意想不到的MySQL复制延迟原因

发布于 2022-12-02 19:26:59 559 0 0 代码可运行 举报

文章被收录于专栏： MySQL修行 | 老叶茶馆

导读

> 线上有个 MySQL 实例，存在严重的复制延迟问题，原因出乎意料。

线上有个MySQL 5.7版本的实例，从 服务器 延迟了3万多秒，而且延迟看起来好像还在加剧。

MySQL版本

代码语言：javascript 代码运行次数：0 运行 AI代码解释

```
Server version:     5.7.18-log MySQL Community Server (GPL)
```

看下延迟状况

代码语言：javascript 代码运行次数：0 运行 AI代码解释

```
yejr@imysql.com:mysql3306.sock : (none) > show slave status\G
                Master_Log_File: mysql-bin.013225
            Read_Master_Log_Pos: 1059111551
         Relay_Master_Log_File: mysql-bin.013161
           Exec_Master_Log_Pos: 773131396
                  Master_UUID: e7c35a95-ffb1-11e6-9620-90e2babb5b90
```

我们看到，**binlog文件落后了64个**，相当的夸张。

MySQL 5.7不是已经实现并行复制了吗，怎么还会延迟这么厉害?

先检查系统负载。

```
[yejr@imysql.com mydata]# top
top - 02:04:15 up 1 day,  1:22,  4 users,  load average: 0.98, 1.54, 1.78
Tasks: 935 total,   1 running, 931 sleeping,   3 stopped,   0 zombie
Cpu10 :  4.4%us,  1.7%sy,  0.0%ni, 84.8%id,  9.1%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu11 : 23.4%us,  0.7%sy,  0.0%ni, 74.9%id,  1.0%wa,  0.0%hi,  0.0%si,  0.0%st
...
Cpu18 :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu19 :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu20 : 10.6%us,  0.7%sy,  0.0%ni, 87.7%id,  1.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu21 :  3.3%us,  0.0%sy,  0.0%ni, 96.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu22 :  1.3%us,  0.0%sy,  0.0%ni, 98.0%id,  0.7%wa,  0.0%hi,  0.0%si,  0.0%st
...
Cpu39 :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  264418192k total, 218209296k used, 46208896k free,    19416k buffers
Swap:  8191996k total,        0k used,  8191996k free, 124159436k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 2384 mysql     20   0  188g  83g  11m S 118.5 32.9  89:17.58 mysqld
```

看到mysqld进程其实负载还好，不算太高，也不存在严重的SWAP等问题。

再看I/O子系统负载，没看到这方面存在瓶颈（await\svctm\%util都不高）。

```
[yejr@imysql.com mydata]# sar -d 1
01:55:58 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:02 AM    dev8-0   1112.46   7482.17  80392.05     78.99      6.40      5.75      0.18     20.32

02:03:02 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:03 AM    dev8-0    664.00      0.00  53848.00     81.10      3.95      5.94      0.18     12.00

02:03:03 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:04 AM    dev8-0    668.69      0.00  54181.82     81.03      3.84      5.74      0.19     12.63

02:03:04 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:05 AM    dev8-0    675.00    128.00  61432.00     91.20      4.59      6.80      0.23     15.50

02:03:05 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:06 AM    dev8-0    790.00     96.00  50098.00     63.54      3.95      4.85      0.13     10.00

02:03:06 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:07 AM    dev8-0    787.88     32.32  64371.72     81.74      4.06      5.31      0.15     11.62

02:03:07 AM       DEV       tps  rd_sec/s  wr_sec/s  avgrq-sz  avgqu-sz     await     svctm     %util
02:03:08 AM    dev8-0    825.00     32.00  82264.00     99.75      4.85      5.88      0.18     14.90
```

再看mysqld进程的CPU消耗。

```
[yejr@imysql.com mydata]# pidstat -u -p `pidof mysqld` 1
Linux 2.6.32-642.el6.x86_64 (imysql.com)  05/23/2017      _x86_64_        (40 CPU)

02:04:25 AM       PID    %usr %system  %guest    %CPU   CPU  Command
02:04:26 AM      2384  100.00   12.00    0.00  100.00    10  mysqld
02:04:27 AM      2384  100.00   11.00    0.00  100.00    10  mysqld
02:04:28 AM      2384  100.00   12.00    0.00  100.00     0  mysqld
02:04:29 AM      2384  100.00   10.00    0.00  100.00     0  mysqld
02:04:30 AM      2384  100.00   10.00    0.00  100.00     0  mysqld
02:04:31 AM      2384  100.00   12.00    0.00  100.00     0  mysqld
02:04:32 AM      2384  100.00    9.00    0.00  100.00     0  mysqld
02:04:33 AM      2384  100.00   10.00    0.00  100.00     0  mysqld
02:04:34 AM      2384  100.00   12.00    0.00  100.00    10  mysqld.
```

虽然mysqld进程的CPU消耗总是超过100%，不过也不算太高。

再检查MySQL复制现场，确认了**几个频繁更新的表都有主键，以及必要的索引**。相应的DML操作也几乎都是基于主键或唯一索引条件执行的，**排除无主键、无合理索引方面的因素**。

最后只能祭出**perf top神**器了。

代码语言：javascript　代码运行次数：0　运行　AI代码解释

```
perf top -p `pidof mysqld`
```

看到perf top最后的报告是这样的

代码语言：javascript　代码运行次数：0　运行　AI代码解释

```
Samples: 107K of event 'cycles', Event count (approx.): 29813195000
Overhead  Shared Object         Symbol
  56.19%  mysqld                [.] bitmap_get_next_set
  16.18%  mysqld                [.] build_template_field
   4.61%  mysqld                [.] ha_innopart::try_semi_consistent_read
   4.44%  mysqld                [.] dict_index_copy_types
   4.16%  libc-2.12.so          [.] __memset_sse2
   2.92%  mysqld                [.] ha_innobase::build_template
```

我们看到， **bitmap_get_next_set** 这个函数调用占到了 56.19%，非常高，其次是 **build_template_field** 函数，占了 16.18%。

经过检查MySQL源码并请教MySQL内核开发专家，最后确认这两个函数跟启用表分区有关系。

```
$ grep -r bitmap_get_next_set *
include/my_bitmap.h:extern uint bitmap_get_next_set(const MY_BITMAP *map, uint bitmap_bit);
mysys/my_bitmap.c:uint bitmap_get_next_set(const MY_BITMAP *map, uint bitmap_bit)
sql/partition_info.h:    return bitmap_get_next_set(&read_partitions, part_id);
sql/sql_optimizer.cc:       i= bitmap_get_next_set(table->read_set, i))
sql/sql_partition.cc:    i= bitmap_get_next_set(&part_info->read_partitions, i - 1);
sql/sql_partition.cc:       i= bitmap_get_next_set(&part_info->read_partitions, i))
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(used_partitions, i))
storage/partition/ha_partition.cc:       j= bitmap_get_next_set(&m_locked_partitions, j))
storage/partition/ha_partition.cc:         i= bitmap_get_next_set(&m_part_info->lock_partitions, i))
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(&m_part_info->lock_partitions, i))
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(&m_bulk_insert_started, i))
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(&m_partitions_to_reset, i))
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(&m_part_info->lock_partitions, i))
storage/partition/ha_partition.cc:  for (i= bitmap_get_next_set(&m_part_info->lock_partitions, i);
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(&m_part_info->lock_partitions, i))
storage/partition/ha_partition.cc:       i= bitmap_get_next_set(&m_part_info->lock_partitions, i))
unittest/gunit/my_bitmap-t.cc:       test_bit= bitmap_get_next_set(map, test_bit))
```

查询下当前实例有多少个表分区：

代码语言：javascript　　代码运行次数：0　　运行　　AI代码解释

```
yejr@imysql.com:mysql3306.sock : (none) > select count(*) from partitions whe
+----------+
| count(*) |
+----------+
|    32128 |
+----------+
1 row in set (11.92 sec)
```

额滴神啊，竟然有3万多个表分区，难怪上面那两个函数调用那么高。

这个业务 数据库 几个大表采用每天一个分区方案，而且把直到当年年底所有分区也都给提前创建好了，所以才会有这么多。

不过，虽然有这么多表分区，在master服务器上却不存在这个瓶颈，看起来是在主从复制以及大量表分区的综合因素下才有这个瓶颈，最终导致主从复制延迟越来越严重。

知道问题所在，解决起来就简单了。**把到下个月底前用不到的表分区全部删除，之后约只剩下1.6万个分区。重启slave线程，问题解决，主从复制延迟很快就消失了。**

《深入浅出MGR》视频课程

戳此 小程序 即可直达B站

https://www.bilibili.com/medialist/play/1363850082?business=space_collection&business_id=343928&desc=0

数据库　云数据库 SQL Server　sql

评论

登录 后参与评论

推荐阅读

编辑精选文章

换一批

- 万字详解高可用架构设计 5552
- Go 开发者必备：Protocol B... 3297
- 10分钟带你彻底搞懂分布式... 2422
- 多租户的 4 种常用方案 4750
- 亿级月活的社交 APP，陌陌... 3369
- 60页PPT全解：DeepSeek... 4723

[MySQL FAQ]系列 — MySQL复制中slave延迟监控

sql 数据库 云数据库 SQL Server unix linux

在MySQL复制环境中，我们通常只根据 Seconds_Behind_Master 的值来判断SLAVE的延迟。这么做大部分情况下尚可接受，但并不够准确，而应该考虑更多因素。

老叶茶馆 · 2022/12/02 863 0

实例解析MySQL性能瓶颈排查定位

云数据库 SQL Server sql 数据库

登入服务器后，我们的目的是首先要确认当前到底是哪些进程引起的负载高，以及这些进程卡在什么地方，瓶颈是什么。

用户1278550 · 2020/02/26 1.8K 0

赞！7000 字学习笔记，MySQL 从入到放弃

云数据库 SQL Server 数据库 数据备份 sql 存储

MySQL近两年一直稳居第二，随时有可能超过Oracle计晋升为第一名，因为MySQL的性能一直在被优化，同时安全机制也是逐渐成熟，更重要的是开源免费的。

民工哥 · 2020/09/15 750 0

MySQL之GTID主从复制

event global ip mysql sequence

GTID即全局事务ID (global transaction identifier), 其保证为每一个在主上提交的事务在复制集群中可以生成一个唯一的ID。

Alone-林 · 2023/03/17 1.5K 0

MySQL 传统复制中常见故障处理和结构优化案例分析

云数据库 SQL Server　sql

虽然MySQL5.7 的主从复制已经很稳定了，但在备库可读写的情况下，总是会出现部分数据不一致的情况，例如常见的1062、1032和1050错误。下面就介绍下这类报错...

数据和云 · 2018/03/08　792　0

轻松搞定！MySQL 主从复制的原理、配置和玩法

配置　日志　数据　原理　mysql

MySQL 主从复制在数据库管理中地位关键，能同步主从库数据。它通过二进制日志等组件实现数据备份、负载均衡与高可用。其原理涉及主从库工作流程及一致性保障机制。实际配置需准备环境，按步骤设置主从库参数。在...

菱悻的小杀马特. · 2025/04/18　331　0

由MySQL复制延迟说起

云数据库 SQL Server　数据库　sql

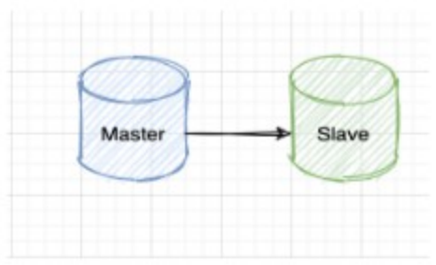
相信 slave 延迟是MySQL dba 遇到的一个老生长谈的问题了。我们先来分析一下slave延迟带来的风险

用户1278550 · 2019/04/25　1.4K　0

看完这篇还不懂 MySQL 主从复制，可以回家躺平了～

云数据库 SQL Server　数据库　sql　负载均衡　负载均衡缓存

我们在平时工作中，使用最多的数据库就是 MySQL 了，随着业务的增加，如果单单靠一台服务器的话，负载过重，就容易造成宕机。

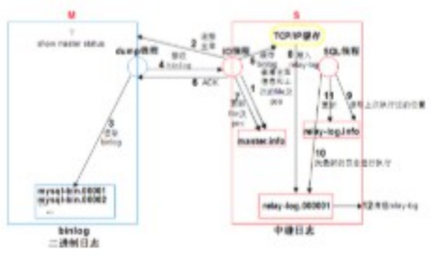


浅羽技术 · 2021/06/22　656　0

搭建Mysql主从复制

mysql　服务器　配置　日志　线程

MySQL主从复制是一种常用的数据库高可用性解决方案，可以提高数据库的可用性和性能。本教程将介绍如何搭建MySQL主从复制。



夕阳也是醉了 · 2023/10/16　547　0

mysql AB复制搭建以及常见故障排查

云数据库 SQL Server　数据库　sql

MySQL主从复制(Master-Slave)也叫AB复制，Mysql作为目前世界上使用最广泛的免费数据库，相信所有从事系统运维的工程师都一定接触过。但在实际的生产环境中...

DevinGeng · 2019/04/09　823　0

MySQL主从复制延迟解决方案

sql commit mysql worker 事务

前面一篇，我们学习到了MySQL多版本并发控制（MVCC）实现原理，这一篇我们接着学习MySQL主从复制模式下的延迟解决方案。


兔云小新LM · 2023/03/16 5K 0

MySQL Replication 主从复制全方位解决方案

数据备份 数据库 sql 云数据库 SQL Server html

在了解主从复制之前必须要了解的就是数据库的二进制日志(binlog),主从复制架构大多基于二进制日志进行，二进制日志相关信息参考：http://www.cnblogs.com/clsn/p/8087678.html#_label6


惨绿少年 · 2019/05/24 925 0

谈谈 MySQL 延迟复制的几个好处

云数据库 SQL Server 数据库 sql javascript node.js

MySQL 的主从复制(Replication)关系，不太严谨的叫法是“同步”或者“主从同步”。实际上在早期，MySQL 的主从并不能实现真正的“同步”(Sync)，而是“异步”的(Async)。


iMike · 2019/06/02 1.5K 0

mysql主从复制

sql 数据库 云数据库 SQL Server 容器镜像服务 容器

由于我这里使用docker搭建，所以需要把配置文件和数据文件映射到宿主机，让容器运行时挂载数据


earthchen · 2020/09/24 1.1K 0

MySQL FAQ 系列 — MySQL 复制中 slave 延迟监控

云数据库 SQL Server

本文介绍了如何利用 MySQL 5.7 的行锁信息表特性，通过解析 binlog，实时监控行锁状态，从而实现对 MySQL 5.7 的行锁的全局统计和优化。


叶金荣 · 2017/05/10 2.8K 1

MySQL优化/面试，看这一篇就够了

存储

price decimal(8,2)有2位小数的定点数，定点数支持很大的数（甚至是超过int,bigint存储范围的数）


乔戈里 · 2019/01/07 1.9K 0

技术分享 | MySQL 突如其来的主从复制延迟

编程算法　数据库　sql　云数据库 SQL Server　linux

爱可生交付服务团队北京 DBA，对数据库及周边技术有浓厚的学习兴趣，喜欢看书，追求技术。

爱可生开源社区 · 2021/11/01　1.8K　0

MySQL复制从库延迟优化思路

事务　优化　mysql　变量　配置

2、主从延迟常见的原因有哪些？ 1、大事务，从库回放时间较长，导致主从延迟 2、主库写入过于频繁，从库回放跟不上 3、参数配置不合理 4、主从硬件差异 5、网络...

老叶茶馆 · 2024/05/09　576　0

由MySQL复制延迟说起

云数据库 SQL Server　数据库　sql

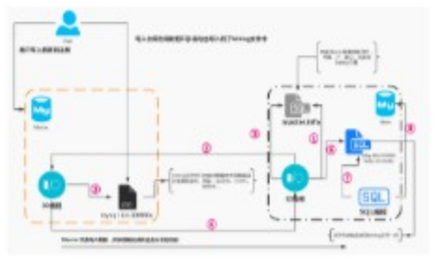
杨奇龙，网名“北在南方”，7年DBA老兵，目前任职于杭州有赞科技DBA，主要负责数据库架构设计和运维平台开发工作，擅长数据库性能调优、故障诊断。

田帅萌 · 2019/05/13　1.2K　0

docker学习系列12 轻松实现 mysql 主从同步

容器镜像服务　云数据库 SQL Server　运维

docker的一大好处是在本地可以很方便快速的搭建负载均衡，主从同步等需要多主机的环境。 可以说是极大方便了运维成本和难度。 本节在本地搭建mysql的一主一从...



mafeifan · 2018/09/10　858　0