

M164 – Importer des données

F. Mauron

EPAI

12 décembre 2023

- 1 Introduction
- 2 Processus d'importation

- 1 Introduction
- 2 Processus d'importation

L'importation en masse de données, également appelée **Bulk Load**, est une opération courante dans les bases de données SQL, y compris MariaDB.

Cela permet d'ajouter rapidement de grandes quantités de données à partir de sources externes telles que des fichiers CSV ou d'autres bases de données.

- 1 Introduction
- 2 Processus d'importation

Étapes de l'importation

- 1 Préparation des données sources
- 2 Choix de l'outil d'import
- 3 Exécution de l'import
- 4 Vérification des données
- 5 Gestion des erreurs
- 6 Index et contraintes

Avant d'importer des données en masse, vous devez vous assurer que les données sources soient prêtes.

C'est-à-dire, qu'elles correspondent à la structure de la table de la base de données cible.

Selon le SGBD utilisé, il y a plusieurs méthodes pour effectuer des imports en masse. Ces méthodes diffèrent d'un SGBD à une autre.

MariaDB en propose plusieurs afin d'effectuer des imports en masse.

Selon l'outil que vous choisissez, l'exécution de l'import va varier.

Astuce : Il est toujours préférable de préparer les importations dans un ou plusieurs scripts !

Une fois l'import effectué, vous devez vérifier les données afin de vous assurer qu'elles ont été correctement importées et qu'elles correspondent à vos attentes.

Astuces : Il est préférable de réaliser l'import dans une table temporaire afin de faire la vérification et seulement après de copier le résultat dans la table cible.

Il est important de gérer les erreurs d'importation.

Les fichiers sources peuvent contenir des données incorrectes ou mal formatées. Assurez-vous de mettre en place des mécanismes de gestion des erreurs appropriés pour identifier et corriger les problèmes potentiels.

Lorsque vous importez de grandes quantités de données, il peut être utile de désactiver les index et les contraintes, puis de les réactiver après l'importation pour améliorer les performances.

Soit le fichier csv suivant :

```
id,nom,prenom,age,poste
1,Smith,John,30,Manager
2,Johnson,Emily,28,Analyste
3,Davis,David,35,Ingénieur
4,Anderson,Susan,27,Assistant
5,Wilson,Michael,32,Développeur
```

Pour le télécharger : [employees.csv](#)

- Hypothèse : la table **Employes** existe.
- Création de la table temporaire **EmployesTemp**

```
CREATE TABLE IF NOT EXISTS EmployesTemp (  
    id INT,  
    nom VARCHAR(255),  
    prenom VARCHAR(255),  
    age INT,  
    poste VARCHAR(255)  
);
```

```
LOAD DATA LOCAL INFILE '<path>/employes.csv'  
INTO TABLE EmployesTemp  
FIELDS TERMINATED BY ','  
ENCLOSED BY '"'  
LINES TERMINATED BY '\n'  
IGNORE 1 ROWS;
```

Faire la vérification des données, corriger les erreurs et copier les données dans la table cible

```
INSERT INTO Employes (id, nom, prenom, age, poste)
SELECT id, nom, prenom, age, poste
FROM EmployesTemp;
```

Il reste à supprimer la table temporaire

```
DROP TABLE EmployesTemp;
```