



HighGo Postgres Server (HG-PGSQL) Installation and Quick Start Guide

Suitable for
HG-PGSQL v2.X

Last updated on
November, 2020

Table of Contents

1	INTRODUCTION	3
1.1	TYPOGRAPHICAL CONVENTIONS.....	3
2	REQUIREMENT	5
2.1	END USER LICENSE AGREEMENT.....	5
2.2	SUPPORTED PLATFORMS.....	5
2.3	INSTALLATION PREREQUISITES.....	5
2.4	DOWNLOAD INDIVIDUAL HG-PGSQL RPM PACKAGES.....	5
3	HG-PGSQL INSTALLATION ON CENTOS 7.X AND 8.X HOST	7
3.1	INSTALLATION OVERVIEW.....	7
3.2	INSTALL HG-PGSQL FROM REMOTE HIGHGO YUM REPOSITORY.....	8
3.3	LOCAL INSTALL HG-PGSQL FROM INDIVIDUALLY DOWNLOADED RPM FILES.....	9
3.4	UPGRADE HG-PGSQL v2.X.....	10
3.4.1	<i>Upgrade HG-PGSQL from Remote HighGo YUM Repository.....</i>	<i>10</i>
3.4.2	<i>Local Upgrade HG-PGSQL from Individually Downloaded RPM Files.....</i>	<i>10</i>
3.5	UPGRADE TO HG-PGSQL 2.0 FROM HG-PGSQL 1.0.....	11
3.5.1	<i>Install HG-PGSQL 2.0 Packages</i>	<i>11</i>
3.5.2	<i>Stop the HG-PGSQL Services.....</i>	<i>11</i>
3.5.3	<i>Check the Upgrade Compatibility without Modifying Data</i>	<i>11</i>
3.5.4	<i>Perform the Upgrade to HG-PGSQL 2.0.....</i>	<i>12</i>
3.5.5	<i>Start the HG-PGSQL Service.....</i>	<i>13</i>
3.5.6	<i>Ensure System is Stable after Upgrade.....</i>	<i>13</i>
3.5.7	<i>Delete Old Package and Data Files</i>	<i>14</i>
4	MANAGING HG-PGSQL INSTALLATION	15
4.1	MANAGING HG-PGSQL ON CENTOS 7.X AND 8.X HOSTS	15
4.1.1	<i>Perform Initdb</i>	<i>15</i>
4.1.2	<i>Start and Stop HG-PGSQL.....</i>	<i>15</i>
4.1.3	<i>Register HG-PGSQL to System Startup Process</i>	<i>16</i>
5	MANAGING MULTIPLE VERSIONS OF POSTGRES BINARIES WITH ALTERNATIVES	17
6	INSTALLATION TROUBLESHOOTING	18
7	CONFIGURING HG-PGSQL	19
7.1	MODIFYING THE POSTGRESQL.CONF	19
7.2	MODIFYING THE PG_HBA.CONF.....	19
7.3	CONNECTING HG-PGSQL WITH PSQL	20
8	UNINSTALLING HG-PGSQL.....	21
8.1	UNINSTALL HG-PGSQL ON CENTOS 7.X AND 8.X.....	21
8.1.1	<i>Uninstall Entire HG-PGSQL and Dependencies.....</i>	<i>21</i>
8.1.2	<i>Uninstall Individual HG-PGSQL Packages</i>	<i>21</i>

1 Introduction

This document is a comprehensive guide to installing Highgo Postgres Server (HG-PGSQL) versions 2.X on a range of supported platforms. In this guide, you will find detailed information about:

- HG-PGSQL Installation Prerequisites on Linux-based hosts
- Installing and updating HG-PGSQL using package manager
- Managing HG-PGSQL installation
- Configuring HG-PGSQL
- Uninstalling HG-PGSQL

1.1 Typographical Conventions

Certain typographical conventions are used in this document to distinguish various commands, statements, programs, examples ...etc. This section provides a summary of these conventions.

<i>Italic font</i>	Italic font represents a new term, typically in the sentence that defines it for the first time. When used in a sentence, it represents special warning or important messages
Fixed-width font	This font is used on user commands, inputs, SQL column names, programming keywords ...etc. For example: <code>SELECT pg_reload_conf();</code>
<i>Italic fixed-width font</i>	This font is used on terms in which the user must substitute a value in actual usage: For example: <code>DELETE FROM <i>table_name</i>;</code>
Dollar sign - \$	Dollar sign represents the start of a user or SQL commands, that the user can issue on a command line terminal. The dollar sign is commonly used with fixed-width font For example: <code>\$ service hg-pgsql12 start</code>
Vertical pipe - 	Vertical pipe denotes a choice between the terms on either side of the pipe. It is commonly used with square brackets or braces to separate two or more alternatives choices.
Square brackets - []	Square brackets denote that one or none of the enclosed terms may be substituted. Normally a vertical pipe is used within the square brackets to denote choices. For example:

Braces - { }	<p>[a b] means to choose one of “a” or “b” or none at all.</p> <p>Braces denote that exactly one of the enclosed terms must be specified. Normally a vertical pipe is used within the braces to denote choices.</p> <p>For example: { a b c } means exactly “a”, “b” or “c” must be specified.</p>
Ellipses - ...	<p>Ellipses denote that the preceding terms may be repeated. Normally a vertical pipe is used together to denote choices.</p> <p>For example: [a b] ... means that you may have the sequence, “a a a b b”.</p>

2 Requirement

2.1 End User License Agreement

Make sure you have read and agreed to the End User License Agreement (EULA) from the link below before installing and using HighGo Postgres Server.

<https://yum.highgo.ca/#license>

2.2 Supported Platforms

HighGo Postgres Server installation is supported on the following platforms

- CentOS (X86_64) 7.x
- CentOS (X86_64) 8.x

2.3 Installation Prerequisites

Prior to installing HG-PGSQL and its supporting components, you will need to install the HighGo yum repository entry on your system so that the *yum* utility is able to download the desired HG-PGSQL packages

```
$ yum -y install https://yum.highgo.ca/dists/rpms/repo/highgo-release-1.1-1.noarch.rpm
```

Upon successful installation, a new HighGo yum repository entry will be created at:

```
/etc/yum.repos.d/highgo.repo
```

And make sure the GPG key for HighGo is also created at:

```
/etc/pki/rpm-gpg/HIGHGO-SOFTWARE-GPG-KEY
```

Enable the hg-pgsql13 repo section

```
yum-config-manager --enable hg-pgsql13
```

2.4 Download Individual HG-PGSQL RPM Packages

This step is only required if you have decided to download individual HG-PGSQL *RPM* files and install them locally (i.e. not using remote HighGo yum repository).

The individual RPM files for all supported platforms can be downloaded from this link below:

<https://yum.highgo.ca/>

HG-PGSQL RPM files for CentOS 7.x:

- hg-pgsql13-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-plperl-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-contrib-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-plpython-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-debuginfo-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-pltcl-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-devel-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-server-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-docs-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-test-13.X-1.rhel7.x86_64.rpm
- hg-pgsql13-libs-13.X-1.rhel7.x86_64.rpm

HG-PGSQL RPM files for CentOS 8.x:

- hg-pgsql13-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-llvmjit-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-contrib-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-plperl-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-debuginfo-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-plpython-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-devel-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-pltcl-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-docs-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-server-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-libs-13.X-1.rhel8.x86_64.rpm
- hg-pgsql13-test-13.X-1.rhel8.x86_64.rpm

3 HG-PGSQL Installation on CentOS 7.x and 8.x Host

3.1 Installation Overview

This section describes the procedure to install HighGo Postgres Server CentOS 7.x and 8.x hosts.

Using package manager such as “**yum**” to install HighGo Postgres Server from HighGo’s yum repository is the recommended method of installation as it will automatically resolve the dependencies as it installs a package. Please ensure that you have done the prerequisites described in section 2 prior to proceeding this section. Alternatively, you may also download individual packages from <https://yum.highgo.ca/> and perform a local installation. Both ways will satisfy the installation requirement.

Upon successful installation, a new system user named “**highgo**” will be created with following parameters. This user will be the owner of all the database folders and files.

Field	Value	Comment
System User Name	highgo	
User ID	26	
Group ID	26	
Default Password	N/A	Use ‘passwd’ command to assign a password for the user.
Default Shell	/bin/bash	
Home Directory	/var/lib/highgo	
User Bash Profile	/var/lib/highgo/.bash_profile	This file stores the user specific Bash environment setups upon user login.

The RPM installer will place the Highgo Enterprise Server components in the following locations

Component	Location
Executables	/usr/local/highgo/hg-pgsql/13/bin
Libraries	/usr/local/highgo/hg-pgsql/13/lib
License and readme	/usr/local/highgo/hg-pgsql/13
Cluster configuration files	/var/lib/highgo/hg-pgsql-13/data
Contrib	/usr/local/highgo/hg-pgsql/13/lib
Data	/var/lib/highgo/hg-pgsql-13
Logs	/var/lib/highgo/hg-pgsql-13/data/log
Log rotation files	/var/lib/highgo/hg-pgsql-13/data/log
Lock files	/var/lock/subsys/
Backups	/var/lib/highgo/hg-pgsql-13/backups
Templates	/usr/local/highgo/hg-pgsql/13/share/
Development Headers	/usr/local/highgo/hg-pgsql/13/include/

Shared data	/usr/local/highgo/hg-pgsql/13/share/
Regression tests	/usr/local/highgo/hg-pgsql/12/lib/test/regress/
Procedural Language - TCL	/usr/local/highgo/hg-pgsql/12/lib
Procedural Language - Python	/usr/local/highgo/hg-pgsql/12/lib
Procedural Language - Perl	/usr/local/highgo/hg-pgsql/12/lib
Procedural Language - PostgreSQL	/usr/local/highgo/hg-pgsql/12/lib

After successful installation, the “**initdb**” process is required to be run to create the default database cluster. This process is described in detail in section 4 and will create a default database having the below parameters. Please note that the default database super user created during “**initdb**” and the system user created during installation are default to be the same (ie. **highgo**).

Field	Value	Comment
Super user	highgo	having attributes <ul style="list-style-type: none"> • Superuser • Create role • Create DB • Replication • Bypass RLS
Database name	Postgres	
encoding	UTF8	
collate	en_US.UTF-8	
C type	en_US.UTF-8	

We will use “**yum**” utility tool quite often to carry out the installation. Use “**yum --help**” on the command line or visit the link below for more detailed documentation.

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/deployment_guide/ch-yum

3.2 Install HG-PGSQL from Remote Highgo YUM Repository

Before you can install HG-PGSQL from remote HighGo yum repository, you must have the repository set up described in section 2.3. This is necessary as it allows “**yum**” to correctly locate and install the package and its components.

To install the core modules of HG-PGSQL:

```
$ yum -y install hg-pgsql13 \
hg-pgsql13-server
```

To install addition utilities and extensions of HG-PGSQL:

```
$ yum -y install hg-pgsql13-contrib \
                  hg-pgsql13-plpython \
                  hg-pgsql13-pltcl \
                  hg-pgsql13-plperl
```

To install optional development components and test suites

```
$ yum -y install hg-pgsql13-devel \
                  hg-pgsql13-docs \
                  hg-pgsql13-debuginfo \
                  hg-pgsql13-test
```

To install LLVM JIT

```
$ yum -y install hg-pgsql13-llvmjit
```

3.3 Local Install HG-PGSQL from Individually Downloaded RPM Files

HG-PGSQL can also be installed from individually downloaded RPM files. Please refer to section 2.4 for the list of RPM files required for download for each supported platform and the download link.

Please note that the example commands presented in this section uses the downloaded RPM packages having version information as **2.0** on CentOS 7.x platform. On CentOS 8.x platform or any other versions of release, the procedure is the same, except with different RPM file names.

Navigate to the directory containing all the downloaded RPM files

```
$ cd $PATH_TO_RPM
```

Local install the core modules of HG-PGSQL:

```
$ yum --disablerepo=* localinstall \
      hg-pgsql13-server-13.0-1.rhel7.x86_64.rpm \
      hg-pgsql13-13.0-1.rhel7.x86_64.rpm \
      hg-pgsql13-libs-13.0-1.rhel7.x86_64.rpm
```

Local install addition utilities and extensions of HG-PGSQL:

```
$ yum --disablerepo=* localinstall \
      hg-pgsql13-contrib-13.0-1.rhel7.x86_64.rpm \
      hg-pgsql13-plpython-13.0-1.rhel7.x86_64.rpm \
      hg-pgsql13-plperl-13.0-1.rhel7.x86_64.rpm \
      hg-pgsql13-pltcl-13.0-1.rhel7.x86_64.rpm
```

Local install optional development components and test suites

```
$ yum --disablerepo=* localinstall \
hg-pgsql13-devel-13.0-1.rhel7.x86_64.rpm \
hg-pgsql13-docs-13.0-1.rhel7.x86_64.rpm \
hg-pgsql13-debuginfo-13.0-1.rhel7.x86_64.rpm \
hg-pgsql13-test-13.0-1.rhel7.x86_64.rpm
```

Local install LLVM JIT

```
$ yum --disablerepo=* localinstall \
hg-pgsql13-llvmjit-13.0-1.rhel7.x86_64.rpm
```

3.4 Upgrade HG-PGSQL v2.x

The “**yum**” utility command can be used to perform upgrade on HG-PGSQL and its components. Like the installation procedure, the upgrade can also be done via remote HighGo repository or via individually downloaded RPM file



*Please note that yum upgrade can only be used to upgrade between minor versions only. To upgrade between major versions (ex. HG-PGSQL 1.0 to HG-PGSQL 2.0), **pg_upgrade** must be used. Refer to the HighGo PostgreSQL Server User Guide for Version 2.X for major version upgrade procedures.*

3.4.1 Upgrade HG-PGSQL from Remote HighGo YUM Repository

Given that you have set up the HighGo YUM Repository on your system according to section 2.3, the upgrade can be executed by this general command:

```
$ yum upgrade $PACKAGE_NAME
```

Where \$PACKAGE_NAME is the name of the available components of HG-PGSQL without version

For example:

```
$ yum upgrade hg-pgsql13
```

3.4.2 Local Upgrade HG-PGSQL from Individually Downloaded RPM Files

Given that you have downloaded all the newer versions of RPM files for HG-PGSQL, the local upgrade can be executed by this general command:

```
$ yum upgrade --disablerepo=* $RPM_FILE_NAME
Or
$ yum upgrade --disablerepo=* *.rpm
```

Where \$RPM_FILE_NAME is the name of the downloaded RPM file, including the component name, version and the (.rpm) extension.

For example:

```
$ yum upgrade --disablerepo=* *.rpm
```

3.5 Upgrade to HG-PGSQL 2.0 from HG-PGSQL 1.0



<< IMPORTANT >>

A major database version upgrade could be a risky process because of many code changes between the versions. Issues may occur during or after the upgrade and for this reason, please make sure you have done the proper back up of your data before attempting the upgrade.

3.5.1 Install HG-PGSQL 2.0 Packages

Refer to previous sections about the installation

3.5.2 Stop the HG-PGSQL Services

CentOS7 & 8

```
$ systemctl stop hg-pgsql-12  
$ systemctl stop hg-pgsql-13
```

CentOS6

```
$ service hg-pgsql-12 stop  
$ service hg-pgsql-13 stop
```

3.5.3 Check the Upgrade Compatibility without Modifying Data

Change to default database user “highgo”

```
$ su highgo
```

check the cluster compatibility

```
$ /usr/local/highgo/hg-pgsql/13/bin/pg_upgrade \  
--old-bindir /usr/local/highgo/hg-pgsql/12/bin \  
--new-bindir /usr/local/highgo/hg-pgsql/13/bin \  
--old-datadir /var/lib/highgo/hg-pgsql-12/data \  
--new-datadir /var/lib/highgo/hg-pgsql-13/data \  
--old-socket-dir /var/lib/highgo/hg-pgsql-12/sockets \  
--new-socket-dir /var/lib/highgo/hg-pgsql-13/sockets \  
--old-user /var/lib/highgo/hg-pgsql-12/owner \  
--new-user /var/lib/highgo/hg-pgsql-13/owner
```

```

--link -check

Performing Consistency Checks
-----
Checking cluster versions                                ok
Checking database user is the install user              ok
Checking database connection settings                  ok
Checking for prepared transactions                     ok
Checking for reg* data types in user tables            ok
Checking for contrib/isn with bigint-passing mismatch ok
Checking for presence of required libraries            ok
Checking database user is the install user              ok
Checking for prepared transactions                     ok
Checking for new cluster tablespace directories        ok

*Clusters are compatible*

```



The `--check` flag here instructs `pg_upgrade` to only check the upgrade without changing any data.

3.5.4 Perform the Upgrade to HG-PGSQL 2.0

Once the consistency check passes, go ahead, and perform the upgrade

```

$ /usr/local/highgo/hg-pgsql/13/bin/pg_upgrade \
  --old-bindir /usr/local/highgo/hg-pgsql/12/bin \
  --new-bindir /usr/local/highgo/hg-pgsql/13/bin \
  --old-datadir /var/lib/highgo/hg-pgsql-12/data \
  --new-datadir /var/lib/highgo/hg-pgsql-13/data \
  --link

Performing Consistency Checks
-----
Checking cluster versions                                ok
Checking database user is the install user              ok
Checking database connection settings                  ok
Checking for prepared transactions                     ok
Checking for reg* data types in user tables            ok
Checking for contrib/isn with bigint-passing mismatch ok
Creating dump of global objects                        ok
Creating dump of database schemas                      ok

Checking for presence of required libraries            ok
Checking database user is the install user              ok
Checking for prepared transactions                     ok
Checking for new cluster tablespace directories        ok

If pg_upgrade fails after this point, you must re-initdb the
new cluster before continuing.

Performing Upgrade
-----
Analyzing all rows in the new cluster                  ok
Freezing all rows in the new cluster                  ok

```

```

Deleting files from new pg_xact                                ok
Copying old pg_xact to new server                             ok
Setting next transaction ID and epoch for new cluster         ok
Deleting files from new pg_multixact/offsets                  ok
Copying old pg_multixact/offsets to new server                ok
Deleting files from new pg_multixact/members                  ok
Copying old pg_multixact/members to new server                ok
Setting next multixact ID and offset for new cluster          ok
Resetting WAL archives                                       ok
Setting frozenxid and minmxid counters in new cluster        ok
Restoring global objects in the new cluster                   ok
Restoring database schemas in the new cluster                 ok
Adding ".old" suffix to old global/pg_control                 ok

If you want to start the old cluster, you will need to remove
the ".old" suffix from /var/lib/highgo/hg-pgsql-12/data/global/pg_control.old.
Because "link" mode was used, the old cluster cannot be safely
started once the new cluster has been started.

Linking user relation files                                   ok

Setting next OID for new cluster                             ok
Sync data directory to disk                                  ok
Creating script to analyze new cluster                       ok
Creating script to delete old cluster                        ok

Upgrade Complete
-----
Optimizer statistics are not transferred by pg_upgrade so,
once you start the new server, consider running:
    ./analyze_new_cluster.sh

Running this script will delete the old cluster's data files:
    ./delete_old_cluster.sh

```



The `--link` flag creates hard links to the files on the old version cluster, so you do not need to copy data. If you do not wish to use the `--link` option, make sure that you have enough disk space to store 2 copies of files for both old version and new version clusters.

3.5.5 Start the HG-PGSQL Service

```
$ systemctl start hg-pgsql-13
```

3.5.6 Ensure System is Stable after Upgrade

Use the “analyze_new_cluster.sh” script to do a post-upgrade check. This script is generated by pg_upgrade and can be found at the directory in which you execute the pg_upgrade command.

```
$ ./analyze_new_cluster.sh
```

3.5.7 Delete Old Package and Data Files

Use the “delete_old_cluster.sh” script to remove the old data cluster. This script is generated by pg_upgrade and can be found at the directory in which you execute the pg_upgrade command.

```
$ ./delete_old_cluster.sh
```

And remove the old HG-PGSQL packages and data files

```
$ yum -y remove hg-pgsql12*
```

```
$ rm -rf /var/lib/highgo/hg-pgsql-12/data
```

4 Managing HG-PGSQL Installation

After the HG-PGSQL is installed, we will need to perform manual database initialization and register HG-PGSQL to the system's startup service. Depending on the platforms, the procedure to configure the installation differs.

4.1 Managing HG-PGSQL on CentOS 7.x and 8.x Hosts

4.1.1 Perform Initdb

Initdb must be run before the HG-PGSQL database can be put into use. This can be done by:

```
$ /usr/local/highgo/hg-pgsql/13/bin/hg-pgsql-13-setup initdb  
Initializing database: [ OK ]
```

This is the general form of initdb, you may also want to specify addition options to the initdb by using the environment variable "PGSETUP_INITDB_OPTIONS"

For example:

```
$ export PGSETUP_INITDB_OPTIONS="--auth-host=trust"  
$ /usr/local/highgo/hg-pgsql/13/bin/hg-pgsql-13-setup initdb  
Initializing database: [ OK ]
```

This will initialize the database with "--auth-host=trust" option

You can also assign a locale to the database cluster when invoking initdb by appending the locale string at the end of the initdb command. For example, to create a database cluster that uses simplified Chinese, use the command:

```
$ /usr/local/highgo/hg-pgsql/13/bin/hg-pgsql-13-setup initdb zh_CN.UTF-8  
Initializing database: [ OK ]
```

The initialized database cluster will be in /var/lib/highgo/hg-pgsql-13/data

4.1.2 Start and Stop HG-PGSQL

Once a default HG-PGSQL database has been initialized using the procedures described in previous sections, you can start and stop the HG-PGSQL by the commands:

```
$ systemctl start hg-pgsql-12  
$ systemctl stop hg-pgsql-12  
$ systemctl restart hg-pgsql-12
```

And check the status with

```
$ systemctl status hg-pgsql-12
```

4.1.3 Register HG-PGSQL to System Startup Process

To register HG-PGSQL to be automatically started up by the system command, run the following command:

```
$ systemctl enable --now hg-pgsql-13
```

You can check the startup process status by:

```
$ systemctl status hg-pgsql-13
● hg-pgsql-13.service - Highgo PostgreSQL 13 database server
   Loaded: loaded (/usr/lib/systemd/system/hg-pgsql-13.service;
          enabled; vendor preset: disabled)
   Active: active (running) since Mon 2020-11-18 23:19:03 UTC; 1min
          55s ago
```

Make sure the service is “**active**”

5 Managing Multiple Versions of Postgres Binaries with Alternatives

It is possible to install multiple versions of HG-PGSQL and community Postgres installations on the same host. The server binaries and libraries from each major version will be installed in separate directories so installing newer version does not overwrite the old.

“**Update-alternatives**” is a common utility command on Unix-based hosts designed to manage multiple versions of the same command line tools. This section explains how it is used in 7.x and 8.x hosts. Many other Unix-based hosts share similar procedures.

For example, in a host having both HG-PGSQL and community Postgres-13 installed, one can use “**update-alternatives**” to switch the versions of the front-end binaries.

Determine the list of front-end binaries managed by update-alternatives:

```
$ ls -ltr /etc/alternatives | grep pgsq
```

```
lrwxrwxrwx 1 root root 34 Nov 13 01:54 libnssckbi.so.x86_64 -> /usr/lib64/pkcs11/pl1-kit-trust.so
lrwxrwxrwx 1 root root 15 Nov 13 01:54 ld -> /usr/bin/ld.bfd
lrwxrwxrwx 1 root root 57 Nov 26 20:42 pgsqld-conf -> /usr/local/highgo/hg-pgsql/13/share/hg-pgsql-13-libs.conf
lrwxrwxrwx 1 root root 38 Nov 26 20:42 pgsq-psql -> /usr/local/highgo/hg-pgsql/13/bin/psql
lrwxrwxrwx 1 root root 47 Nov 26 20:42 pgsq-pg_basebackup -> /usr/local/highgo/hg-pgsql/13/bin/pg_basebackup
lrwxrwxrwx 1 root root 42 Nov 26 20:42 pgsq-dropuser -> /usr/local/highgo/hg-pgsql/13/bin/dropuser
lrwxrwxrwx 1 root root 40 Nov 26 20:42 pgsq-dropdb -> /usr/local/highgo/hg-pgsql/13/bin/dropdb
lrwxrwxrwx 1 root root 44 Nov 26 20:42 pgsq-createuser -> /usr/local/highgo/hg-pgsql/13/bin/createuser
lrwxrwxrwx 1 root root 42 Nov 26 20:42 pgsq-createdb -> /usr/local/highgo/hg-pgsql/13/bin/createdb
lrwxrwxrwx 1 root root 43 Nov 26 20:42 pgsq-clusterdb -> /usr/local/highgo/hg-pgsql/13/bin/clusterdb
lrwxrwxrwx 1 root root 42 Nov 26 20:42 pgsq-vacuumdb -> /usr/local/highgo/hg-pgsql/13/bin/vacuumdb
lrwxrwxrwx 1 root root 43 Nov 26 20:42 pgsq-reindexdb -> /usr/local/highgo/hg-pgsql/13/bin/reindexdb
lrwxrwxrwx 1 root root 44 Nov 26 20:42 pgsq-pg_restore -> /usr/local/highgo/hg-pgsql/13/bin/pg_restore
lrwxrwxrwx 1 root root 44 Nov 26 20:42 pgsq-pg_dumpall -> /usr/local/highgo/hg-pgsql/13/bin/pg_dumpall
lrwxrwxrwx 1 root root 41 Nov 26 20:42 pgsq-pg_dump -> /usr/local/highgo/hg-pgsql/13/bin/pg_dump
```

As an example, let us select a different alternative for front-end tool “**psql**”

```
$ update-alternatives --config pgsq-psql
```

There are 2 programs which provide ‘pgsq-psql’.

Selection	Command
* + 1	/usr/local/highgo/hg-pgsql/13/bin/psql
2	/usr/pgsq-13/bin/psql

Enter to keep current selection[+], or type selection number:

As you can see, we have two choices for psql, one from HG-PGSQL and the other from the community pgsq-13 and currently the system is using the psql version provided by HG-PGSQL.

The “**update-alternatives**” will prompt you for a selection, type ‘2’ followed by ‘enter’ key will immediate switch the “**psql**” front-end tool to use the version provided by the community psql-13.

After confirming the selection, when “**psql**” is invoked anywhere on the system, the one provided by the community psql-13 will be used.

Repeat the above steps to change the alternatives for other front-end binaries as needed.

6 Installation Troubleshooting

If you encounter an error message like this that prevents you from initializing the database:

```
Data directory is not empty!  
[FAILED]
```

or

```
The Installation Fails to Complete Due to Existing data Directory  
Contents
```

You will have to examine the data directory and make sure there is no folder or files there prior to running the **initdb** command.

On CentOS8, you may encounter this error during the initdb step:

```
Initdb fails with "error: Invalid locale settings"
```

You will need to reinstall the locale package using this command, then initdb should succeed.

```
$ yum install glibc-all-langpacks
```

7 Configuring HG-PGSQL

7.1 Modifying the postgresql.conf

postgresql.conf is the main configuration file that specifies HG-PGSQL behavior with regards to auditing, authentication, encryption, and several others. This file can be found at the installed **data** directory.

Parameters that are preceded by a pound sign (#) are set to their default value as shown. To change a parameter value, remove the pound sign and enter a new value. Please note that you must perform a **reload** or **restart** of the HG-PGSQL for new values to take effect

Important default settings:

Parameter	Default value
listen_addresses	localhost
Port	5333
log_destination	stderr
logging_collector	on
log_directory	log
log_filename	Postgresql-%a.log
log_truncate_on_rotation	on
log_rotation_age	1d

By default, HG-PGSQL logging is configured to write files to the “**log**” subdirectory of the “**data**” directory, rotating the files each day and retaining one week of log entries.

7.2 Modifying the pg_hba.conf

pg_hba.conf controls the authentication and security methods for incoming client connections. A client connection that does not satisfy the entries in pg_hba.conf will receive authentication failure error during connection.

This is the default entries in pg_hba.conf

```
# TYPE DATABASE USER ADDRESS METHOD
# "local" is for Unix domain socket connections only
local all all peer
# IPv4 local connections:
host all all 127.0.0.1/32 ident
# IPv6 local connections:
host all all ::1/128 ident
# Allow replication connections from localhost, by a user with the
# replication privilege.
local replication all peer
host replication all 127.0.0.1/32 ident
host replication all ::1/128 ident
```

Please ensure all trusted client connections are listed in this file. For example, if you would like allow a client connection from 192.168.15.200 to operate on database “db1” as user “client1” you will add this line in pg_hba.conf:

```
Host          db1          client1      192.168.15.200/32      trust
```

Make sure HG-PGSQL is reloaded to ensure new entries are loaded. You can reload HG-PGSQL by one of the following methods:

```
$ /usr/local/highgo/hg-pgsql/13/bin/pg_ctl reload
```

Or

```
$ psql -d postgres -U highgo -c "SELECT pg_reload_conf()"
```

7.3 Connecting HG-PGSQL with psql

psql is a command line client tool that allows you to execute SQL commands and obtain results. Given that HG-PGSQL is installed correctly, the psql should be in the search path and accessible from anywhere within the host. It is located under the executable folder of the HG-PSQL installation.

It is possible to use community version of psql to access a HG-PGSQL database server. See below example of community psql (v13.0) tool to access HG-PGSQL with default user credentials.

```
$ psql -d postgres -U highgo
Psql (13.0)
Type "help" for help.

Postgres=#
```

8 Uninstalling HG-PGSQL

8.1 Uninstall HG-PGSQL on CentOS 7.x and 8.x

8.1.1 Uninstall Entire HG-PGSQL and Dependencies

To remove the HG-PGSQL and its dependencies. Use the command:

```
$ yum remove hg-pgsql113
```

This command will resolve the dependency tree and presents to you the list of dependencies that will also be removed. Confirm the details and yum will remove the HG-PGSQL packages and its dependencies from the system

8.1.2 Uninstall Individual HG-PGSQL Packages

To Remove individual packages, use this general command, and replace `$package_name` with desired package to remove

```
$ yum remove $package_name
```

Where `$package_name` is the name of the package that you would like to remove. Without the `.rpm` extension and version information.



Please note that “yum” and “rpm” will not remove a package that is required by another package. If you attempt to remove a package that satisfies a package dependency, “yum” or “rpm” will provide a warning.

Refer to section 2.4 for a list of HG-PGSQL packages that is available for removal.



Copyright © 2020 Highgo Software, Inc. All rights reserved. HighGo Postgres Server®, HighGo DB®, HighGo DW®, HG Backup®, HData® and certain other marks are registered trademarks of Highgo Software, Inc., in Canada and other jurisdictions, and other HighGo names herein may also be registered and/or common law trademarks of HighGo. All other product or company names may be trademarks of their respective owners. Performance and other metrics contained herein or published on HighGo website were attained in internal lab tests under ideal conditions, and actual performance and other results may vary. Network variables, disk IOs and other conditions may affect performance results. Nothing herein represents any binding commitment by HighGo, and HighGo disclaims all warranties, whether express or implied, except to the extent HighGo enters a binding written contract, signed by HighGo's General Counsel, with a purchaser that expressly warrants that the identified product will perform according to certain expressly-identified performance metrics and, in such event, only the specific performance metrics expressly identified in such binding written contract shall be binding on HighGo. For absolute clarity, any such warranty will be limited to performance in the same ideal conditions as in HighGo's internal lab tests. In no event does HighGo make any commitment related to future deliverables, features or development, and circumstances may change such that any forward-looking statements herein are not accurate. HighGo disclaims in full any covenants, representations, and guarantees pursuant hereto, whether express or implied. HighGo reserves the right to change, modify, transfer, or otherwise revise this publication without notice, and the most current version of the publication shall be applicable.