链滴

mysql explain 执行计划

作者：xiaokedamowang
原文链接：https://ld246.com/article/1628788667974
来源网站：链滴

<h2 id="mysql-explain执行计划">mysql explain 执行计划</h2>
<h3 id="一--什么是执行计划">一. 什么是执行计划</h3>
<blockquote>
执行计划可以模拟优化器执行 sql, 分析查询语句的性能瓶颈
<pre><code class="language-sql highlight-chroma">
在select 前面加上explain就可以查看执行计划
explain SELECT
    u.*,
    r.`name`
FROM
    sys_user u
    JOIN user_role us ON us.user_id = u.id
    JOIN role r ON r.id = us.role_id
WHERE
    u.id < (select avg(id) from sys_user)
</code></pre>
</blockquote>

</code></pre>
<img src="https://ld246.com/images/img-loading.svg" alt="image-20210812231623583" data-src="https://b3logfile.com/file/2021/08/solo-fetchupload-5385492950852491120-ff92fb7.png?imageView2/2/interlace/1/format/jpg">
</blockquote>
<h3 id="二--id-不重要-">二. id(不重要)</h3>
<blockquote>
id 是执行 sql 的顺序, 越大越早执行, id 相同, 从上往下执行, id 为 Null 最后执行
</blockquote>
<h3 id="三--select-type-不重要-">三. select type(不重要)</h3>
<blockquote>
select type 表示是简单还是复杂的查询。

simple：简单查询。查询不包含子查询和 union
primary：复杂查询中最外层的 select
subquery：包含在 select 中的子查询（不在 from 子句中）
derived：包含在 from 子句中的子查询。MySQL 会将结果存放在一个临时表中，也称为派生表
union：在 union 关键字随后的 selelct。

</blockquote>
<h3 id="四--table">四. table</h3>
<blockquote>
使用了哪个表, 有时不是真实的表名, 可能是临时表
</blockquote>
<h3 id="五--type-重要-">五. type(重要)</h3>
<blockquote>
常用的类型有： ALL、index、range、index_merge 、ref、eq_ref、const、syste 、NULL（从左到右，性能从差到好）
一般需要达到 range 及以上

ALL 全表扫描
index 覆盖索引,扫描了全部的索引 并且 select 查询的字段, 在索引里面就可以找到
range 范围查找 >, <, in, between
index_merge 使用了索引合并优化 就是 where 条件 or 的字段都是索引
ref 非唯一索引扫描
eq_ref 唯一索引扫描
const 常量 where 条件指定了写死的值, 并且走了唯一索引或主键
system 表里只有一条记录, 并且是 const 级别, 是 const 的特例
NULL 执行阶段不用再访问表

</blockquote>
<h3 id="六--possible-keys-不重要-">六. possible_keys(不重要)</h3>
<blockquote>
查询可能用到的列, 但并不一定用到
</blockquote>
<h3 id="七--key-重要-">七. key(重要)</h3>
<blockquote>
实际真正用到的索引, 如果为 NULL, 表示没有用到索引
</blockquote>
<h3 id="八--key-len-或许重要-">八. key_len(或许重要)</h3>
<blockquote>
使用的索引长度, 可以根据长度算出组合索引用了哪些列
规则如下:

允许为空 长度 1

字符串 UTF8 每个字符长度为 3 UTF8MB4 每个字符 4 字节

char
<blockquote>
UTF8MB4 编码的 char(20) 允许为 NULL 长度位: 20 * 4 + 1 = 81
</blockquote>

varchar
<blockquote>
字符串保存长度再加 2
UTF8MB4 编码的 varchar(20), 允许为 NULL 长度位: 20 * 4 + 2 + 1 = 83
</blockquote>

数字

tinyint: 1 字节
smallint: 2 字节
int: 4 字节
bigint: 8 字节

时间

date: 3 字节
timestamp: 4 字节
datetime: 8 字节

</blockquote>
<h5 id="索引最大长度是768字节-当字符串过长时--会截取前半段">索引最大长度是 768 字节，当字符串过长时, 会截取前半段</h5>
<h3 id="九--ref-不重要-">九. ref(不重要)</h3>
一般为表关联的列
<h3 id="十--rows-不重要-">十. rows(不重要)</h3>
估算的结果集行数
<h3 id="十一--Extra-比较重要-">十一. Extra(比较重要)</h3>

Using index：使用覆盖索引
Using index condition：查询的列不完全被索引覆盖 (索引下推)
Using where：在查找使用索引的情况下, 需要回表去查询所需的数据
using index & using where: 查找使用了索引, 并且需要的数据都在索引列中能找到, 所以需要回表查询数据

Using temporary：mysql 需要创建一张临时表来处理查询 出现这种情况一般要进行优化, 常见排序或分组查询
Using filesort: 无法使用索引排序, 将使用外部排序, 数据较小时从内存排序, 否则需要在磁盘完排序。
select tables optimized away：使用聚合函数来处理索引的某个字段
Using index for group-by: 使用 group by 或 distinct 时, 分组的字段在索引中
Using join buffer: 表连接没有使用索引, 使用连接缓冲区来存储中间结果
Distinct: 优化了 distinct
