Quick Links

Online Demo

Download

FAQ

Latest Docs

Ask a Question

Report a Bug

Screenshots

Warning: This documentation is for a pre-release version of pgAdmin 4

Contents

Import/Export Servers ¶

Server definitions (and their groups) can be exported to a JSON file and re-imported to the same or a different system to enable easy pre-configuration of pgAdmin. The `setup.py` script is used for this purpose.

Note

To export or import servers, you must use the Python interpreter that is normally used to run pgAdmin to ensure that the required Python packages are available. In most packages, this can be found in the Python Virtual Environment that can be found in the installation directory. When using platform-native packages, the system installation of Python may be the one used by pgAdmin.

Exporting Servers ¶

To export the servers defined in an installation, simply invoke `setup.py` with the `--dump-servers` command line option, followed by the name (and if required, path) to the desired output file. By default, servers owned by the desktop mode user will be dumped (pgadmin4@pgadmin.org by default - see the DESKTOP_USER setting in `config.py`). This can be overridden with the `--user` command line option. For example:

```
/path/to/python /path/to/setup.py --dump-servers
output_file.json

# or, to specify a non-default user name:

/path/to/python /path/to/setup.py --dump-servers
output_file.json --user user@example.com
```

To export only certain servers, use the `--servers` option and list one or more server IDs. For example:

```
/path/to/python /path/to/setup.py --dump-servers
output_file.json --server 1 2 5
```

Importing Servers

To import the servers defined in a JSON file, simply invoke `setup.py` with the `--load-servers` command line option, followed by the name (and if required, path) of the JSON file containing the server definitions. Servers will be owned by the desktop mode user (pgadmin4@pgadmin.org by default - see the DESKTOP_USER setting in `config.py`). This can be overridden with the `--user` command line option. For example:

```
/path/to/python /path/to/setup.py --load-servers
input_file.json

# or, to specify a non-default user name to own the new
servers:

/path/to/python /path/to/setup.py --load-servers
input_file.json --user user@example.com
```

If any Servers are defined with a Server Group that is not already present in the configuration database, the required Group will be created.

JSON format

The JSON file format used when importing or exporting servers is quite straightforward and simply contains a list of servers, with a number of attributes. The following attributes are required to be present in every server definition: Name, Group, Port, Username, SSLMode, MaintenanceDB and one of Host, HostAddr or Service.

Password fields cannot be imported or exported.

The following example shows both a minimally defined and a fully defined server:

```
{
    "Servers": {
        "1": {
            "Name": "Minimally Defined Server",
            "Group": "Server Group 1",
            "Port": 5432,
            "Username": "postgres",
            "Host": "localhost",
            "SSLMode": "prefer",
            "MaintenanceDB": "postgres"
        },
        "2": {
            "Name: "Fully Defined Server",
            "Group": "Server Group 2",
            "Host": "host.domain.com",
            "HostAddr": "192.168.1.2",
            "Port": 5432,
            "MaintenanceDB": "postgres",
            "Username": "postgres",
            "Role": "my_role_name",
            "SSLMode": "require",
            "Comment": "This server has every option
configured in the JSON",
            "DBRestriction": "live_db test_db",
            "PassFile": "/path/to/pgpassfile",
            "SSLCert": "/path/to/sslcert.crt",
            "SSLKey": "/path/to/sslcert.key",
            "SSLRootCert": "/path/to/sslroot.crt",
            "SSLCrl": "/path/to/sslcrl.crl",
            "SSLCompression": 1,
            "BGColor": "#ff9900",
            "FGColor": "#000000",
            "Service": "postgresql-10",
            "Timeout": 60,
            "UseSSHTunnel": 1,
            "TunnelHost": "192.168.1.253",
            "TunnelPort": 22,
            "TunnelUsername": "username",
            "TunnelAuthentication": 0
        }
    }
}
```