数据库引论 Project——数据库查询优化

胡旻 13307130086
周吉 13307130227

2015 年 6 月 30 日

Contents

1 实验概述

1.1 主要内容

给定大众点评中的店铺信息，将其分解至满足第三范式。然后在数据集上，针对不同类型的查询语句，进行分析优化。

1.2 分析方式

由于给定的信息数据实在太小，慢查询等优化手段无法体现效果，所以在实验中，主要采用了 explain 对查询语句进行分析，根据分析结果中的 row，对查询的效率进行评判，从而找出一种优化的方式，并以此判断优化的比例。

1.3 实验环境

mysql Ver 14.14 Distrib 5.6.24
Mac OS

2 实验内容

2.1 Task 1

对数据进行分析以后，可以分成 4 个表:

- CITY : cityi, city, province
- ADDRESS : originallatitude, originallongitude, address, area, businessarea, cityi
- Tag : tags, shopid
- SHOP : shopid, name, alias, cityi, phone, hours, avgprice, stars, photos, description, tags, navigation, characteristics, productrating, environmentrating, servicerating, verygoodremarks, goodremarks, commonremarks, badremarks, verybadremarks, recommendeddishes, ischains, groupon, card

根据第三范式的规则，以上拆分满足第三范式。

2.2 Task 2

2.2.1 单表查询

(1) 查询表中的所有字段

explain select * from city;

查询全部 city 中的字段，所以至少读取 59 行。(0.00123325s)

```
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra |
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL |   59 | NULL  |
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
```

无法优化

(2) 查询表中的指定字段

explain select name from city;（0.00054675s）

查询全部 city 中的 name 字段，所以至少读取 59 行。

```
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra |
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL |   59 | NULL  |
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
```

无法优化

(3) 查询表中没有重复的字段（distinct）的使用

explain select distinct province from city;

查询不重名的所有省份（0.04483825s）

```
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra |
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL |   59 | NULL  |
+----+-------------+-------+------+---------------+------+---------+------+------+-------+
```

无法优化

(4) 条件查询各表主键的字段（单值查询或范围查询）

i.explain select * from city where cid=1;

查询 cid 为 1 的城市

```
+----+-------------+-------+-------+---------------+---------+---------+-------+------+-------
| id | select_type | table | type  | possible_keys | key     | key_len | ref   | rows | Extra |
+----+-------------+-------+-------+---------------+---------+---------+-------+------+-------
|  1 | SIMPLE      | city  | const | PRIMARY       | PRIMARY | 4       | const |    1 | NULL  |
+----+-------------+-------+-------+---------------+---------+---------+-------+------+-------
```

主键本身就是一个索引，所以无法优化

ii.explain select * from city where cid>1 and cid<10;

查询 cid 在 1 到 10 之间的城市

```
+----+-------------+-------+-------+---------------+---------+---------+------+------+--------
| id | select_type | table | type  | possible_keys | key     | key_len | ref  | rows | Extra      |
+----+-------------+-------+-------+---------------+---------+---------+------+------+--------
|  1 | SIMPLE      | city  | range | PRIMARY       | PRIMARY | 4       | NULL |    7 | Using where |
+----+-------------+-------+-------+---------------+---------+---------+------+------+--------
```

无需优化

(5) 条件查询各表中普通字段（单值查询或范围查询）

explain select * from city where province=' 浙江';

查询浙江省的全部信息，扫描了 59 行。

```
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra     |
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL |   59 | Using where |
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
```

create index index_province on city(province);

explain select * from city where province=' 浙江';

```
+----+-------------+-------+------+---------------+----------------+---------+-------+------+
| id | select_type | table | type | possible_keys | key            | key_len | ref   | rows | Extra
+----+-------------+-------+------+---------------+----------------+---------+-------+------+
|  1 | SIMPLE      | city  | ref  | index_province | index_province | 33      | const |    6 | Using index
+----+-------------+-------+------+---------------+----------------+---------+-------+------+
```

建立以 province 为关键字的索引，可以看到估计所用的行数大大减少了，只有原来的 1/10。

(6) 一个表中多个字段条件查询（单值查询或范围查询）

explain select sid from shop where good_remarks>20 and very_bad_remarks<10;

查询 shop id，好评大于 20 极差小于 10 的情况。读取行数 979 行。

```
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra     |
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
|  1 | SIMPLE      | shop  | ALL  | NULL          | NULL | NULL    | NULL |  979 | Using where |
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
```

优化：

create index index_goodremarks on shop(good_remarks);

create index index_verybadremarks on shop(very_bad_remarks);

explain select sid from shop where good_remarks>20 and very_bad_remarks<10;

```
+----+-------------+-------+-------+-----------------------------------------+----------------
| id | select_type | table | type  | possible_keys                           | key            | key_len | ref
+----+-------------+-------+-------+-----------------------------------------+----------------
|  1 | SIMPLE      | shop  | range | index_goodremarks,index_verybadremarks | index_goodremarks | 5
+----+-------------+-------+-------+-----------------------------------------+----------------
```

建立了两个索引，大大减少了查询的行数，变成了 1/5。

(7) 用”in”进行条件查询

i. explain select province from city where cid in (’1’, ’2’, ’3’);

查询 city id 为 1,2,3 的省份，因为是主键就有索引。所以只有 3 行查询。

```
+----+-------------+-------+-------+---------------+---------+---------+------+------+-------
| id | select_type | table | type  | possible_keys | key     | key_len | ref  | rows | Extra     |
+----+-------------+-------+-------+---------------+---------+---------+------+------+-------
|  1 | SIMPLE      | city  | range | PRIMARY       | PRIMARY | 4       | NULL |    3 | Using where |
+----+-------------+-------+-------+---------------+---------+---------+------+------+-------
```

无需优化

ii. explain select name from city where province in (’ 浙江’, ’ 广州’);

查询省份是浙江，广州的城市。读取了 59 行（全部数据）。

```
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra     |
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL |   59 | Using where |
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
```

create index index_province on city(province);

explain select name from city where province in (’ 浙江’, ’ 广州’);

```
+----+-------------+-------+-------+----------------+----------------+---------+------+------+
| id | select_type | table | type  | possible_keys  | key            | key_len | ref  | rows | Extra
+----+-------------+-------+-------+----------------+----------------+---------+------+------+
|  1 | SIMPLE      | city  | range | index_province | index_province | 33      | NULL |    7 | Using inde
+----+-------------+-------+-------+----------------+----------------+---------+------+------+
```

优化：以 province 建立索引以后，估计读取行数减少为 1/8。

(8) 一个表中 group by、order by、having 联合条件查询

explain select * from shop where avg_price<20 order by avg_price;

排序 avg_price，找到小于 20 的商铺信息。读取了全部数据。

```
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
|  1 | SIMPLE      | shop  | ALL  | NULL          | NULL | NULL    | NULL |  979 | Using where; Using fileso
+----+-------------+-------+------+---------------+------+---------+------+------+-----------
```

create index price on shop(avg_price);

explain select * from shop where avg_price<20 order by avg_price;

```
+----+-------------+-------+-------+--------------+-------+---------+------+------+----------
| id | select_type | table | type  | possible_keys | key   | key_len | ref  | rows | Extra
+----+-------------+-------+-------+--------------+-------+---------+------+------+----------
|  1 | SIMPLE      | shop  | range | price        | price | 5       | NULL |  233 | Using index condition |
+----+-------------+-------+-------+--------------+-------+---------+------+------+----------
```

优化：建立 avg_price 索引，读取次数减少到了 1/4.

2.2.2 复合查询

(1) 多表联合查询

explain select sid from shop, city, address where shop.address=address.address and address.cid=city.cid and city.cid=1;

合并 shop、city、address 两个表，查找出 cid = 1 的商铺编号。

```
+----+-------------+---------+-------+---------------+---------+---------+-------+------+-----
| id | select_type | table   | type  | possible_keys | key     | key_len | ref   | rows | Extra
+----+-------------+---------+-------+---------------+---------+---------+-------+------+-----
|  1 | SIMPLE      | city    | const | PRIMARY       | PRIMARY | 4       | const |    1 | Using index
|  1 | SIMPLE      | address | ref   | cid           | cid     | 5       | const |  255 | NULL
|  1 | SIMPLE      | shop    | ALL   | NULL          | NULL    | NULL    | NULL  |  961 | Using where; Using j
+----+-------------+---------+-------+---------------+---------+---------+-------+------+-----
```

create index index_address on address(address);

explain select sid from shop, city, address where shop.address=address.address and address.cid=city.cid and city.cid=1;

```
+----+-------------+---------+-------+-------------------+---------------+---------+-----------
| id | select_type | table   | type  | possible_keys     | key           | key_len | ref       | row
+----+-------------+---------+-------+-------------------+---------------+---------+-----------
|  1 | SIMPLE      | city    | const | PRIMARY           | PRIMARY       | 4       | const     |   1 | Us
|  1 | SIMPLE      | shop    | ALL   | NULL              | NULL          | NULL    | NULL      | 961 | Usi
|  1 | SIMPLE      | address | ref   | cid,index_address | index_address | 303     | sqlopt.shop.addres
+----+-------------+---------+-------+-------------------+---------------+---------+-----------
```

优化建立 address 的索引，因为 cid 是主键已经有索引，所以 address 和 city 只需查找一行。

(2) join 查询

explain select sid, name, city from shop, city, address where shop.address=address.address and address.cid=city.cid and avg_price<20;(约 0.0024s)

对于上述查询，sql 会先将 shop,city,address 做笛卡尔积，再从中选择出符合条件的元组

优化：可以将选择操作提前到叉积之前，并且将叉积操作变为连接操作，这样可以减少连接的数量，也避免了大量的叉积操作，优化之后

explain select sid,name,city from (select sid,name,address from shop where avg_price<20) as A inner join city using(cid);(约 0.0015s)

进一步优化，可以对 avg_price 添加索引，使得查询更快。

(3) 存在量词（exists）查询

explain select cid from address where exists (select * from shop, address where shop.address=address.address and shop.avg_price<20);

```
+----+-------------+---------+-------+---------------+---------------+---------+-------------
| id | select_type | table   | type  | possible_keys | key           | key_len | ref             | rows |
+----+-------------+---------+-------+---------------+---------------+---------+-------------
|  1 | PRIMARY     | address | index | NULL          | cid           | 5       | NULL            |  996 | Using
|  2 | SUBQUERY    | shop    | range | price         | price         | 5       | NULL            |  233 | Using
|  2 | SUBQUERY    | address | ref   | index_address | index_address | 303     | sqlopt.shop.address |
+----+-------------+---------+-------+---------------+---------------+---------+-------------
```

添加索引以后发现效率得不到提升，于是，考虑是 exist 本身的问题。

修改查询方式：

explain select cid from address, shop where shop.address=address.address and shop.avg_price<20;

修改为普通方式之后再添加索引就大大提升了效率。

(4) 嵌套子查询（select …from (select …)）

explain select name from (select * from shop where avg_price < 20) as D;

```
+----+-------------+------------+------+---------------+------+---------+------+------+-------
| id | select_type | table      | type | possible_keys | key  | key_len | ref  | rows | Extra     |
+----+-------------+------------+------+---------------+------+---------+------+------+-------
|  1 | PRIMARY     | <derived2> | ALL  | NULL          | NULL | NULL    | NULL |  961 | NULL      |
|  2 | DERIVED     | shop       | ALL  | NULL          | NULL | NULL    | NULL |  961 | Using where |
+----+-------------+------------+------+---------------+------+---------+------+------+-------
```

create index price on shop(avg_price);

explain select name from (select * from shop where avg_price < 20) as D;

```
+----+-------------+------------+-------+---------------+-------+---------+------+------+-----
| id | select_type | table      | type  | possible_keys | key   | key_len | ref  | rows | Extra
+----+-------------+------------+-------+---------------+-------+---------+------+------+-----
|  1 | PRIMARY     | <derived2> | ALL   | NULL          | NULL  | NULL    | NULL |  233 | NULL            |
|  2 | DERIVED     | shop       | range | price         | price | 5       | NULL |  233 | Using index conditi
+----+-------------+------------+-------+---------------+-------+---------+------+------+-----
```

change to:

explain select name from shop where avg_price < 20;

把嵌套的查询改成普通的查询可以更加快。

```
+----+-------------+-------+-------+---------------+-------+---------+------+------+----------
| id | select_type | table | type  | possible_keys | key   | key_len | ref  | rows | Extra
+----+-------------+-------+-------+---------------+-------+---------+------+------+----------
|  1 | SIMPLE      | shop  | range | price         | price | 5       | NULL |  233 | Using index condition |
+----+-------------+-------+-------+---------------+-------+---------+------+------+----------
```

2.2.3 其他查询

(1) 向表中插入记录

explain insert into city values('test', 'test', 11111);

```
+----+-------------+-------+------+---------------+------+---------+------+------+------------
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra       |
+----+-------------+-------+------+---------------+------+---------+------+------+------------
|  1 | SIMPLE      | NULL  | NULL | NULL          | NULL | NULL    | NULL | NULL | No tables used |
+----+-------------+-------+------+---------------+------+---------+------+------+------------
```

无需优化

(2) 向表中删除记录

explain delete from city where name='test';

```
+----+-------------+-------+------+---------------+------+---------+------+------+------------
| id | select_type | table | type | possible_keys | key  | key_len | ref  | rows | Extra      |
+----+-------------+-------+------+---------------+------+---------+------+------+------------
|  1 | SIMPLE      | city  | ALL  | NULL          | NULL | NULL    | NULL |   59 | Using where |
+----+-------------+-------+------+---------------+------+---------+------+------+------------
```

create index index_name on city(name);

添加索引可以让删除时候的查找更加快。

explain delete from city where name='test';

```
+----+-------------+-------+-------+---------------+------------+---------+-------+------+----
| id | select_type | table | type  | possible_keys | key        | key_len | ref   | rows | Extra      |
+----+-------------+-------+-------+---------------+------------+---------+-------+------+----
|  1 | SIMPLE      | city  | range | index_name    | index_name | 303     | const |    1 | Using where |
+----+-------------+-------+-------+---------------+------------+---------+-------+------+----
```

2.3 Task 3

现在的数据库设计中，我们大量使用了 varchar 类型，我们可以将其改为 char，在速度上会有一定的提升，但是由于数据规模实在太小，所以并没有任何明显的变化。另外，还可以对表中的一些数字变量添加索引，这样可以加快查找的速度。

3 实验总结

通过此次 pj 中对查询语句的分析，我们加深了我们对数据库查询操作的理解以及对系统的各种查询优化及其理论的理解。特别是索引操作，对什么时候能用索引，什么时候不用加索引，什么时候索引有效，什么时候索引无效，都有了初步的认识。此次 pj 也遇到了一些问题，比如每一类查询可能出现不同方法的查询优化，这考虑周全很困难。另外，数据库系统优化的智能性也出乎我们的意料。比如在使用查询语句的时候，系统会对查询结果进行评估，有时候使用索引，有时候顺序查找，这与书中 DBMS 系统数据字典等优化相符。另外，本次实验还有些遗憾的地方，比如我们虽然完成了老师布置的任务，但是举的例子不够多，可能考虑的不够周全；还有因为对于每次查询操作的时间分析较少（数据量比较小），所以一些查询优化体现不出来。