

# Migrating to Xpand from MySQL

This section discusses steps and some of the best practices recommendations for migrating an application or set of applications that are currently deployed on MySQL database(s) to Xpand.

- [Migration Overview](#)
- [Migration Prerequisites](#)
- [Migration Steps](#)
  - [Take a consistent dump of the database](#)
  - [Import the database dump using clustrix\\_import utility](#)
  - [Migrate Permissions with clustrix\\_clone\\_users](#)
  - [Start Replication Slave on Xpand](#)
- [Application Server Cutover](#)
  - [Methods to cut over application servers](#)
- [Risk Mitigation](#)
  - [Validating Xpand Compatibility](#)
  - [Enable failing back to MySQL](#)
    - [Reverting to MySQL](#)
  - [Best practice during cutover](#)
- [Cutover Caveats](#)
  - [Replication lag](#)
  - [Rogue servers/scripts](#)

## Migration Overview

There are essentially three basic steps to achieve successful migration from MySQL environment:

- Dump the MySQL database with mysqldump and import into Xpand with clustrix\_import
- Use MySQL replication to sync Xpand with the production MySQL database
- Cut over application servers to Xpand

There are several additional steps which may be taken to minimize risk:

- Validate that Xpand responds to all read queries appropriately
- Configure Xpand slave to facilitate switching back to MySQL as a roll back process

## Migration Prerequisites

In order to migrate your application from MySQL to Xpand, the following must be true:

- MySQL server has binary logging enabled (--log-bin and other supporting arguments)
- Using SBR mode replication during the migration process has the benefit of validating that write queries are handled properly by Xpand. However, if the application workload is characterized to be extremely busy and/or exhibits many write heavy transactions, Xpand recommends using RBR mode to achieve better replication throughput.
- Check whether tables being migrated are InnoDB or MyISAM. MyISAM requires some special care to get a consistent dump. All tables must be locked or the database must be quiesced completely since MyISAM provides no transaction isolation.

## Migration Steps

### Take a consistent dump of the database

- It is important to note that database dump would need to be taken using mysqldump utility. No other existing backup methods e.g. LVM snapshots, xtrabackup, etc. may be used for migrating the database.
- mysqldump command to be used for dumping MySQL database is provided below

```
mysql> mysqldump -u user -h mysql_host --single-transaction --master-data=2 --all-databases > mydumpfile.dump
```

Please note that --single-transaction argument is important in order to get a consistent snapshot from where to start the replication slave. Additionally, --master-data argument stores the binlog position corresponding to the snapshot in the dump file.

It is recommended to use the linux screen window manager to insure that the session is not killed before the backup finishes (better than the & and nohup alternative). For monitoring the dump and ensure successful completion, the tail command may be used. Using tail on the dump file should show something like:

```
-- Dump completed on 2016-08-02 19:50:56
```

If the dump is incomplete or incorrect due to wrong usage of mysqldump arguments, lots of time may be wasted before finding out that replication won't work. **Correctness of mysqldump command is critical.**

## Import the database dump using clustrix\_import utility

- `clustrix_import` command to be used for importing the dump file is provided below

```
shell> clustrix_import -i dumpfile.sql -H
clustrix_ip
```

- A few best practices recommended for using `clustrix_import` command line tool
  - Use `screen`
  - tee the output
  - Pay close attention to the final output indicating success or failure

`clustrix_import` has many advantages over `mysql` client in loading data as it imports data in parallel, taking full advantage of cluster resources. This tool is also designed to optimally distribute the data across all Xpand nodes and automatically retries transient errors.

## Migrate Permissions with clustrix\_clone\_users

- `mysqldump --all-databases` will dump the `mysql` database but Xpand cannot use this data to instantiate users
- Use `clustrix_clone_users` instead, available at `/opt/clustrix/bin/`.

`clustrix_clone_users` utility will query a `MySQL` (or Xpand) database to dump the users and permissions, generating `SQL` which can then be imported, per this example.

```
shell> ./clustrix_clone_users -H localhost > /tmp/grants.sql
shell> head /tmp/grants.sql
--
-- Clustrix Users dumpfile ver: 113:82f8694c98db
-- Host: localhost
--
GRANT ALL PRIVILEGES ON *.* TO 'mysql_slave'@'' WITH GRANT OPTION;
GRANT ALL PRIVILEGES ON *.* TO 'clustrix_ui'@'127.0.0.1' IDENTIFIED BY PASSWORD
'*46A23F3EF4B5568CD0D6951239A0345A78DDF61A' WITH GRANT OPTION;
GRANT ALL PRIVILEGES ON `statd`.* TO 'statd'@'%' IDENTIFIED BY PASSWORD
'*58D9255AEB513581F38430D559A1455461E6B74E';
shell> mysql -h mogwai -u root < /tmp/grants.sql
```

## Start Replication Slave on Xpand

Once the import is complete, the slave can be created on Xpand using the following command:

```
sql> CREATE SLAVE 'slavel' MASTER_LOG_FILE = 'foo.000001'
, MASTER_LOG_POS = 4
, MASTER_HOST = 'host_name'
, MASTER_USER = 'user_name'
, MASTER_PASSWORD = 'password';

sql> START SLAVE 'slavel';
```

The proper log file and position are obtained from the beginning of the `mysqldump` (as generated by `--master-data=2` argument).

Please refer to [Configuring Replication](#) module for obtaining information on monitoring slave status.

## Application Server Cutover

### Methods to cut over application servers

There are two common methods for switching app servers from `MySQL` to Xpand:

1. Reconfiguring application servers to point to Xpand instead of `MySQL`
2. Using an external load balancer e.g. `HAProxy` to direct traffic to Xpand instead of `MySQL`. Please refer [Load Balancing Xpand with HAProxy](#).

## Risk Mitigation

### Validating Xpand Compatibility

To reduce cutover surprises, ensure that Xpand properly handles all queries generated by the application

- To validate **write** statements, SBR mode of replication ensures Xpand slave's ability to handle write queries. Although RBR will provide better performance for write heavy workloads, it is recommended to use SBR initially for validation of write queries. SBR mode also comes handy in troubleshooting replication issues during initial adoption. Once all statements are validated, SBR mode can be converted to RBR for better replication performance.
- To validate **read** statements, SQL queries may be captured either using tcpdump utility or enabling full query logging. For tcpdump output, Xpand support can help converting it into valid SQL sessions. Thereafter, these queries can be replayed using MySQL client for validation.

## Enable failing back to MySQL

The ability to switch back to MySQL greatly minimizes the risk of impact to production. Configuring MySQL to slave from Xpand beforehand ensures a smooth transition should the need arise.

Steps to fail over are discussed under [Configuring Replication Failover](#) section and also outlined below:

1. Retain privileges for all application logins on the MySQL slave instance, but keep the instance read only, by setting read\_only global:

```
slave> SET GLOBAL read_only = true;
```

2. Enable binlogging on Xpand, ensuring format is same as MySQL master. To enable failing back to MySQL, the binlog must be created on Xpand before application writes are allowed.

```
CREATE BINLOG b
inlog_name
[format='row']
```

3. Once application servers have been cut over to Xpand, the slave on Xpand (from MySQL master) can be stopped. Alternatively, bi-directional, or master-master replication can be configured, where Xpand continues to replicate from MySQL, while MySQL also replicates from Xpand. This is a more complex configuration, with some caveats, as discussed in [Configuring Replication Failover](#).
4. Configure MySQL as a slave from Xpand, using MySQL's CHANGE MASTER TO syntax, specifying the beginning (position 4) of the binlog created in step 2.

## Reverting to MySQL

In the event that it becomes necessary to revert to MySQL, given the steps above have been taken, the following steps are necessary:

1. Change Xpand to read only mode:

```
master> SET GLOBAL read_only = true;
```

2. Ensure slave has caught up by comparing binlog file and position shown by SHOW MASTER STATUS on Xpand and SHOW SLAVE STATUS on MySQL
3. Recreate or re-enable Xpand slave from MySQL, specifying current binlog file and position shown in MySQL's SHOW MASTER STATUS.
4. Enable MySQL to take writes again:

```
slave> SET GLOBAL read_only = false;
```

5. Application servers can now be pointed back to MySQL.

## Best practice during cutover

Configure read-only on the inactive side (slave and root users are exempted):

```
sql> SET GLOBAL read_only = true;
```

## Cutover Caveats

### Replication lag

Make sure that replication is caught up before cutting over. If Xpand is significantly behind, an auto-increment INSERT coming from the newly cut-over app server will conflict with a prior INSERT in the replication stream. In order to avoid getting into such issues, it may be necessary to quiesce the MySQL server for some period of time to ensure that the Xpand slave is caught up before cutting over.

### Rogue servers/scripts

Post cutover there could still be some applications or scripts that are still attempting to write to the MySQL database. These would be either failing as MySQL instance is set to read\_only mode or silently manipulating data as root. Examples of such possibilities could be admin application deployed locally on MySQL server or some of the database admin kind of shell scripts that are executed locally as root by the DBAs and manipulates data. It is necessary to review and migrate those scripts to Xpand instance going forward as otherwise it would introduce data mismatch between Xpand and MySQL