Sqlplus Commando

Michel Casabianca
casa@sweetohm.net

Sur Github: https://github.com/c4s4/sqlplus_command

L'installation d'un pilote Oracle sur une machine est parfois pénible, voire impossible. En outre, vous pouvez souhaiter distribuer des scripts autonomes qui accèdent à Oracle sans avoir à installer de logiciel supplémentaire. Enfin, vous pouvez vouloir automatiser les scripts qui doivent être lancés avec SQL*Plus.

sqlplus_commando est un pur pilote Oracle Python qui appelle *sqlplus* sur la ligne de commande. Il a été conçu pour que vous puissiez l'utiliser en copiant son source dans votre répertoire des sources ou même en recopiant ses classes dans votre propre code source.

Installation

Pour installer *sqlplus_commando*, vous pouvez utiliser l'une des méthodes suivantes:

- Extrayez les classes `SqlplusCommando`, `OracleResultParser` et `OracleErrorParser` de la tarball (dans le fichier *sqlplus_commando/sqlplus_commando.py*) et placez-les dans votre propre code source.
- Déposez son module (fichier *sqlplus_commando/sqlplus_commando.py* dans l'archive) dans votre répertoire source.
- Installez-le en utilisant PIP, avec la commande `pip install sqlplus_commando`.
- Installation à partir de tarball en tapant `python setup.py install`.

La licence Apache vous accorde le droit d'utiliser ce pilote dans n'importe quel projet (même commercial) à condition que vous indiquiez que vous utilisez *sqlplus_commando* dans votre avis de copyright.

Usage

Vous pouvez utiliser ce pilote dans votre code comme ceci:

```
from sqlplus_commando import SqlplusCommando

sqlplus = SqlplusCommando(hostname='localhost', database='test',
                          username='test', password='test')
result = sqlplus.run_query("SELECT 42 AS response, 'This is a test' AS question FRO
print result
```

Quand la requête ne renvoie rien (après un `INSERT` par exemple), la méthode `run_query()` retournera un tuple vide `()`. Si la requête renvoie un ensemble de résultats, il s'agira d'un tuple de

dictionnaires. Par exemple, l'exemple de code précédent pourrait afficher:

```
({'RESPONSE': 42, 'QUESTION': 'This is a test'},)
```

Au lieu d'exécuter une requête, vous pouvez exécuter un script comme suit :

```
result = sqlplus.run_script('my_script.sql')
```

Paramètres

Vous pouvez avoir des valeurs telles que `%(foo)s`; elles seront remplacées dans votre requête par les valeurs correspondantes du dictionnaire de paramètres. Par exemple :

```
from sqlplus_commando import SqlplusCommando

sqlplus = SqlplusCommando(hostname='localhost', database='test',
                          username='test', password='test')
parameters = {'name': 'Reglisse'}
result = sqlplus.run_query(query="SELECT * FROM animals WHERE name=%(name)s",
                           parameters=parameters)
print result
```

Vous ne pouvez pas fournir de paramètres en exécutant un script. Pour ce faire, appelez `run_query()` avec les paramètres en passant la requête `open('my_script.sql').read()`.

Types de result sets

sqlplus_commando effectue un cast automatique avant de retourner les ensembles de résultats. Comme il appelle *sqlplus* sur la ligne de commande, chaque valeur du jeu de résultats est une chaîne. Par commodité, il convertit les entiers, les flottants, les dates et les valeurs NULL en types Python natifs correspondants.

Il y a des situations où cela pourrait ne pas être exact. Par exemple, si une colonne est de type SQL `VARCHAR(10)` et contient des numéros de téléphone, toutes ses valeurs seront castées en entiers Python. C'est incorrect parce que les numéros de téléphone peuvent commencer avec un *0* qui serait perdu lors de la conversion en nombre entier.

Pour éviter cela, vous pouvez passer `cast=False` quand vous appelez `run_query()` ou `run_script()`, comme ceci :

```
from sqlplus_commando import SqlplusCommando

sqlplus = SqlplusCommando(hostname='localhost', database='test',
                          username='test', password='test')
result = sqlplus.run_query("SELECT phone FROM users WHERE name='bob')", cast=False)
```

```
print result
```

Vous pouvez également désactiver la conversion lors de l'instanciation du pilote, en passant `cast=False` au constructeur. Cette désactivation de la conversion s'appliquera à tous les appels à `run_query()` ou `run_script()` sauf si vous transmettez une valeur différente en appelant ces méthodes.

Gestion des erreurs

Lors de l'exécution d'une requête ou d'un script avec *sqlplus*, vous devez ajouter les commandes SQL suivantes afin que la valeur renvoyée soit différente de *0* si une erreur se produit:

```
WHENEVER SQLERROR EXIT SQL.SQLCODE;
WHENEVER OSERROR EXIT 9;
```

Ces lignes sont ajoutées avant les requêtes ou les scripts à exécuter pour éviter d'avoir à analyser le résultat des messages d'erreur. Néanmoins, il y a des cas où ces lignes ne vont pas aider à la détection d'erreur. Par exemple, la requête suivante :

```
BAD SQL QUERY;
```

Cela n'entraînera pas une erreur dans *sqlplus* et nous devons analyser le résultat pour rechercher la chaîne d'erreur `SP2-0734: unknown command`. Ceci est fait par défaut, mais vous pouvez l'éviter en passant le paramètre `check_unknown_command=False` lors de l'appel des fonctions `run_query` ou `run_script`.

De plus, une erreur de compilation entraînera un avertissement, il est donc souvent nécessaire de vérifier les avertissements dans la sortie de *sqlplus*. Ceci est fait par défaut et entraînera une exception, sauf si vous passez le paramètre `check_warning=False` en appelant les fonctions `run_query` ou `run_script`.

Note

Ce module n'est pas destiné à remplacer un pilote Oracle authentique que vous **DEVRIEZ** utiliser si vous pouvez l'installer sur la machine cible.

Enjoy!