SLOB - Silly Little Oracle Benchmark

INDEX

        INTRO
        NOTE ABOUT SMALL SGA
        SETUP STEPS
        RELOADING THE TABLES
        RESULTS
        TERMINOLOGY
        HOW MANY PROCESSES DO I RUN
        NON-LINUX PLATFORMS

INTRO

This kit does physical I/O. Lot's of it.

The general idea is that schema users connect to the instance and execute SQL on their own tables and indexes so as to eliminate as much SGA *application* sharing as possible. SLOB aims to stress Oracle internal concurrency as opposed to application-contention. It's all about database physical IO (both physical and logical) not application scaling.

The default kit presumes the existence of a tablespace called IOPS. If you wish to supply another named tablespace it will be given as a argument to the setup.sh script. More on this later in this README.

To create the schemas and load data simply execute setup.sh as the Oracle sysdba user. The setup.sh script takes two arguments the first being the name of the tablespace and the second being how many schema users to load. A high-end test setup will generally load 128 users. To that end, 128 is the default.

To run the test workload use the runit.sh script. It takes two arguments the first being the number of sessions that will attach and perform modify DML (UPDATE) on their data (writer.sql) and the second directs how many sessions will connect and SELECT against their data (reader.sql).

NOTE ABOUT SMALL SGA

The key to this kit is to run with a small SGA buffer pool to force physical I/O. For instance, a 40MB SGA will be certain to result in significant physical IOPS when running with about 4 or more reader sessions. Monitor free buffer waits and increase db_cache_size to ensure the run proceeds without free buffer wait events.

Oracle SGA sizing heuristics may prevent you from creating a very small SGA if your system has a lot of processor cores. There are remedies for this. You can set cpu_count in the parameter file to a small number (e.g., 2) and this generally allows one to minimize db_block_buffers. Another approach is to create a recycle buffer pool. The setup.sh script uses the storage clause of the CREATE TABLE command to associate all SLOB users' tables with a recycle pool. If there happens to be a recycle pool when the instance is started then all table traffic will flow through that pool.

SETUP STEPS

1. First, create the trigger tools. Change directory to ./wait_kit
   and execute "make all"

```
2. Next, execute the setup.sh script, e.g., sh ./setup.sh IOPS 128
3. Next, run the kit such as sh ./runit.sh 0 8

RELOADING THE TABLES
--------------------
When setup.sh executes it produces a drop_users.sql file. If you need to
re-run setup.sh it is optimal to execute drop_users.sql first and then
proceed to re-execute setup.sh.

RESULTS
-------
The kit will produce a text awr report named awr.txt. The "awr" directory
scripts can be modified to produce a HTML awr report if so desired.

TERMINOLOGY
-----------
SLOB is useful for the following I/O and system bandwidth testing:

1. Physical I/O (PIO) - Datafile focus
        1.1 This style of SLOB testing requires a small db_block_cache
        setting. Small means very small such as 40MB. Some
        users find that it is necessary to over-ride Oracle's built
        in self-tuning even when supplying a specific value to
        db_cache_size. If you set db_cache_size small (e.g., 40M)
        but SHOW SGA reveals an over-ride situation, consider
        setting cpu_count to a very low value such as 2. This will
        not spoil SLOB's ability to stress I/O.
        1.2 Some examples of PIO include the following:
                $ sh ./runit.sh 0 32   # zero writers 32 readers
                $ sh ./runit.sh 32 0   # 32 writers zero readers
                $ sh ./runit.sh 16 16  # 16 of each reader/writer
2. Logical I/O (LIO)
        2.1 LIO is a system bandwidth and memory latency test. This
        requires a larger db_block_cache setting. The idea is to
        eliminate Physical I/O. The measurement in this testing mode
        is Logical I/O as reported in AWR as Logical reads.
3. Redo Focused (REDO)
        3.1 REDO mode also requires a large SGA. The idea is to
        have enough buffers so that Oracle does not need to
        activate DBWR to flush. Instead, LGWR will be the
        only process on the system issuing physical I/O. This
        manner of SLOB testing will prove out the maximum theoretical
        redo subsystem bandwidth on the system. In this mode
        it is best to run with zero readers and all writers.


HOW MANY PROCESSES DO I RUN?
----------------------------
I recommend starting out small and scaling up. So, for instance,
a loop of PIO such as the following:
        $ for cnt in 1 2 4 8
        do
                sh ./runit.sh 0 $cnt
        done

Take care to preserve the AWR report in each iteration of the loop.
The best recipe for the number of SLOB sessions is system specific.
If your system renders, say, 50,000 PIOPS with 24 readers but starts
to tail beyond 24 then stay with 24.
```

In general I recommend thinking in terms of SLOB sessions per core.

In the LIO case it is quite rare to run with more readers.sql than the number of cores (or threads in the case of threaded cores). On the other hand, in the case of REDO it might take more than the number of cores to find the maximum redo subsystem throughput--remember, Oracle does piggy-back commits so over-subscribing sessions to cores might be beneficial during REDO testing.

NON-LINUX PLATFORMS

The SLOB install directory has of README.{PLATFORM} files and user-contributed, tested scripts under the ./misc/user-contrib directory.