Workload Characterization and Quantitative Modeling

Brahmaiah Jarugumilli is an Oracle System Designer and Architect with a deep experience in Oracle High Performance Architectures and a wide exposure to a variety of Industries. He is currently associated with GE-Cosnumer Finance as an Enterprise Architect for its Data Warehouses.

Shankar JayaGanapathy is a Technical Manager and Clustered Database Systems Architect with Advance Technology Solutions (Oracle Consulting, Oracle Corporation) and consults in the area of Performance and Availability.

Introduction:

Prepare for a journey into the world of Oracle statistics and their use in workload analysis, characterization and capacity planning. We introduce, the internal 'wait statistics' and 'system statistics', provided by Oracle for workload, diagnostics and performance analysis. The goal is to utilize these statistics as load-factors to qualitatively model an Oracle Database System and extend the idea to develop 'Capacity Planning Models' through linear and non-linear regression techniques.

What is system workload?

In short, the demand placed on a system's resources to do work can be defined as workload. In a database system, the demand arises from queries and transactional requests. The impact of this load on the system is observed in terms of CPU, Memory and I/O related resource consumption as the main components. In the context of this paper, workload is characterized based on Oracle Statistics as load-factors.

How is a system's workload influenced?

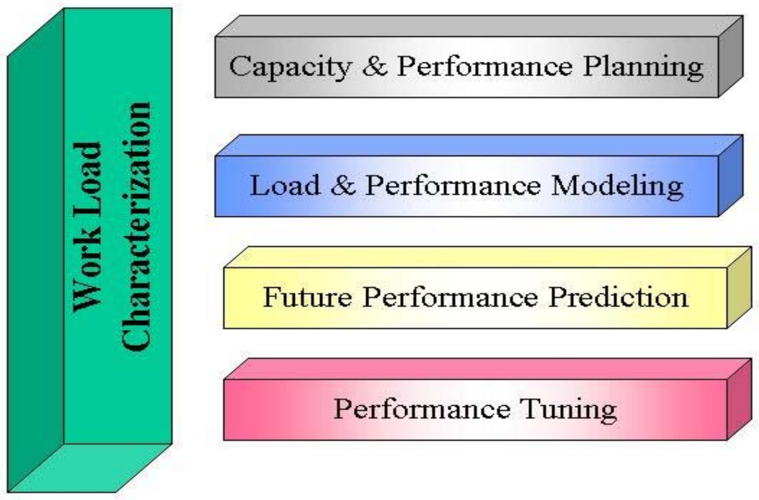
Here are some influential factors:

- End users and their behavior—the intensity with which they perform work.
- Concurrency of online users and the mixture of the executed programs.
- Nature of the end-user layer of the application—for example, various forms, canned reports or parameterized reports, etc.
- Application execution architecture—multiple tiers, proxy serviceability, threads of execution, serial vs. parallel execution, etc.
- Security and audit features
- Layers, levels and sizes of caching—within Oracle, O/S, I/O sub-system, Network, Application Server (JAVA caching), Web Server (HTTP Caching), etc.
- Data models and nature of the data—like volatility, reuse, concurrency etc.

Why should we characterize a system's workload?

Accurate characterization is important for modeling and capacity planning. The goal of workload characterization is to observe and accurately understand the stress and strain placed on a system. This understanding, in turn will help in defining objectives and building a competent model. Ideally, we isolate and identify the sensitive system components that widely influence the observed system behavior. A model based on these influential system components will there by firm up the foundation and also lends credibility to related quantitative simulations.

Fig-1:


Work Load Characterization
Capacity & Performance Planning
Load & Performance Modeling
Future Performance Prediction
Performance Tuning

Oracle database has built-in mechanisms to collect system statistics. It also offers application interfaces (APIs such as bundled software packages or GUI based tools) to query, display, monitor and report on system statistics. There are several different statistics that are collected by an Oracle database routinely. Each of them offers a different perspective on database activity. In reality, not all of them may be very meaningful under all situations. There are several interfaces that one can interact with, to query and display these statistics. In general, querying fixed views (V$ or GV$) provide a means to report on these statistics. The querying and reporting strategy usually depends on the objective. Some common views that provide database statistics are: v$sysstat, v$sesstat, v$sgastat, v$pgastat, v$rollstat, v$undostat, v$filestat, v$latch and v$session_wait.

Some Tactical Issues in Collecting work load statistics

Load Imposed On the Database System By Collection of Statistics

The collection of statistics is work in itself and as such will impose a load on the system. However, the load imposed on the system by collecting statistics is insignificant compared to the overall production workload on many systems. The enormous value and insight provided by system statistics offsets the load imposed on the system by its collection. In other words, collection of statistics in a complex system is invaluable.

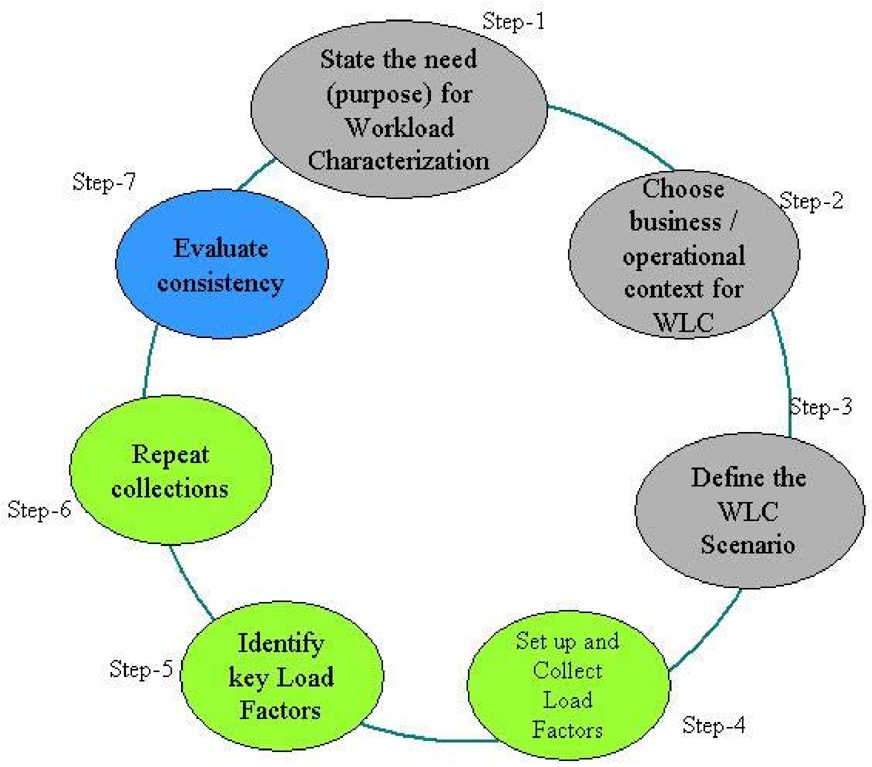
Statistics Collection Strategy

As stated earlier, setting up an infrastructure for collection of statistics depends on the objectives for such collection. For example, if the objective is to simulate and understand batch load processes that occur at evening times on a given system, then we need to collect statistics at that time. On the other hand, if the goal is to understand and predict online activity that occurs very early in the morning, when the database experiences high login rates, then we should collect statistics during that time. The key message here is to emphasize the importance of appropriate data collection strategy and to point out that it is just as important as data analysis and interpretation.

There may be cases where a clear delineation of system load is not possible, under such circumstances the statistics need to be collected on a continuous basis and subjected to characterization that determines peak activity.

Refer to Figure 2, for a comprehensive plan for a statistics collection strategy.

Fig-2: Plan for statistics collection strategy


Step-1
State the need (purpose) for Workload Characterization
Step-2
Choose business / operational context for WLC
Step-3
Define the WLC Scenario
Step-4
Set up and Collect Load Factors
Step-5
Identify key Load Factors
Step-6
Repeat collections
Step-7
Evaluate consistency

Step-1: State the need for workload characterization (WLC): Why do you want to characterize load? Is it for determining optimization scope? Or for capacity planning? Or for simulating load conditions for a test? Or for estimating response times for end users?
Step-2: Choose business context: Estimating the impact of adding another company of 150 OLTP users? Measuring the effect of adding a DWH ETL process on top of current usage on the OLTP system?
Step-3: Define WLC Scenario: The scenario comprising 40 Order Entry clerks and 10 shipping staff? 10 designers plus 50 manufacturing engineers? The scenario determines the load composition.
Step-4: Set up and collect Load-factors: You have several options here. Use Oracle supplied Statspack tool to set up collections, or alternatively write your own simple code to SELECT periodically from v$sysstat and v$system_event views INTO a user defined table for a later analysis. The reporting interface provided through statspack is a little incomplete for our purpose, since it generates difference analysis only between two snapshots, where as, we may want to trend the statistics over a number of snapshots. Also the statspack report generates a very large analysis that may not be required for our present use. Yet another option is to develop your own queries against statspack tables.
Step-5: Identify Key Load-factors – more on this step later!
Step-6: Repeat collection- One reading is never complete. We have to cater to anomalies and hence repeated multiple collections.
Step-7: Evaluate consistency- Check for changes and divergence patterns. We will touch more on this aspect in the next chapter.

Distinguishing LOAD from EFFECT

A database environment is a complex knitting of related and unrelated elements interacting with each other to produce a load effect. Therefore, understanding the distinction between ‘LOAD’ and ‘EFFECT’ is relevant. It is particularly important to understand the techniques discussed here.

Load and Dimensions: Oracle system statistics reflect and represent system load. The differences (deltas) of each statistic represent the amount of work done, albeit, in a singular dimension. For example, a number like 3,000,000 SORTS (ROWS) done in a system during the last one hour represents the workload of that system in the SORTS (ROWS) dimension. Similarly, a workload of having to perform 275,000 PHYSICAL READS in the same time frame represents another dimension of the same system load. Based on the preceding example, it can be observed that workload in a system essentially is multi-dimensional and is a complex non-linear addition of them. Therefore Oracle Statistics can be considered as ‘load-factors’. These load-factors represent distinctive load dimensions and when grouped together, they signify a load-set.

Load Effects: Oracle wait events are significant in observing the resultant effects on the system behavior under a given load. Oracle, being a management system, sets up service counters for each and every resource and monitors their usage. Monitoring insures proper and legitimate queuing up for the resources within the domains of transactional integrity rules. The types of queues and amount of waiting will fall into different categories (enqueues, child latches, etc.) and will also vary in terms of waited time (nanoseconds to microseconds to seconds).

Database Environment Specific Wait Events

Depending on varying load conditions and application behavior, many different types of wait events can occur in databases. However, some of them are more common than the others. In other words, an OLTP database will predominantly spend most of its time waiting on certain specific resources (latch free, log file sync, db file scattered read etc.) to be available which will be reflected in its wait interface, which may be different for a data warehouse (DWH) where parallel query waits are more common (PX Deq: Execute Reply, PX Deq: Execution Msg etc.).

Oracle System Statistics and Database Operations

This section builds on load-factors discussed in the earlier sections and approaches load management in a quantitative manner. The relationship between database internal operations in terms of system statistics is presented here. The table below (Table 1) briefly indicates the name and importance of some of these statistics.

Observe that, Table-1 is sorted by the ‘class’ of statistics: a convenient grouping to represent sub-areas of activity in the database. The last three columns in the table state whether the statistic is important for OLTP or DWH environment and the reason for its importance.

Table-1: Classes of Statistics

CLASS	NAME	For OLTP	For DSS	Reason for Importance
1	logons cumulative	X		User Traffic- Log-Ins are expensive
1	opened cursors cumulative	X		Library Cache Load
1	user commits	X	X	In conjunction with Rollbacks reveals utility and/or Wastage of database write work
1	user calls	X	X	Relates to work intensity across programs
1	session connect time	X	X	Resource Quota guidance
1	session uga memory	X	X	Memory usage pattern
1	session pga memory	X		Memory usage pattern
1	CPU used by this session	X	X	Get hold of resource hogs
1	recursive calls	X	X	Recursive loads in a non(limited) PL/SQL environment point to extraneous un-planned for work worthy of investigation
1	session logical reads	X	X	Provide idea of amount of data blocks usage
1	bytes received via SQL*Net from dblink	X		Usage of DB Links needs to be tailored for applications and highly optimized on time-windows and needs
1	bytes received via SQL*Net from client	X		Traffic study across client connections may provide valuable clues to software topology and process execution architecture
2	redo entries	X	X	Redo operations yield valuable information on the database usage intensity across time periods and user groups. Also useful for fixing resource group quotas
2	redo size	X	X	Redo operations in DWH environments are good candidates to optimize ETTL processes and Temporary segment designs
4	enqueue waits	X		Enqueues reveal contentious resources and patterns of usage
8	total file opens	X	X	Direct file based operations in an OLTP environments are expensive due to context switches and other system calls and file system resources like locks and permission nuances. DWH systems include usage of External table operations in this category which requires careful watching for overuse and non-optimized accesses
8	consistent changes	X	X	Contention signals!
8	DBWR checkpoints	X	X	Check for optimal balancing of check pointing between Recovery optimization and performance optimizations
8	DBWR buffers scanned	X	X	Block buffer nuances
8	DBWR summed scan depth	X		Block buffer demand for reuse points to contentious blocks and possible row-density optimization
8	physical writes	X	X	This statistic reflects ‘physical’ writes from Oracle perspective and may actually involve writes o OS and/or Disk system caches. The ‘warmness’ of a disk system cache and the algorithms used in such system are internal and specific to your disk system suppliers.
8	CR blocks created	X		Read contention with simultaneous writes is a crucial application behavior perspective.
32	queries parallelized	X	X	Parallel operations in DWH systems are dealt with in relevant section. In OLTP applications too many parallel operations should ring alarm bells!
32	Parallel operations downgraded to serial	X	X	Indicative of excessive parallelism and/or inadequate number of CPUs in the system
32	global lock releases	X	X	Crucial for Parallel server systems
64	index fast full scans (full)	X	X	In applications demanding sub-second responses even Index scans can be expensive!
64	sorts (rows)	X	X	Limited Sort activity in OLTP systems is tolerable

64	table fetch by rowid	X	X	In DWH systems with Bitmap index/join-index operations this statistic may be misleading, for bitmap operations follow a separate mechanism

Load Patterns, Load Dimensions and System Statistics

The following chart (Fig:3) shows the load patterns of a system of overlapping Oracle databases, with a particular focus on “physical reads” as a load-factor. The subsequent chart (Figure 4) shows the SORTING activity in the system during the same time period, which is another load-factor.

Figure-3: Intensity of Physical Reads during the course of a day, plotted over several days

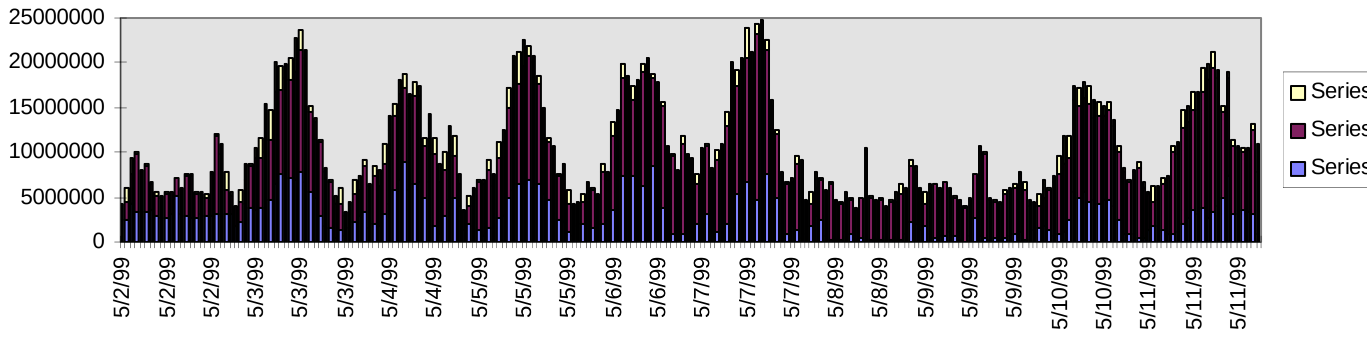
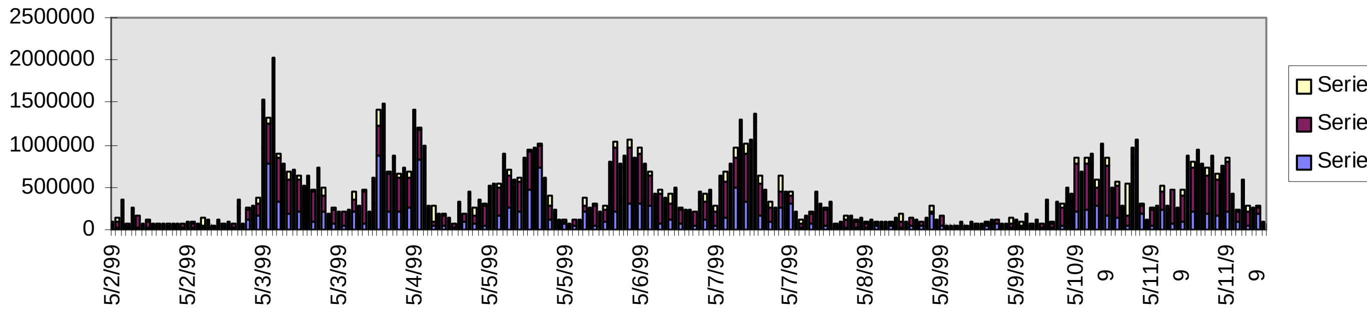


Figure-4: Intensity of Sorts(Rows) during the course of a day, plotted over several days



Repetitive patterns observed here (Figure 2 and 3) illustrate the ‘steady state’ aggregate behavior in the applications. Looking into smaller time periods, for example, hourly or half-hourly numbers, indicates load intensities across specific time frames. The shapes of different load-factors (statistics) tend to differ based on the activity and the load dimension that they reflect.

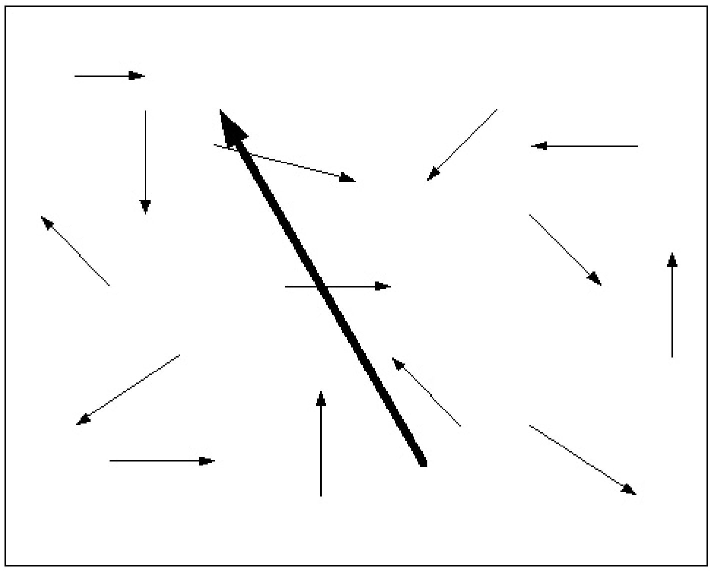
Note that the system behavior or load pattern is observed based on load-factors and not cumulative CPU utilization or Memory usage.

Extracting a Meaningful Correlation Between Load-factors

This section explores the possible relationship between various load-factors and attempts to illustrate a complex work area that is comprised of multiple such dimensions.

Consider a vector space as in figure 5. There are a number of vectors and each one of them has its own direction and magnitude. The dark arrow is the resultant vector, meaning a composite of all these directions and magnitudes.

Fig-5. Vectors in a vector space



Load-factors in an Oracle database system are analogous to the vectors in the vector space and observed workload is a combination of several load-factors. The constituent load-factors in a complex application environment are many and divergent. Resource demands by various processes within a database, constantly changes and with it the dimension of the resulting load-factors. This in turn will change the resulting overall magnitude and dimension.

Positive correlation is used to represent direction of movement. In other words, they increase or decrease in lock step fashion with the load scenario. This phenomenon can also be called as a uniform variance, making it measurable as a co-variance statistic.

Identifying the load-factors: Identification of load-factors that display independence (but still influent) is required to develop mathematical models for the load in the system. Once identified, note the weights and scales of the load-factors that exhibit correlation.

This in turn will aide in the development of meaningful relationships for the load-factors. For example, as the number of COMMITS increase, BLOCK CHANGES will tend to increase.

Load-factors and load scenarios: For specific load scenarios specific load-factors attain importance. The strength of the load model is displayed by the proportion of load that could be strongly explained or predicted by the matrix of such load-factors. For example, USER CALLS, USER COMMITS, PHYSICAL WRITES, and CPU USED BY THIS SESSION will all show strong correlation with increased workload on a given system.

Relative movements between the load-factors: Identifying relative movements between the load factors is an important step to determine whether a primary to secondary relationship exists. It is possible to have load factors (primary) that influence other load-factors (secondary). For example USER CALLS and PARSE COUNT (TOTAL) as primary load-factors will increase RECURSIVE CALLS and CPU USED BY THIS SESSION which are secondary load factors.

Natural Groupings of Load-factors

Interesting observations can be made on the relative behavior of the load-factors in varying situations. What is referred here as ‘groupings’ are technically termed ‘factors’ in a quantitative research.

Relationships on Similar Physical Activity (Class of a statistic)

Load factors that are related to each other, i.e., the oracle defined class of statistic (v$sysstat) represents the same operational area inside the database and provides insight on different aspects of the operation. Following table lists database statistics representing REDO activity (class 2):

Table 2. Redo Statistics

Class 2 Statistics
redo blocks written
redo buffer allocation retries
redo entries
redo log space requests
redo log space wait time
redo ordering marks
redo size
redo synch time
redo synch writes
redo wastage
redo write time
redo writer latching time
redo writes

The above grouping is a convenient way to classify the load-factors into an ‘accounting group’, but for the purpose of load analysis this will not suffice.

Relationship based on ‘related’ set of activities

The grouping alluded to, is the set of load-factors that signify the ‘WRITE’ aspect of the environment. For example the write activity is reflected in:

- DBWR related activity
- REDO activity
- PHYSICAL Writes
- UNDO segment activity
- DB Block changes
- Transaction table writes.

This section concludes by pointing out that analyzing load-factors helps to derive logial relationships among them. This in turn can be harnessed to model workload of a system. The next section presents analysis techniques to determine these relationships.

Regression Analysis to Determine Relationships

Statistical regression modeling helps to build a mathematical model based on cause and effect. In analyzing load-factors, this approach yields better results for the following reasons:

1. Traditional Variance analyses are inappropriate to analyze a complex system such as an Oracle database, which has layers of functionality. These layers represent different technical operational structures are independent but at times follow a daisy chain like activity-precedence approach. Independence is prevalent in the distinctness of each activity of the system--consistency management, bulk or highly selective index fetching, redo guarantee and undo management, etc. Interdependence becomes obvious in layering of functionality such as physical disk reads as a result of failure to cache a data block, creation of a consistent read (CR) block to satisfy the SELECT needs on dirty and uncommitted block, etc.

2. The load behavior is therefore not conducive for a straightforward variance analysis as here we are not dealing with a bunch of linear (or non-linear) deterministic set of variables. Oracle load-factors exhibit both determinism as well as spontaneity. One may wonder about the futility of an exercise to characterize load and model it due to these reasons. The answer lies in the fact that a subset of the load variables are deterministic, based on the physical and logical architecture of the application and database, the hardware provisioned for the system and the business and the nature of the software execution plans fulfilling business objectives.

This approach to determine logical relationship is explored further in the ensuing sections.

Simplicity of Regression Models

Statistical regression analysis offers a simplistic solution by:

- Offering a conditional set of relationship between all relevant variables validated as constituent elements in determining a load level—as both dependent and non-dependent variables.
- Strength of the model is quantified through the R-Squared value of the regression model. This may also be construed as a reliability measure for the regression output (equation).

Identifying Clustering Load-factors by Non-Linear Regression

This section explores approaches to build load-factor relationships and there strengths using statistical regression modeling. The following table (Table.3) is an illustration of a multi-variable regression conducted to predict the rate of CR Block creation in a production system, as related to a number of other independent load-factors. The high R-Squared value (0.93) and the low F-Statistic value (2.38 E –44) signify strong relationship as determined by the model.

Table-3: Illustration of a multi-variable regression.

SUMMARY OUTPUT		**CR BLOCKS CREATED**						
Regression Statistics								
Multiple R	0.964626							
R Square	**0.930504**							
Adjusted R Square	0.92151							
Standard Error	0.092282							
Observations	97							
ANOVA								
	Df	SS	MS	F	Significance F			
Regression	11	9.691956	0.881087	103.4625	**2.38E-44**			
Residual	85	0.72386	0.008516					
Total	96	10.41582						
	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%	Lower 95.0%	Upper 95.0%
Intercept	-3.10908	0.971842	-3.19916	0.001938	-5.04136	-1.17679	-5.04136	-1.17679
X Variable 1	-1.63339	0.225192	-7.25333	1.76E-10	-2.08113	-1.18565	-2.08113	-1.18565
X Variable 2	-0.01199	0.041317	-0.29028	0.772311	-0.09414	0.070156	-0.09414	0.070156
X Variable 3	-0.15072	0.065202	-2.31161	0.023219	-0.28036	-0.02108	-0.28036	-0.02108
X Variable 4	-5.02366	1.662474	-3.0218	0.00332	-8.32911	-1.71821	-8.32911	-1.71821
X Variable 5	-0.64851	0.237838	-2.7267	0.007769	-1.1214	-0.17563	-1.1214	-0.17563
X Variable 6	2.722918	0.158554	17.1734	2.25E-29	2.407669	3.038167	2.407669	3.038167

X Variable 7	0.383586	0.123069	3.116834	0.002494	0.138891	0.62828	0.138891	0.62828
X Variable 8	-0.00618	0.186118	-0.03318	0.973609	-0.37623	0.363877	-0.37623	0.363877
X Variable 9	5.524968	1.780838	3.102455	0.002605	1.984184	9.065753	1.984184	9.065753
X Variable 10	0.111868	0.04582	2.441486	0.016701	0.020766	0.20297	0.020766	0.20297
X Variable 11	-0.09114	0.063939	-1.42541	0.157701	-0.21827	0.035989	-0.21827	0.035989

Although regression modeling is typically employed to determine the cause-effect relationship between a set of variables, it can also be used to establish the relevance and strength of such relationships. For example, consider the following regression relationship taken from the above example:

“CrVAL = -3.109 + [X1 *(-1.633)] +...+ [X6 * (2.72)] +….[X9 * (0.056)]…

It is evident that the variable X1 and X6 exhibit larger influence in determining the variable CrVAL than X9. This is by virtue of the regressin coefficients- (-1.633) and (2.72) being several times higher than that of X9 (0.056). Another fact is the inverse relationship between CrVAL and X1 (Negative sign).

Using Logarithmic values

Load-factor relationships are not always linear due to their complex interactions with one another and therefore non-linear regression techniques need to be employed. Linear regressions are easy to predict and equate relative to non-linear models. Since we are dealing with a large number of variables and the relation between them is not known mathematically, we can safely assume non-linear (even curvilinear) equations. In order to simplify the regression model, we then use the logarithmic values of the variables. Logarithmic values reduce the non-linearity to linearity between log predicates. Once the prediction is made, we then Inverse-log the predicate values to derive our numerical constants. Although it is a round about way, the models are more reliable because they address both linear and non-linear behavior.

Deploying Workload Characteristics for Capacity Planning

This section illustrates the use of load-factors to compute CPU utilization. This exercise involves two distinct steps – (1)- of computing basic resource requirements and (2)- of estimating the response times for end user transactions. These steps are illustrated in tables 4 and 5 in the following sections. Note that these illustrations serve as examples for putting workload characterization to use and do not serve as a tutorial for capacity planning.

Step-1: Compute basic resources requirement

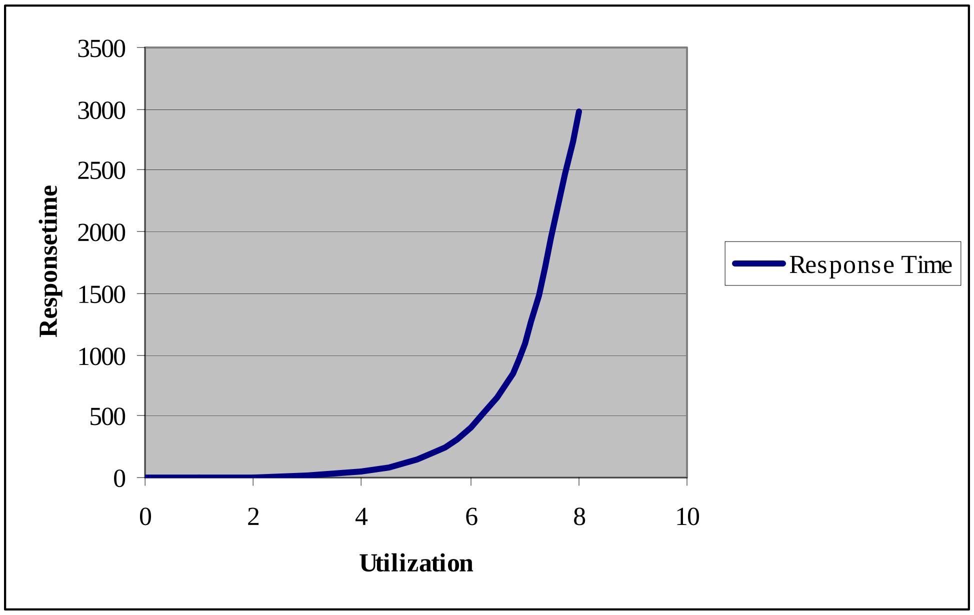
The following table (Table-4) contains CPU related load-factors. Observation of the table data shows workload scenarios 2 and 3 have been developed from scenario-1 through regression models and related extrapolations, to represent possible loads. Here we are looking at CPU as the main resource being planned for, with the appropriate load factors that effect CPU utilization directly. The utilization is computed as the CPU time used against the load capture window of 10 minutes (600 seconds). Note the simulated # of jobs in the last line as a count of 'virtual users' that induced the extrapolated amount of workload in the system.

Table-4:

All values gathered and regressed over 10 Minute intervals			
Regression	Scenario-1	Scenario-2	Scenario-3
stat name	**CPU RELATED**		
USER CALLS	1684	17777	129339
CURSOR AUTHENTICATIO	68	598	4383
EXECUTE COUNT	454	4588	32905
OPENED CURSORS CUMUL	70	620	4539
PARSE COUNT	84	703	5092
RECURSIVE CALLS	97	532	3536
SORTS (ROWS)	14470	15563	26063
CPU Used (seconds)	75.3	284.1	2218.3
Number of CPUs	1	1	6
Projected Utilization	12.55	47.35	61.62
CPU used per sec	**0.1255**	**0.4735**	**3.6971667**
Simulated # of Jobs/Sec	**1**	**4**	**29.00**

Step-2: Computing the response times for jobs running in the system:
One should be aware that response times in a system vary exponentially with the utilization of the resources in the system. See the graph below:

Figure 6. Response time utilization graph



The mean response times can be computed from the utilization values derived above by developing a queuing model. See Table- 5 (Column M stands for the number of resource counters, CPUs in this simulation).

This table (Table 5) shows the response times developed for the above load system through a M/M/1 queuing model. One can develop such models through guidance from any standard book on Queuing Theory. Observe the following while reading this table.

1. Recollect that our goal is to simulate system usage for 29 virtual users; the above simulation shows the performance numbers up to 31 users, towards the bottom of the chart in the 24th row.
2. The second column is the service rate, a constant number per device. The only way to increase throughput is by increasing devices, in this case the number of CPUs, shown in column 'm'.
3. The third column- λ is the real variable here. By increasing the arrival rate of jobs (in this case virtual users hitting the system on a per second basis), we compute the other dependent results like- probability of 'n' jobs in queue, Utilization, Response and waiting times, jobs in queue, etc. Utilization increase results in exponential increase of response times (see the graph above) become too large, we add a CPU and bring down the utilization by sharing the queues.
4. Observe how the response times grow in simulations 7,8, 14- 16. The response times increase by an order of 600%. The system users are going to panic. These are the times where the system administrators and managers will have to decide to augment resources. The simulations provide a way to foresee what the users will experience. The system utilization is close to 90% at these times.

5. Notice how the job queue builds up during these scenarios to double digit numbers and remember that these are bundles of virtual users.

Table 5. Response times developed through a M/M/1 queuing model.

Simulation Number	μ	λ	ρ	m	Probability of zero jobs in Queue	Prob for 5 jobs	Prob for 20 jobs	Jobs in Queue	Utilization	Mean Response time	Mean waiting time
1	8	1	0.125	1	0.875	3.34E-06	1.02E-10	0.017857	12.5	0.142857	0.017857
2	8	4	0.5	1	0.5	0.007813	0.000244	0.5	50	0.25	0.125
3	8	6	0.75	1	0.25	0.044495	0.010559	2.25	75	0.5	0.375
4	8	8	0.5	2	0.5	0.007813	0.000244	0.5	50	0.25	0.125
5	8	12	0.75	2	0.25	0.044495	0.010559	2.25	75	0.5	0.375
6	8	13	0.8125	2	0.1875	0.053944	0.019101	3.520833	81.25	0.666667	0.541667
7	8	14	0.875	2	0.125	0.056099	0.028774	6.125	87.5	1	0.875
8	8	15	0.9375	2	0.0625	0.042433	0.03073	14.0625	93.75	2	1.875
9	8	16	0.666667	3	0.333333	0.029264	0.003854	1.333333	66.66667	0.375	0.25
10	8	17	0.708333	3	0.291667	0.036839	0.006569	1.720238	70.83333	0.428571	0.303571
11	8	18	0.75	3	0.25	0.044495	0.010559	2.25	75	0.5	0.375
12	8	19	0.791667	3	0.208333	0.051288	0.015949	3.008333	79.16667	0.6	0.475
13	8	20	0.833333	3	0.166667	0.055816	0.022431	4.166667	83.33333	0.75	0.625
14	8	21	0.875	3	0.125	0.056099	0.028774	6.125	87.5	1	0.875
15	8	22	0.916667	3	0.083333	0.049441	0.032	10.08333	91.66667	1.5	1.375
16	8	23	0.958333	3	0.041667	0.032277	0.02609	22.04167	95.83333	3	2.875
17	8	24	0.75	4	0.25	0.044495	0.010559	2.25	75	0.5	0.375
18	8	25	0.78125	4	0.21875	0.049738	0.014476	2.790179	78.125	0.571429	0.446429
19	8	26	0.8125	4	0.1875	0.053944	0.019101	3.520833	81.25	0.666667	0.541667
20	8	27	0.84375	4	0.15625	0.056377	0.024109	4.55625	84.375	0.8	0.675
21	8	28	0.875	4	0.125	0.056099	0.028774	6.125	87.5	1	0.875
22	8	29	0.725	5	0.275	0.039936	0.007999	1.911364	72.5	0.454545	0.329545
23	8	30	0.75	5	0.25	0.044495	0.010559	2.25	75	0.5	0.375
24	8	31	0.775	5	0.225	0.048752	0.01363	2.669444	77.5	0.555556	0.430556

Conclusion

This paper presents an introduction and an alternative approach to quantitative modeling of load and performance in Oracle systems. Load metrics comprising of Oracle system statistics and Oracle wait events are useful in presenting an internal view of the load and resultant resource based queues. Capturing these statistics helps in developing workload characteristics, which in turn are useful in evolving capacity models for the underlying architecture and environment.

References

[1]	Raj Jain, 1991, “The Art Of Computer System Performance Analysis”, John Wiley & Sons.
[2]	Daniel A. Menasce, Virgilio A.F. Almedia, 1998, Prentice Hall.
[3]	Hennessey and Patterson; ‘*Computer architecture- A Quantitative Approach’*; Morgan Kaufmann
[4]	Killelea, Patrick- ‘Web Performance Tuning’; O’Reilly
[5]	Gunther, Neil J.; *‘The Practical Performance Analyst’*; McGrawHill.