

Transferring data from MySQL to PostgreSQL



Do I need to migrate my Qube data?

If both of the following conditions are true, you will need to take the extra steps outlined in this document during your supervisor upgrade process:

- You wish to bring your Qube 6.x data forward into Qube 7
- You are upgrading the supervisor from Qube 6.x to Qube 7 (for example from 6.10-0a to 7.0-0)



Minimum Qube version requirements for a successful data migration

Only data from Qube schema version 37 (present from 6.10-0 onwards) can be brought forward into Qube 7. If you're upgrading a Qube supervisor from a version prior to 6.10-0, you can upgrade only the schema to version 37; see the second phase of "Step Two: Export MySQL data to .csv"



Ensure that `database_port` is commented out (if present) in your supervisor's `qb.conf` file

We have seen cases with customers upgrading from Qube 6 to Qube 7 where the supervisor is attempting to connect to the PostgreSQL server on the default port for MySQL. This is due to the `database_port` being defined in the `qb.conf` file, but still set to the value for a MySQL server.

- Introduction:
- Step One: Stop All Activities on the Current Farm
- Step Two: Export Data from MySQL to .csv (Comma-separated values) files
- Step Three: Upgrade the Supervisor
- Step Four: Import Data into PostgreSQL from the .csv files
- Step Five: Enable Activities on the New Supervisor

Introduction:

One of the major feature updates to Qube 7.0 is the switch of the supervisor's database server from MySQL to PostgreSQL. Accordingly, when upgrading from a pre-7.0 version (such as 6.10-0a) to 7.0 or above, there are extra manual steps that must be taken outside of the Qube Installer (or the RPM/MSI/PKG installer, if you chose to bypass the QubeInstaller), particularly if you require that the old data from MySQL be transferred to PostgreSQL. This step-by-step document will guide you through the straightforward process.

Upgrading the supervisor and transferring the data is essentially a five-step process, and the order is critical:

1. Stop all activities on the current farm
2. Export data from MySQL to csv files on disk
3. Upgrade the supervisor
4. Import data into PostgreSQL from the csv files
5. Enable activities on the new supervisor



IMPORTANT!

The order of these steps are crucial! Note that there are steps to be taken *before upgrading the supervisor*.



Python on Windows

Python 2.x (2.6 or above) is now a prerequisite for supervisor installation. It comes standard on Linux and Mac, but you will have to install it on Windows in advance. Also **make sure that python is added to the System PATH**. It's an opt-in feature for the Python.org MSI installer. <https://www.python.org/downloads/>

Step One: Stop All Activities on the Current Farm

You will want to first stop all current farm activities before doing the upgrade of the Supervisor. Run the following commands on a command prompt with a qube administrator account to prevent the supervisor from dispatching jobs, and stop accepting new job submissions from users:

```
qbadmin supervisor --set stop_activity  
qbadmin supervisor --set reject_submit
```

You will also want stop workers from picking up new work. If you want to allow workers to finish up their current frames, do:

```
qblock --all
```

If you'd rather stop running jobs immediately, do:

```
qblock --all --purge
```

Then wait for all activities to stop.

Step Two: Export Data from MySQL to .csv (Comma-separated values) files



Estimate time for data export

The time it takes for the export depends on many parameters, such as your supervisor system's hardware (especially the speed of the database and export destination disk), the number of existing jobs, and number of frames (aka agenda items) for each job. For reference, on a Mac system with an SSD, every 256 jobs, each with 100 frames, took about 6 seconds to process. That would add up to **about 4 minutes to process 10,000 jobs**.

In this step, you'll be using a script that we provide to dump MySQL data to .csv files into a folder on disk.

1. Make sure that the MySQL server is running, and that you can connect to it using the mysql client, and that your qube table version is at 37.
 - a. If you haven't changed the database administrator user and password, you should be able to do the following on a command prompt to confirm that the MySQL server is running:

```
Linux: /usr/bin/mysql -u root -e 'SELECT * FROM qube.tableversion'
```

```
Mac: /usr/local/mysql/bin/mysql -u root -e 'SELECT * FROM  
qube.tableversion'
```

```
Windows: "C:\Program Files\pfx\qube\mysql\bin\mysql" -u root -e "SELECT *  
FROM qube.tableversion"
```

- b. Make sure that the above command works and returns:

```
+-----+
| version |
+-----+
| 37      |
+-----+
```

2. If you get something less than 37 returned by the above command, it means that your current Qube supervisor version is older than 6.10-0, and that you need to update your MySQL database tables first, before you can upgrade the database schema. To do so:
 - a. Download the **"upgrade_supervisor"** program suitable for your supervisor platform from http://repo.pipelinefx.com/downloads/pub/db_migration_tools/
 - b. On a command prompt, run the **upgrade_supervisor** program that you just downloaded.
 - i. On Windows, you will need to unzip the `upgrade_supervisor_WIN32-6.1-x64.zip` file first, and run the `upgrade_supervisor.bat` file found in the unzipped folder.
 - c. Check that there weren't any critical errors reported by **upgrade_supervisor**.
 - d. Check that the version is now indeed updated to 37, by running the `mysql -u root -e 'SELECT * FROM qube.tableversion'` command again
3. Choose a destination folder on your supervisor for the MySQL csv files. **Make sure that your user and the mysql server process both have write permission to this folder and all its parent folders**, and that the volume is sufficiently large. Also note that a faster disk, such as an SSD, will help speed up the export/import process.



On CentOS 7.x and possibly other Linux distros, create a working directory under /opt and do the export while running as the root user

Do **NOT** use `/tmp`, `/var/tmp` (`/usr/tmp`), or any subdirectories under them. These OSs give the MySQL service its own private `/tmp` and `/var/tmp` folders, which prevents the `mysqldump` command from running correctly. **Creating a subdirectory under /root does not work either, nor will a subdirectory in any user's home directory**, since non-root users home directories are usually mode 700, so the MariaDB server can't access it.

One approach that **doeswork** is creating a directory under `/opt` and opening up the permissions:

```
sudo mkdir -p /opt/mysql_dump
sudo chmod 755 /opt/mysql_dump
```

Then, install the **export_data_from_mysql.py** script from the next step into this directory as the root user, and run the export script as root.

4. Download the **export_data_from_mysql.py** script from http://repo.pipelinefx.com/downloads/pub/db_migration_tools/ and copy it into the destination folder.
5. On a command prompt, go to the destination folder, and run **export_data_from_mysql.py**. Running it without any argument will create a subfolder in the current directory named **"qube_mysqlbackup"** and dump all files into it.

```
python export_data_from_mysql.py
```

- a. You may override the dump subfolder and DB username, password, and mysql install location. Run **"export_data_from_mysql.py -h"** to see the list of options.
6. Sit back. This process can take a long time to complete, depending on how many jobs you have on the system.
7. Once the process completes, make sure there were no errors reported on the terminal. Also have a look at the dump directory to confirm that there is a subfolder **"qube"** and a bunch of subfolders like **"<number>qube"**.
8. Take a note of the dump directory location, and proceed to the next step, "Upgrade the Supervisor".

Proceed with the upgrade of the supervisor software. Using the QubeInstaller is recommended, but you can also run the individual installer packages (RPMs, DEBs, MSIs, or PKGs), should you choose.

Step Three: Upgrade the Supervisor

See [Upgrading Qube!](#) for details, but come back here after upgrading the supervisor software.

Step Four: Import Data into PostgreSQL from the .csv files



Do this before making any change to your farm, or submitting new jobs.



Estimated time for data import

We have found that **the import takes roughly 1/4 of the time for the export**. Importing 10,000 jobs with 100 frames on average on a Mac system with an SSD took about 33 seconds.

Importing the previously exported data

Once you upgrade the supervisor, you are ready to import data into the new PostgreSQL server.

1. Make sure that PostgreSQL server is running, and accepting connections:

```
Linux: /usr/local/pfx/pgsql/bin/psql -p 50055 -d pfx -U qube -c "SELECT * FROM
qube.tableversion"

Mac: /Applications/pfx/pgsql/bin/psql -p 50055 -d pfx -U qube -c "SELECT * FROM
qube.tableversion"

Windows: "C:\Program Files\pfx\pgsql\bin\psql" -p 50055 -d pfx -U qube -c "SELECT
* FROM qube.tableversion"
```

Note that this should return:

```
version
-----
      51
(1 row)
```

2. On a command prompt, **go to the folder where you ran the export script earlier**. This should be the parent folder of the "qube_mysqldump" folder, by default.
3. Run the **import_data_into_pgsql.py** script to import data from the csv files that were generated earlier.

```
Linux: python /usr/local/pfx/qube/utils/pgsql/import_data_into_pgsql.py

Mac: python /Applications/pfx/qube/utils/pgsql/import_data_into_pgsql.py

Windows: python "C:\Program Files\pfx\qube\utils\pgsql\import_data_into_pgsql.py"
```

4. Sit back. This process will also take some time to complete, although it should be significantly faster than the export.
5. Make sure there weren't any errors reported on the terminal.

Step Five: Enable Activities on the New Supervisor

Run the following commands to enable the new supervisor to accept new jobs and start dispatching jobs to workers

```
qbadmin supervisor --unset stop_activity
qbadmin supervisor --unset reject_submit
```

You'll also need to unlock the workers you want to start using again. If you'd like to unlock all workers, then do:

```
qbunlock --all
```

Congratulations, you are done. Enjoy the new ride!