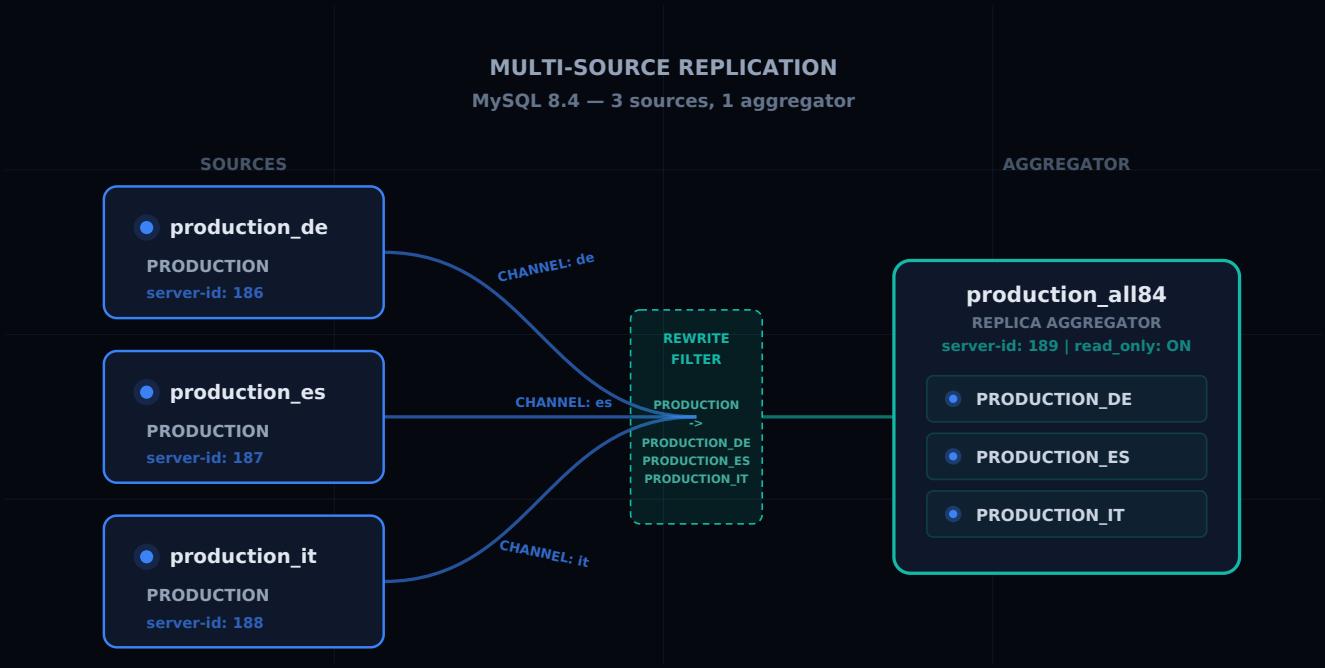
Multi-Source репликация с MySQL 8.4

Aurélien LEQUOY · April 12, 2026

MYSQL REPLICATION MULTI-SOURCE MYSQL-8.4



Введение

Multi-source репликация MySQL 8.4 позволяет одной и той же реплике получать транзакции от нескольких серверов-источников параллельно. Каждый источник привязан к реплике через отдельный канал репликации.

Этот механизм используется в основном для:

- консолидации нескольких серверов в одном узле
- агрегации потоков из разных стран, площадок или приложений
- централизации данных для чтения
- подготовки узла отчётности или сверки

При этом это не кластер с общей записью. MySQL не выполняет разрешение конфликтов между несколькими источниками. Если два источника пишут в одни и те же логические объекты, согласованность должна обеспечиваться на стороне приложения или через

строгое разделение данных.

Что реально поддерживает MySQL 8.4

В MySQL 8.4:

- multi-source реплика открывает канал для каждого источника
- каждый канал должен указывать на отдельный источник
- репликация может быть основана на GTID или на позициях binlog
- фильтры репликации могут применяться по каналам
- репозитории метаданных реплики должны быть в режиме `TABLE`, что является поведением по умолчанию в 8.4

Важные моменты:

- multi-source предназначен для консолидации, а не для multi-primary с арбитражем
- встроенного обнаружения или разрешения конфликтов нет
- одна реплика не может открыть несколько каналов к одному источнику

Пример топологии

Простой пример с тремя источниками и агрегатором:

```
production_de  ─┐
production_es  ─┼──> production_all84
production_it  ─┘
```

В этом примере:

- `production_de` содержит базу `PRODUCTION`
- `production_es` тоже содержит базу `PRODUCTION`
- `production_it` тоже содержит базу `PRODUCTION`
- `production_all84` получает три потока, но перенаправляет их в разные базы:
  - `PRODUCTION_DE`
  - `PRODUCTION_ES`
  - `PRODUCTION_IT`

Такое перенаправление предотвращает запись трёх потоков в одну базу на реплике.

Предварительные требования

На каждом источнике:

- уникальный `server-id`
- включённый бинарный лог
- TCP/IP-доступ к порту MySQL
- выделенный пользователь репликации

На multi-source реплике:

- уникальный `server-id`
- настроенный `relay_log`
- MySQL 8.4
- начальная загрузка данных перед запуском репликации

Типичная конфигурация источников

Пример минимальной конфигурации источника:

```
[mysqld]
bind-address = 0.0.0.0
server-id = 186
log_bin = mysql-bin
binlog_format = ROW
binlog_row_image = FULL
sync_binlog = 1
innodb_flush_log_at_trx_commit = 1
```

Аналогичная логика на других источниках с разным `server-id` на каждом сервере.

Типичная конфигурация реплики-агрегатора

Пример:

```
[mysqld]
bind-address = 0.0.0.0
server-id = 189
log_bin = mysql-bin
relay_log = mysql-relay-bin
binlog_format = ROW
binlog_row_image = FULL
skip_replica_start = ON
read_only = ON
super_read_only = ON
```

`skip_replica_start=ON` полезен во время фазы настройки или обслуживания, так как предотвращает автоматический перезапуск каналов до подтверждения.

Создание учётной записи репликации

На каждом источнике:

```
CREATE USER 'repl'@'10.68.68.%' IDENTIFIED BY 'Repl84Geo2026x';
GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'repl'@'10.68.68.%';
```

Привилегия `REPLICATION SLAVE` остаётся документированной MySQL для этого типа настройки в 8.4.

Начальная загрузка данных

Перед подключением канала необходимо загрузить начальные данные на реплику.

Пример с дампами, снятыми с источников:

```
mysqldump --single-transaction --set-gtid-purged=OFF PRODUCTION > PRODUCTION_de.sql
mysqldump --single-transaction --set-gtid-purged=OFF PRODUCTION > PRODUCTION_es.sql
mysqldump --single-transaction --set-gtid-purged=OFF PRODUCTION > PRODUCTION_it.sql
```

Затем на реплике:

```
CREATE DATABASE PRODUCTION_DE;
CREATE DATABASE PRODUCTION_ES;
CREATE DATABASE PRODUCTION_IT;
```

И импорт трёх дампов в три соответствующие целевые базы.

Получение координат binlog

Если GTID не используется, необходимо получить для каждого источника:

- имя файла binlog
- начальную позицию

Пример:

```
SHOW BINARY LOG STATUS;
```

Реплика затем стартует с этих координат через `SOURCE_LOG_FILE` и `SOURCE_LOG_POS`.

Создание каналов

В MySQL 8.4 каждый источник настраивается через `CHANGE REPLICATION SOURCE TO ... FOR CHANNEL`.

Пример для немецкого источника:

```
CHANGE REPLICATION SOURCE TO
  SOURCE_HOST='10.68.68.186',
  SOURCE_PORT=3306,
  SOURCE_USER='repl',
  SOURCE_PASSWORD='Repl84Geo2026x',
  SOURCE_LOG_FILE='mysql-bin.000001',
  SOURCE_LOG_POS=158,
  SOURCE_AUTO_POSITION=0,
  GET_SOURCE_PUBLIC_KEY=1
FOR CHANNEL 'production_de';
```

Тот же принцип для:

- `production_es`
- `production_it`

Фильтры по каналам

Сила multi-source заключается в фильтрации по каждому каналу.

Если все три источника имеют базу `PRODUCTION`, её можно перенаправить на стороне реплики:

```
CHANGE REPLICATION FILTER
  REPLICATE_REWRITE_DB=((PRODUCTION,PRODUCTION_DE))
FOR CHANNEL 'production_de';

CHANGE REPLICATION FILTER
  REPLICATE_REWRITE_DB=((PRODUCTION,PRODUCTION_ES))
FOR CHANNEL 'production_es';

CHANGE REPLICATION FILTER
  REPLICATE_REWRITE_DB=((PRODUCTION,PRODUCTION_IT))
FOR CHANNEL 'production_it';
```

Важный момент:

- канал должен существовать до применения `CHANGE REPLICATION FILTER ... FOR CHANNEL`
- если канал ещё не существует, MySQL вернёт ошибку

Запуск каналов

```
START REPLICA FOR CHANNEL 'production_de';
START REPLICA FOR CHANNEL 'production_es';
START REPLICA FOR CHANNEL 'production_it';
```

Затем, если реплика должна оставаться пассивной:

```
SET GLOBAL read_only = ON;
SET GLOBAL super_read_only = ON;
```

Проверка

Проверка по каналам:

```
SHOW REPLICA STATUS FOR CHANNEL 'production_de'\G
SHOW REPLICA STATUS FOR CHANNEL 'production_es'\G
```

```
SHOW REPLICA STATUS FOR CHANNEL 'production_it'\G
```

Ожидаемые показатели:

- `Replica_IO_Running: Yes`
- `Replica_SQL_Running: Yes`
- `Seconds_Behind_Source: 0` или близко к `0`
- `Replicate_Rewrite_DB` корректно определён

Функциональная проверка:

```
SELECT * FROM PRODUCTION_DE.germany_feed;
SELECT * FROM PRODUCTION_ES.spain_feed;
SELECT * FROM PRODUCTION_IT.italy_feed;
```

Типичные операции

Остановить один канал:

```
STOP REPLICA FOR CHANNEL 'production_es';
```

Перезапустить один канал:

```
START REPLICA FOR CHANNEL 'production_es';
```

Сбросить канал:

```
RESET REPLICA ALL FOR CHANNEL 'production_es';
```

Удалить фильтр перенаправления на канале:

```
CHANGE REPLICATION FILTER REPLICATE_REWRITE_DB=() FOR CHANNEL 'production_es';
```

Чего следует избегать

- направлять несколько источников в одну целевую базу без разделения
- предполагать, что MySQL будет разрешать конфликты ключей или порядка выполнения

- использовать разные версии MySQL без строгого контроля совместимости
- забывать, что `SOURCE_PASSWORD` в `CHANGE REPLICATION SOURCE TO` имеет ограничение по длине
- применять фильтры до создания канала

Техническое позиционирование

Multi-source MySQL 8.4 подходит для:

- консолидации нескольких стран
- централизованной отчётности
- сверки данных
- приёма нескольких независимых потоков

Он не подходит, сам по себе, для:

- настоящего multi-master с конкурентной записью
- архитектуры консенсуса
- автоматического разрешения конфликтов

Заключение

С MySQL 8.4 multi-source репликация чистая, зрелая и пригодная для агрегации нескольких серверов-источников в одну реплику.

Рекомендуемая схема проста:

1. чётко разделить роли источника и агрегатора
2. обеспечить уникальный `server-id` повсюду
3. загрузить начальный снимок
4. создать канал для каждого источника
5. применить фильтры перенаправления по каналам
6. проверить каждый канал независимо

Если данные источников имеют одинаковые имена баз, `REPLICATE_REWRITE_DB` по каналам — наиболее полезный механизм для сохранения читаемой и эксплуатируемой финальной

реплики.

Официальные источники

- MySQL 8.4 — Multi-Source Replication
- Configuring Multi-Source Replication
- Adding Binary Log Based Sources
- Channel Based Filters
- CHANGE REPLICATION SOURCE TO