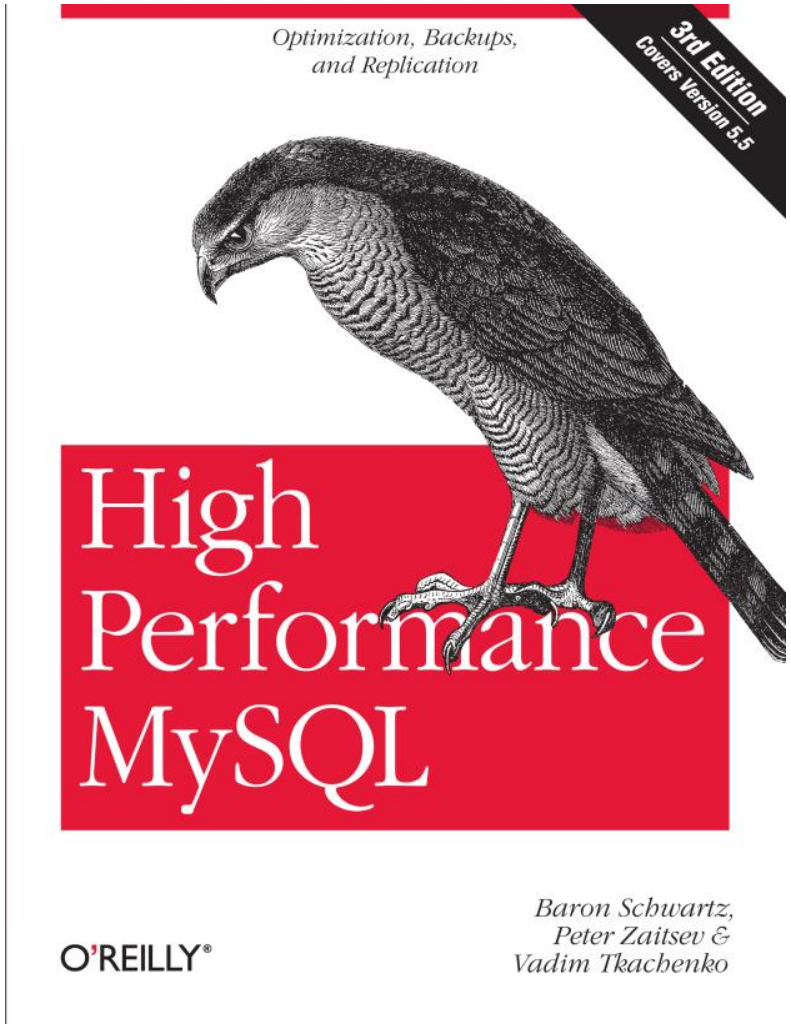
MySQL调优案例分享

2020-04-27

书籍推荐



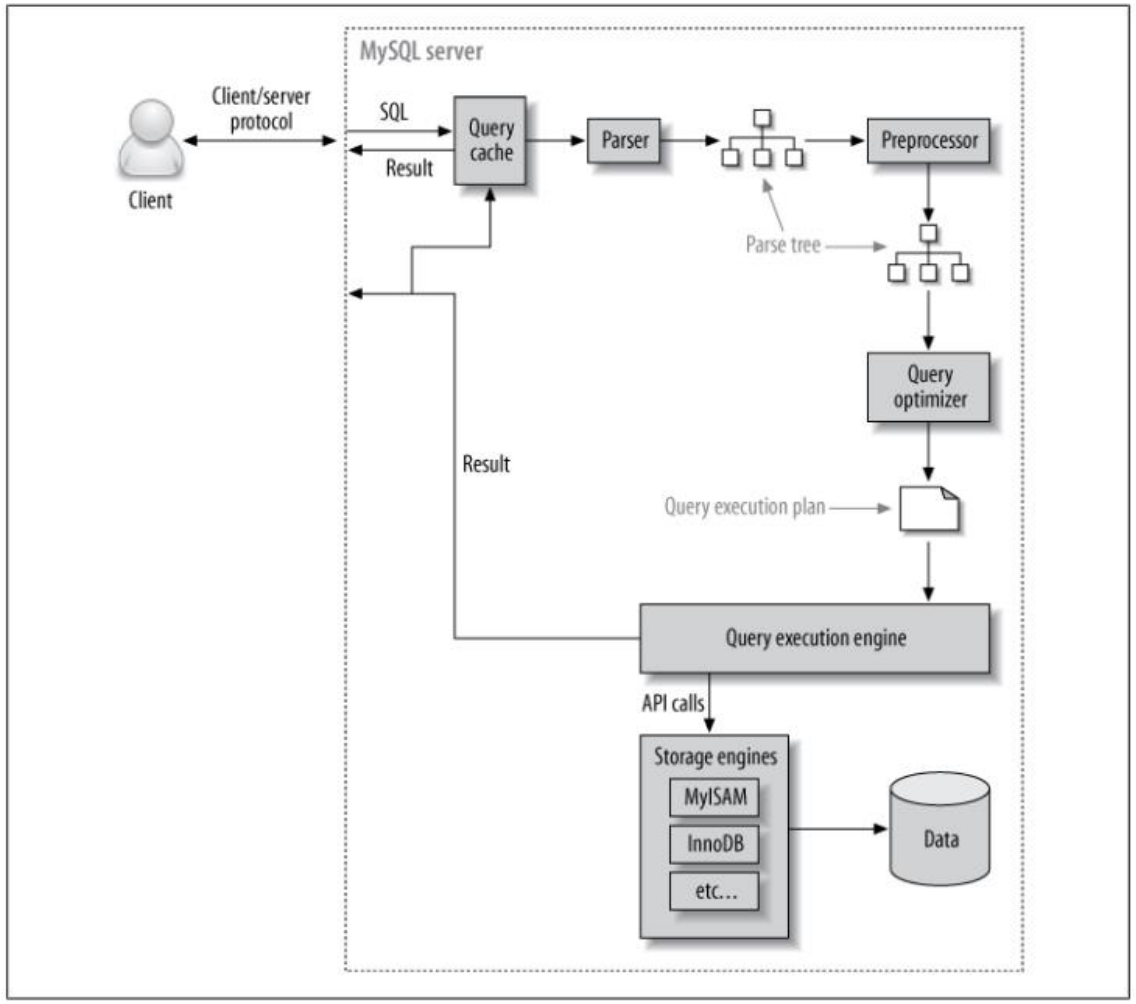
《高性能MySQL》

《MySQL 是怎样运行的：从根儿上理解 MySQL》

目录
CONTENTS

一、概念知识

1.1 SQL执行过程



SQL语句执行过程

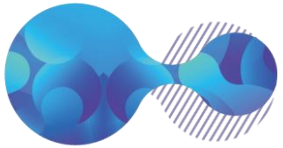
（1）客户端发送一条查询SQL给服务器。

（2）服务器先检查查询缓存，如果命中缓存，则立刻返回存储在缓存中的结果，否则进入下一阶段。

（3）服务器进行SQL解析、预处理、再由优化器生成对应的执行计划。

（4）MySQL根据优化器生成的执行计划，调用存储引擎的API来执行查询。

（5）将结果返回给客户端。



1.2 数据库的本质

```
-rw-r-----. 1 mysql mysql        61 Aug 16  2019 db.opt
-rw-r-----. 1 mysql mysql      8840 Aug 16  2019 gsms_abandon_frame_ticket.frm
-rw-r-----. 1 mysql mysql     98304 Oct 10  2019 gsms_abandon_frame_ticket.ibd
-rw-r-----. 1 mysql mysql      9320 Mar 31 09:24 gsms_account_carrier_price.frm
-rw-r-----. 1 mysql mysql    360448 Mar 31 10:01 gsms_account_carrier_price.ibd
-rw-r-----. 1 mysql mysql     21275 Aug 16  2019 gsms_alarm_enterprise.frm
-rw-r-----. 1 mysql mysql    131072 Aug 16  2019 gsms_alarm_enterprise.ibd
-rw-r-----. 1 mysql mysql     13304 Aug 16  2019 gsms_announcement.frm
-rw-r-----. 1 mysql mysql     98304 Aug 16  2019 gsms_announcement.ibd
-rw-r-----. 1 mysql mysql      8677 Aug 16  2019 gsms_app_enabler.frm
-rw-r-----. 1 mysql mysql     98304 Apr 27 11:38 gsms_app_enabler.ibd
-rw-r-----. 1 mysql mysql     40251 Aug 16  2019 gsms_app.frm
-rw-r-----. 1 mysql mysql    163840 Apr 27 11:38 gsms_app.ibd
-rw-r-----. 1 mysql mysql      8956 Aug 16  2019 gsms_applydetailchange_history.frm
-rw-r-----. 1 mysql mysql     98304 Aug 16  2019 gsms_applydetailchange_history.ibd
-rw-r-----. 1 mysql mysql      1298 Aug 16  2019 gsms_app.TRG
-rw-r-----. 1 mysql mysql      8947 Aug 16  2019 gsms_audit_record.frm
-rw-r-----. 1 mysql mysql    131072 Aug 16  2019 gsms_audit_record.ibd
-rw-r-----. 1 mysql mysql      8700 Aug 16  2019 gsms_biz_application.frm
-rw-r-----. 1 mysql mysql     98304 Aug 16  2019 gsms_biz_application.ibd
-rw-r-----. 1 mysql mysql      8734 Aug 16  2019 gsms_biz_app_template_detail.frm
-rw-r-----. 1 mysql mysql    114688 Aug 16  2019 gsms_biz_app_template_detail.ibd
-rw-r-----. 1 mysql mysql      8712 Aug 16  2019 gsms_biz_app_template.frm
-rw-r-----. 1 mysql mysql    114688 Aug 16  2019 gsms_biz_app_template.ibd
-rw-r-----. 1 mysql mysql      8845 Aug 16  2019 gsms_biz_enable_permission.frm
-rw-r-----. 1 mysql mysql    114688 Apr 22 14:47 gsms_biz_enable_permission.ibd
-rw-r-----. 1 mysql mysql      8859 Aug 16  2019 gsms_biztype_specnum.frm
-rw-r-----. 1 mysql mysql    131072 Nov  6 16:31 gsms_biztype_specnum.ibd
-rw-r-----. 1 mysql mysql      1370 Aug 16  2019 gsms_biztype_specnum.TRG
-rw-r-----. 1 mysql mysql      9412 Aug 16  2019 gsms_business_type.frm
-rw-r-----. 1 mysql mysql    114688 Mar 31 09:41 gsms_business_type.ibd
-rw-r-----. 1 mysql mysql      1358 Aug 16  2019 gsms_business_type.TRG
-rw-r-----. 1 mysql mysql      9057 Sep  3  2019 gsms_call_records.frm
-rw-r-----. 1 mysql mysql    425984 Apr 17 17:24 gsms_call_records.ibd
-rw-r-----. 1 mysql mysql      9180 Aug 16  2019 gsms_capital_account_book.frm
-rw-r-----. 1 mysql mysql    114688 Apr 22 17:49 gsms_capital_account_book.ibd
-rw-r-----. 1 mysql mysql      9262 Aug 16  2019 gsms_capital_account.frm
-rw-r-----. 1 mysql mysql    131072 Apr 22 17:49 gsms_capital_account.ibd
-rw-r-----. 1 mysql mysql     17966 Aug 16  2019 gsms_carrier_channel.frm
```

MySQL数据库数据目录下的文件

.frm文件：用来保存每个数据表的元数据(meta)信息，包括表结构的定义等，与存储引擎无关。

.ibd文件：单表表空间文件，每个表使用一个表空间文件，存放用户数据库表数据、索引。

.TRG文件：触发器定义文件。

.MYD文件：MyISAM数据库标文件。

...

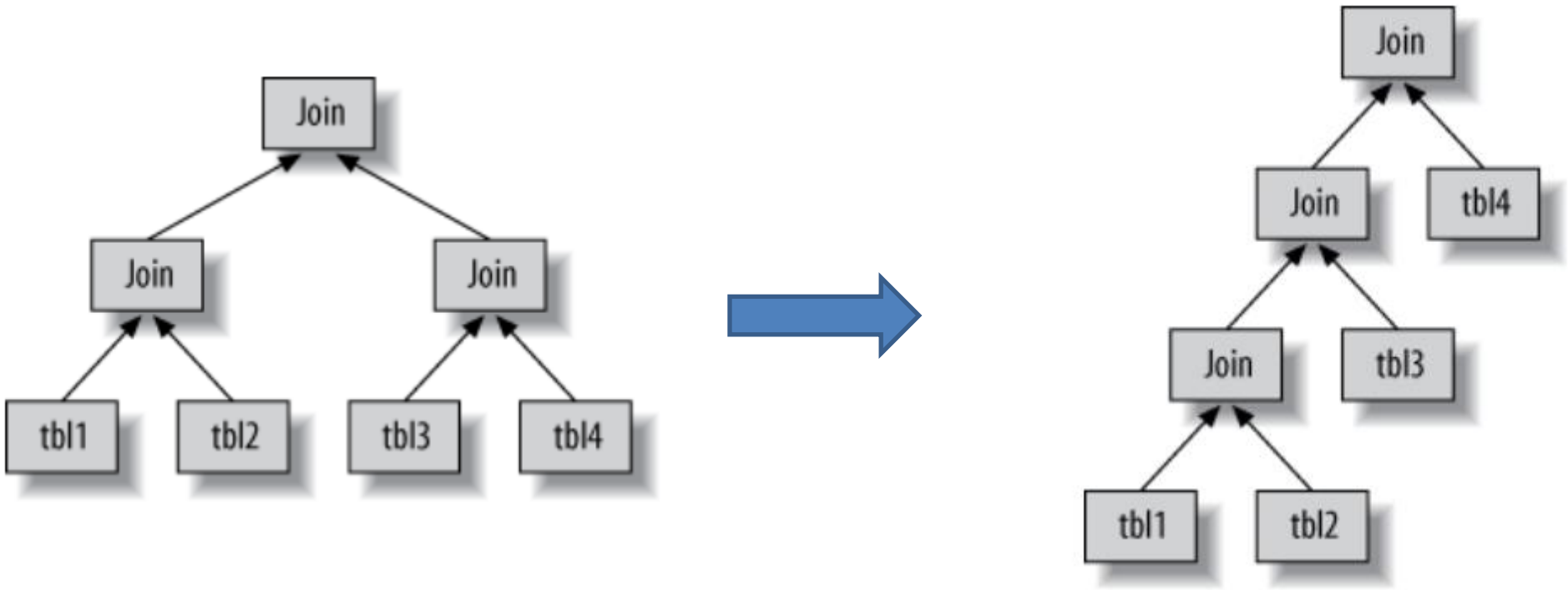
1.3 全表扫描

01 使用null做为判断条件

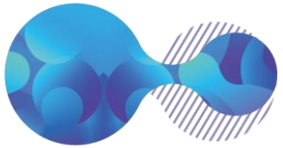
02 使用or做为连接条件

03 对字段有操作时

1.4 连接查询



MySQL多表关联是一棵左侧深度优先树



连接中的where子句与on子句

- `WHERE` 子句中的过滤条件

  `WHERE` 子句中的过滤条件就是我们平时见的那种，不论是内连接还是外连接，凡是不符合 `WHERE` 子句中的过滤条件的记录都不会被加入最后的结果集。

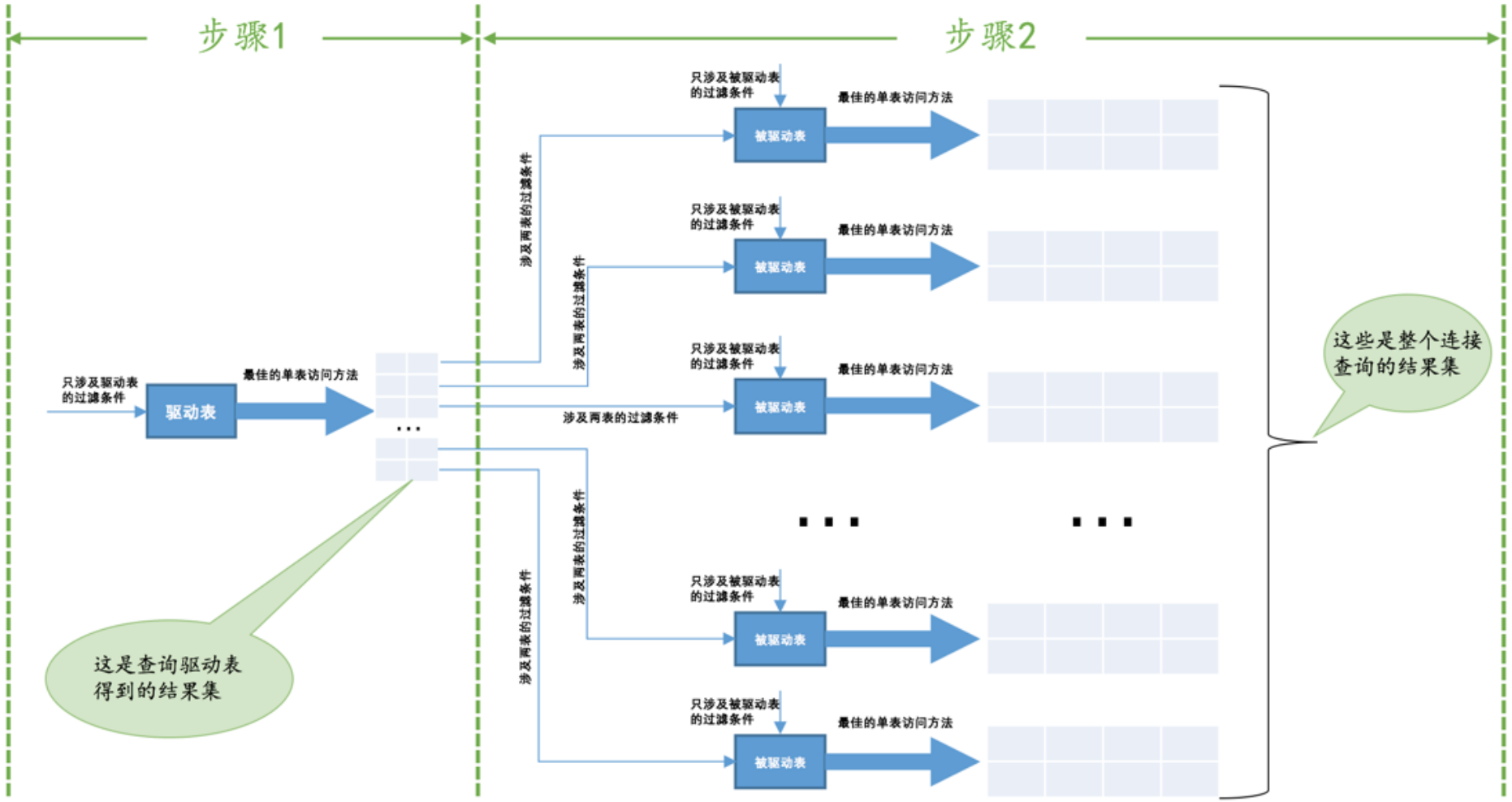
- `ON` 子句中的过滤条件

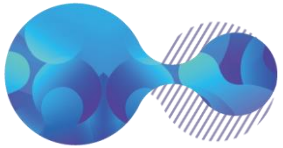
  对于外连接的驱动表的记录来说，如果无法在被驱动表中找到匹配 `ON` 子句中的过滤条件的记录，那么该记录仍然会被加入到结果集中，对应的被驱动表记录的各个字段使用 `NULL` 值填充。

  需要注意的是，这个 `ON` 子句是专门为外连接驱动表中的记录在被驱动表找不到匹配记录时应不应该把该记录加入结果集这个场景下提出的，所以如果把 `ON` 子句放到内连接中， `MySQL` 会把它和 `WHERE` 子句一样对待，也就是说：内连接中的WHERE子句和ON子句是等价的。

一般情况下，我们都把只涉及单表的过滤条件放到 `WHERE` 子句中，把涉及两表的过滤条件都放到 `ON` 子句中，我们也一般把放到 `ON` 子句中的过滤条件也称之为 `连接条件` 。

连接过程图


步骤1
步骤2
只涉及驱动表的过滤条件
驱动表
最佳的单表访问方法
涉及两表的过滤条件
只涉及被驱动表的过滤条件
被驱动表
最佳的单表访问方法
这是查询驱动表得到的结果集
这些是整个连接查询的结果集



1.5 count(*)和count(1)

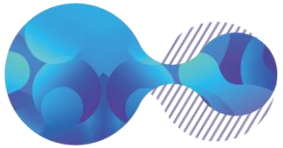
InnoDB 引擎

InnoDB handles SELECT COUNT(*) and SELECT COUNT(1) operations in the same way. There is no performance difference.
InnoDB 引擎下，这两个其实没有性能上的差异。

MyISAM引擎

For MyISAM tables, COUNT(*) is optimized to return very quickly if the SELECT retrieves from one table, no other columns are retrieved, and there is no WHERE clause. For example: This optimization only applies to MyISAM tables, because an exact row count is stored for this storage engine and can be accessed very quickly. COUNT(1) is only subject to the same optimization if the first column is defined as NOT NULL.

MyISAM引擎对count(*)进行了优化，可以快速得到结果. 但是如果使用count(1),那么如果当第一列的结果不为空的时候才有这种优化，所以在MyISAM引擎下 如果第一列为空的话count(1)的查询效率是没有count(*)高。



MySQL对count()、min()和max()的优化

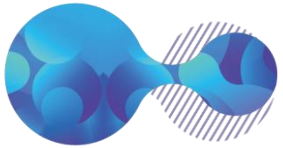
优化 COUNT()、MIN() 和 MAX()

索引和列是否可为空通常可以帮助 MySQL 优化这类表达式。例如，要找到某一列的最小值，只需要查询对应 B-Tree 索引最左端的记录，MySQL 可以直接获取索引的第一行记录。在优化器生成执行计划的时候就可以利用这一点，在 B-Tree 索引中，优化器会将这个表达式作为一个常数对待。类似的，如果要查找一个最大值，也只需读取 B-Tree 索引的最后一条记录。如果 MySQL 使用了这种类型的优化，那么在 EXPLAIN 中就可以看到“Select tables optimized away”。从字面意思可以看出，它表示优化器已经从执行计划中移除了该表，并以一个常数取而代之。

类似的，没有任何 WHERE 条件的 COUNT(*) 查询通常也可以使用存储引擎提供的一些优化（例如，MyISAM 维护了一个变量来存放数据表的行数）。

目录
CONTENTS

二、项目优化案例



2. 项目优化案例

从MySQL的慢日志中发现一条执行耗时为12秒的SQL

```
     duct_record b on b.msg_frame_id = a.frame_id left join gsms_deduct_record_detail_sms_0429 c on c.report_
9827 # Time: 2020-04-29T02:46:54.303834Z
9828 # User@Host: root[root] @  [172.16.0.99]  Id:  9009
9829 # Query_time: 12.585251  Lock_time: 0.000110 Rows_sent: 100  Rows_examined: 3855797
9830 SET timestamp=1588128414;
9831 select distinct a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.
     duct_record b on b.msg_frame_id = a.frame_id left join gsms_deduct_record_detail_sms_0429 c on c.report_
9832 # Time: 2020-04-29T02:47:08.001702Z
9833 # User@Host: root[root] @  [172.16.0.99]  Id:  9009
```

SQL语句如下：

```
SELECT DISTINCT
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.report_id IS NULL AND a.done_time < '2020-04-30 00:00:00'
ORDER BY a.id LIMIT 100;
```

2.1 数据库表关系

SQL功能：根据返回的状态报告生成详细扣款和返款记录，每3分钟执行一次

id	channel_id	msg_id	smsc_sequence	dest_phone
1	4	42989444	4	18829010002
2	4	42989445	5	18829010003
3	4	42989446	6	18829010004
4	4	42989447	7	18829010005
5	4	42989448	8	18829010006
6	4	42989449	9	18829010007
7	4	429894410	10	18829010008
8	4	429894411	11	18829010009
9	4	429894412	12	18829010010
10	4	429894413	13	18829010011
11	4	429894414	14	18829010012
12	4	429894415	15	18829010013
13	4	429894416	16	18829010014
14	4	429894417	17	18829010015
15	4	429894418	18	18829010016
16	4	429894419	19	18829010017
17	4	429894420	20	18829010018
18	4	429894421	21	18829010019
19	4	429894422	22	18829010020
20	4	429894423	23	18829010021
21	4	429894424	24	18829010022
22	4	429894425	25	18829010023
23	4	429894426	26	18829010024
24	4	429894427	27	18829010025
25	4	429894428	28	18829010026
26	4	429894429	29	18829010027
27	4	429894430	30	18829010028

gsms_statereport_0429

report_id	price	biz_enabler	app_id	capital_account_id
1	0.0300	110	33	2
2	0.0300	110	33	2
3	0.0300	110	33	2
4	0.0300	110	33	2
5	0.0300	110	33	2
6	0.0300	110	33	2
7	0.0300	110	33	2
8	0.0300	110	33	2
9	0.0300	110	33	2
10	0.0300	110	33	2
11	0.0300	110	33	2
12	0.0300	110	33	2
13	0.0300	110	33	2
14	0.0300	110	33	2
15	0.0300	110	33	2
16	0.0300	110	33	2
17	0.0300	110	33	2
18	0.0300	110	33	2
19	0.0300	110	33	2
20	0.0300	110	33	2
21	0.0300	110	33	2
22	0.0300	110	33	2
23	0.0300	110	33	2
24	0.0300	110	33	2
25	0.0300	110	33	2
26	0.0300	110	33	2

gsms_deduct_record_detail_sms_0429

2.2 执行耗时

```
EXPLAIN SELECT SQL_NO_CACHE  DISTINCT
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL
ORDER BY a.id LIMIT 100;
```

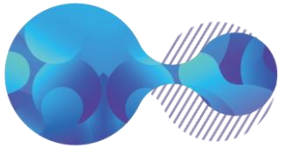
1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

id	channel_id	msg_id	smsc_sequence	dest_phone	post_time	result	app_id	user_id	frame_id	t
739621	4	429921231707	394917	18829241321	2020-04-29 08:12:37	0	33	8	117295	
739622	4	429921231708	394918	18829241322	2020-04-29 08:12:37	0	33	8	117295	
739623	4	429921231709	394919	18829241323	2020-04-29 08:12:37	0	33	8	117295	
739624	4	429921231710	394920	18829241324	2020-04-29 08:12:37	0	33	8	117295	
739625	4	429921231711	394921	18829241325	2020-04-29 08:12:37	0	33	8	117295	
739626	4	429921231712	394922	18829241326	2020-04-29 08:12:37	0	33	8	117295	
739627	4	429921231713	394923	18829241327	2020-04-29 08:12:37	0	33	8	117295	
739628	4	429921231714	394924	18829241328	2020-04-29 08:12:37	0	33	8	117295	
739629	4	429921231715	394925	18829241329	2020-04-29 08:12:37	0	33	8	117295	
739630	4	429921231716	394926	18829241330	2020-04-29 08:12:37	0	33	8	117295	
739631	4	429921231717	394927	18829241331	2020-04-29 08:12:37	0	33	8	117295	
739632	4	429921231718	394928	18829241332	2020-04-29 08:12:37	0	33	8	117295	
739633	4	429921231719	394929	18829241333	2020-04-29 08:12:37	0	33	8	117295	
739634	4	429921231720	394930	18829241334	2020-04-29 08:12:37	0	33	8	117295	
739635	4	429921231721	394931	18829241335	2020-04-29 08:12:37	0	33	8	117295	
739636	4	429921231722	394932	18829241336	2020-04-29 08:12:37	0	33	8	117295	
739637	4	429921231723	394933	18829241337	2020-04-29 08:12:37	0	33	8	117295	
739638	4	429921231724	394934	18829241338	2020-04-29 08:12:37	0	33	8	117295	

ECT SQL_NO_CACHE distinct a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id, a.frame_id,a.ticket_id,a.is_norm

执行：11.541 sec　　总数：11.542 sec　　100 行　　Ln 4, Col 30



2.3 执行计划

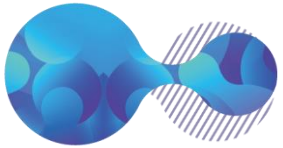
自动完成：[Tab]-> 下一个标签，[Ctrl+Space]-> 列出所有标签，[Ctrl+Enter]-> 列出匹配标签

```
1   EXPLAIN SELECT SQL_NO_CACHE DISTINCT
2   a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
3   a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
4   FROM gsms_statereport_0429 a
5   JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
6   LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
7   WHERE c.report_id IS NULL
8   ORDER BY a.id LIMIT 100;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读) 限制行 第一行: 0 行数

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	a	(NULL)	index	ix_report_0429_cmid	PRIMARY	8	(NULL)	81	100.00	Using temporary
1	SIMPLE	c	(NULL)	eq_ref	PRIMARY	PRIMARY	8	cmp_gsms_2.0.a.id	1	100.00	Using where; Not exists; Using index
1	SIMPLE	b	(NULL)	ref	ix_deduct_frame_id	ix_deduct_frame_id	9	cmp_gsms_2.0.a.frame_id	1	100.00	(NULL)



2.4 尝试第一步优化：筛选不必要的行

```
EXPLAIN SELECT SQL_NO_CACHE  DISTINCT
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL AND a.id > 1450000
ORDER BY a.id LIMIT 100;
```

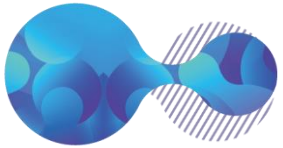
1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

（只读）

id	channel_id	msg_id	smsc_sequence	dest_phone	post_time	result	app_id	user_id	frame_id	ticket_id	i
1450001	4	42910203557057	1105617	18829452024	2020-04-29 08:20:34	0	33	8	117580	3157017	
1450002	4	42910203557058	1105618	18829452025	2020-04-29 08:20:34	0	33	8	117580	3157018	
1450003	4	42910203557059	1105619	18829452026	2020-04-29 08:20:34	0	33	8	117580	3157019	
1450004	4	42910203557060	1105620	18829452027	2020-04-29 08:20:34	0	33	8	117580	3157020	
1450005	4	42910203557061	1105621	18829452028	2020-04-29 08:20:34	0	33	8	117580	3157021	
1450006	4	42910203557062	1105622	18829452029	2020-04-29 08:20:34	0	33	8	117580	3157022	
1450007	4	42910203557063	1105623	18829452030	2020-04-29 08:20:34	0	33	8	117580	3157023	
1450008	4	42910203557064	1105624	18829452031	2020-04-29 08:20:34	0	33	8	117580	3157024	
1450009	4	42910203557065	1105625	18829452032	2020-04-29 08:20:34	0	33	8	117580	3157025	
1450010	4	42910203557066	1105626	18829452033	2020-04-29 08:20:34	0	33	8	117580	3157026	
1450011	4	42910203557067	1105627	18829452034	2020-04-29 08:20:34	0	33	8	117580	3157027	
1450012	4	42910203557068	1105628	18829452035	2020-04-29 08:20:34	0	33	8	117580	3157028	
1450013	4	42910203557069	1105629	18829452036	2020-04-29 08:20:34	0	33	8	117580	3157029	
1450014	4	42910203557070	1105630	18829452037	2020-04-29 08:20:34	0	33	8	117580	3157030	
1450015	4	42910203557071	1105631	18829452038	2020-04-29 08:20:34	0	33	8	117580	3157031	
1450016	4	42910203557072	1105632	18829452039	2020-04-29 08:20:34	0	33	8	117580	3157032	
1450017	4	42910203557073	1105633	18829452040	2020-04-29 08:20:34	0	33	8	117580	3157033	
1450018	4	42910203557074	1105634	18829452041	2020-04-29 08:20:34	0	33	8	117580	3157034	

ECT SQL_NO_CACHE distinct a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id, a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.pric

执行：0.809 sec　　总数：0.810 sec　　100 行　　Ln 4, Col 30　　连接：3



2.5 执行计划

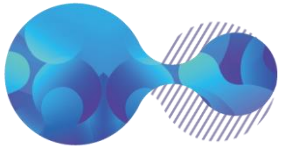
```
EXPLAIN SELECT SQL_NO_CACHE  DISTINCT
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL AND a.id > 1450000
ORDER BY a.id LIMIT 100;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读) 限制行 第一行: 0 行数: 1000

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	a	(NULL)	range	PRIMARY,ix_report_0429_cmid	PRIMARY	8	(NULL)	113266	100.00	Using where; Using temporary
1	SIMPLE	c	(NULL)	eq_ref	PRIMARY	PRIMARY	8	cmp_gsms_2.0.a.id	1	100.00	Using where; Not exists; Using index
1	SIMPLE	b	(NULL)	ref	ix_deduct_frame_id	ix_deduct_frame_id	9	cmp_gsms_2.0.a.frame_id	1	100.00	(NULL)

使用where条件筛选掉了大部分记录，执行耗时变小



2.6 调整SQL

```
EXPLAIN SELECT SQL_NO_CACHE DISTINCT
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL AND a.id > (SELECT MAX(report_id) FROM gsms_deduct_record_detail_sms_0429)
ORDER BY a.id LIMIT 100;
```

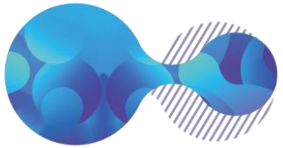
1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

id	channel_id	msg_id	smsc_sequence	dest_phone	post_time	result	app_id	user_id	frame_id	ticket_id	is
745121	4	429921507207	400417	18829246821	2020-04-29 08:12:37	0	33	8	117297	2451817	
745122	4	429921507208	400418	18829246822	2020-04-29 08:12:37	0	33	8	117297	2451818	
745123	4	429921507209	400419	18829246823	2020-04-29 08:12:37	0	33	8	117297	2451819	
745124	4	429921507210	400420	18829246824	2020-04-29 08:12:37	0	33	8	117297	2451820	
745125	4	429921507211	400421	18829246825	2020-04-29 08:12:37	0	33	8	117297	2451821	
745126	4	429921507212	400422	18829246826	2020-04-29 08:12:37	0	33	8	117297	2451822	
745127	4	429921507213	400423	18829246827	2020-04-29 08:12:37	0	33	8	117297	2451823	
745128	4	429921507214	400424	18829246828	2020-04-29 08:12:37	0	33	8	117297	2451824	
745129	4	429921507215	400425	18829246829	2020-04-29 08:12:37	0	33	8	117297	2451825	
745130	4	429921507216	400426	18829246830	2020-04-29 08:12:37	0	33	8	117297	2451826	
745131	4	429921507217	400427	18829246831	2020-04-29 08:12:37	0	33	8	117297	2451827	
745132	4	429921507218	400428	18829246832	2020-04-29 08:12:37	0	33	8	117297	2451828	
745133	4	429921507219	400429	18829246833	2020-04-29 08:12:37	0	33	8	117297	2451829	
745134	4	429921507220	400430	18829246834	2020-04-29 08:12:37	0	33	8	117297	2451830	
745135	4	429921507221	400431	18829246835	2020-04-29 08:12:37	0	33	8	117297	2451831	
745136	4	429921507222	400432	18829246836	2020-04-29 08:12:37	0	33	8	117297	2451832	
745137	4	429921507223	400433	18829246837	2020-04-29 08:12:37	0	33	8	117297	2451833	
745138	4	429921507224	400434	18829246838	2020-04-29 08:12:37	0	33	8	117297	2451834	

CT SQL_NO_CACHE distinct a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id, a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.pr

执行：9.530 sec　　总数：9.534 sec　　100 行　　Ln 3, Col 51　　连接：3



2.7 执行计划

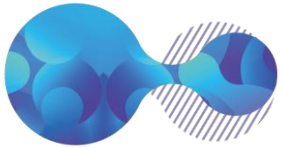
```
1   EXPLAIN SELECT SQL_NO_CACHE  DISTINCT
2   a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
3   a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
4   FROM gsms_statereport_0429 a
5   JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
6   LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
7   WHERE a.id > (SELECT MAX(report_id) FROM gsms_deduct_record_detail_sms_0429)
8   ORDER BY a.id LIMIT 100;
9
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读) | 限制行 第一行: 0 行数:

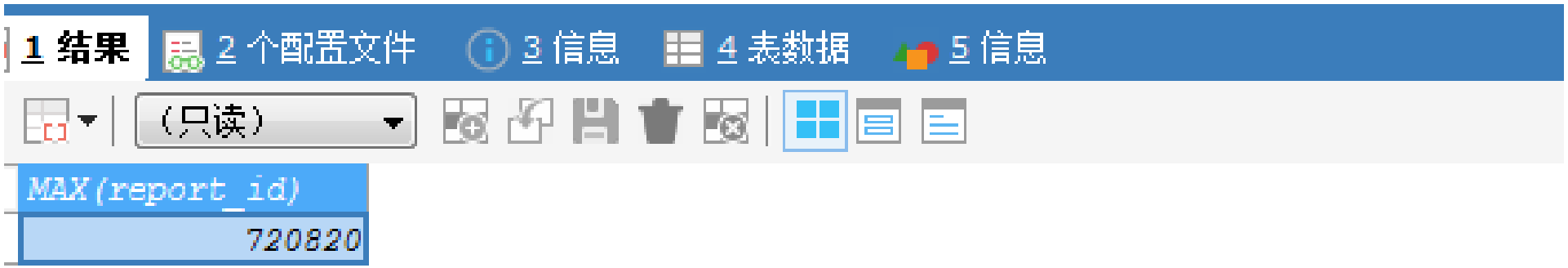
id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	PRIMARY	a	(NULL)	range	PRIMARY,ix_report_0429_cmid	PRIMARY	8	(NULL)	741903	100.00	Using where; Using temporary
1	PRIMARY	c	(NULL)	eq_ref	PRIMARY	PRIMARY	8	cmp_gsms_2.0.a.id	1	100.00	Using index
1	PRIMARY	b	(NULL)	ref	ix_deduct_frame_id	ix_deduct_frame_id	9	cmp_gsms_2.0.a.frame_id	1	100.00	(NULL)
2	SUBQUERY	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	(NULL)	Select tables optimized away

根据执行计划显示，上述子查询在优化阶段已经获取到值，但执行耗时仍然很长

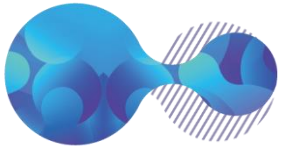


2.8 原因分析1

```
SELECT MAX(report_id) FROM gsms_deduct_record_detail_sms_0429
```



查询执行过程中，where条件并不是总能过滤掉大部分记录



2.9 原因分析2

再次查看最初的执行计划

```
EXPLAIN SELECT SQL_NO_CACHE DISTINCT
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL
ORDER BY a.id LIMIT 100;
```

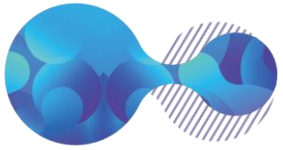
1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读) | 限制行 第一行: 0 行数:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	a	(NULL)	index	ix_report_0429_cmid	PRIMARY	8	(NULL)	81	100.00	Using temporary
1	SIMPLE	c	(NULL)	eq_ref	PRIMARY	PRIMARY	8	cmp_gsms_2.0.a.id	1	100.00	Using where; Not exists; Using index
1	SIMPLE	b	(NULL)	ref	ix_deduct_frame_id	ix_deduct_frame_id	9	cmp_gsms_2.0.a.frame_id	1	100.00	(NULL)

Extra： Using temporary,用临时表保存中间结果

常用于GROUP BY 和 ORDER BY操作中，一般看到它说明查询需要优化了，就算避免不了临时表的使用也要尽量避免硬盘临时表的使用



2.10 再次优化：避免使用临时表

```
EXPLAIN SELECT SQL_NO_CACHE
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL
ORDER BY a.id LIMIT 100;
```

去除DISTINCT关键字

1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

(只读)

id	channel_id	msg_id	smsc_sequence	dest_phone	post_time	result	app_id	user_id	frame_id	ticket_id	is
735421	4	429921263042	390717	18829237121	2020-04-29 08:12:37	0	33	8	117293	2442117	
735422	4	429921263043	390718	18829237122	2020-04-29 08:12:37	0	33	8	117293	2442118	
735423	4	429921263044	390719	18829237123	2020-04-29 08:12:37	0	33	8	117293	2442119	
735424	4	429921263045	390720	18829237124	2020-04-29 08:12:37	0	33	8	117293	2442120	
735425	4	429921263046	390721	18829237125	2020-04-29 08:12:37	0	33	8	117293	2442121	
735426	4	429921263047	390722	18829237126	2020-04-29 08:12:37	0	33	8	117293	2442122	
735427	4	429921263048	390723	18829237127	2020-04-29 08:12:37	0	33	8	117293	2442123	
735428	4	429921263049	390724	18829237128	2020-04-29 08:12:37	0	33	8	117293	2442124	
735429	4	429921263050	390725	18829237129	2020-04-29 08:12:37	0	33	8	117293	2442125	
735430	4	429921263051	390726	18829237130	2020-04-29 08:12:37	0	33	8	117293	2442126	
735431	4	429921263052	390727	18829237131	2020-04-29 08:12:37	0	33	8	117293	2442127	
735432	4	429921263053	390728	18829237132	2020-04-29 08:12:37	0	33	8	117293	2442128	
735433	4	429921263054	390729	18829237133	2020-04-29 08:12:37	0	33	8	117293	2442129	
735434	4	429921263055	390730	18829237134	2020-04-29 08:12:37	0	33	8	117293	2442130	
735435	4	429921263056	390731	18829237135	2020-04-29 08:12:37	0	33	8	117293	2442131	
735436	4	429921263057	390732	18829237136	2020-04-29 08:12:37	0	33	8	117293	2442132	
735437	4	429921263058	390733	18829237137	2020-04-29 08:12:37	0	33	8	117293	2442133	
735438	4	429921263059	390734	18829237138	2020-04-29 08:12:37	0	33	8	117293	2442134	

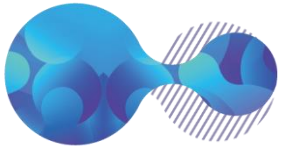
CT SQL_NO_CACHE a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id, a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.rep

执行：2.480 sec　总数：2.480 sec　100 行　Ln 7, Col 29　连接：3

4.11 gsms_statereport表信息

Create Table

```
CREATE TABLE `gsms_statereport_0429` (
  `id` bigint(20) NOT NULL AUTO_INCREMENT,
  `channel_id` int(11) NOT NULL DEFAULT '0' COMMENT '通道ID',
  `msg_id` varchar(50) NOT NULL DEFAULT '' COMMENT '消息ID',
  `smsc_sequence` bigint(20) NOT NULL COMMENT '序列号',
  `dest_phone` varchar(30) NOT NULL COMMENT '目标号码',
  `post_time` datetime NOT NULL COMMENT '客户提交到即信网关时间',
  `submit_time` datetime NOT NULL COMMENT '运营商网关发送到用户手机时间',
  `done_time` datetime NOT NULL COMMENT '处理完成时间',
  `result` int(11) NOT NULL COMMENT '处理结果 DELIVERED:0 EXPIRED:1 UNDELIVERABLE:2 REJECTED:3 UNKNOWN:4 DELE
  `origin_result` varchar(30) DEFAULT NULL COMMENT '运营商原始结果',
  `app_id` int(11) NOT NULL DEFAULT '0' COMMENT '应用ID',
  `user_id` int(11) NOT NULL COMMENT '用户ID',
  `frame_id` bigint(20) NOT NULL COMMENT '信息帧ID',
  `ticket_id` bigint(20) NOT NULL COMMENT '信息话单 ID, frame_id, ticket_id 组合唯一键',
  `pack_uuid` varchar(36) NOT NULL DEFAULT '' COMMENT '信息包 UUID',
  `pack_id` bigint(20) NOT NULL DEFAULT '0' COMMENT '信息包ID',
  `enterprise_id` int(11) NOT NULL DEFAULT '0' COMMENT '企业ID',
  `custom_msg_id` varchar(50) DEFAULT NULL COMMENT '自定义消息ID',
  `is_normal` int(11) DEFAULT '0' COMMENT '是否正常数据0正常1人工',
  `is_history` varchar(50) DEFAULT NULL COMMENT '改变状态前原始数据',
  `reserve` varchar(50) DEFAULT NULL COMMENT '扩展保留字段',
  PRIMARY KEY (`id`) USING BTREE,
  UNIQUE KEY `ix_report_0429_cmid` (`frame_id`,`ticket_id`) USING BTREE,
  KEY `ix_report_0429_ptid` (`post_time`,`id`) USING BTREE,
  KEY `ix_report_0429_uid` (`user_id`,`id`) USING BTREE
) ENGINE=InnoDB AUTO_INCREMENT=1507918 DEFAULT CHARSET=utf8 ROW_FORMAT=DYNAMIC
```



2.12 执行计划

```
EXPLAIN SELECT SQL_NO_CACHE
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL
ORDER BY a.id LIMIT 100;
```

1 结果 | 2 信息 | 3 表数据 | 4 信息

(只读) 限制行 第一行: 0 行数:

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	a	(NULL)	index	ix_report_0429_cmid	PRIMARY	8	(NULL)	81	100.00	(NULL)
1	SIMPLE	c	(NULL)	eq_ref	PRIMARY	PRIMARY	8	cmp_gsms_2.0.a.id	1	100.00	Using where; Not exists; Using index
1	SIMPLE	b	(NULL)	ref	ix_deduct_frame_id	ix_deduct_frame_id	9	cmp_gsms_2.0.a.frame_id	1	100.00	(NULL)

避免了临时表的使用，查询耗时降低

2.13 再次进行优化调整

```
EXPLAIN SELECT SQL_NO_CACHE
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.report_id
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE c.`report_id` IS NULL AND a.id > (SELECT MAX(report_id) FROM gsms_deduct_record_detail_sms_0429)
ORDER BY a.id LIMIT 100;
```

1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

（只读）

id	channel_id	msg_id	smsc_sequence	dest_phone	post_time	result	app_id	user_id	frame_id	ticket_id
737721	4	4299211365342	393017	18829239421	2020-04-29 08:12:37	0	33	8	117294	2444417
737722	4	4299211365343	393018	18829239422	2020-04-29 08:12:37	0	33	8	117294	2444418
737723	4	4299211365344	393019	18829239423	2020-04-29 08:12:37	0	33	8	117294	2444419
737724	4	4299211365345	393020	18829239424	2020-04-29 08:12:37	0	33	8	117294	2444420
737725	4	4299211365346	393021	18829239425	2020-04-29 08:12:37	0	33	8	117294	2444421
737726	4	4299211365347	393022	18829239426	2020-04-29 08:12:37	0	33	8	117294	2444422
737727	4	4299211365348	393023	18829239427	2020-04-29 08:12:37	0	33	8	117294	2444423
737728	4	4299211365349	393024	18829239428	2020-04-29 08:12:37	0	33	8	117294	2444424
737729	4	4299211365350	393025	18829239429	2020-04-29 08:12:37	0	33	8	117294	2444425
737730	4	4299211365351	393026	18829239430	2020-04-29 08:12:37	0	33	8	117294	2444426
737731	4	4299211365352	393027	18829239431	2020-04-29 08:12:37	0	33	8	117294	2444427
737732	4	4299211365353	393028	18829239432	2020-04-29 08:12:37	0	33	8	117294	2444428
737733	4	4299211365354	393029	18829239433	2020-04-29 08:12:37	0	33	8	117294	2444429
737734	4	4299211365355	393030	18829239434	2020-04-29 08:12:37	0	33	8	117294	2444430
737735	4	4299211365356	393031	18829239435	2020-04-29 08:12:37	0	33	8	117294	2444431
737736	4	4299211365357	393032	18829239436	2020-04-29 08:12:37	0	33	8	117294	2444432
737737	4	4299211365358	393033	18829239437	2020-04-29 08:12:37	0	33	8	117294	2444433
737738	4	4299211365359	393034	18829239438	2020-04-29 08:12:37	0	33	8	117294	2444434

ECT SQL_NO_CACHE a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id, a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price ,c.

执行：0.003 sec　　总数：0.003 sec　　100 行　　Ln 3, Col 37　　连接：3

最终优化结果

```
EXPLAIN SELECT SQL_NO_CACHE
a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id,
a.frame_id,a.ticket_id,a.is_normal,a.reserve,b.price
FROM gsms_statereport_0429 a
JOIN gsms_deduct_record b ON b.msg_frame_id = a.frame_id
# LEFT JOIN gsms_deduct_record_detail_sms_0429 c ON c.report_id = a.id
WHERE  a.id > (SELECT MAX(report_id) FROM gsms_deduct_record_detail_sms_0429)
ORDER BY a.id LIMIT 100;
```

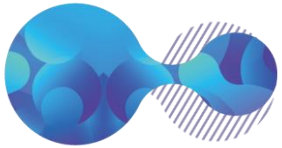
1 结果 | 2 个配置文件 | 3 信息 | 4 表数据 | 5 信息

（只读）

id	channel_id	msg_id	smsc_sequence	dest_phone	post_time	result	app_id	user_id	frame_id	t
1083921	4	429950418506	739391	18829085797	2020-04-29 08:20:34	0	33	8	117434	
1083922	4	429950418507	739392	18829085798	2020-04-29 08:20:34	0	33	8	117434	
1083923	4	429950418508	739393	18829085799	2020-04-29 08:20:34	0	33	8	117434	
1083924	4	429950418509	739394	18829085800	2020-04-29 08:20:34	0	33	8	117434	
1083925	4	429950418510	739395	18829085801	2020-04-29 08:20:34	0	33	8	117434	
1083926	4	429950418511	739396	18829085802	2020-04-29 08:20:34	0	33	8	117434	
1083927	4	429950418512	739397	18829085803	2020-04-29 08:20:34	0	33	8	117434	
1083928	4	429950418513	739398	18829085804	2020-04-29 08:20:34	0	33	8	117434	
1083929	4	429950418514	739399	18829085805	2020-04-29 08:20:34	0	33	8	117434	
1083930	4	429950418515	739400	18829085806	2020-04-29 08:20:34	0	33	8	117434	
1083931	4	429950418516	739401	18829085807	2020-04-29 08:20:34	0	33	8	117434	
1083932	4	429950418517	739402	18829085808	2020-04-29 08:20:34	0	33	8	117434	
1083933	4	429950418518	739403	18829085809	2020-04-29 08:20:34	0	33	8	117434	
1083934	4	429950418519	739404	18829085810	2020-04-29 08:20:34	0	33	8	117434	
1083935	4	429950418520	739405	18829085811	2020-04-29 08:20:34	0	33	8	117434	

ECT SQL_NO_CACHE a.id,a.channel_id,a.msg_id,a.smsc_sequence,a.dest_phone,a.post_time,a.result,a.app_id,a.user_id, a.frame_id,a.ticket_id,a.is_normal,a.rese

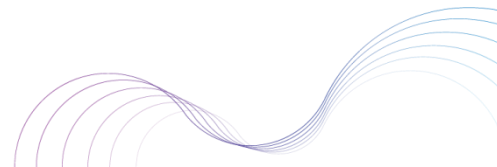
执行：0.001 sec　　总数：0.002 sec　　100 行　　Ln 10, Col 1



总结

与其坐而论道，不如起而行之。

学习一门技术需要系统全面的学习。



谢谢观看！