

Teedy with PostgreSQL

Teedy 1 was tested successfully with PostgreSQL Version 16 and lower

Install and configure PostgreSQL

Teedy requires at least PSQL 9.4 (PostgreSQL94Dialect)

PostgreSQL 10 and upwards are configured to deliver SSL by standard! You will need to configure its SSL cert!

```
sudo apt install -y postgresql postgresql-client libpq-dev postgresql-contrib
```

```
sudo vim /etc/postgresql/<VERSION>/main/pg_hba.conf
```

```
#add local to trust to omit password input. If you change to md5 you will need to enter
passwords if you run scripts (e.g. bash)
# "local" is for Unix domain socket connections only
local      all      all      trust
#host      all      all      0.0.0.0/0 md5
hostssl    all      all      0.0.0.0/0 md5
```

```
sudo vim /etc/postgresql/<VERSION>/main/postgresql.conf
```

```
listen_addresses = '*'           # what IP address(es) to listen on;
ssl = on
ssl_cert_file = '/etc/ssl/yourdomain.de.pem'
ssl_key_file = '/etc/ssl/private/yourdomain.de.key'
```

```
#login as postgres user
su - postgres
psql

CREATE USER teedy WITH PASSWORD 'password';
CREATE DATABASE teedy_db WITH ENCODING 'UNICODE' LC_COLLATE 'C' LC_CTYPE 'C' TEMPLATE
template0;
```

```
GRANT ALL PRIVILEGES ON DATABASE teedy_db TO teedy ;

#remove old database if required
#service postgresql restart #kick old connections
#REVOKE ALL PRIVILEGES ON DATABASE teedy_db FROM teedy;
#DROP DATABASE teedy_db;
#DROP USER teedy;
```

PostgreSQL SSL

```
cd /etc/letsencrypt/live/yourdomain.de/
cp privkey.pem /etc/ssl/private/yourdomain.de.key

(cat privkey.pem; printf "\n\n"; cat cert.pem; printf "\n\n"; cat chain.pem; printf "\n\n") >>
/etc/ssl/yourdomain.de.pem
cd /etc/ssl/

chgrp ssl-cert /etc/ssl/private/yourdomain.de.key
chmod 640 /etc/ssl/private/yourdomain.de.key
chgrp ssl-cert /etc/ssl/yourdomain.de.pem
chmod 640 /etc/ssl/yourdomain.de.pem

less /var/log/postgresql/postgresql-9.5-main.log #check for occuring errors belonging to SSL
cert
```

Configure dms.xml (optional)

```
vim /opt/jetty-home-11.0.15/jetty-base/webapps/dms.xml
```

```
<?xml version="1.0"?>
<!DOCTYPE Configure PUBLIC "-//Jetty//Configure//EN"
"http://www.eclipse.org/jetty/configure_10_0.dtd">
<Configure class="org.eclipse.jetty.webapp.WebAppContext">
  <Set name="contextPath">/dms</Set>
  <Set name="war"><SystemProperty name="jetty.data" default="."/>/webapps/dms.war</Set>
  <Call class="java.lang.System" name="setProperty">
    <Arg>docs.home</Arg>
    <Arg>/var/docs</Arg>
  </Call>
```

Configuration for usage of PostgreSQL instead H2

Note: The database connection is set via a central environment variable configuration for the entire Jetty service and cannot be set for individual WebApplicationContext.

Have a look at [Environment Configuration](#) on how to swap to PostgreSQL

External connection test with Oracle SQL Developer

- enable ports and configure firewall correctly
- download SQL Developer driver for PostgreSQL → <https://jdbc.postgresql.org/download.html#current> (Treiber = postgresql-42.2.6)
- add the driver in system settings
- User and database differ from the name. Therefore, the entry must be entered as follows:

The screenshot shows the 'New Database Connection' dialog in Oracle SQL Developer. The 'Name' field is set to 'dms.yourdomain.de'. The 'Datenbanktyp' (Database Type) is set to 'PostgreSQL'. The 'Benutzername' (Username) is 'teedy'. The 'Kennwort' (Password) field is masked with dots. The 'Kennwort speichern' (Save Password) checkbox is checked. The 'Hostname' field is 'nain.de:5432/teedy_db?'. The 'Port' field is empty. At the bottom, there is a 'Datenbank wählen' (Choose Database) button and a dropdown menu showing 'te'.

- user: teedy
- hostname: yourdomain.de:5432/teedy_db?
- port: stays empty
- database: teedy_db