Database

- SQLite database omzetten naar Postgresql

SQLite database omzetten naar Postgresql

Standaard maakt Home Assistant gebruik van een SQLite database. Wanneer er veel informatie wordt weggeschreven is het verstandig om te kiezen voor een wat robuustere oplossing als database. In dit artikel wordt beschreven hoe je de bestaande SQLite database kan omzetten naar een Postgresql database en deze actief maakt in Home Assistant.

Benodigde pakketten installeren

Eerst installeren we de benodigde Debian pakketten:

```
apt-get install postgresql postgresql-server-dev-13 pgloader
```

Postgresql gebruiker maken voor homeassistant

Nu gaan we eerst een gebruiker maken binnen Postgresql speciaal voor de database die we gaan gebruiken voor Home Assistant:

```
sudo -u postgres createuser homeassistant
```

Negeer de melding: “could not change directory to “/root”: Permission denied”.

Vervolgens kennen we een wachtwoord toe voor de gebruiker (role) homeassistant:

```
su - postgres
# psql -c "ALTER USER homeassistant WITH PASSWORD 'securepass_here';"
```

Postgres database maken voor homeassistant

Nu maken we de database aan voor Home Assistant.

```
sudo -u postgres createdb -O homeassistant homeassistant
```

Hier wordt de database met de naam homeassistant gemaakt en de eigenaar is de gebruiker homeassistant.
Negeer wederom de melding: “could not change directory to “/root”: Permission denied”.

Python pakket psycopg2 installeren in de virtuele omgeving

Om Python te laten communiceren met Postgresql moeten we een pakketje installeren genaamd psycopg2.

Dit doen we in ons geval binnen de virtuele Python omgeving waar Home Assistant draait.
Eerst zorgen we dat we binnen de virtuele omgeving werken:

```
sudo -u homeassistant -H -s
cd /srv/homeassistant/
source bin/activate
```

Je moet nu in de virtuele omgeving zitten. Dit zie je aan de prompt:

```
(homeassistant) homeassistant@hass:/srv/homeassistant$
```

Vervolgens gaan we psycopg2 installeren binnen de virtuele omgeving:

```
pip3 install psycopg2
```

Wanneer dit succesvol is verlopen kan je de virtuele omgeving verlaten met <CTRL-D> of exit. Je komt dan weer in de normale omgeving terecht.

Home Assistant configuratie aanpassen

Nu gaan we de Home Assistant configuratie aanpassen zodat de Postgresql database gebruikt wordt in plaats van de SQLite database.
Open hiervoor het bestand configuration.yaml, in de editor. In ons geval is dat:

```
nano /home/homeassistant/.homeassistant/configuration.yaml
```

Plaats het volgende in dit configuratie bestand:

```
recorder:
  db_url: postgresql://@/homeassistant
```

De recorder integratie binnen Home Assistant is verantwoordelijk voor de opslag van gegevens in de database.

Herstart vervolgens Home Assistant om de nieuwe configuratie actief te maken:

```
systemctl restart home-assistant@homeassistant.service
```

Controleer de status met:

```
systemctl status home-assistant@homeassistant.service
```

Als alles goed is gegaan mogen er geen fouten optreden. Mocht er iets niet werken, bekijk dan de logfiles van Home Assistant en/of Postgresql. Home Assistant werkt nu met een “lege” Postgresql database. In onderstaande stappen gaan we de bestaande SQLite data overzetten naar de Postgresql database.

SQLite database omzetten naar Postgresql

Het overzetten van de SQLite database naar Postgresql doen we met de tool genaamd pgloader. We maken eerst een commando bestand aan die we straks aanroepen met pgloader. In dit bestand staan o.a. de regels vanuit welke database de gegevens gehaald moeten worden en in welke Postgresql database ze weggeschreven moeten worden. Het bestand slaan we even op onder de naam pglcommand. Open de editor:

```
nano pglcommand
```

Hier plaatsen we de volgende regels in:

```
load database
    from '/home/homeassistant/.homeassistant/home-assistant_v2.db'
    into postgresql://homeassistant:securepass_here@localhost/homeassistant

with data only, drop indexes, reset sequences, truncate

set work_mem to '16MB', maintenance_work_mem to '512 MB';
```

Pas bovenstaand bestand aan voor jouw situatie. Dus het juiste pad naar de Home Assistant database (home-assistant_v2.db) en de juiste Postgresql gebruiker met bijbehorend wachtwoord (homeassistant:securepass_here). In bovenstaande situatie gaan we er van uit dat de Postgresql machine lokaal draait en de database heet homeassistant (localhost/homeassistant).

Wanneer dit bestand is opgeslagen gaan we Home Assistant stoppen met:

```
systemctl stop home-assistant@homeassistant.service
```

Vervolgens gaan we de gegevens overzetten naar de Postgresql database met:

```
pgloader pglcommand
```

Pas bovenstaand commando eventueel aan met het juiste pad naar pglcommand.

Je zou nu een overzicht moeten krijgen met de tabellen die aangemaakt zijn en het aantal items die overgezet zijn. Zie je geen tabellen en staan de items op 0, controleer dan het pglcommand bestand.

Start vervolgens Home Assistant weer op met :

```
systemctl start home-assistant@homeassistant.service
```

Foutmeldingen na omzetten: “duplicate key value violates unique constraint”

Na het omzetten van de database constateerde ik veel foutmeldingen in de logfile van Postgresql (/var/log/postgresql/postgresql-13-main.log) en de logfile van Home Assistant. Deze foutmeldingen hadden allemaal dezelfde melding: **duplicate key value violates unique constraint**. Het betrof de onderstaande tabellen in Postgresql:

- statistics_short_term
- statistics_runs
- statistics

Dit bleek te maken te hebben met de waarden in de “id” kolom van de desbetreffende tabellen. De waarde die in deze kolom weggeschreven moet worden blijkt dan al te bestaan. Het lijkt er dus op dat deze “id” kolom niet leeg wordt gemaakt met het omzetten van de SQLite naar de Postgresql database. In mijn ogen lijkt dit toch wel te moeten gebeuren met het uitvoeren van pgloader. Hier staat immers de optie “reset sequences” in het commando bestand. Om het probleem op te lossen voeren we onderstaande SQL commando’s uit in de psql omgeving. Log dus eerst in als postgres gebruiker:

```
su - postgres
```

En ga vervolgens naar de psql omgeving:

```
psql
```

Je zit nu in de juiste omgeving. Vervolgens moeten we opgeven met welke database we willen gaan werken. In ons geval is dat de database “homeassistant”. Dit doen we met het volgende commando:

```
\c homeassistant
```

Je krijgt dan de melding “You are now connected to database “homeassistant” as user “postgres”. Hierna kunnen onderstaande SQL commando’s één voor één uitgevoerd worden. Dus na iedere regel een <ENTER> om het commando uit te voeren:

```
SELECT setval('statistics_short_term_id_seq', (SELECT MAX(id) FROM statistics_short_term)+1);
SELECT setval('statistics_runs_run_id_seq', (SELECT MAX(run_id) FROM statistics_runs)+1);
SELECT setval('statistics_id_seq', (SELECT MAX(id) FROM statistics)+1);
```

Deze commando’s zorgen er voor dat de hoogste waarde in de kolom “id” of “run_id” wordt opgezocht van de desbetreffende tabellen. Vervolgens wordt deze waarde met 1 opgehoogd en weggeschreven in de bijbehorende sequence tabel. Je zou hierna geen foutmeldingen meer mogen krijgen.

Om alle tabellen te bekijken in de homeassistant database kan je het volgende commando uitvoeren:

```
\d
```

Je krijgt dan onderstaand overzicht:

```
                      List of relations
 Schema |             Name              |   Type   |     Owner
--------+-------------------------------+----------+---------------
 public | events                        | table    | homeassistant
 public | events_event_id_seq           | sequence | homeassistant
 public | recorder_runs                 | table    | homeassistant
 public | recorder_runs_run_id_seq      | sequence | homeassistant
 public | schema_changes                | table    | homeassistant
 public | schema_changes_change_id_seq  | sequence | homeassistant
 public | states                        | table    | homeassistant
 public | states_state_id_seq           | sequence | homeassistant
 public | statistics                    | table    | homeassistant
 public | statistics_id_seq             | sequence | homeassistant
 public | statistics_meta               | table    | homeassistant
 public | statistics_meta_id_seq        | sequence | homeassistant
 public | statistics_runs               | table    | homeassistant
 public | statistics_runs_run_id_seq    | sequence | homeassistant
 public | statistics_short_term         | table    | homeassistant
 public | statistics_short_term_id_seq  | sequence | homeassistant
(16 rows)
```

Wil je zien welke kolommen er in een tabel zijn, dan kan je onderstaand commando uitvoeren:

```
\d <tabelnaam>
```