



How MoreVRP helps you utilize your hardware better

By using Oracle RAC

White Paper

1133 Broadway, Suite 706, New York, NY 10010
info@more-resource.com www.more-resource.com
Tel: +1-212-502-3744

Oracle RAC advantages

Oracle Real Application Cluster (RAC) is a solution that allows your applications to maintain high availability, good SLA, easy and cost effective scalability.

RAC allows you to scale at the CPU level only, by adding more servers, but the shared storage remains a single point of failure in terms of both availability and **performance**.

Managing the Load in RAC

In many organizations around the globe, customers use Oracle RAC to split the load of different types of users or activities among different servers, and by doing so – they try to maintain stable QoS for important transactions.

This can be done either manually (by directing explicitly where each application is to connect to a specific Node in the RAC) or by using the Oracle RAC services features that do this automatically.

Is it really good enough for QoS ?

This strategy of Load balancing the application allows you to separate the CPU load only between the various servers. Today most servers come with at least 4 Cores – and a CPU bottleneck is a rare occurrence of performance problem in RAC.

The IO bottleneck is the most common one

In most of the databases, an IO bottleneck is the first bottleneck the system encounters. As described previously - In RAC, the storage remains the point of contention for all applications. 2 transactions that come from 2 different nodes in the RAC, **compete for the same IO resources since they use the same Physical storage layer.**

The common solution

Because of the lack of capability to manage the resource consumption of IO, most IT managers decide not to re-enforce the RAC with extra servers, but build 2 independent environments.

So when you buy 4 servers for your database layer, you eventually decide to build 3 servers in a RAC and another single stand alone server for the intensive IO activity, while synchronizing the 2 environments (Using Oracle DataGuard or any other type of synchronization). The application load is split either manually or automatically among the 2 environments.

For example: In a typical database with 4 available servers, Instead of running the OLTP and DW (reporting) activities on a single 4 server RAC, most DBAs decided to take one node and dedicate it to the DW activities only. So at the end you have 3 servers in a RAC and one dedicated server for DW and reporting.

The benefit of having mixed DW/OLTP RAC

Using a single RAC environment for both OLTP and reporting instead of having a dedicated reporting server has numerous advantages:

- **Simple Architecture**

An immediate ROI since your architecture remains simple:

- No need for synchronization jobs and additional maintenance
- No need for twice the storage for the same database

- **Real time data analysis**

Users will be able to analyze real-time data which is up to date. The need for “Real-Time BI” is a growing need in almost any organization.

- **Better utilization of existing hardware**

OLTP and DW have different characteristics of transactions types. If you maintain 2 independent environments (DW & OLTP), the load is not equally spread among them. Sometimes only one environment is CPU loaded while the other environment is loaded with IO activity. Sometimes the DW is completely idle since no reports are being executed while your OLTP is completely loaded with online traffic.

Having a single RAC serving both OLTP and DW, allows you to fully utilize your existing hardware so all your hardware resources are available to both DW and OLTP activities.

- **Better scalability**

Adding a server to a single RAC environment contributes both to the DW and OLTP simultaneously in terms of scalability.

- **Better Availability**

Usually, with a given amount of servers, you compromise with the availability of your DW environment. Since high availability is most important for your OLTP system, you tend to use Oracle RAC only there, leaving the DW to work with a single server only.

Having a single RAC for both OLTP & DW allows you to deliver the same high availability SLA to both activities.

Don't let the Load control you. You control the Load !

The benefits of having a single RAC environment for both DW and OLTP are clear. However, as mentioned before, the IO layer (Storage) is the place where those 2 activities can clash with one another.

***MoreVRP* allows you to keep your architecture simple and easy to manage.**

Using ***MoreVRP***, you can manage a single RAC environment for both activities while managing the Load based on your IT & business needs.

With ***MoreVRP*** you can define the exact amount of CPU and IO bandwidth allocated to each transaction in Real-Time and maintain a good Quality of Service (QoS) to your OLTP transaction, while still allowing the DW reports to run on real-time data.

If you already have a separated DW environment, it is not too late.

Just change your reporting tools and users back to your OLTP database, and control the Load using ***MoreVRP***. You can then use the old DW

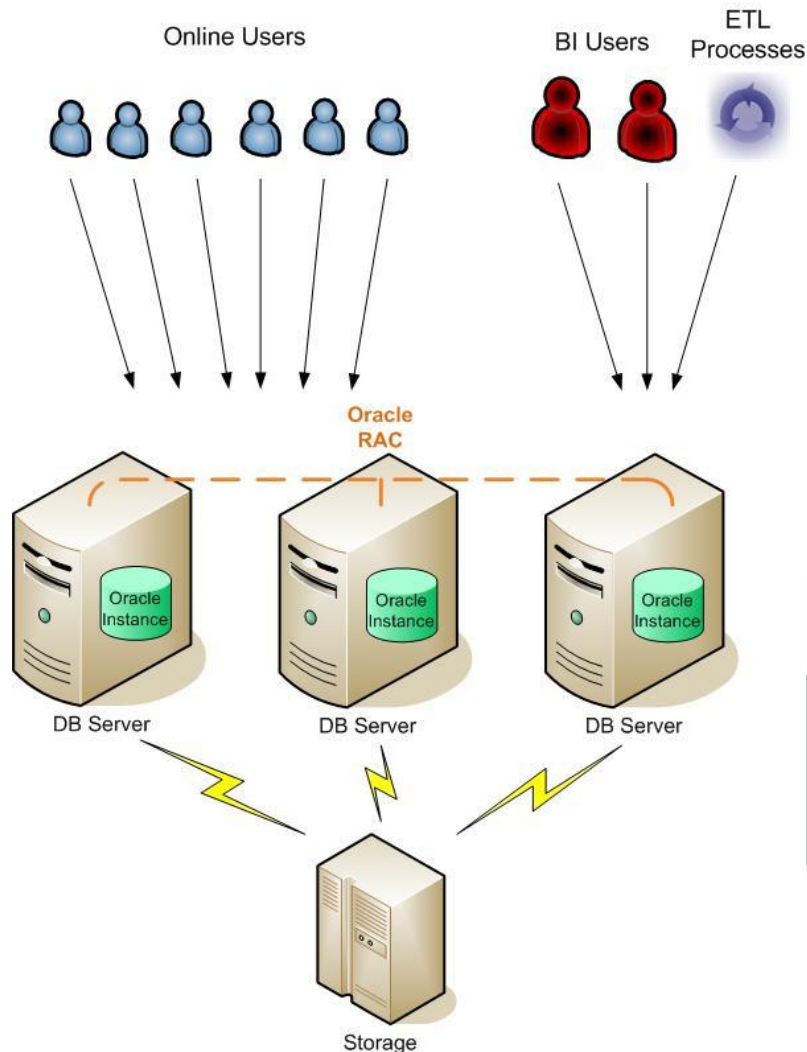
servers for other applications, or just use it to create a RAC environment for your OLTP system.

An Illustration of the problem

Here is a simple illustration that almost every customer goes through while planning the architecture.

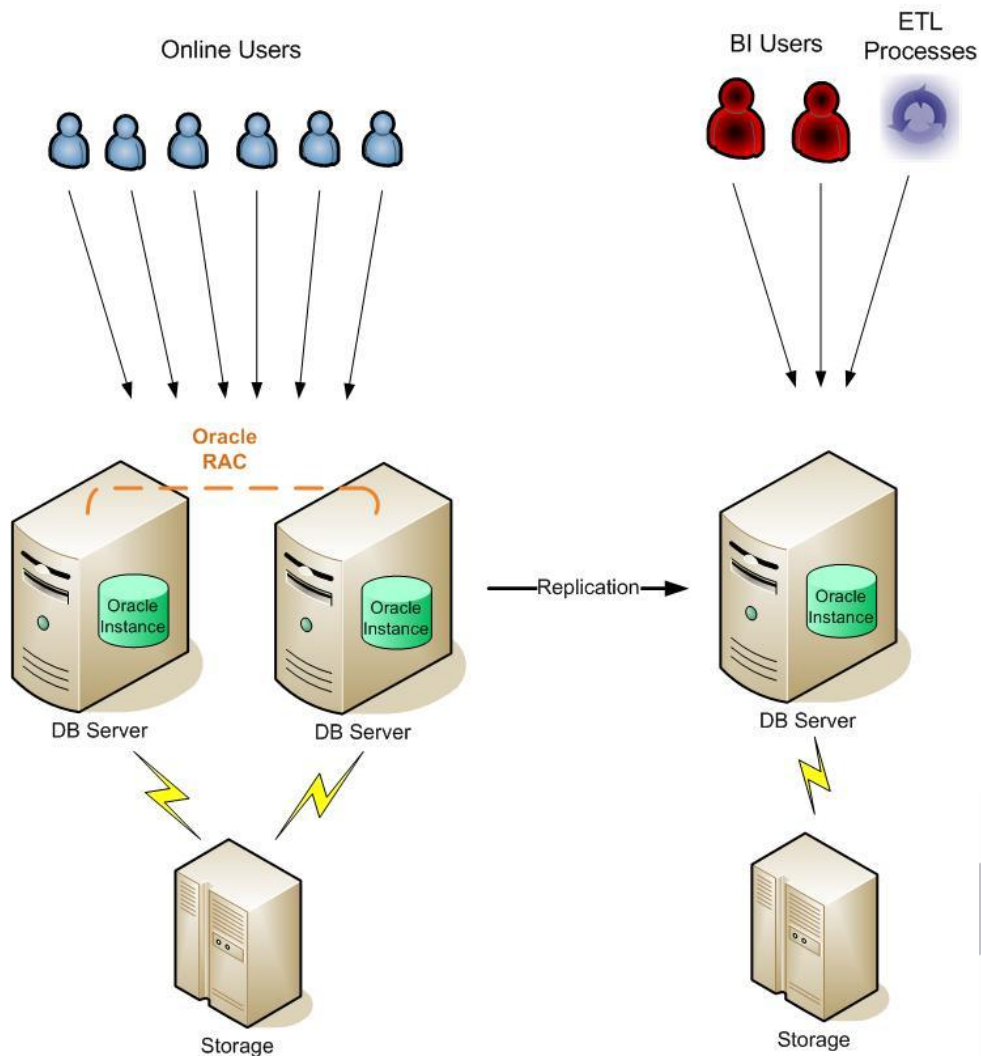
1. Desired Architectures

The DBA would like to have an easy way to manage the environment. Therefore he wants to have a big RAC while splitting the various activities on the different nodes of the RAC manually or using RAC Services feature:



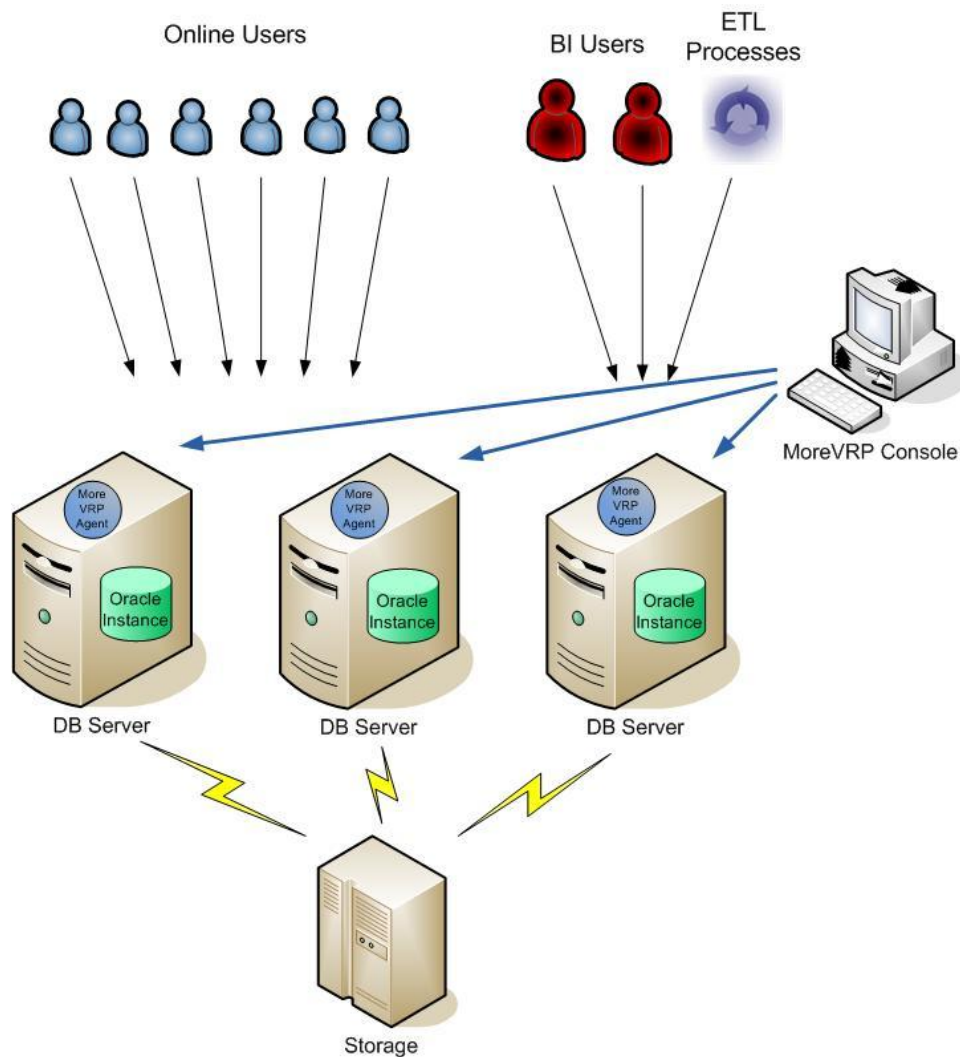
2. Split Architecture

Due to the problem of IO bottleneck, the DBA decides not to use all the servers in a single RAC environment but rather use one server as a dedicated server for the IO consuming tasks such as reports, analysis etc...



3. Architecture With MoreVRP

Using **MoreVRP**, the customer can use all its available servers in a single RAC environment while managing the IO & CPU load using **Morevrp** :



The architecture delivers 2-3 times better performance for all applications, and eliminate the mutual influence of one application over the other, which dramatically delivers better Qos