

# DBaaS Cloud Capacity Planning - Accounting for Dynamic RDBMS Systems that Employ Clustering and Standby Architectures

Antony S. Higginson  
Oracle Advanced Customer Support Services  
Didsbury, Manchester, UK  
antony.higginson@oracle.com

Suzanne M. Embury  
School of Computer Science  
University Of Manchester  
Manchester, M13 9PL, UK.  
suzanne.m.embury@manchester.ac.uk

Norman W. Paton  
School of Computer Science  
University Of Manchester  
Manchester, M13 9PL, UK.  
norman.paton@manchester.ac.uk

Clive Bostock  
Oracle Advanced Customer Support Services  
Didsbury, Manchester, UK  
clive.bostock@oracle.com

## ABSTRACT

There are several major attractions that Cloud Computing promises when dealing with computing environments, such as the ease with which databases can be provisioned, maintained and accounted for seamlessly. However, this efficiency panacea that company executives look for when managing their estates often brings further challenges. Databases are an integral part of any organisation and can be a source of bottlenecks when it comes to provisioning, managing and maintenance. Cloud computing certainly can address some of these concerns when *Database-as-a-Service* (DBaaS) is employed. However, one major aspect prior to adopting DBaaS is Capacity Planning, with the aim of avoiding *under-estimation* or *over-estimation* of the new resources required from the cloud architecture, with the aim of consolidating databases together or provisioning new databases into the new architecture that DBaaS clouds will provide. Capacity Planning has not evolved sufficiently to accommodate complex database systems that employ advanced features such as Clustered or Standby Databases that are required to satisfy enterprise SLAs. Being able to efficiently capacity plan an estate of databases accurately will allow executives to expedite cloud adoption quickly, allowing the enterprise to enjoy the benefits that cloud adoption brings. This paper investigates the extent to which the physical properties resulting from a workload, in terms of CPU, IO and memory, are preserved when the workload is run on different platforms. Experiments are reported that represent OLTP, OLAP and Data Mart workloads running on a range of architectures, specifically single instance, single instance with a standby, and clustered databases.

This paper proposes and empirically evaluates an approach to capacity planning for complex database deployments.

## Keywords

Cloud, DBaaS, Capacity Planning, Database, Provisioning, Standby, Clustering

## 1. INTRODUCTION

Traditionally, companies accounted for the cost of assets associated with their I.T. using Capex (Capital Expenditure) type models, where assets such as hardware, licenses and support, etc, were accounted for yearly. For example, a software license usually is based on an *on-premises* model that would be user based, or by the CPU if the application served many thousands of users. The advent of Cloud computing, with the *pay-as-you-go* subscription based model, has changed the way company executives look at the costing models of their I.T.

A similar paradigm unfolds when I.T. departments such as Development, Delivery and Support teams need to provision environments quickly to meet their business goals. Traditional project methodologies would request environments aiding development and testing with the goal of going live. Procurement and provisioning took time that was often added to the project lifecycle. Cloud computing addresses such issues so that a user can, with ease, request the rapid provision of resources for a period of time.

Once the system went live those Delivery and Support teams would then need to account for resources those particular systems consumed, reconciling with the Line Of Business (LOB). The results of that analysis would then feed back into next year's Capex model. This ongoing capacity planning to assess if they have enough resources is needed to ensure is that, as systems grow, there are enough resources to ensure that the system is able to meet QoS (Quality of Service) expectations. Cloud Computing has also made some advances here by enabling a metering or charge-back facility that can accurately account for the resources used (CPU, Memory, Storage). Cloud Computing can dynamically modify the cloud to reduce or increase those resources

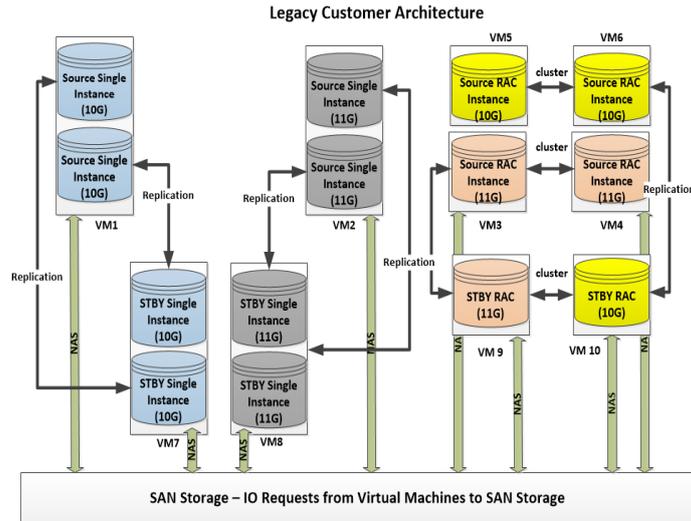


Figure 1: Example Architecture: Typical customer legacy database architecture.

as needed by the client.

However, companies with large estates have the additional challenge of having a plethora of database versions, for example, each database version offering a different feature that has a performance benefit over another database version. Similarly, the databases may be running on an eclectic set of operating systems and hardware, each affecting the workload in a subtle or major way. For example, the latest running version of a database may run on a highly configured SAN utilising the latest techniques in query optimization and storage. Comparing this footprint with an older version of software and infrastructure often leads to a *Finger-in-the-air* type approach.

A key feature of DBaaS is the ability to multi-tenant those databases where different workloads and database configurations can coexist in the shared resources, adding to the challenge of making effective capacity planning decisions. Determining the allocation is further complicated if the database utilises advanced features such as Clustering or Failover Technology, as workloads shift from one instance to another or are shared across multiple instances based on their own resource allocation managers. Furthermore, if a database employs a standby, this further complicates capacity planning decisions.

Cloud Computing is in its infancy, with incremental adoption within the industry as companies try and determine how to unpick their database estates and move them to cloud infrastructure. Databases often grow organically over many years in terms of their data and complexity, which often leads to major projects being derived when a major upgrade or re-platform exercise is required. With the introduction of cloud these exercises are becoming more prudent. This often leads to a series of questions on Capacity Planning.

- What is the current footprint of the database including any advanced features such as Standby or Clustering?
- What is the current configuration of the database?
- What type of DBaaS should I create?

- What size of DBaaS should I create?
- Can I consolidate databases that have similar configuration and utilisation foot-prints?
- Will my SLAs be compromised if I move to a cloud?

Such questions become very important prior to any provisioning or migration exercise.

The time taken to perform this analysis on databases also has a major impact on a company's ability to adopt cloud technologies often squeezing the bandwidth of the delivery and support teams. The departments suffer *paralysis-by-analysis*, and the migration to the cloud becomes more protracted to the frustration of all involved. If the analysis is not performed accurately then the risks of *over-estimation* and *under-estimation* increase. Being able to automate the gathering of data, analysing the data and then making a decision becomes ever more important in enterprises with large estates.

In this paper we look at the challenges of Capacity Planning for advanced database systems that employ clustering and standby databases, with a view to migration to a cloud. Our hypothesis is: "That a model based on physical measures can be used to provide dependable predictions of performance for diverse applications". We make two main contributions:

1. We propose an approach to workload analysis based on physical metrics that are important to capacity planning for database systems with advanced configurations.
2. We report the results of an empirical analysis of the metrics for several representative workloads on diverse real-life configurations.

The remainder of the paper is organised as follows. Section 2 introduces the Background and Related Work. In Section 3 we detail the environmental setup for conducting experiments outlining the database capacity planning problem. In Section 4 we introduce our solution in detail and

provide details on the experiments and analysis. Section 5 gives conclusions and future work.

## 2. BACKGROUND AND RELATED WORK

### 2.1 Background

Fig 1 shows an example environment of a company that is running different versions and configurations of databases on VM hardware. Physical machines are dissected into 10 VM's giving a level of separation. On these 10 VM's a total of 12 databases are run, of which 6 are primary databases and 6 are standby databases. This MAA (*Maximum Availability Architecture*) allows the company some comfort by running their primary (*Platinum*) SLA level applications on VM numbers 3, 4, 5 and 6, which host two clustered databases (offering a degree of resilience against node failure). In addition, these clustered databases have a physical standby database running on VM's 9 and 10 in case of database failure or corruption. Similarly, the 4 single instance stand alone databases that are running on VM's 1 and 2 also have a replicated standby database running on VM's 7 and 8, again offering the company some comfort that their secondary (*Gold*) level of applications will have a standby database for failover, should they need it.

The company also wish to increase their ROI (*Return on Investment*) with this environment and thus often open up the standby databases in "Read Only" mode during special times for applications that need to run year-end or month-end type BI (*Business Intelligent*) reports. This particular type of architectural pattern is a typical configuration companies use today to manage their database environments and applications that have 24\*7 type SLAs. The difficulty becomes apparent when a new exercise is introduced that looks at consolidating, upgrading and migrating those environments listed in Fig 1 to a new cloud architecture, where resources can be tightly accounted and dynamically assigned. We are then faced with a capacity planning exercise.

### 2.2 Related Work

The objective of capacity planning is to provide an accurate estimate of the resources required to run a set of applications in a database cloud. Achieving this answer relies on the accurate capture of some base metrics, based on historical patterns, and applying some modelling techniques to form a prediction. There are two main viewpoints: the viewpoint of the CSP (*Cloud Service Provider*) in what they offer and their capabilities, i.e are there enough resources to provide services to consumers; and the viewpoint of the consumer, for example, can a customer capacity plan their systems against the CSP's capability? Indeed if the customer wishes to become a CSP but in a private cloud configuration, then the first viewpoint also becomes important.

A CSP offers resources, and existing models use various techniques to help customers assess the CSP capabilities. MCDM (Multi Criteria Decision Making) weighs the attributes of an individual database by their importance in helping to choose the right cloud (Mozafari et al 2013 [16] and Shari et al 2014 [19]). CSP's can also be assessed using a pricing model to validate their capability based on a consumers single systems workload as suggested by (Shang et al [20]); using this financial approach contributes to the *value-for-money* question that many enterprises seek when deciding on the right cloud.

If a consumer has a cloud, knowing where to place the workload based on utilisation to achieve the best fit is critical when beginning to answer the QoS (Quality of Service) question, and techniques such as *bin-packing* algorithms (Yu et al [21]) help achieve this answer. However systems may have dynamic workloads, which may evolve organically as datasets and/or numbers of users grow or shrink, as is especially common in internet based systems. There is a need for constant assessment of said workloads. Hacıgumus et al [10] and Kouki et al [11] both look at the workload of an application or the query being executed, and then decide what type of database in a cloud would satisfy QoS. Mozafari et al [15] suggests using techniques that capture log and performance data over a period of time, storing them in a central repository, and modelling the workloads at a database instance level. With the advent of Virtualisation that enterprises utilise, including CSP's, when running their estates, several techniques such as coefficient of variation and distribution profiling are used to look at the utilisation of a Virtual Machine to try and capacity plan. Mahambre and Chafle [13] look at the workload of a Virtual Machine to create relationship patterns of workloads to understand how resources are being utilised, analysing the actual query being executed to predict if and when it is likely to exhaust resources available.

There seems to be a consensus among several academics (Shang et al [20], Lobo [12] and Guidolin et al 2008 [9]) on the need for long term capacity planning and the inadequacy of capacity planning in this new age of cloud computing using current techniques. The techniques used today assume that the architecture is simple, in that the architecture does not utilise virtualisation or advanced database features such as standby's and clustering technology, but in the age of consolidation and drive for standardisation, the architecture is not simple. Enterprises use combinations of technology in different configurations to achieve their goals of consolidation or standardisation. Most models use a form of linear regression to predict growth patterns. Guidolin et al 2008 [9] conducted a study of those linear regression models and came to the conclusion that as more parameters are added the models become less accurate, something also highlighted by Mozafari et al 2013 [15]. To mitigate against this inaccuracy more controls are added at the cost of performance of the model itself. For example, predicting the growth of several databases based on resource utilisation may become more inaccurate as the number of source systems being analysed increases, therefore requiring more controls to keep the accuracy. This is certainly interesting when trying to capacity plan several applications running on different configurations prior to a migration to a cloud. In addition, trying to simulate cloud computing workloads to develop new techniques is also an issue; Moussa and Badir 2013 [14] explained that the TPC-H [4] and TPC-DS [3] benchmarks are not designed for Data Warehouses in the cloud, further adding to the problem of developing and evaluating models.

## 3. EXPERIMENTAL SETUP

Given a description of an existing deployment, including the Operating System, Database and Applications running on that database (Activity), a collection of monitors on the existing deployment that report on CPU, Memory, IOPS's and Storage, the goal is to develop models of the existing configuration that contain enough information to allow reliable estimates to be made of the performance of a deploy-

Workload Type	Workload Profile	DBNAME(S)	Workload Description	Number of Users	Duration (hh:mi)	Avg Transaction per sec
OLTP	General usage	RAPIDKIT RAPIDKIT2 DBM01	General Online Application with updates, inserts and deletes simulate working day	100 2000 (DBM01)	23:59	0.2
OLTP	Morning Peak Logon Surge	RAPIDKIT RAPIDKIT2 DBM01	Morning Surge to simulate users logging on to the Online Application with updates, inserts and deletes	100 1000 (DBM01)	2:00	0.2
OLTP	Lunch Time Peak Logon Surge	RAPIDKIT RAPIDKIT2 DBM01	Lunch Time Surge to simulate users logging on to the Online Application with updates, inserts and deletes	100 1000 (DBM01)	1:00	0.2
OLTP	Evening Time Peak Logon Surge	RAPIDKIT RAPIDKIT2 DBM01	Evening Time Surge to simulate users logging on to the Online Application with updates, inserts and deletes	100 1000 (DBM01)	5:00	0.2
Daily OLTP Hot Backup taken at 23:00						
OLAP	Data Warehouse General Usage	RAPIDKIT RAPIDKIT2 DBM01	General Data Warehousing Application with heavy Selects taking place out of hours building Business Intelligence data	5 400 (DBM01)	8:00	0.4
Daily OLAP Hot Backup taken at 06:00						
Daily OLAP archivelog backups taken at 12:00,18:00,00:00						
DM	OLTP Usage General	RAPIDKIT RAPIDKIT2 DBM01	Combination of DML taking place during the business day and heavy DML taking out of ours	200 1000 (DBM01)	23:59	0.2
DM	OLTP Morning Peak Surge Logon	RAPIDKIT RAPIDKIT2 DBM01	Morning Surge to simulate users logging on to the Online Application with updates, inserts and deletes	100 500 (DBM01)	2:00	0.2
DM	OLTP Lunch Time Peak Logon Surge	RAPIDKIT RAPIDKIT2 DBM01	Morning Surge to simulate users logging on to the Online Application with updates, inserts and deletes	100 500 (DBM01)	2:00	0.3
DM	OLTP Evening Time Peak Logon Surge	RAPIDKIT RAPIDKIT2 DBM01	Evening Time Surge to simulate users logging on to the Online Application with updates, inserts and deletes	100 500 (DBM01)	5:00	0.3
DM	OLAP Batch Loads Peak	RAPIDKIT RAPIDKIT2 DBM01	Evening Time Surge to simulate users logging on to the Online Application with updates, inserts and deletes	5 400 (DBM01)	8:00	0.3
Daily DM Hot Backup taken at 06:00						
Daily DM archivelog backups taken at 12:00,18:00,00:00						

Table 1: Database Workloads

ment when it is migrated to a cloud platform that may involve a Single Database, a Clustered Database or Standby Databases. To meet this objective, we must find out if a workload executed on one database is comparable to the same workload running on the same database on a different host.

Our approach to this question is by way of empirical evaluation. Using increasingly complex deployments, of the type illustrated in Fig 2, and representative workloads, we establish the extent to which we can predict the load on a target deployment based on readings on a source deployment. This section describes the workloads and the platforms used in the experiments.

### 3.1 Workloads

A Workload can be described as the activity being performed on the database at a point-in-time, and essentially is broken down into the following areas:

- *Database* - An Oracle database is a set of physical files on disk(s) that store data. Data may be in the form of logical objects, such as tables, Views and indexes, which are attached to those tables to aid speed of access, reducing the resources consumed in accessing the data.
- *Instance* - An Oracle instance is a set memory structures and processes that manage the database files. The instance exists in memory and a database exists on disk, an instance can exist without a database and a database can exist without an instance.
- *Activity* - The DML (Data Modification Language)/DDL (Data Definition Language) i.e. SQL that is being executed on the database by the application, creates load consisting of CPU, memory and IOPS/s.

The monitors used to capture the data report on IOPS's (Physical reads and Physical Writes), Memory (RAM assigned to a database or host) and CPU (SPECINT's). SPECInt is a benchmark based on the CINT92, which measures the integer speed performance of the CPU, (Dixit) [6]. The experiments involve controlled execution of several types of workloads on several configurations of database. Moussa and Badir 2013 [14] describe how running of controlled workloads using TPC has not evolved for clouds, therefore we will use a utility called *swingbench* (Giles)[8] to generate a controlled load based on TPC-C [5]. The workload is generated on several Gb's of sample data based on the Orders Entry (OE) schema that comes with Oracle 12C. The OE schema is useful for dealing with intermediate complexity and is based on a company that sells several products such as software, hardware, clothing and tools. Scripts are then executed to generate a load against the OE schema to simulate DML transactions performed on the database of a number of users over a period of Hour.

### 3.2 Outline of the Platforms

Three different types of workload were created (*OLTP, OLAP and Data Mart*) as shown in Table 1. The Database is placed in archivelog mode during each execution of the workload further creating IO on the Host and allowing for a hot backup to be performed on the database. The backup acts as a 'housekeeping' routine by clearing down the archivelogs to ensure the host does not run out of storage space. This type of backup routine is normal when dealing with databases and each backup routine is executed periodically depending upon the workload.

## 4. EXPERIMENTS AND ANALYSIS

A number of experiments were conducted to investigate if a workload executed on one machine consumes similar resources when the workload is executed on another environment. The aim was to investigate what could cause dif-

VM Name	OS Type	CPU De-tails	Memory	Storage	Database Type	Products and Versions
Single Database Instance Configuration						
Virtual Machine 1	OEL Linux 2.6.39	4 * 2.9 Ghz	32Gb	300Gb	Oracle Single Instance Database (RapidKit)	<ul style="list-style-type: none"> <li>Enterprise Edition (12.1.0.2),</li> <li>Data Guard (12.1.0.2),</li> <li>Enterprise Manager Agent (12.1.0.4),</li> </ul>
Virtual Machine 2	OEL Linux 2.6.39	4 * 2.9 Ghz	32Gb	300Gb	Oracle Single Instance Database (RapidKit2)	<ul style="list-style-type: none"> <li>Enterprise Edition (12.1.0.2),</li> <li>Data Guard (12.1.0.2),</li> <li>Enterprise Manager Agent (12.1.0.4),</li> </ul>
Clustered Database Instance Configuration						
Clustered Compute Node 1	OEL Linux 2.6.39	24 * 2.9 Ghz	96Gb	14Tb	Oracle Clustered Multi-tenant Database Instance (DBM011)	<ul style="list-style-type: none"> <li>Enterprise Edition (12.1.0.2),</li> <li>Data Guard (12.1.0.2),</li> <li>Enterprise Manager Agent (12.1.0.4),</li> <li>Grid Infrastructure (12.1.0.2),</li> <li>Oracle Automatic Storage Manager (12.1.0.2),</li> </ul>
Clustered Compute Node 2	OEL Linux 2.6.39	24 * 2.9 Ghz	96Gb	14Tb	Oracle Clustered Multi-tenant Database Instance (DBM012)	<ul style="list-style-type: none"> <li>Enterprise Edition (12.1.0.2),</li> <li>Data Guard (12.1.0.2),</li> <li>Enterprise Manager Agent (12.1.0.4),</li> <li>Grid Infrastructure (12.1.0.2),</li> <li>Oracle Automatic Storage Manager (12.1.0.2),</li> </ul>
Standby Database Instance Configuration						
Virtual Machine 3	OEL Linux 2.6.39	4 * 2.9 Ghz	32Gb	1Tb	Oracle Single Instance Standby Database (STBYRapidKit, STBYRapidKit2)	<ul style="list-style-type: none"> <li>Enterprise Edition (12.1.0.2),</li> <li>Data Guard (12.1.0.2),</li> <li>Enterprise Manager Agent (12.1.0.4),</li> </ul>
Central Repository Details						
Storage Repository	OEL Linux 2.6.39	24 * 2.4 Ghz	32Gb	500Gb	Oracle Single Instance Database (EMREPCTA)	<ul style="list-style-type: none"> <li>Enterprise Edition (11.2.0.3),</li> <li>Enterprise Manager R4 including Webserver and BIPublisher (12.1.0.4),</li> <li>Enterprise Manager Agent (12.1.0.4),</li> </ul>

Table 2: Platform Outline

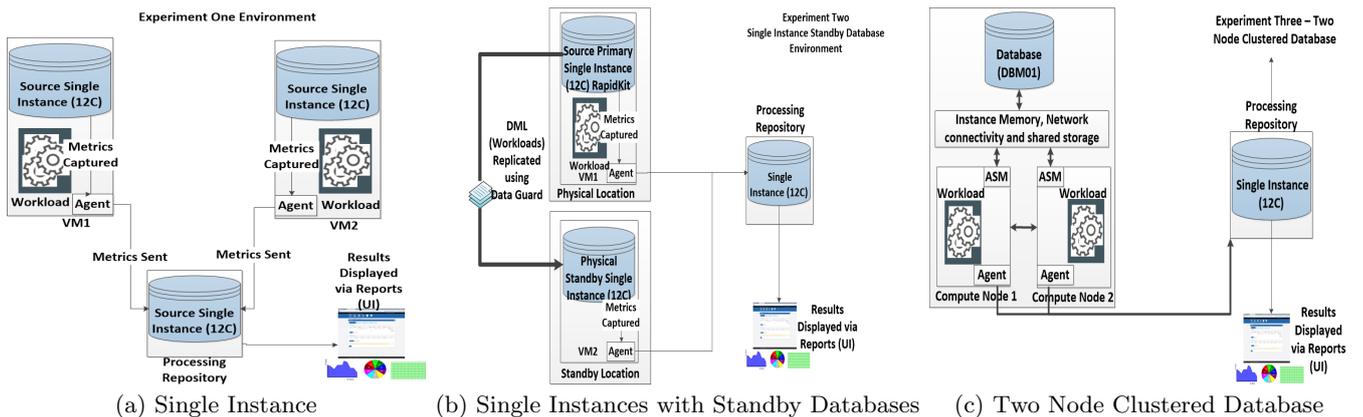


Figure 2: Experiment Architecture: different database combinations used for experiments.

ferences in the consumption of resources between workloads. The experiment focused on three types of database configuration:

- *Experiment 1* - Running three workloads (*OLTP, OLAP and DM*) on a single instance database.
- *Experiment 2* - Running the workloads (*OLTP, OLAP and DM*) on a single instance database with a Physical Standby Database.
- *Experiment 3* - Running the three workloads (*OLTP, OLAP and DM*) on a two node clustered database.

The database was always the same version between each host, the data set was always the same size to start, the workload was always repeatable in that the workload could be executed, stopped, the database reset and the same workload replayed.

## 4.1 Experimental Methodology

The experiments involve an eclectic set of hardware configured to run several different types of database as shown in Table 2. An agent polls the database instance every few minutes for specific metrics namely; Database Instance Memory, IOPS's (physical Reads/Writes) and CPU per sec. The metric results are stored in a central repository database, and are aggregated at hourly intervals. The configuration of the hardware, such as CPU Make model and SPECInt, and the database configuration are also stored in a central repository, which is then used as lookup data when performing comparisons between the performance of one workload on one database with the same workload on another database.

## 4.2 Experiment One - Single Database Instance

The first experiment was to execute three workloads on one single instance database on a virtual host (VM1) and

then execute the same three workloads on another single instance database on another virtual host (VM2) as shown in Fig 2a. The database configurations were the same in Instance Parameters, Software Version and Patch Level. The Hardware configurations were the same in OS Level, Kernel Version, and memory configuration. Some differences exist in the underlying architecture such as the Physical hardware and the Storage as these were VM's created on different physical machines. We capture the metrics for each workload and analyse the extent to which physical properties are consistent across platforms. This is shown graphically in Fig 3

### 4.3 Results and Analysis Experiment One - OLTP Workload

The results for OLTP, covering Memory, CPU and IOPS/s are shown graphically in Fig 3. These are simple line graphs from the OLTP workload shown in Table 1. It was observed that the OLTP workload from a CPU perspective had several distinguishing features. It clearly shows that the workload starts off low until the beginning of the experiment where a sudden jump takes place and the OLTP workload begins. Then there is a general plateau that relates to the 24 hour periods and at various times from there on in there are spikes.

- *CPU utilisation* - CPU over a 72 hour period was not the same between the two databases but at its largest peak (evening surge) there was a difference of approximately 300 SPECInts or +88% (day 24 hour 11) in its utilisation. The difference in utilization between the two workloads without the peaks was approximately +20%.
- *CPU Spikes (Backup)* - There were several spikes in CPU at 00:00 - 02:00 and relate to the daily hot RMAN backup that is taken for the databases.
- *CPU Spikes (Morning Surge)* - A large CPU spike was observed for several hundred users accessing the database at 08:00.
- *IOPS/s (general)* - There is a large difference in IOPS (day 23 hour 9) where the difference at peak is +88%. The difference in general usage (i.e. without the peaks) was +7%.

CPU, Memory and IOPS/s over a 72 hour period show similar traits in that the workload begins and there is a jump in the activity as the users logon. The first set of results show that even when executed on similar platforms, the metrics for the OLTP workloads can be substantially different, especially in the CPU and IOPS utilisation.

### 4.4 Results and Analysis Experiment One - OLAP Workload

The results for OLAP covering Memory, CPU and IOPS/s are shown graphically in Fig 4. The difference between the OLTP and OLAP workload is that the OLAP workload is high in Select statements and the result set is larger. The IO is representative of a Data Warehouse building cubes for interrogation by a Business Intelligence reporting tool. The execution times for the workload are also different; OLTP is fairly constant in its usage, whereas OLAP is more concentrated out of normal working hours. It was observed that

the OLAP workload runs out of hours for a period of around five hours and this matches the description shown in Table 1.

- *CPU Spikes (General Usage)* - CPU over a 72 hour period was not the same for the two databases, but at its largest peak there was a difference of only +1% (day 17 hour 05) in utilisation. Two workloads outside the peaks were essentially the same.
- *IOPS/s utilisation* - IOPS over a 72 hour period had a difference of approximately +50% in utilisation (Day 16 Hour 8); outside the peaks (Day 16 Hour 19) the utilisation is 0%.
- *IOPS/s Spikes (Backup)* - There are four backups that run during the 24 hours. Three of those backups are used as housekeeping routines that backup and delete the archivelogs; these backups are executed at 12:00, 18:00 and 00:00. One backup backs up the database (level-0) and the associated archivelogs, and this is executed at 06:00. There was no spike for 18:00 because the backup at 12:00 had removed the archivelogs and thus there was nothing to backup.

The OLAP Memory chart also showed the same characteristics as the IOPS/s and CPU charts in that there is a uniform pattern to there being a plateau and a spike over the 72 hours. Each of the databases had a memory configuration of 3.5Gb, given the OLAP workload would have had SQL requiring larger memory than 3.5Gb for sorting, thus sorts would have gone to disk rather than memory, accounting for the higher IOPS's readings in Fig 4 than in Fig 3.

### 4.5 Results and Analysis Experiment One - DataMart Workload

The results for the Data Mart covering Memory, CPU and IOPS/s are shown in Fig 5. It was observed that the Data Mart workload from a CPU perspective had several distinguishing features. It clearly shows that the workload starts off as the users connect and the workload is running, a sudden jump takes place at Day 10 Hour 3 as the Batch Loads are executed for approximately 6 hours, and this is repeated twice more throughout the 72 hours. There are also other peaks and troughs observed and these are consistent with the workload described in Table 1.

- *CPU utilisation* - CPU over a 72 hour period between the two databases and had a difference of approximately +64% during the normal day (Day 9 Hour 21). When the batch loads ran (Day 11 Hour 05) the difference in utilisation was +1%.
- *CPU Spikes (General)* - generally, the CPU utilisation between the two databases was the same, there is a difference of +1% at peak times.
- *IOPS/s Utilisation* - IOPS at peak (Day 9 Hour 21) had a difference of approximately +24%
- *Memory utilisation* - Memory was the same in general footprint however there were differences at peak times of 300mb or +4%

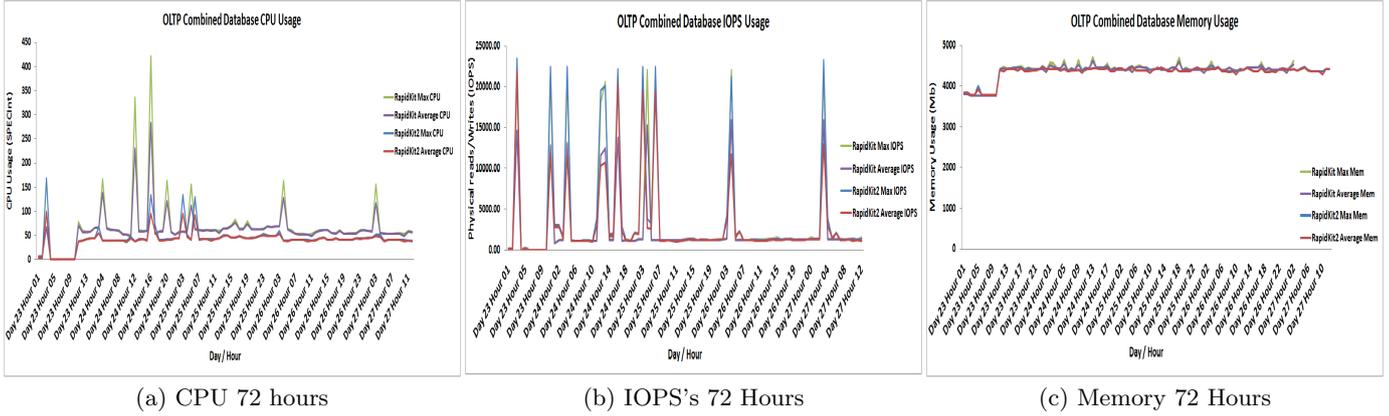


Figure 3: Results Single Instance OLTP: workload patterns for the 72 hour period.

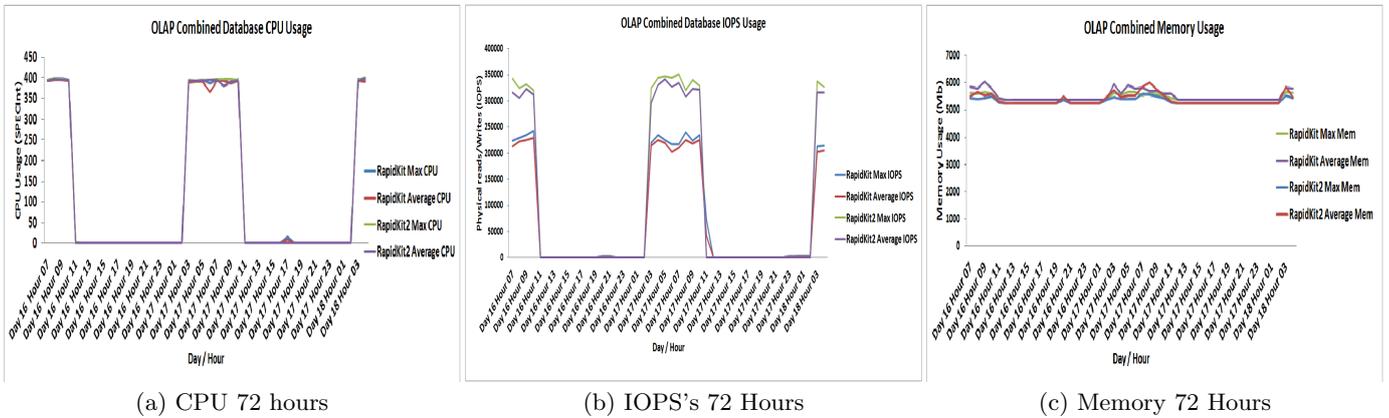


Figure 4: Results Single Instance OLAP: workload patterns for the 72 hour period.

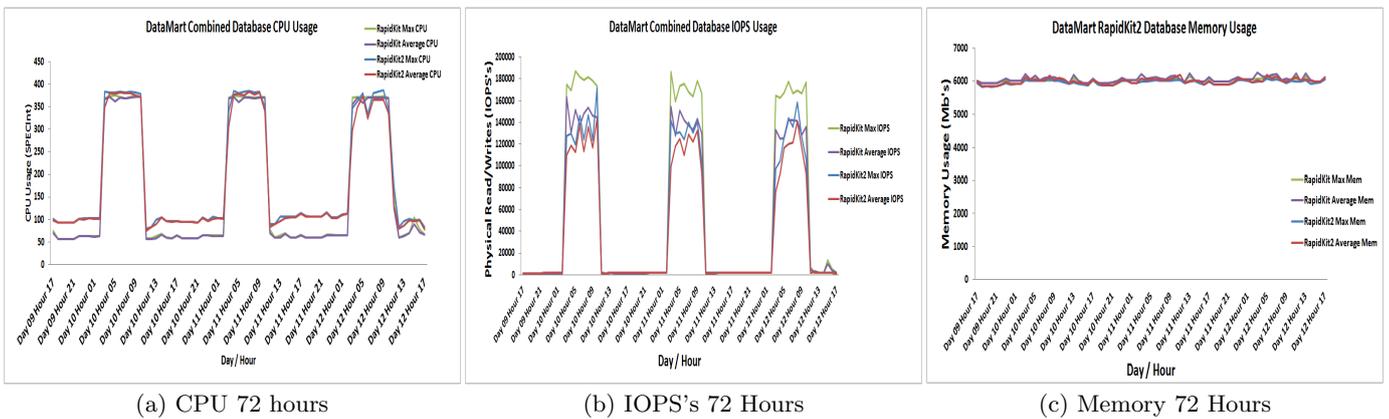
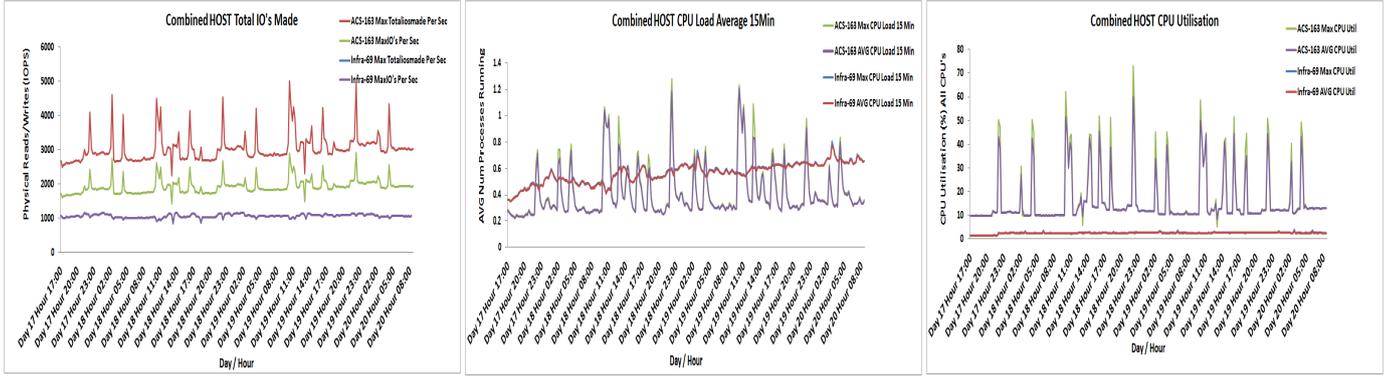


Figure 5: Results Single Instance Data Mart: workload patterns for the 72 hour period.

In general there is a difference in the VM's at a CPU level. The VM named *acs-163* has a configuration of 16 Threads(s) per core (based on the `lscpu` command) from the VM *infra-69* which only has 1 thread per core. We believe this accounts for the difference in CPU for small concurrent transactions in the OLTP workload. Each of the databases had a memory (*SGA*) configuration of 3.5Gb, if the SQL state-

ment executed in the workload requires a memory larger than 3.5Gb, which is more common in OLAP and Data Mart workloads then sorts will go to disk. Database memory configurations influence the database execution plans and optimisers and this sensitivity is reflected in the IOPS's charts shown in Fig's 3b, 4b and 5b.



(a) Host Total IO's Made 72 hours (b) Host CPU Load Avg (15Mins) 72 hours (c) Host CPU Utilisation 72 Hours

Figure 6: Results HOST Metrics OLTP: workload patterns for the 72 hour period.

## 4.6 Experiment Two - Single Instance Standby Configurations

The Second set of experiments was to introduce a more complicated environment executing one workload (*OLTP*) on a single instance *primary* database with a physical standby database kept in sync using the Data Guard technology (Oracle Data Guard [18]) across the two sites, as shown in Fig 2b. A key factor in this experiment is that the physical standby database is always in a recovering state and therefore is not opened to accept SQL connections in the same way as a normal (*primary*) database. Therefore the agent is unable to gather the instance based metrics, so we capture host based metrics to compare and contrast the workload:

- CPU load over 15mins - This is the output from the “*Top*” command executed in linux, this measurement is a number using or waiting for CPU resources. For example if there is a 1, then on average 1 process over the 15 min time period is using or waiting for CPU.
- CPU Utilisation Percentage - This is based on the “*MPSTAT -P ALL*” command and looks at the percentage of all cpu’s being used .
- TotalIOSMade - This is the total physical reads and total physical writes per 15 minute interval on the host.
- MaxIOSperSec - This is the Maximum physical reads and physical writes per sec.

The two VM’s are located within the same site but in different rooms, Data Guard is configured using Maximum Performance mode to allow for network drops in the connectivity between the two physical locations. The database configurations were the same in Instance Parameters, Software Version and Patch Level. The Hardware configurations were the same in OS Level, Kernel Version and memory configuration. We capture the metrics of each workload and analyse the consistency of the metrics, as shown graphically in Figure 6.

## 4.7 Results and Analysis Experiment Two - OLTP Workload

The results for OLTP covering CPU and IOPS/s are shown graphically in Figure 6. Relying on host based metrics has

a profound effect in the ability to compare and contrast different CPU models, as there is no common denominator (SPECInt) calculated. It also becomes difficult if there are multiple standby databases existing in the same environment. When the workloads were compared between the hosts, due to the nature of the physical standby and the primary behaving, as designed, in a completely different way, the graphs clearly show that the standby database has a considerably lower utilisation of CPU and IO resources. This is for several reasons:

- A physical Standby Database is in recovery mode therefore is not open for SQL DML or DDL in the same manner as a primary database is opened in normal mode. Therefore processes are not spawned at OS level/Database level, consuming resources such as Memory, CPU.
- A Physical standby applies “*Archivelogs*” and therefore is much more dependent on Physical Writes as these logs (*changes*) are applied on the standby from the primary database, therefore less IO load is generated.
- The reduction in IOPS/s is also attributed to DML/DDDL is not being executed on the standby database in the same manner as a primary database (e.g. rows are not being returned as part of a query result set).

It was clear after the first experiment *OLTP*, that the workloads would be profoundly different in their footprint regardless of the workload being executed, so we have not included the results of the other workloads namely, OLAP and Data Mart.

## 4.8 Experiment Three - Clustered Database (Advanced Configuration)

The final set of experiments was to execute three the workloads on a more advanced configuration, a two-node clustered database running in an Engineered system (Exadata X5-2 platform) [1], illustrated in Fig 2. During the experiment, compute nodes are closed down to simulate a fail-over. The database configurations were the same in Instance Parameters, Software Version and Patch Level. The hardware configurations were the same in OS Level, Kernel Version and memory configuration. A difference in this experiment

from the previous two is that the physical hardware and database are clustered. In this experiment we leverage the Exadata Technology in the IO substructure.

#### 4.9 Results and Analysis Experiment Three - OLTP Workload

The results for OLTP covering Memory, CPU and IOPS/s are shown graphically in Fig 7. The OLTP workload was amended to run from node 1 for the second 24 hours and this is reflected in all three of the graphs, when the instance DBM012 is very much busier than instance DBM011. The workloads are then spread evenly for the following 48 hours.

- *CPU utilisation* - for the first 24 hours, the workloads were executed fairly evenly across the cluster with a workload of 2000 users connecting consistently with peaks of 1000 users at peak times, and the CPU showed similar patterns during the workload execution.
- *CPU utilisation* - When the workload ran abnormally and all users (3000 users) ran from one node, in the second 24 hours, then the CPU utilisation did almost double in usage as expected. The increase was approximately +99% (Day 7 Hour 15)
- *IOPS/s* - The IOPS's utilisation for the first 24 hours was similar, as expected, when the workloads were evenly spread. However when the workloads were run from node 2 in the second 24 hours the IOPS increase significantly, as expected. The IOPS during the failure period was as expected, an increase of +99% (Day 7 Hour 15).
- *IOPS/S Spike* - there are two major spikes occurring at Day 7 Hour 2 and Day 8 Hour 2, these are Level 0 database backups than only run from node 1 (DBM011)
- *Memory Consumption* - The maximum memory utilisation across both instances was consistent during the first 24 hours when the workload was evenly spread. The memory configuration on DBM012 is sufficient to handle the 3000 users during the failover period, although the increase in memory used on DBM012 was only +45%

In general, the conclusion from this experiment when executing the OLTP workloads was, it cannot be assumed that when a workload fails over from one node (database instance) to another node (database instance) the footprint will be double in terms of Memory. The workload did double for CPU and IOPS/s. The results show there is an increase in IOPS/s, Memory and CPU. The difference during normal running conditions (i.e. when workloads are evenly spread) was the following: +31% (Day 7 Hour 3) CPU, +2% Memory (Day 6 Hour 21) and +1% (Day 6 Hour 12) IOPS. When the workload failed over there was a difference of +97% (Day 7 Hour 9) CPU, +99% (Day 7 Hour 20) Memory and +99% (Day 08 Hour 10) IOPS. There are two large spikes at Day 7 Hour 2 and Day 8 Hour 2; these are Level 0 RMAN backups which account for the large IOPS readings. The database instance was sufficiently sized to handle both workloads otherwise we would of expected to see out of memory errors in the database instance alert file.

#### 4.10 Results and Analysis Experiment Three - OLAP Workload

The results OLAP covering Memory, CPU and IOPS/s are shown graphically in Fig 8. The OLAP workload was amended to run from node 1 for the first 24 hours and this is clearly reflected in all three of the graphs, as the instance DBM011 is very much busier than instance DBM012 during this period. The workloads are then spread evenly for the following 48 hours.

- *CPU utilisation* - for the first 24 hours, node 1 ran the whole workload of 400 users and thus the DBM011 instance is busier compared with the workload across days two and three; as expected, utilization is effectively doubled, at +99%.
- *CPU utilisation* - when the workload ran normally (400 users) across both nodes then the utilisation was similar in its SPECint count with a difference of approximately +20%.
- *IOPS/s* - The IOPS's utilisation for the first 24 hours was busier on node 1, as expected, than node 2 given that both workloads were executed from DBM011 instance. The IOPS utilisation was almost double +99% (Day 25 Hour 05) the amount from the second period of time (Day 26 Hour 05) when the workloads were spread evenly across both instances.
- *Memory Consumption* - The maximum memory utilisation observed across both instances was consistent with the workload, the first 24 hours when the workload ran from node 1 is as expected in that there was sufficient memory to serve both workloads. However there is a difference of +55% (Day 25 Hour 04) in memory between nodes 1 and 2. For the second 24 hours, as the workloads reverted back to their normal hosts I.E. spread evenly across both nodes, their utilisation is similar with a difference of +1% (Day 26 Hour 04) between the nodes in memory utilisation.

In general, the conclusion from this experiment when executing the OLAP workloads was that it cannot be assumed that when a workload fails over from one node (database instance) to another node the footprint will be double in terms of Memory. For the metrics IOPS and CPU the increase was almost double; CPU had a difference of +99% (Day 25 Hour 04) and IOPS +99% (Day 24 Hour 04). When the workload was spread evenly across both nodes the differences between the nodes where CPU +20% (Day 26 Hour 3), Memory +2% (Day 26 Hour 3) and IOPS +1% (Day 26 Hour 4). The database instance was sufficiently sized to handle both workloads otherwise we would of expected to see out of memory errors in the database instance alert file.

#### 4.11 Results and Analysis Experiment Three - Data Mart Workload

The results are as follows for the Data Mart workloads covering Memory, CPU and IOPS/s, as shown graphically in Fig 9. The Data Mart workload was run normally for the first 24 hours, which is reflected in the workloads being similar for this period. A simulated failure of database instance DBM011 is then performed and all connections then fail-over to DBM012 on node 2 for the second 24 hours. This is

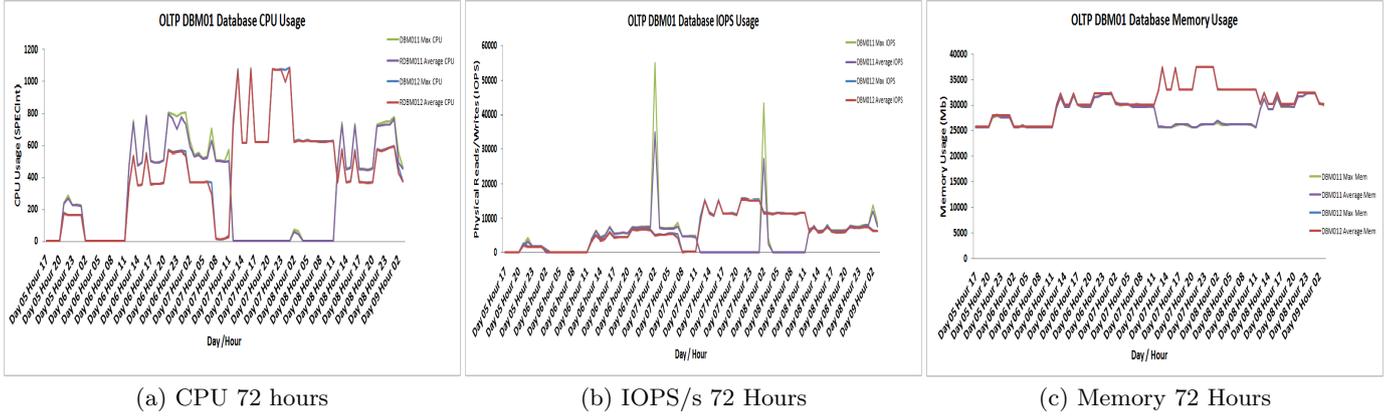


Figure 7: Results RAC OLTP: workload patterns for the 72 hour period.

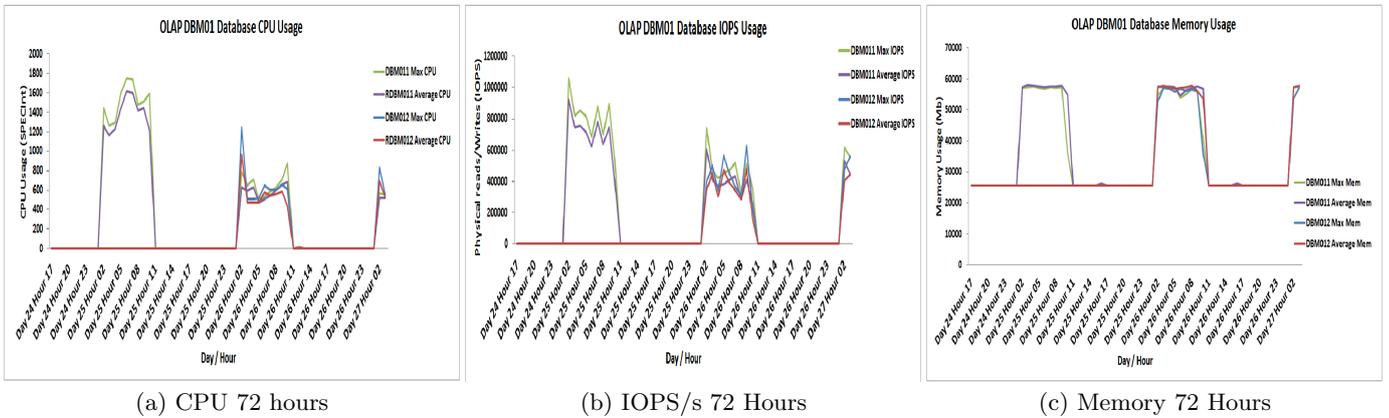


Figure 8: Results RAC OLAP: workload patterns for the 72 hour period.

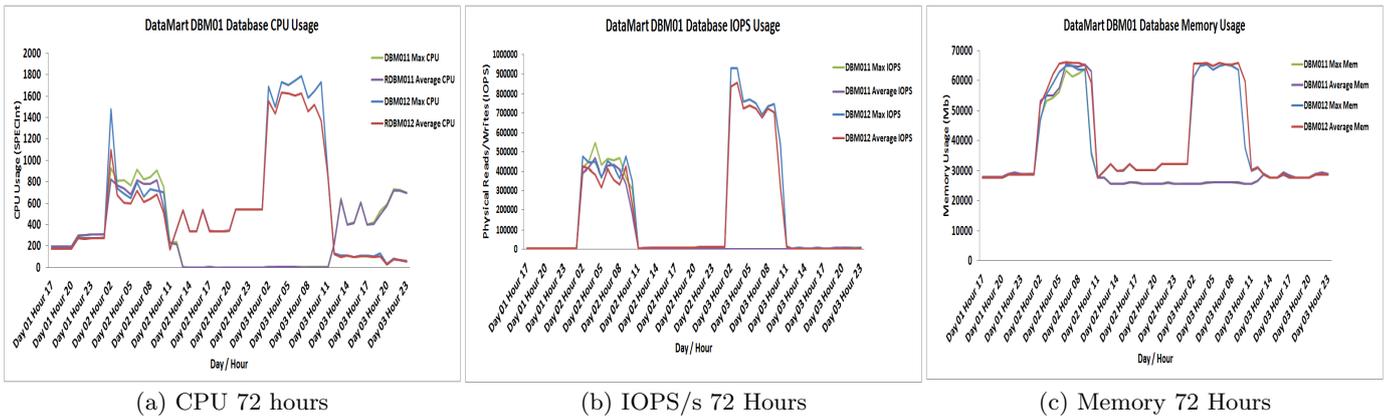


Figure 9: Results RAC Data Mart: workload patterns for the 72 hour period.

reflected in all three of the graphs as the instance DBM012 becomes much busier than instance DBM011.

- *CPU utilisation* - For the first 24 hours, the workloads were executed fairly evenly across the cluster with a workload of 2700 users connecting at different times from the two nodes and the SPECInt count was similar

with a average CPU difference of +15% (Day 2 Hour 04).

- *CPU utilisation* - When the workload ran abnormally and all users (2700 users) ran from one node, in the second 24 hours, then the CPU utilisation almost doubled in usage as expected +99% (Day 3 Hour 04).

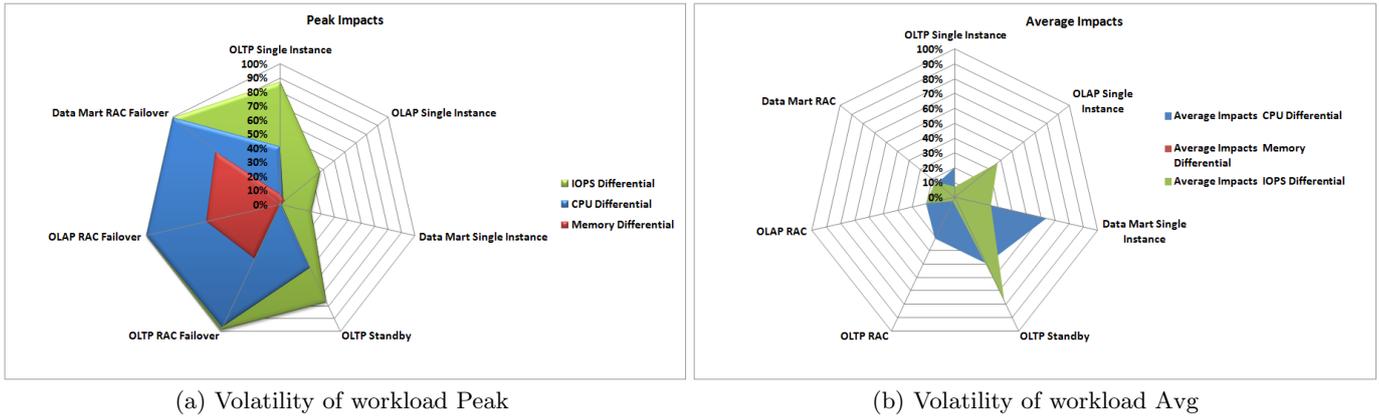


Figure 10: Workload Impacts

- *IOPS/s* - The IOPS's utilisation for the first 24 hours was similar, as expected, when the workloads were evenly spread with a difference on average of +17% (Day 2 Hour 04). However, when the workloads were run from node 2 in the second 24 hours the IOPS increased significantly, rising to almost double at +99% (Day 3 Hour 04).
- *Memory Consumption* - The maximum memory utilisation across both instances was as expected during the first 24 hours, when the workloads were evenly spread, showing a difference of +9% (Day 2 Hour 04). This behaviour was not expected during the failover period when all users execute their workload on DBM012 as the utilisation difference is +60% (day 3 Hour 04). The memory configuration on DBM012 is sufficient to handle the 2700 users.

In general, the conclusion from this experiment when executing the Data Mart workloads was, it cannot be assumed that when a workload fails over from one node (database instance) to another node the footprint will be double in terms of memory, as it only increased by approximately +60%. CPU and IOPS however, did double in its usage to approximately +99%. When the workload was spread evenly the average utilisation had a difference of CPU +15% (Day 2 Hour 04), Memory +9% (Day 2 Hour 04) and IOPS +17% (Day 2 Hour 04).

## 5. CONCLUSIONS AND FUTURE WORK

From the experiments conducted and the model we proposed, we conclude that capacity planning of databases that employ advanced configurations such as Clustering and Standby Databases is not a simple exercise. Taking the Average and Maximum readings for each metric (CPU, Memory Utilisation and IOPS) over a period of 72 hours, the outputs are volatile. One should not assume that a workload running on one database instance configured in one type of system will consume the same amount of resource as an another database instance running on another system, regardless of similarity; this is clearly shown in Fig 10 (a) (OLTP, OLAP, Data Mart RAC Failovers). These charts show us that as workloads become assimilated they completely change as the difference grows, sometimes considerably. The differences

between the footprints based on configuration can vary between +10% (CPU OLAP RAC) in normal circumstances shown in Fig 10 (b) to 99% (CPU OLAP RAC) as shown in Fig 10 (a). Fig 10(a & b, OLTP Standby) also highlights that configuration has a big impact on capacity planning databases with advanced configurations, such as standby databases.

In this paper we highlighted the problems that organisations are faced with *over-estimation* and *under-estimation* when trying to budget on non-cloud compliant financial models such as capex or cloud compliant models, which are subscription based. Accurate capacity planning can help in reducing wastage when metrics are captured and the assumption of workloads being the same is not employed. Capturing and storing the data in a central repository, like the approach we proposed, allowed us to mine the data successfully without the labour intensive analysis that often accompanies a capacity planning exercise.

The main points from this work are.

1. When capacity planning DBaaS, it should be done on a *instance-by-instance* basis and not at a database level - this is especially the case in clustered environments where workloads can move between one database and another or fail-over technology is employed.
2. Metrics need to be captured at different layers of the infrastructure in advanced configurations, for example in the storage layer, caching can mask IOPS causing the workload to behave differently.
3. Hypervisors and VMManagers can influence capacity planning as these tools allocate resource. For example, a CPU can be dissected and allocated as a vcpu (Oracle VM) [2]. How does one know that the CPU assigned is a full CPU? The Oracle Software and the database itself may assume that a full CPU was made available, when in fact it was assigned 0.9 of a CPU due to overheads.
4. CPU configuration (*Thread(s) per core*) within a VM has a profound effect when capacity planning. We observed in experiment one (OLTP and Data Mart) that small concurrent transactions in the OLTP workload executed on VM acs-163 were a lot more efficient than

the same workload executed on another VM with lower thread(s) per core, and this is reflected in Figures 3, 4 and 5.

5. SPECInt benchmark is a valid benchmark when comparing one variant of CPU with another, especially when trying to capacity plan databases with a view to a migration or upgrade of the infrastructure.
6. Standby Databases presented a different footprint. A standby database is always in a *mounted* state and therefore is configured in a recovering mode by applying logs or changes from the primary. It should not be assumed that the footprints are the same.
7. In environments that employ standby database configurations, metrics that are available for collection on the primary database are not available on the standby, namely physical reads/writes, CPU and memory, thus gathering accurate metrics is impractical. Metrics can be gathered at a host level, however if multiple standby databases are running on the same host this makes reconciliation of which database is using what more challenging.
8. In environments that employ clustered databases, if a workload running on one node fails-over from another node within the cluster, one should not assume that the properties of the composed workload will follow obviously from its constituents. Upon failover, the workload from the failing node is assimilated, with the result being the formation of a completely new footprint.

Future work is to conduct the same type of experiments between different database versions, for example a workload running on Oracle Database Version 10G/11G and Oracle Database Version 12C, analysing if the internal database algorithms have any influence and by how much. However techniques already exist that go some way to answering this question through the use of a product called Database replay [7]. Being able to gather metrics from a standby database instance for CPU, IOPS and Memory is critical for our model as this would allow us to accurately analyse the CPU such as SPECInt, Memory and IOPS's. We could configure a custom metric to execute internal queries against the standby database, and this is now in the design phase, but until then capacity planning architectures with standby database will need to rely on host metrics.

## 6. REFERENCES

- [1] U. Author. Oracle exadata database machine. *Security Overview*, 2011.
- [2] O. Corporation. Oracle vm concept guide for release 3.3.
- [3] T. P. P. Council. Tpc-ds benchmark.
- [4] T. P. P. Council. Tpc benchmark h standard specification version 2.1. 0, 2003.
- [5] T. P. P. Council. Tpc benchmark c (standard specification, revision 5.11), 2010. URL: <http://www.tpc.org/tpcc>, 2010.
- [6] K. M. Dixit. Overview of the spec benchmarks., 1993.
- [7] L. Galanis, S. Buranawatanachoke, R. Colle, B. Dageville, K. Dias, J. Klein, S. Papadomanolakis, L. L. Tan, V. Venkataramani, Y. Wang, and G. Wood. Oracle database replay. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1159–1170, New York, NY, USA, 2008. ACM.
- [8] D. Giles. Swingbench 2.2 reference and user guide.
- [9] M. Guidolin, S. Hyde, D. McMillan, and S. Ono. Non-linear predictability in stock and bond returns: When and where is it exploitable? *International journal of forecasting*, 25(2):373–399, 2009.
- [10] H. Hacigümüş, J. Tatemura, Y. Chi, W. Hsiung, H. Jafarpour, H. Moon, and O. Po. Clouddb: A data store for all sizes in the cloud. *Internet: <http://www.neclabs.com/dm/CloudDBweb.Pdf>*, [January 25, 2012], 2012.
- [11] Y. Kouki and T. Ledoux. Sla-driven capacity planning for cloud applications. In *Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on*, pages 135–140. IEEE, 2012.
- [12] C. Lobo. Cloud resource usage—Tailed distributions invalidating traditional capacity planning models. *Journal of grid computing*, 10(1):85–108, 2012.
- [13] S. Mahambre, P. Kulkarni, U. Bellur, G. Chafle, and D. Deshpande. Workload characterization for capacity planning and performance management in iaas cloud. In *Cloud Computing in Emerging Markets (CCEM), 2012 IEEE International Conference on*, pages 1–7. IEEE, 2012.
- [14] R. Moussa and H. Badir. Data warehouse systems in the cloud: Rise to the benchmarking challenge. *IJ Comput. Appl.*, 20(4):245–254, 2013.
- [15] B. Mozafari, C. Curino, A. Jindal, and S. Madden. Performance and resource modeling in highly-concurrent oltp workloads. In *Proceedings of the 2013 acm sigmod international conference on management of data*, pages 301–312. ACM, 2013.
- [16] B. Mozafari, C. Curino, and S. Madden. Dbseer: Resource and performance prediction for building a next generation database cloud. In *CIDR*, 2013.
- [17] J. Murphy. Performance engineering for cloud computing. In *European Performance Engineering Workshop*, pages 1–9. Springer, 2011.
- [18] A. Ray. Oracle data guard: Ensuring disaster recovery for the enterprise. *An Oracle white paper*, 2002.
- [19] S. Sahri, R. Moussa, D. D. Long, and S. Benbernou. Dbaas-expert: A recommender for the selection of the right cloud database. In *International Symposium on Methodologies for Intelligent Systems*, pages 315–324. Springer, 2014.
- [20] S. Shang, Y. Wu, J. Jiang, and W. Zheng. An intelligent capacity planning model for cloud market. *Journal of Internet Services and Information Security*, 1(1):37–45, 2011.
- [21] T. Yu, J. Qiu, B. Reinwald, L. Zhi, Q. Wang, and N. Wang. Intelligent database placement in cloud environment. In *Web Services (ICWS), 2012 IEEE 19th International Conference on*, pages 544–551. IEEE, 2012.