# Oracle 的执行计划

## 优化定义

优化是选择最有效的执行计划来执行sql语句的过程,这是在处理任何数据的语句(select,insert,update Or delete)中的一个重要步骤.对Oracle来说,执行这样的语句有许多不同的方法,比如,将随以什么顺序访问那些表或索引的不同而不同.所使用的执行计划可以决定语句能执行的多块.Oracle中称之为优化器(Optimizer )的组件用来选择他认为最有效的

执行计划.

## 概念分析

### 共享SQL语句

**为了不重复解析相同的SQL语句(因为解析操作比较费资源，会导致性能下降)，在第一次解析之后，ORACLE将SQL语句及解析后得到的执行计划存放在内存中。**

**这块位于系统全局区域SGA(system global area)的共享池(shared buffer pool)中的内存可以被所有的数据库用户共享。因此，当你执行一个SQL语句(有时被称为一个游标)时，如果该语句和之前的执行过的某一语句完全相同，并且之前执行的该语句与其执行计划仍然在内存中存在，则ORACLE就不需要再进行分析，直接得到该语句的执行路径。ORACLE的这个功能大大地提高了SQL的执行性能并大大节省了内存的使用。使用这个功能的关键是将执行过的语句尽可能放到内存中，所以这要求有大的共享池(通过设置shared buffer pool参数值)和尽可能的使用绑定变量的方法执行SQL语句。**

### RowId

**rowid是一个伪列，既然是伪列，那么这个列就不是用户定义，而是系统自己给加上的。**

**对每个表都有一个rowid的伪列，但是表中并不物理存储ROWID列的值。不过你可以像使用其它列那样使用它，但是不能删除该列，也不能对该列的值进行修改、插入。一旦一行数据插入数据库，则rowid在该行的生命周期内是唯一的，即即使该行产生行迁移，行的rowid也不会改变**

### 为什么使用Rowid：

**rowid对访问一个表中的给定的行提供了最快的访问方法，通过ROWID可以直接定位到相应的数据块上，然后将其读到内存。我们创建一个索引时，该索引不但存储索引列的值，而且也存储索引值所对应的行的ROWID，这样我们通过索引快速找到相应行的ROWID后，通过该ROWID，就可以迅速将数据查询出来。这也就是我们使用索引查询时，速度比较快的原因。**

**在ORACLE8以前的版本中，ROWID由FILE 、BLOCK、ROW NUMBER构成。随着oracle8中对象概念的扩展，ROWID发生了变化，ROWID由OBJECT、FILE、BLOCK、ROW NUMBER构成。利用DBMS\_ROWID可以将rowid分解成上述的各部分，也可以将上述的各部分组成一个有效的rowid。**

### Recursive SQL概念

**为了执行用户发出的一个sql语句，Oracle必须执行一些额外的语句，我们将这些额外的语句称之为‘recursive calls’或‘recursive SQL statements’。如当一个DDL语句发出后，ORACLE总是隐含的发出一些recursive SQL语句，来修改数据字典信息，以便用户可以成功的执行该DDL语句。当需要的数据字典信息没有在共享内存中时，经常会发生Recursive calls，这些Recursive calls会将数据字典信息从硬盘读入内存中。用户不必关心这些recursive SQL语句的执行情况，在需要的时候，ORACLE会自动的在内部执行这些语句。当然DML语句与SELECT都可能引起recursive SQL。简单的说，我们可以将触发器视为recursive SQL。**

### Row Source(行源)

**用在查询中，由上一操作返回的符合条件的行的集合，即可以是表的全部行数据的集合；也可以是表的部分行数据的集合；也可以为对上2个row source进行连接操作(如join连接)后得到的行数据集合。**

### Predicate(谓词) ：

**一个查询中的WHERE限制条件**

### Driving Table(驱动表)：

**该表又称为外层表(OUTER TABLE)。这个概念用于嵌套与HASH连接中。如果该row source返回较多的行数据，则对所有的后续操作有负面影响。注意此处虽然翻译为驱动表，但实际上翻译为驱动行源(driving row source)更为确切。一般说来，是应用查询的限制条件后，返回较少行源的表作为驱动表，所以如果一个大表在WHERE条件有限制条件(如等值限制)，则该大表作为驱动表也是合适的，所以并不是只有较小的表可以作为驱动表，正确说法应该为应用查询的限制条件后，返回较少行源的表作为驱动表。**

**在执行计划中，应该为靠上的那个row source，后面会给出具体说明。在我们后面的描述中，一般将该表称为连接操作的row source 1。**

### Probed Table(被探查表)：

**该表又称为内层表(INNER TABLE)。在我们从驱动表中得到具体一行的数据后，在该表中寻找符合连接条件的行。所以该表应当为大表(实际上应该为返回较大row source的表)且相应的列上应该有索引。在我们后面的描述中，**

**一般将该表称为连接操作的row source 2。**

### 组合索引(concatenated index)：

**由多个列构成的索引，如create index idx\_emp on emp(col1, col2, col3, ……)，则我们称idx\_emp索引为组合索引。在组合索引中有一个重要的概念：引导列(leading column)，在上面的例子中，col1列为引导列。当我们进行查询时可以使用”where col1 = ? ”，也可以使用”where col1 = ? and col2 = ?”，这样的限制条件都会使用索引，但是”where col2 = ? ”查询就不会使用该索引。所以限制条件中包含先导列时，该限制条件才会使用该组合索引。**

### 可选择性(selectivity)：

**比较一下列中唯一键的数量和表中的行数，就可以判断该列的可选择性。如果该列的”唯一键的数量/表中的行数”的比值越接近1，则该列的可选择性越高，该列就越适合创建索引，同样索引的可选择性也越高。在可选择性高的列上进行查询时，返回的数据就较少，比较适合使用索引查询。**

## 执行计划分析

* **为了执行语句，Oracle可能必须实现许多步骤。这些步骤中的每一步可能是从数据库中物理检索数据行，或者用某种方法准备数据行，供发出语句的用户使用。Oracle用来执行语句的这些步骤的组合被称之为执行计划**
* **执行计划是SQL优化中最为复杂也是最为关键的部分，只有知道了ORACLE在内部到底是如何执行该SQL语句后，才能知道优化器选择的执行计划是否为最优的。执行计划对于DBA来说，就象财务报表对于财务人员一样重要。所以我们面临的问题主要是：**

1. 如何得到执行计划；

2. 如何分析执行计划;

**从而找出影响性能的主要问题。**

* **如何得到执行计划**

显示下面sql语句的执行计划：