

# PGcollect 1.6

Datenabgleich zwischen offline erhobenen und zentral gepflegten Datenbeständen

Nationalparkamt Hunsrück-Hochwald, RMK, 2021, 2022

## Zweck des Programmes

Auf mehreren Clientrechnern werden mit QGIS / Qfield Daten im Gelände offline erhoben und gepflegt. Die Daten werden als räumliche oder nicht-räumliche Tabellen zunächst lokal in SQLite oder Geopackages gehalten. Nach der Rückkehr ins Büro werden die Felddaten gesichert und in eine zentrale Postgis-Datenbank eingestellt. Dieser zentral gepflegte Datenbestand wird bei Bedarf wieder komplett zu den Feldrechnern zurückübertragen und offline erweitert und editiert. Es sind Punkt-, Linien- und Flächenobjekte möglich.

## Detailorganisation

PGcollect läuft ohne GUI und ohne Benutzereingaben im Batchbetrieb. Die Parametrisierung geschieht über eine DBF-Datei, in der jede Datenbanktabelle durch eine Zeile repräsentiert wird. Damit ist das Tool für die verschiedensten Projekte einsetzbar, ohne dass Anpassungen am Code notwendig sind.

PGcollect generiert mehrere Batches, die dann aus der Exe-Datei heraus aufgerufen werden. Die Software wurde in Xharbour geschrieben, die Zugriffe auf die Postgis-Datei nutzen die Funktionen des C-API, wodurch die Rückgabewerte zur Programmsteuerung und Fehlererkennung ausgewertet werden.

PGcollect setzt voraus, dass QGIS mit GDAL-Tools installiert ist und letztere in den Pfadeinstellungen eingetragen sind.

## Ablauf

- Die lokale Tabelle wird zunächst über das GDAL-Tool *ogr2ogr* in zwei Dateien exportiert. CSV für die Dateninhalte, CSVT für die Tabellenstruktur.
- Die CSV-Datei wird zeilenweise eingelesen und daraus ein UPDATE-Statement für das Bezugsobjekt gebildet, wobei in der WHERE-Klausel definiert ist, dass nur neuere Sätze (timestamp) berücksichtigt werden. Die Datentypen jeder Spalte werden der CSVT-Datei entnommen und in der Syntax berücksichtigt.
- nimmt ein definierbares Feld (Konfigurationsdatei) einen bestimmten Wert an, wird dieses Objekt auf der Postgis-Datenbank gelöscht.
- Ist das Objekt in der zentralen DB nicht vorhanden, erfolgt ein INSERT

- Bei entsprechender Konfiguration werden danach die Inhalte der lokalen Datenbank gelöscht und durch die Inhalte der zentralen DB ersetzt. (*ogr2ogr, Spatialite*)

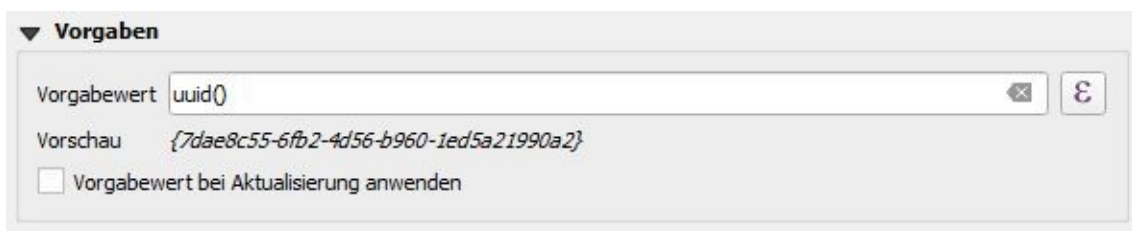
Alle Operationen einschliesslich eventueller Fehlermeldungen werden in ein Logfile geschrieben.

## Vorgehensweise

Das Projekt sollte auf einer lokalen Geopackage oder SQLITE Datenbank entwickelt werden, damit die Restriktionen bezüglich nutzbarer Datentypen eingehalten werden.

Jedes Objekt benötigt eine eindeutige ID und einen Zeitstempel. Dazu lassen sich händisch eingegebene Werte (z.B. Parkplatz-Name, Sensor-Nummer...) nutzen, sicherer ist es, diese Werte automatisch zu generieren. Im Aufnahmeformular können sie verborgen bleiben, lediglich die Vorgabewerte müssen definiert werden:

id, Zeichenkette, 40 Zeichen lang



Die id darf natürlich bei Aktualisierungen nicht geändert werden.

timestamp, numerisch integer Bit



Die Funktion `epoch()` gibt die Millisekunden seit dem 01.01.1970 zurück. Der Ausdruck `round(epoch(now()) / 10000, 0)` zählt also alle 10 Sekunden um 1 hoch. Bei jeder Aktualisierung muss der Zeitstempel mitgeführt werden. (Häkchen)

Steht die Datenstruktur fest, wird die lokale Tabelle aus QGIS heraus in die zentrale Postgis-DB geschrieben. Dabei können (im Gegensatz zu PGAdmin und anderen Programmen) einige Parameter festgelegt werden:

kleinbuchstaben!

Vektorlayer importieren

Eingabe: rettungspunkte-BB-2019

Nur gewählte Objekte importieren

Ausgabetable

Schema: wege

Tabelle: rettungspunkte-BB-2019

Optionen

Primärschlüssel: id

Geometriespalte: geom

Quell-SRID: EPSG:25832 - ETRS89 / UTM zone 32N

Ziel-SRID: EPSG:25832 - ETRS89 / UTM zone 32N

Kodierung: UTF-8

Zieltabelle ersetzen, falls vorhanden

Nicht mehrteilig machen

Feldnamen in Kleinschreibung umwandeln

Räumlichen Index erzeugen

Kommentar

OK Abbrechen

Für die Funktionalität von PGcollect wird für jede Datenbanktabelle in der Datei PGcollect.dbf eine Zeile eingefügt und die Felder gefüllt:

COMMENT

Kommentar. Ist das erste Zeichen ein # wird die Zeile übersprungen.

LDATEI

Lokale Datenbankdatei incl. Pfadangabe

LTABELLE

Name der lokalen Tabelle

LGEOM

Lokale Geometriespalte

SIP

Server-IP oder Servername der Postgis-Datei

SDB

Name der Postgis-Datenbank

SUSER

Benutzername der Postgis-Datenbank

SPASSWORT

Passwort der Postgis-Datenbank

## STABELLE

Name der Tabelle in der Postgis-Datenbank

## SGEOM

Geometriespalte in der Postgis-Datenbank

## SRID

Koordinatensystem z.B. 25832

## LOESCHKZ

SQL-Statement zum Löschen eines Objektes in der Postgis-Datenbank. Hier müssen Hochkommata (') als # geschrieben werden, da Erstere je nach verwendetem DBF-Editor durch andere Zeichen ersetzt werden. Beispiel: status = #geloescht#

## ZURUECK

Hier können die Zeichenketten *up*, *down* und *bi* stehen.

**Up** bewirkt nur einen Upload, der zentrale Datenbestand wird nicht rückübertragen.

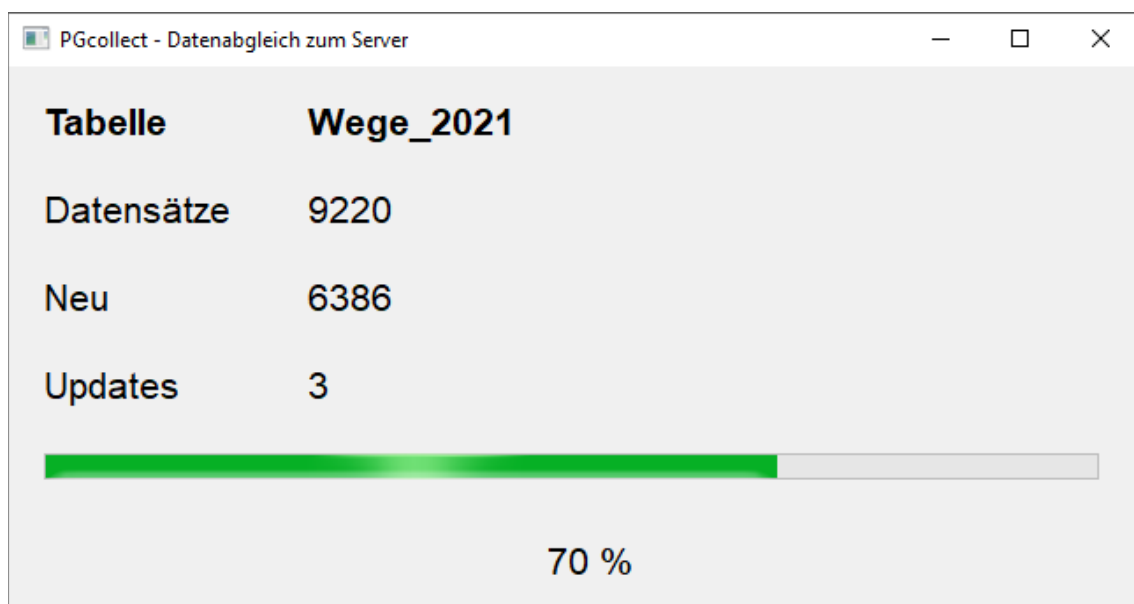
Bei **Down** wird der lokale Datenbestand gelöscht und durch die Daten des Servers ersetzt.

Der Wert **bi** führt einen bidirektionalen Datenabgleich durch.

Werden mehrere, verknüpfte Tabellen abgeglichen, kommt die Haupttabelle als letzte Zeile.

Hinweis: Es macht Sinn, PGcollect aus MDESich heraus zu starten (NACHLAUF.BAT), da so vorab der exklusive Dateizugriff, sowie die Verbindung zum Server geprüft werden und eine versionierte, filebasierte Sicherungskopie der lokalen Datenbank erstellt wird. Aus Benutzersicht werden alle Prozesse mit nur einem Klick im GUI gestartet.

Beim Hochladen der Daten informiert ein Fenster über den Stand des Programmablaufes:



## Installation

Das Archiv wird mit allen Dateien in ein Verzeichnis entpackt. Der Programmaufruf geschieht über die Datei PGcollect.exe . Prüfungen: sind *ogr2ogr* und *SPATIALITE* aufrufbar? Wenn nein: Pfad anpassen. Ist die zentrale Postgis-Datenbank erreichbar, d.h. Port 5432 offen?



### Kopiervorlagen:

Zeitstempel      `round(epoch(now()) / 10000,0)`

ID                `uuid()`