UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ – UTFPR
CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ALEXANDRE GODOY KOUPAKA

ALTA DISPONIBILIDADE EM BANCO DE DADOS, UTILIZANDO AS FERAMENTAS REAL APPLICATION CLUSTERS (RAC) E DATA GUARD DA TECNOLOGIA ORACLE 10 g

TRABALHO DE DIPLOMAÇÃO

MEDIANEIRA
2014

ALEXANDRE GODOY KOUPAKA

ALTA DISPONIBILIDADE EM BANCO DE DADOS, UTILIZANDO AS FERAMENTAS REAL APPLICATION CLUSTERS (RAC) E DATA GUARD DA TECNOLOGIA ORACLE 10 g

Trabalho de Diplomação apresentado à disciplina de Trabalho de Diplomação, do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas – COADS – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Claudio Leones Bazzi

MEDIANEIRA

2014

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria de Graduação e Educação Profissional
Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas

TERMO DE APROVAÇÃO

Alta disponibilidade em banco de dados, utilizando as ferramentas Real Application Clusters (Rac) e Data Guard da tecnologia Oracle 10 G

Por

Alexandre Godoy Koupaka

Este Trabalho de Diplomação (TD) foi apresentado às 16:40 hs, do dia 02 de fevereiro de 2014, como requisito parcial para a obtenção do título de Tecnólogo no Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, da Universidade Tecnológica Federal do Paraná, *Campus* Medianeira. O acadêmico foi argüido pela Banca Examinadora, composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado com louvor e mérito.

Prof. Dr Claudio Leones Bazzi
UTFPR – *Campus* Medianeira
(Orientador)

Prof. Valter Ekert
UTFPR – *Campus* Medianeira
(Convidado)

Prof. Marcio Matté
UTFPR – *Campus* Medianeira
(Convidado)

Prof. Me. Juliano Rodrigo Lamb
UTFPR – *Campus* Medianeira
(Responsável pelas atividades de TCC)

RESUMO

KOUPAKA, Alexandre Godoy. Alta disponibilidade em banco de dados utilizando as ferramentas *Real Application Cluster* (RAC) e *Data Guard* da tecnologia *Oracle 10 g* – Trabalho de Diplomação (curso superior de Tecnologia em Análise e Desenvolvimento de Sistemas). Universidade Tecnológica Federal do Paraná – UTFPR. Medianeira. 2014

As organizações estão cada vez mais dependentes do funcionamento adequado de seus bancos de dados e cada mau funcionamento tem impacto direto em seus resultados. Baseando-se nisso, a *Oracle* criou um conjunto de soluções para prover alta disponibilidade. Este trabalho tem como finalidade, realizar um estudo sobre alta disponibilidade em banco de dados, utilizando duas ferramentas da tecnologia *Oracle: Oracle Real Application Cluster* (RAC) e *Data Guard,* descrevendo a arquitetura desses produtos, usando também máquinas virtuais e detalhando a implementação de acordo, com as devidas orientações do fabricante, tendo como objetivo que as informações contidas em bancos de dados estejam disponíveis, ainda que aconteça algum tipo de indisponibilidade de seus serviços, devido à destruição inesperada do local físico onde se encontra a base de dados primária.

Palavras-Chave: Oracle, Alta-Disponibilidade, Cluster, RAC.

ABSTRACT

KOUPAKA, Alexandre Godoy. High Availability in the Database using the tools Real Application Cluster (Rac) and Oracles's Data Guard Technology 10g – Work Graduation (Technology Analysis and Systems Development), Federal Technological University of Paraná. Medianeira, 2014.

Organizations are increasingly dependent on the proper functioning of your databases and each wrong operation has a direct impact on your results. Based on this, Oracle has developed a set of solutions to provide high availability. This work, aims to conduct a study of high availability in database using two tools of Oracle technology: Oracle Real Application Cluster (RAC) and Data Guard, describing the architecture of these products and, use virtual machines, detailing the implementation according to the manufacturer's guidelines, in order that the information contained in databases are available even if an outage of its services, by the unexpected destruction, the physical location of the primary database.

Keywords: Oracle, High-Availability, Clustering, RAC

LISTA DE FIGURAS

LISTAS DE SIGLAS

ASM	*Automatic Storage Management*
CPU	*Central Processing Unit*
CRS	*Cluster Ready Service*
CSS	*Cluster Synchronization Services*
DBA	*Data Base Administrator*
DDL	*Data Definition Language*
DML	*Data Manipulation Language*
EVM	*Event Manager*
IP	*Internet Protocol*
LAN	*Redo de Area Local*
MRP	*Managed Recovery Process*
OCFS	*Oracle Cluster File System*
OCR	*Oracle Clusterware Registry*
PGA	*Program Global Area*
RAC	*Real Cluster Application.*
RMAN	*Recovery Manager*
SGA	*System Global Area*
SGBD	Sistema Gerenciador de Banco de Dados
SQL	*Structured Query Language*
TI	Tecnologia da Informação.
VIP	*Virtual Ip Address*

SUMÁRIO

1 INTRODUÇÃO

Com a grande modernização na área de informática, a informação passou a ser item fundamental dentro de qualquer empresa, sendo também muito importante a segurança de dados e informações, contidas nos bancos de dados das organizações. Em busca de prevenir perda de dados, as empresas investem em qualidade e segurança em suas infraestruturas de TI (Tecnologia de informação), pois, já que há uma grande preocupação com ocorrência de falhas ou erros, prejudicando assim o acesso aos serviços e as empresas que dependem de que seus bancos de dados estejam disponíveis para darem continuidade nos seus negócios. Por isso que é importante e necessário, desenvolver métodos para armazenamento de informações, evitando que em algum momento fiquem inacessíveis.

Segundo contempla Ramos (2006), uma informação disponível é aquela que pode ser acessada por aqueles que dela necessitam, no momento que precisam. Além da disponibilidade da informação, ela precisa ser íntegra e somente ser acessível a quem é de direito.

Para garantir que as informações estejam sempre acessíveis, é necessário gerenciar os incidentes, de forma que o serviço mantenha-se sempre contínuo, pois a cada interrupção podem ocorrer impactos diretos no desempenho geral da empresa.

Com o intuito de diminuir esses impactos, a tecnologia *Oracle* 10g criou soluções para aumentar-se a disponibilidade de dados. Dentre elas, são oferecidas pela *Oracle* 10g o *Real Application Clusters (RAC)* e o *Data Guard* (ferramentas que serão estudadas nesse trabalho), com o objetivo de minimizar a inatividade dos serviços, e que em caso de ocorrência de uma interrupção de seus serviços, pela destruição, do local físico onde se encontra a base de dados primária (natural ou causada pelo homem) que faça com que inesperadamente os bancos de dados fiquem inacessíveis.

Com a utilização dessas ferramentas da *Oracle*, se terá uma cópia de dados altamente disponível em um ambiente de Banco de dados. Assim, as empresas terão vantagens, pois em caso de falhas, acontecimento de algum desastres ou inatividade inesperada, os sistemas dos bancos de dados permanecerão ativos para seus usuários, sem qualquer indisponibilidade, já que o objetivo é a replicação de dados transacionais e no caso de falha de bases, fazer um direcionamento de servidores, garantindo que esses dados estejam sempre acessíveis e disponíveis. (LONEY; BRYLA, 2005).

1.1 OBJETIVO GERAL

Propor um banco de dados, em alta disponibilidade, usando o *Real Application Clusters* e *Data Guard* (ferramentas da tecnologia *Oracle 10g,)* que segundo a *Oracle*, ao utilizá-las as empresas garantirão que as informações contidas nesses bancos de dados, estejam sempre disponíveis e acessíveis, mesmo com a ocorrência de alguma indisponibilidade, causando inatividade dos serviços pela destruição inesperada, do local físico onde se encontra a base de dados, fazendo com que esses dados fiquem inacessíveis. Essa indisponibilidade pode ser causada naturalmente, por acontecimentos ligados a natureza, ou não (como ocorreu durante *World Trade Center* que devido a um atentado terrorista, as duas torres nas quais se localizavam muitas empresas, foram destruídas, destruindo também as máquinas de servidores dessas corporações, as quais continham informações, que não puderam ser recuperadas).

1.1.1 OBJETIVOS ESPECÍFICOS

- Apresentar as ferramentas da *Oracle 10g (Real Application Clusters* e *Data Guard)*, mostrando uma topologia, que garanta a alta disponibilidade das informações contidas em banco de dados, mesmo em caso de comprometimento da estrutura física do local devido ao acontecimento de uma inatividade inesperada no acesso as informações contidas no banco de dados.
- Buscar comprovar a eficiência dos produtos RAC (*Real Application Clusters*) e *Data Guard* da *Oracle* 10g, na manutenção do funcionamento dos bancos de dados, para que ainda com a ocorrência de uma destruição inesperada, do local físico onde se encontra a base de dados primária, mantenha-se a integridade dos dados e a continuidade dos serviços.

1.2 JUSTIFICATIVA

No atual cenário da tecnologia da informação, tem-se a grande necessidade de manipulação dos dados. Com o aumento da demanda no acesso e armazenamento de dados e informações, é notório o crescimento das bases de dados, bem como se faz presente, a preocupação em manter os dados disponíveis em qualquer cenário. As empresas necessitam ter a segurança de que ainda que aconteça indisponibilidade, (ou seja, algo que possa suspender de maneira inesperada, haja continuidade de acesso as informações transmitidas pelos servidores e que as informações contidas em seus bancos de dados sejam preservadas, permanecendo integras e acessíveis.

Em 11 de setembro de 2001, como o atentado terrorista ao *World Trade Center*, Nova York, Estados Unidos, inúmeras empresas tiveram suas informações perdidas definitivamente, devido à destruição dos servidores, que estavam contidas somente naquele espaço físico. A alta disponibilidade garantiria a existências dessas informações em outro espaço geográfico.

2 REVISÃO BIBLIOGRÁFICA

2.1 BANCO DE DADOS *ORACLE* 10g (*ORACLE DATA BASE* 10g)

Segundo Ramalho (2005 p. 1), "um banco de dados *Oracle* tem uma estrutura física e lógica. Como essas estruturas no servidor são separadas, o armazenamento físico dos dados pode ser gerenciado, sem afetar o acesso às estruturas lógicas de armazenamento.".

> "Um banco de dados é uma coleção de dados em disco, em um ou mais arquivos em um servidor de banco de dados, que coleta e mantém informações relacionadas. O banco de dados consiste em várias estruturas físicas e lógicas, sendo a tabela a estrutura lógica mais importante no banco de dados. Uma tabela consiste em linhas e colunas, que contem dados relacionados. No mínimo, um banco de dados deve ter pelo menos tabelas para armazenar informações úteis" (LONEY; BRYLA, 2005 p.4).

Os arquivos do sistema operacional, que constituem os bancos de dados são quem determinam a estrutura física desses bancos, enquanto um único ou mais *tablespaces* (que são unidades lógicas de armazenamento, e cada banco de dados pode ter uma ou mais tabelas), compõe a estrutura lógica desses bancos de dados (LONEY; BRYLA, 2005).

> "Cada banco de dados *Oracle* em execução, é associado a uma instância *Oracle*. Quando um banco de dados é iniciado em um servidor de banco de dados, o *software Oracle* aloca uma área de memória compartilhada, chamada SGA (*System Global Area*) e inicia vários processos de *background Oracle*. Essa combinação de SGA e processos *Oracle* é chamada de instância *Oracle*" (BEST; BIILINGS, 2006 p.32).

A *Structured Query Language* (SQL) é uma linguagem de programação utilizada para se ter acesso a um banco de dados *Oracle*. Essa linguagem caracteriza o banco de dados. Cada banco de dados contém uma ou mais tabelas, que são definidas por linhas e colunas (LONEY; BRYLA, 2005).

> "Um esquema é uma coleção de objetos, que, por sua vez, são as estruturas lógicas que se referem diretamente aos dados do banco de dados. Os objetos do esquema incluem estruturas como tabelas, visões, sequências, procedimentos armazenados, sinônimos, índices, agrupamentos e links de banco de dados. As estruturas de armazenamento lógico, incluindo os *tablespaces*, os segmentos e as extensões, dizem como é usado o espaço físico de um banco de dados. Os objetos de esquema e os relacionamentos entre eles formam o projeto relacional de um banco de dados" (RAMALHO, 2005 p.2).

2.2 ALTA DISPONIBILIDADE EM BANCO DE DADOS

Com a alta disponibilidade as empresas se asseguram que os dados e informações contidas em seus bancos não sejam perdidos, e ainda, que tenham acesso disponível sempre e em todo lugar. A utilização da alta disponibilidade faz com que se tenha um disfarce, caso venha ocorrer falhas, tanto no *hardware* quanto no *software*, e essa camuflagem faz com que o usuário do serviço não perceba o tempo de ociosidade. (LONEY; BRYLA, 2005).

A tecnologia *Oracle* oferece algumas ferramentas capazes de proporcionarem aos adeptos alta disponibilidade em seus serviços. Neste trabalho estudaremos duas dessas ferramentas *Oracle*: *Oracle RAC e Oracle Data Guard.*

Caso haja lapsos de atividade nos bancos de dados que estejam num local com alta disponibilidade, a utilização dessas ferramentas fará com que os bancos de dados permanecem disponíveis, pois o *Oracle* conserva a execução de suas aplicações.

3 ORACLE REAL APPLICATION CLUSTER (RAC)

Com os avanços e desenvolvimentos constantes no setor de sistema de informação, há um interesse maior pelas empresas em interligar seus sistemas, além de, um anseio de poder disponibilizar e adquirir informações, com total segurança e agilidade, tornando a consulta a seus bancos de dados sempre disponíveis (FERREIRA E LIMA, 2013).

> "À medida que aumenta o tamanho dos seus bancos de dados e o número de usuários, a necessidade de disponibilidade torna-se ainda mais crucial. O *Real Application Clusters* (RAC) unificará o OMF, espaços de tabela *bigfile*, uma infraestrutura robusta de rede e o *Automatic Storage Managementd* (ASM) a elementos-chave da arquitetura RAC" (LONEY; BRYLLA, 2005 p. 421).

Para se evitar problemas com disponibilidade de dados, a tecnologia *Oracle* oferece como uma de suas opções o *Oracle Real Application Clusters* (Oracle RAC), apresentando-o como um recurso para bancos de dados, possuindo um ou mais nós em *clusters*, controlados pelo sistema operacional. Quando a conexão com um dos nós se perde, há ainda a conexão com o outro nó. Esse recurso garante a disponibilidade dos serviços, fazendo com que estejam 100% disponíveis e escaláveis (FERREIRA; LIMA, 2013).

Há a possibilidade, durante a instalação do RAC, pode-se optar pelas ferramentas *Enterprise Manager* e o *Enterprise Manager Database e Control*, que são responsáveis pela gerência de um *cluster* (LONEY; BRYLA, 2005).

3.1 COMPUTAÇÃO DISTRIBUÍDA

A computação distribuída (*Grid Computing*) é baseada na distribuição de energia elétrica, servindo como base para a alta disponibilidade (LONEY; BRYLA, 2005).

É através da Computação distribuída que se tem um conceito mais ampliado dos meios computacionais, agrupados por redes, onde os processadores e mecanismos que o computador usa para armazenamento, sejam utilizados de uma maneira distribuída. Assim, usuários de qualquer parte do mundo, de empresas diferentes, conseguem fazer atividades que necessitem capacidade de armazenamento ou de processamento maiores.

> Ferreira e Lima apud Faria (2005) explicam que "a ideia central do *Grid Computing* (computação em grade / distribuída) é oferecer a computação como um serviço público. Não se deve ter preocupação com o local onde os dados residam ou qual computador processe a solicitação. O usuário deve ser capaz de solicitar e de receber informações ou recursos de computação, no volume e frequência que desejar. Podendo-se fazer uma analogia com o modo como funcionam os serviços públicos elétricos, já que não se sabe onde está o gerador, nem como é o funcionamento da rede elétrica. O que se deseja é eletricidade, e a consegue. O objetivo é tornar a computação um serviço público e onipresente."

Para quem está utilizando o sistema em *grid*, não é importante onde as informações estejam ou qual servidor responde as solicitações, mas sim que os anseios dos usuários sejam atendido prontamente. A computação em *Grid* fornece ao usuário um serviço sempre disponível e seguro (GUEDES; TECLES, 2014).

3.2 CLUSTERIZAÇÃO

Para Ferreira e Lima *apud* Alecrim "um *cluster* pode ser definido como um sistema onde dois ou mais computadores, que trabalham de maneira conjunta, para realizar processamento pesado", trabalhando como se fossem o mesmo computador.

Na ocorrência de alguma falha ou erro em alguns dos nós, nem o acesso realizado pelo cliente nem a disponibilidade do *cluster* serão atingidos (LONEY; BRYLA).

> "O *Oracle* RAC, cuidará de transmitir aos servidores sobreviventes às consultas que estavam "rodando" no servidor que falhou. Assim, um servidor sobrevivente, receberá as requisições oriundas do servidor falho, sem que o usuário perceba a troca de máquina que processava sua requisição". […] Um banco de dados RAC, permite que múltiplas instâncias *Oracle*, cada uma com uma estrutura, residam em diferentes servidores de um *cluster*, compartilhando os arquivos do banco de dados, ou seja, do mesmo conjunto de arquivos, *Data File, Control File e Redo Log file* (GUEDES; TECLES, 2014).

Siqueira (2010) define *redo log files* como estruturas alocadas nos bancos de dados, que refazem as transações que ficaram inativas, ou que apresentaram alguma indisponibilidade. Os *redo log files* contem as informações transacionais que conservam tanto os dados velhos quanto os novos.

Souza e Campos (2008), afirmam que as solicitações dos clientes são atendidas da seguinte forma: Primeiramente os clientes se conectam a um dos nós através de uma Rede Local. Os servidores, onde estão localizados esses nós, realizam suas obrigações, e fazem

com que os bancos de dados permaneçam seguros através da utilização de discos compartilhados (onde estão armazenados os bancos de dados e data files acessados pelos nós) e interconexão privadas.

> De acordo com Guedes e Tecles (2014) o Rac pode funcionar de duas formas: "Na primeira delas há um nível de *software* do *cluster* operando os nós, onde cada um desses nós pode inspecionar os outros nós por meio de uma interconexão de rede. Havendo uma inatividade no nó que gerencia, as outras que não apresentarem falhas, poderão se apoderar de sua memória, retomando suas atividades, enquanto a máquina que inspeciona permanece fora de serviço. Neste caso os clientes distinguem apenas uma pequeno cessação no acesso. Na segunda forma de funcionamento do *cluster* é o trabalho utilizando-se *cluster,* onde paralelamente cada nó controla o *cluster,* e a condução ao disco compartilhado dos bancos de dados será realizada por uma camada de *software.* Neste caso todos os nós podem acessar todos os discos contidos nos bancos."

3.3 *CLUSTERWARE*

Para Souza e Campos (2008) "*Clusterware* é o conjunto de *software* que faz o gerenciamento do *cluster*." É o principal item para configura-se o RAC, pois coordena o cluster e providencia a fusão entre os nós e os bancos de dados, abolindo falhas e fazendo com que os itens sejam alocados em outro lugar no caso da ocorrência de danos.

Proni (2014) afirma que o *Oracle Clusterware* é um sistema móvel, onde vários servidores colaboram com um único sistema, podendo trabalhar independentemente dando segurança a seus clientes, principalmente por garantir que os aplicativos estejam protegidos.

> Ferreira e Lima destacam que os principais componentes da *Clusteware* são:
> 1. *Cluster Synchronization Services* - CSS: É o fundamento para comunicação entre processos num cluster, fazendo a sincronização entre os membros.
> O CSS é a interação entre instâncias, *Automatic Storage Management* (ASM) e o banco de dados, fornecendo informações sobre cada máquina, e ainda, gerenciando bloqueios de registros no cluster.
> 2. *Cluster Ready Services* - CRS: É responsável pelo controle de alta disponibilidade e por se recuperar de falhas. Além disso, quando um dos nós falha ele transfere o serviço para outro nó, guardando os eventos que aconteceram no OCR (*Oracle Cluster Register)* que deve ser instalado em cada máquina do cluster. Ele tem três componentes essenciais que "se manifestam como *daemons* (processos) executados no *inittab* do *Linux* ou como um serviço, em um ambiente *Windows".* Além disso, ainda existe o *Virtual Ip Address* (VIP) que diminui o tempo do *failover* (ou *standby*) e faz com que cada máquina além de seu *Internet Protocol* (IP) ainda possua um que seja virtual e ligado a cada nó.
> 3. *Event Manager* - EVM: Notifica e chamadas do *Oracle Clusterware, guardando* os *logs* sobre cada evento dos *daemons (*processos) do RAC.

3.4 *CLUSTER READY SERVICES* (CRS)

> "O *Cluster Ready Services* (CRS, também conhecido como OCR) é a solução de *cluster* que pode ser utilizada nas principais plataformas em vez de um fornecedor de OS ou um *clusterware* de terceiros. O CRS é o responsável pelo processo de realocação dos componentes de uma instância e pelo controle da reedição destes em uma nova instância, além de recuperar parte das transações que falharam em algum nó" (LONEY; BRYLA, 2005 p.423).

A instalação dos CRS deve ser feita em um local próprio, no CRS_HOME, configurando essa instalação em dois lugares usados pelo *cluster*: o OCR e o disco de votação (OLIVEIRA, 2009).

3.5 ARQUITETURA DO RAC

Segundo Souza e Campos (2008) em um Oracle RAC, as máquinas que compõem o *cluster* de banco de dados operam em conjunto, com o objetivo de atender as solicitações de aplicações ou de usuários. O usuário que acessa o banco de dados não nota diferença comparando a uma conexão a banco de dados sem a utilização de um *cluster*.

Para que isso funcione são necessárias infraestruturas de *hardware* e *software*.

A Figura 1 ilustra um ambiente típico de RAC.

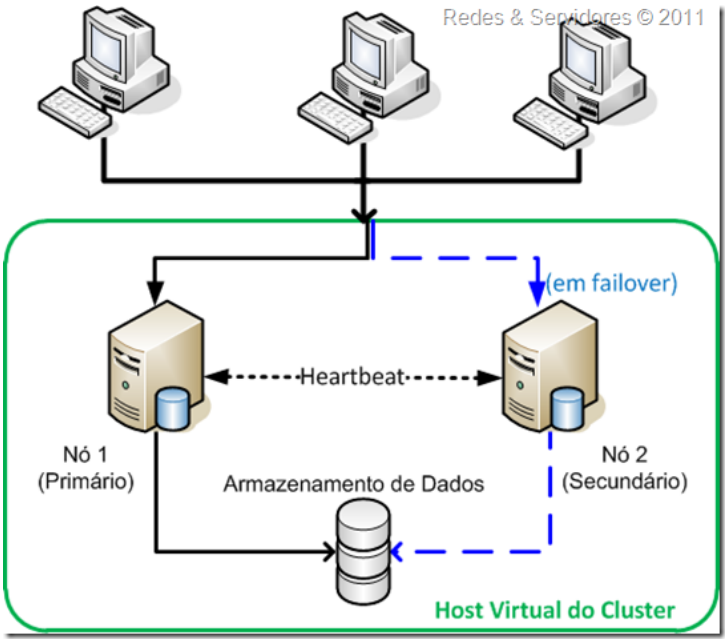

Figura 1 - Arquitetura RAC
Fonte: Fórum Imasters

3.6 CONFIGURAÇÕES

Segundo Oliveira et al (2009), "instalação do *Oracle* RAC requer algumas configurações e recomendações relacionadas ao *hardware*, *software* e topologia de redes."

3.61 Configuração de *Hardware*

Oliveira et al (2009) sugere que se tenha dois ou três nós para cada um dos bancos de dados RAC, cada uma com seu processador, memória disponível, fonte de energia, também devendo estar em cada um dos bancos de dados do *cluster.* O autor ainda afirma que, quanto mais máquinas configuradas no *cluster*, menor será o embate causado caso haja falha de alguma das máquinas.

> "Os subsistemas de discos compartilhados também devem ter a redundância de *hardware* pré-definida – Múltiplas fontes de alimentação de energia, [...] Divide a redundância pré-definida do disco compartilhado entre os tipos de grupos de discos que serão criados para o RAC. Uma redundância mais alta, pré-definida no *hardware* do subsistema de disco, pode reduzir potencialmente a quantidade de redundância do *software*, que você especifica, ao criar dos grupos de disco do banco de dados" (LONEY; BRYLA 2005 p 422).

3.6.2 Configuração de *Software*

> "O CRS é instalado antes do RDBMS, e precisa estar em diretório inicial próprio, conhecido como CRS_HOME. Se, no futuro próximo, você utilizar uma única instância, mas pensar um clusterizá-la, posteriormente será útil primeiro instalar o CRS, de modo que os componentes do CRS necessário para o ASM e RAC, já estejam na estruturas de do diretório RDBMS. Se não instalar o CRS primeiro, mais tarde, terá de seguir alguns passos extras para remover os executáveis relacionados ao processo CRS do diretório inicial do RDBMS" (LONEY; BRYLA, 2005 p.423).

Depois que o CRS é instalado, instale-se o *software* do banco de dados no diretório inicial, conhecido como ORACLE_HOME (LONEY; BRYLA, 2005).

3.6.3 Configuração de Rede

> "Cada nó em um RAC tem no mínimo três endereços IP: um para rede pública, um para interconexão de rede privada, e um endereço IP virtual para suportar *failover* mais rápido, no caso de uma falha no nó. Como resultado, no mínimo duas placas de rede físicas são necessários para suportar o RAC; as placas de redes adicionais são utilizadas para fornecer redundância na rede pública e assim um caminho de rede alternativo para as conexões recebidas. Para a rede privada, placas de redes adicionais podem aprimorar o desempenho, fornecendo mais largura de banda total, para o tráfego interconectado" (LONEY; BRYLA 2005 p.423).

No caso de conexões rotineiras, a rede de conexão utilizada é a rede pública. Para se ter a comunicabilidade entre os nós do cluster, as redes que suportam são a de interconexão ou rede privada. (OLIVEIRA et al, 2009).

3.7 VANTAGENS

Um banco de dados RAC, sendo altamente disponível e escalonável, permite que seus dados sejam acessados constantemente, e quando alguma falha em uma das maquinas do *cluster* ocorre, não altera a funcionalidade do *cluster*, até que haja falha no último nó. Dependendo da quantidade de nós, a única coisa perceptível será uma pequena demora no tempo de resposta de solicitação (FERREIRA; LIMA, 2013).

O RAC oferece ainda, uma grande capacidade para adicionar e remover as máquinas do *cluster*, atendendo assim a demanda atual. (RAMALHO, 2005).

Conforme ensinamentos de Oliveira et al (2009), entre as principais vantagens destacam-se:

- Se um nó falha, outro continua trabalhando, não afetando a acessibilidade ao banco de dados, e havendo assim, uma continuidade de serviço e uma alta disponibilidade.
- Escalabilidade sob a demanda: É possível aumentar ou diminuir a capacidade do RAC, adicionando ou removendo novos servidores ao *cluster*;
- *Desempenho Recorde* mundial: Foram realizado testes que demonstraram que o RAC é mais veloz do que um ambiente *mainframe*.

- Recuperabilidade: O sistema não cessa quando um nó falha, pois outro nó continua operando, não tendo assim interrupção de atividade devido à falha de uma máquina.
- Detecção de erros e manutenção facilitada

3.8 DESVANTAGENS

Um banco de dados Oracle RAC possui algumas desvantagens em relação à utilização de outros bancos de dados. Uma dessas desvantagens é o alto custo de licenciamento, pois é necessário uma licença *Oracle* para cada máquina do *cluster* (LONEY; BRYLA, 2005)

Segundo Oliveira et al (2009), entre as desvantagens pode-se citar:

- O *software* do RAC possui alguns custos complementares de armazenamento e administração.
- É necessário que o *Data Base Administrator* (DBA), tenha um bom conhecimento de sistema operacional, para que tenha uma facilidade na montagem e manutenção do RAC (FERREIRA; LIMA, 2013).

Loney e Bryla (2005) contemplam ainda que, ao se projetar um banco de dados em um ambiente RAC, deve-se pensar na utilização de CPU, para que sejam feitas as pesquisas e consultas mais pesadas, havendo assim, um pequeno aumento no custo de manutenção de um banco de dados RAC.

4. ESTRATÉGIA DE *BACKUP*.

> "Em todo sistema de banco de dados, sempre existe a possibilidade de uma falha de sistema ou de *hardware*. Caso ocorra alguma, que afete o banco de dados, ele deve ser recuperado. Os objetivos esperados depois de uma falha são garantir que os efeitos de todas as transações submetidas se reflitam naquele banco de dados que será recuperado, e que o retorno seja o mais rápido possível a operação normal, isolando os usuários dos problemas que foram causados" (RAMALHO, 2005 p. 29).

A *Oracle* dispõe de uma diversidade de opções de *backups,* que colaboram para seguridade das informações de um banco de dados. "Se bem usadas permitem um *backup* de qualidade para o banco, além de uma rápida recuperação. A *Oracle* conta com backups físicos e *backups* lógicos, cada um com opções disponíveis diferentes" (LONEY; BRYLA, 2005 p.466).

4.1 CONFIGURAÇÕES DE *BACKUP*

Há três métodos padrão para se realizar *backup* de um banco de dados *Oracle*: exportações, *backups off-line* e *backups on-line*. "Uma exportação é um *backup lógico* do banco de dados. Os outros dois métodos de *backup* são *backups físicos* de arquivos" (LONEY; BRYLA 2005).

4.2 *BACKUP* LÓGICO

Um *backup* lógico de um banco de dados lê os registros de um banco, independentemente de seus locais físicos, gravando esses registros em um arquivo. O responsável por esse *backup* no banco de dados do *Oracle* é o *Data Pump Export*. Para recuperação, com o arquivo gerado em um *Data Pump Export*, é necessário utilizar o *Data Pump Import* (LONEY, BRYLA, 2005).

O *Data Pump Export* realiza a consulta ao banco de dados e ao dicionário do banco, além de, salvar a saída em um arquivo de *dump* de exportação (arquivo XML), contendo comandos necessários para recriar e compartilhar os dados selecionados, conforme compartilha Oliveira et al 2009 apud WATSON; BERSINIC, (2006).

> "Depois que os dados foram exportados por meio do *Data Pump Export*, eles poderão ser importados por meio do utilitário *Data Pump Import*. O *Data Pump Import* lê o arquivo *dum*, criado pelo *Data Pump Export*, e executa os comandos localizados. Por exemplo, esses comandos podem incluir: comando *CREATE TABLE*, seguido de um comando *INSERT* para carregar dados para tabela" (LONEY; BRYLA, 2005 p. 467).

Usando o arquivo de *dump* de exportação, as informações ou parte das que não foram exportadas, serão importadas para o mesmo banco de dados ou para o mesmo esquema, criando-se assim uma duplicação de objetos exportados perante um esquema, ou em diferentes bancos de dados. Caso haja uma exportação completa, serão criados todos os bancos de dados durante a importação (LONEY; BRYLA 2005).

4.3. *BACKUPS* FÍSICOS

Backups físicos condizem à cópia dos arquivos que constituem o banco de dados. O *Oracle* comporta até dois tipos diferentes de *backups* físicos de arquivos: *backups off-line* e *backups online*, podendo-se ainda usar o utilitário RMAN (*Recovery Manager*) para realizar todos os *backups* físicos (LONEY; BRYLA, 2005).

4.3.1 *Backups Off-line*

Loney e Bryla (2005) enfatizam que *Backups off-line* acontece quando o banco de dados é desligado de maneira normal, ou seja, sem falha de uma das máquinas. Neste momento, deve-se fazer o *backup* de todos os arquivos de dados, de controle e *logs* de rede *on-line* e ainda pode fazer opcionalmente do arquivo *init.ora*, e parâmetro de servidor (*spfile*) caso esteja sendo utilizado.

> "Fazer o *backup* de todos esses arquivos, enquanto o banco de dados esta desligado, fornece uma imagem completa do banco de dados, tal como ele existia no momento em que foi desligado. [...] Não é valido realizar um *backup* do sistema de arquivos do banco de dados enquanto ele está aberto, a menos que, um *backup on-line* está sendo realizado. *Backups off-line* que ocorrem depois da interrupção do banco de dados, também serão considerados inconsistentes, e talvez requeiram mais esforços para que possam ser utilizados durante a recuperação" (LONEY; BRYLA, p. 468).

4.3.2 *Backups On-line*

De acordo com Ramalho os *backups* on-line podem ser utilizados, para qualquer banco de dados que estejam em execução no modo ARCHIVELOG, guardando os *logs* de *redo on-line* neste modo, e criando um registro de todas as transações, que ocorrem dentro do banco de dados (RAMALHO, 2005).

> "O *Oracle* grava nos arquivos de *redo log on-line* de uma maneira cíclica: depois de preencher o primeiro arquivo de *log*, começa-se a gravar o segundo, até que este esteja preenchido, e então se começa a gravar o terceiro. Depois que o ultimo arquivo de *log* de *rede on-line* é preenchido, o segundo plano do LGWR (*Log Writer*) começa-se a sobrescrever o conteúdo do primeiro arquivo do *redo log* Quando o *Oracle* é executado no modo *ARCHIVELOG*, o processo do segundo plano do ARCH (*Archiver*), cria uma cópia de cada arquivo de *redo log* antes de sobrescrevê-lo. Esses arquivos de *redo log* arquivados em repositórios de arquivos normalmente são gravados em um dispositivo de disco. Os arquivos de *redo log* arquivados em repositórios de arquivos também podem ser gravados em fita, mas isso tende a utilizar o operador intensivamente" (LONEY; BRYLA, 469).

4.4 SUGESTÕES DE *BACKUPS*

Nos *backups* físicos pode-se recompor o banco de dados, a partir de cada um dos *backups* anteriores, restaurando o banco até o ponto atual no tempo ou em algum ponto há a segurança de que a transação que não confirmada se perca, possibilitando assim, fazer uma restauração do banco através do anterior até o ponto atual no tempo ou qualquer ponto intermédio, assim ensina Oliveira et al (2009).

Já em relação aos *backups* lógicos, Loney e Bryla (2005) ensinam que, o DBA ou usuário pode apossar-se do teor de objetos e de dados, individualmente, em algum período no

tempo, oferecendo uma possibilidade recuperação alternativo, havendo um embate no resto do banco de dados, que possua uma restauração de banco de dados completa (LONEY; BRYLA, 2005).

O *Recovery Manager* (RMAN) realiza a leitura dos blocos de dados, fazendo com que o usuário não tenha que inserir manualmente, numa situação de *backup*, em cada lacuna de tabela (OLIVEIRA, et al, 2009).

> O *Recovery Manager* (RMAN) do *Oracle* leva o *backup* e a recuperação a um novo nível de proteção e facilidade de uso. Desde o aparecimento do RMAN, na versão oito do Oracle, há varias melhoras e aprimoramentos importantes que podem fazer do RMAN uma solução única para quase todo o ambiente de banco de dados (LONEY; BRYLA, 2005 p.495).

4.4.1 Componentes do RMAN

RMAN é o primeiro item no ambiente executável, a partir de um *prompt* de comando RMAN, disponível junto no diretório $ORACLE_HOME/Bin e podendo ser solicitado com ou sem linha de comando. O RMAN pode ser utilizado para tornar mais fácil, a constituição de um banco de dados físico de espera, esses que são gerados de cópias de segurança de seu banco de dados primário ou criados em um banco de dados de espera na rede (LONEY; BRYLA, 2005).

> O RMAN é mais do que um simples executável do lado do cliente que pode ser utilizado com uma *interface Web*. Ele abrange vários outros componentes, inclusive o banco de dados do qual será feito o *backup* (Banco de dados Alvo), um catalogo de recuperação opcional, uma *Flash Recovery Area* opcional e um *software* de gerenciamento de mídia para suportar sistemas de *backup* de fita (LONEY; BRYLA, 2005 p. 496).

5 ORACLE *DATA GUARD*

Loney e Bryla (2005) definem o *Oracle Data Guard* como "uma infraestrutura de *software*", que possibilita manter um ou vário bancos de dados em alerta, caso um deles venha precisar de algum tipo de manutenção por ter sofrido algum tipo de dano ou falha.

> *"Data Guard* é um recurso de banco de dados *Oracle*, cujo objetivo é garantir alta disponibilidade, proteção e recuperação de dados corporativos. O *Data Guard* é composto de um pacote de serviços de criação, manutenção, gerenciamento e monitoração de um ou mais bancos de dados em modo de espera (*standby*), o que garante que bancos de dados de produção sobrevivam a desastres e corrupção de dados. O *Data Guard* mantém esses *standby databases*, como cópias do banco de produção. Então, se um serviço de banco de dados de produção torna-se indisponível devido a uma interrupção, planejada ou não, o *Data Guard* pode redirecionar o serviço para o *standby data base*, minimizando o *downtime* e os transtornos causados pela indisponibilidade do serviço" (SILVA, 2008).

5.1 ARQUITETURA DO *DATA GUARD*

> "O *Oracle Data Guard* fornece uma solução para alta disponibilidade, desempenho aprimorado e *failover* automatizado. Você pode utilizar o *Oracle Data Guard* para criar e manter múltiplos bancos de dados de reserva (*standby*), para um banco de dados principal. Os bancos de dados de reserva podem ser iniciados no modo somente leitura para suportar usuários que geram relatórios, e podem ser retornados ao modo de reserva. As alterações no banco de dados principal podem ser automaticamente transmitidas, do bando de dados principal para os bancos de dados de reserva, com a garantia de que os dados não serão perdidos no processo" (LONEY; BRYLA, 2005 p.552).

O *Data Guard* oferece proteção aos dados dos bancos, gerando ou mantendo um ou mais bancos como *standby*, assim em caso de uma inatividade causada pela destruição da base de dados primária, fazendo com que os bancos não percam suas informações em determinado local físico. O *data guard* através dos bancos em *standby* faz a recuperação desses dados (SOARES, 2013).

De acordo com Armbrust (2013) os bancos de dados de *standby* só serão ativados quando tiver a necessidade de utilizá-los, ou ainda quando ocorrer algum problema no banco de dados primário.

> "O *Data Guard* transmite automaticamente informações do *redo* para os bancos de dados de reserva, onde elas são aplicadas. Como resultado, o banco de dados de reserva é transacionalmente consistente. Dependendo de como você, configura o processo de aplicação do *redo*, os bancos de dados de reserva, podem estar em sincronia com o banco de dados principal ou ficar atrás dele. Os dados de *log* de redo são transferidos para os bancos de dados de reserva via *Log Transport Services*. Os *Log Apply Services* aplicam nas informações de *redo* aos bancos de dado de reserva" (LONEY; BRYLA, 2005 p. 552 e 553).

Para Campos e Souza (2008) para que o *Data Guard* funcione, existem alguns requisitos necessários, como ter o mesmo sistema operacional nos *hosts* (ainda que com versões diferentes), ter a mesma versão do *Oracle Database Enterprise* e cada banco ter seu próprio *control file.* Há ainda, necessidade de os bancos primários estarem em *ARCHIVE LOG MODE* e *FORCE LOGGIN* e, caso haja ASM e/ou OMF cada *standby* precisar ter também.

A Figura 2 ilustra o funcionamento do *Data Guard também.*

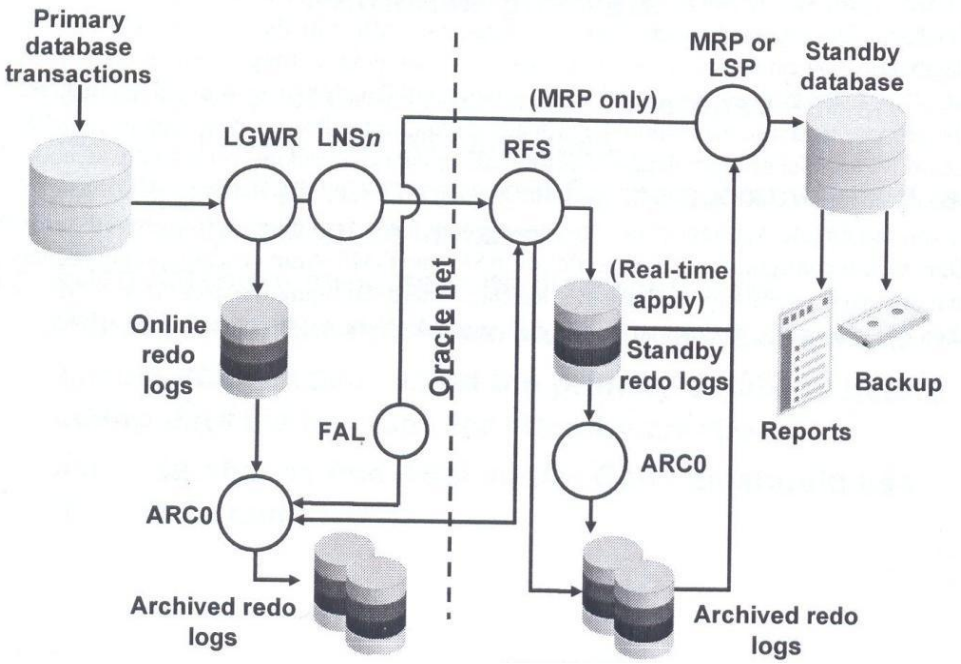

Figura 2 - KEESLING, 2006

5.2 BANCOS DE DADOS DE RESERVA FÍSICA E BANCO DE DADOS DE RESERVA LÓGICA

> "Dois tipos de bancos de dados de reserva são suportados: de reserva física e de reserva lógica. O banco de dados de reserva física tem as mesmas estruturas do banco de dados principal. O banco de dados de reserva lógica pode ter estruturas internas diferentes [...]. Você sincroniza um banco de dados de reserva lógica com o principal, transformando os dados de *redo* em instruções SQL, que são executadas no banco de dados de reserva" (LONEY; BRYLA, 2005 p.553).

Os bancos de reserva física são cópias perfeitas de seu banco de dados bloco a bloco. Os dados que são recebidos do banco principal pelo MPR (Processo de Recuperação Gerenciada), faz o reconhecimento do *Data Guard*, sendo executado em paralelo, potencializando assim, o tempo de aplicação de *Redo Logs* e seu desempenho (ORACLE, 2009).

> Segundo Armbrust (2013), "seus principais Aspectos são:
> - Fisicamente idêntico ao banco de dados principal;
> - Enquanto o banco principal está trabalhando, o *standby* pode estar em dois modos: a) Modo de recuperação – logs de alterações gerados no banco principal são enviados e aplicados no *standby*, bloco a bloco; b) Aberto somente para leituras – usado para consultas;
> - Estruturas em disco idênticas;
> - Cada bloco de dados do banco *standby* é exatamente igual ao do banco principal."

Já os bancos de dados de Reserva Lógica (*Logical Database*) se diferenciam por sua organização física e por sua estrutura, permanecendo sincronizado através do *SQL Apply*, possuindo as mesmas informações lógicas que o banco principal (ORACLE, 2009).

Para criar-se um ambiente de reserva (*standby)*, deve-se utilizar *backup* do Ambiente produtivo, sincronizando o banco primário com todos seus bancos de dados de reserva, por uma aplicação de *Redo Logs* ou pela SQL (ARMBRUST, 2013).

5.3 SERVIÇOS DE TRANSPORTE DO *DATA GUARD*

Quando é realizado um *COMMIT* em uma transação no banco de dados, essas mudanças são escritas em um arquivo de *redolog local,* e enviadas pelo *Data Guard* , do

banco de dados primário para o de reserva, através de transporte síncrono ou assíncrono (LONEY; BRYLA, 2005).

5.3.1 Transporte de Redo Síncrono (SYNC)

O banco de dados primário espera o banco de dados de reserva confirmar a descarga do *Redo* em disco, para então depois fazer o *COMMIT*, assegurando-se assim, que os dados não serão perdidos. O desempenho do banco de dados primários, está relacionado de modo direto, ao tempo gasto para enviar o *redo log* para o banco de dados de reserva. (chamado de tempo de rede) (ARMBRUST, 2013).

> "O Transporte de dados de recuperação síncrono (SYNC) faz com que o banco de dados principal, aguarde por uma confirmação do banco de dados em *standby*, de que os dados de recuperação foram gravados em disco, antes de reconhecer o sucesso da confirmação para o aplicativo, proporcionando proteção, com zero de perda de dados. O desempenho do banco de dados principal é impactado pela soma do tempo necessário, para que a E/S do arquivo de *redo log* de *standby* seja concluída e o tempo do percurso de ida e volta da rede" (ORACLE, 2009).

5.3.2 Transporte de *Redo* Assíncrono (ASYNC)

Esse tipo de transporte evita que se tenham problemas no desempenho do banco de dados primário, porém, há a possibilidade que com esse transporte, o banco de reserva perca alguma informação, pois não há a certeza nem garantias de que houve a aplicação dos *redo logs*, já que não existe a confirmação alguma (ARMBRUST, 2013).

5.4 FUNCIONALIDADE DO *DATA GUARD*

Conforme já visto, e segundo a Oracle (2009), ao serem confirmadas as transações dos usuários no banco principal, o *Oracle* criará arquivos de registros de recuperação, gravando-os em um arquivo *log*, que serão posteriormente transmitidos por para o banco de dados de reserva por transporte síncrono ou assíncrono. Há a opção de comprimir o formato dos dados de recuperação, diminuindo a largura de banda, sendo essa opção realizada através da Compressão avançada *Oracle*.

5.5 SERVIÇOS DE APLICAÇÃO DO *DATA GUARD (APPLY SERVICES)*

O *Apply Service* utiliza dois mecanismos para sincronizar no banco de dados reserva: *redo apply* (reserva física) ou *SQL apply* (reserva lógica), responsáveis por interpretar os *redo logs* do banco de dados principal. Como as aplicações são independentes entre si (atuam individualmente), não causando qualquer conflito no trafego dos *redo logs*, do banco de dados principal para o banco reserva, como ressalta Armbrust (2013).

A principal parte do processo do ponto de recuperação é o transporte de dados de recuperação, pois se algo se abalar de forma negativa durante o transporte, aumentará a possibilidade de perda de dados. Se o transporte dos dados de recuperação estiver configurado sincronamente, poderá também ter um aumento do tempo de resposta e *through put* do banco de dados principal. Ao isolar-se o transporte e a aplicação, espera-se um maior desempenho e segurança do banco de dados, e uma diminuição do tempo de resposta (ORACLE, 2009).

5.6 VALIDAÇÃO DE DADOS

A arquitetura totalmente ajustável associada do *Data Guard,* permitem que os bancos de dados em *standby* se mantenham sincronizados, aplicando os blocos de recuperação, plenamente desagregados de corrupções, na E/S (Entrada/ Saída) no *primary database*. Durante o transporte e a aplicação dos dados de recuperação, são feitas apurações para checar a existência de dados corrompidos (ORACLE, 2009).

> "Uma das grandes vantagens do *Oracle Data Guard* é capacidade de validação dos *Redo Logs* antes da aplicação no *Standby Database*. Somente são aplicados os *Redos* recebidos do *Primar,y* depois de uma validação e garantia de que não existam blocos corrompidos ou algum tipo de inconsistência de dados" (ARMBRUST, 2013).

5.7 GERENCIANDO UMA CONFIGURAÇÃO DO *DATA GUARD*

Uma opção é a utilização do *Broker Framework,* que é uma ferramenta para facilitar o gerenciamento, manutenção e o controle do funcionamento e disponibilidade do *Data Guard,* podendo ser acessado pelo *Entriprise Manager* ou pelo aplicativo DGMGRL (SOUZA; CAMPOS, 2008).

> Para Armbrust (2013) "a alteração de um banco *standby* para primário é por meio de roles e se dá de duas maneiras:
> 1. *Switchover*: ocorre a inversão das roles nos bancos e não há perda de dados;
> 2. *Failover*: é desconsiderado o primário anterior, que sai da configuração atual, e o *standby* passa para a *role primary*. Aqui há risco de perda de dados."

O *failover* para um sistema em *standby* pode ser feito de forma manual ou automática (ORACLE, 2009)

O banco de dados em *sdandby* ao ser ativado passa a ser o banco primário, e o banco primário passa a ser o *standby*. Essa troca pode ser desfeita, através de um *switchback*, neste caso, o atual banco em reserva volta a ser o primário e o primário volta a ser o *standby* (SOARES, 2012).

5.8 *ENTERPRISE MANAGER*

O *Oracle Enterprise Manager* (OEN) é ferramenta responsável por facilitar o gerenciamento de uma infraestrutura *Oracle*, e caso havendo um agente responsável pelo gerenciamento para aplicações de terceiros, o OEM pode também gerenciar essa aplicação no mesmo *framework* (LONEY; BRYLA, 2005).

"A arquitetura de estrutura do *Enterprise Manager* fornece um alto nível de flexibilidade e funcionalidade. É possível personalizar facilmente o *Enterprise Manager* de acordo com as necessidades de monitoramento e administração do ambiente" (RAMALHO, 2005 p.345).

Através *do Enterprise Manager* há a possibilidade de configuração de alarmes de notificação, que avisem aos administrados caso a métrica ultrapasse o valor do limite configurado, simplificando-se assim, a configuração do *Data Guard* (ORACLE, 2009).

5.9 RESOLUÇÃO AUTOMÁTICA DE FALHAS

Quando o banco de dados principal e de reserva se desconectarem por uma falha na rede ou no servidor em *standby*, o banco de dados primário continuará processando as transações, gerando *logs* de dados de recuperação, que não serão emitidos ao banco de dados em *standby*, até que uma nova conexão seja estabelecida. Enquanto a conexão não é reestabelecida, o *Data guard* monitora de maneira continuada o banco de dados em *standby*, até que seja estabelecida novamente a conexão e assim, sincroniza automaticamente o banco de dados principal com o de *standby*, não necessitando uma intervenção administra caso os arquivos de *logs* estejam disponíveis no banco de dados principal (ORACLE, 2009).

Se esse tempo desconectado for muito extenso, não é vantajoso guardar os *logs* arquivados, e um *standby físico* poderá ser de novo sincronizado por um *backup* incremental do RMAN do banco primário (LONEY; BRYLA, 2005).

5.10 MODOS DE PROTEÇÃO

Segundo Souza e Campos, a arquitetura flexível do *Data Guard* oferece proteção de dados e disponibilidade ideais, oferecendo três modos de proteção:

> "1. Máxima proteção: o *commit* é feito após a confirmação de pelo menos um banco de dados em *standby*. O banco de dados para se não conseguir confirmar a gravação no *standby*.
> 2. Máxima disponibilidade: o *commit* é feito após a confirmação de pelo menos um banco de dados em *standby*. Se ocorrer uma falha na gravação no *standby*, o Banco opera como máxima desempenho até conseguir operar novamente como máxima disponibilidade;
> 3. Máxima desempenho:o *commit* é feito sem a confirmação da gravação no *standby*".

5.10.1 Máxima Proteção

> "No modo de proteção máxima (ou nenhuma perda de dados) pelo menos uma localização de reserva deve ser gravada antes que uma transação seja confirmada no banco de dados principal. O banco de dados principal desliga se a localização de *log* do banco de dados de reserva estiver indisponível (LONEY; BRYLA, 2005 p.554)**."**

Segundo Armbrust (2013), suas principais características são:

- Elevado nível de proteção de dados;
- Configuração: LGWR SYNC;
- Segurança a cada transação;
- Transação liberada apenas após o *redo* ser aplicado em ambos os bancos de dados;
- O ambiente produtivo é parado caso ocorra algum problema com o site *standby;*
- Sincronismo entre dois sites distintos;

5.10.2 Máxima Disponibilidade

> No modo de disponibilidade máxima, pelo menos uma localização de reserva deve ser gravada antes que uma transação seja confirmada no banco de dados principal. Se a localização de reserva não estiver disponível, o banco de dados principal não é desligado. Quando a falha for corrigida, o *redo* que foi gerado desde a falha é transportado e aplicado ao banco de dados de reserva (LONEY; BRYLA, 2005 p. 554).

Armbrust (2013) ressalta que as principais características são:

- Proteção a cada transação;
- Configuração: LGWR SYNC;
- As transações só serão liberadas depois de o *redo* ser aplicado em ambos os sites;
- Ainda que ocorra algum problema com o site em standby, o ambiente de produção continua ativo.
- Quando o ambiente *standby* é reconectado, a sincronização é feita de forma automática;
- Pré-requisito para *Failover* automático (*Fast-start failover*);

5.10.3 Máxima Desempenho

Para Maciel et al (2012), este é o modo padrão e apresenta o mais alto nível de proteção de dados possível, sem que o desempenho do banco de dados seja afetado.

> "No modo de desempenho máximo (o padrão), as transações podem ser confirmadas antes que suas informações de *redo* sejam enviadas para as localizações de reserva. As confirmações no banco de dados principal ocorrem logo que as gravações no logs de *redo* on-line locais são completadas. As gravações nas localizações de reserva são tratadas pelo processo *ARCH* por padrão" (LONEY; BRYLA, 2005 p. 554).

Para Armbrust (2013), as principais características são:

- Alto desempenho;
- Possui a configuração: LGWR ASYNC, ou ARCH;

- Embate pequeno ao ambiente de produção;
- É utilizado para aplicações que toleram perda de dados;

5.11 VANTAGENS DO *DATA GUARD*

Uma das grandes vantagens do *Data Guard* é que ele garante que os serviços não sejam interrompidos, minimizando o tempo e uma inatividade.

Através dos bancos de dados de reserva, o *Data Guard* fornece consultas *ad-hoc*, com gerando relatórios, *backups* ou atividades de teste e ainda proporcionando segurança caso um desastre natural aconteça (MACIEL et al, 2012).

> Segundo a Oracle (2009) o uso do *Data Guard* proporciona algumas vantagens aos clientes Oracle. Entre elas:
> "• Utilização eficiente da rede: é uma solução de recuperação de catástrofe com reconhecimento do banco de dados que utiliza com eficiência os recursos de rede, transmitindo somente as alterações mínimas necessárias para manter o banco de dados de *standby* sincronizado;
> • Proteção de dados abrangente: evita que as corrupções de dados sejam propagadas aos bancos de dados de *standby*;
> • Econômico: é uma solução com reconhecimento de armazenamento, reduzindo os custos de *hardware* e a dependência de um único fornecedor.
> • Utilização do *standby*: As opções de configuração flexíveis do *Data Guard*, permitem que os clientes utilizem o banco de dados de *standby* e servidores para *backup*, geração de relatórios e outras finalidades;
> • Gerenciamento integrado: Configurações, monitoração e manutenção de *standby*, incluindo *failover e switchover*, são facilmente gerenciadas pelo *Oracle Enterprise Manager Grid Control*.".

6 TESTE

Depois de instalado o banco de dados, foi realizado um teste de perda de um dos nós, simulando assim, uma falha inesperada, como o acontecimento de uma indisponibilidade de seus serviços pela destruição do local físico onde se encontra a base de dados primaria, havendo a perda repentina do acesso as informações e dados contidos em um banco altamente disponível. O objetivo era avaliar se a infraestrutura do banco de se matéria disponível e acessível, no caso da perda de um nó do cluster. Com a parada de um dos nós foi testado o outro nó e obteve-se o seguinte status:

```
[oracle@rac2 ~]$ /u01/app/oracle/product/10.2.0/crs/bin/crs_stat -t
Name           Type         Target    State     Host
ora....C1.inst application    OFFLINE   OFFLINE
ora....C2.inst application    ONLINE    ONLINE    rac2
ora.RAC.db     application    ONLINE    ONLINE    rac2
ora....SM1.asm application    OFFLINE   OFFLINE
ora....C1.lsnr application    OFFLINE   OFFLINE
ora.rac1.gsd   application    OFFLINE   OFFLINE
ora.rac1.ons   application    OFFLINE   OFFLINE
ora.rac1.vip   application    OFFLINE   OFFLINE
ora....SM2.asm application    ONLINE    ONLINE    rac2
ora....C2.lsnr application    ONLINE    ONLINE    rac2
ora.rac2.gsd   application    ONLINE    ONLINE    rac2
ora.rac2.ons   application    ONLINE    ONLINE    rac2
ora.rac2.vip   application    ONLINE    ONLINE    rac2
```

Com um dos nós do RAC parado, foi criado uma tabela de teste e inserido uma linha na tabela:

```
[oracle@rac1 ~]$ sqlplus / as sysdba
```

```
SQL> create table employed
(employed_id int not nul,
firstName varchar(100),
lastName varchar(100),
primary key (employed_id));

SQL> insert into employed (employed_id, fisrtName, lastName) values (1, ‘Tirso’, ‘Mevio');
```

Após a criação da tabela foi iniciado o rac1 e parado o rac2.

```
[oracle@rac1 ~]$ /u01/app/oracle/product/10.2.0/crs/bin/crs_stat -t
Name           Type          Target    State     Host
ora....C1.inst application    ONLINE    ONLINE    rac1
ora....C2.inst application    OFFLINE   OFFLINE
ora.RAC.db     application    OFFLINE   OFFLINE
ora....SM1.asm application    ONLINE    ONLINE    rac1
ora....C1.lsnr application    ONLINE    ONLINE    rac1
ora.rac1.gsd   application    ONLINE    ONLINE    rac1
ora.rac1.ons   application    ONLINE    ONLINE    rac1
ora.rac1.vip   application    ONLINE    ONLINE    rac1
ora....SM2.asm application    OFFLINE   OFFLINE
ora....C2.lsnr application    OFFLINE   OFFLINE
ora.rac2.gsd   application    OFFLINE   OFFLINE
ora.rac2.ons   application    OFFLINE   OFFLINE
ora.rac2.vip   application    OFFLINE   OFFLINE
```

Com a nó rac1 ativo, foi realizado uma consulta para verificar a integridade da RAC.

```
SQL>  select * from employed;
EMPLOYED_ID       FIRSTNAME       LASTNAME
-------------------       ----------------       ----------------
                  1                 Tirso               Mevio
```

6.1 RESULTADO

Durante o teste realizado, percebeu-se que o banco de dados continuou funcionando, mesmo com a perda de um dos nós, não se desconectando do banco. Ou seja, uma das máquinas falhou e outra continuou a operar, evitando a inatividade, tornando quase que imperceptível a falha, a não ser por um pequeno atraso no tempo de resposta.

7 CONCLUSÃO

Com os avanços tecnológicos, há uma grande preocupação das empresas em manterem seus dados sempre disponíveis. Uma indisponibilidade de seus serviços, causada pela destruição do local físico onde se encontra a base de dados primária (decorrente de causas naturais ou perca imprevisível), pode prejudicar a produtividade da empresa, caso não haja medidas para combater uma suposta indisponibilidade.

As organizações que dependem de seus bancos de dados para processar transações e organizar informações devem considerar todas as opções para reduzir uma possibilidade de inatividade e consequentemente à perda de dado.

Neste trabalho foi estudado a continuidade da disponibilidade no acesso as informações, no caso do acontecimento de alguma inatividade, causada por meio natural ou por um motivo não esperado. O objetivo era ter uma alta disponibilidade nos bancos de dados, através das ferramentas *Oracle Rac* e *Data Guard*.

Com a pesquisa e o teste pôde-se concluir que a *Oracle* oferece uma solução adequada para alta disponibilidade em banco de dados, e que a proposta que foi apresentada mostrou-se adequada a suportar o tipo de falha sugerida pelo trabalho, no caso, a continuidade dos serviços numa inatividade não esperada.

Em contrata partida percebeu-se que o sistema apresentado necessita de uma complexidade no projeto para implantação superior a pensada, atualmente empresas utilizam-se dessas aplicações esperando manter seus sistemas sempre atuantes, porem não estudando o balanço de carga caso aconteça efetivamente a perca do sistema, um ponto muito agravante é o “efeito domino”, ocasionado quando o RAC opera no ápice das conexões, ou acima de um patamar que os nós sofram sobrecarga do que possa operar, caso ocorra falha em um dos nós.

Apesar de ser financeiramente mais caro e ter uma administração completa, o *Oracle* RAC em conjunto com o *Data Guard, é uma* opção eficaz quando se fala em alta disponibilidade e, mesmo havendo alguma inatividade inesperada e imprevisível, ainda mantém o banco de dados íntegro e disponível (LONEY; BRYLA, 2005).

A *Oracle* ainda disponibiliza para seu banco de dados uma ferramenta para gerenciá-los: *o Enterprise Manager*. (LONEY; BRYLA, 2005)

O *Data Guard* permite aos clientes consultar, fazer relatórios e *backups* utilizando o banco de dados de reserva, ou ate mesmo disponibilizando bancos de dados, com os registros reais, para a linha de produção de novos projetos para empresa.

Ao realizar os testes, pôde-se notar que ainda que uma das máquinas seja desligada ou perca o acesso por qualquer outro motivo, outro nó continuará em atividade, fazendo com que seja quase imperceptível o período de inatividade.

O *Data Guard* tem um papel significativo quando trabalhado conjuntamente com o RAC. Ele complementa uma configuração RAC, e mantém um banco de dados de *standby* em um local remoto, para garantir proteção de dados e alta disponibilidade.

Um ponto a ser levantado pelo estudo desse projeto é até que ponto o custo/beneficio dessa plataforma deve ser implementado, pois em se tratando por exemplo para empresas como hospitais que trabalham com vidas, (considerando a possibilidade de perdê-las caso o sistema venha a ficar inoperante) é primordial que não se tenha uma inatividade. Para outros casos necessita-se de um levantamento, detalhado, considerando se existe a necessidade de investimento de tal proporção.

8 TRABALHOS FUTUROS

Para trabalho futuro tem-se como objetivo estudar análises de custo de implementação e testes em ambiente de produção.

REFERÊNCIA BIBLIOGRÁFICA

FARIA, Elizabeth. **Grid e Oracle 10g**. Oracle Corporation, 2005.

LONEY, Kevin; BRYLA, Bob. **O Manual do DBA.** São Paulo: Campus, 2005

RAMALHO, José Antônio. **Oracle 10 g**. São Paulo: Pioneira Thomson Learning, 2005

ALECRIM, Emerson. **Cluster: principais conceitos.**
Disponível em: <http://www.infowester.com/cluster.php> Acesso em 28 de agosto de 2014.

ARMBRUST, Victor. **Oracle Data Guard – Conceitos e arquitetura**.
Disponível em: <http://imasters.com.br/banco-de-dados/oracle/oracle-data-guard-conceitos-e-arquitetura/>. Acesso em 18 de setembro de 2014.

BABINEAU, Brian. **Oracle Data Guard dobra proteção para o banco de dados e recuperação de catástrofe**. Disponível em: <http://www.oracle.com/technetwork/pt/database/dtp-esg-wp-oracle-data-guard1.pdf>. Acesso em 22/11/2014

BORGES, Anselmo. **Oracle Data Guard – Função e principais parâmetros.**
Disponível em: <http://www.anselmodba.com/2011/12/28/oracle-dataguard-funcao-e-principaisparâmetros/>. Acesso em 15 de setembro de 2014.

FERREIRA, Eduardo Amaral; LIMA, Iremar Nunes de. **Funcionalidades do Oracle RAC**. Disponível em: <http://blog.newtonpaiva.br/pos/wp-content/uploads/2013/02/E2-SI-29.pdf> Acesso em: 05 de novembro de 2014.

GUEDES, Anne Margareth de Souza; TECLES, José Eduardo T. **Artigo SQL Magazine 0 – Oracle 10g, um Banco de dados para Computação em Grid.** Disponível em: <http://www.devmedia.com.br/artigo-sql-magazine-9-oracle-10g-um-banco-de-dados-para-computacao-em-grid/7569#ixzz3LJRlU7. Acesso em 07 de dezembro de 2014.

MACIEL, Fabiane Telles L. et al. **Oracle Data Guard.** Disponível em: < http://pt.slideshare.net/washingtonlvaz/oracle-data-guard>. Acesso em 05 de dezembro de 2014

NETWORK, Oracle Technology (Org.). **Instalação.** Disponível em: <http://download.oracle.com/docs/cd/B25329_01/doc/install.102/b25144.pdf>. Acesso em: 22 de abril 2014.

ORACLE (Org.). **Oracle.** Disponível em: <http://download.oracle.com/docs/pdf/B25801_01.pdf>. Acesso em: 03 jul. 2014.

ORACLE (Org.). **Oracle Real Application Clusters**. Disponível em: <http://www.oracle.com/br/products/index.html> Acesso em: 22 de setembro de 2014.

ORACLE (Org.). **Oracle.** Disponível em: <http://download.oracle.com/docs/pdf/B25801_01.pdf>. Acesso em: 03 julho 2014.

ORACLE (Org). **Oracle Data Guard com Oracle Database 11g - Release 2.** Disponível em: <http://www.oracle.com/technetwork/pt/database/enterprise-edition/documentation/data-guard-com-database-11g-r2-1721669-ptb.pdf>. Acesso em 22 de novembro de 2014.

ORACLE. **Oracle Database 11g com Alta Disponibilidade.** Disponível em: <http://www.oracle.com/technetwork/pt/database/enterprise-edition/documentation/alta-disponibilidade-database-11g-431926-ptb.pdf>. Acesso em 12 de novembro de 2014.

OLIVEIRA, Pamela Cristina Machado de. Et al. **Alta disponibilidade em Sistema de Banco de Dados Oracle**. Disponível em: http://engenharia.anhembi.br/tcc-09/cco-03.pdf. Acesso em 22 de novembro de 2014.

PRONI, Ricardo Portilho**.** Artigo SQL Magazine 67 - **Oracle RAC - Parte 1**. Disponível em: <http://www.devmedia.com.br/artigo-sql-magazine-67-oracle-rac-parte 1/14033#ixzz3LbRu1TAh> Acesso em: 22 de novembro de 2014.

SILVA, Adilson T. **Visão Geral Sobre o Oracle Data Guard**. Disponível em: <http://4dba.wordpress.com/2008/07/14/visao-geral-sobre-o-oracle-data-guard/ > Acesso em: 04 de dezembro de 2014.

SIQUEIRA, David. **Conhecendo melhor o seu Banco de Dados: Redo Log File**. Disponível em: <http://imasters.com.br/artigo/18105/banco-de-dados/conhecendo-melhor-o-seu-banco-de-dados-redo-log-file/>. Acesso em: 15 de novembro de 2014.

SOARES, Flávio. **Oracle Data Guard 11g com VirtualBox** – Parte 2. Disponível em: http://flaviosoares.com/tag/data-guard/>. Acesso em: 22 de setembro de 2014.

SOUZA, Carlos Roberto Kich; CAMPOS, Paulo Cesar Guimarães. **Alta Disponibilidade.** Disponível em: < http://www.kich.com.br/wp-content/uploads/Monografia-formatada.pdf>. Acesso em 20 de setembro de 2014.

ANEXO A – INSTALAÇÃO DO ORACLE RAC

Para instalação utilizou-se como base o tutorial de Soares (2011).

CONFIGURAÇÃO PARA INSTALAÇÃO DO RAC

Após a instalação do *Oracle Linux* é requisito a instalação dos pacotes:

- binutils-2.17.50.0.6-14.el5
- compat-libstdc++-33-3.2.3-61
- elfutils-libelf-0.137-3.el5
- elfutils-libelf-devel-0.137-3.el5
- gcc-4.1.1-50.el5
- gcc-c++-4.1.2-50.el5
- glibc-2.5-12
- glibc-common-2.5-58
- glibc-devel-2.5-58
- glibc-headers-2.5-58
- libaio-0.3.106-5
- libaio-devel-0.3.106-5
- libgcc-4.1.2-50.el5
- libstdc++-4.1.2
- libstdc++-devel-4.1.2-50.e15
- make-3.81-3.el5
- sysstat-7.0.2-3.el5_5.1
- unixODBC-2.2.11-7.1
- unixODBC-devel-2.2.11-7.1
- libXp-1.0.0-8.1.el5

CONFIGURAÇÃO *HANGCHECK*

Conforme a documentação da *Oracle,* o módulo *hangcheck-timer* é responsável por monitorar travamentos no *kernel* do *Linux*, e caso venha a acontecer algum travamento, o módulo encarrega-se de reiniciar o nó do RAC, evitando danos que possam ser causados no banco. Para verificar se o módulo encontra-se ativo use o comando:

```
# more /etc/modprobe.conf |grep hang
```

Caso o módulo não esteja em execução será necessário carregá-lo manualmente.

```
# echo "options hangcheck-timer hangcheck_tick=1
hangcheck_margin=10 \
hangcheck_reboot=1" >>/etc/modprobe.conf
```

CRIAÇÃO E CONFIGURAÇÃO DOS GRUPOS E USUÁRIO DO *LINUX*

Criam-se dois grupos e um usuário utilizando.

```
# groupaddoinstall
# groupadddba
# useradd –g oinstall–G dba oracle
```

Com o comando`# passwdoracle` o *Linux* necessitará uma senha para o usuário *Oracle*

Deve então alterar os limites no *Shell* do usuário modificando o arquivo `limits.conf`

```
/etc/security/limits.conf
oracle soft nproc 2047
oracle hard nproc 16384
oracle soft nofile 1024
oracle hard nofile 65536
```

Adiciona-se a linha ao arquivo `/etc/pam.d/login`

```
session required pam_limits.so
```

Altere os parâmetros de *Kernel* do *Linux* modificando-se o arquivo `/etc/sysctl.conf` incluindo as linhas:

```
Kernel.sem = 250 32000 100 128
Kernel.shmall = 2097152
```

```
Kernel.shmmni = 4096
Kernel.shmmax = 2147483648
Fs.file-max = 655336
Net.ipv4.ip_local_port_range = 1024 65000
```

CONFIGURAÇÃO DOS HOSTS DOS NÓS

Os arquivos de hosts são utilizados para o *cluster* realizar a comunicação entre os nós, que se dá através da comunicação entre os IP's. Para isso é necessário que se configure o arquivo `/etc./hosts` na configuração:

```
10.0.0.10       rac1            rac1.localdomain
10.0.0.20       rac2            rac2.localdomain
10.0.0.100      rac1-vip        rac1-vip.localdomain
10.0.0.200      rac2-vip        rac2-vip.localdomain
192.168.1.10    rac1-priv       rac1-priv.localdomain
192.168.1.20    rac2-priv        rac2-priv.localdomain
```

CLONAGEM DA MÁQUINA, CRIAÇÃO DOS DISCOS ASM E CONFIGURAÇÃO DE COMUNICAÇÃO ENTRE AS MÁQUINAS.

Numa aplicação real, todos os passos acima devem ser realizados nos nós de cada *cluster*, porém neste exemplo, está sendo utilizado uma máquina virtual. Então, deve-se criar uma nova máquina virtual e apontar-se para uma cópia do disco. Criam-se também três discos que serão compartilhados entre os dois nós, devendo ser similar a Figura 3:

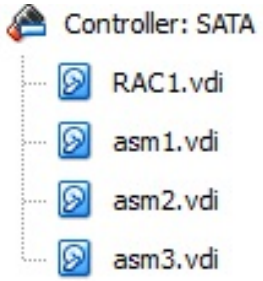

Figura 3- Estrutura de discos para criação do *cluster*

A configuração de comunicação entre os nós será por SSH, com o usuário ORACLE.

Necessita-se gerar as chaves privadas e publica do SSH, para que posteriormente possa realizar a comunicação entre as máquinas.

```
$ cd /home/oracle
$ mkdir .ssh
$ cd .ssh
$ ssh-keygen -t das
$ ssh-keygen -t rsa
```

Esses comandos devem ser realizados em todos os nós do *cluster*.

Das chaves que foram criadas, as chaves públicas devem ser copiadas para o outro nó.

```
$ cat *.pub >>authorized_keys
$ ssh rac2 cat /home/oracle/.ssh/*.pub >>authorized_keys
$ scp -p authorized_keys mvrac2:`pwd`
```

Confirmam-se as sincronizações dos *hosts*.

```
$ ssh rac1 date
$ ssh rac2 date
$ ssh rac1-priv date
$ ssh rac2-priv date
$ ssh rac1.localdomain date
$ ssh rac2.localdomain date
$ ssh rac1-priv.localdomain date
$ ssh rac2-priv.localdomain date
```

Com isso, gera-se o arquivo `known_hosts` que deve ser copiado para o rac2.

```
$ scp known_hosts rac2:`pwd`
```

CRIANDO DISCOS ASM

Com os discos criados do ASM no tamanho de 5GB, realiza-se o particionamento, criando uma partição estendida e três partições lógicas, com o usuário `root` executando os

comandos:

```
fdisk /dev/sdb
Command (m for help): n
Commandaction
extended
primarypartition (1-4)
e
Partition number (1-4): 1
Firstcylinder (1-652, default 1):
Using default value1
Lastcylinderor +sizeor +sizeMor +sizeK (1-652, default 652):
Using default value 652
Command (m for help): n
Commandaction
logical (5 or over)
primarypartition (1-4)
l
Firstcylinder (1-652, default 1): 1
Lastcylinderor +sizeor +sizeMor +sizeK (1-652, default 652):
+256M
Command (m for help): n
Commandaction
logical (5 or over)
primarypartition (1-4)
l
Firstcylinder (33-652, default 33):
Using default value 33
Lastcylinderor +sizeor +sizeMor +sizeK (33-652, default 652):
+256M
Command (m for help): n
      Commandaction
      logical (5 or over)
      primarypartition (1-4)
l
Firstcylinder (65-652, default 65):
Using default value 65
Lastcylinderor +sizeor +sizeMor +sizeK (65-652, default 652):
+256M
Command (m for help): n
Commandaction
      logical (5 or over)
      primarpartition (1-4)
l
Firstcylinder (97-652, default 97):
Using default value 97
Lastcylinderor +sizeor +sizeMor +sizeK (97-652, default 652):
+256M
Command (m for help): n
      Commandaction
```

```
      logica (5 or over)
      primarypartition (1-4)
l
Firstcylinder (129-652, default 129):
Using default value 129
Lastcylinderor +sizeor +sizeMor +sizeK (129-652, default 652):
Using default value 652
      Command (m for help): n
      Commandaction
      logica (5 or over)
      primarypartition (1-4)
l
Firstcylinder (129-652, default 129):
Using default value 129
Lastcylinderor +sizeor +sizeMor +sizeK (129-652, default 652):
      Using default value 652
```

Para verificar se todas as partições estão criadas corretamente utilize o comando:

```
      Command (m for help): p


      Device Boot  StartEndBlocksId   System
      /dev/sdb1     1        652      5237158+5 Extended
      /dev/sdb5     1         32      25697783  Linux
      /dev/sdb6   33         64      257008+83  Linux
      /dev/sdb7    65        96      257008+83  Linux
      /dev/sdb8    97        128     257008+83  Linux
      /dev/sdb9   129        652     4208998+83 Linux
```

Após isso, é necessário ir ao `rac2` e gravar as partições criadas no `rac1`.

```
fdisk /dev/sdb
Command (m for help): w
The partitiontablehasbeenaltered!
Callingioctl() tore-readpartitiontable.
Syncing disks.
```

Feito isso, realiza-se o mesmo particionamento no disco `/dev/sdc` com a mesma estrutura que o `/dev/sdb`, utilizando os mesmos comandos. Já no disco `/dev/sdd` haverá uma partição estendida do tamanho total do disco, dois partições de 256MB e um partição com o restante do espaço.

```
$ fdisk /dev/sdd
Command (m for help): n
Commandaction
extended
primarypartition (1-4)
e
Partitionnumber (1-4): 1
Firstcylinder (1-652, default 1):
Using default value1
Lastcylinderor +sizeor +sizeMor +sizeK (1-652, default 652):
Using default value 652
Command (m for help): n
Commandaction
logical (5 or over)
primarypartition (1-4)
      l
Firstcylinder (1-652, default 1):
Using default value1
Lastcylinderor  +sizeor  +sizeMor  +sizeK  (1-652,  default  652):
+256M
Command (m for help): n
Commandaction
logical (5 or over)
primarypartition (1-4)
l
Firstcylinder (33-652, default 33):
Using default value 33
Lastcylinderor  +sizeor  +sizeMor  +sizeK  (33-652,  default  652):
+256M
      Command (m for help): n
      Commandaction
      logical (5 or over)
primarypartition (1-4)
l
Firstcylinder (65-652, default 65):
Using default value 65
Lastcylinderor +sizeor +sizeMor +sizeK (65-652, default 652):
Using default value 652
```

Para verificar-se se as partições foram criadas de forma correta:

```
Command (m for help): p
Device Boot   Start   EndBlocksId     System
/dev/sdd1        1        652     5237158+5  Extended
/dev/sdd5        1         32     25697783   Linux
/dev/sdd6       33         64     257008+83  Linux
/dev/sdd7       65        652     4723078+83 Linux
Command (m for help): w
The partitiontablehasbeenaltered!
Callingioctl() tore-readpartitiontable.
Syncing disks.
Confirmando o particionamento no rac2
fdisk /dev/sdd
Command (m for help): w
The partitiontablehasbeenaltered!
Callingioctl() tore-readpartitiontable.
Syncing disks.
```

Deve se criar os discos ASM com usuário rootno *Linux.*

```
/etc/init.d/oracleasmcreatedisk ASM1 /dev/sdb9
/etc/init.d/oracleasmcreatedisk ASM2 /dev/sdc9
/etc/init.d/oracleasmcreatedisk ASM3 /dev/sdd7
```

INSTALAÇÃO DO CLUSTER

É necessário criar as variáveis de ambiente do *Oracle* editando-se o arquivo `.bash_profile` e adicionando-se as linhas:

```
      export ORACLE_BASE=/u01/app/oracle
      export ORA_CRS_HOME=${ORACLE_BASE}/product/10.2.0/crs
      export ORACLE_HOME=${ORACLE_BASE}/product/10.2.0/db_1
      export
PATH=${ORA_CRS_HOME}/bin:${ORACLE_HOME}/bin:/usr/kerberos/bin:
/usr/local/bin:/bin:/usr/bin:/usr/sbin:/sbin
```

Iniciando-se a instalação do *cluster* executa-se o comando `./runInstaller` localizado na raiz da pasta de instalação do *cluster* , ele será instalado no caminho correspondente ao *Path* e terá um nome que informa no *Name*, escolhendo o idioma durante a instalação, sendo necessário ainda informar se os nomes que as máquinas respondem na rede pública e privada. É o mesmo nome que esta no arquivo `/etc/hosts` e ainda a placa de rede e qual faixa de *ip* essa placa responde na rede.

Durante a instalação deve-se indicar em qual disco será gravado o OCR e o *Voting Disk*, sendo necessário o compartilhamento desses discos entre os nós do *cluster*. Gera-se um alerta que solicita a execução do arquivo `root.sh` nos dois nós, juntamente com o usuário `root`. Deve-se executar os *scripts* e a instalação do *cluster* estará finalizada.

PRÉ-INSTALAÇÃO DO BANCO DE DADOS

Primeiramente o banco deve ser instalado para depois criar-se a base de dados onde serão armazenadas as informações. Para esta instalação há uma pasta raiz de instalação do Oracle Database 10g e dentro dela existe um arquivo denominado *runInstaller* que deve ser executado com o comando *`./runInstaller`* com o usuário *Oracle*, disponibilizando os tipos de banco de dados que deseja-se criar.

A instalação do banco de dados deve ser direcionada para a pasta que será feita a instalação, onde encontram-se todos os arquivos de instalação da base de dados. Então, o programa identifica que existe *cluster* instalado logo mostra a lista para escolha dos nós que se desejam instalar. Escolha a opção *Install database Software only*, para realizar a instalação do software do banco de dados. Ao final da instalação do *software* é necessário executar o arquivo *root.sh*, que fará a configuração final e sincronização entre os nós do *cluster*.

CRIAÇÃO DO LISTENER

Para que o banco de dados seja conectado por outros programas deve-se configurar o *Listener* com o comando `netca`, abrindo o programa de configuração da *Oracle, no qual deve-se* optar por quais nós o *Listener* atenderá a configuração e a chamados externos, sendo necessário adicionar uma configuração para esse *Listener*.

Com o sistema de configuração pode-se escolher o protocolo de conexão, sendo o mais usual o protocolo TCP, adicionando-o na área de *Selected Protocols*.

Por padrão o *Listener* utiliza a porta 1521. Mantendo-se esse o padrão, finaliza-se a instalação.

CRIAÇÃO DO BANCO DE DADOS

Para a criação do banco é necessário estar conectado no usuário *oracle*, abrindo-se um terminal no *Linux* e executando-se o comando `dbca` e selecionando-se a criação de banco de dados em *cluster.*

Selecionam-se os nós do *cluster* e logo após a opção de propósito geral. Então, é necessário inserir um nome para o banco de dados, alem de marcar a opção de configurar a base de dados com *Enterprise Manager.*

Logo após será definida a senha do banco e selecionada a opção do ASM. Os discos de grupos *storage* devem ser selecionados, sendo necessário especificar o local para a criação do banco de dados, optando pelo *Use Oracle-Managed Files.* É ainda necessário colocar o comando `+ORADATA1` no *Database Area.*

Na tela de recuperação do banco de dados deve-se especificar a área de recuperação para `+ORADATA2` usando o tamanho de 2048MB. Habilita-se o arquivamento e edita-se o parâmetro para o formato `%t_%s_%r.dbf`.

Na criação do serviço desmarca-se a opção *Sample Schema*s e seleciona-se a opção `Typical – Allocatememory as a percentage of the total phycialmemory (500 MB)` e em *`Percentage`* deve ser completado com o número 44. Assim, o *Database* está pronto para ser criado.

ANEXO B - INSTALAÇÃO DO *DATA GUARD*

A Instalação do *Data Guard* foi feita de acordo com Soares (2011).

Para criação do *data Guard* é necessário que os bancos de dados tenham as mesmas configurações, neste exemplo de teste teremos o `dbtst` que será o banco de produção e o `sbsty` que será o banco de replicação.

Com os bancos exatamente configurados, é necessário habilitar-se a base de dados para forçar os registros conectados no banco, digitando-se o comando:

```
ALTER DATABASE FORCE LOGGING
```

Configura-se então o arquivo `redo log` para que tenha o mesmo tamanho criando-se o `redo log` online:

```
ALTER DATABASE ADD STANDBY LOGFILE GROUP 10 SIZE 100M
ALTER DATABASE ADD STANDBY LOGFILE GROUP 11 SIZE 100M
ALTER DATABASE ADD STANDBY LOGFILE GROUP 12 SIZE 100M
```

Nesse ponto, após a criação dos *Standby logs*, deve-se alterar arquivo de parâmetro do banco primário:

```
DB_NAME=dbtst
DB_UNIQUE_NAME=dbstby
LOG_ARCHIVE_CONFIG='DG_CONFIG=(dbtst, dbstdy)'
LOG_ARCHIVE_DEST_1=
'LOCATION=/home/archivelog/dbtst/
VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
DB_UNIQUE_NAME=dbtst
LOG_ARCHIVE_DEST_2=
'SERVICE=dbstby LGWR ASYNC
VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE|
DB_UNIQUE_NAME=dbstby
LOG_ARCHIVE_DEST_STATE_1=ENABLE
LOG_ARCHIVE_DEST_STATE_2=ENABL
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
LOG_ARCHIVE_FORMAT=%t_%s_%r.arc
LOG_ARCHIVE_MAX_PROCESSES=30
FAL_SERVER=dbtst
FAL_CLIENT=dbstby
DB_FILE_NAME_CONVERT='/home/oracle/dbstby/oradata','/home
```

```
/oracle/dbtst/oradata'
      LOG_FILE_NAME_CONVERT=
      '/home/oracle/dbstby/oradata','/home/oracle/dbtst/oradata
'
      STANDBY_FILE_MANAGEMENT=AUTO
```

Ativa-se o arquivamento no banco *Primary*:

```
      SHUTDOWN IMMEDIATE;
      STARTUP MOUNT;
      ALTER DATABASE ARCHIVELOG;
      ALTER DATABASE OPEN;
```

Faça um *backup* frio do banco primário:

```
      SHUTDOWN IMMEDIATE;
      $ cp -r /home/oracle/dbtst/oradata/*.*
/home/oracle/dbstby/oradata
```

Inicia-se então o banco primário no modo `MOUNT` para criarmos o *standby controlfile*:

```
      STARTUP MOUNT;
      ALTER DATABASE CREATE STANDBY CONTROLFILE AS
'/home/oracle/dbstby/standy.ctl';
      ALTER DATABASE OPEN;
```

Cria-se então o arquivo de parâmetro para o banco *Standy* conforme o *Primary*:

```
      CREATE PFILE='?/dbs/intistandy.ora' from spfile;
```

Para alterar o arquivo de parâmetro do banco *Standy* utiliza-se:

```
      DB_NAME=dbtst
      DB_UNIQUE_NAME=dbstby
      LOG_ARCHIVE_CONFIG='DG_CONFIG=(dbtst, dbstby)'
      DB_FILE_NAME_CONVERT='/home/oracle/dbtst/oradata','/home/
oracle/dbstby/oradata'
      LOG_FILE_NAME_CONVERT=
      '/home/oracle/dbtst/oradata','/home/oracle/dbstby/oradata
'
      LOG_ARCHIVE_FORMAT=log%t_%s_%r.arc
```

```
LOG_ARCHIVE_DEST_1= 'LOCATION=/home/oracle/dbstby/arc/
VALID_FOR=(ALL_LOGFILES,ALL_ROLES)
DB_UNIQUE_NAME=dbstby
LOG_ARCHIVE_DEST_2=
'SERVICE=primary LGWR ASYNC
VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE)
DB_UNIQUE_NAME=dbtst'
LOG_ARCHIVE_DEST_STATE_1=ENABLE
LOG_ARCHIVE_DEST_STATE_2=ENABLE
REMOTE_LOGIN_PASSWORDFILE=EXCLUSIVE
STANDBY_FILE_MANAGEMENT=AUTO
FAL_SERVER=dbtst
FAL_CLIENT=dbstby
```

Configura-se o `tnsnames` e `os listeners`.

Nesse caso foi colocado um `listener` para cada banco, um deles na `porta1521` e o outro na `1522`.

```
Listener
LISTENER =
(DESCRIPTION_LIST =
(DESCRIPTION =
(ADDRESS_LIST =
(ADDRESS = (PROTOCOL = IPC)(KEY = EXTPROC))
)
(ADDRESS_LIST =


(ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.10) (PORT =
1521))
 )
 )
)

dbstby =
(DESCRIPTION =
(ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.10)(PORT =
1522))
(CONNECT_DATA =
(SERVER = DEDICATED)
(SERVICE_NAME = standy)
)
 )

INBOUND_CONNECT_TIMEOUT_listener=0

SID_LIST_LISTENER =
(SID_LIST =
(SID_DESC =
(GLOBAL_DBNAME = dbtst)
```

```
      (ORACLE_HOME = /home/oracle/10gR2)
      (SID_NAME = primary)
      )
      )

      SID_LIST_STANDY =
      (SID_LIST =
      (SID_DESC =
      (GLOBAL_DBNAME = dbstby)
      (ORACLE_HOME = /home/oracle/10gR2)
      (SID_NAME = dbstby)




      dbtst =
      (DESCRIPTION =
      (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.10)(PORT =
1521))
      )
      (CONNECT_DATA =
      (SERVICE_NAME = dbtst)
      (SERVER = DEDICATED)
      )
      )

      dbstby =
      (DESCRIPTION =
      (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.10)(PORT =
1522))
       )
      (CONNECT_DATA =
      (SERVICE_NAME = dbstby)
      (SERVER = DEDICATED)
      )
      )
```

Inicia-se os `listeners`:

```
      $ lsnrctl start|
      $ lsnrctl start standy
```

Neste ponto, inicia-se o banco`standy`:

```
      SQL> STARTUP MOUNT;
```

Coloca-se então, o banco secundário para receber os *archivelogs* do banco primário.

Para isso é necessário que se se inicie o *Redo Apply* no banco secundário e depois conclua a instalação:

```
      Alter Database Recover Managed Standby Database
Disconnect From Session;
```