

Re: an old worn interview question

Source: <http://coding.derkeiler.com/Archive/General/comp.programming/2007-09/msg00262.html>

- *From:* Tegiri Nenashi <TegiriNenashi@xxxxxxxx>
 - *Date:* Tue, 18 Sep 2007 09:26:57 -0700
-

On Sep 17, 4:38 pm, Logan Shaw <lshaw-use...@xxxxxxxxxxxxxxxx> wrote:

user923005 wrote:

I suppose Oracle could implement this in a cheesy fashion: it knows the size of the graph (the number of rows in the table), so if your table has N rows and you have gone N steps and not reached the root, you're in a loop. But one would hope they did something a little nicer than that. (You're supposed to be able to put zillions of gigabytes of data in an Oracle database without it falling over dead.)

In fact, there might be another interesting angle to this: Oracle lets you specifying "constraints" so that modifications to the database are disallowed if they would violate some particular property (such as a column being a unique key). I don't know whether it does or not, but if Oracle supports defining constraints that prohibit loops in hierarchical relationships, then there would be value in having an algorithm that can incrementally detect loops. That is, given a graph that doesn't have loops, if you make a set of changes to that graph (add/modify/delete nodes/edges), is it still free of loops?

- Logan

[1]http://download-west.oracle.com/docs/cd/B10501_01/server.920/a96540/q...

I doubt hierarchical query loop detection is possible. (Remember, hierarchical query is more general than just traversing a graph). Let me copy and paste a snippet from <http://www.bookpool.com/sm/0977671542> that partially addresses these issues. (There is also a dedicated chapter on graphs and trees)

Integer generators with Hierarchical Query

I have mentioned already DB2 integer generator leveraging recursive SQL. Oracle doesn't have recursive SQL capabilities (at the time of

Re: an old worn interview question

this writing), so that users have to use non-standard hierarchical query extension. A contrast between the Oracle and DB2 solutions is often enlightening. One fundamental difference between these two platforms is that Oracle seems to be able to detect loops, while DB2 doesn't make such a claim. Detecting loops, in general, is undecidable, which is the basis for DB2's position. Does Oracle's loop detection work because hierarchical extension has narrowed query expression capabilities compared to recursive SQL? Can we challenge it?

Consider a typical hierarchical query

```
select ename from emp
connect by empno = prior mgr
start with empno = 1000
```

First, Oracle finds all the nodes in the graph satisfying the start with condition. Then, for each batch of nodes found on a previous step, it finds a batch of extra nodes that satisfy the connect by condition. Any time the new node collides with the nodes that have been already discovered, it signals the connect by loop error. How does it identify the nodes? Should it compare all the relation attributes, or only those in the select clause, or choose some other ones? It is easy to see that the attributes in the select clause shouldn't matter. Indeed, adding a rownum pseudo column would artificially make the next node appear to be always different from its predecessors. The loop, however, is a property of the graph. Graph either has a cycle or not, no matter what node labels there may be. Therefore, the only columns which should be relevant for loop detection are the ones in the predicate with the prior prefix.

What if we write hierarchical query without the prior? This experiment reveals a remarkably succinct integer generator

```
select rownum from dual
connect by 0=0
```

As an alternative to the rownum pseudo column, we could use level

```
select level from dual
connect by 0=0
```

Both queries produce infinite result set. It is pipelined, however, so that the execution can be stopped either on the client, or explicitly on the server

```
select level from dual
connect by level < 100
```

.....

Re: an old worn interview question