1. EXPLAIN语句
2. 概述

MySQL查询优化器在基于成本和规则对查询语句进行了优化后，会生成一个执行计划，展示了执行查询的具体方式，如多表连接的顺序，每个表采用的访问方法。EXPLAIN语句可以让我们查询出某个查询语句的具体执行计划，直接在具体的查询语句之前加上一个EXPLAIN即可。通过EXPLAIN语句，我们可以进一步地思考如何改进自己的查询语句。

1. EXPLAIN语句输出字段介绍
2. table

EXPLAIN语句输出的每条记录都对应着某个单表的访问方法，每条记录的table列代表着该表的表名。

1. id

查询语句一般以SELECT开头，每个SELECT关键词都会被分配一个唯一的id值，如连接查询只有一个SELECT，则两个记录的id值是相同的。而对于子查询，可能有多个SELECT，则不同记录的id是不同的，也有可能优化器将子查询重写为了连接查询，id变为相同的了。对于UNION子句，每个SELECT对应一个id值，但是还有一个记录id为NULL，因为UNION子句的作用是将多个查询的结果集合起来并去重，就会用到临时表，该临时表的id为NULL。

1. select\_type

每一个SELECT关键字代表的小查询都定义了一个名为select\_type的属性，这个属性表示了小查询在整个大查询中扮演的角色。

1. SIMPLE

查询语句中不包含UNION或者子查询的查询都算作SIMPLE类型，连接查询也是SIMPLE。

1. PRIMARY

对于包含子查询的大查询来说，最左边的那个查询就是PRIMARY。

1. UNION

对于UNION和UNION ALL 的大查询来说，除了最左边的那个小查询之外，其余的小查询是UNION。

1. UNION RESULT

使用临时表来完成UNION查询的去重工作，临时表就是UNION RESULT。

1. SUBQUERY

包含子查询的查询语句不能够转为对应的半连接形式，并且子查询是不相关查询，且优化器采用子查询物化的方案执行，该子查询的第一个SELECT代表的那个查询就是SUBQUERY。

1. DEPENDENT SUBQUERY

与上相同，只是子查询是相关子查询。

1. DEPENDENT UNION

包含UNION 或 UNION ALL 的大查询中，如果各个小查询都依赖于外层查询，则除了最左边的，其他的子查询都是DEPENDENT UNION。

1. DERIVED

包含派生表的差啊讯，如果以物化派生表的方式查询，则派生表对应就是DERIVED。

1. MATERIALIZED

子查询物化之后与外层查询进行连接查询，子查询就是MATERIALIZED。

1. type

type列表名对某个表执行查询的访问方法。

1. system

当表中只有一条记录且使用的存储引擎的统计数据时准确的（MyISAM, MEMORY），那么该表的访问方法就是system。

1. const

根据主键或者唯一二级索引列于常数进行等值匹配时，对单表的访问方法就是const。

1. eq\_ref

执行连接查询时，如果被驱动表是通过主键或者不允许存储NULL值的唯一二级索引列等值匹配的方式进行访问的，则对该被驱动表的访问方法就是eq\_ref。

1. ref

通过普通的二级索引列与常量进行等值匹配时。

1. ref\_or\_null

通过普通的二级索引列于常量进行等值匹配，且该索引列的值也可以是NULL。

1. index\_merge

索引合并的方式执行查询。

1. range

使用索引获取某些单点扫描区间的记录，或者获取某些范围扫描区间的记录。

1. index

使用索引覆盖，但需要扫描全部的索引记录。

1. ALL

全表扫描。

1. possible\_keys 和 key

可能使用的索引和实际使用的索引。

1. key\_len

我们希望直接从执行计划中看出形成扫描区间的边界条件是什么，key\_len列通过计算能够直接告诉我们。key\_len值由三部分组成：

1. 该列的时机数据最多占用的存储空间长度，如果形成扫描区间的边界条件使用的列定义为VARCHAR(100)，且使用的字符集为utf8，那么一个字符最多对应3个字节，那么该列最多占用的字节长度为 100 \* 3 = 300
2. 如果该列可以存储NULL值，则再加1个字节
3. 对于变长类型的列来说，都有2个字节的空间存储该列的实际数据占用的存储空间长度

如303就可以判断出有一个列VARCHAR(100)的列，606可以判断出有两个VARCHAR(100)的列。

1. ref

ref列展示的是与索引列进行等值匹配的东西是什么，如一个常数const或者某个列。

1. rows

在查询优化器选择使用全表扫描查询时，rows列代表这个该表的估计行数；

如果使用索引查询，则rows列代表着预计扫描的索引记录行数；

1. filtered

该列代表着查询优化器预测出占多少百分比的记录会满足其余的搜索条件，对于单表查询这个列无意义，更关注在连接查询中驱动表对应的filtered值。

1. Json格式的执行计划

刚才看到的EXPLAIN语句中缺少了成本这个重要属性，我们可以通过查看Json格式的执行计划来查询成本，在EXPLAIN与真正的查询语句之间加上FORMAT=JSON即可。

成本在cost\_info中，

read\_cost = I/O成本 + 检测 rows \* (1-filter)条记录的CPU成本

eval\_cost = 检测rows \* filter条记录的成本

prefix\_cost = read\_cost + eval\_cost (对于单表查询来说)

对于多表查询，prefix\_cost不等于read\_cost和eval\_cost，因为可能会访问多次，总之记住prefix\_cost是整个成本即可，无论是单表查询还是连接查询。