A STANDARD BENCHMARK FOR ORACLE APPLICATIONS

Tim Harris, Oracle Corporation

INTRODUCTION

Despite the importance of the Oracle Applications product suite, we have traditionally not had a common metric by which to evaluate and compare the performance of the suite on various hardware platforms and while utilizing various feature sets. Oracle has recently provided such a tool, now called the *Oracle Applications Standard Benchmark*, and in this white paper, we outline its goals and composition of this new performance product.

Understanding the detailed performance characteristics of a product suite that is as broad and sophisticated as Oracle Applications is far from trivial. One useful tool in developing such an understanding is a well-defined and reproducible workload. This workload should represent as closely as possible the work that the customer base does on a daily basis, but also be easy to port and run on a wide range of configurations and platforms. The Oracle Applications Standard Benchmark represents such a workload, and the overall goals are two-fold. Firstly to provide specific focus for Oracle developers and field organizations on how to make the performance of the Applications suite perform optimally for our customers, and secondly, to increase the amount of collateral available to customers who want to understand sizing and capacity requirements for a variety of hardware platforms.

Oracle has provided early versions of this workload for over a year, and across both the 10.7 NCA and 11.0 product suites. However, the new Oracle Applications Standard Benchmark not only provides a common workload, but also an audit mechanism such that we can confirm that one run of the benchmark truly compares to another run, hence providing and apples-to-apples comparison that should make results substantially more useful to our customer base and development organizations. We hope that the resulting set of information will simplify sizing decisions of our customers in the future, and also help maintain and improve the overall performance of the Applications suite.

WORKLOAD OVERVIEW

Our target for the benchmark is a realistic workload mix that represents accurately a common customer scenario, with both a high volume of OLTP users and a substantial batch component. The key is to ensure that the chosen workload mix is accurately reproduced as the benchmark is run on both large and small systems, with a wide range of operating systems and configurations. Over time we may introduce other related workloads and hence cover a larger set of the product suite, but initially we have focused on two of the most common subject areas across our ERP customer base: Financials and Supply Chain Management (SCM). Seven modules are utilized across these two subject areas, namely:

- Financials:

  Accounts Payable (AP), Accounts Receivable (AR), General Ledger (GL), and Fixed Assets (FA)

- Supply Chain Management:

  Purchase Orders (PO), Order Entry (OE), and Inventory (Inv)

We provide a broad set of 18 OLTP transactions across this set of modules. Given that most ERP users also have substantial batch components to their workload, we have also included seven batch jobs. Both the OLTP and batch mix are maintained to be consistent across arbitrarily large runs of the benchmark. The specific definitions of each transaction and their target rates were provided by functional consultants from Oracle who work with such ERP customers on a daily basis. The batch component consists of four reports, Autopost for outstanding journal entries, the postings of those journal entries themselves, and an Inventory batch job that performs inserts. We have worked to ensure that the Read: Write ratio of the standard workload is quiet similar to that observed in true customers.

The workload represents the normal daily activity of a mid-market sized company, with a daytime batch load of roughly 30%. The database used is a synthetic database of substantial size. Called the *Vision*++ database due to its connection to the Vision Demo database, it contains hundreds of thousands to millions of rows in all the tables utilized in benchmark. We do not populate the tables that are not used by this mix of transactions, and hence the overall size of the database is still manageable, being roughly 30 Gigabytes. However, if all tables were populated to the same degree then we would expect the database to be over 200 Gigabytes in size, and for comparison with customer databases, this is probably a more relevant number.

Load-Driver Technology

To further simplify use of the benchmark kit we use a commercial load-driver tool, currently Mercury Interactive. This provides the simplicity of a fully GUI interface for collecting and evaluating results. Oracle has collaborated with the vendors of such tools such that we may now drive such a high-volume workload at the network layer, which accurately reflects the entire three-tier nature of Oracle Applications. This technology module is now called the *Oracle NCA* component of Mercury Interactive's load driver tools, and is available from them.

Run-Time Objectives

The first order results of the benchmark are user count and 90th percentile response time for a given run. It is expect that users of the benchmark will optimize results by iteratively increasing user counts for each run of the benchmark until unacceptable response time is observed. The audit criteria is that the 90th percentile of response time for the run does not exceed four seconds, that the transaction rates for all users are maintained across the run, and that the specified mix of users is consistent. We also ensure that the batch component of the workload is effectively executed throughout the run.

We expect much of the benefits of using the kit will come from more detailed analysis of results, results that include CPU and memory utilization on both the data server and the middle-tier during the steady state phase of the run, as well as Oracle database statistics such as UTLBSTAT/ULTESTAT reports. The target of a run is generally to configure your system such that CPU on the data server becomes the limiting resource and to evaluate what user count can be achieved with the data server CPU resources that are available.

Use Of Results For Sizing

The Oracle Applications Standard Benchmark is designed to provide experience and confidence that running high numbers of concurrent users is achievable both in the benchmark lab and for our customers. We believe that this workload will provide collateral that will prove useful in a sizing process, but that using benchmark numbers within the context of sizing is a somewhat subtle process, and should be performed by those with substantial experience in this area. In particular, since Oracle Applications is such a broad range of products and possible configurations, we expect that benchmark results should be used only as general guidelines for sizing, or else they should be scaled to make them more applicable to a customer with a well known but different workload mix. For example, a customer may have a different mix of modules, a larger or smaller daily batch load, and a larger or smaller database, and all these components can affect the load they see substantially.

Configuration Guidelines

Running the benchmark requires a full three tier setup, with a data server, fully configured middle tier and a load driver tier. We show a rough outline of such a configuration below in figure 1. The ellipses show what components can scale to an arbitrary number. In general, a configuration under test will have multiple load drivers and multiple middle tiers for a single data server. We typically run with a single CPU controller box though the controller can also be run on the load driver tier.

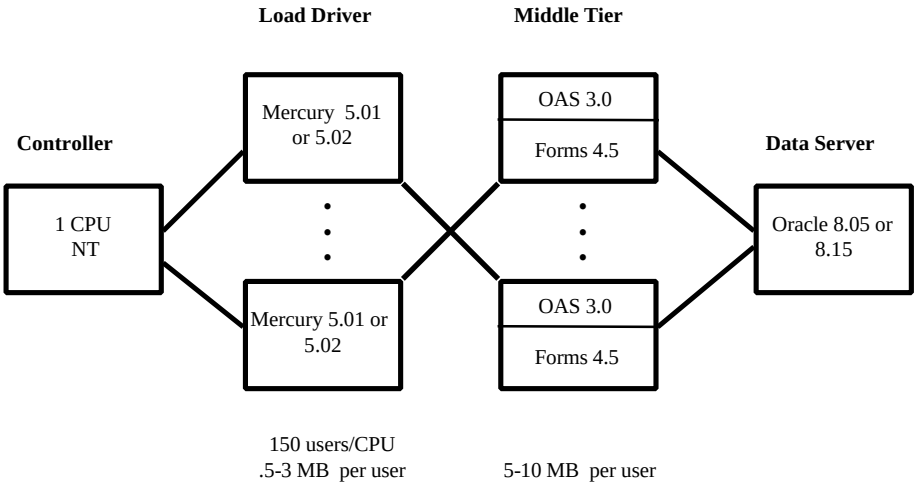

Figure 1: Typical Configuration for Benchmark Runs

Overall, certification for the benchmark follows closely that of Oracle Applications in general - configuration and tuning of the benchmark must be with certified techniques and versions of the entire product stack. In this way, we ensure that the tuning, which is done to optimize running of the benchmark, is directly applicable to improving the experience of our customer base.

WORKLOAD SCENARIO

Scenario is the term used for the detailed definition of the workload's run-time characteristics. For example:

- How many users of what type will run?
- How often will they sleep and when will they be active?
- When will they complete?

In the Oracle Applications Standard Benchmark, (as opposed to the earlier prototypes), we provide a fixed scenario for use with Mercury Interactive, which should be used with little or no changes. The mix of transactions and the rate at which each user executes transactions both represent commonly seen scenarios from customers of Oracle Application, as reported by the consulting and field organizations of Oracle.

ATOMIC USER GROUP AND BASE USER COUNTS

A list of the transactions and their suggested completion rate is specified below in **Table 1.** To ensure that our workload characteristics remain constant across a variety of user counts we define the concept of an *Atomic Group* of users. The atomic group is the smallest number of users that can be added at any given time while still ensuring that the workload remains unchanged, specifically in terms of the percentages of each type of user in the mix. To run more users you simply add an integer number of atomic groups, and thereby the characteristics of the workload remain constant.

The atomic user group consists of the sum of all the base user counts specified in the table. The key components of a scenario are to build an atomic user group with this set of transactions with the specified base user counts.

Name of Transaction	Trans/HR/User	Base User Count
AP Enter Asset Invoice	6	2
AP Enter Inv Multi Dist	6	2
AP Invoice Dist Inquiry	7	4
AP Invoice Inquiry	7	3
AR Collections Report	4	1
AR Customer Summary	8	4
AR Insert Manual Invoice	6	2
AR View Customer Trans	6	2
FA Asset Inquiry	7	4
GL Account Inquiry	10	4
GL Journal Entry	8	4
INV Detail Report	4	1
INV Insert Misc Trans	9	4
INV View Item Attribs	8	3
INV View Item Cost	8	3
OE Detail Report	4	1
OE Insert Order	8	3
OE View Order	7	2
PO Detail Report	4	1
PO Enter Purchase Order	6	2
PO New Purchase Req	6	2
PO View Purchase Order	6	2
Total	145	56

Table 1: The Atomic Group

In the case where it is desired to add more users but there are not enough system resources to add an entire atomic group you may add individual users, but only within a specified order as specified in the documentation for the kit. Adding individual users in an order other, then this will result in an invalid run.

Transaction Rates And The User Model

In the Standard Benchmark, it is expected that little or no changes to the included scenario will be necessary. However, some explanation of the scenario characteristics will still prove useful. Overall, one of the basic tenants of the benchmark is that the user rate of work is constant, independent of system load or response time. Therefore, throughput of the benchmark is strictly a function of user count, not of response time. This is ensured primarily through the use of the iteration pacing function of the load-driver tools. The iteration pacing should correspond exactly to the minimum transaction rates specified above if **Table 1**. I.e., if a transaction is required to run at least 6 times per hour, then the iteration pacing is set to 10 minutes or 600 seconds. This should ensure that the minimum transaction pacing is maintained, and if for some reason that is not the case then a warning will appear in the log files of the run, stating that the iteration pacing setting has been violated. Such warnings are an indication that at least for that one transaction the minimum transaction rate was not achieved. The run may therefore be deemed invalid during an audit.

TIMED EVENTS AND RESPONSE TIME

Response time for a run of the benchmark is calculated as a 90th percentile of all the measured response times for the run. A *Timed Event* (TE) is anything that we have explicitly timed in the transaction scripts and for which we receive response times. The Timed Events included in the current kit represent events we deemed most significant in terms of the user experience for someone running the transaction manually. Saving an invoice or submitting a query are examples of such important Timed Events. These timed events are then summarized in the "overall_timed_event" which is shown at the end of the Transaction Performance Summary Report, one of the commonly used outputs from a benchmark run.

RUNTIME USAGE

There are four stages to a benchmark run:

1. Ramp up

   Users are logged in to the system
2. Settle Time

   Wait for the resource usage of the new users to stabilize
3. Steady State

   The benchmark run data is captured
4. Ramp down

   The run is terminated and users log out

A complete benchmark run consisting of all four stages commonly consumes about an hour, though this time will increase as user counts get higher and the ramp up phase takes longer. An overview of this can be seen in **Figure 2**. For a benchmark run to be accurate, you restore the database to its original state before beginning a run. However, multiple iterations of the benchmark may be run without database restores during the initial testing phases.

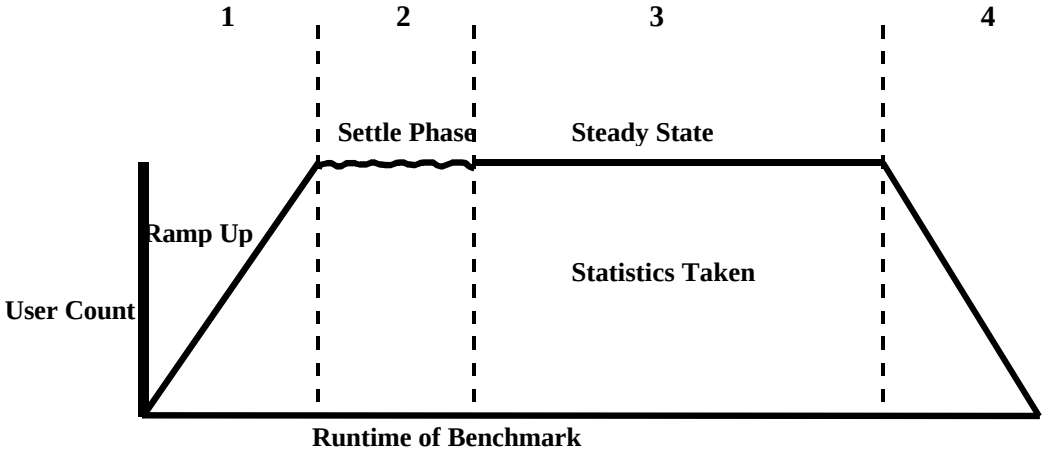

Figure 2: The four phases of a benchmark run

During the ramp up phase, new users are started at a general rate of one atomic group every few minutes. The number of users that are initialized at any time is determined by a Mercury parameter, which defaults to five. If this initialization quota is set too high than the CPU usage during initialization is high enough that Mercury may allow some users to fail. Keeping this value near the default of five should allow the high overhead of login users to be amortized across the pauses. Using Mercury to delay the start of various groups or users within a group may also be effective. Settle time is the final opportunity to allow any spiky resource consumption to subside. Effective ramp-up

and settle phases will result in a transaction load that is well distributed, therefore, allowing CPU utilization on the data server and middle-tier to be steady over time. These two phases should last roughly 20 minutes for user counts less than 1000, and will increase slowly as the need to start more groups increases.

The steady state phase is the true benchmark run where statistics and results are generated and collected. The first order results are the response time statistics generated for the window of time associated with the steady state phase. The only requirement for the steady state phase is that it is at least 30 minutes long. The steady state start and end time are chosen by the vendor, and this information is then provided to the auditors of a benchmark run. The "time filter" function of Mercury allows one to generate results that correspond only to this steady state phase of the benchmark, thereby ignoring the statistics generated during the ramp-up and ramp-down phases of the run.

DATA COLLECTION AND ANALYSIS

The data collected during a run should allow an observer to see the memory and CPU consumption on both the data server and the middle tier. This is commonly done through use of simple monitoring tools such as "sar," though the best tool will depend on the operating system is use. Additionally we require that utlbstat and utlestat are run - Oracle database monitoring scripts. The results of these scripts should provide direction in assessing what Oracle resources may be in contention. Finally the Mercury log file and results will allow us to verify the transaction rate of each user as well as how many, if any, users have failed during a run. These three pieces of data are then provided as input to the audit process.

The report batch jobs will generate reports on disk that may take up disk space. These reports may be checked during auditing but don't need to be explicitly submitted to the auditors.

AUDITING AND VERIFICATION

Overall, the audit process consists of the pre-audit, where the validity of the configuration and setup of the benchmark are evaluated, and the run-time audit, where the validity of the workload during execution is evaluated.

The pre-audit will address the following:

- Setup and Installation:
  1. Versions for entire product stack, including load driver, middle-tier, and data server.
  2. Initialization parameters for the database (init.ora)
  3. Database object count and schema verification
- Source Code Validation:

  Validate that the Mercury scripts have not been changes, and that the provided scenario is unchanged except for think-time multipliers.
- The Run-time audit will address the following:
  1. Verify that the overall 90th percentile of response time does not exceed the 4-second limit. Additionally the overall average should be below this 90th percentile value to ensure that the excluded 10% are not unreasonably large.
  2. Verify that each user process has maintained at least the minimum transaction rate during the steady state period.
  3. Verify that at least 90% of the batch requests submitted during the steady state of the benchmark have completed during that steady state period.
  4. Verify that no user threads have died during the steady state period.

INITIAL RESULTS

We expect to have the first audited results appearing in the fall and winter of 1999, and these results will be made available via Oracle's main Web site and in various documents. However, before this time the benchmark has proven useful in many development oriented projects, each designed to test new technology and evaluated its benefits for our customers. We highlight a few of these preliminary results below. In development, we are primarily focused on adding new features and increasing the quality of future products, and hence most of the current development projects are targeting the 11i product and its use of the 8.1.6 RDBMS version. Many of the on-going development projects revolve around the goal of *globalization*, the move from businesses running many disjoint data centers, to running on large instance of Oracle Applications in a centralized location and having that instance support your entire global business.

ORACLE PARALLEL SERVER

Oracle Parallel Server (OPS) is a long-standing product designed to increase the scalability of the Oracle Data server to span multiple disjoint boxes which are all accessing the same shared database. Though there are many well-established and successful customers for this product, it has traditionally been little used in the applications space, mostly due to the lack of information regarding best practices for its use. Using Oracle's new 8*i* database we have set out using the benchmark to address this important product combination. The 8*i* database is particularly well suited to supporting Applications with OPS based on the new technology included in the server. In particular, the *Consistent Read Server* (CR Server) is a component designed to allow information to be shared across the interconnect network by multiple OPS nodes, whereas earlier the data would have been written back to disk and then read from disk. CR Server thereby substantially reduces the cost of such data sharing in OLTP oriented applications.

Though results are preliminary, the use early versions of the Oracle Applications Standard Benchmark to evaluate this technology have been very promising. CR Server serves to substantially reduce the response time observed in OPS configurations, and therefore holds good promise in allowing customers and even more scalable platform for Oracle Applications. Some results are presented below for the case of a two-node OPS cluster running the workload. We see a good relative gain in overall user count as we move from one node to two nodes, corresponding to a scalability gain of roughly 1.8 on the two-node system. Relevant data points are those where response time is below four seconds, as required by the benchmark run-time rules.

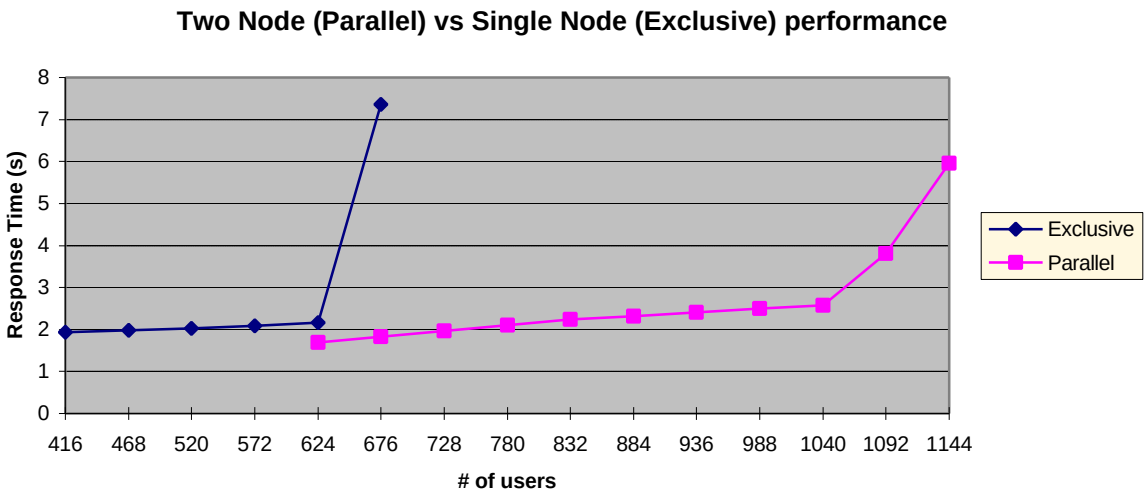

Figure 3: Initial results of OPS testing with Oracle Applications

USE OF ADVANCED CHARACTER SETS: UTF8

Part of the requirements for globalization is good support for many different languages, and hence many different character sets. *UTF*8 will be a supported character set in 11i, and use of this character set on the data server allows the use of many other multibyte character sets at the client level. Using the benchmark, we set out to evaluate the overhead involved in running with the UTF8 dataset on the data server. Initially we tested using only ASCII data, but with the UTF8 code path enabled. In this way we see the cost associated with the increased processing required for a multibyte character set, without the increase in I/O we would expect if we have true multibyte data.

Overall, the overhead of running with UTF8 on the data server looks very low, with response time degrading by well under 10%. We are currently building a new database that incorporates non-ASCII data so that we can also determine the increased I/O costs from this configuration. However, from the results to date we expect UTF8 to be an efficient option for those global installations of Oracle Applications that would like to support a variety of character sets for a diverse user population.

THE COST BASED OPTIMIZER

Traditionally Oracle Applications has used the Rule Based Optimizer (RBO) feature of the Oracle RDBMS. However, as we move to the 11i product we will be incorporating a large number of new database features into the product, including the *Cost Based Optimizer* (CBO). The CBO provides optimization of complex SQL code, and also helps in the support of new database technology, such as Partitioned Tables and Index Only Tables. Given the large and sophisticated base of code in Oracle Applications, the move from RBO to CBO is a crucial one and must be evaluated thoroughly before shipping of the 11*i* product can safely take place. One aspect of this project has been to use the Oracle Applications Standard Benchmark and earlier such prototypes to understand the performance issues which we may encounter when making this transition.

Overall, it appears that the Cost Based Optimizer is going to add significant value to Oracle Applications in the long term. Development teams have been working hard to analyze the execution plans of the existing and new SQL being used in the product. Oracle Application uses many sophisticated techniques, such as multi-table joins and large views, which now seem to result in efficient plans for CBO execution. The project has designed an analyze procedure for customers to ensure that database statistics are accurate such that the CBO has up-to-date information upon which to base its optimizations. Though the benchmark has been used largely for functional verification here, we plan to evaluate response time and throughput results from the use of CBO and compare them to the Rule Based Optimizer baseline.

SUMMARY AND CONTACTS

For regular updates on the Oracle Applications Standard Benchmark, be sure to regularly check the Web site at *www.oracle.com/apps_benchmark*. This has links leading to further documentation and sets of recently posted audited results, which we expect will appear before the end of the 1999 calendar year. See *The Oracle Applications Standard Benchmark: A Users Guide* for a more detailed description of the benchmark and how best to use it. Write *oabinfo@us.oracle.com* with any specific questions you may have on obtaining results or documentation or on how to obtain the latest benchmark kit.