[mysql explain分析](http://www.cnblogs.com/lpfuture/p/5756543.html)

通过explain可以知道mysql是如何处理语句，分析出查询或是表结构的性能瓶颈。通过expalin可以得到：

1. 表的读取顺序

2.表的读取操作的操作类型

3.哪些索引可以使用

4. 哪些索引被实际使用

5.表之间的引用

6.每张表有多少行被优化器查询

explain显示字段

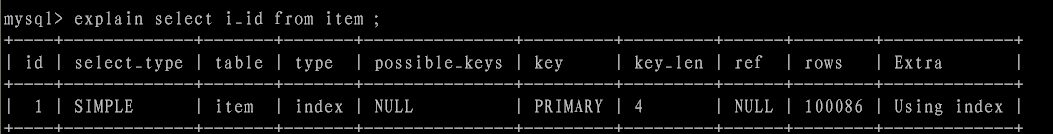
http://up.2cto.com/2013/0723/20130723084340512.png

**1. id :语句的执行顺序标识**

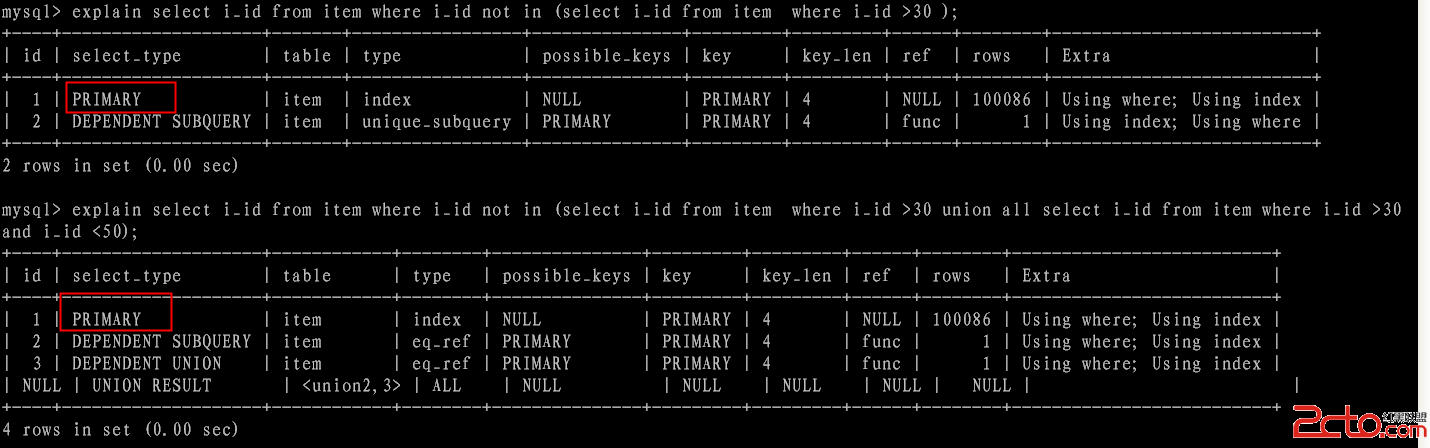
**2. select\_type：使用的查询类型，主要有以下几种查询类型：**

1）simple 简单类型

 语句中没有子查询或union



2）primary 最外层的select，不是主键

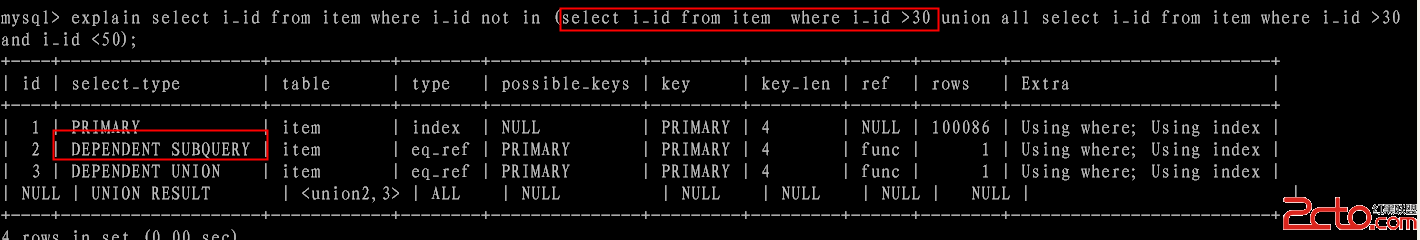


3）union

union是在select 语句中第二个select语句后面所有的select，第一个select 为primary

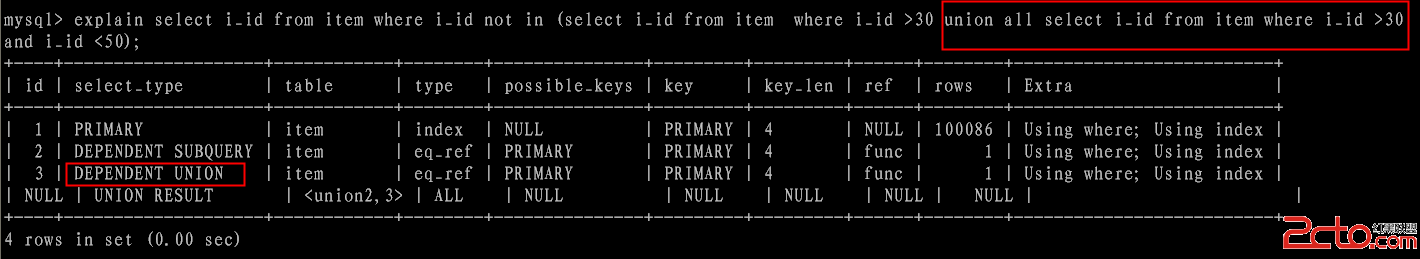
4)dependent subquery

子查询中内层中第一个select语句

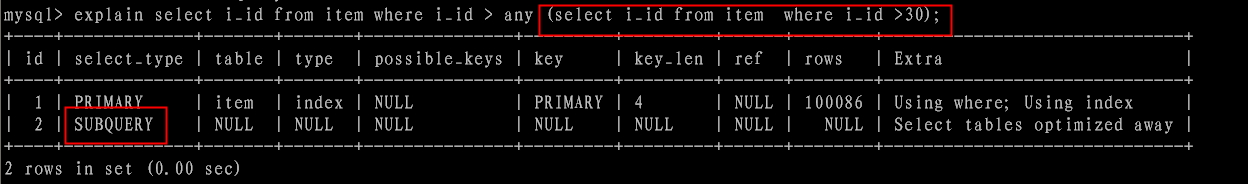


5).  dependent  union

子查询中union且为union中第二个select开始的后面所有select，依赖于外部的结果集。

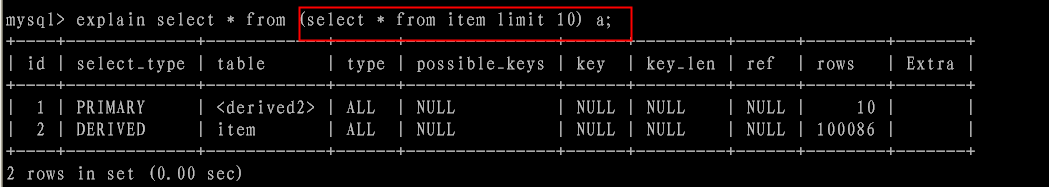


6). SUBQUERY



7).devived

派生表的查询语句

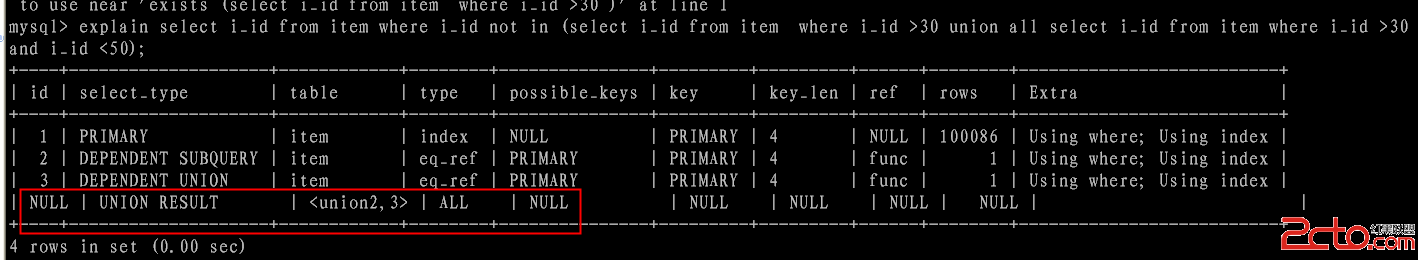


8）. uncacheable subquery

结果集无法缓存的子查询

9). union result

union中合并的结果



**3. table**

显示这一步所访问的[数据库](http://www.2cto.com/database/" \t "_blank)中表的名称

**4. type**

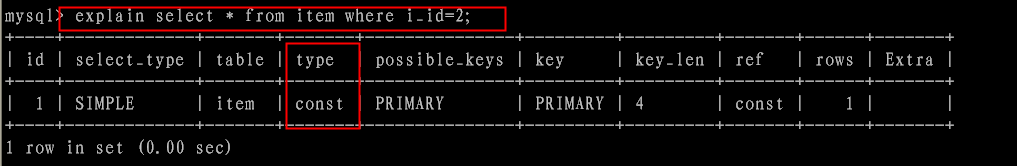
这列很重要,显示了连接使用了哪种类别,有无使用索引。type代表查询执行计划（QEP）中指定的表使用的连接方式。从最好到最差的连接类型为 1.system、2.const、3. eq\_reg、4. ref、5. range、6.index、7. all

1) system

system为const一个特例，**即表中只有一条记录**。

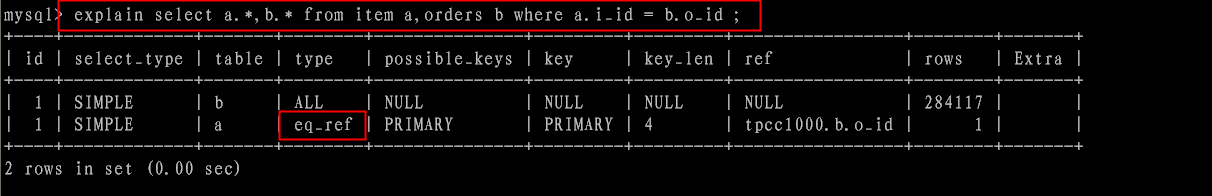
2) const

const是在where条件以常量作为查询条件，**表中最多有一条记录匹配。**由于是常量，所以实际上只需要读一次。



3) eq\_reg

最多只会有一条匹配结果，**一般是通过主键或是唯一索引来访问。**一般会出现在连接查询的语句中。



4）ref

join 语句中被驱动的表索引引用查询。这个值表示所有具有匹配的索引值的行都被用到。

5）range

索引范围扫描

6）index

全索引树被扫描

7) all

全表扫描，效果是最不理想的。

**5. possible\_keys**

查询可以利用的索引，如果没有任何索引可以使用，就会显示成null，这项对内容的优化时索引的调整非常重要。

**6.key**

从possible\_keys中所选择使用的索引

**7. key\_len**

key\_len列显示[mysql](http://www.2cto.com/database/MySQL/" \t "_blank)决定使用的键长度，如果键是null，则长度为null。使用的索引长度，一般越短越好。

**8. ref**

列出的是通过常量const，或是某个表的某个字段来过滤的。

**9.rows**

通过[系统](http://www.2cto.com/os/" \t "_blank)收集到的统计信息，估计出来的结果集记录条数

### 10. filtered 列

　　在mysql5.1里新加的，在使用explain extended时出现。它显示的是针对表里符合条件的记录数的百分比所做的一个悲观估算值。

　　filtered 列给出了一个百分比的值，这个百分比值和rows 列的值相乘，可以估计出那些将要和QEP 中的前一个表进行连接的行的数目。前一个表就是指id 列的值比当前表的id 小的表。这一列只有在EXPLAIN EXTENDED 语句中才会出现。

**11. extra**

 Extra：查询中每一步实现的额外细节信息，主要可能会是以下内容：

 1). Distinct：查找distinct值，所以当mysql找到了第一条匹配的结果后，将停止该值的查询而转为后面其他值的查询； FullscanonNULLkey：子查

询中的一种优化方式，主要在遇到无法通过索引访问null值的使用使用；

2).  ImpossibleWHEREnoticedafterreadingconsttables：MySQLQueryOptimizer通过收集到的统计信息判断出不可能存在结果；

3). Notables：Query语句中使用FROMDUAL或者不包含任何FROM子句；

4). Notexists：在某些左连接中MySQLQueryOptimizer所通过改变原有Query的组成而使用的优化方法，可以部分减少数据访问次数；

5). Rangecheckedforeachrecord(indexmap:N)：通过MySQL官方手册的描述，当MySQLQueryOptimizer没有发现好的可以使用的索引的时候，如果

发现如果来自前面的表的列值已知，可能部分索引可以使用。对前面的表的每个行组合，MySQL检查是否可以使用range或index\_merge访问方法来索取

行。

 6). Selecttablesoptimized away：当我们使用某些聚合函数来访问存在索引的某个字段的时候，MySQLQueryOptimizer会通过索引而直接一次定位到

所需的数据行完成整个查询。当然，前提是在Query中不能有GROUPBY操作。如使用MIN()或者MAX（）的时候；

 7). Usingfilesort：当我们的Query中包含ORDERBY操作，而且无法利用索引完成排序操作的时候，MySQLQueryOptimizer不得不选择相应的排序算法

来实现。

8).  Usingindex：所需要的数据只需要在Index即可全部获得而不需要再到表中取数据（索引覆盖）；

9). Usingindexforgroup-by：数据访问和Usingindex一样，所需数据只需要读取索引即可，而当Query中使用了GROUPBY或者DISTINCT子句的时候，

如果分组字段也在索引中，Extra中的信息就会是Usingindexforgroup-by；

10). Usingtemporary：当MySQL在某些操作中必须使用临时表的时候，在Extra信息中就会出现Usingtemporary。主要常见于GROUPBY和ORDERBY等

操作中。

11). Usingwhere：如果我们不是读取表的所有数据，或者不是仅仅通过索引就可以获取所有需要的数据，则会出现Usingwhere信息；

12). Usingwherewithpushedcondition：这是一个仅仅在NDBCluster存储引擎中才会出现的信息，而且还需要通过打开ConditionPushdown优化功能

才可能会被使用。控制参数为engine\_condition\_pushdown。