链滴

一条查询 SQL 的执行过程

作者：jianzh5

原文链接：https://ld246.com/article/1590546306076

来源网站：链滴

许可协议：署名-相同方式共享 4.0 国际 (CC BY-SA 4.0)

需要从数据库检索某些符合要求的数据，我们很容易写出 Select A B C FROM T WHERE ID = X 这样的 SQL，那么当我们向数据库发送这样一个请求时，数据库到底做了什么？
我们今天以 MYSQL 为例，揭示一下 MySQL 数据库的查询过程，并让大家对数据库里的一些零有所了解。
<h3 id="MYSQL架构">MYSQL 架构</h3>
<img src="https://ld246.com/images/img-loading.svg" alt="image.png" data-src="https:/b3logfile.com/file/2020/05/image-1799a999.png?imageView2/2/interlace/1/format/jpg"></
>
MySQL 主要可以分为 Server 层和存储引擎层。
Server 层 包括连接器、查询缓存、分析器、优化器、执行器等，所有跨存引擎的功能都在这一层实现，比如存储过程、触发器、视图、函数等，还有个通用的日志模块 binlog 日志模块；
存储引擎层 负责数据的存储和提取。其架构模式是插件式的，支持 InnoD、MyISAM、Memory 等多个存储引擎。现在最常用的存储引擎是 InnoDB（支持事务），它从 My QL5.5.5 版本开始成为默认存储引擎。
<h3 id="连接器">连接器</h3>
连接器主要负责用户登录数据库，进行用户的身份认证，包括校验账户密码，权限等操作。
如果用户密码不对，你就会收到一个"Access denied for user"的错误，然后客户端程序结束执。
如果用户账户密码已通过，连接器会到权限表中查询该用户的所有权限，之后在这个连接里的权逻辑判断都是会依赖此时读取到的权限数据，也就是说，后续只要这个连接不断开，即时管理员修改该用户的权限，该用户也是不受影响的。
<h3 id="查询缓存--Query-cache-">查询缓存（ Query cache）</h3>
客户端与服务端建立连接后，MySQL 在执行查询语句时会先查询缓存，校验这条 SQL 是不是在前执行过。之前执行过的语句及其结果会以 key-value 对的形式，被直接缓存在内存中。key 是查询语句，value 是查询的结果。如果你的查询能够直接在这个缓存中找到 key，那么这个 value 就会被接返回给客户端。如果没有命中，则需要执行后续的操作，完成后也会把结果缓存起来，方便下一次用。
看到这里大家会不会眼前一亮，会不会有这个功能很好要好好利用的冲动。
其实这里并不建议使用查询缓存，查询缓存的失效非常频繁，只要有对一个表的更新，这个表上有的查询缓存都会被清空。因此很可能你费劲地把结果存起来，还没使用就被一个更新全清空了。对更新压力大的数据库来说，查询缓存的命中率会非常低。除非是那种很长时间不会更新的表，比如系配置表，但是这种系统配置我们放在配置平台不好吗？
在 MYSQL8.0 中已经删除了查询缓存这个功能，官方也是认为该功能在实际的应用场景比较少所以干脆直接删掉了。
<h3 id="分析器">分析器</h3>
Mysql 没有命中查询缓存，那么就会进入分析器，分析器主要是用来分析 SQL 语句是来干嘛的分析器主要分为以下两步：

词法分析：一条 SQL 语句有多个字符串组成，首先要提取关键字，比如 se ect，提出查询的表，提出字段名，提出查询条件等等。
语法分析：根据词法分析的结果，语法分析主要就是判断你输入的 SQL 语是否正确，是否符合 MYSQL 语法，如果你的语句不对，就会收到“You have an error in your SQL yntax”的错误提醒。

词法分析程序将整个查询语句分解成各类标志，语法分析根据定义的系统语言将“各类标志”转对 MySQL 有意义的组合。最后系统生成一个语法树(AST)，语法树便是优化器依赖的数据结构。</p

<h3 id="优化器">优化器</h3>
经过了分析器，MySQL 就知道你要做什么了。在开始执行之前，还要先经过优化器的处理。</
>
为什么需要优化器？

优化器中包含了许多复杂的优化技术，这些优化技术往往比最好的程序员握的还要多。系统的自动优化相当于使得所有人都拥有这些优化技术。

优化器可以从数据字典中获取许多统计信息，例如表中的行数、表中的每列的分布情况等。优化器优化器可以考虑百种不同的执行计划，而程序员一般只能考虑有限的几种可；

可以根据这些信息选择有效的执行计划，而用户程序则难以获得这些信息

总之优化器是对语法分析树的形态进行修改，把语法分析树变为查询树，确定执行方案。
<h3 id="执行器">执行器</h3>
MySQL 通过分析器知道了你要做什么，通过优化器知道了该怎么做，于是就进入了执行器阶段开始执行语句。
开始执行的时候，要先校验改用户是否有执行查询的权限，如果没有，就会返回没有权限的错误如果有权限，就会去调用引擎的接口，返回接口执行的结果。
<h2 id="语句分析">语句分析</h2>
我们以下面一条真实 SQL 查询语句来进行分析下 MYSQL 查询的执行过程
<code>select id,name,sex,phoone from user t where t.age='26' and t.account='javadaily' </code>

首先客户端需要连接上数据库，如果账号密码错误直接返回错误信息，如果正确则进入下一步。
在 MYSQL8.0 之前会先去查询缓存中，以这条 SQL 语句作为 key 在内存中查询是否有结果，如有则先判断是否有权限，有权限则返回客户端，否则报错；如果没有从查询缓存命中则进入下一步</l>
通过分析器进行词法分析，提取 sql 语句的关键元素，比如提取上面这个语句是查询 select，提需要查询的表名为 <code>user</code>，需要查询的列为 <code>id,name,sex,phoone</code> 查询条件是 <code>age=26</code> 和 <code>account=javadailly</code>。然后判断这个 sql 语句是否有语法错误，比如关键词是否正确等等，如果检查没问题就执行下一步。
上面的 SQL 有两种执行方案，优化器根据自己的优化算法选择执行效率最高的 a 方案（统计信不准可能导致优化器选择错误的执行方案），确定了优化方案后就开始执行。

<blockquote>
a. 先查询 <code>account=javadaily</code> 的用户，然后判断 <code>age</code> 是否于 26
b. 先找出 <code>age=26</code> 的用户，再查询 <code>account=javadaily</code> 的用户
</blockquote>

进行权限校验，如果有查询权限则调用数据库引擎接口返回执行结果；否则报错。