FINDING THE UNUSED INDEXES

I am occasionally faced with the task of removing unused indexes from the database. In concept, this is very simple, but there are a few pitfalls that can lead to disastrous problems. For instance, if you unknowingly remove a critical (but infrequently used) index, you can wreck the performance of a critical month-end report.

The essence of the problem is this: How do you know *for a fact*, that a given index won't be needed at some critical time? Sure, you can be *pretty* confident, but is that good enough for a production system?

On several occasions I have witnessed the DBA removing supposedly unneeded indexes, only to have critical reports blow-up a few weeks later, at the end of the quarter. Just like the stock advisor's caveat, "Past performance is no assurance of future success," it's simply not possible to *guarantee* that past index non-use guarantees future non-use.

Where's my report?

A FEW OPTIONS

In this paper, we discuss two way to approach this problem. The first method, Oracle's index monitor facility, simply tells us whether an index has been used during a set period of time. This method is very simple to use, but the output is very limited. Furthermore, our confidence of index non-usage will only be as strong as the duration of the monitoring period. If we don't cover significant periods (such as end-of-quarter processing), we might make a big mistake in dropping an index.

The second method, based on Statspack, is far more sophisticated, and yields a lot more information—but takes a bit more work to setup. With this second method, we don't need to leave the monitor on; instead, we look back in time, for as long as we have been running Statspack. In addition, we glean a lot more information than just a simple Yes/No of index usage.

Let's take a look at the two methods and contrast their setup and use.

METHOD 1: INDEX MONITORING

Oracle provides a monitoring facility whereby you can see if an index is ever used during the monitoring period. The command to start monitoring is really simple:

```
ALTER INDEX CHRIS1 MONITORING USAGE;
```

After several days (or weeks), check the results by running:

```
SELECT INDEX_NAME, TABLE_NAME, USED FROM V$OBJECT_USAGE;

INDEX_NAME                     TABLE_NAME                     USE
------------------------------ ------------------------------ ---
CHRIS1                         CHRISTEST                      NO
```

To turn off monitoring, simply run:

```
ALTER INDEX CHRIS1 NOMONITORING USAGE;
```

ADVANTAGES OF INDEX MONITORING

While useful, this simple "Yes"/"No" answer for each index checked is not that helpful. It would be much more useful to identify the exact SQL that used the index, along with the resource usage. For example, Oracle might be using a certain index only because you haven't given it any better alternatives. Just because the index is used, doesn't mean it's a good choice. By knowing statistics such as elapsed time or disk i/o, you can identify tuning opportunities.

Let's now check the second method of monitoring index usage, using Oracle's *Statspack*. Of course, there are many tools and packages that you can install, which will monitor resource usage and identify problem areas. It turns out, however, that the Statspack facility already offers much of this functionality.

Many shops run snapshots every hour or so, so you may already have all the data you need. The only question is, how do you extract this critical information?

METHOD 2: ORACLE STATSPACK

The beauty of the statspack method, is that Oracle has already stored the information you'll likely need. When you activate the Statspack snapshots, Oracle stores the Sql that exceed the threshold settings for buffer_gets, physical I/O, parse calls, or executions. For example, statspack saves the statistics for all Sql that consume more than 1,000 disk reads. This information is displayed when you run the *spreport.sql* script.

In our case, however, *spreport* is not too helpful. We're not really interested in a summary of resource hogs; instead, we want to zero-in on index information. So, instead of running the spreport, we will query the Stats$ objects directly, extracting only that data relevant to our purposes.

TWO USEFUL SCRIPTS

Let's take a look at two simple scripts. Our first retrieves all Sql related to a specific object name—in our case "CHRIS_TEST." We'll also add a runtime threshold, so that we only look at Sql that runs longer than just a few seconds. Remember that the field *Elapsed_Time* is in units of microseconds, so be sure to scale accordingly. If you have stored a large number of snapshots, this script will take a few minutes:

```
Break on SECS

Select Elapsed_Time/1000000 SECS, Sql_Text From (
Select /*+Use_Hash (St Use Sum T) Ordered*/
Distinct St.Snap_Id, Elapsed_Time,T.Hash_Value, T.Piece, T.Sql_Text
From Stats$Sql_Plan St,
Stats$Sql_Plan_Usage Use,
Stats$Sql_Summary Sum,
Stats$Sqltext T
Where St.Plan_Hash_Value = Use.Plan_Hash_Value
And Use.Hash_Value = Sum.Hash_Value
And Sum.Hash_Value = T.Hash_Value
And Object_Name In ('CHRIS_TEST')
And Elapsed_Time > 10000000
Order By St.Snap_Id,T.Hash_Value, T.Piece);

      SECS SQL_TEXT
---------- ----------------------------------------
 15.585015 select OBJ_TYPE, count(*) from CHRIS_TEST
```

ANOTHER EASY SCRIPT

If you just want to find-out when a particular object was used, you can also get that from Statspack. Here's a simple script to find out when a particular object was used in some prior execution plan:

```
Select St.Snap_Id, To_Char(Snap_Time,'Dd-Mon-HH24:Mi') Snap
From Stats$Sql_Plan St,
Stats$Snapshot Sn
Where Object_Name In ('CHRIS_TEST')
And St.Snap_Id = Sn.Snap_Id
Order By 2;

  SNAP_ID SNAP
--------- ------------
    12152 13-Jun-12:45
    12153 13-Jun-13:45
```

In the above scripts, you can easily customize your queries to list just index references, or perhaps just index references for which the query ran a long time. The view *Stats$Sql_Plan* is nearly an identical copy of *V$Sql_Plan*, so you can customize your queries to focus only on certain schemas or object types.

IT'S EASY, BUT BE CAREFUL!

We have looked at two easy ways to check index usage in Oracle. Both are simple, and neither requires a lot of preparation. You can easily turn on index monitoring at any time. For the second method, the main requirement is that you must already have a long history of stored snapshots.

Always keep in mind the potentially serious consequences of removing an index that later proves important. Don't be too hasty to drop production database objects of *any type*. Plus, have a fallback strategy ready, just in case.