



Your Performance IS Our Business

10713 RR 620 North, Building 400

Austin, Texas 78726-1708

(512) 249-2324 (866) 773-8789

www.DBISoftware.com

Frequently Asked Questions (FAQs) About Simulation

Simulation – what it is and what it isn't?

Simulation is a predictive technique for assessing the performance of SQL statements by modeling the key events that take place during the lifetime of a statement's execution. For people with a Computer Science background it is in fact discrete event simulation modeling – i.e., execution of the discrete events within the statement's execution path against a pseudo or simulation time clock that runs orders of magnitude faster than wallclock time.

Simulation should not be mistaken for analytical modeling that is ostensibly mathematical and relies on the solution of multiple simultaneous equations that effectively model or describe a system in steady state. Analytical techniques are suitable for workload and system modeling at a high level where workload measurements (established from monitors) are extrapolated mathematically based upon growths in transaction rates.

Analytical techniques should not be undersold as they underpin BEST/1 Perform and Predict. The poor relation of simulation and analytical modeling from a predictive perspective is trending – which really relies upon fitting some form of curve (usually a straight line) through a mass of historical data and assuming that future behavior will conform to the extrapolation of that curve. This can be quite effectively performed through EXCEL and requires no real technology. It is of limited value in IT systems that rarely exhibit standard growth behaviors – who ever grew a business based upon a straight line or (perhaps more attractively) exponential growth? The key to this one is that the devil is in the detail – if you don't account for the detail, you will never find the devil!

Does it model Oracle's optimizer? If not, what does it model?

Simulation makes no attempt to model Oracle's optimizer – output is taken from the plan table after the SQL statement has been optimized according to Oracle's prevailing view of the world. It would be foolish to attempt to predict, reverse engineer or even guess the outcome of Oracle optimization – that is Oracle's core business and optimizer code is jealously guarded and highly changeable across releases. After all, it is a significant element of their competitive advantage.

Simulation does model the anatomical operations performed by the RDBMS against the underlying objects of the database (as outlined in previous sections). It works out the logical IOs required to service each database access (at the operation level), determines the physical IOs required to meet those logical requests for data (using a model of the buffer cache function), evaluates the CPU time associated with serving that data request (similarly at the operation level) and estimates the elapsed time to perform the overall query.

In short, it executes a previously explained SQL query against a virtual instance in a fraction of the time taken to execute that query in the real world and without any intrusion against the real database instance.

Does it require cost or rule based optimizer? Do objects have to be analyzed?

Simulation requires cost based optimization to be employed for the session used to execute the explain plan. This is due to the fact that cardinality values cannot be established in the plan table under rule-based optimization. This is simply an Oracle fact and there is no way of getting around it. The key is that Oracle is reaching the point where rule based optimization is virtually unsupported from their perspective (though this is still something of an issue with certain packaged application providers)!



Your Performance IS Our Business

10713 RR 620 North, Building 400

Austin, Texas 78726-1708

(512) 249-2324 (866) 773-8789

www.DBISoftware.com

Packaged application vendors tend to tie down their SQL statements and prevent code change. Consequently, in this kind of environment, Index Advisor is the real “killer capability” as it provides optimization capability by tuning schema for SQL rather than SQL for schema. Note: nobody else does this as well as DBI does.

Simulation requires cost based optimization for the cardinality values to appear in explain plan, so it is clear that analyzed tables are a pre-requisite to getting accurate answers. Remember – the better the cardinality estimates that the optimizer can make (and Oracle demands that all objects are analyzed), the better the accuracy of the simulation. The more up-to-date the statistics are, the better the simulation is in terms of accuracy.

What are the limitations of simulation?

It should always be remembered that simulation is a modeling solution – it is a model and models have limitations. The primary limitation is the quality of data supplied by Oracle explain plan output which is, at best, a guess of cardinality made by the optimizer. It is the best guess in town right now and it is the guess that will be used by Oracle to satisfy the query. The secondary limitation is the use of a variety of different predicates against the same bind variables in an environment where underlying data distributions are highly skewed. In such a query simulation will give a more consistent basis for tuning than the random substitution of bind variables in real queries run against the real database instance.

Simulation will provide a consistent basis for tuning and improving the query whereas real world substitution will confuse the potential benefits associated from better SQL structures with lucky bind variable substitution. Those operations that are inherently compromised by these two weaknesses are identified wherever possible. Most particularly, where the Oracle optimizer provides insufficient cardinality information, accesses are marked for intermediate support. The key argument here is the difference between accuracy and consistency. The simulation model will inherently be consistent and consequently provides a good basis for tuning – isolated from the vagaries of an ever-changing production instance. However, accuracy is more subjective as any external action on an instance which may have significant affect upon the query under test.

Market evaluations appear to have indicated that consistency has been valued over accuracy although achieving 100% in both camps is clearly desirable. If it was achievable most DBAs who made a living out of tuning would be out of a job so remember, this is not a silver bullet but a highly powerful weapon!

How accurate is it?

The simulation is expected to be within 20% of real world observation. Reference should be made to the limitations expressed above – i.e., here Oracle does not give any clues through explain plan the task becomes decidedly more difficult. When Oracle does not give any clues you are going to be pretty stuck anyway.

SQL Explorer always allows you to run the query against the real instance if Oracle isn't being helpful enough! Fortunately this does not occur very often. The simulation works things out in a manner that effectively mirrors Oracle's execution stack. Logical IOs are determined first and are likely to be the most accurate. Physical IOs are determined after subjecting logical IOs to the buffer cache model and will inherently be less accurate. CPU time is derived from logical and physical IOs and numerous other significant factors, while elapsed time is subject to the addition of estimations of disk IO times and network transfer delays (as established by calibration).

The trick is retaining accuracy as evaluations are made down the stack and additional levels of modeling are incurred. Most experienced DBAs are taught to tune queries based upon logical IO counts and all inexperienced



Your Performance IS Our Business

10713 RR 620 North, Building 400

Austin, Texas 78726-1708

(512) 249-2324 (866) 773-8789

www.DBISoftware.com

DBAs should be taught that to do so is good practice. After all – reduced the calls for data and you will make everything run faster. This key fact – along with the consistency versus accuracy argument, represent the prime arguments for converting users to the benefits of simulation.

Does the simulation assume that data is cached or being pulled from disk?

The simulation models the behavior of the buffer cache but clearly has no prior state information or representation of other queries impacting the environment. It is this isolation that makes the virtual instance consistent and an ideal base for performing tuning. For caching purposes, the following three mechanisms will apply:

- **RANDOM HIT CANDIDATES** (those blocks images that are already cache resident in an operational environment). These include particular block types such as index root, index intermediate and segment headers. For these block types, it is assumed that if n blocks are read and n blocks reside in the cache, the logical read requests will be satisfied by those n resident blocks and no physical IOs will be generated.
- **RANDOM MISS CANDIDATES** (those block images that are likely to be cache volatile in an operational environment). These include table blocks and index leaf blocks (when not accessed sequentially in multiblock units ref. **SEQUENTIAL CANDIDATES**). For these block types the physical IO counts are determined by evaluating the probability of a block being non-resident in the cache based upon proportional occupancy. For example if an operation qualifying for this treatment were to generate 100 logical IOs against an object of 500 blocks in size where 50 blocks were already cache resident, the physical IO count will be calculated based upon the proportion of the object not cached i.e. 450 blocks out of 500 or 90%. Consequently in this example 90 physical IOs would be generated from the 100 logical IOs. The operations that will use this behavior the most are **INDEX RANGE SCAN**, **INDEX UNIQUE** and **TABLE BY ROWID**.
- **SEQUENTIAL CANDIDATES** (those operations that will perform IOs exploiting the multiblock read capability). In this case the number of physical IOs is determined by dividing the logical IO count by the multiblock read count and rounding up. These blocks are always placed at the LRU end of the chain and will be subject to being immediately forced out of the buffer cache by the next physical reads (either within the same query execution or by the subsequent execution of the same query). The operations that will use this behavior the most are **TABLE FULL** and **INDEX FAST FULL**.

The simulations are run five times (one after the other) and measurements reported are based upon means taken over those five iterations. It is worth noting that the cache performance of **SEQUENTIAL CANDIDATES** will be linear and consistent across those five iterations as will the cache performance of **RANDOM HIT CANDIDATES**. On the other hand, **RANDOM MISS CANDIDATES** will produce progressively lower physical IO counts across the multiple iterations (as more and more of the object is loaded into the cache). This might well result in higher than normal physical IO counts for random accesses against very large objects. While this is not accurate, it will be consistent and will consequently provide a better basis for tuning than real world execution (which is always exposed to high volatility in cache behavior).

What type of queries does it model best – short OLTP or long data warehouse?

Clearly, the difference between using simulation to evaluate a long Data Warehouse query in minutes that would run for hours or maybe days against the real world database is compelling. In fact, this is perhaps the most compelling argument for simulation. The comments made in previous sections regarding the enhanced consistency afforded by the simulation environment (remember it is insulated from the varied activities and inconsistent measurements of the production instance) make it a compelling tuning yardstick in a short query dominated OLTP



Your Performance IS Our Business

10713 RR 620 North, Building 400

Austin, Texas 78726-1708

(512) 249-2324 (866) 773-8789

www.DBISoftware.com

type environment. The easier sell is for long running queries however, the potential for all queries should not be overlooked.

Can I change initialization parameters and/or object attributes e.g. sort area size or num_rows?

Absolutely, this is a powerful capability that will provide significant differentiation from OEM (for 9i).

Does it run on the client or the server?

The simulation process runs on the client without impacting the database at all. However, there is a small overhead on the server in extracting the information from key system tables that enables the simulation to run.

Do I need to be connected to the database to use it?

Yes. As mentioned above, the simulation is driven by the information contained in the plan table and consequently connection to the database to extract information from the plan table is still required.

Does it require any space on the database?

The only space required on the database for simulation to take place is for the execution plan table and the calibration table. The impact upon storage requirements should be minimal.

What type of load does it put on the database?

The only loads placed on the database are in the execution of the calibration queries, in explaining the query into the plan table and in the queries required to set up the model. Calibration queries can run for some time, but once calibrated, the calibration file can be saved for future use. Setting up the model involves activity against the DBA views and a query to determine the average row length of an index using an INDEX FULL SCAN. If tables are unanalyzed, they will be subject to a FULL TABLE SCAN as part of model construction and this may prove costly for large tables. Consequently, all tables should be analyzed.