

# Database Migrations

When managing production-grade services, the ability to perform reliable and repeatable database migrations is critical. Whether you're applying schema changes, updating seed data, or managing version-controlled transitions, Elestio provides a built-in mechanism to execute migrations safely from the dashboard. This functionality is especially relevant when running containerized database services like PostgreSQL, MySQL, or similar within a managed cluster.


## Need for Migrations

Database migrations are commonly required when updating your application's data model or deploying new features. Schema updates such as adding columns, modifying data types, creating indexes, or introducing new tables need to be synchronized with the deployment lifecycle of your application code.

Migrations may also be needed during version upgrades to introduce structural or configuration changes required by newer database engine versions. In some cases, teams use migrations to apply baseline datasets, adjust permissions, or clean up legacy objects. Running these changes through a controlled migration system ensures consistency across environments and helps avoid untracked manual changes.

## Running Database Migration

To run a database migration in Elestio, start by logging into the [Elestio dashboard](#) and navigating to the **Clusters** section. Select the cluster that contains the target database service. From the **Cluster Overview** page, scroll down until you find the **"Migration"** option.

mysql-320dd

MySQL

Cluster

Running

> Open terminal


Delete cluster

Add node

OverviewNodesBackupsAudit


Termination protection

Disabled. VM can be powered off and terminated.

Protection deactivated 

Auto-Failover

Enabled. In case of failure, the cluster will automatically attempt to recover

Auto-Failover activated 

Nodes

2 Nodes: 1 Primary, 1 Replica

Add node

Database Admin

Display your database credentials

Display DB Credentials

Support plan

Level1

Upgrade plan

Resync Cluster

Resync cluster on all nodes.

Resync Cluster

Migration


Migrate database

Show migration logs

Migrate Database

Clicking this option will open the migration workflow, which follows a **three-step process**: **Configure**, **Validation**, and **Migration**. In the **Configure** step, Elestio provides a migration configuration guide specific to the database type, such as MySQL. At this point, you must ensure that your target service has sufficient **disk space** to complete the migration. If there is not enough storage available, the migration may fail midway, so it's strongly recommended to review storage utilization beforehand.

Migrate database



●

Validation

●

Configure

Validation

Migration

MySQL migration configuration guide

Before you start the migration, you need to ensure that your target service has enough disk space to migrate your database.

Cancel

Get started

Once configuration prerequisites are met, you can proceed to the **Validation** step. Elestio will check the secondary database details you have provided for the migration.

Migrate database

X

✓

Configure

Validation

Migration

Please provide the connection details from your source database

Enter hostname

Enter port

Enter Database name

kaiwalya@elest.io

.....

Back

Run check

If the validation passes, the final **Migration** step will become active. You can then initiate the migration process. Elestio will handle the actual data transfer, schema replication, and state synchronization internally. The progress is tracked, and once completed, the migrated database will be fully operational on the target service.

## Considerations Before Running Migrations

- Before running any migration, it's important to validate the script or changes in a staging environment. Since migrations may involve irreversible changes—such as dropping columns, altering constraints, or modifying data—careful review and version control are essential.

- In production environments, plan migrations during maintenance windows or low-traffic periods to minimize the impact of any schema locks or temporary unavailability. If you're using replication or high-availability setups, confirm that the migration is compatible with your architecture and will not disrupt synchronization between primary and secondary nodes.
  - You should also ensure that proper backups are in place before applying structural changes. In Elestio, the backup feature can be used to create a restore point that allows rollback in case the migration introduces issues.
- 

Revision #1

Created 23 April 2025 15:50:26 by kaiwalya

Updated 23 April 2025 15:57:02 by kaiwalya