问题整理

# 问题描述

nucleic\_acid\_testing表全量数据91274058，加上过滤条件后数据：2925634。

Sql:

|  |
| --- |
| Selectcount(1) from epidemic\_graycode.gz\_health\_code\_day3check2\_data where status=1 and id\_card is not null and scan\_time<'2022-07-07 00:00:00' |

gz\_health\_code\_day3check2\_data表全量数据127554，加上过滤条件后数据：52472。

Sql:

|  |
| --- |
| select count(1) from epidemic\_data.nucleic\_acid\_testing where datetime>'2022-06-30 00:00:00' and id\_card is not null |

两表关联后数据量为：36259

两表直接进行inner join关联，做count统计，耗时只需要9秒。Sql语句如下：

|  |
| --- |
| select */\*+ parallel(108) \*/ count*(*\**) from(select id\_card,scan\_time,status,remark from epidemic\_graycode.gz\_health\_code\_day3check2\_data where status=1 and id\_card is not null and scan\_time<'2022-07-07 00:00:00') t1inner join(select id\_card,hscjsj from epidemic\_data.nucleic\_acid\_testing where datetime>'2022-06-30 00:00:00' and id\_card is not null) gon t1.id\_card=g.id\_card; |

两表进行inner join关联后，对关联后数做group by统计，并对统计结果加过滤条件，却耗时达到5分钟。

Sql语句如下：

|  |
| --- |
| select */\*+ parallel(108) \*/* id\_card, *count*(hscjsj) num from(select t1.\*,g.hscjsj from(select id\_card,scan\_time,status,remark from epidemic\_graycode.gz\_health\_code\_day3check2\_data where status=1 and id\_card is not null and scan\_time<'2022-07-07 00:00:00') t1inner join(select id\_card,hscjsj from epidemic\_data.nucleic\_acid\_testing where datetime>'2022-06-30 00:00:00' and id\_card is not null) gon t1.id\_card=g.id\_card) r group by id\_card having num > 2; |

对于以上两个sql语句，同样的数据量，group by 耗时比直接count耗时差距太大，麻烦Oceanbase社区老师看看是什么原因导致group by查询语句那么慢？

# 二、建表语句

nucleic\_acid\_testing：

|  |
| --- |
| CREATE TABLE `nucleic\_acid\_testing` ( `check\_id` varchar(100) DEFAULT NULL, `name` varchar(100) DEFAULT NULL COMMENT '', `phone` varchar(20) DEFAULT NULL COMMENT '', `hsjcjgmc` varchar(100) DEFAULT NULL COMMENT '', `hscjsj` varchar(100) DEFAULT NULL COMMENT '', `hsjcjg` varchar(100) DEFAULT NULL COMMENT '', `id\_card` varchar(30) DEFAULT NULL COMMENT '', `query\_time` bigint(20) DEFAULT NULL COMMENT '', `datetime` datetime(3) DEFAULT NULL COMMENT '（datetime）', `source` varchar(50) DEFAULT NULL COMMENT '', `cj\_datetime` datetime(3) DEFAULT NULL COMMENT '（datetime）', KEY `nucleic\_acid\_testing\_phone\_IDX` (`phone`) BLOCK\_SIZE 16384 LOCAL, KEY `nucleic\_acid\_testing\_id\_card\_IDX` (`id\_card`) BLOCK\_SIZE 16384 LOCAL, KEY `nucleic\_acid\_testing\_\_index\_cjdt` (`cj\_datetime`) BLOCK\_SIZE 16384 LOCAL, KEY `nucleic\_acid\_testing\_\_index\_jcdt` (`datetime`) BLOCK\_SIZE 16384 LOCAL) DEFAULT CHARSET = utf8mb4 ROW\_FORMAT = COMPACT COMPRESSION = 'zstd\_1.3.8' REPLICA\_NUM = 3 BLOCK\_SIZE = 16384 USE\_BLOOM\_FILTER = FALSE TABLET\_SIZE = 134217728 PCTFREE = 0 TABLEGROUP = 'tpch\_epidemic\_zjhm\_phone\_data' COMMENT = ' ' partition by key(id\_card,phone) |

gz\_health\_code\_day3check2\_data：

|  |
| --- |
| CREATE TABLE `gz\_health\_code\_day3check2\_data` ( `id` bigint(20) NOT NULL, `name` varchar(255) DEFAULT NULL COMMENT ' ', `id\_card` varchar(50) DEFAULT NULL COMMENT ' ', `phone` varchar(50) DEFAULT NULL COMMENT ' ', `scan\_time` datetime NOT NULL COMMENT ' ', `status` tinyint(4) DEFAULT NULL COMMENT ' ', `remark` varchar(512) DEFAULT NULL COMMENT ' ', `create\_time` datetime DEFAULT NULL COMMENT ' ', `update\_time` datetime DEFAULT NULL COMMENT ' ', `operator` varchar(50) DEFAULT NULL COMMENT ' ', `reason` tinyint(4) NOT NULL COMMENT ' ', PRIMARY KEY (`id`), KEY `gz\_health\_code\_day3check2\_data\_IDX` (`id\_card`) BLOCK\_SIZE 16384 LOCAL) DEFAULT CHARSET = utf8mb4 ROW\_FORMAT = COMPACT COMPRESSION = 'zstd\_1.3.8' REPLICA\_NUM = 3 BLOCK\_SIZE = 16384 USE\_BLOOM\_FILTER = FALSE TABLET\_SIZE = 134217728 PCTFREE = 0; |

# 三、sql explain信息

|  |
| --- |
| 执行慢语句,执行完花费时间5分钟左右:explainselect */\*+ parallel(108) \*/* id\_card, *count*(hscjsj) num from(select t1.\*,g.hscjsj from(select id\_card,scan\_time,status,remark from epidemic\_graycode.gz\_health\_code\_day3check2\_data where status=1 and id\_card is not null and scan\_time<'2022-07-07 00:00:00') t1inner join(select id\_card,hscjsj from epidemic\_data.nucleic\_acid\_testing where datetime>'2022-06-30 00:00:00' and id\_card is not null) gon t1.id\_card=g.id\_card) r group by id\_card having num > 2;=========================================================================================|ID|OPERATOR |NAME |EST. ROWS|COST |---------------------------------------------------------------------------------------*--*|0 |PX COORDINATOR | |4 |349255||1 | EXCHANGE OUT DISTR |:EX10002 |4 |349250||2 | HASH GROUP BY | |4 |349250||3 | EXCHANGE IN DISTR | |135 |349147||4 | EXCHANGE OUT DISTR (HASH) |:EX10001 |135 |348949||5 | MATERIAL | |135 |348949||6 | NESTED-LOOP JOIN | |135 |348183||7 | EXCHANGE IN DISTR | |131 |343104||8 | EXCHANGE OUT DISTR (BC2HOST)|:EX10000 |131 |343046||9 | PX BLOCK ITERATOR | |131 |343046||10| TABLE SCAN |gz\_health\_code\_day3check2\_data|131 |343046||11| PX PARTITION ITERATOR | |1 |39 ||12| TABLE SCAN |nucleic\_acid\_testing |1 |39 |=========================================================================================Outputs & filters: -----------------------------------*--* 0 - output([gz\_health\_code\_day3check2\_data.id\_card], [T\_FUN\_COUNT(nucleic\_acid\_testing.hscjsj)]), filter(nil) 1 - output([gz\_health\_code\_day3check2\_data.id\_card], [T\_FUN\_COUNT(nucleic\_acid\_testing.hscjsj)]), filter(nil), dop=108 2 - output([gz\_health\_code\_day3check2\_data.id\_card], [T\_FUN\_COUNT(nucleic\_acid\_testing.hscjsj)]), filter([T\_FUN\_COUNT(nucleic\_acid\_testing.hscjsj) > 2]),  group([gz\_health\_code\_day3check2\_data.id\_card]), agg\_func([T\_FUN\_COUNT(nucleic\_acid\_testing.hscjsj)]) 3 - output([gz\_health\_code\_day3check2\_data.id\_card], [nucleic\_acid\_testing.hscjsj]), filter(nil) 4 - (*#keys=1, [gz\_health\_code\_day3check2\_data.id\_card]), output([gz\_health\_code\_day3check2\_data.id\_card], [nucleic\_acid\_testing.hscjsj]), filter(nil), dop=108* 5 - output([gz\_health\_code\_day3check2\_data.id\_card], [nucleic\_acid\_testing.hscjsj]), filter(nil) 6 - output([gz\_health\_code\_day3check2\_data.id\_card], [nucleic\_acid\_testing.hscjsj]), filter(nil),  conds(nil), nl\_params\_([gz\_health\_code\_day3check2\_data.id\_card]) 7 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter(nil) 8 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter(nil), dop=108 9 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter(nil) 10 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter([gz\_health\_code\_day3check2\_data.status = 1], [gz\_health\_code\_day3check2\_data.scan\_time < ?], [(T\_OP\_IS\_NOT, gz\_health\_code\_day3check2\_data.id\_card, NULL, 0)]),  access([gz\_health\_code\_day3check2\_data.status], [gz\_health\_code\_day3check2\_data.id\_card], [gz\_health\_code\_day3check2\_data.scan\_time]), partitions(p0) 11 - output([nucleic\_acid\_testing.hscjsj]), filter(nil) 12 - output([nucleic\_acid\_testing.hscjsj]), filter([nucleic\_acid\_testing.datetime > ?], [(T\_OP\_IS\_NOT, nucleic\_acid\_testing.id\_card, NULL, 0)]),  access([nucleic\_acid\_testing.id\_card], [nucleic\_acid\_testing.datetime], [nucleic\_acid\_testing.hscjsj]), partitions(p[0-215])直接关联，count语句,执行完花费时间9s：explain select */\*+ parallel(108) \*/* count(\*) from(select id\_card,scan\_time,status,remark from epidemic\_graycode.gz\_health\_code\_day3check2\_data where status=1 and id\_card is not null and scan\_time<'2022-07-07 00:00:00') t1inner join(select id\_card,hscjsj from epidemic\_data.nucleic\_acid\_testing where datetime>'2022-06-30 00:00:00' and id\_card is not null) gon t1.id\_card=g.id\_card;=======================================================================================|ID|OPERATOR |NAME |EST. ROWS|COST |-------------------------------------------------------------------------------------*--*|0 |SCALAR GROUP BY | |1 |260652||1 | PX COORDINATOR | |1 |260626||2 | EXCHANGE OUT DISTR |:EX10001 |1 |260625||3 | MERGE GROUP BY | |1 |260625||4 | NESTED-LOOP JOIN | |139 |260598||5 | EXCHANGE IN DISTR | |135 |255373||6 | EXCHANGE OUT DISTR (BC2HOST)|:EX10000 |135 |255312||7 | PX BLOCK ITERATOR | |135 |255312||8 | TABLE SCAN |gz\_health\_code\_day3check2\_data|135 |255312||9 | PX PARTITION ITERATOR | |1 |39 ||10| TABLE SCAN |nucleic\_acid\_testing |1 |39 |=======================================================================================Outputs & filters: -----------------------------------*--* 0 - output([T\_FUN\_COUNT\_SUM(T\_FUN\_COUNT(\*))]), filter(nil),  group(nil), agg\_func([T\_FUN\_COUNT\_SUM(T\_FUN\_COUNT(\*))]) 1 - output([T\_FUN\_COUNT(\*)]), filter(nil) 2 - output([T\_FUN\_COUNT(\*)]), filter(nil), dop=108 3 - output([T\_FUN\_COUNT(\*)]), filter(nil),  group(nil), agg\_func([T\_FUN\_COUNT(\*)]) 4 - output([1]), filter(nil),  conds(nil), nl\_params\_([gz\_health\_code\_day3check2\_data.id\_card]) 5 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter(nil) 6 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter(nil), dop=108 7 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter(nil) 8 - output([gz\_health\_code\_day3check2\_data.id\_card]), filter([gz\_health\_code\_day3check2\_data.status = 1], [gz\_health\_code\_day3check2\_data.scan\_time < ?], [(T\_OP\_IS\_NOT, gz\_health\_code\_day3check2\_data.id\_card, NULL, 0)]),  access([gz\_health\_code\_day3check2\_data.status], [gz\_health\_code\_day3check2\_data.id\_card], [gz\_health\_code\_day3check2\_data.scan\_time]), partitions(p0) 9 - output([1]), filter(nil) 10 - output([1]), filter([nucleic\_acid\_testing.datetime > ?], [(T\_OP\_IS\_NOT, nucleic\_acid\_testing.id\_card, NULL, 0)]),  access([nucleic\_acid\_testing.id\_card], [nucleic\_acid\_testing.datetime]), partitions(p[0-215]) |