MySQL Index Condition Pushdown(ICP)性能优化方法实例

这篇文章主要介绍了MySQL Index Condition Pushdown(ICP)性能优化方法实例,本文讲解了概念介绍、原理、实践案例、案例分析、ICP的使用限制等内容,需要的朋友可以参考下

一 概念介绍

Index Condition Pushdown (ICP)是MySQL 5.6 版本中的新特性,是一种在存储引擎层使用索引过滤数据的一种优化方式。

a 当关闭ICP时,index 仅仅是data access 的一种访问方式，存储引擎通过索引回表获取的数据会传递到MySQL Server 层进行where条件过滤。

b 当打开ICP时,如果部分where条件能使用索引中的字段,MySQL Server 会把这部分下推到引擎层,可以利用index过滤的where条件在存储引擎层进行数据过滤,而非将所有通过index access的结果传递到MySQL server层进行where过滤.

优化效果:ICP能减少引擎层访问基表的次数和MySQL Server 访问存储引擎的次数,减少io次数，提高查询语句性能。

二 原理

Index Condition Pushdown is not used:

1 Get the next row, first by reading the index tuple, and then by using the index tuple to locate and read the full table row.
2 Test the part of the WHERE condition that applies to this table. Accept or reject the row based on the test result.
Index Condition Pushdown is used
1 Get the next row s index tuple (but not the full table row).
2 Test the part of the WHERE condition that applies to this table and can be checked using only index columns.
If the condition is not satisfied, proceed to the index tuple for the next

row.
  3 If the condition is satisfied, use the index tuple to locate and read the full table row.
  4 est the remaining part of the WHERE condition that applies to this table. Accept or reject the row based on the test result.

三 实践案例

a 环境准备
  数据库版本 5.6.16
  关闭缓存

```
  set query_cache_size=0;
  set query_cache_type=OFF;
```

  测试数据下载地址
b 当开启ICP时

```
mysql> SET profiling = 1;
Query OK, 0 rows affected, 1 warning (0.00 sec)
mysql> select * from employees where first_name='Anneke' and
last_name like '%sig' ;
+--------+------------+------------+-----------+--------+------------+
| emp_no | birth_date | first_name | last_name | gender | hire_date |
+--------+------------+------------+-----------+--------+------------+
| 10006  | 1953-04-20 | Anneke     | Preusig   | F      | 1989-06-02 |
+--------+------------+------------+-----------+--------+------------+
1 row in set (0.00 sec)
mysql> show profiles;
+----------+------------+-----------------------------------------
-----------------------------+
| Query_ID | Duration   | Query                                    |
+----------+------------+-----------------------------------------
-----------------------------+
| 1        | 0.00060275 | select * from employees where
first_name='Anneke' and last_name like '%sig'    |
```

```
+---------+----------+-----------------------------------------
-----------------------------+
3 rows in set, 1 warning (0.00 sec)
```

此时情况下根据MySQL的最左前缀原则, first_name 可以使用索引，last_name采用了like 模糊查询，不能使用索引。
c 关闭ICP

```
mysql> set optimizer_switch='index_condition_pushdown=off';
Query OK, 0 rows affected (0.00 sec)
mysql> SET profiling = 1;
Query OK, 0 rows affected, 1 warning (0.00 sec)
mysql> select * from employees where first_name='Anneke' and
last_name like '%sig' ;
+--------+-----------+-----------+----------+-------+-----------+
| emp_no | birth_date | first_name | last_name | gender | hire_date |
+--------+-----------+-----------+----------+-------+-----------+
| 10006  | 1953-04-20 | Anneke     | Preusig   | F      | 1989-06-02 |
+--------+-----------+-----------+----------+-------+-----------+
1 row in set (0.00 sec)
mysql> SET profiling = 0;
Query OK, 0 rows affected, 1 warning (0.00 sec)
mysql> show profiles;
+---------+-----------+-----------------------------------------
-----------------------------+
| Query_ID | Duration    | Query                                                |
+---------+-----------+-----------------------------------------
-----------------------------+
| 2        | 0.00097000 | select * from employees where
first_name='Anneke' and last_name like '%sig'     |
+---------+-----------+-----------------------------------------
-----------------------------+
6 rows in set, 1 warning (0.00 sec)
```

当开启ICP时 查询在sending data环节时间消耗是 0.000189s

```
mysql> show profile cpu,block io for query 1;
+----------------------+----------+----------+------------+--------------+---------------+
| Status               | Duration | CPU_user | CPU_system | Block_ops_in | Block_ops_out |
+----------------------+----------+----------+------------+--------------+---------------+
| starting             | 0.000094 | 0.000000 | 0.000000   | 0            | 0             |
| checking permissions | 0.000011 | 0.000000 | 0.000000   | 0            | 0             |
| Opening tables       | 0.000025 | 0.000000 | 0.000000   | 0            | 0             |
| init                 | 0.000044 | 0.000000 | 0.000000   | 0            | 0             |
| System lock          | 0.000014 | 0.000000 | 0.000000   | 0            | 0             |
| optimizing           | 0.000021 | 0.000000 | 0.000000   | 0            | 0             |
| statistics           | 0.000093 | 0.000000 | 0.000000   | 0            | 0             |
| preparing            | 0.000024 | 0.000000 | 0.000000   | 0            | 0             |
| executing            | 0.000006 | 0.000000 | 0.000000   | 0            | 0             |
| Sending data         | 0.000189 | 0.000000 | 0.000000   | 0            | 0             |
| end                  | 0.000019 | 0.000000 | 0.000000   | 0            | 0             |
| query end            | 0.000012 | 0.000000 | 0.000000   | 0            | 0             |
| closing tables       | 0.000013 | 0.000000 | 0.000000   | 0            | 0             |
| freeing items        | 0.000034 | 0.000000 | 0.000000   | 0            | 0             |
| cleaning up          | 0.000007 | 0.000000 | 0.000000   | 0            | 0             |
+----------------------+----------+----------+------------+--------------+---------------+
```

```
--+--------------+
15 rows in set, 1 warning (0.00 sec)
```

当关闭ICP时 查询在sending data环节时间消耗是 0.000735s

```
mysql> show profile cpu,block io for query 2;
+--------------------+---------+---------+-----------+-----------
--+--------------+
| Status             | Duration | CPU_user | CPU_system | Block_ops_in |
Block_ops_out |
+--------------------+---------+---------+-----------+-----------
--+--------------+
| starting           | 0.000045 | 0.000000 | 0.000000  | 0          | 0            |
| checking permissions | 0.000007 | 0.000000 | 0.000000  | 0          |
0            |
| Opening tables     | 0.000015 | 0.000000 | 0.000000  | 0          |
0            |
| init               | 0.000024 | 0.000000 | 0.000000  | 0          | 0            |
| System lock        | 0.000009 | 0.000000 | 0.000000  | 0          |
0            |
| optimizing         | 0.000012 | 0.000000 | 0.000000  | 0          | 0
|
| statistics         | 0.000049 | 0.000000 | 0.000000  | 0          | 0            |
| preparing          | 0.000016 | 0.000000 | 0.000000  | 0          | 0
|
| executing          | 0.000005 | 0.000000 | 0.000000  | 0          | 0
|
| Sending data       | 0.000735 | 0.001000 | 0.000000  | 0          |
0            |
| end                | 0.000008 | 0.000000 | 0.000000  | 0          | 0            |
| query end          | 0.000008 | 0.000000 | 0.000000  | 0          |
0            |
| closing tables     | 0.000009 | 0.000000 | 0.000000  | 0          |
0            |
| freeing items      | 0.000023 | 0.000000 | 0.000000  | 0          |
```

0 |
| cleaning up | 0.000007 | 0.000000 | 0.000000 | 0 | 0 |
+--------------------+----------+----------+----------+----------+--------------+
15 rows in set, 1 warning (0.00 sec)

从上面的profile 可以看出ICP 开启时整个sql 执行时间是未开启的2/3, sending data 环节的时间消耗前者仅是后者的1/4。
ICP 开启时的执行计划 含有 Using index condition 标示， 表示优化器使用了ICP对数据访问进行优化。

mysql> explain select * from employees where first_name='Anneke' and last_name like '%nta' ;
+----+-------------+-----------+-------+---------------+---------------+---------+-------+------+----------------------+
| id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra |
+----+-------------+-----------+-------+---------------+---------------+---------+-------+------+----------------------+
| 1 | SIMPLE | employees | ref | idx_emp_fnln | idx_emp_fnln | 44 | const | 224 | Using index condition |
+----+-------------+-----------+-------+---------------+---------------+---------+-------+------+----------------------+
1 row in set (0.00 sec)

ICP 关闭时的执行计划显示use where.

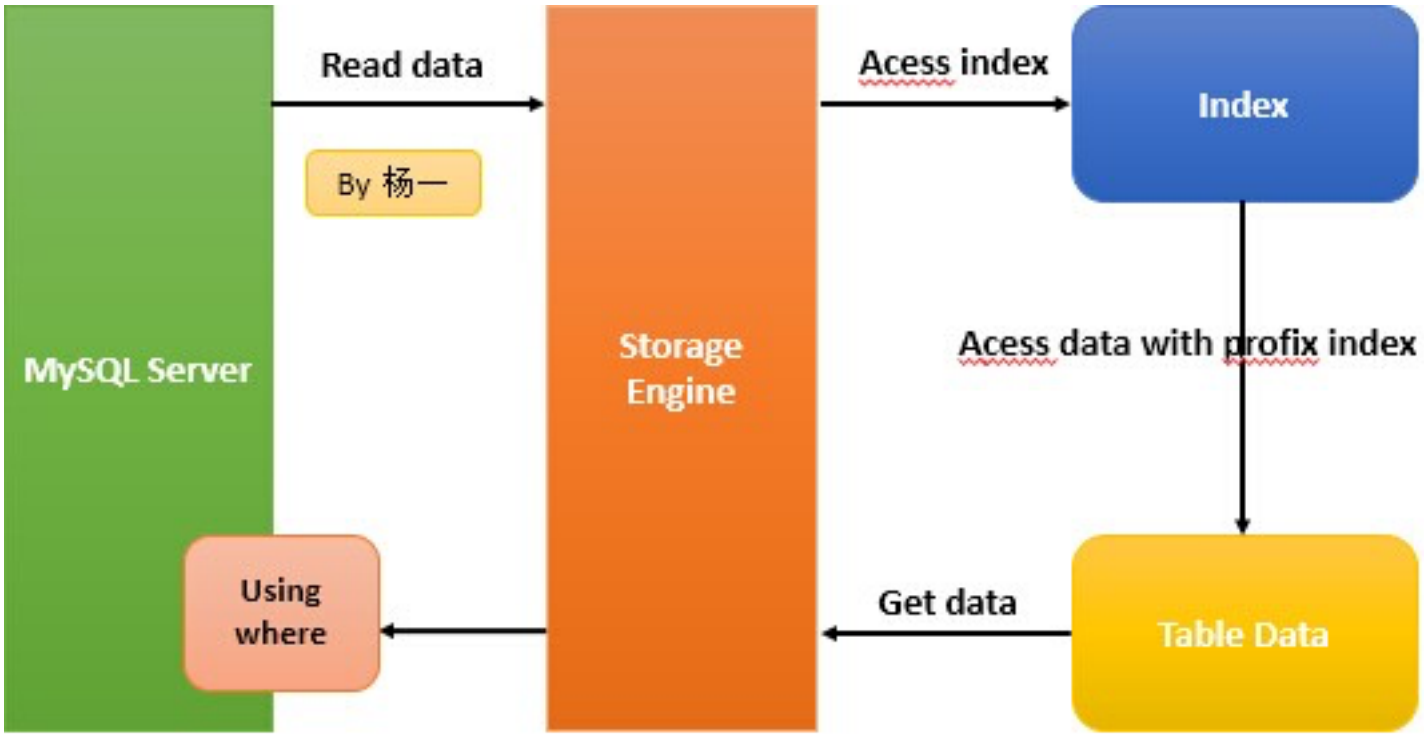
mysql> explain select * from employees where first_name='Anneke' and last_name like '%nta' ;
+----+-------------+-----------+-------+---------------+---------------+---------+-------+------+-------------+
| id | select_type | table | type | possible_keys | key | key_len | ref | rows | Extra |
+----+-------------+-----------+-------+---------------+---------------+---------+-------+------+-------------+
+----+-------------+-----------+-------+---------------+---------------+---------+-------+------+-------------+

| 1 | SIMPLE | employees | ref | idx_emp_fnln | idx_emp_fnln | 44 | const | 224 | Using where |
+----+-------------+----------+-----+--------------+--------------+---------+-------+-----+-------------+
1 row in set (0.00 sec)

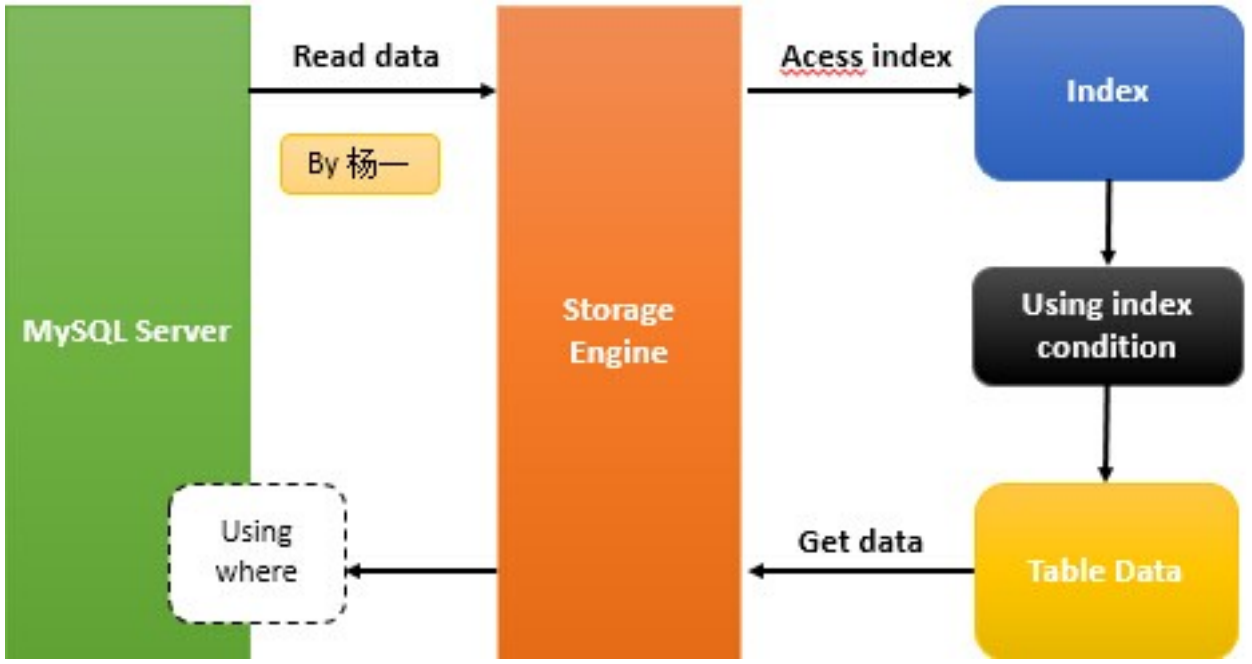
案例分析

以上面的查询为例关闭ICP 时,存储引擎通前缀index first_name 访问表中225条first_name 为Anneke的数据,并在MySQL server层根据last_name like '%sig' 进行过滤
开启ICP 时，last_name 的like '%sig'条件可以通过索引字段last_name 进行过滤，在存储引擎内部通过与where条件的对比，直接过滤掉不符合条件的数据。该过程不回表,只访问符合条件的1条记录并返回给MySQL Server ,有效的减少了io访问和各层之间的交互。

ICP 关闭时 ， 仅仅使用索引作为访问数据的方式。



ICP 开启时 ， MySQL将在存储引擎层 利用索引过滤数据，减少不必要的回表，注意 虚线的using where 表示如果where条件中含有没有被索引的字段，则还是要经过MySQL Server 层过滤。



四 ICP的使用限制

1 当sql需要全表访问时,ICP的优化策略可用于range, ref, eq_ref, ref_or_null 类型的访问数据方法 。
2 支持InnoDB和MyISAM表。
3 ICP只能用于二级索引，不能用于主索引。
4 并非全部where条件都可以用ICP筛选。
　如果where条件的字段不在索引列中,还是要读取整表的记录到server端做where过滤。
5 ICP的加速效果取决于在存储引擎内通过ICP筛选掉的数据的比例。
6 5.6 版本的不支持分表的ICP 功能，5.7 版本的开始支持。
7 当sql 使用覆盖索引时，不支持ICP 优化方法。

```
mysql> explain select * from employees where first_name='Anneke' and last_name='Porenta' ;
+----+-------------+-----------+------+---------------+-------------+---------+-------------+------+-----------------------+
| id | select_type | table     | type | possible_keys | key         | key_len | ref         | rows | Extra                 |
+----+-------------+-----------+------+---------------+-------------+---------+-------------+------+-----------------------+
| 1  | SIMPLE | employees      | ref  | idx_emp_fnln  | idx_emp_fnln | 94      | const,const | 1    | Using index condition |
+----+-------------+-----------+------+---------------+-------------+---------+-------------+------+-----------------------+
1 row in set (0.00 sec)
```

```
mysql> explain select first_name,last_name from employees where
first_name='Anneke' and last_name='Porenta' ;
+----+-------------+----------+------+--------------+-------------
+--------+-------------+------+-----------------------+
| id | select_type | table    | type | possible_keys | key          | key_len |
ref         | rows | Extra                 |
+----+-------------+----------+------+--------------+-------------
+--------+-------------+------+-----------------------+
| 1  | SIMPLE      | employees | ref  | idx_emp_fnln  | idx_emp_fnln | 94      |
const,const | 1    | Using where; Using index |
+----+-------------+----------+------+--------------+-------------
+--------+-------------+------+-----------------------+
1 row in set (0.00 sec)
```

您可能感兴趣的文章:

- mysql服务性能优化—my.cnf_my.ini配置说明详解(16G内存)
- mysql性能优化之索引优化
- 详解MySQL性能优化（一）
- MySQL延迟关联性能优化方法
- MySQL 5.7增强版Semisync Replication性能优化
- MySQL order by性能优化方法实例
- MySQL数据库21条最佳性能优化经验