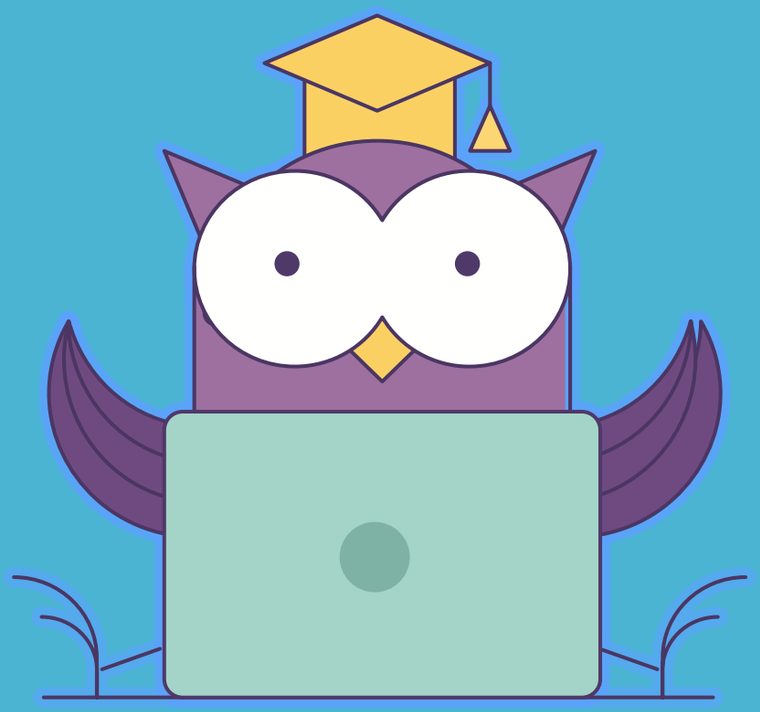
ОНЛАЙН-ОБРАЗОВАНИЕ

Репликация (часть 1)

Юрочко Юрий

Как меня слышно и видно?



> Напишите в чат

+ если все хорошо

– если есть проблемы со звуком или с видео

Преподаватель

Юрочко Юрий

- окончил МГТУ им. Н.Э. Баумана в 2016 (ИУ-7)
- 5+ лет разработки на C++/Go
- руководитель команды Go

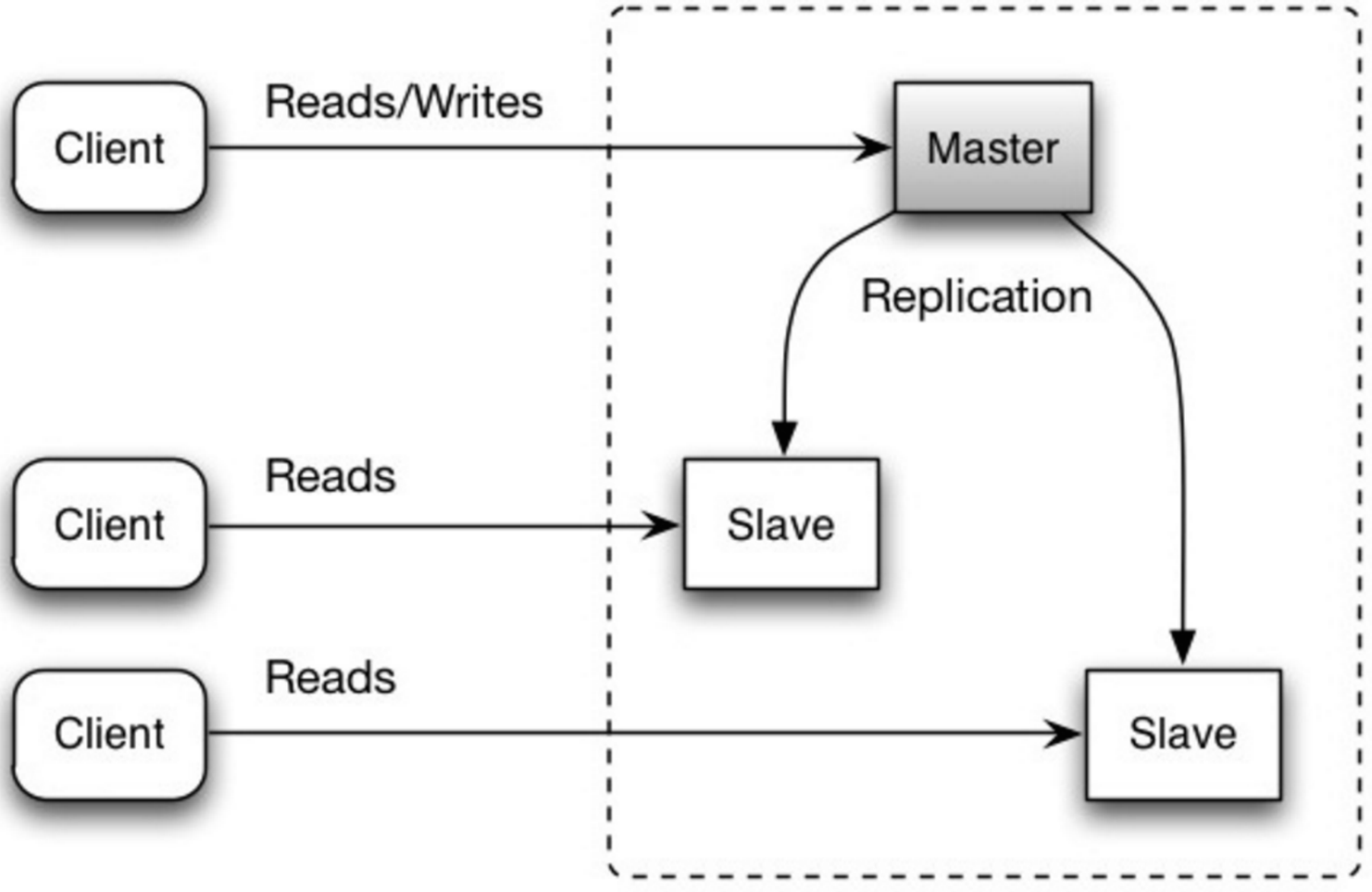
Цель занятия

- Понять, что такое репликация и зачем она нужна
- Обсудить виды репликации и связанные механизмы
- Понять разницу между подходами к репликации
- Сравнить особенности репликации в MySQL и Postgres
- Познакомиться с групповой репликацию в MySQL

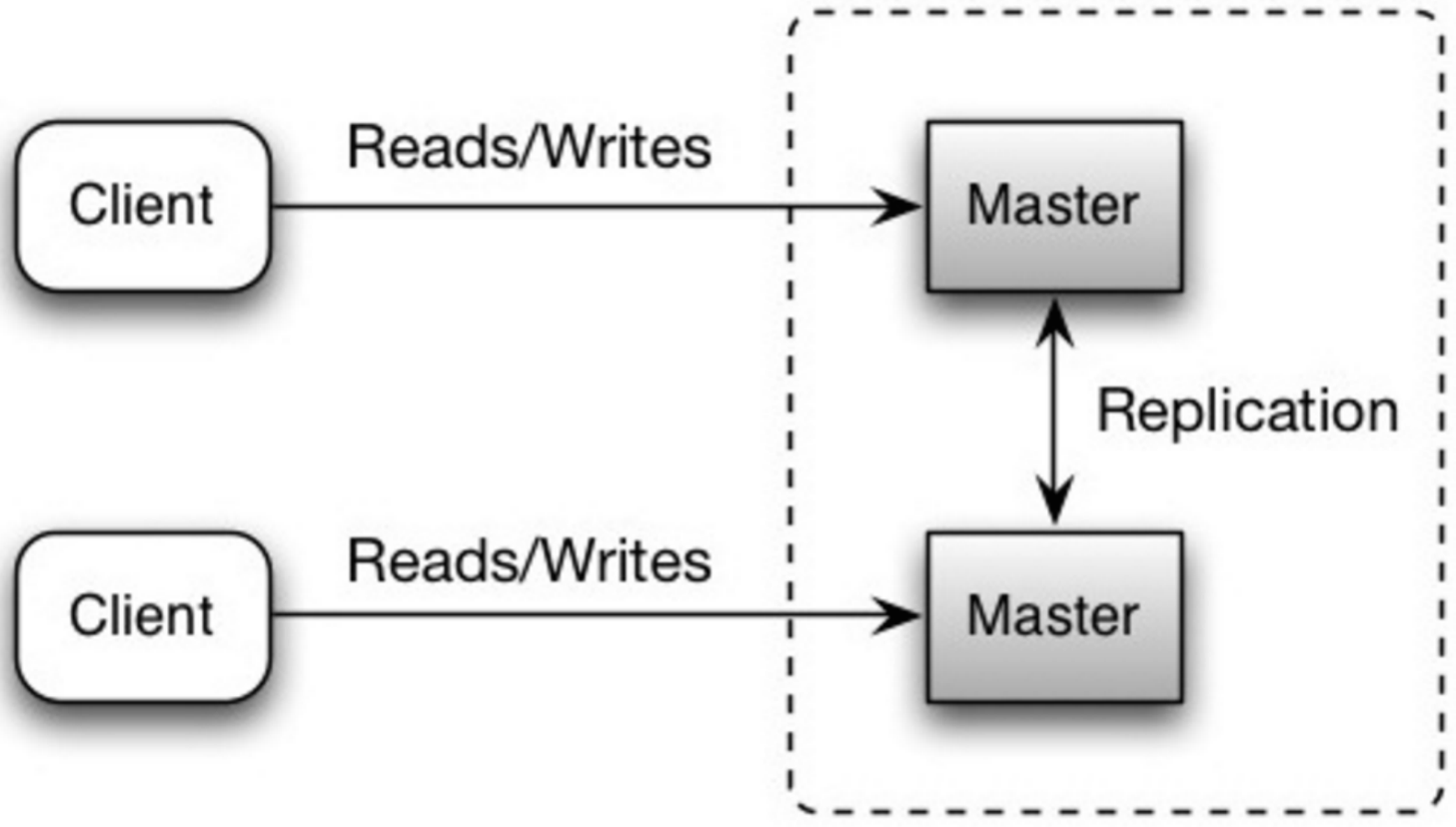
- копирование (реплицирование) данных
- один из способ масштабирования
- можно использовать много серверов для обработки запросов
- не backup
- не магия
- вам придется заплатить

Масштабирование?

- не помогает ускорить запись
- помогает ускорить чтение
- помогает при падениях


Client
Reads/Writes
Master
Replication
Client
Reads
Slave
Client
Reads
Slave

- один источник данных
- наиболее распространенный подход
- относительно просто и понятно
- отключить/включить реплику
- мастер иногда крашится...


Client
Reads/Writes
Master
Replication
Client
Reads/Writes
Master

- нет единой точки отказа
- постоянное время работы
- легкий failover
- нет консистентности (все обязательно поломается)
- group replication?

Статус мастера

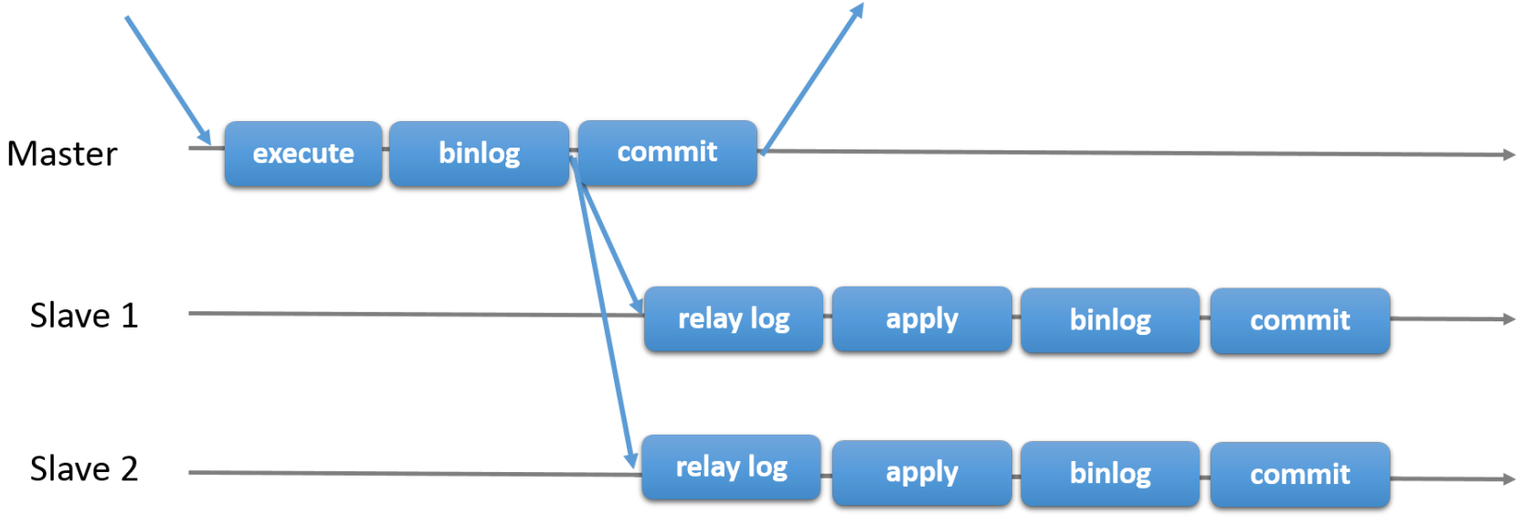
```
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000001 |      107 | example      |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)
```

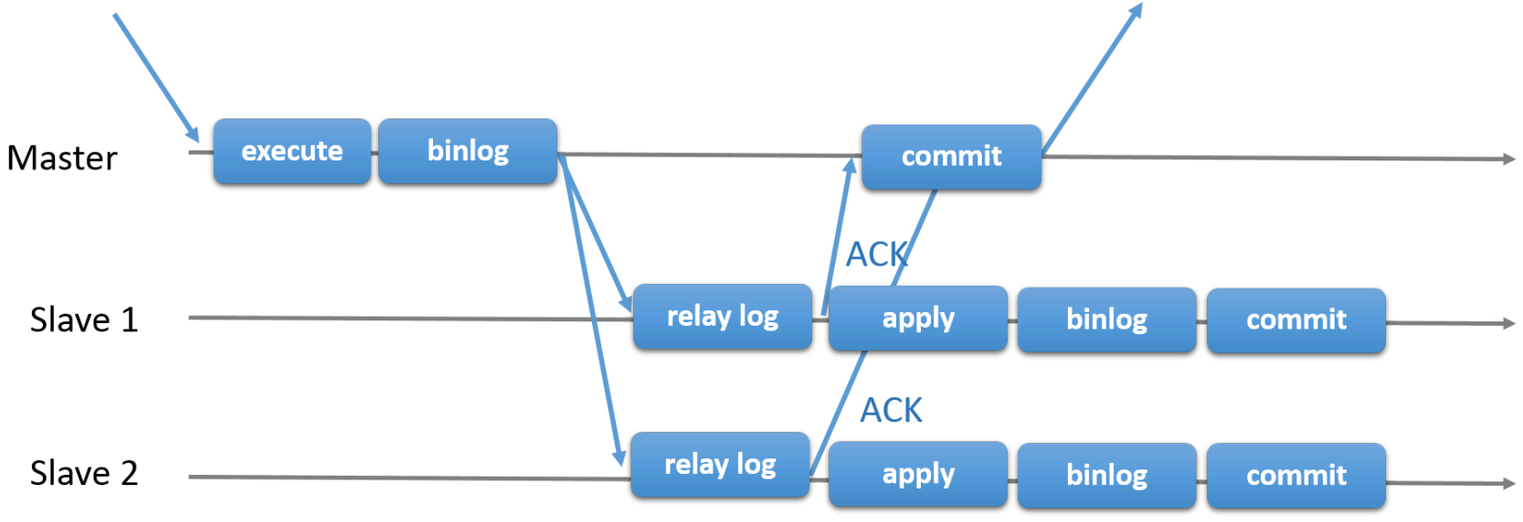
Статус слейва

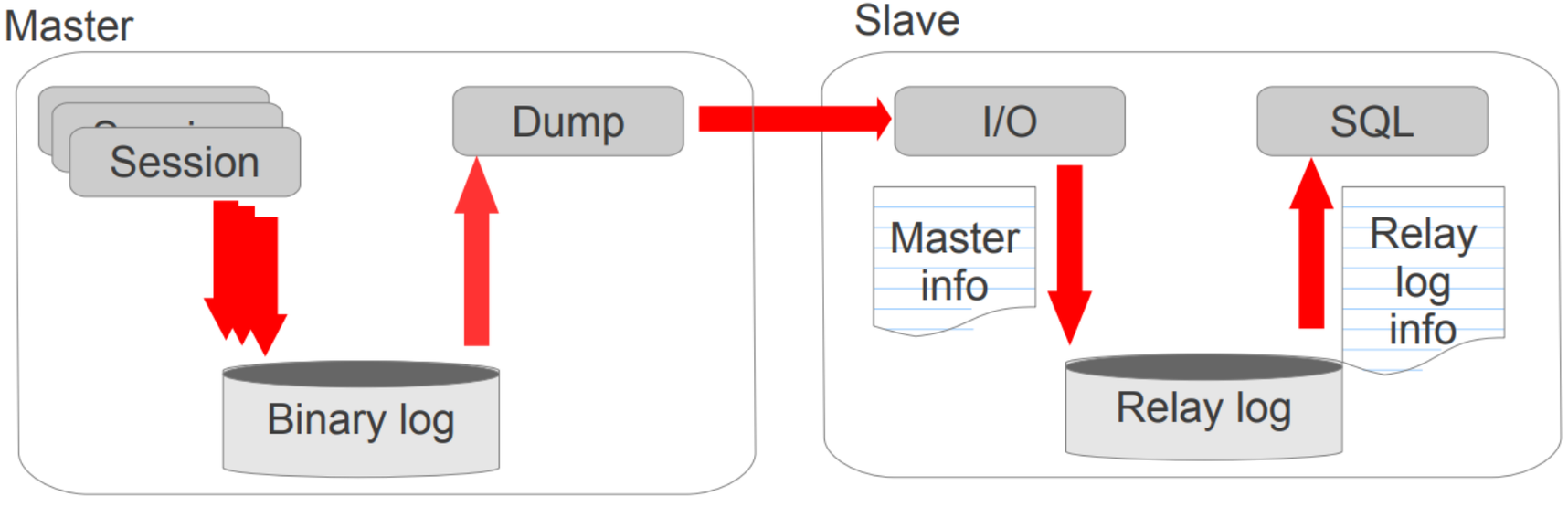
```
mysql> SHOW SLAVE STATUS\G

            Slave_IO_State: Waiting for master to send event
               Master_Host: localhost
               Master_User: root
               Master_Port: 3306
             Connect_Retry: 3
           Master_Log_File: gbichot-bin.005
       Read_Master_Log_Pos: 79
            Relay_Log_File: gbichot-relay-bin.005
             Relay_Log_Pos: 548
     Relay_Master_Log_File: gbichot-bin.005
          Slave_IO_Running: Yes
         Slave_SQL_Running: Yes
           Replicate_Do_DB:
       Replicate_Ignore_DB:
        Replicate_Do_Table:
    Replicate_Ignore_Table:
   Replicate_Wild_Do_Table:
Replicate_Wild_Ignore_Table:
               Last_Errno: 0
               Last_Error:
             Skip_Counter: 0
       Exec_Master_Log_Pos: 79
           Relay_Log_Space: 552
           Until_Condition: None
            Until_Log_File:
             Until_Log_Pos: 0
        Master_SSL_Allowed: No
        Master_SSL_CA_File:
        Master_SSL_CA_Path:
           Master_SSL_Cert:
         Master_SSL_Cipher:
            Master_SSL_Key:
     Seconds_Behind_Master: 8
```

- sync - закомитили локально, закомитили удаленно (Postgres)
- async - закомитили локально, все (MySQL, Postgres))
- semi-sync - закомитили локально, получили ack (MySQL)

- sync - сделанное изменения видные всем и везде
- async - никаких гарантий на другом конце
- semi-sync - доступно здесь, уже скопировано на другой конец


Master
execute
binlog
commit
Slave 1
relay log
apply
binlog
commit
Slave 2
relay log
apply
binlog
commit


Master
execute
binlog
commit
ACK
Slave 1
relay log
apply
binlog
commit
ACK
Slave 2
relay log
apply
binlog
commit


Master
Session
Dump
Binary log
Slave
I/O
SQL
Master info
Relay log info
Relay log

Master:

- обрабатывает запросы
- делает транзакции
- пишет binary logs

Slave:

- стягивает изменения с мастера, кладет в relay log
- читает данные из relay log и применяет изменения

Потоки:

- binlog dump thread (SHOW PROCESSLIST на мастере)
- slave I/O thread (SHOW SLAVE STATUS на слейве)
- slave SQL thread

https://dev.mysql.com/doc/refman/5.7/en/replication-implementation-details.html

- statement based
- row based
- mixed

https://dev.mysql.com/doc/refman/5.7/en/binary-log-setting.html

- передаются сами запросы
- гоняется небольшое количество данных
- все запросы в логе
- но...

- **UPDATE items SET enabled=1 WHERE time < UNIX_TIMESTAMP(NOW())-60** и все поломалось
- каждый запрос считается на каждой ноде

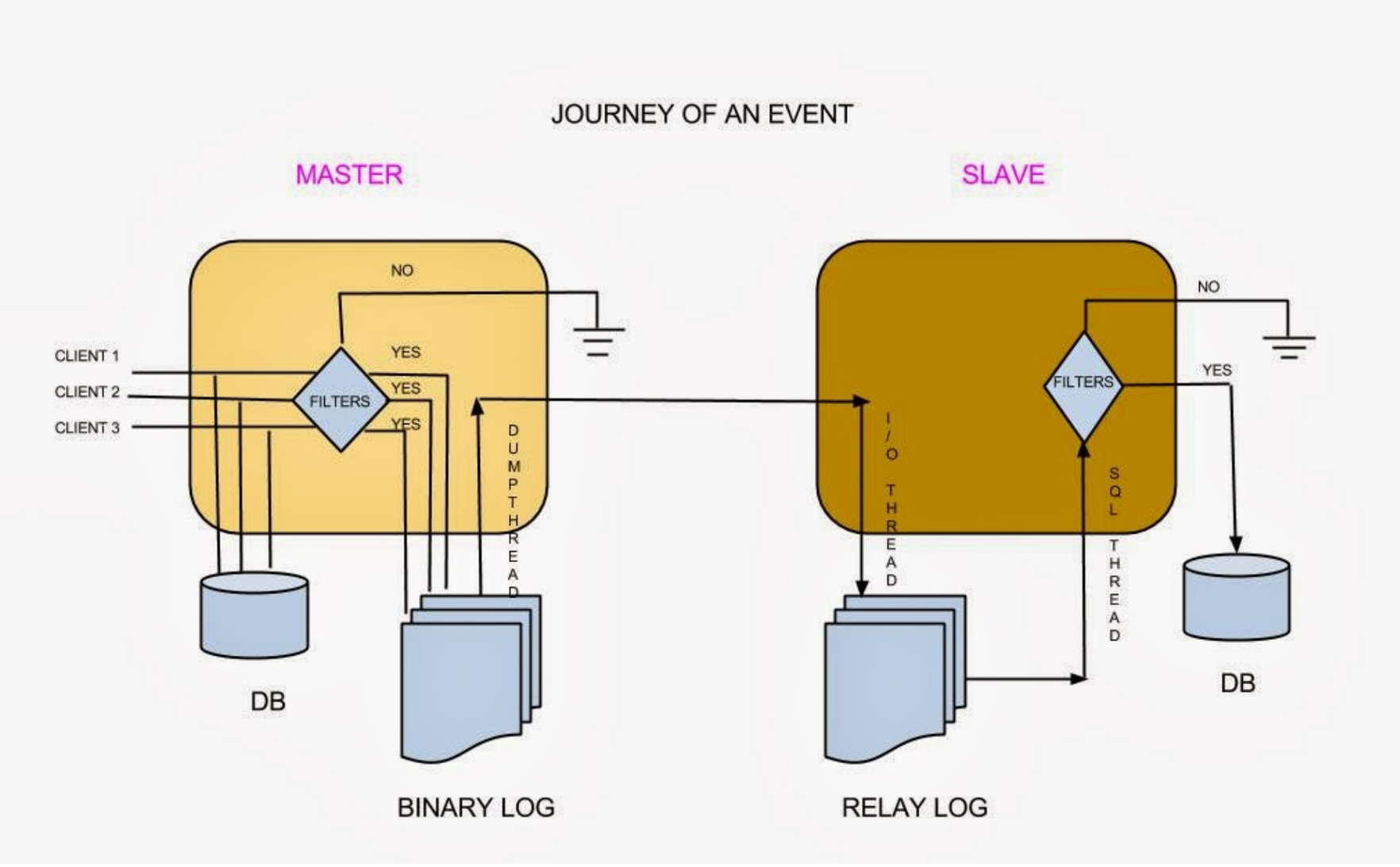
https://dev.mysql.com/doc/refman/8.0/en/replication-rbr-safe-unsafe.html

- передаются измененные строчки
- бинарный формат
- непонятны границы statementa
- его трудно читать
- before/after image

Посмотрите на binlog_row_image:

- full
- minimal
- blob

https://dev.mysql.com/doc/refman/5.7/en/replication-options-binary-log.html#sysvar_binlog_row_image


JOURNEY OF AN EVENT
MASTER
SLAVE
NO
CLIENT 1
CLIENT 2
CLIENT 3
FILTERS
YES
YES
YES
DUMP THREAD
I/O THREAD
FILTERS
NO
YES
SQL THREAD
DB
DB
BINARY LOG
RELAY LOG

Фильтрация репликации

- она есть
- можно реплицировать данные частично
- можно обогощать данные слейва
- использовать осторожно!

Опции:

- replicate_do_db, replicate_ignore_db, replicate_do_table...

https://dev.mysql.com/doc/refman/5.7/en/change-replication-filter.html

binary log position (FILE NAME + OFFSET)

- mysql-bin.00078:44
- локальный для сервера
- обязательно разъедется!

GTID (SOURCE_ID:TRANSACTION_ID)

- 7F33BC78-56CA-44B3-5E33-B34CC7689333:44
- глобален, генерируется автоматически при коммите
- бесплатная трассировка
- простой slave promotion
- используйте его!

https://dev.mysql.com/doc/refman/5.7/en/replication-gtids.html

Параллельная репликация

- обычно используется однопоточная репликация всех данных
- с MySQL5.6 можно реплицировать параллельно несколько баз данных
- с MySQL5.7 можно реплицировать параллельно одни и те же таблицы

Полезные опции:

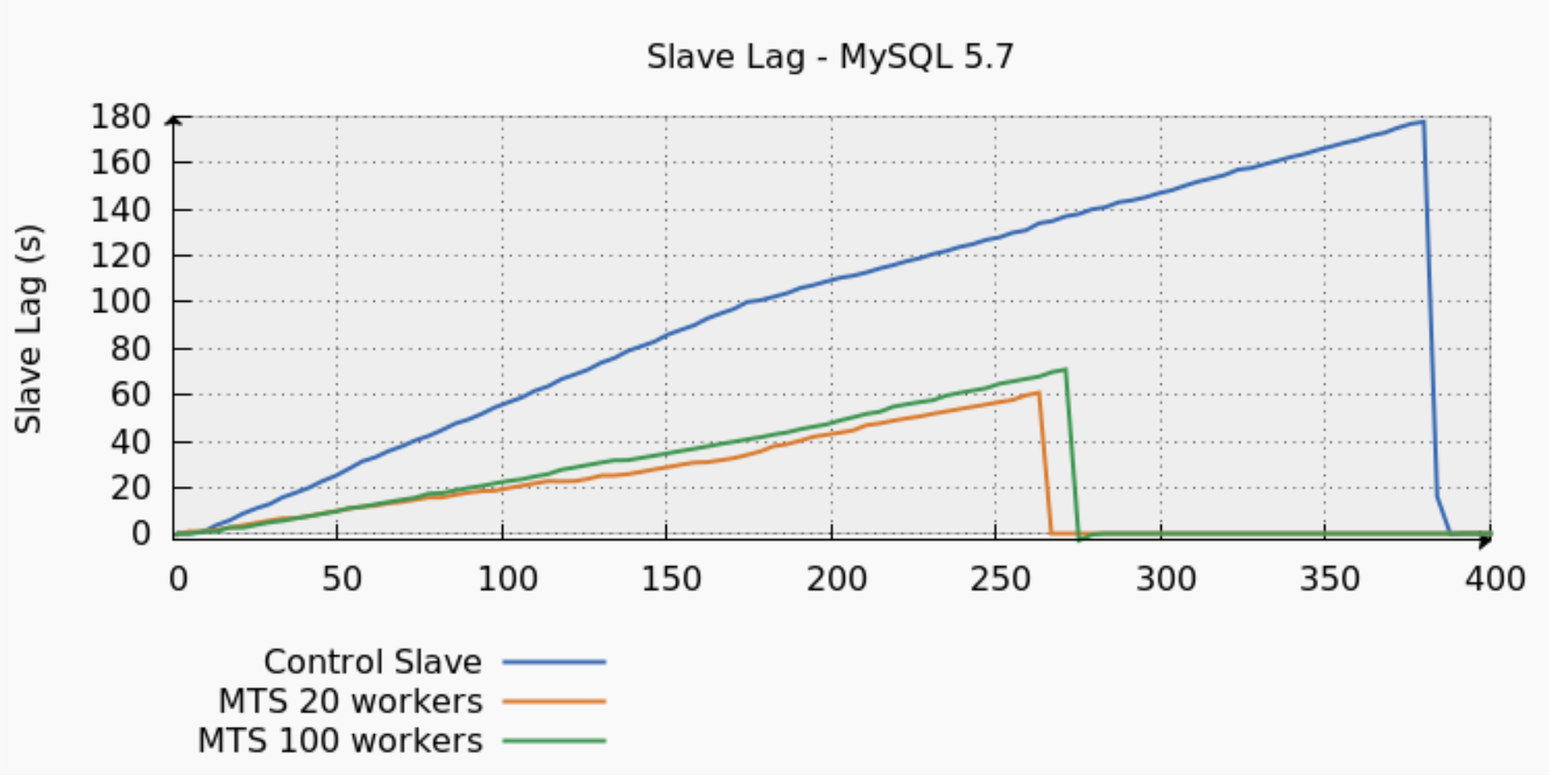
- slave-parallel-workers
- slave-parallel-type (DATABASE|LOGICAL_CLOCK)

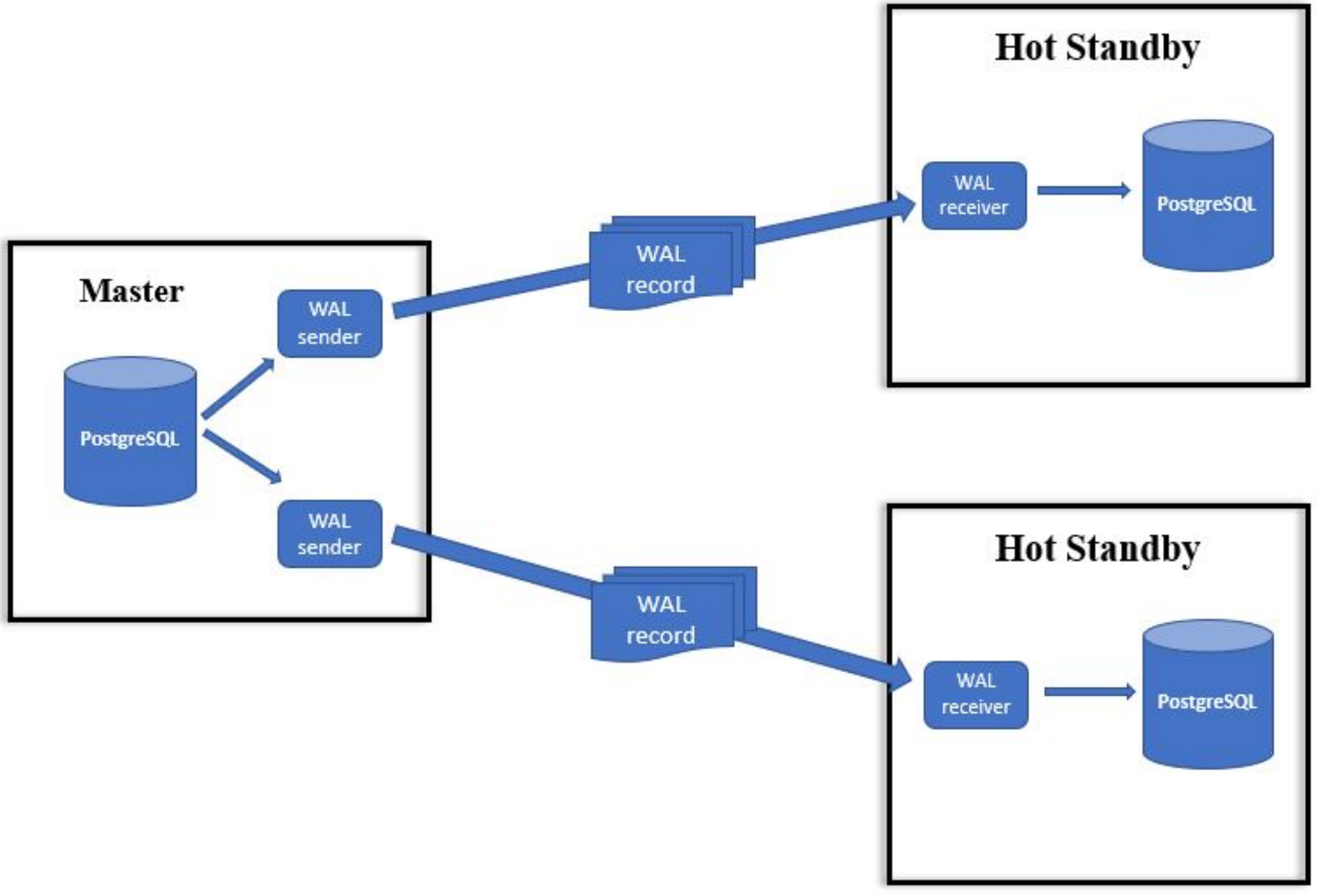
https://dev.mysql.com/doc/refman/5.7/en/replication-options-slave.html

Сценарий:

- 1 мастер, 3 слейва
- первый реплицирует в 1 поток
- второй реплицирует в 20 потоков
- третий реплицирует в 100 потоков
- вставка в 25 различных таблиц внутри одной базы в 100 потоков (sysbench)

https://www.percona.com/blog/2016/02/10/estimating-potential-for-mysql-5-7-parallel-replication/

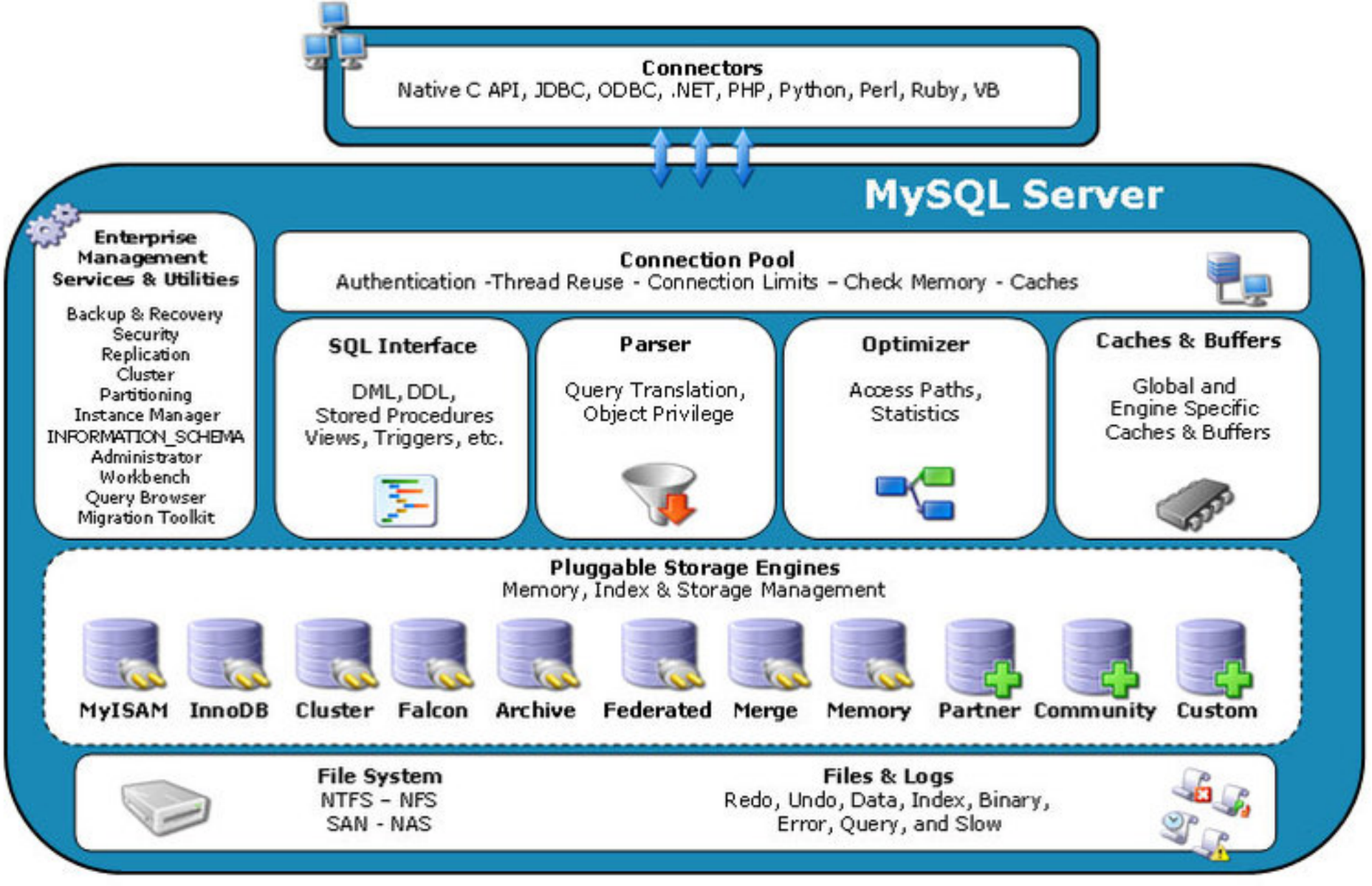

Slave Lag - MySQL 5.7
Slave Lag (s)
180
160
140
120
100
80
60
40
20
0
0
50
100
150
200
250
300
350
400
Control Slave
MTS 20 workers
MTS 100 workers


Hot Standby
WAL receiver
PostgreSQL
WAL record
Master
WAL sender
PostgreSQL
WAL sender
WAL record
Hot Standby
WAL receiver
PostgreSQL

- физические изменения страниц (значение в блоке 2564 равно 154)
- сюда попадают абсолютно все операции
- один журная на все

https://wiki.postgresql.org/images/a/af/FOSDEM-2015-New-WAL-format.pdf

А что с этим в MySQL?



А что с этим в MySQL?

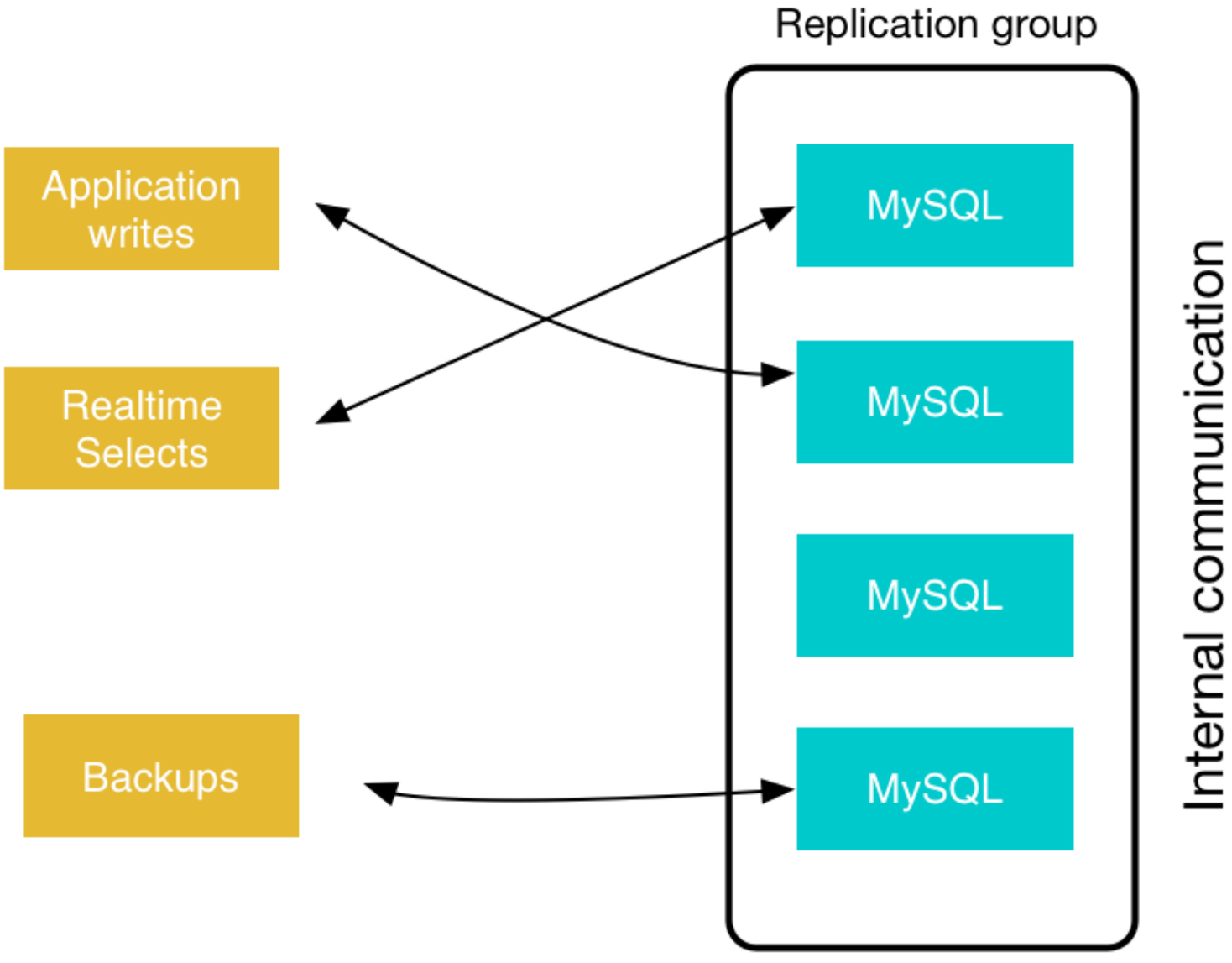
- есть binlog
- у некоторых движков есть аналог WAL (InnoDB Undo/Redo Log)
- с точки зрения MySQL - это разные логи!
- нарушение абстракций

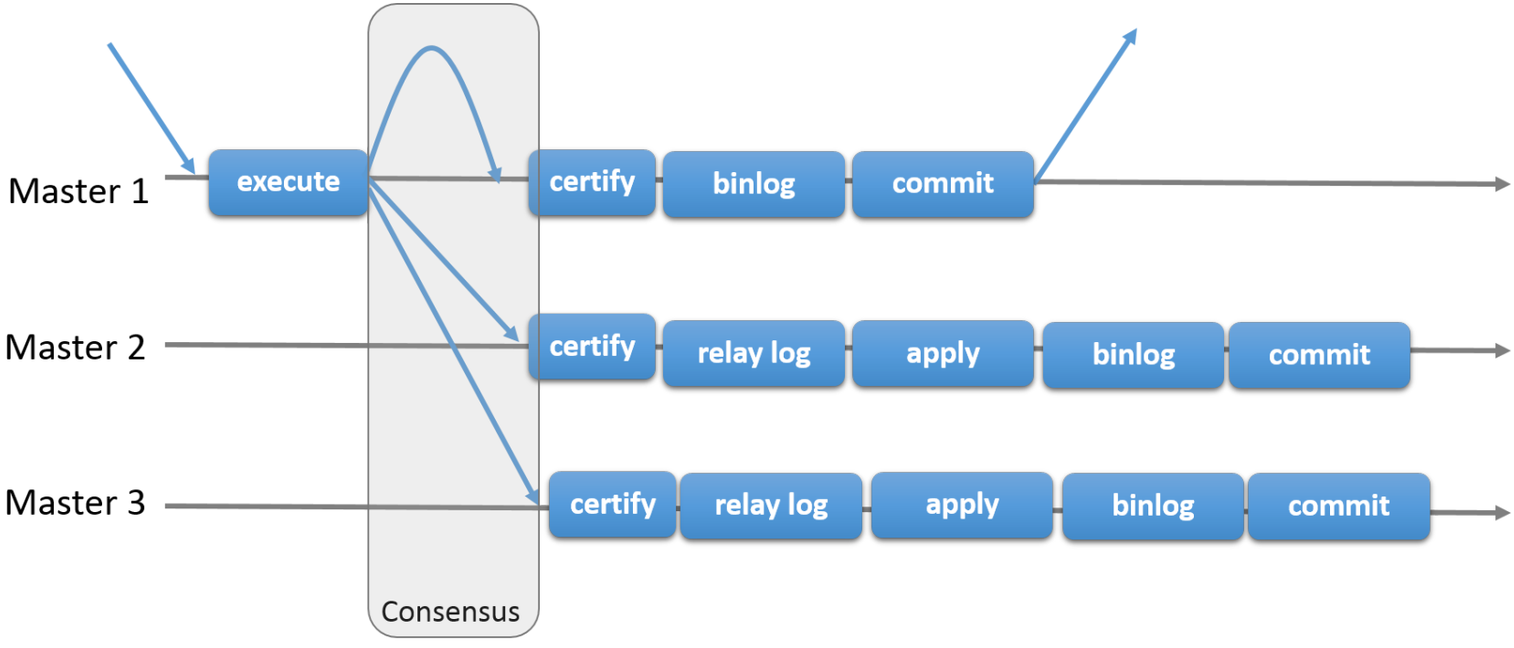
MySQL пишет больше данных для репликации (~1.5 раза)
https://dev.mysql.com/doc/refman/5.7/en/innodb-undo-logs.html

- тот самый row based формат
- работает с кортежами данных
- не знает, как они хранятся на диске
- CPU-bound, можно параллелить по процессорам

В Postgres10+ тоже есть - https://www.postgresql.org/docs/10/logical-replication.html

- работает со страницами
- slave = master байт в байт
- Postgres WAL, InnoDB Undo/RedoLog - как примеры работающих со страницами
- IO-bound, нет смысла параллелить


Replication group
Application writes
Realtime Selects
Backups
MySQL
MySQL
MySQL
MySQL
Internal communication


Master 1
Master 2
Master 3
execute
certify
binlog
commit
certify
relay log
apply
binlog
commit
certify
relay log
apply
binlog
commit
Consensus

- плагин начиная с версии MySQL5.7
- по сути - синхронная репликация
- нет концепта master-slave, скорее master-master
- репликация между всеми нодами
- single primary (по умолчанию)
- кворум, умеет automatic failover
- flow control (я что-то очень отстал)
- лимит в 9 node

Можно попробовать групповую репликацю в докере:

- https://mysqlhighavailability.com/setting-up-mysql-group-replication-with-mysql-docker-images/

- Документация: https://dev.mysql.com/doc/refman/5.7/en/group-replication.html
- Обзор, бенчмарки, сравнения: http://mysqlhighavailability.com/performance-evaluation-mysql-5-7-group-replication/

Доп. материалы

- https://www.youtube.com/watch?v=ppI74hTuXO0
- https://www.youtube.com/watch?v=_r8vLPTl0PE
- https://m.habr.com/ru/company/oleg-bunin/blog/414111/
- https://severalnines.com/database-blog/overview-logical-replication-postgresql
- документация MySQL
- документация Postgres
- блог Percona

Итоги занятия

- Поняли, что такое репликация и зачем она нужна
- Обсудили виды репликации и связанные механизмы
- Поняли разницу между подходами к репликации
- Сравнили некотороые особенности репликации в MySQL и Postgres
- Познакомились с групповой репликацией в MySQL

Вопросы?

Пройдите, пожалуйста, опрос

https://otus.ru/polls/5529/

Спасибо за внимание!