Contents

[Step-by-Step Guide: Installing MyDumper 0.11.3 with MySQL 8.0.27 on Ubuntu 20.04 2](#_Toc191294082)

[Required Files 2](#_Toc191294083)

[Step 1: Remove Existing MySQL Client and MyDumper Installations 3](#_Toc191294084)

[1. Remove Existing MySQL Client Development Packages 3](#_Toc191294085)

[2. Remove Any Existing MyDumper Installation 3](#_Toc191294086)

[Step 2: Install MySQL 8.0.27 Client Libraries 3](#_Toc191294087)

[1. Install the MySQL Client Runtime Library 3](#_Toc191294088)

[2. Install the MySQL Client Development Package 3](#_Toc191294089)

[3. Fix Any Dependency Issues 3](#_Toc191294090)

[4. Verify MySQL Client Installation 3](#_Toc191294091)

[Step 3: Extract MyDumper Source Code 3](#_Toc191294092)

[Step 4: Configure the Build with CMake 4](#_Toc191294093)

[1. Create and Enter a Build Directory 4](#_Toc191294094)

[2. Run CMake 4](#_Toc191294095)

[3. Install Sphinx (Optional - For Documentation) 4](#_Toc191294096)

[Step 5: Compile MyDumper 4](#_Toc191294097)

[Step 6: Install MyDumper (Optional) 4](#_Toc191294098)

[Step 7: Verify Installation 5](#_Toc191294099)

[1. Check MyDumper Version 5](#_Toc191294100)

[2. Check MyLoader Version 5](#_Toc191294101)

[Step 8: Perform Backup Using MyDumper 5](#_Toc191294102)

[Parameter Explanation 5](#_Toc191294103)

[Step 9: Restore Backup Using MyLoader 6](#_Toc191294104)

[Parameter Explanation 6](#_Toc191294105)

[Best Practices for Backup & Restore 6](#_Toc191294106)

[Performance Optimization 6](#_Toc191294107)

[Data Consistency 6](#_Toc191294108)

[Logging and Monitoring 6](#_Toc191294109)

[Commonly Used MyDumper & MyLoader Options 7](#_Toc191294110)

[MyDumper - Backup Command Options 7](#_Toc191294111)

[MyLoader - Restore Command Options 8](#_Toc191294112)

[Backup & Restore Performance Summary with MyDumper/MyLoader 9](#_Toc191294113)

[Performance Comparison: Compressed vs. Uncompressed Backup & Restore 10](#_Toc191294114)

[Key Observations: 10](#_Toc191294115)

[Conclusion: 10](#_Toc191294116)

# Step-by-Step Guide: Installing MyDumper 0.11.3 with MySQL 8.0.27 on Ubuntu 20.04

This guide provides a step-by-step approach to installing **MyDumper 0.11.3**, ensuring compatibility with **MySQL 8.0.27** on Ubuntu 20.04.

### Required Files

Download the following files before proceeding:

* **MySQL Client Runtime Library**
  + [libmysqlclient21\_8.0.27-1ubuntu20.04\_amd64.deb](https://downloads.mysql.com/archives/get/p/23/file/libmysqlclient21_8.0.27-1ubuntu20.04_amd64.deb)
* **MySQL Client Development Package**
  + [libmysqlclient-dev\_8.0.27-1ubuntu20.04\_amd64.deb](https://downloads.mysql.com/archives/get/p/23/file/libmysqlclient21_8.0.27-1ubuntu20.04_amd64.deb)
* **MyDumper Source Code**
  + [mydumper-0.11.3.tar.gz](https://github.com/mydumper/mydumper/archive/refs/tags/v0.11.3.tar.gz)

## Step 1: Remove Existing MySQL Client and MyDumper Installations

### 1. Remove Existing MySQL Client Development Packages

sudo apt remove --purge libmysqlclient-dev libmysqlclient21  
sudo apt autoremove

### 2. Remove Any Existing MyDumper Installation

sudo rm -f /usr/local/bin/mydumper /usr/local/bin/myloader

## Step 2: Install MySQL 8.0.27 Client Libraries

### 1. Install the MySQL Client Runtime Library

Ensure you are in the directory where the .deb files are located, then run:

sudo dpkg -i libmysqlclient21\_8.0.27-1ubuntu20.04\_amd64.deb

### 2. Install the MySQL Client Development Package

sudo dpkg -i libmysqlclient-dev\_8.0.27-1ubuntu20.04\_amd64.deb

### 3. Fix Any Dependency Issues

sudo apt-get install -f

### 4. Verify MySQL Client Installation

mysql\_config --version

**Expected output:**

8.0.27

## Step 3: Extract MyDumper Source Code

tar xzf mydumper-0.11.3.tar.gz  
cd mydumper-0.11.3

## Step 4: Configure the Build with CMake

### 1. Create and Enter a Build Directory

mkdir build  
cd build

### 2. Run CMake

cmake -DMYSQL\_CONFIG=/usr/bin/mysql\_config \  
 -DMYSQL\_LIBRARIES\_ssl="/usr/lib/x86\_64-linux-gnu/libssl.so" \  
 -DMYSQL\_LIBRARIES\_crypto="/usr/lib/x86\_64-linux-gnu/libcrypto.so" \  
 ..

### 3. Install Sphinx (Optional - For Documentation)

If warnings related to Sphinx documentation appear, install it:

sudo apt install python3-sphinx

## Step 5: Compile MyDumper

make -j$(nproc)

## Step 6: Install MyDumper (Optional)

sudo make install

This installs mydumper and myloader in /usr/local/bin.

## Step 7: Verify Installation

### 1. Check MyDumper Version

mydumper --version

**Expected output:**

mydumper 0.11.3, built against MySQL 8.0.27

### 2. Check MyLoader Version

myloader --version

**Expected output:**

myloader 0.11.3, built against MySQL 8.0.27

## Step 8: Perform Backup Using MyDumper

Run the following command to back up your database:

mydumper --host=localhost --user=<username> --password=<password> \  
--outputdir=./backup --rows=100000 --compress --build-empty-files \  
--threads=4 --compress-protocol --trx-tables \  
--regex '^(<Db\_name>\.)' -L mydumper-logs.txt

### Parameter Explanation

* --outputdir=./backup → Backup directory
* --rows=100000 → Splits tables into chunks of 100,000 rows for parallel processing
* --compress → Compresses backup files
* --threads=4 → Uses 4 parallel threads (adjust based on CPU cores)
* --regex '^(<Db\_name>\.)' → Filters by database name
* -L mydumper-logs.txt → Logs output to mydumper-logs.txt
* --trx-tables → Optimizes for transactional tables

## Step 9: Restore Backup Using MyLoader

Run the following command to restore your database:

myloader --host=localhost --user=<username> --password=<password> \  
--directory=./backup --queries-per-transaction=500 --threads=4 \  
--compress-protocol --verbose=3 --overwrite-tables 2>myloader-logs.txt

### Parameter Explanation

* --directory=./backup → Backup directory
* --queries-per-transaction=500 → Executes 500 queries per transaction
* --threads=4 → Uses 4 parallel threads
* --overwrite-tables → Drops existing tables before restoring
* 2>myloader-logs.txt → Redirects errors to myloader-logs.txt

## Best Practices for Backup & Restore

### Performance Optimization

* **Threads:** Adjust --threads to match CPU cores for better performance.
* **Chunk Size:** Use --rows to optimize data chunking and parallelism.

### Data Consistency

* **Use --trx-tables** for transaction consistency in backups.
* **Enable --lock-all-tables** for consistency across non-transactional tables.

### Logging and Monitoring

* **Use -L mydumper-logs.txt** for detailed logging in MyDumper.
* **Redirect stderr output** to logs in MyLoader (2>myloader-logs.txt).

# Commonly Used MyDumper & MyLoader Options

### MyDumper - Backup Command Options

#### **Connection Options**

* -h, --host → MySQL server hostname
* -u, --user → MySQL username
* -p, --password → MySQL password
* -a, --ask-password → Prompt for user password
* -P, --port → TCP/IP port (default: 3306)

#### **Database & Output Options**

* -B, --database → Comma-separated list of databases to dump
* -o, --outputdir → Directory for backup files

#### **Data Splitting & Optimization**

* -r, --rows → Split tables into chunks (format: MIN:START\_AT:MAX, MAX=0 for no limit)
* -F, --chunk-filesize → Split data files into pieces (in MB)
* -c, --compress → Compress output files (GZIP or ZSTD, default: GZIP)

#### **Schema & Data Dump Options**

* -m, --no-schemas → Dump only data, skip table schemas
* -d, --no-data → Dump only schemas, skip table data
* -x, --regex → Filter databases/tables using regex
* -G, --triggers → Include triggers in the backup
* -E, --events → Include events in the backup
* -R, --routines → Include stored procedures & functions

#### **Logging & Execution Control**

* -L, --logfile → Log output file (default: stdout)
* --tz-utc → Maintain TIMESTAMP consistency (--skip-tz-utc to disable)
* --use-savepoints → Reduce metadata locking issues (SUPER privilege required)
* -U, --updated-since → Dump only tables updated in the last U days

#### **Multi-Threading & Performance**

* -t, --threads → Number of threads (0 uses all CPUs, default: 4)
* -C, --compress-protocol → Use compression on MySQL connection

#### **Miscellaneous**

* -V, --version → Show program version
* -v, --verbose → Set verbosity level (0 = silent, 1 = errors, 2 = warnings (default), 3 = info)

### MyLoader - Restore Command Options

#### **Connection Options**

* -h, --host → MySQL server hostname
* -u, --user → MySQL username
* -p, --password → MySQL password
* -a, --ask-password → Prompt for user password
* -P, --port → TCP/IP port (default: 3306)

#### **Restore Options**

* --directory → Backup directory
* -s, --source-db → Database to restore
* -B, --database → Restore into a different database
* -o, --overwrite-tables → Drop existing tables before restoring
* -T, --tables-list → Comma-separated table list for restore (e.g., test.t1,test.t2)

#### **Data & Schema Control**

* -x, --regex → Filter databases/tables using regex
* --no-data → Do not import table data
* --no-schema → Do not import table schemas & triggers
* --skip-triggers → Exclude triggers from restore
* --skip-indexes → Exclude secondary indexes on InnoDB tables

#### **Performance Optimization**

* -r, --rows → Split INSERT statements into chunks
* -q, --queries-per-transaction → Queries per transaction (default: 1000)
* -t, --threads → Number of threads (0 uses all CPUs, default: 4)

#### **Logging & Debugging**

* --show-warnings → Display warnings during INSERT IGNORE
* -V, --version → Show program version
* -v, --verbose → Set verbosity level (0 = silent, 1 = errors, 2 = warnings (default), 3 = info)

### Backup & Restore Performance Summary with MyDumper/MyLoader

#### **Test Conditions:**

* **Data Size:** ~600 MB (compressed)
* **Database Engine:** MySQL 8.0.27
* **Tool:** MyDumper 0.11.3 / MyLoader
* **System:** Ubuntu 20.04

#### **Performance Comparison:**

| Threads | Backup Start | Backup End | Backup Duration | Restore Start | Restore End | Restore Duration |
| --- | --- | --- | --- | --- | --- | --- |
| **2 Threads** | 12:11 | 12:15 | **4 mins** | 12:17 | 12:47 | **30 mins** |
| **8 Threads** | 10:57 | 11:00 | **3 mins** | 11:02 | 11:31 | **29 mins** |

#### **Observations & Insights:**

1. **Backup Time:**
   * Increasing threads from 2 to 8 reduced backup time by **1 minute** (4 mins → 3 mins).
   * Since the backup was already small (600 MB), the improvement was minimal.
2. **Restore Time:**
   * Increasing threads from 2 to 8 **only reduced restore time by 1 minute** (30 mins → 29 mins).
   * This suggests that MySQL write operations were likely the bottleneck, not the MyLoader threads.
3. **Performance Bottlenecks:**
   * **Backup is CPU-bound** (compression and data dumping). More threads helped slightly.
   * **Restore is disk I/O-bound** (inserts and indexing). More threads had minimal impact.

### Performance Comparison: Compressed vs. Uncompressed Backup & Restore

| **Threads** | **Compression** | **Backup Start** | **Backup End** | **Backup Time** | **Backup Size** | **Restore Start** | **Restore End** | **Restore Time** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2 | ✅ Compressed | 12:11 | 12:15 | **4 min** | 600 MB | 12:17 | 12:47 | **30 min** |
| 8 | ✅ Compressed | 10:57 | 11:00 | **3 min** | 600 MB | 11:02 | 11:31 | **29 min** |
| 8 | ❌ Uncompressed | 12:05 | 12:06 | **1 min** | **6.9 GB** | 12:14 | 12:42 | **28 min** |

### Key Observations:

* **Backup Speed:**
  + **Uncompressed backup (1 min) is significantly faster** than compressed (3-4 min).
  + Compression adds overhead but reduces storage usage.
* **Backup Size:**
  + **Uncompressed (6.9 GB) is ~11.5x larger** than compressed (600 MB).
* **Restore Speed:**
  + Uncompressed restore (28 min) is slightly faster than compressed restore (29-30 min).
  + Decompression overhead is minimal compared to MySQL’s write operations.

### Conclusion:

* **Use Compression** if storage is a concern (saves ~90% space with minimal impact on restore time).
* **Skip Compression** if backup speed is critical and storage is not an issue.
* **More Threads Improve Backup Speed** but have little impact on restore time due to MySQL’s write constraints.