Cloud Consolidation with Oracle (RAC) – How much is too much?

Markus.Michalewicz@oracle.com
Senior Principal Product Manager
Oracle RAC, Oracle America
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.

Agenda

• Database Cloud Architectures
  • General Considerations
  • Database Consolidation
    – Memory Management
    – CPU(_COUNT) Management
      • A word on Instance Caging
    – Database Resource Management
    – Summary Database Consolidation
  
• Oracle RAC-specific Considerations for Consolidation
  – Per Server Limits
  – Real Time (RT) Processes
  – Cluster Limits
  
• Summary and Q&A
Database Cloud Architectures

Common building blocks are shared server and storage pools

General Considerations
**Oracle (RAC) Database Consolidation**

**Registered vs. Running**

- Registered databases and instances could potentially start and run at the same time.

- Oracle’s Quality of Service Management or scripts can be used to model policies to run certain databases only at certain times; e.g. geographic region over time

- Assume the cluster is PST based:
  - EMEA based DBs run 10:00pm – 8am PST
  - APAC based DBs run 6:00pm – 3am PST
  - USA based DBs run 8:00am – 6pm PST

**Registered:**
- n DB instances are defined to run on a machine (potentially)

**Running:**
- Registered databases and instances are (concurrently) running (active workload)

---

**Database Consolidation**

**General Considerations**

- Simplest of all consolidation cases:
  - One database Instance per Server

- When consolidating more than one database on a server, consider the server capacity with any DB added.

- More details:
General Considerations
Consider Workload Characteristics during Capacity Planning

Existing Workload

New Workloads

OR

Peak
Average

The smaller this gap, the better.

Utilization
Time

Workload A

Workload B

Utilization
Time

Poor match:
Gap increases
(antagonistic)

Resulting Workload

Utilization
Time

Workload B

+
General Considerations
Consider Workload Characteristics during Capacity Planning

Existing Workload

Peak
Average
Utilization
Time

Resulting Workload

Time

Peak
Average
Utilization

The smaller this gap, the better.

Good match: Gap decreases (complimentary)

Existing Workload + Workload A

Utilization
Time

General Considerations
Consider Workload Characteristics during Downtime

Utilization Normal Operation

Peak
Average
Time

DB_A
DB_B
OS
OS
OS
OS
HR
DW
ERP
CRM
General Considerations

Consider Workload Characteristics during Downtime

Utilization Normal Operation

- During Normal Operation, the utilization is at its average level with slight fluctuations.

Utilization During Downtime

- During Downtime, the utilization is lower than during normal operation, indicating a reduction in workload.

The Benefits of Standardization

Easier deployment and better predictability

- Standardization of software and hardware simplifies planning.
  - Standardized hardware means a predictable behavior should demand increase and additional hardware needs to be added (horizontal scaling approach).

- Using "application profiling" (template based deployment) based on current system(s) and performance baselines allows for a predictable deployment of new applications on the same system using existing profiles.
Database Consolidation

Starting block: One Database instance per server

- Components to consider:
  - Memory
  - CPU
  - I/O
  - Processes
  - Network
Database Consolidation

One instance per server as the basis

- An Oracle database by default assumes that there is only one database instance running on the server:
  - Instance parameters are based on this assumption
  - Consolidation changes that premise

- Main resources used:
  - Memory
  - CPU
  - I/O

- Resources "regulated by default":
  - Memory
    - SGA / PGA Targets
  - CPU

Database Consolidation

Recommendation 1: Manage Memory carefully (and dynamically)

- Avoid memory starvation and swapping as it has negative impact on the system.
- Do not oversubscribe memory resources

- Define memory settings carefully – rule of thumb:
  - For the OS in general:
    - Shared Memory identifiers and segments
    - Use Hugepages, if possible – details:
      - MOS notes 361323.1 and 401749.1
  - For OLTP applications:
    - \[\text{SUM (sga\_target + pga\_aggregated\_target)}\] <= 80% of physically available memory per DB server
  - For DW / BI applications:
    - \[\text{SUM (sga\_target + 3\times pga\_aggregated\_target)}\] <= 80% of physically available memory per DB server
Database Consolidation

Recommendation 2: Use CPU_COUNT to “cage instances”

- CPU usage should be regulated.
  - The OS scheduler schedules CPU as requested by each individual instance.
  - The OS scheduler does not know about the priority of the various instances on the server.

- Use CPU_COUNT or ideally Instance Caging
  - Instance Caging is configured in just 2 steps:
    1. Set “cpu_count” parameter
       - Max. number of CPUs the instance can use at any time
    2. Set “resource_manager_plan” parameter
       - Enables CPU Resource Manager
       - E.g. out-of-box plan “DEFAULT_PLAN”

Database Consolidation

Using CPU_COUNT as a “central knob”

- CPU_COUNT regulates CPU usage and dependent resources (to a certain degree):
  - Parallelism (PQ operations)
  - Processes
  - Load Calculation

- Processes and PQ operations should be considered explicitly.
A Word on Instance Caging

Database Consolidation
Instance Caging: Partitioning Approach

- Provides maximum isolation
- For performance-critical databases
- If one database instance is idle, its CPU allocation is unused
- The rule of thumb for the partitioning approach is to set a general limit:
  - \(\text{SUM (CPU\_COUNT)} < 75\% \times \text{Total CPUs}\)
Database Consolidation
Instance Caging: Over-Subscription Approach

- Best used for non-critical databases that are typically well-behaved
- Contention for CPU if database instances are sufficiently loaded
- Typically not enough contention to destabilize OS or DB instances
- Best approach if the goal is fully utilized CPUs

Database Consolidation
Instance Caging – under the covers

- If cpu_count is set to 4 on a 16 CPU server
  - All foreground processes make progress
  - But only 4 foregrounds are running at any time
- Most backgrounds are not managed
  - Critical and use very little CPU
  - MMON, Job Scheduler slaves are managed
- No CPU affinity
  - Not meant for hard-partitioning or licensing
  - All CPUs may be used
  - CPU utilization averaged across all CPUs ≤ 25%
- More information:
Database Consolidation

Over-Subscription Approach – It’s still hardware that’s the limit

• Best used for non-critical databases that are typically well-behaved – examples:
  • Complimentary workload
  • Systems with little contention for CPU if database instances are sufficiently loaded.

• Do not use,
  • If the load is significant and of longer duration, as system stability can get impacted.
  • For highly critical systems

How much over-subscription is OK?

• As CPU_COUNT does not consider the “quality of a CPU” an absolute maximum is hard to determine / depends on the system.

• The general rule of thumb is:
  • SUM (CPU_COUNT) <= up to 2x Total CPUs

• Consider different system types:

<table>
<thead>
<tr>
<th></th>
<th>Threaded</th>
<th>Core based</th>
<th>Engineered</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Total CPUs&quot;</td>
<td># of threads</td>
<td># of cores</td>
<td># of threads</td>
</tr>
<tr>
<td>are based on</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Max. over-provisioning factor</td>
<td>1.5 (HT*)</td>
<td>1.0 (hHT**)</td>
<td>2.0</td>
</tr>
<tr>
<td>(HT*): ratio 1:2; (hHT**): ratio 1:n, with n &gt;2</td>
<td>3.0 (DBM)</td>
<td>2.0 (ODA)</td>
<td></td>
</tr>
</tbody>
</table>
Database Resource Management

Database Consolidation

**Recommendation 3: Use DB Resource Manager**

3 steps to use Resource Manager:

1. Group sessions with similar performance objectives into **Consumer Groups**
2. Allocate resources to consumer groups using **Resource Plans**
3. Enable Resource Plan
1. Define memory settings carefully:
   - For OLTP applications:
     - \( \text{SUM (sga\_target + pga\_aggregated\_target)} \leq 80\% \text{ of physically available memory per DB server} \)
   - For DW / BI applications:
     - \( \text{SUM (sga\_target + 3* pga\_aggregated\_target)} \leq 80\% \text{ of physically available memory per DB server} \)

2. Use CPU\_COUNT or ideally Instance Caging
   - The general rule of thumb is:
     - \( \text{SUM (CPU\_COUNT)} \leq \text{up to 2 x Total CPUs} \)
   - Consider different system types.

3. These are the most crucial per server limits.
Oracle RAC-specific Considerations for Consolidation

Oracle RAC based Consolidation

Server limits are mostly reached before cluster limits apply

- Most customers will experience “per server limits” before “cluster limits” apply.
  - Oracle RAC introduces a few more processes (potential limits) to consider.

- Oracle RAC DBs use LMS Real Time (RT) processes per instance.
  - LMS RT processes need to be considered in particular
Real Time (RT) Processes

Oracle RAC based Consolidation

Considerations for Real Time (RT) Processes in general

- A Real Time process can only run on one CPU (core) at a time.
- The usage of the CPU is typically short.

- The general rule of thumb is:
  - The aggregated number of RT processes per server should not exceed the number of cores per server
  - One Oracle RAC instance has typically at least one RT process (LMS) per default
  - An Oracle ASM instance has one RT process
  - Oracle Clusterware uses various RT processes
Oracle RAC based Consolidation

Background for LMS Real Time (RT) Process recommendation

- The number of LMS RT processes per instance is determined by a function on CPU_COUNT.
- In order to guarantee optimized performance and reliability, the general rule of thumb for RAC is:
  - The aggregated number of LMS RT processes per server should not exceed [cores per server]-1
    - See MOS note: 558185.1 for details
  - This leaves one core free for additional RT processes to be assigned as needed, as LMS RT can stay on a core for a moment

Oracle RAC based Consolidation

Automatic adjustment of LMS process priority in 11.2.0.3

- With 11.2.0.3 the number of LMS RT processes are monitored and adjusted according to the number of cores on the node periodically.
  - The goal is to keep RT LMSs per server <= # cores per server
    - For details, see MOS note 1392248.1 – Auto-Adjustment of LMS Process Priority in Oracle RAC with 11.2.0.3 and later
  - This excludes any ASM instance running on the system as well as any pre-11.2.0.3 database instance
    - See also MOS note 1439551.1 – Oracle (RAC) Database Consolidation Guidelines for Environments using mixed Database Versions
**Oracle RAC based Consolidation**

**Recommendation 4: Over-provision only based on CPU_COUNT**

- Do not over-provision the number of LMS RT processes on one server.

- Limit the number of RT LMSs by:
  - Using CPU_COUNT
    - Directly reducing the number of LMS RT processes (gcs_server_processes)
    - Downgrading additional LMS RT processes to time share (TS).
  - In 11.2.0.3 and later, the RT to CPU rule will be enforced by automatically downgrading subsequently started LMS RT processes to TS.

---

**Cluster Limits**
Oracle RAC based Consolidation

Cluster Limits – Starting DBs are currently the main concern

• In most cases, per server limits will be reached before cluster limits are reached.

• Cluster limits apply to:
  • Registered resources (databases)
  • Starting databases in the cluster – reason:
    • Starting databases need to register with the cluster (Oracle Clusterware).

• Currently and for example, 100 starting Oracle RAC databases on a 4 node cluster are supported.

• The Number assumes each Oracle RAC database uses an instance on each node.

Summary
(Oracle RAC) Database Consolidation

Summary

- For a successful database consolidation, consider the following as rules of thumb:
  1. General considerations for capacity planning
  2. Manage Memory carefully (and dynamically)
  3. Use CPU_COUNT
  4. Use DB Resource Manager
  5. Over-subscribe only based on CPU_COUNT