

---

# **OmniDB**

***Release 2.15.0***

**Oct 22, 2020**



---

## Contents:

---

<b>1</b>	<b>1. Introduction</b>	<b>1</b>
<b>2</b>	<b>2. Installation</b>	<b>3</b>
<b>3</b>	<b>3. omnidb-server Usability</b>	<b>5</b>
<b>4</b>	<b>4. Migrating From OmniDB 2 to 3</b>	<b>9</b>
<b>5</b>	<b>5. Deploying omnidb-server</b>	<b>11</b>
<b>6</b>	<b>6. Accessing OmniDB</b>	<b>15</b>
<b>7</b>	<b>Indices and tables</b>	<b>17</b>



# CHAPTER 1

---

## 1. Introduction

---

**OmniDB** is an open source browser-based app designed to access and manage many different Database Management systems, e.g. PostgreSQL, Oracle and MySQL. OmniDB can run either as an App or via Browser, combining the flexibility needed for various access paths with a design that puts security first.

Since early development, OmniDB was designed as a browser-based app. Consequently, it runs in most browsers, from any operational system. It can be accessed by several computers and multiple users, each one of them with his/her own group of connections. It also can be hosted in any operational system, without the need of install any dependencies. We will see further details on installation in the next chapters.

OmniDB's main objective is to offer an unified workspace with all functionalities needed to manipulate different DMBS. DBMS specific tools aren't required: in OmniDB, the context switch between different DBMS is done with a simple connection switch, without leaving the same page. The end-user's sensation is that there is no difference when he/she manipulates different DBMS, it just feels like different connections.

Despite this, OmniDB is built with simplicity in mind, designed to be a fast and lightweight browser-based application.



OmniDB provides 2 kinds of packages to fit every user needs:

- **OmniDB Application:** Runs a web server on a random port, and provides a simplified web browser window to use OmniDB interface without any additional setup. Just feels like a desktop application.
- **OmniDB Server:** Runs a web server on a port specified by the user. User needs to connect to it through a web browser. Provides user management, ideal to be hosted on a server on users' networks.

Both application and server can be installed on the same machine.

- Linux 64 bits
  - DEB installer
  - RPM installer
  - tar.gz package
- Windows 64 bits
  - EXE installer
- Mac OSX
  - DMG installer

Use the specific installer for your Operating System and it will be available through your desktop environment application menu or via command line with `omnidb-app` or `omnidb-server`.

On Linux, OmniDB Server installer will also create a service that allows you start and stop OmniDB.

You can also download the `tar.gz` package and extract it at your preferred location. You can then start OmniDB by running the extracted binary.

### 2.1 2.1. OmniDB Server

Here is an example of execution of `omnidb-server`:

```
user@machine:~$ omnidb-server
Starting OmnidB server...
Running database migrations...
Operations to perform:
  Apply all migrations: OmnidB_app, admin, auth, contenttypes, sessions, social_django
Running migrations:
  No migrations to apply.
Checking port availability...
Starting server OmnidB 3.0.0b at 127.0.0.1:8000.
Open OmnidB in your favorite browser
Press Ctrl+C to exit
```

Note how OmnidB starts a *web server* in port 8000. You can also specify port and listening address:

```
user@machine:~$ omnidb-server -p 8080 -H 0.0.0.0
Starting OmnidB server...
Running database migrations...
Operations to perform:
  Apply all migrations: OmnidB_app, admin, auth, contenttypes, sessions, social_django
Running migrations:
  No migrations to apply.
Checking port availability...
Starting server OmnidB 3.0.0b at 0.0.0.0:8080.
Open OmnidB in your favorite browser
Press Ctrl+C to exit
```

OmnidB will be accessible through any browser using the address displayed in the startup message.

More details about `omnidb-server` can be found in Chapter 3 of this documentation.

## 2.2 OmnidB With Oracle

OmnidB app and server does not require any piece of additional software, as explained above. But if you are going to connect to an *Oracle* database, then you need to download and install *Oracle Instant Client* (or extract it to a specific folder, depending on the operating system you use):

- **MacOSX:** Download Oracle Instant Client (64-bit) and extract in `~/lib`;
- **Linux:** Download Oracle Instant Client (32-bit) (64-bit) and install it on your system, then set `LD_LIBRARY_PATH`;
- **Windows:** Download Oracle Instant Client (32-bit) (64-bit) and extract it into OmnidB's folder.

**Note for Windows users using OmnidB app:** For OmnidB 2.8 and above, you will need to extract Oracle Instant Client libraries inside of folder `OMNIDBAPPINSTALLFOLDER\resources\app\omnidb-server`.



---

## 3. omnidb-server Usability

---

omnidb-server comes with several command line arguments to facilitate usage:

```
Usage: omnidb-server [options]

Options:
  --version          show program's version number and exit
  -h, --help        show this help message and exit

General Options:
  -d HOMEDIR, --homedir=HOMEDIR
                    home directory containing config and log files
  -C CONF, --configfile=CONF
                    configuration file
  -i, --init        Create home directory containing config and log files

Webserver Options:
  -H HOST, --host=HOST
                    listening address
  -p PORT, --port=PORT
                    listening port
  -P PATH, --path=PATH
                    path to access the application, other than /

Management Options:
  Options to list, create and drop users and connections.

  -M dbfile, --migratedatabase=dbfile
                    migrate users and connections from OmniDB 2 to 3: -M
                    dbfile
  -r, --resetdatabase
                    reset user and session databases
  -j, --jsonoutput  format list output as json
  -l, --listusers   list users
  -u username password, --createuser=username password
```

(continues on next page)

(continued from previous page)

```

        create user: -u username password
-s username password, --createsuperuser=username password
        create super user: -s username password
-x username, --dropuser=username
        drop user: -x username
-m username, --listconnections=username
        list connections: -m username
-c username technology title host port database dbuser, --
↪createconnection=username technology title host port database dbuser
        create connection: -c username technology host port
        database dbuser
-z connid, --dropconnection=connid
        drop connection: -z connid

```

- General Options: Options to provision/specify configuration and directory locations.
- Webserver Options: Webserver related settings.
- Management Options: Several options to manage the currently configured backend database.

### 3.1 3.1. OmniDB User Files

omnidb-server is supposed to be pointed to a directory that contains files being used at execution time:

- config.py: Configuration file.
- omnidb.db: Local database containing user session data and also OmniDB related metadata, if the user decides to deploy OmniDB using SQLite as backend database.
- omnidb.log: Log file, automatically rotated.

A directory is specified with: omnidb-server -d /path/to/dir. If not specified, OmniDB will use the default ~/.omnidb/omnidb-server.

When running omnidb-server for the first time, or pointing to a directory that does not contain existing files, OmniDB will create them for you and run. Users may prefer to run first omnidb-server --init, which will just create the default files allowing you can adjust settings before starting the application.

### 3.2 3.2. OmniDB Backend Database

OmniDB requires a database to store its metadata, containing user and connection details.

By default OmniDB uses SQLite, and will store its data in file omnidb.db located in the target runtime directory explained in the previous section.

Version 3.0 introduces the ability to deploy OmniDB using PostgreSQL as the backend database. This can be achieved by configuring config.py with the following section:

```

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'dbname',
        'USER': 'postgres',
        'PASSWORD': '',
        'HOST': '127.0.0.1',

```

(continues on next page)

(continued from previous page)

```

    'PORT': '5432',
}
}

```

Adjust the connection parameters accordingly.

When running `omnidb-server`, the target database will be configured and migrations will be executed to create required objects.

### 3.3.3. Authentication Methods

By default OmniDB authenticates users using its backend database.

OmniDB 3 introduces the ability to use different authentication methods. Version 3.0.0 comes with AD/LDAP, which is enabled by adding the following section in `config.py`:

```

import ldap
import django_auth_ldap.config
from django_auth_ldap.config import LDAPSearch

AUTH_LDAP_SERVER_URI = 'SERVER'
AUTH_LDAP_BIND_DN = "uid=example,dc=example,dc=com"
AUTH_LDAP_BIND_PASSWORD = "password"
AUTH_LDAP_USER_SEARCH = django_auth_ldap.config.LDAPSearch(
    "uid=example,dc=example,dc=com", ldap.SCOPE_SUBTREE, "uid=%(user)s"
)

AUTHENTICATION_BACKENDS = [
    'django_auth_ldap.backend.LDAPBackend',
    'django.contrib.auth.backends.ModelBackend'
]

```

Parameters must be adjusted according to the LDAP server being used.

OmniDB's LDAP authentication uses `django-auth-ldap` library, which contains several customization options. More details in: <https://django-auth-ldap.readthedocs.io/en/latest/>

If additional settings are needed, just amend `config.py` accordingly.

### 3.4.3.4. OmniDB Configuration File

Apart from database and authentication settings, the configuration file (`config.py`) contains all settings required to properly deploy OmniDB according to the user's environment.

- `LISTENING_ADDRESS`: specifies in what address the server will listen, the default value is `127.0.0.1`.
- `LISTENING_PORT`: specifies in what port OmniDB server will listen, this is the port used in the browser's URL if OmniDB is being accessed directly. The default value is `8000`.
- `CUSTOM_PATH`: specifies a custom path to access OmniDB in the browser URL. The default value is empty, meaning that no custom path is used. If user specifies `'test'` in this setting, OmniDB will be accessible with `http://127.0.0.1:8000/test`. This setting is useful when OmniDB is configured behind Apache or NGINX, so that all requests to the `CUSTOM_PATH` can be easily redirected to OmniDB.

- `IS_SSL`, `SSL_CERTIFICATE_FILE`, `SSL_KEY_FILE`: settings to configure SSL connection. If `IS_SSL` is `True` then certificate and key files must be provided.

The previous settings are enough to securely provision `omnidb-server`. The Configuration file contains additional settings for environments with additional configuration requirements.

---

## 4. Migrating From OmniDB 2 to 3

---

### 4.1 4.1 Backend Database

**IMPORTANT: Before running the migration**, make sure that OmniDB 3 already has a `config.py` file with backend database properly configured, as explained in chapter 3.

As explained in the previous chapter, OmniDB 3 can be deployed pointing to a PostgreSQL database as opposed to using the default SQLite.

Even if users decide to stick to SQLite, OmniDB 3 uses a completely different database schema so migration steps are needed.

If the user decides not to migrate users and connections, no further steps are needed.

In order to migrate users and connections `omnidb-server` comes with option `-M`:

```
-M dbfile, --migratedatabase=dbfile
    migrate users and connections from OmniDB 2 to 3: -M
    dbfile
```

OmniDB 2 stored its data in SQLite database file located by default in `~/.omnidb/omnidb-server/omnidb.db`. If being used as a service, the file was located in the home of the root user, `/root/.omnidb/omnidb-server/omnidb.db`. If users were pointing OmniDB 2 to a custom directory, location would be: `/path/to/dir/omnidb.db`.

Migration is done simply by pointing `omnidb-server` with the appropriate parameters to the location of the old file:

```
omnidb-server -M ~/.omnidb/omnidb-server/omnidb.db
Starting migration...
Creating user admin...
User admin already exists...
User admin already contains connections in the target database. Skip this user? (y/n) 
↩n
Delete existing connections of user admin from target database before migrating 
↩existing ones? (y/n) y
```

(continues on next page)

(continued from previous page)

```
Existing connections deleted.  
Attempting to create connections of user admin...  
Connection with alias 'connection 1' created.  
Connection with alias 'connection 2' created.  
Database migration finished.
```

## 4.2 4.2 Configuration File

As explained in chapter 3, OmniDB 3 uses the file `config.py` to retrieve custom settings.

OmniDB 2, on the other hand, used a file called `omnidb.conf`, which has a different syntax.

Users will have to manually edit the new configuration file with the appropriate settings.

---

## 5. Deploying omnidb-server

---

OmniDB's settings allows users to deploy `omnidb-server` in different scenarios.

This section will provide details on how to properly configure OmniDB in the following scenarios:

- Direct visibility: no applications between users accessing through the browser and OmniDB
- Behind a reverse proxy: OmniDB is only visible by the intermediate proxy application (Apache or NGINX)

Regardless of what method is used, it is **EXTREMELY** important that environment is configured so that communication between users browsers and the machine hosting OmniDB (or the intermediate application) is encrypted.

### 5.1 5.1. omnidb-server Post Installation

After installing `omnidb-server` in your preferred Linux distro, a service will be automatically configured.

If you read the third chapter of this doc, you will know that OmniDB is supposed to be started pointing (with `-d`) to a directory containing the configuration file.

OmniDB's service will **NOT** point to any specific directory so the default will be used, which is `~/.omnidb/omnidb-server` as `root` user. Make sure to edit that file if deploying OmniDB using the service.

If you're not using the service, edit the file that was created following the guidelines present in the third chapter (`omnidb-server Usability`).

### 5.2 5.2. Deploying OmniDB directly

In this case no reverse proxies are used, OmniDB is accessed directly.

For this scenario the user needs to specify the following parameters:

- `LISTENING_ADDRESS`: Specify the address visible to the clients, can be a domain.
- `LISTENING_PORT`: Specify a port that will be used in the browser url: `https://mydomain.com:PORT`

- IS\_SSL: True
- SSL\_CERTIFICATE\_FILE: /path/to/file
- SSL\_KEY\_FILE: /path/to/file

Authentication and database settings explained in the third chapter can also be configured according to the needs.

## 5.3 5.3. Deploying OmniDB behind a reverse proxy

In this case OmniDB won't be accessed directly but through a properly configured reverse proxy.

For this scenario a recommended approach is to run `omnidb-server` listening to the local address `127.0.0.1` and without SSL, given that proxy will handle the security part.

The following parameters are required:

- LISTENING\_ADDRESS: `127.0.0.1`.
- LISTENING\_PORT: Specify a port to which the load balancer will redirect all OmniDB server requests.

**IMPORTANT:** OmniDB will not use SSL but it is recommended that you also enable the following two settings:

- SESSION\_COOKIE\_SECURE: True
- CSRF\_COOKIE\_SECURE: True

These will make sure that the client connecting to OmniDB (through proxy) will only provide cookies if the connection is being done via HTTPS. Some browsers initially connect via HTTP so you may have a security breach without those settings.

Consider this example of OmniDB being hosted behind Nginx:

- Starting `omnidb-server`:

```
omnidb-server -d /home/user/omnidb_dir
```

- `/home/user/omnidb_dir`:

```
LISTENING_ADDRESS = '127.0.0.1'  
LISTENING_PORT    = 8000  
IS_SSL            = False  
SESSION_COOKIE_SECURE = True  
CSRF_COOKIE_SECURE = True
```

In this case OmniDB can only be accessed locally.

- NGINX configuration file:

```
server {  
    listen 443 ssl;  
    listen [::]:443 ssl;  
    include snippets/ssl-domain.conf;  
    include snippets/ssl-params.conf;  
    server_name domain.org;  
    client_max_body_size 75M;  
  
    location / {  
        proxy_pass http://127.0.0.1:8000;  
        proxy_set_header    X-Real-IP $remote_addr;
```

(continues on next page)



(continued from previous page)

```
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Ssl https;
proxy_set_header X-Forwarded-Proto https;
proxy_set_header X-Forwarded-Port 443;
proxy_set_header Host $host;
proxy_http_version 1.1;
proxy_set_header Upgrade $http_upgrade;
proxy_set_header Connection "upgrade";
}
}
```

As can be seen, NGINX is listening for requests to domain.org in port 443. All requests will be redirected to http://127.0.0.1:8000. Users will access OmniDB with https://domain.org

As explained in chapter three, users may want to configure CUSTOM\_PATH in config.py, which is useful when the domain is being shared with multiple applications.

For instance, if setting CUSTOM\_PATH = 'omnidb', NGINX can have the following redirection rule:

```
location /omnidb {
    proxy_pass http://127.0.0.1:8000/omnidb;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Ssl https;
    proxy_set_header X-Forwarded-Proto https;
    proxy_set_header X-Forwarded-Port 443;
    proxy_set_header Host $host;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection "upgrade";
}
```

And OmniDB is accessed with https://domain.org/omnidb



---

## 6. Accessing OmniDB

---

- Web version: OmniDB comes only with the user *admin*. If you are using the server version, the first thing to do is sign in as *admin*, the default password is *admin*.

**IMPORTANT:** create a new superuser, login with the new user and remove the old admin.

- App version: No login.

If accessing for the first time, you will see the Welcome screen containing several useful actions.

One of the most important features is accessible by clicking in the `Getting Started` button or in the small OmniDB icon at the bottom right of the screen. It will start `Omniis`, a helper that will guide you through the basics on the interface.



## CHAPTER 7

---

### Indices and tables

---

- `genindex`
- `modindex`
- `search`