SpaceFox



Zeste de savoir

# Charger des données depuis un CSV dans Oracle avec Liquibase

11 mai 2021

# Table des matières

Aujourd'hui, on résous un problème très spécifique et très casse-pieds quand on se retrouve à devoir le gérer.

Liquibase est un outil très pratique qui permet de gérer les migrations de schémas de données, il est surtout utilisé avec des langaes à JVM (Java, Kotlin, etc). Il permet de charger des données directement depuis des fichiers CSV, ce qui est très pratique pour livrer des données «standard» avec l'application (configurations, référentiels, etc). Ceci se fait avec les commandes `loadData` ou `loadUpdateData` .

L'avantage, c'est que c'est très simple, ici avec un fichier de configuration XML mais la logique est la même quel que soit le format utilisé:

```
<loadData file="path/to/file.csv" tableName="table_name"
    relativeToChangelogFile="true" />
```

Et le système va lire simplement les en-têtes de colonnes dans le CSV, trouver les colonnes correspondantes dans la table et faire les insertions qui vont bien.

Sauf que…

> Cette commande est extrêmement lente avec Oracle!

Je parle d'une commande qui peut mettre plusieurs secondes pour insérer une seule ligne d'une table avec une dizaine de colonnes, sur une base de données accessible sur le même réseau local. Et je ne parle même pas des temps de réponse qui explosent dès que vous n'êtes plus dans ce cas (au hasard, un test sur une base distante pendant que vous êtes en télétravail au travers d'un VPN sur une ADSL, et voilà plusieurs minutes passées à simplement insérer une ligne dans une table).

La solution bourrine et illisible, c'est de remplacer les imports de CSV par des batteries de `INSERT`, mais on peut faire mieux.

En effet, **ce qui prends aussi longtemps, c'est la récupération des métadonnées sur les colonnes à insérer** (quelles colonnes existent réellement dans la table et quel est leurs types, pour que Liquibase puisse convertir correctement les données depuis le CSV vers la BDD). Et c'est cette procédure qui est particulièrement inefficace sur Oracle[1], elle nécessite énormément

de requêtes avant de pouvoir faire la moindre insertion, et donc est très sensible au *ping* de la connexion utilisée.

L'astuce, c'est qu'ont peut **définir les noms et types de colonnes à la main**, ce qui évite à Liquibase de se lancer dans une introspection de la BDD pour avoir l'information. La définition de l'import devient quelque chose comme ceci – en supposant un CSV avec des headers, chaque colonne du CSV a le même nom que la colonne de la BDD:

```
<loadData file="path/to/file.csv" tableName="table_name"
    relativeToChangelogFile="true">
  <column header="ID" name="ID" type="NUMERIC"/>
  <column header="LABEL" name="LABEL" type="STRING"/>
  <column header="DATE_MAJ" name="DATE_MAJ" type="DATE"/>
  <!-- … etc. -->
</loadData>
```

Et le plus beau, c'est qu'on peut se contenter de **faire générer toutes les balises `<column>` à Oracle!**

```
select '<column header="' || col.column_name || '" name="' ||
    col.column_name || '" type="' ||
       decode(col.DATA_TYPE,
              'VARCHAR2', 'STRING',
              'NUMBER', 'NUMERIC',
              'DATE', 'DATE',
              'BLOB', 'BLOB',
              'CLOB', 'CLOB',
              'NCLOB', 'CLOB')
              -- Il manque sans doute des conversions ici.
              -- en cas de type="" dans le résultat, ajouter la
                  conversion à ce decode() à l'aide de
              --
                  https://www.liquibase.org/javadoc/liquibase/change/core/LoadD
           || '" />'
from sys.all_tab_columns col
where col.owner = 'USER_NAME'
  and col.table_name = 'TABLE_NAME'
order by col.column_id;
```

Et voilà, votre import de donnée est un peu moins lisible mais *beaucoup* plus rapide. Ça peut dépasser un facteur cent (moins de 20 ms contre plus de 2 secondes auparavant).

---



1. En réalité le problème existe peut-être avec d'autre SGBD. Mais je ne me rappelle pas avoir eu le cas avec PostgreSQL, et j'ai la flemme de tester. Si vous avez des informations à ce sujet, n'hésitez pas à les partager en commentaire.