

GraceTHD-Check - Manuel administrateur Postgis

Révision : 16/05/2017 - sby

Applicable à partir de GraceTHD-Check v0.01-alpha6

- GraceTHD-Check est un projet GraceTHD-Community.
<https://github.com/GraceTHD-community>
- Rédacteurs : Stephane Byache (Aleno).
- Licence de cette documentation : CC-by-sa v2.0
- GraceTHD-Check est sous licence libre : GPLv3

Préambule

Cette documentation est rédigée sur le wiki du Groupe Experts Contrôle sur le Redmine GraceTHD.

<https://redmine.gracethd.org/redmine/projects/gracethd-check/wiki>

Elle fait partie d'un ensemble de documentations.

- GraceTHD-Check - Présentation
- GraceTHD-Check - Manuel utilisateur Postgis
- GraceTHD-Check - Manuel utilisateur avancé Postgis
- GraceTHD-Check - Manuel administrateur Postgis
- GraceTHD-Check - Manuel développeur
- GraceTHD-Check - Index des scripts
- GraceTHD-Check - Résumé du catalogue des points de contrôle

Notes :

- Préalablement à l'utilisation de GraceTHD-Check, un administrateur GraceTHD-Check doit être désigné et mettre en place GraceTHD-Check en relation avec le DBA PostgreSQL. L'administrateur GraceTHD-Check peut être DBA, mais pas obligatoirement, par contre il doit comprendre GraceTHD-Check et doit pouvoir assurer un support à ses utilisateurs. Un manuel utilisateur avancé permet également à un utilisateur de partager cette responsabilité.

I. Installer GraceTHD-Check

Télécharger :

<https://gracethd-community.github.io/GraceTHD-Check/>

Décompresser :

Le dossier dans un espace accessible en lecture / écriture aux utilisateurs. Attention, certaines opérations lisent et créent de fichiers avec le compte PostgreSQL qui sera déclaré dans les fichiers de configuration. Cet utilisateur aura besoin de droits en lecture et écrite également, pas seulement le compte Windows de l'utilisateur.

PostgreSQL :

GraceTHD-Check est actuellement développé sur PostgreSQL 9.5 / Postgis 2.3. GraceTHD-Check peut tourner soit directement, soit avec de minimales adaptations, sur d'autres versions.

Les scripts sont pour l'instant seulement disponibles en batch (Windows). Si votre serveur est sous un système Unix :

- option 1 : installer un client sur une machine Windows pour exécuter les batchs. Attention dans ce cas l'utilisateur doit pouvoir disposer notamment de shp2pgsql et pgsql2shp qui sont déployés avec Postgis. Une installation complète d'un serveur PostgreSQL/Postgis en local peut être une bonne solution, notamment pour disposer d'un serveur de test.
- option 2 : porter les scripts en bash. Si vous avez besoin de ce portage, si vous souhaitez y contribuer, merci de soumettre une demande :
<https://redmine.gracethd.org/redmine/projects/gracethd-check/issues/new>

QGIS :

Certains scripts utilisent des outils en ligne de commande déployés notamment avec QGIS. Toutefois ceux-ci ne sont pour l'instant pas utiles pour les utilisateurs Postgis, mais plutôt pour les utilisateurs Spatialite.

II. La grille de remplissage

II.1 Grille de remplissage conforme

Normalement, vous avez produit une (ou plusieurs) grille de remplissage conforme à votre projet (cahier des charges). Si ce n'est pas le cas, récupérez le modèle de grille de remplissage (adapté à ce jour à 3 cas) et l'adapter à votre contexte. Normalement ce document devrait être annexé au cahier des charges.

<http://gracethd-community.github.io/GraceTHD-MOD/>

II.2 Mettre à jour les fichiers SQL

Chaque table produit un sql (colonne avec une police de petite taille en bleu clair).

Copier cette colonne pour les tables à fournir dans `gracethdcheck_31_filltab_insert.sql` en remplacement du script par défaut. Bien conserver la ligne

```
SET search_path TO gracethdcheck, public;
```

Copier la colonne pour la grille de remplissage de vos attributs dans `gracethdcheck_32_fillatt_insert.sql` en remplacement du script par défaut. Bien conserver la ligne

```
SET search_path TO gracethdcheck, public;
```

II.3 Mettre à jour la grille de remplissage dans la base de données

Il est bien évidemment possible de remettre à jour à tout moment l'intégration de la grille de remplissage. Exécuter ce script : `gracethdcheck_pg_update_fill.bat`

III. Configuration

"GraceTHD-Check - Présentation" présente les fichiers de configuration. Cette section présente les principales variables.

Configurations génériques - **config.bat**:

- **GLCONF** : Dossier qui accueille les fichiers de configuration. A priori pas besoin de modifier.
- **GLPAUSE** :
 - **GLPAUSE=PAUSE** alors des pauses s'activeront pour visualiser l'affichage lors des opérations.
 - **GLPAUSE=** alors les pauses ne s'activeront pas.

Configuration de l'arborescence - **config_tree.bat**:

Dans la section **GL_CONFIG_GRACETHDCHECK_TREE** :

- **GLCTPGDB** : Le nom de la base de données à contrôler.
- **GLCTPGTEMP** : Dossier temporaire pour lequel il faut s'assurer que l'utilisateur référencé sur la variable **PGUSER** possède les droits de lecture et d'écriture. *Note: Par défaut il hérite de la valeur de **GLTEMP**. Si le dossier est différent de celui de **GLTEMP**, alors il faudra collecter des éléments dans ces deux dossiers.*
- **GLPGCHECKPATH** : Dossier qui va recevoir les rapports d'anomalies exportés en CSV.

Configuration des applications - **config_apps.bat**:

Dans la section **GL_CONFIG_APPS_PATH** :

- **GL_PGBIN** : le chemin vers les binaires (exécutables) correspondant à votre version de PostgreSQL.
 - Normalement **PSQL**, **PG_DUMP**, **SHP2PGSQL**, **PGSQL2SHP** sont dans ce dossier, donc inutile de modifier les valeurs.

*Note: Normalement les chemins concernant **QGIS** ne sont pas utiles pour les utilisateurs PostgreSQL.*

Configuration de Postgis - **config_db_pg.bat**:

Dans la section GL_CONFIG_GRACETHD_POSTGIS_PATH :

- **GLPGPATHSHARE** : dossier pour lequel l'utilisateur référencé dans la variable PGUSER a les droits de lecture et d'écriture.

Note: Normalement les variables suivantes ne devraient pas avoir besoin de configuration. Toutefois s'assurer que les sous-dossiers sont bien existants.

- PGSHPINPATH : Dossier comportant les shp/csv a importer dans Postgis.
- PGSHPOUTPATH : Dossier comportant les shp/csv a importer dans Postgis.
- GLPGCONFPATH : Chemin ou l'utilisateur PGUSER peut lire et ecrire des fichiers de configuration. *Note: ATTENTION l'utilisateur définit pour PGUSER doit avoir les droits d'accès au contenu de ce dossier.*

Dans la section :GL_CONFIG_GRACETHD_POSTGIS_PROD :

Note: concerne la base de production (le référentiel). GraceTHD-Check peut contrôler cette base ou une autre, d'autres paramètres dédiés plus bas.

- **PGPASSWORD** : il est attendu que votre password file soit renseigné (pgpass.conf). Toutefois, il est toujours nécessaire pour certaines opérations (shp2pgsql) de renseigner cette variable avec le mot de passe du compte PostgreSQL défini pour PGUSER. <https://www.postgresql.org/docs/current/static/libpq-pgpass.html>
- **PGUSER** : le compte postgresql à utiliser.
- **PGROLE** : le rôle postgresql correspondant au compte.
- **PGHOSTNAME** : l'hôte postgresql
- **PGPORT** : port de la base postgresql à utiliser.
- **PGDB** : Base de données Postgis de référence.
- PGSCHEMA : Schéma a utiliser dans la base de données (gracethd, ne pas modifier).
- PGSRID : code du système de coordonnées utilise dans cette base de données. Lambert 93 par défaut, modifier pour les DOM/TOM.
- PGCODE : encodage des caractères.
- PGCSVCONF : paramètres pour PSQL pour la structure des csv.

Dans la section :GL_CONFIG_GRACETHDCHECK_POSTGIS :

Note: spécifique à la base où déployer GraceTHD-Check.

- GLCTPGTEMP : Dossier temp utilise par POSTGRESQL. ATTENTION l'utilisateur de PGUSER doit avoir les droits en lecture et écriture.
- GLCTPGSQLPATH : Dossier des .sql Postgis.
- GLPGCHECKPATH : Chemin ou l'utilisateur PGUSER peut lire et écrire les rapports d'anomalies.
- GLCTPGSCHEMA : Nom du schéma qui accueille Gracethd-MCD dans la base à contrôler (gracethd ou PGSCHEMA, ce qui revient au même, ne pas modifier).
- GLCTPGSCHEMACHECK : Nom du schéma qui accueille Gracethd-Check dans la base à contrôler (gracethdcheck, ne pas modifier).
- GLCTPGSCHEMACHECKPUB : Nom du schéma qui accueille les vues publiant les résultats des contrôles. (gracethdcheckpub, ne pas modifier).
- **GLCTPGDB** : Nom de la base de données à contrôler

Note: une fois la configuration faite par l'administrateur, cette dernière variable sera plus ou moins la seule que l'utilisateur modifiera.

IV. Déployer

IV.1 Rôles et privilèges

Les scripts vont créer de nombreuses vues qui doivent être accessibles aux utilisateurs et que les utilisateurs doivent pouvoir recréer. Il est donc hautement conseillé d'appliquer des privilèges aux utilisateurs au niveau du schéma gracethdcheck. Certaines tables comme les listes de valeurs (tables l_*) peuvent être protégées.

Les utilisateurs doivent pouvoir créer des bases de données temporaires, pour contrôler des données avant intégration. Il est

également possible de déployer GraceTHD-Check sur une base GraceTHD-MCD qui est le référentiel (base de production). Dans ce cas il est souhaitable de protéger plus finement cette base. Dans l'absolu le référentiel (base de production) est sur un serveur de production, les bases de contrôle plutôt sur un serveur d'intégration. Il peut être envisageable que les requêtes de contrôle puissent consommer beaucoup de ressources, par conséquent l'utilisation de serveurs distincts peut-être une bonne stratégie.

Il peut également être utile de distinguer un rôle développeur qui autorise des modifications sur `t_c_cat` et les tables `t_ct_code_*`. Mais il est préférable de fournir aux développeurs de points de contrôle des bases de test, des jeux de données test et un partage de test, idéalement sur un serveur de développement. Lors des tests d'optimisation des requêtes il est courant de déclencher des requêtes consommatrices. Un serveur de développement n'a pas les mêmes besoins d'administration.

IV.2 Administration

Les bases de contrôle, qui doivent être créées à chaque nouvelle réception à contrôler, peuvent être conservées, archivées, ou simplement détruites après chaque utilisation. C'est une décision propre à chaque environnement.

Des scripts permettent d'exporter certaines tables en CSV ou en SQL.

- `gracethdcheck_pg_export_tab_csv.bat`
- `gracethdcheck_pg_export_tab_sql.bat`

Ils sont utiles pour les développeurs qui souhaitent exporter les données pour pouvoir transmettre tout ou partie.

Les tables GraceTHD-Check contiennent peut de données mais beaucoup de vues. Les dumps ne sont donc pas lourds.

IV.3 Intégration dans le SI

GraceTHD-Check n'est qu'un moteur livré avec les scripts élémentaires pour le déploiement et l'administration des données. Il est bien évidemment possible d'intégrer cela dans un robot qui réceptionne automatiquement, gère la file d'attente, déclenche les contrôles, envoie ou publie (web ?) les rapports d'anomalies, etc. Il est possible d'interfacer cela avec un BPM ou outil de workflow. Le fonctionnement en base autorise toutes les formes d'urbanisation SI.

Si des développements sont envisagés, il est pertinent de se rapprocher de la communauté GraceTHD pour évoquer le besoin ou les projets. Il est absolument certain qu'un projet open source de type robot serait bénéfique pour le plus grand nombre. Toutefois à ce jour il n'est pas envisagé que la communauté GraceTHD ne développe des outils, auquel cas il serait pertinent d'envisager un autre projet open source externe à la gouvernance GraceTHD.

<https://redmine.gracethd.org>

IV.4 Développement

Pour ceux qui souhaitent développer ou modifier des points de contrôle, voir le manuel dédié aux développeurs.

IV.5 Versions PostgreSQL

- 9.6 : pas de tests pour l'instant.

Versions supportées :

- **9.5 : version utilisée pour le développement.**
- 9.4 : pas de tests pour l'instant, mais ça devrait fonctionner.
- 9.3 : fonctionne. Ca devrait être la version la plus ancienne qui sera officiellement supportée par le projet.

Depuis la version v0.01-alpha6, GraceTHD-Check utilise des vues matérialisées. Celles-ci sont apparues avec PostgreSQL 9.3.

IV.6 Tester

Choisir un nom de base de test comme par exemple :

```
SET GLCTPGDB=%PGDB%ct
```

Cet exemple utilise la valeur de la variable PGDB et ajoute ct, soit un nom de base qui est par exemple `gracethd20ct`.

Le script `gracethdcheck_pg_create_db.bat` permet de créer une base de données GraceTHD-MCD (schéma `gracethd`) et d'ajouter les schémas `gracethdcheck` et `gracethdcheckpub`.

Le script `gracethdcheck_pg_create_on_db.bat` permet d'ajouter la structure GraceTHD-Check sur une base de données GraceTHD existante. Mais normalement on utilise GraceTHD-Check sur une base créer "temporairement" pour contrôler un livrable.

Pour plus de détails, voir le manuel utilisateur.