



Администратор Linux

MySQL: BackUp + Репликация

• REC

Проверить, идет ли запись

Меня хорошо видно
&& слышно?



Ставим “+”, если все хорошо
“-”, если есть проблемы

Тема вебинара

MySQL: BackUp + Репликация



Федоров Иван Романович

Технический директор ГК "Инотех"

Опыт:

Более 10 лет в IT-сфере

Аспирант университета ИТМО по направлению "Информационная безопасность"

Многократный победитель различных конкурсов и хакатонов (команда IBI Solutions)

Эл. почта: ifedorov.devops@gmail.com

Правила вебинара



Активно
участвуем



Off-topic обсуждаем
в группе Telegram



Задаем вопрос
в чат или голосом



Вопросы вижу в чате,
могу ответить не сразу

Маршрут вебинара

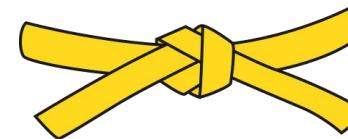
- Знакомство
- Введение и терминология
- Репликация
- Резервное копирование
- Практика
- Рефлексия



Цели вебинара

К концу занятия вы сможете

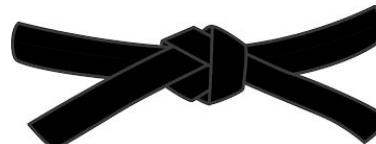
-
1. Настраивать репликацию master-slave MySQL



-
2. Настраивать резервное копирование

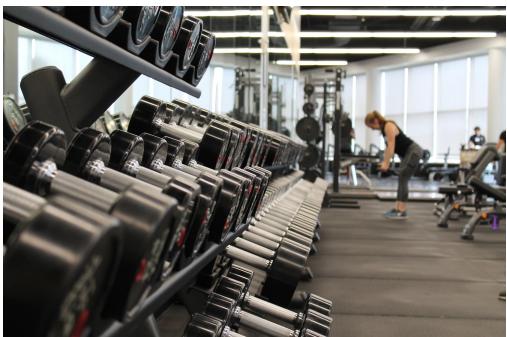


-
3. Создавать бэкапы с помощью xtrabackup



Смысл

Зачем вам это уметь



Введение и терминология

Задачи резервного копирования

- Создание одной или нескольких резервных копий на случай сбоя оборудования
- Защита от логической потери данных
- Восстановление данных на определённый момент в прошлом
- Снятие копии БД для тестирования и разработки



Терминология резервного копирования

- **RTO (Recovery Time Objective)** — определяет время, требуемое на восстановление из резервной копии. Это не то, что диктуется процессом РК, это то требование, которому процесс должен соответствовать. Например, “восстановление из РК должно занимать не более 1 часа”.
- **RPO (Recovery Point Objective)** — точка во времени (Point in Time), на которую должны быть восстановлены данные. Например, “Данные должны быть восстановлены по состоянию не “далее”, чем 24 часа с момента сбоя”.
- **Уровень резервного копирования (Backup Level)** — 0-1-2, Full, Differential, Incremental. Различные стратегии выбора данных для копирования.
- **Глубина резервного копирования** — определяет как долго хранятся резервные копии.
- **Стратегия хранения** — определяет “детализация”, с которой можно восстановиться.



RTO (Recovery Time Objective)



RPO (Recovery Point Objective)



Уровни резервного копирования

- **Full** — полное резервное копирование. Для восстановления требуется только эта резервная копия.
- **Differential** — разностное резервное копирование. Копируется только то, что изменилось с последнего резервного копирования. Для восстановления требуется последняя полная и последняя дифференциальная копии.
- **Incremental** — инкрементальное резервное копирование. Копируется только то, что изменилось с последнего прохода резервного копирования. Для восстановления требуется: последняя полная; последняя дифференциальная (если есть); ВСЕ инкрементальные копии с момента последней полной/дифференциальной копии.

TYPES OF BACKUP: FULL, DIFFERENTIAL, AND INCREMENTAL

Full Backups: Entire data set, regardless of any previous backups or circumstances.



Differential Backups: Additions and alterations since the most recent full backup.



Incremental Backups: Additions and alterations since the most recent incremental backup.



Initial Full Backup • 1st Backup 2nd Backup 3rd Backup 4th Backup 5th Backup
Data subject to backup



Cold vs Hot

- Cold («Холодное») – остановка базы данных и копирование всех файлов.
- Hot («Горячее») – копирование файлов базы данных без остановки базы.

Вопросы для планирования РК

- Репликация/Дублирование
- От каких сбоев мы хотим защититься? (Зачем?)
- Что надо копировать?
- Как быстро надо восстанавливать? (RTO)
- На сколько “близко” точка восстановления? (RPO)
- Где все это хранить?
- Сколько это стоит?

Сопутствующие действия

- Проверка целостности копий/проверка хранилища
- Мониторинг:
 - Хранилища
 - Агентов
 - ПО
 - Процесса
- Проверка восстановляемости.

Все ли понятно?



MySQL Конфигурация

Просмотр системных переменных

В консоли mysql

```
# Просмотр параметра 'max_connections' (вариант 1)
mysql> SHOW GLOBAL VARIABLES LIKE 'max_connections';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| max_connections | 151   |
+-----+-----+
1 row in set (0.00 sec)

# Просмотр параметра 'max_connections' (вариант 2)
mysql> select @@global.max_connections;
+-----+
| @@global.max_connections |
+-----+
|          151 |
+-----+
1 row in set (0.00 sec)
```

Изменение системных переменных

В консоли mysql

```
# Изменение параметра 'max_connections' (вариант 1)
mysql> SET GLOBAL max_connections=100\g
```

```
# Изменение параметра 'max_connections' (вариант 2)
mysql> SET @@global.max_connections=100\g
```

Изменение системных переменных

В конфигурационном файле

```
$ vim /etc/my.cnf.d/mysql-server.cnf
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
log-error=/var/log/mysql/mysqld.log
pid-file=/run/mysqld/mysqld.pid
```

```
$ systemctl restart mysqld
```

MySQL: репликация

Задачи репликации

- **Повышение отказоустойчивости**

Репликация позволяет избавиться от единственной точки отказа, которой является одиночный сервер БД.
В случае аварии на основном сервере, есть возможность быстро переключить нагрузку на резервный.

- **Повышение производительности чтения данных**

С помощью репликации возможно поддерживать несколько копий сервера, и распределять между ними нагрузку.

- **Распределение нагрузки**

В случае, если БД обслуживает запросы разных типов (быстрые и легкие, медленные и тяжелые), может иметь смысл развести эти запросы по разным серверам, для увеличения эффективности работы каждого типа.

- **Распространение данных**

Минимизация задержек сети для удаленных офисов.

- **Тестирование новых конфигураций**

С помощью репликации есть возможность проведения тестирования новых версий сервера БД, изменения параметров конфигурации, и даже изменения типов хранилища данных.

- **Резервное копирование**

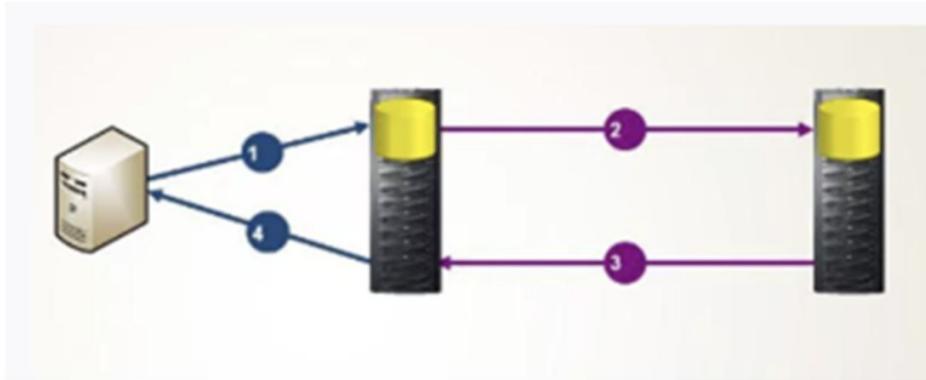
С помощью репликации есть возможность делать механизмы резервного копирования более гибкими и вносить меньше негативных эффектов в работающую систему.



Виды репликации

- Асинхронная репликация
- Полусинхронная репликация
- Синхронная репликация
- Групповая репликация

Синхронная репликация

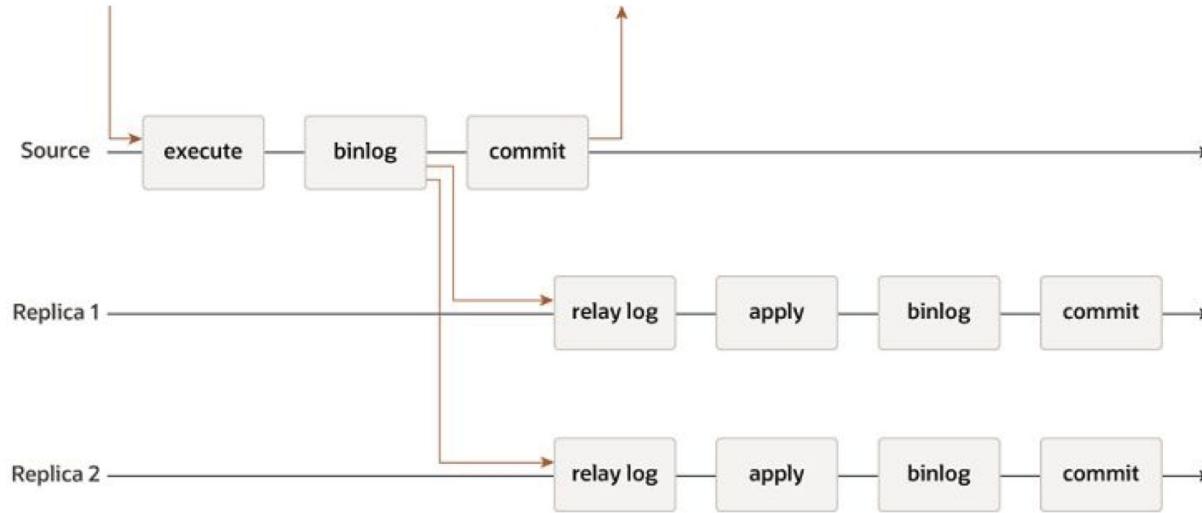


Такая конфигурация используется при работе NDB Cluster,
Galera Cluster, Percona XtraDB Cluster

<https://dev.mysql.com/doc/refman/8.0/en/mysql-cluster.html>



Асинхронная репликация



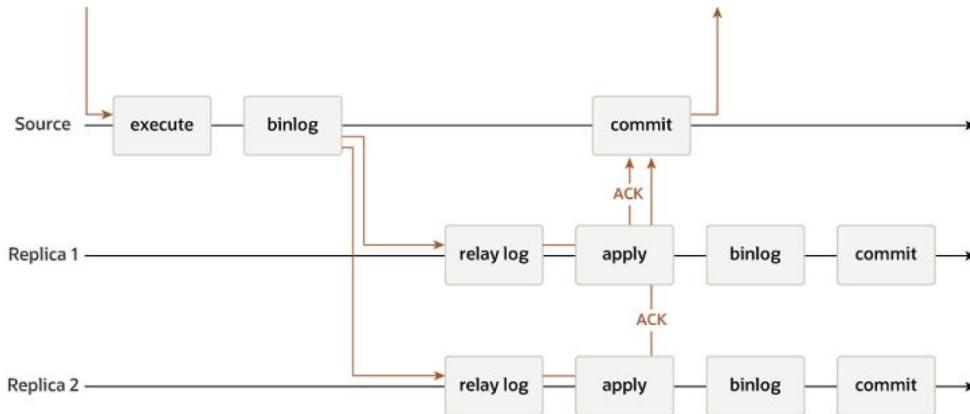
Плюсы:

- скорость работы мастера, не ждет слейва

Минусы:

- возможна потеря данных
- задержки на слейве

Полусинхронная репликация

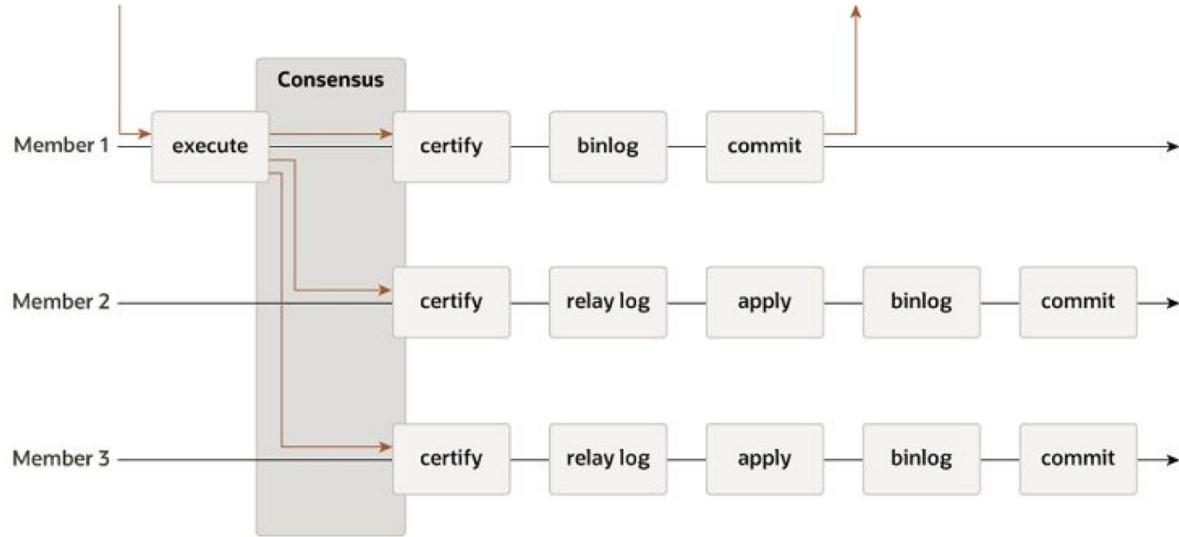


Key

ACK = Acknowledged

Commit транзакции будет подтвержден, только тогда, когда хотя бы одна из реплик подтвердит, что событие получено и зафиксировано/логировано

Групповая репликация



Такая конфигурация используется при работе InnoDB Cluster



Путь insert без репликации

```
INSERT INTO mytest VALUES (123, 'hello');
```

Приложение-писатель

- таблица на master mysqld
- приложение-читатель



Путь insert с репликацией

```
INSERT INTO mytest VALUES (123, 'hello');
```

Приложение-писатель

- таблица на master mysqld
- binary log на master
- relay log на slave
- таблица на slave mysqld
- приложение-читатель



Форматы бинарного лога

statement-based

- в лог пишутся SQL (update for ...)
- тригеры будут отрабатывать и на мастере и на слейвах
- опасность разных таймстампов

row-based

- в лог пишутся строки (каждую измененную строку..)
- получаются большие логи
- отключены тригеры

mixed

- для safe-стейментов пишутся SQL
- для unsafe - строки

show variables like '%binlog%';



Что записать в binary log?

Вариант 1:

```
UPDATE users SET x=123 WHERE id=456
```

Вариант 2:

```
UPDATE users SET bonus=bonus+100
```

Вариант 3:

```
UPDATE users SET disabled=1 WHERE last_login <  
UNIX_TIMESTAMP(NOW())-100*86400
```

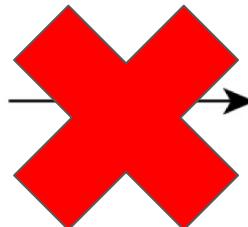
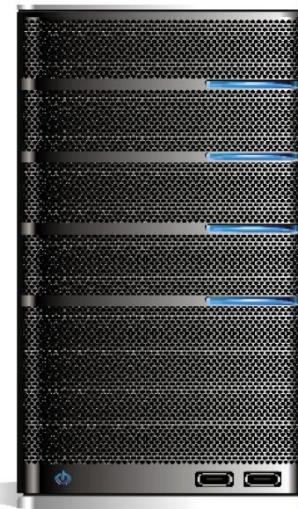


Репликация НЕ отменяет бэкапы

Master



Slave as backup storage



DROP TABLE ...

DROP TABLE ...

Настройка репликации

Master (в консоли MySQL)

```
# Проверка Server ID
mysql> select @@server_id;

# Создание пользователя и выдача прав на репликацию
mysql> create user repl@<IP SLAVE> identified by '****';
mysql> grant replication slave on *.* to repl@<IP SLAVE>;
mysql> flush privileges;

# Проверяем статус
mysql> show master status;
+-----+-----+-----+-----+
| File | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+-----+-----+-----+-----+
| binlog.000010 | 4402 | | |
+-----+-----+-----+-----+
1 row in set (0.00 sec)
```

Настройка репликации

Slave (в консоли MySQL)

```
# Проверка Server ID (должен отличаться)
mysql> select @@server_id;

# Создание пользователя и выдача прав на репликацию
mysql> change master to master host=<MASTER IP>, master user='repl', master password='***',
master_log_file='binlog.000010', master_log_pos=4402, get_master_public_key = 1;

mysql> start slave;

# Проверяем статус
mysql> mysql> show slave status\G
***** 1. row *****
Slave_IO_State: Waiting for source to send event
      Master_Host: 10.0.0.32
      Master_User: repl
      Master_Port: 3306
     Connect_Retry: 60
    Master_Log_File: binlog.000010
   Read_Master_Log_Pos: 4402
          Relay_Log_File: nginx-relay-bin.000007
            Relay_Log_Pos: 1341
   Relay_Master_Log_File: binlog.000010
      Slave_IO_Running: Yes
     Slave_SQL_Running: Yes
```

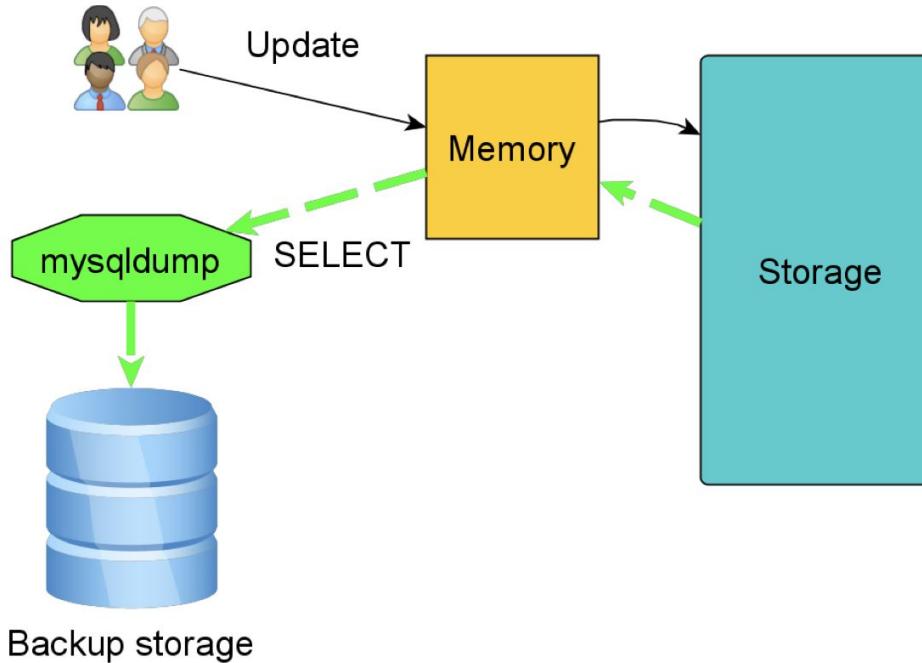
Все хорошо?
Есть ли вопросы?

MySQL: backup

Типы бэкапа

- Логический
- Физический

mysqldump



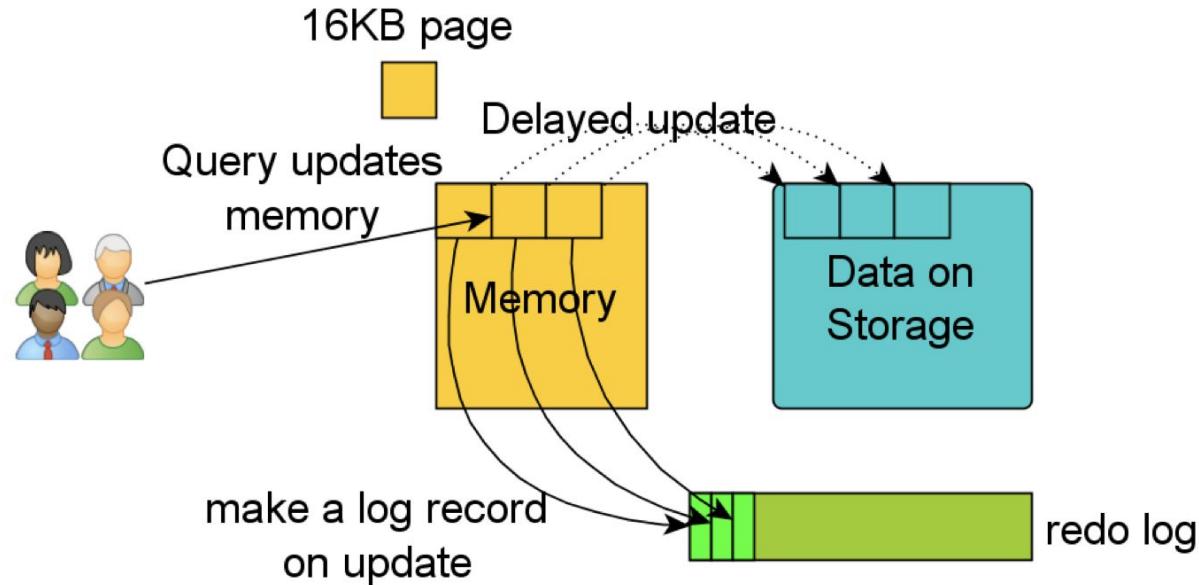
Создание бэкапа

В терминале

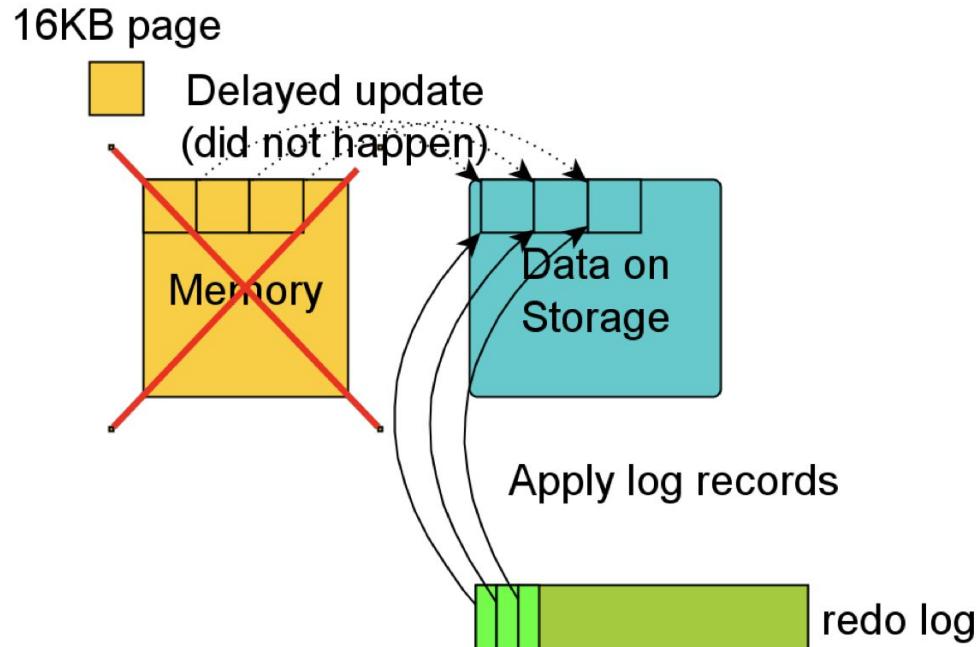
```
# Создание бэкапа
$ mysqldump -u root -p DATABASE > backup.sql
```

```
# Восстановление бэкапа
$ mysql -u root -p NEW_DATABASE < backup.sql
```

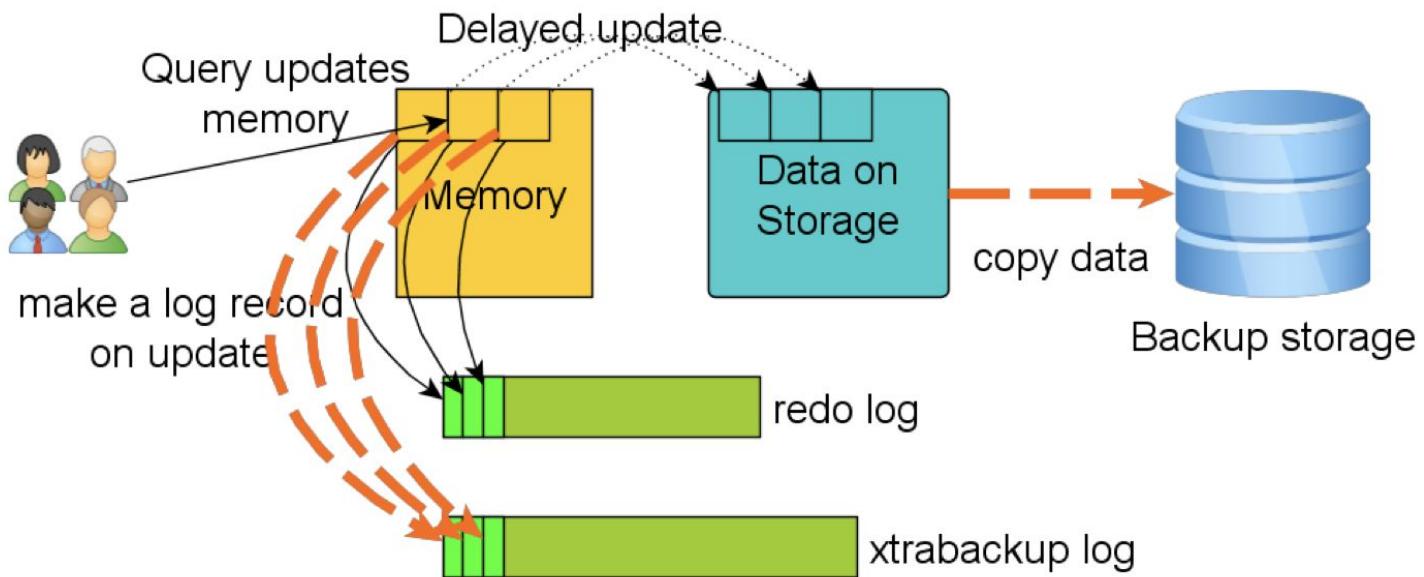
InnoDB процессы



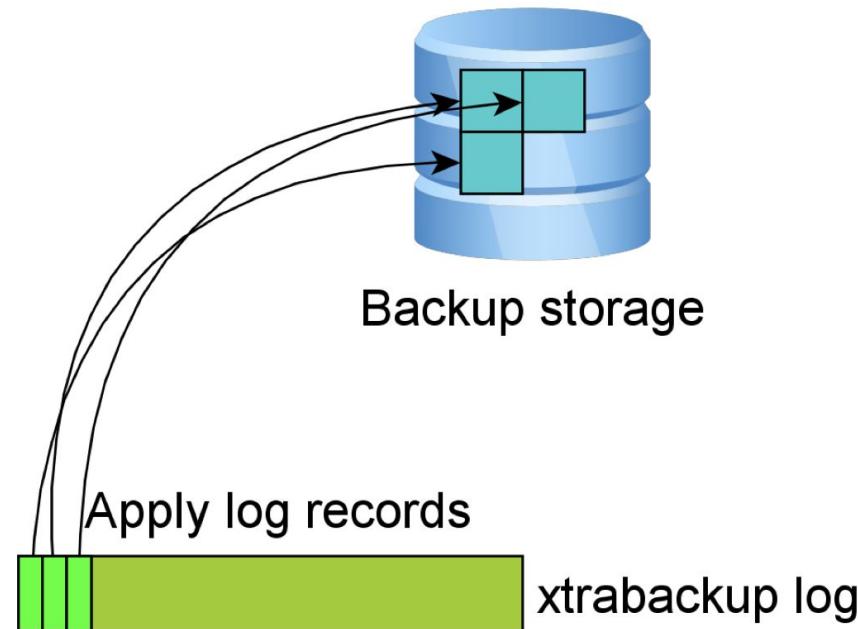
InnoDB восстановление



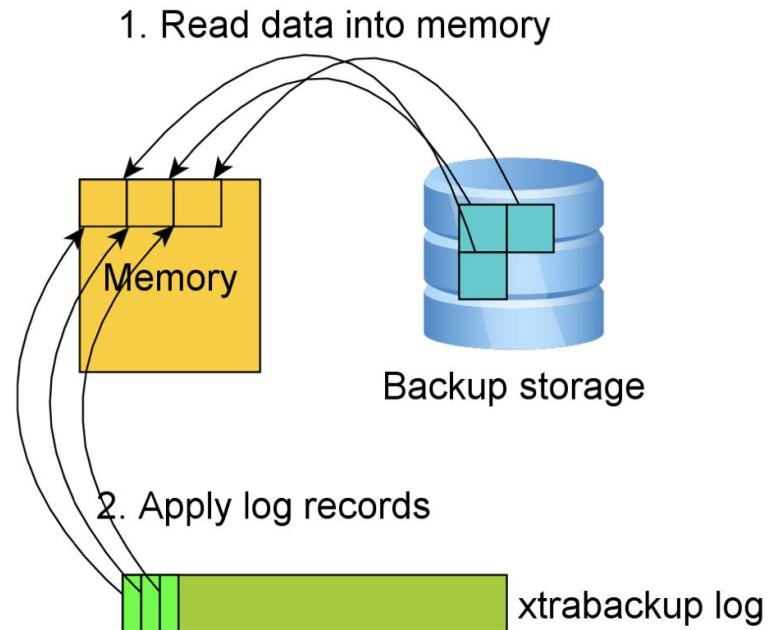
Идея бэкапа



Идея бэкапа



При восстановлении работаем с памятью



Установка xtrabackup

Centos 8

```
# Установка
$ yum install https://repo.percona.com/yum/percona-release-latest.noarch.rpm
$ percona-release enable-only tools release
$ yum install percona-xtrabackup-80

# Проверка
$ xtrabackup -v
2023-06-21T16:08:46.802215-00:00 0 [Note] [MY-011825] [Xtrabackup] recognized server
arguments: --datadir=/var/lib/mysql --server-id=2
xtrabackup version 8.0.33-27 based on MySQL server 8.0.33 Linux (x86_64) (revision id:
6743d8c7)
```

Создание бэкапа

В терминале

```
# Создание бэкапа
$ xtrabackup --backup --target-dir=/backup

# Подготовка бэкапа для развертывания
$ xtrabackup --prepare --target-dir=/backup

# Восстановление бэкапа (datadir должна быть пустой)
$ systemctl stop mysqld
$ xtrabackup --copy-back --target-dir=/backup
$ chown -R mysql:mysql /var/lib/mysql
$ systemctl start mysqld
```

Вопросы?



Ставим “+”,
если вопросы есть



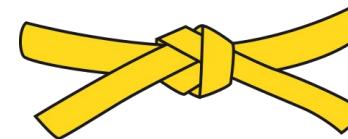
Ставим “-”,
если вопросов нет

Рефлексия

Цели вебинара

К концу занятия вы сможете

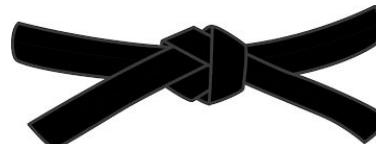
-
1. Настраивать репликацию master-slave MySQL



-
2. Настраивать резервное копирование



-
3. Создавать бэкапы с помощью xtrabackup



Рефлексия



С какими впечатлениями уходите с вебинара?



Как будете применять на практике то,
что узнали на вебинаре?

**Заполните, пожалуйста,
опрос о занятии
по ссылке в чате**