MySQL Replication

Seminario de Desarrollo de Software - Casa Central.

Maximiliano Osorio
mosorio@inf.utfsm.cl

Universidad Técnica Federico Santa María

28 de noviembre de 2017

Replication events

- Statement based – in which case these are write queries
- Row based – in this case these are changes to records, sort of row diffs if you will

On the master

- For replication to work, first of all master needs to be writing replication events to a special log called binary log.
- This is usually very lightweight activity (assuming events are not synchronized to disk), because writes are buffered and because they are sequential.
- The binary log file stores data that replication slave will be reading later.

On the master

- Whenever a replication slave connects to a master, master creates a new thread for the connection
- And then it does whatever the client – replication slave in this case – asks
  1. feeding replication slave with events from the binary log
  2. notifying slave about newly written events to its binary log

On the master

- Slaves that are up to date will mostly be reading events that are still cached in OS cache on the master, so there is not going to be any physical disk reads on the master in order to feed binary log events
- However, when you connect a replication slave that is few hours or even days behind, it will initially start reading binary logs that were written hours or days ago.

On the replica

When you start replication, two threads are started on the slave:

- IO thread: This process called IO thread connects to a master, reads binary log events from the master, they come in and just copies them over to a local log file called relay log.
- SQL thread – reads events from a relay log stored locally on the replication slave (the file that was written by IO thread) and then applies them as fast as possible.

If you want to see where IO thread currently is, check the following in

```
show slave status \G:
```

show slave status

- Master_Log_File – last file copied from the master (most of the time it would be the same as last binary log written by a master)
- Read_Master_Log_Pos – binary log from master is copied over to the relay log on the slave up until this position.

And then you can compare it to the output of `show master status \G` from the master.

SQL thread

reads events from a relay log stored locally on the replication slave (the file that was written by IO thread) and then applies them as fast as possible.

- Relay_Master_Log_File – binary log from master, that SQL thread is "working on"(in reality it is working on relay log, so it's just a convenient way to display information)
- Exec_Master_Log_Pos – which position from master binary log is being executed by SQL thread.

Replication lag

First thing you want to know is which of the two replication threads is behind.

- Most of the time it will be the SQL thread

1. IO bound
2. CPU bound

vmstat

```
[root@smt1 mysql]# vmstat 10
procs -----------memory---------- ---swap-- -----io---- --system-- -----cpu-----
 r  b   swpd   free   buff  cache   si   so    bi    bo   in   cs us sy id wa st
13  0      0 880156  40060 2140824    0    0    12   750  895 1045 32  8 54  6  0
 0  0      0 877116  40060 2141312    0    0     0  1783 2185 23112 44 10 41  5  0
15  0      0 872648  40068 2141960    0    0     0  1747 2204 25743 41 11 46  2  0
 0  0      0 868056  40068 2142604    0    0     0  1803 2164 26224 40 11 44  5  0
17  1      0 863216  40068 2143160    0    0     0  1875 1948 23020 36  9 50  5  0
 0  0      0 858384  40168 2143656    0    0     0  1063 1855 21116 32  9 45 14  0
23  0      0 855848  40176 2144232    0    0     0  1755 2036 23181 36 10 48  6  0
49  0      0 851248  40184 2144648    0    0     0  1679 2313 22832 45 10 40  5  0
10  0      0 846292  40192 2145248    0    0     0  1911 1980 23185 36  9 50  4  0
 0  0      0 844260  40196 2145868    0    0     0  1757 2152 26387 39 11 45  5  0
 0  3      0 839296  40196 2146520    0    0     0  1439 2104 25096 38 10 50  1  0
```

show processlist

```
mysql> show processlist;
+-----+---------------+--------+---------+------+---------------+-------------------------
| Id  | User          |db      | Command | Time | State         | Info
                                                                  | Rows_sent |
    Rows_examined | Rows_read |
+-----+---------------+--------+---------+------+---------------+-------------------------

|   1 | root          |  NULL  | Sleep   |    0 |               | NULL
                                                                   |         0 |
                0 |         0 |
| ...
|  32 | root          |  sbtest | Execute |    0 | NULL          | COMMIT
                                                                  |         0 |
                0 |         0 |
|  33 | root          |  sbtest | Execute |    0 | NULL          | COMMIT
                                                                  |         0 |
                0 |         0 |
|  34 | root          |  sbtest | Execute |    0 | Sorting result | SELECT c from
    sbtest where id between 365260 and 365359 order by c |         0 |             0 |
             0 |
|  35 | root          |  sbtest | Execute |    0 | NULL          | COMMIT
                                                                  |         0 |
                0 |         0 |
|  36 | root          |  sbtest | Execute |    0 | NULL          | COMMIT
                                                                  |         0 |
                0 |         0 |
|  37 | root          |  sbtest | Execute |    0 | NULL          | COMMIT
                                                                  |         0 |
                0 |         0 |
|  38 | root          |  sbtest | Execute |    0 | Writing to net | DELETE from sbtest
    where id=496460                                               |         0 |             1 |
```

fully ACID mode

```
mysql> select @@innodb_flush_log_at_trx_commit;
+----------------------------------+
| @@innodb_flush_log_at_trx_commit |
+----------------------------------+
|                                1 |
+----------------------------------+
```

References


"Mysql replication." https://www.percona.com/resources/mysql-ebooks/mysql-replication.
(Accessed on 10/10/2017).