新炬网络
SNC www.shsnc.cn

Greenplum在大数据生产环境中的运维实战

王江

2016-11-25

上海新炬网络技术有限公司
shanghai new century network co.ltd

目录

GP运维案例一：锁问题处理

业务介绍
GP数据库，每15分钟需要录入一批数据并分析。

应用流程
每15分钟新建一分区 -> 分区导入数据 -> Vacuum analyze分区 -> 应用业务计算（通常是全表操作） -> 删除旧分区

问题出现
经常出现新分区无法创建成功，导致应用持续报错，业务流程无法继续。

原因分析
长时间表查询sql导致长时间表锁，而创建新分区操作需要AccessExclusiveLock，锁冲突因此无法创建分区。

问题定位
查看gp_locks_on_relation，发现该表被一个语句锁住，导致分区无法创建，锁住时间一长，应用重试失败便会报错。

锁类型
关系锁:gp_toolkit.gp_locks_on_relation
资源队列锁:gp_toolkit.gp_locks_on_resqueue

表操作	锁类型
select	AccessShareLock
insert	RowExclusiveLock
update	ExclusiveLock
delete	ExclusiveLock
truncate	AccessExclusiveLock
select for update	ExclusiveLock + AccessShareLock
drop	AccessExclusiveLock
alter table	AccessExclusiveLock
Vacuum/analyze	ShareUpdateExclusiveLock

分区表
1、分区的select、insert等操作，只对分区加锁
2、分区的create/drop操作，实际上是：
alter table 表名 add/drop partition 分区名

GP运维案例一：锁问题处理

现场重现

1、开启事务，对test_par分区表做全表操作：select count(*) from test_par;
2、再对test_par执行新增分区操作： alter table test_par add partition p4 start (30001) end (40000);
3、观察gp_toolkit.gp_locks_on_relation视图

锁对象	表名	锁类型	锁占有	语句
relation	test_par	AccessExclusiveLock	f	alter table test_par add partition p4 **
relation	test_par	AccessShareLock	t	<IDLE> in transaction
relation	test_par_1_prt_p1	AccessShareLock	t	<IDLE> in transaction
relation	test_par_1_prt_p2	AccessShareLock	t	<IDLE> in transaction
relation	test_par_1_prt_p3	AccessShareLock	t	<IDLE> in transaction

解决方法

1、定时统一创建/删除多个分区
2、及时查杀长时间sql
3、尽量减少全表扫描
4、改vacuum为analyze

锁问题引申

1、业务流程类似，但正式系统很少出现此类问题（正式系统的业务处理窗口以天计算）
2、后期业务繁忙，同一个用户有多个并发连接操作，数据库又开始出现应用失败现象。

GP运维案例一：锁问题处理

GP锁冲突关系表

锁冲突表格	Current Lock Mode							
	ACCESS SHARE	ROW SHARE	ROW EXCLUSIVE	SHARE UPDATE EXCLUSIVE	SHARE	SHARE ROW EXCLUSIVE	EXCLUSIVE	ACCESS EXCLUSIVE
ACCESS SHARE							X	X
ROW SHARE							X	X
ROW EXCLUSIVE					X	X	X	X
SHARE UPDATE EXCLUSIVE				X	X	X	X	X
SHARE			X	X		X	X	X
SHARE ROW EXCLUSIVE			X	X	X	X	X	X
EXCLUSIVE		X	X	X	X	X	X	X
ACCESS EXCLUSIVE	X	X	X	X	X	X	X	X

GP运维案例二：资源队列

资源队列：资源队列可以控制执行成本、活动语句数量、执行优先级等。

每个新建用户默认属于pg_default资源队列。

Superuser用户不受资源队列限制。

create resource queue rsqname with （****）

资源队列参数：

active_statements： 同时执行的sql个数
max_cost： cost最大值，超过会报错，无法执行
min_cost： cost在伐值以下，可直接执行
cost_overcommit： 系统没有其他语句执行时，超过资源队列Cost阀值的语句可以被执行。
priority： 资源队列的优先级
memory_limit： 该资源队列的最大内存限制，建议和active_statements一同使用。

常用视图

gp_toolkit.gp_locks_on_resqueue	--资源队列中的锁
gp_toolkit.gp_resq_priority_statement	--资源队列中语句运行的优先级
gp_toolkit.gp_resq_activity	--资源队列中语句的状态

GP运维案例二：资源队列

锁问题引申：多并发情况下，导致应用执行失败。

现场重现
1、创建资源队列，限制sql并发为2。
2、在该资源队列同时运行3个语句。

gp_toolkit.gp_resq_activity_by_queue

resoid	resqname	resqlast	resqstatus	resqtotal
25547	test	2016-11-23 10:33:**	running	2
25547	test	2016-11-23 10:33:**	waiting	1

gp_toolkit.gp_locks_on_resqueue

表名	资源队列	lorlocktype	lorpid	lormode	lorgranted	lorwaiting
test	test	resource queue	8457	ExclusiveLock	t	f
test	test	resource queue	8634	ExclusiveLock	t	f
test	test	resource queue	8546	ExclusiveLock	f	t

解决方法
1、根据现场情况增大并发sql数
2、根据现场情况，重新规划资源队列（如内存、cost限制）

GP运维案例三：数据倾斜

数据倾斜

数据按照指定的分布键插入到数据表后，并未平均的分布到各个segnemt节点，而是集中存放在少数的segment节点。

数据倾斜的缺点

1、导致少数节点存储不足。
2、导致计算倾斜。
3、影响sql执行效率

数据倾斜解决方法

1、alter table，重新指定分布键
2、指定分布方式为random。

检查数据是否倾斜

gp_skew_coefficients --过计算各实例之间的差异系数显示数据分布的倾斜
gp_skew_idle_fractions --通过计算表扫描期间的系统空闲百分比显示数据分布的倾斜

```
select gp_segment_id,count(*) from test group by gp_segment_id;

 gp_segment_id    | count
------------------+-------
            1     |     4
            0     |    32
```

GP运维案例四：统计信息、执行计划

统计信息

GP的统计信息为基于代价的方式（COSTS），可以通过analyze命令收集统计信息。

统计信息参数

gp_autostats_mode：设置统计信息收集方式。

none ： 不开启自动收集。

on_no_stats：默认方式，当新建表或者第一次插入数据时触发。

on_change：当更新或者插入的数据量到达一定级别时触发。（现场使用）

缺乏统计信息的缺点

1、导致错误的执行计划

执行计划

```
postgres=# explain select * from test;
                                      QUERY PLAN
---------------------------------------------------------------------------------------
 Gather Motion 2:1  (slice1; segments: 2)  (cost=0.00..827.00 rows=72700 width=15)
   ->  Seq Scan on test  (cost=0.00..827.00 rows=36350 width=15)
(2 rows)

postgres=# select count(*) from test;
 count
-------
     0
(1 row)
```

GP运维案例五：系统表膨胀

案例1：查询系统表pg_class缓慢。

pg_class数据量20多万，大小100多GB！

通过explain查看发现执行计划有误。

解决方法：

定时vacuum analyze系统表

小提示：
普通表load数据后，记得进行analyze

案例2：查询系统表pg_class缓慢。

pg_class 数据量140万，大小200多GB！

已定时vacuum analyze系统表。

pg_namespace 表大小，200多MB！

解决方法：

定时统一清理残留临时表。

小提示：
如果会话异常中断，可能会导致temp表残留

GP运维案例六：pgsql_tmp目录膨胀

现象描述

GP测试库数据目录暴增到90%，近3天目录平均增长在10%左右。

问题定位

经过观察，发现单个segment实例的pgsql_tmp目录大小达到了3T以上，并且还在继续增长。

pgsql_tmp目录主要存放临时文件，用于内存外排序。

出现该问题的原因可能是某个高耗sql造成的。

现场重现

Test1(id, name, time),1亿条数据

Test2(id, name, time),2亿条数据

语句

```
select * from test1,test2 where test1.id=test2.id order by test1.time;
```

```
[gpadmin@greenplum pgsql_tmp]$ du -sh
1.9G    .
```

```
postgres=# select * from test1,test2 where test1.id=test2.id order by test1.time;
ERROR:  could not write to temporary file: No space left on device  (seg0 slice1 greenplum:50000 pid=15178)
```

解决方法

1、查找并杀掉长时间执行sql。（找到一个连续运行了4天的sql）

2、长耗时sql监控告警

3、磁盘目录（ pgsql_tmp ）空间不足告警

4、资源队列限制（内存使用限制）

GP运维案例七：常用数据库命令

gpstart –m ： 单独启动master进程，进入限制模式，常用于维护（只允许超级用户登录）

gpstop –fa ： 在数据库还有应用连接时强制断开连接，再关闭数据库

gpstop –u ： 重新加载配置（pg_hba.conf等）

gpstate –f ： 查看master standby节点状态

gpstate –e ： 查看segment节点状态

gprecoverseg ：segment节点修复

gprecoverseg –r ： segment主备节点角色的恢复

gpconfig –s 参数名 ： 查看master和segment指定参数设置情况

gpactivatestandby ： 激活备节点（master和standby的切换需要手动激活）

PGOPTIONS=`-c gp_session_role=utility’ psql ： segment节点登陆
master节点限制模式登陆

Greenplum运维总结

前期规划

1、《greenplum数据库最佳实践》姚延栋 刘奎恩

2、《 Pivotal践行见远技术篇》

3、磁盘规划（XFS文件系统、filespace、表空间）

4、操作系统设置（IO、内存、共享内存、gpcheckperf）

5、创建资源队列

6、制定开发规范（禁止高耗语句，表压缩，列存储、表分区）

7、制定运维方案（定期备份元数据，定期巡检磁盘等）

8、GPCC （图形监控。以及部分常用表：如queries_history）

9、布置监控程序（nmon等）

Greenplum运维总结

后期维护

1、常用脚本

- 定时vacuum analyze系统表
- 数据库状态检查（gpstate）
- IDLE连接查杀
- 超长耗时sql检查、长时间锁检查
- 磁盘监控（data目录、pgsql_tmp目录等）

2、sql优化：

- 分布键（数据分布是否均衡，分布键是否为关联键等）
- 数据量，是否需要分区
- 统计信息是否正确
- 高耗语句（多个大表关联拆分sql）
- 执行计划

3、数据库调优（max_connections、gp_vmem_protect_limit等参数）

4、数据库检查（gpcheckcat）

Greenlum运维提升

1、熟悉业务，站在业务的角度，因地制宜的给出相应的对策

2、数据资产管理（数据质量、数据标准、生命周期、元数据管理）

3、数据模型

4、不断提升技术视野，了解其他大数据架构和技术

Thank you

为客户创造价值

服务至上

诚信至上