

# Clustering Databases

## Overview

From [Wikipedia](#):

*High-availability clusters are groups of computers that support server applications that can be reliably utilized with a minimum of down-time. They operate by using high availability software to harness redundant computers in groups or clusters that provide continued service when system components fail.*

While there are many MySQL-based clustering solutions available, Ubersmith presently supports and advocates the use of [Percona XtraDB Cluster](#).

*Percona XtraDB Cluster is an active/active high availability and high scalability open source solution for MySQL clustering. It integrates Percona Server and Percona XtraBackup with the Galera library of MySQL high availability solutions in a single package [...]*

## Prerequisites

In versions of Ubersmith prior to the 3.3.0 release, application advisory locking was provided directly by the MySQL database server, by way of the [GET\\_LOCK](#) built-in function. Most modern MySQL-based clustering solutions do not allow the use of this function as it is not replicated across all nodes in the database cluster.

This limitation requires the use of a software component outside of Ubersmith and its associated database to provide a locking mechanism. Ubersmith has chosen [Redis](#) for this purpose.

*Redis is an open source, in-memory data structure store, used as a database, cache and message broker.*

## Redis Configuration

### Installing Redis

Ubersmith provides a Redis container configuration which can be brought online by running the following command as `root` from your Ubersmith installation directory:

```
docker-compose -p ubersmith up --scale redis=3 -d redis
```

This command will automatically bring up a cluster of three Redis containers.

### Updating Ubersmith's `config.ini.php`

Ubersmith's default behavior is to use MySQL's internal `GET_LOCK` function for advisory locking, so it's necessary to reconfigure the system to make it aware of the new Redis installation. Edit `docker-compose.override.yml` and uncomment the following lines:

```
# LOCK_BACKEND: redis
# LOCK_SERVERS: "ubersmith_redis_1:6379:1,ubersmith_redis_2:6379:1,ubersmith_redis_3:6379:1"
```

With those lines uncommented, run the following command as `root`:

```
docker-compose -p ubersmith up -d web
```

This will reconfigure Ubersmith to use Redis for advisory locking.

## Migration from standalone MySQL

### Installing Percona XtraDB Cluster

Percona's [manual](#) describes the process for installing and configuring XtraDB Cluster.

### HAProxy

Once the cluster is online, the redundant nature of the hosts can be leveraged by placing a load balancer such as [HAProxy](#) between the Ubersmith instance and the database hosts. Percona's documentation has a suggested configuration that directs `INSERT` and `UPDATE` queries to a single host, and directs `SELECT` queries to all nodes using either a 'round robin' or 'least connection' metric.

There are [other](#) load balancing options available, but at present Ubersmith advocates the use of HAProxy.

Ensure that the Ubersmith instance host can reach HAProxy and that HAProxy can reach each node on the database cluster. For a simple beginning configuration, it may be easiest to install and configure HAProxy directly on the Ubersmith instance host.

## Database Backup

Before beginning the migration to the new cluster, place Ubersmith in maintenance mode. Update or add the following entry to Ubersmith's `config.ini.php` file:

```
[maintenance]
enable = 1
```

Placing Ubersmith in maintenance mode ensures that a consistent database snapshot will be collected. Before starting the backup process, ensure that you have enough disk space to create a full dump of the Ubersmith database. Create a dump of the Ubersmith database using the `mysqldump` utility:

```
# mysqldump --opt --quote-names ubersmith > ubersmith_backup.sql
```

The command above assume a database name of `ubersmith` and that you do not need to provide login credentials to MySQL. You can verify your database name by reviewing the `dsn` configuration in `config.ini.php`, which follows this format:

```
[database]
dsn = mysql://USERNAME:PASSWORD@HOST/DATABASE
```

Depending on the size of the database and other factors, the `mysqldump` command may take several minutes to over an hour to run.

## Database Restore

Select one node in your cluster to restore the Ubersmith database to, and copy the `ubersmith_backup.sql` to it. Create the `ubersmith` database by running the following command from the MySQL command prompt:

```
mysql> CREATE DATABASE ubersmith;mysql> GRANT ALL ON ubersmith.* TO 'ubersmith'@'HOSTNAME' IDENTIFIED BY
'PASSWORD';
```

where `HOSTNAME` is the hostname of the Ubersmith instance and `PASSWORD` is a strong unique password.

The database is created on all nodes in the cluster.

To restore the database, issue the following command:

```
# mysql ubersmith < ubersmith_backup.sql
```

Depending on the size of the database, this restore may take up to an hour to complete.

## Go Live

Once the database restoration is complete, update `config.ini.php` again to modify the `dsn` entry to the load balancer previously configured. In the HAProxy example provided by Percona, the configuration should be:

```
dsn = mysql://ubersmith:PASSWORD@HAPROXY_ADDRESS/ubersmith
dsn2 = mysql://ubersmith:PASSWORD@HAPROXY_ADDRESS:3307/ubersmith
```

The addition of a `dsn2` line improves Ubersmith performance by offloading some reporting tasks to different nodes in the cluster.

Verify that the username, password, hostnames, and ports defined in your `config.ini.php` also work with the MySQL command line client. Finally, take Ubersmith out of maintenance mode by updating the configuration directive in `config.ini.php`:

```
[maintenance]
enable = 0
```

Ubersmith is now back online using Percona XtraDB Cluster.

## Resources

- <https://redis.io/>
- <https://www.percona.com/software/percona-xtradb-cluster>
- <http://www.haproxy.org/>
- <http://galeracluster.com/documentation-webpages/userguide.html>
- <http://dev.mysql.com/doc/refman/5.6/en/index.html>