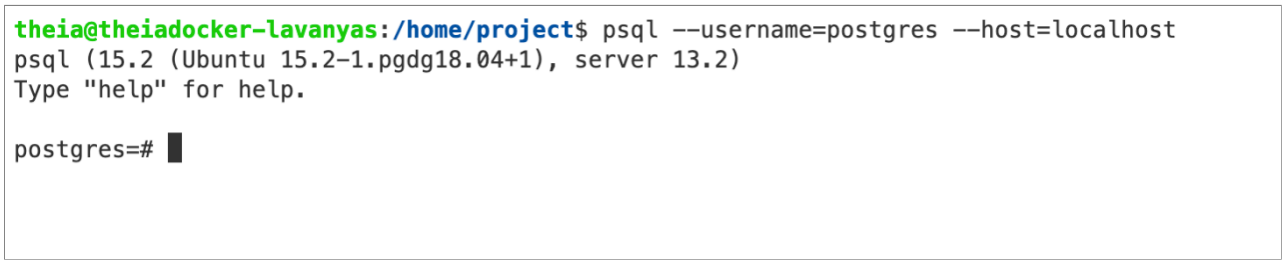
1. Dans les outils SkillsNetwork, sous Bases de données, choisissez `PostgresQL`Serveur de base de données et cliquez `Start`pour démarrer le serveur. Cela prendra quelques minutes.



2. Cliquez `PostgresSQL CLI`sur l'écran pour commencer à interagir avec le serveur PostgresSQL.



Cela démarrera le `psql`client interactif qui se connecte au serveur PostgreSQL avec `postgres=#`l'invite comme indiqué ci-dessous.



Exercice 4 – Créer un tableau

Dans cet exercice, nous allons créer une table appelée `users`dans la base de données PostgreSQL à l'aide de la CLI PostgresSQL. Cette table contiendra les informations du compte utilisateur.

Le tableau `users`comportera les colonnes suivantes :

1. nominatif
2. identifiant utilisateur
3. maison
4. Vous vous connecterez à `template1`une base de données qui est déjà disponible par défaut. Pour vous connecter à cette base de données, exécutez la commande suivante à l'invite « postgres=# ».

1. 1

```
1. \c template1
```

Copié!

Vous recevrez le message suivant.

```
You are now connected to database "template1" as user "postgres".
```

De plus, votre invite changera en « template1=# ».

2. Exécutez l'instruction suivante à l'invite « template1=# » pour créer la table.

```
1. 1
```

```
1. create table users(username varchar(50),userid int,homedirectory varchar(100));
```

Copié!

Si le tableau est créé avec succès, vous recevrez le message ci-dessous.

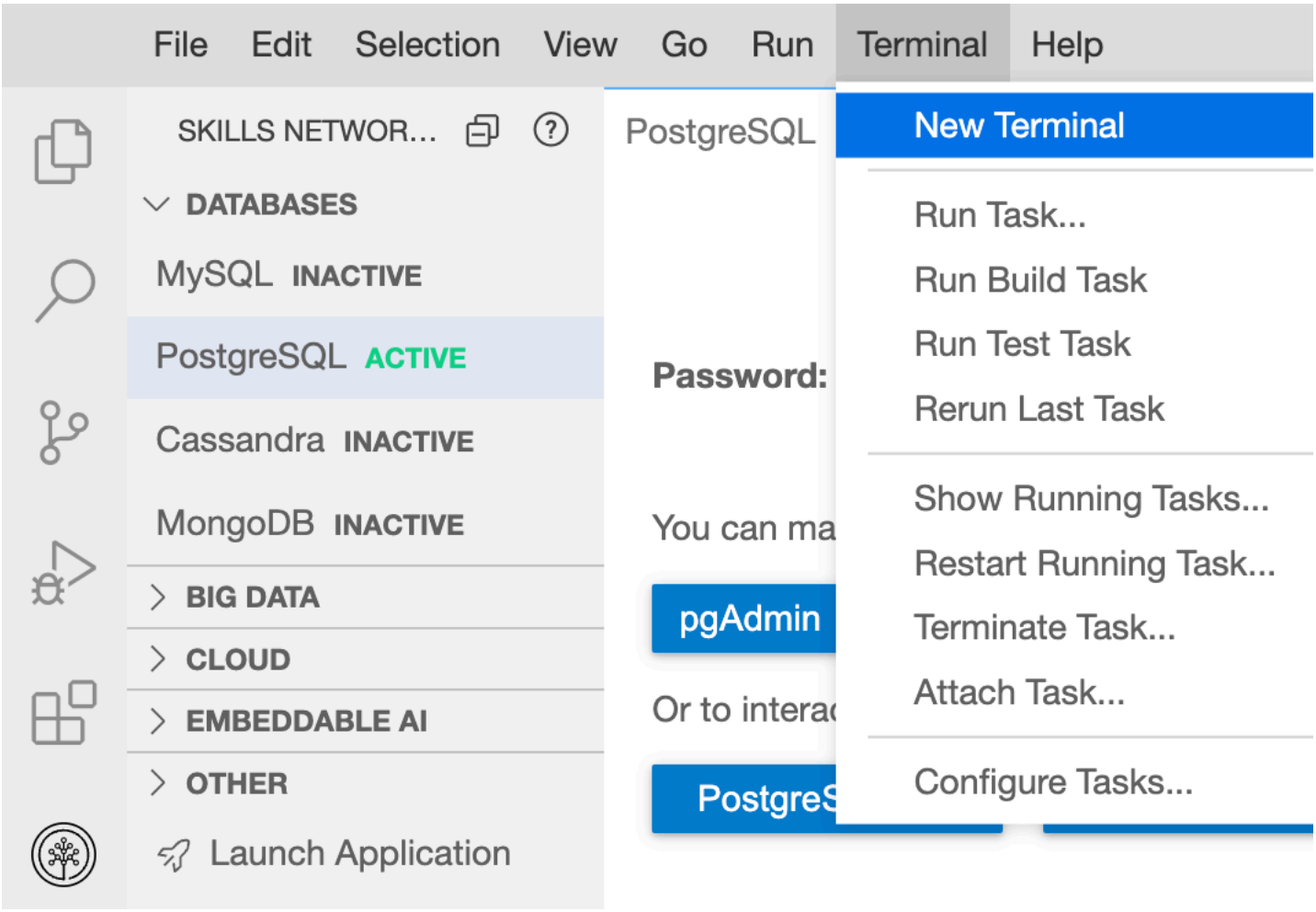
```
CREATE TABLE
```

Exercice 5 – Chargement de données dans une table PostgreSQL.

Dans cet exercice, vous allez créer un script shell qui effectue les opérations suivantes.

- Extrayez le nom d'utilisateur, l'ID utilisateur et le chemin du répertoire personnel de chaque compte utilisateur défini dans le fichier /etc/passwd.
- Enregistrez les données dans un format séparé par des virgules (CSV).
- Chargez les données du fichier csv dans une table de la base de données PostgreSQL.

1. Ouvrez un nouveau terminal.



2. Dans le terminal, exécutez la commande suivante pour créer un nouveau script shell nommé `csv2db.sh`.

```
1. 1
```

```
1. touch csv2db.sh
```

Copié! Exécuté!

3. Ouvrez le fichier dans l'éditeur. Copiez et collez les lignes suivantes dans le fichier nouvellement créé.

Ouvrir **csv2db.sh** dans l'IDE

```
1. 1
2. 2
3. 3
4. 4
```

```
5. 5
6. 6
7. 7
8. 8
```

```
1. # This script
2. # Extracts data from /etc/passwd file into a CSV file.
3.
4. # The csv data file contains the user name, user id and
5. # home directory of each user account defined in /etc/passwd
6.
7. # Transforms the text delimiter from ":" to ",".
8. # Loads the data from the CSV file into a table in PostgreSQL database.
```

Copié!

4. Enregistrez le fichier en appuyant sur `Ctrl+s`ou en utilisant l' option de menu **Fichier->Enregistrer** .

5. Vous devez ajouter des lignes de code au script qui extrairont le nom d'utilisateur (champ 1), l'ID utilisateur (champ 3) et le chemin du répertoire personnel (champ 6) du fichier /etc/passwd à l'aide de la `cut`commande.

Copiez les lignes suivantes et collez-les à la fin du script et enregistrez le fichier.

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
```

```
1. # Extract phase
2.
3. echo "Extracting data"
4.
5. # Extract the columns 1 (user name), 2 (user id) and
6. # 6 (home directory path) from /etc/passwd
7.
8. cut -d":" -f1,3,6 /etc/passwd
```

Copié!

6. Exécutez le script.

```
1. 1
```

```
1. bash csv2db.sh
```

Copié! Exécuté!

7. Vérifiez que la sortie contient les trois champs que vous avez extraits.

8. Modifiez le script pour rediriger les données extraites dans un fichier nommé`extracted-data.txt`

Remplacez la commande cut à la fin du script par la commande suivante.

```
1. 1
```

```
1. cut -d":" -f1,3,6 /etc/passwd > extracted-data.txt
```

Copié!

9. Exécutez le script.

```
1. 1
```

```
1. bash csv2db.sh
```

Copié! Exécuté!

10. Exécutez la commande ci-dessous pour vérifier que le fichier `extracted-data.txt`test créé et possède le contenu.

```
1. 1
```

```
1. cat extracted-data.txt
```

Copié! Exécuté!

11. Les colonnes extraites sont séparées par le délimiteur « : » d'origine. Vous devez convertir cela en un fichier délimité par « , ». Ajoutez les lignes ci-dessous à la fin du script et enregistrez le fichier.

```
1. 1
2. 2
3. 3
4. 4
5. 5
```

```
1. # Transform phase
2. echo "Transforming data"
3. # read the extracted data and replace the colons with commas.
4.
5. tr ":" "," < extracted-data.txt  > transformed-data.csv
```

Copié!

12. Exécutez le script.

```
1. 1
```

```
1. bash csv2db.sh
```

Copié! Exécuté!

13. Exécutez la commande ci-dessous pour vérifier que le fichier `transformed-data.csv`est créé et possède le contenu.

```
1. 1
```

```
1. cat transformed-data.csv
```

Copié!

14. Pour charger des données à partir d'un script shell, vous utiliserez l' `psql`utilitaire client de manière non interactive. Cela se fait en envoyant les commandes de la base de données via un pipeline de commandes à `psql`l'aide de `echo`la commande.

La commande PostgreSQL pour copier des données d'un fichier CSV vers une table est `COPY`.

La structure de base de la commande que nous utiliserons dans notre script est la suivante :

```
COPY table_name FROM 'filename' DELIMITERS 'delimiter_character' FORMAT;
```

Maintenant, ajoutez les lignes ci-dessous à la fin du script « csv2db.sh » et enregistrez le fichier.

```
1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
```

```
1. # Load phase
2. echo "Loading data"
3. # Set the PostgreSQL password environment variable.
4. # Replace <yourpassword> with your actual PostgreSQL password.
5. export PGPASSWORD=<yourpassword>;
6. # Send the instructions to connect to 'template1' and
7. # copy the file to the table 'users' through command pipeline.
8. echo "\c template1;\COPY users  FROM '/home/project/transformed-data.csv' DELIMITERS ',' CSV;" | psql --username=postgres --host=pos
```

Copié!

Exercice 6 – Exécuter le script final

1. Exécutez le script.

```
1. 1
```

```
1. bash csv2db.sh
```

Copié!

2. Maintenant, ajoutez la ligne ci-dessous à la fin du script « csv2db.sh » et enregistrez le fichier.

```
1. 1
```

```
1. echo "SELECT * FROM users;" | psql --username=postgres --host=postgres template1
```

Copié!

3. Exécutez le script pour vérifier que la table users est renseignée avec les données.

```
1. 1
```

```
1. bash csv2db.sh
```

Copié!

Félicitations ! Vous avez créé un script ETL à l'aide de scripts shell.

Exercices pratiques

1. Copiez les données du fichier 'web-server-access-log.txt.gz' dans la table 'access_log' de la base de données PostgreSQL 'template1'.

   Le fichier est disponible à l'emplacement : https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Bash%20Scripting/ETL%20using%20shell%20scripting/web-server-access-log.txt.gz

   Voici les colonnes et leurs types de données dans le fichier :

   - a. horodatage - TIMESTAMP
   - b. latitude - flottant
   - c. longitude - flottant
   - d. visitorid - char(37)
   - e. accessed_from_mobile - booléen
   - f. code_du_navigateur - int

   Les colonnes que nous devons copier dans le tableau sont les quatre premières colonnes : horodatage, latitude, longitude et identifiant du visiteur.

   REMARQUE : le fichier est fourni avec un en-tête. Utilisez donc l'option « HEADER » dans la commande « COPY ».

Le problème peut être résolu en effectuant les tâches suivantes :

1. Accédez au menu Outils SkillsNetwork et démarrez le serveur Postgres SQL s'il n'est pas déjà en cours d'exécution.

2. Créez une table nommée `access_log`pour stocker l'horodatage, la latitude, la longitude et l'identifiant du visiteur.

▶ Cliquez ici pour un indice
▶ Cliquez ici pour la solution

Tâche 3. Créez un script shell nommé `cp-access-log.sh`et ajoutez des commandes pour terminer les tâches restantes pour extraire et copier les données dans la base de données.

Créez un script shell pour ajouter des commandes afin de terminer le reste des tâches.

▶ Cliquez ici pour un indice

Tâche 4. Téléchargez le fichier journal d'accès.

Ajoutez la `wget`commande au script pour télécharger le fichier.

```
1. 1
```

```
1. wget "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-DB0250EN-SkillsNetwork/labs/Bash%20Scripting/ETL%20usir
```

Copié!

Tâche 5. Décompressez le fichier gzip.

Ajoutez le code pour exécuter la commande gunzip pour décompresser le `.gz`fichier et extraire le `.txt`fichier, au script.

```
1. 1
2. 2
```

```
1. # Unzip the file to extract the .txt file.
2. gunzip -f web-server-access-log.txt.gz
```

Copié!

L' option **-f** de gunzip permet d'écraser le fichier s'il existe déjà.

Tâche 6. Extraire les champs obligatoires du fichier.

Extrayez l'horodatage, la latitude, la longitude et l'identifiant du visiteur qui sont les quatre premiers champs du fichier à l'aide de la `cut`commande.

Les colonnes du fichier web-server-access-log.txt sont délimitées par « # ».

▶ Cliquez ici pour un indice
▶ Cliquez ici pour la solution

Tâche 7. Rediriger la sortie extraite vers un fichier.

Rediriger les données extraites dans un fichier nommé`extracted-data.txt`

▶ Cliquez ici pour un indice
▶ Cliquez ici pour la solution

Tâche 8. Transformer les données au format CSV.

Les colonnes extraites sont séparées par le délimiteur « # » d'origine.

Nous devons convertir cela en un fichier délimité par des « , ».

▶ Cliquez ici pour un indice
▶ Cliquez ici pour la solution

Tâche 9. Charger les données dans la table `access_log`dans PostgreSQL

La commande PostgreSQL pour copier des données d'un fichier CSV vers une table est `COPY`.

La structure de base de la commande est la suivante :

```
1. 1
```

```
1. COPY table_name FROM 'filename' DELIMITERS 'delimiter_character' FORMAT;
```

Copié!

Le fichier est fourni avec un en-tête. Utilisez donc l'option « HEADER » dans la commande « COPY ».

Appelez cette commande à partir du shellscript, en l'envoyant comme entrée à la commande de filtre « psql ».

▶ Cliquez ici pour un indice
▶ Cliquez ici pour la solution

Tâche 10. Exécutez le script final.

Exécutez le script final.

▶ Cliquez ici pour la solution

Tâche 11. Vérifiez en interrogeant la base de données.

▶ Cliquez ici pour un indice
▶ Cliquez ici pour la solution

Auteurs

Ramesh Sannareddy

Autres contributeurs

Rav Ahuja