Лекция
09.04.2019

DBA2 Theory. p6.

Параллельная репликация MariaDB (+GTID)

Ильшат Каразбаев
руководитель группы DBA
АО ТК Центр

Немного обо мне

Вместе со своей командой администрирую:

СУБД MySQL, Mariadb, galeracluster, Postgres

Главный по базам в ТК Центр

Повестка дня:

1. Вводная
2. Слайды с прошлых лекций по механике репликации
3. Однопоточная репликация
4. Параллельная репликация
5. Ordered commit
6. Настройки репликации
7. Типы параллельности
8. Aggressive/Optimistic/Conservative/Minimal modes
9. Out-of-order replication
10. Group commit
11. Литература

Вводная

Мастер выполняет транзакции параллельно

В бинлог события пишутся последовательно

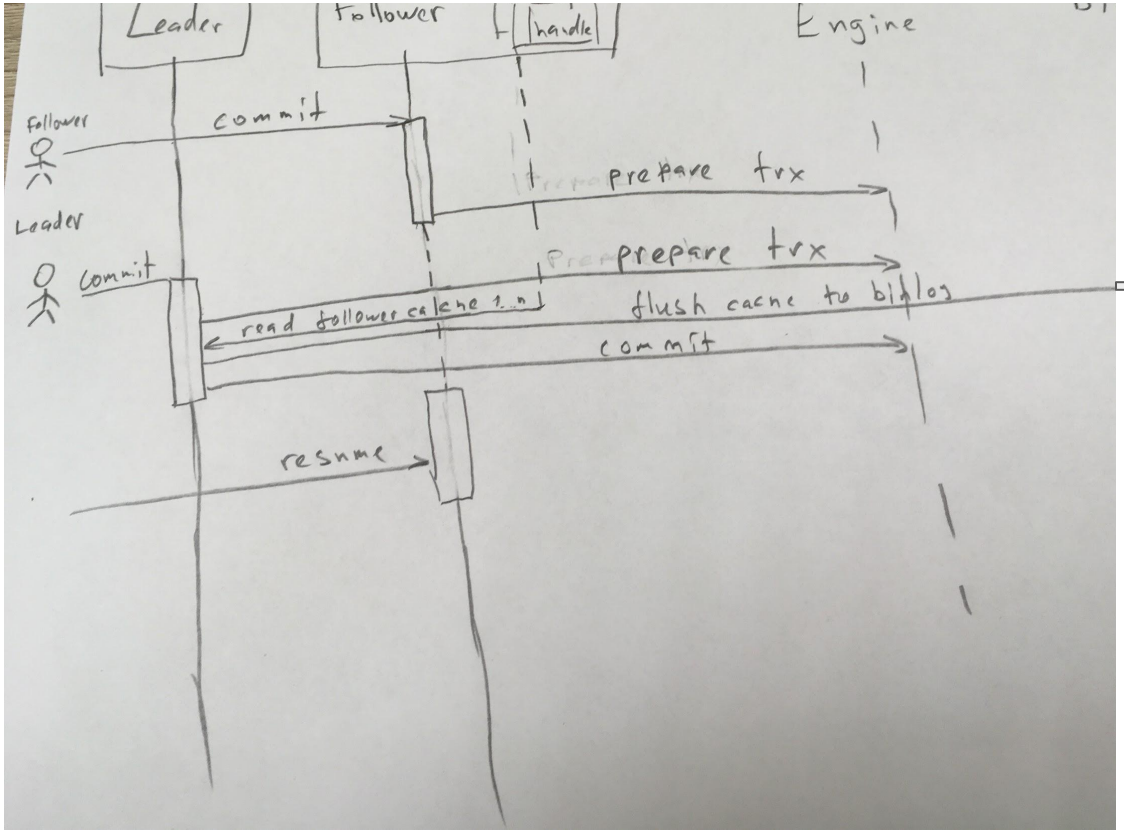
Реплика читает и применяет транзакции последовательно

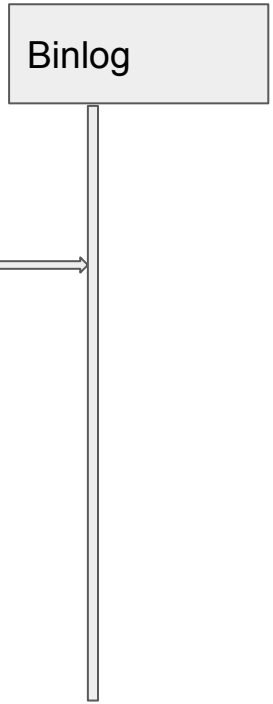
CPU используется неэффективно

Реплика отстает (лагает)

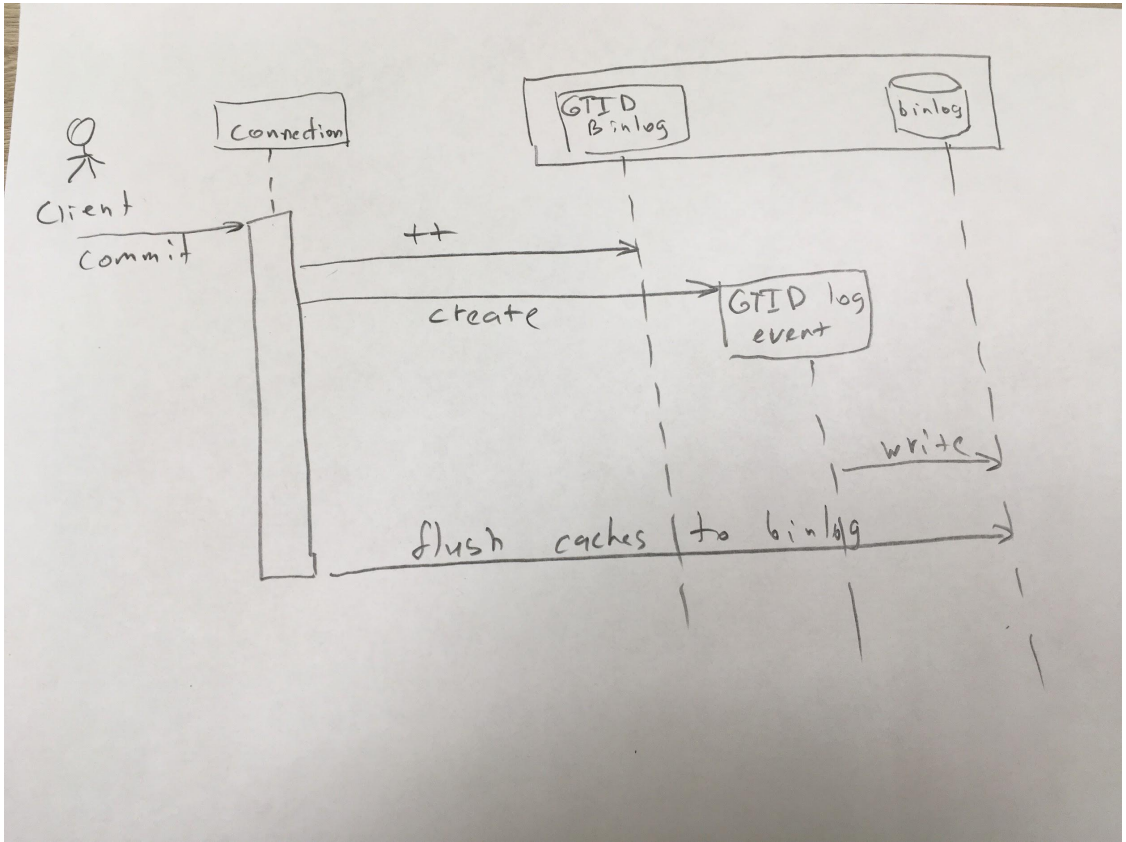
А что, если репликацию сделать параллельной?

Binlog group commit

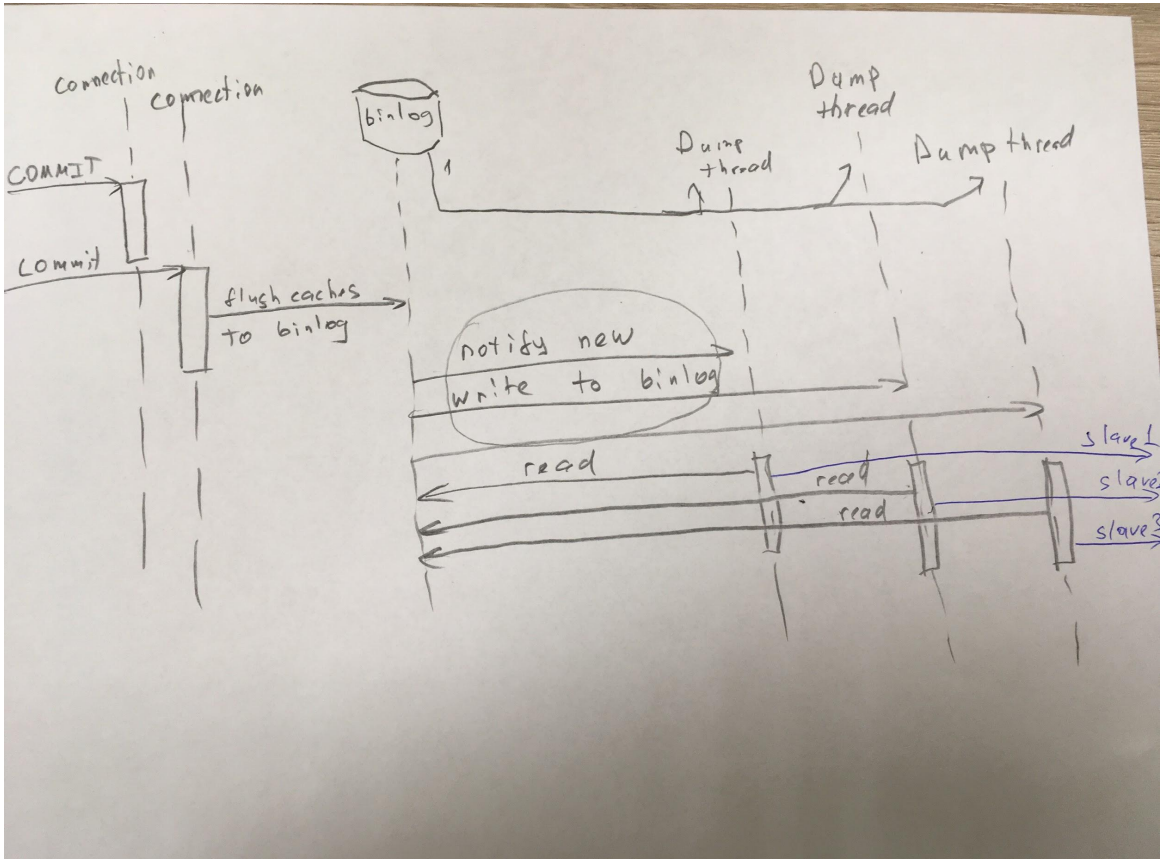




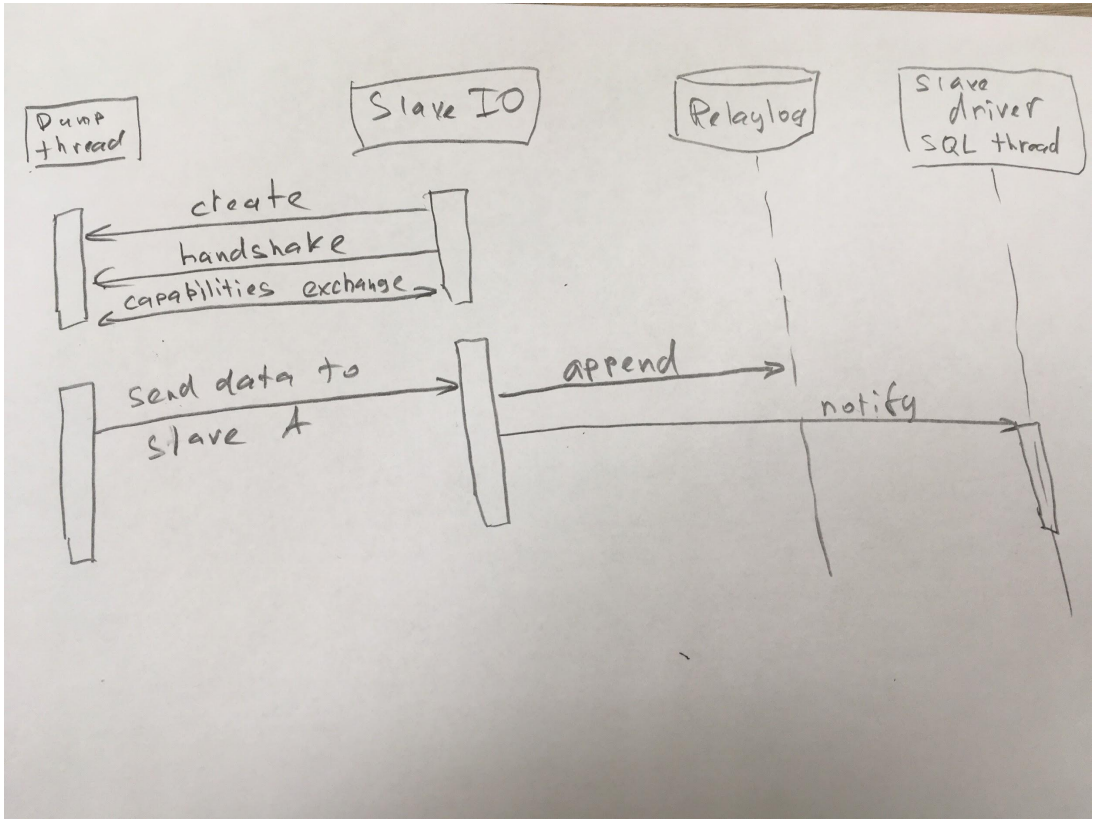
Запись GTID


Client
commit
connection
GTID
Binlog
binlog
++
create
GTID log
event
write
flush caches to binlog

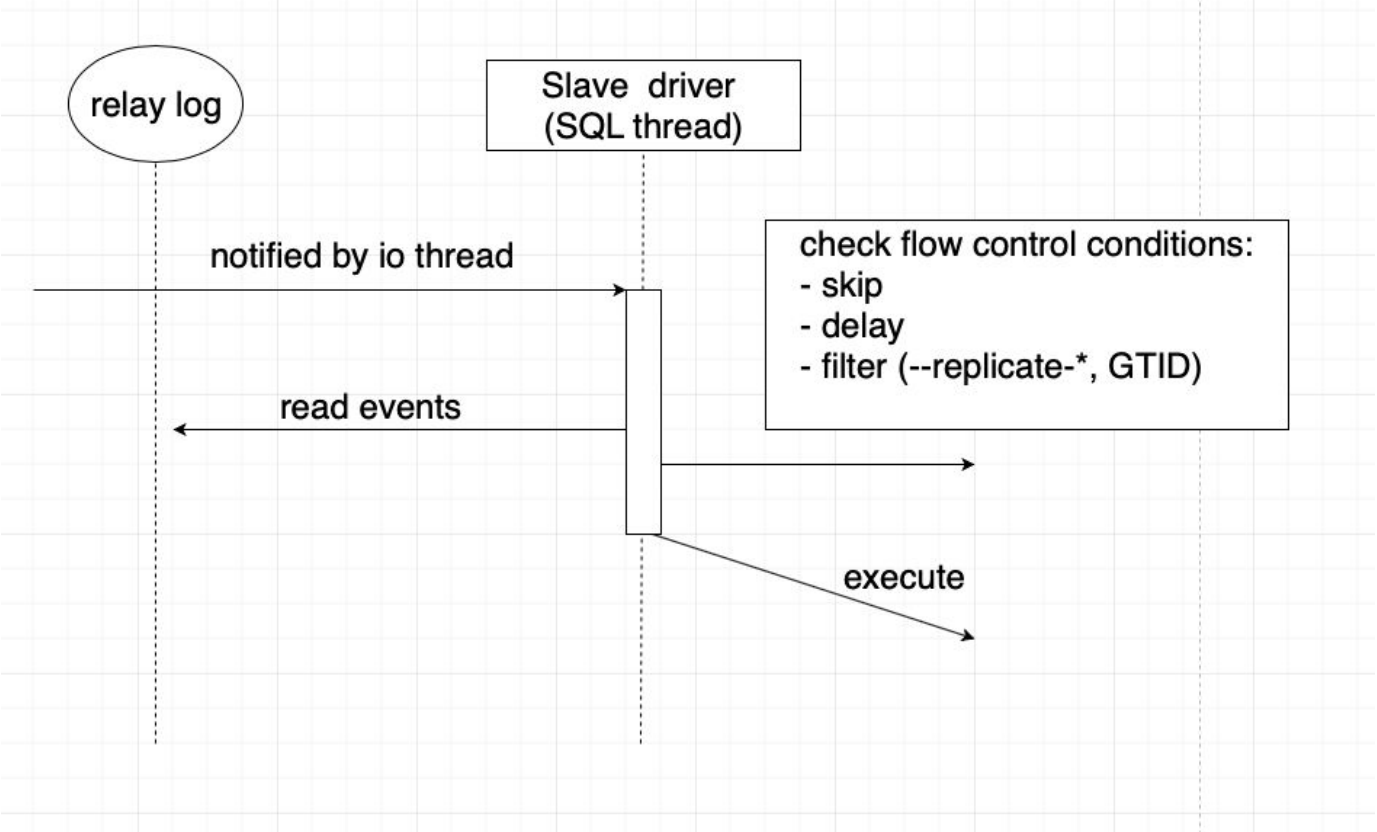
Доставка событий бинлога. Dump thread



Прием событий бинлога. IO thread



Однопоточное применение изменений



Параллельная репликация

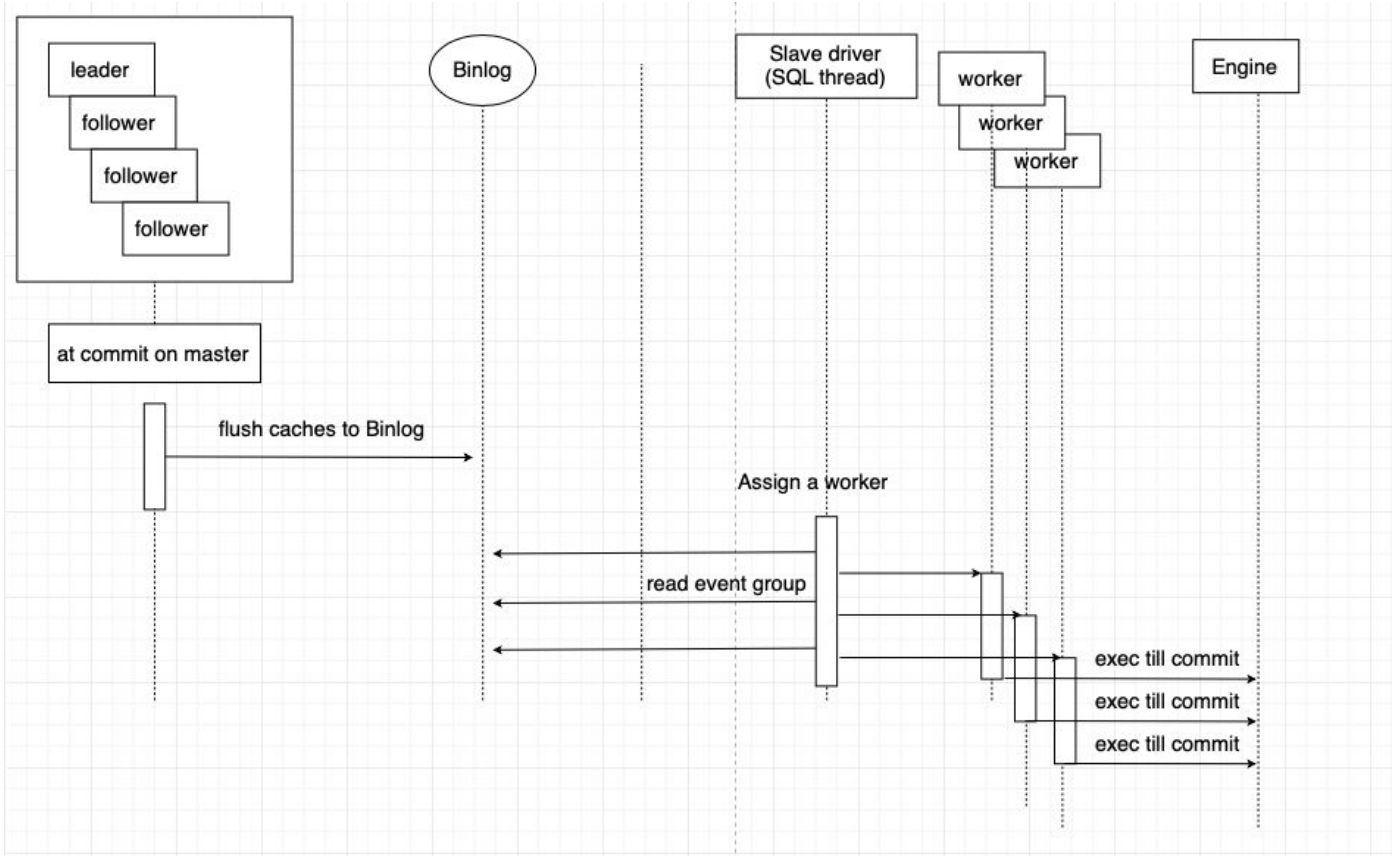
Репликация происходит в три этапа:

1. События репликации считываются с мастера с помощью IO thread и скапливаются в relay log
2. SQL thread считывает события репликации из relay log последовательно
3. Каждой событие репликации применяется на реплике, таким образом все изменения с мастера появляются в реплике

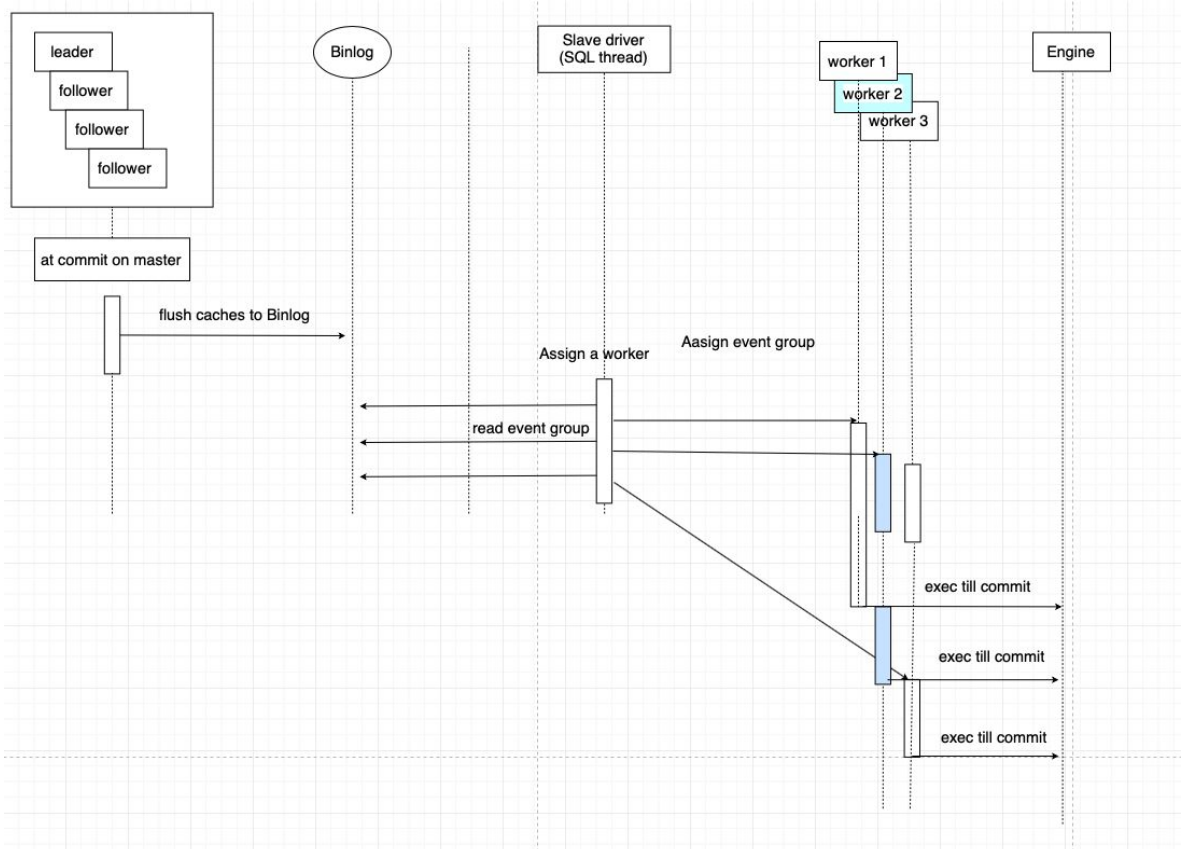
При последовательной репликации третий шаг происходит также в SQL thread

При параллельной репликации worker threads применяют изменения параллельно

Параллельная репликация



ordered commit



Включить репликацию

slave_parallel_threads=X

slave_paralllel_mode=VALUE

Для multisource репликации:

@@connection_name.slave_parallel_mode=VALUE

slave_parallel_threads - количество тредов репликации для всех источников одновременно

Типы параллельности

In-order: транзакции исполняются параллельно, но комммиты упорядочены точно так же, как и на мастере

Out-of-order: транзакции исполняются параллельно, очередность коммитов различна с мастером. Приложение должно быть спроектировано таким образом, чтобы минимизировать конфликты транзакций, которые очередность коммитов которых различается в мастере и реплике. Этот тип параллельности возможен только при использовании GTID (используя различные GTID domain id для транзакций, которые могут работать параллельно)

In-order replication. Aggressive mode

Все транзакции, которые могут быть выполнены параллельно, выполняются параллельно

При конфликте более ранняя транзакция имеет приоритет

Более поздняя транзакция откатывается и перезапускается

Коммит происходит с учетом очередности

In-order replication. Optimistic mode

slave_parallel_mode=optimistic

- DML (insert/update/delete) могут работать параллельно со степенью до slave_parallel_threads
- Могут возникать конфликты
- Конфликты разрешаются через retry транзакции
- Можно потенциально опасные транзакции маркировать на мастере @@skip_parallel_replication (не работает для aggressive режима)
- non-transactional DML, DDL не могут быть откачены, поэтому параллельно не применяются

In-order replication. Conservative mode

slave_parallel_mode=conservative

- Режим по умолчанию
- для 10.0 - единственно возможный
- Используется group commit для определения безопасности параллельного применения (одинаковый cid для нескольких транзакций)
- Можно повысить эффективность параллельного применения через настройки на мастере: binlog_commit_wait_count, binlog_commit_wait_usec. Мастер будет работать медленнее, реплика быстрее
- Хорошо включить log_slave_updates=1

In-order replication. Minimal mode

slave_parallel_mode=minimal

Параллельно работает только стадия коммита

Остальная часть репликации транзакций идет последовательно

Не работает out-of-order репликация

Out-of-order replication

Работает только если включить репликацию по GTID

Суть в том, что транзакции, запущенные в разных gtid domain id, работают полностью независимо с точки зрения репликации и будут полностью параллельны независимо от стадии транзакции.

Хорошо использовать при командах alter table. Пример:

SET SESSION gtid_domain_id=1

ALTER TABLE t ADD INDEX myidx(b)

SET SESSION gtid_domain_id=0

Статусы в show processlist

- "Waiting for work from main SQL threads". Нет текущих задач для worker
- "Waiting for prior transaction to start commit before starting next transaction". worker ожидает применения группы транзакций на мастере
- "Waiting for prior transaction to commit". Транзакция выполнена и дожидается своей очереди коммита

slave_paralllel_max_queued

slave_parallel_nax_queued - количество памяти на тред (умножить на slave_parallel_threads), выше которого сервер не будет использовать

Это полезно, чтобы реплика не выжрала всю ОЗУ и потом была убита OOM при сильно отстающей реплике.

slave_parallel_domain_threads

Если используется out-of-order репликация или multi-source (что тоже out-of-order), все треды из slave_parallel_threads могут быть использованы одним доменом

Для того, чтобы оставить треды и на другой домен, можно ограничить максимальное количество тредов на домен

slave_domain_parallel_threads должен быть меньше slace_parallel_threads

Group commit

Начиная с MariaDB 10.3 включена поддержкаа group commit

При настройках innodb_flush_log_at_trx_commit=1, sync_binlog=1 каждая транзакция записывается на диск после исполнения

Можно записывать на диск группу транзакций, для этого можно отложить коммит отдельных транзакций, для того, чтобы они складывались в группы

- binlog_commit_wait_count
- binlog_commit_wait_usec

Show global status where Variable_name in(‘Binlog_commits’, ‘Binlog_group_commits’);

Group commit

Длинные транзакции на мастере могут плохо сказаться на параллельном исполнении

```
On the master:
---------------- Time --------------->
T1: B-----------------------------C
T2:                              B--C
T3:                              B--C
On the slaves:
T1: B-----------------------------C
T2: B-- . . . . . . . . . . . . . C
T3: B-- . . . . . . . . . . . . . C
```

Вопросы?

Все анонсы здесь:

- telegram чат: t.me/mariadb_course

Материалы курса:

- видео: https://www.youtube.com/channel/UCGsmu6YDpcR_kWcXzeQkWrA
- слайды лекций и примеры: git@github.com:barazbay/mariadb_course.git

Меня можно найти:

- vk, instagram: barazbay
- twitter: karazbay

Литература

1. https://mariadb.com/kb/en/library/gtid/
2. https://mariadb.com/resources/blog/enabling-gtids-for-server-replication-in-mariadb-server-10-2/
3. https://mariadb.com/kb/en/library/parallel-replication/
4. https://vimeo.com/258533271
5. https://mariadb.com/kb/en/library/using-mariadb-gtids-with-mariadb-galera-cluster/
6. https://mariadb.com/kb/en/library/multi-source-replication/
7. https://www.osp.ru/netcat_files/userfiles/Hadoop_TBD_2_2016/Petrunya_tbd_2.pdf

Список литературы

8. https://www.youtube.com/watch?v=v79m2PVGpcY
9. https://mariadb.com/kb/en/library/group-commit-for-the-binary-log/
10. http://kristiannielsen.livejournal.com/18435.html
11. https://mariadb.com/resources/blog/better-parallel-replication-for-mariadb-and-mysql/
12. https://www.percona.com/live/17/sites/default/files/slides/2017-04-26_plsc_mysql-mariadb_parallel_replication-inventory_use-case_and_limitations_v1.0.pdf
13. https://mariadb.com/resources/blog/better-parallel-replication-for-mariadb-and-mysql/
14. https://mariadb.com/resources/blog/evaluating-mariadb-mysql-parallel-replication-part-2-slave-group-commit/