Comparação entre Servidores de Banco de Dados Tradicionais e Sistemas Dedicados: Um Estudo de Caso sobre a Aplicação do Oracle Exadata Software

Comparison between Traditional Database Servers and Dedicated Systems: A Case Study on the Application of Oracle Exadata Software

Jean Nascimento [1]

Danilo Ricardo Barbosa de Araújo [1]

[1] Escola Politécnica de Pernambuco, Universidade de Pernambuco, Recife, Brasil

E-mail do autor principal: Jean Nascimento jean.rnascimento@gmail.com

Resumo

A busca por ambientes de alto desempenho foi um dos motivadores da área da implementação de softwares específicos para extrair o máximo de desempenho do hardware, e o termo appliance surgiu para descrever essa integração. Quando existe a integ ração do hardware com o software são minimizados problemas como compatibilidade e indisponibilidade além de serem maximizados itens como as possibilidades de acesso a r ecursos do hardware. Este trabalho apresenta o funcio namento do software de banco de dados desenvolvido para um hardware específico, o Ora cle Exadata Software. Foi realizado um estudo de caso comparando o desempenho de consultas SQL no ambiente tradicional de banco de dados e no ambiente com Exadata Machine, o que possibilita entender que as funcionalidades disponíveis no Exadata Software trazem um ganho tanto de desempenho quanto de escalabilidade. O trabalho conclui que para um banco de dados Oracle existe um ganho real de 45% de redução do tempo de espera do resultado de uma c onsulta no ambiente Exadata comparado ao tradicional.

Palavras-Chave: *Exatada; Volumetria; Banco de Dados Relacional; SQL; Tuning; IOPS.*

Abstract

The search for high performance environments was one of the motivators in the area of implementation of specific software to extract the maximum performance of the hardware, and the term appliance came to describe this integration. When hardware integration with the software exists, problems such as compatibility and unavailability are minimized in addition to maximizing items such as the possibilities of accessing hardware resources. This work presents the operation of database software developed for a specific hardware, the Oracle Exadata Software. A case study comparing SQL query performance in the traditional database environment and the Exadata Machine environment, which makes it possible to understand that the features available in Exadata Software bring both performance and scalability gain. The work concludes that for an Oracle database there is a real gain of 45% reduction of the waiting time of the result of a query in the Exadata environment compared to the traditional one.

http://dx.doi.org/10.25286/repa.v2i2.552

Key-words: *Paper; Engineering; Template; NBR 14724 ABNT.*

1 Introdução

Atualmente se vive a era da informação e nela o valor associado ao dado muitas vezes supera os valores de bens materiais, tanto no mundo corporativo quanto no meio social. A forma como os dados são manipulados têm ganhado cada vez mais relevância, e a velocidade com que são gerados e armazenados se torna um quesito de grande importância. Conforme previsão da IDC Brasil, o volume de dados criado no mundo crescerá anualmente 44 vezes até 2020 e mais de um terço de toda essa massa de informação será armazenada em servidores nas nuvens ou passará por eles. No Brasil, o volume de dados deve ir de 212 bilhões de gigabytes em 2013 para 1.600 bilhões de gigabytes em 2020. (IDC BRASIL, 2013).

Com foco nessa necessidade, diversas empresas que possuem soluções para servidores de banco de dados passaram a oferecer alternativas baseadas em *appliance*, dentre as quais podemos destacar a *Oracle* (WEISS, 2009), IBM (SINGH, 2011) e *Teradata Technology* (TERADATA.COM, 2016).

A Oracle em 2008 deu início ao projeto *Exadata* inicialmente em parceria com a HP e posteriormente com a *Sun Microsystems.* Nessa parceria a HP e a *Sun Microsystems* produziram o *hardware* e a *Oracle* produziu o *software*, surgindo dessa união o *appliance* que permitiria um grande poder computacional. Nas primeiras versões, o projeto focou apenas em soluções para *Data Warehouse*, mas logo foi observado que poderia também tratar os dados transacionais das operações do dia a dia das organizações. A versão oferece alto desempenho tanto para dados analiticos quando para transacionais (OSBORNE, 2015).

Dentro do mercado de soluções *appliance* como alternativas a solução da *Oracle*, o *IBM PureData System for Analytics* que hoje é projetado e desenvolvido pela IBM, foi originalmente construído pela *Netezza* e adquirido pela IBM (SINGH, 2011). Esta solução basicamente é constituída de três camadas: as unidades de armazenamento, os servidores *S-Blades* e os servidores de aplicação. Outra solução disponível no mercado é o *Teradata Extreme Data Appliance* desenvolvido pela *Teradata Technology*, mas focada no *hardware* e disponibiliza um grande conjunto de plataformas *hardware* para permitir projetar o *software* em conjunto (TERADATA.COM, 2016).

A proposta deste trabalho tem como objetivo complementar as referências bibliográficas em português sobre o *Oracle Exadata* e em seu funcionamento do *Oracle Exatada Database Machine* mais especificamente o *Exadata software*. Além disso, este artigo oferece uma comparação entre ambientes tradicionais e o *Exadata*, demonstrada por meio de um estudo de caso do sistema de classificação e tributação de preço do *mix* de produtos de uma unidade varejista. O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta um breve histórico sobre os trabalhos que estão relacionados a servidores de banco de dados OLAP e OLTP, comparativos de banco de dados relacional e *data warehouse*. A Seção 3 engloba o referencial teórico e elenca as técnicas e tecnologias geralmente utilizadas pela administração de banco de dados. A Seção 4 explica a metodologia utilizada no desenvolvimento deste trabalho. A Seção 5 explica em detalhes o *Oracle Exatada Database Machine*, mais especificamente a versão x5-2, arquitetura, componentes, funcionalidades e aborda de forma detalhada o *Oracle Exadata Software*, demonstrando sua arquitetura e suas funcionalidades. A Seção 6 apresenta os resultados relacionados com a comparação entre o *Oracle Exatada Database Machine* e um servidor de banco de dados não *appliance* com um estudo de caso simulado de manipulação de grande volume de dados. Por fim, na Seção 7 apresenta as considerações finais sobre todo o trabalho.

2 Metodologia

A metodologia empregada neste trabalho pode ser resumida por meio de cinco macro etapas:

1. Revisão bibliográfica usando literatura especializada no tema;
2. Estudo da migração do ambiente tradicional para o *Exadata*;
3. Criação de um exemplo simulando a operação com grande carga de dados;
4. Criação de um ambiente simulado para colocar a prova os conceitos e definições;
5. Consolidação dos resultados.

As etapas são descritas adiante com maior riqueza de detalhes.

Na primeira etapa, foi realizada a revisão da literatura técnica e científica a qual consistiu a buscas por trabalhos que abordassem os conceitos, tecnologias e trabalhos relacionados. A partir dela foi possível criar uma base de referências para um melhor refinamento da proposta. A revisão da literatura ficou voltada para trabalhos referentes à proposta de detalhamento do ambiente *Exadata* e a

comparação com o ambiente de banco de dados tradicional. Os motores de busca utilizados foram o *Google* e *Google* Acadêmico através das palavras-chaves: *Exatada*, volumetria, Banco de dados Relacional, SQL *Tuning* e IOPS.

Na segunda etapa, foi feito um estudo sobre o ambiente Exatada, incluindo um acompanhamento e documentação da migração do ambiente com servidores de banco de dados tradicionais para uma nova estrutura com *Exadata Database Machine* em uma corporação internacional de hipermercados. O processo foi realizado com um ambiente de QA (*Quality Assurance*) que sustentou a operação de teste e simulação da produção durante 5 semanas, com 2 profissionais especialistas em ambiente *Exadata*, 2 DBA Oracle certificados, 2 Analistas de sistema, 1 Analista de integração, 1 Analista de infraestrutura e 1 gerente de projeto.

Na terceira etapa, foi criado um exemplo simulando a operação com grande carga de dados, o exemplo demonstra a operação de classificação e carga tributária para cálculo do preço de venda dos produtos de um hipermercado. O volume de dados gerado foi da ordem de 20 milhões de registros resultado do plano cartesiano de um *mix* de produtos de 50 mil itens vezes 5 distribuidores/fornecedores por item vezes 27 UFs (Unidade da Federação) vezes 3 CFOPs (Código Fiscal de Operações e Prestações). Com este estudo de caso, é demonstrado o comportamento das consultas SQL (Structured Query Language) em uma massa de dados de alta volumetria no ambiente tradicional e no ambiente *Exadata*.

Na quarta etapa, foi iniciada a criação de um ambiente simulado para prova de conceito do exemplo simulado. Devido a limitações de acesso aos recursos de *hardware* necessários para estudo do ambiente *Exadata*, não foi possível realizar o comparativo do seu *hardware* com produtos concorrentes. Adicionando-se a esse fato, uma restrição em seu contrato de uso, que descreve em cláusulas do EULA (*End-User License Agreement*) ser proibido a comparação do *hardware* por terceiros. O artigo foca na comparação do *Exadata Software*com o ambiente de banco de dados não *appliance* executando um sistema varejista .A última etapa consistiu na consolidação dos resultados e elaboração deste artigo contendo os resultados finais da pesquisa.

3 Trabalhos Relacionados

Nesta seção serão abordados alguns estudos encontrados durante a elaboração destes artigo. São análises relativas ao desempenho de diversos sistemas de bancos de dados relacional, análises relativas a sistemas de armazenamento de dados e análises das linguagens SQL e NoSQL.

Jonathan da Silva e colaboradores (DA SILVA, 2016) realizaram um estudo abordando a implementação de um projeto de BI comparando os banco de dados MySQL e Postgres com o auxílio da ferramenta de ETL Pentaho Data Integration da suíte Pentaho Community. Com o trabalho foi possível dizer que o *Postgres* se mostrou melhor do que o *MySQL* como opção para uso como *data warehouse* em um projeto de BI, sendo o *Postgres* o que utiliza menor quantidade de recursos e se mostra mais eficiente.

Ronald Weiss em nome da Oracle Corporate (WEISS, 2012) apresentou os pontos técnicos relacionados a versão da Exadata Database Machine X2-2 e X2-8. Foram abordados na apresentação detalhes como funcionamento, arquitetura, componentes e modelos de escalabilidade.

Kai Yu e colaboradores (YO, 2015) propõem uma nova arquitetura para um sistema de banco de dados *appliance* composto pelo software da Oracle integrado com o hardware da Dell para aceleração de bancos de dados. O estudo aponta que o ambiente da Oracle de fato é líder no mercado, mas pode ter o seu desempenho melhorado se novos investimentos em hardware forem executados.

4 Referencial Teórico

4.1 Princípios de Banco de Dados

Banco de dados é um sistema de manipulação de registro com o propósito geral de manter os dados e torná-lo disponível sempre que necessário. O banco de dados normalmente armazena os dados relacionados a um sistema de computador (FOSTER, 2014).

Em um banco de dados relacional uma série de tabelas de registros com Atributos são ligadas entre si por uma ou mais relações. Essas relações são criadas usando chaves estrangeiras que são atributos que contém os mesmos dados em ambas as tabelas (CROWTHER, 2013).

A maior parte da programação de hoje é feita em uma linguagem orientada a objetos que introduz um ambiente rico onde os dados, os procedimentos e as

http://dx.doi.org/10.25286/repa.v2i2.552

funções de manipulação são armazenadas juntas. Um banco de dados orientados a objetos é visto como um único objeto persistente no qual operações podem ser realizadas (CROWTHER, 2013).

Uma característica dos bancos de dados que utilizam o SQL como linguagem é a forma de organizar os dados, todos são bancos de dados relacionais, e isso quer dizer que os dados estão organizados em linhas e colunas de tabelas relacionadas entre si.

Um sistema de gerenciamento de banco de dados (SGBD) é um conjunto de programas que permitem a gestão de um banco de dados. Das funções mais importantes de um SGBD se destacam as seguintes (FOSTER, 2014):

- Definição de dados (tabelas, dependências, de integridade, visões);
- Manipulação de dados (adicionar, atualizar, deletar, recuperar, re-organizar e agregar
- Verificações de segurança e integridade de dados;
- Apoio a linguagens de programação.

O modelo relacional de banco de dados é fundamentado em princípios matemáticos, principalmente da álgebra relacional. Representado por uma coleção de tabelas (entidade) e um conjunto de linhas (tuplas), uma lista de valores de atributos. Algebra relacional consiste em um conjunto de operações sobre as relações, onde cada operação produz uma nova relação a partir de um ou mais relações já existente (COUGO, 2011). Há oito operações básicas de álgebra relacional, são elas: união, diferença, restrição, produto, projeção, *join*, cruzamento e divisão (FOSTER, 2014).

4.1.1 SQL

Para a manipulação de dados em um banco de dados foi criado o padrão de linguagem bastante difundido no mercado conhecido pela sigla SQL (*Structured Query Language*) que possibilita aos usuários operações de inclusão, consulta, alteração e deleção de dados. O SQL é a linguagem utilizada em diversos produtos do mercado tais como banco de dados Oracle, SQL Server, e o Postgres (FOSTER, 2014).

Além da estrutura para armazenar os dados existem também outros objetos em um banco de dados. Por exemplo, alocação de memória para execução de tarefas, armazenamento em arquivos, *trace* das transações e, por fim, os objetos de transação como índices, procedimento, *triggers*, *sequences* e os dados transacionais em si (PETERSEN, 2003).

Em 1998 foi utilizado o termo NoSQL para um banco de dados relacional que omitiu o uso da linguagem SQL. Posteriormente em 2009 foi utilizado novamente em uma conferência em São Francisco por defensores de um modelo não relacional (COSTA, 2016).

4.2 OLTP e OLAP

O trabalho do banco de dados de forma geral se concentra em armazenamento e consulta de dados (COUGO, 2011). Com a evolução das aplicações e o grau de importância que a informação ganhou dentro das organizações, o banco de dados passou a ter não só a necessidade de armazenar e consultar, mas também de ser capaz de transformar a informação de modo que permita o uso de dados históricos para ser utilizado como base na tomada de decisões estratégicas.

Nesse contexto, foram difundidas duas formas de processar a informação: OLTP (*On-line Transaction Processing*) e OLAP (*On-line Analytical Processing*).

O formato OLTP voltado para a operação do dia a dia das organizações, são os processos que dão sustentação a operação, tarefas rotineiras, geradas a partir de sistemas computacionais utilizados para guiar a produção do produto fim da organização. Nesse formato as informações são voláteis, não existem históricos de dados, é a tarefa que está sendo executada agora, e as consultas SQLs não são focadas em alta performance pois o volume de dados é baixo (KOROTKEVITCH, 2015).

O formato OLAP é voltado para a análise analitica dos dados, diferente da operação com dados transacionais, trata da capacidade de analisar grandes volumes de informações nas mais diversas perspectivas dentro de um *Data Warehouse*. Nesse formato, a informação não é volátil, existe um denso histórico de dados e as consultas SQLs são focadas em desempenho extremo, voltada a auxiliar as organizações na tomada de decisão (COUGO, 2011).

Mesmo se tratando de formas diferentes de processar os dados, OLTP e OLAP não são excludentes, são formas complementares de processar os dados. Todas as organizações precisam do OLPT para manter-se funcionando nas operações do dia a dia e também precisam do OLAP para estudar o histórico da organização e do mercado e assim ser

capaz de tomar decisões fundamentadas em estatística e comportamentos conhecidos.

4.3 IOPS

IOPS (*Input/Output per Second*) é a capacidade de ler ou escrever no disco por segundo e isso impacta diretamente no desempenho da aplicação que por consequência no tempo de resposta de uma consulta SQL. Uma das formas de analisar o tempo de resposta de consultas SQLs é verificar a quantidade de leituras e escrita que são realizadas no *Storage* para se obter o resultado de uma consulta realizada, mesmo considerando que a percepção do usuário ou fatores externos podem interferir para determinar se um tempo de resposta é ou não aceitável. Com isso, é possível usar o IOPS da unidade de armazenamento, seja ela um HDD (*Hard Disk Drive*) ou uma memória *Flash* como um SSD (*Solid-State Drive*) para entender e melhorar os resultado de uma consulta SQL (SINCIC, 2012).

Para calcular o IOPS de um disco é necessário obter os valores de latência rotacional (tempo que o disco leva para girar e posicionar-se no ponto de leitura) e a latência de procura (tempo que a cabeça de leitura leva até o ponto a ser lido), e uma vez com esses dados basta aplicar a fórmula de 1000 sobre a latência rotacional, somada a latência de procura (SINCIC, 2012).

O disco HDD padrão de 7200 RPMs chega a uma taxa de 75 IOPS enquanto um SSD padrão atinge 1500 IOPS. Para compensar essa diferença na taxa de IOPS é possível utilizar algumas técnicas de RAID para aumentar a taxa de IOPS dos HDD, como por exemplo, utilizando o RAID 10 com 6 discos de 7200 RPMs, é possível chegar a uma taxa de 294 IOPS; ou utilizando um RAID 5 com 4 discos de 15000 RPMs, é possível alcançar a taxa de 274 IOPS. Para chegar a um denominador comum entre usar o HDD ou SSD é necessário avaliar quanto de IOPS será necessário e verificar se a relação custo-benefício entre SSD e HDD é vantajosa considerando a taxa IOPS e o custo financeiro para aquisição dos equipamentos (SINCIC, 2012).

4.4 Appliance Computer

O termo *Appliance Computer* é usado para indicar que uma solução de *software* foi construída especificamente para um determinado *hardware*. Talvez o caso mais conhecido da aplicação desse termo seja o *software* iOS construído pela *Apple* para executar inicialmente no iPhone e que hoje tem uma variação para o iPad e outra para o iPod Touch.

No mercado de soluções para banco de dados, a *Oracle* em 2008 construiu o *Software Exadata* para executar sobre a solução da HP e nessa parceria lançaram o *HP Oracle Database Machine*, com recursos de servidor de banco de dados que apenas podiam ser obtidos pela solução Oracle/HP (OSBORNE, 2015).

5 Arquitetura e Detalhamento Técnico do Oracle Exadata

O *Oracle Exadata Software* foi desenvolvido pela *Oracle* em 2008 para executar em servidores de banco de dados produzidos pela HP com o objetivo de dar ao banco de dados um desempenho superior. O *software* foi baseado no SAGE (*Storage Appliance for Grid Environments*) e voltado para banco de dados *Data Warehouse*.

Com a aquisição da empresa SUN pela Oracle em 2009, a empresa que era proprietária tanto do *software* quanto *hardware*, lançou a segunda versão do *Exadata Database Machine* desta vez totalmente produzido pela Oracle. A versão mais recente da *Exadata Machine* pode ser encontrada em 6 modelos, partindo dos modelos base X6-2 e X6-8, nas variações *quarter-rack, half-rack e full-rack*. Como introdução à arquitetura e funcionalidades do *Exadata Software* será feita uma breve descrição do *hardware* que acompanha a versão X5-2 do *Exadata* (OSBORNE, 2015).

5.1 Hardware

Avaliando o hardware da *Exadata Machine,* não há grandes diferenças para os produtos concorrentes. Observa-se que em quesitos de hardware existem opções no mercado superiores ao *Exadata Machine* (WEISS, 2012). O *hardware* do *Exadata Machine* pode ser dividido em três grandes partes: o servidores de banco de dados, o *Storage Server* e a *Infiniband,* como pode ser visto na Figura 1.

Na versão *Oracle Exadata* X5-2 o servidor de banco de dados conta com dois processadores Intel Xeon E5 2600 v3. Cada processador Intel Xeon é composto por até 18 núcleos que na versão *full rack* entrega 288 núcleos, com uma frequência de até 2.6 GHz e tem 45 MB L3 *cache*. Um conjunto de 24 *slots* DIMM (*dual inline memory module*), onde cada *slots* permite o uso

http://dx.doi.org/10.25286/repa.v2i2.552

de um placa de memória de até 32 GB DDR4 com frequência de 2133Hz, totalizando 768 GB de memória, quando todos os slots estão sendo utilizados. A largura de banda alcança os 2133MT/sec por canal, conforme pode ser observado na Figura 2.

Figura 1: Divisão do Exadata Machine.

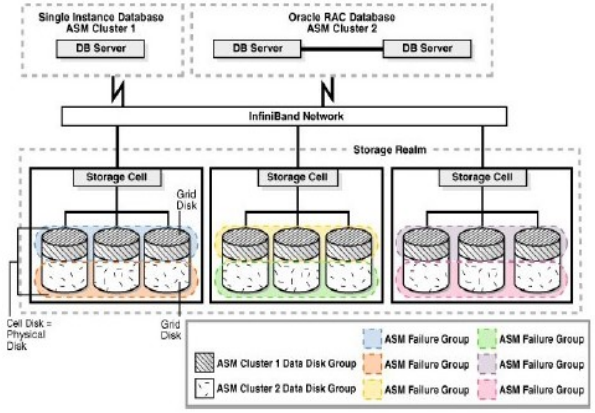

Figura 2: Arquitetura do Oracle Exadata Machine Database.

O *Storage Server* do *Oracle Exadata* X5-2 traz consigo 4 discos SAS (*Serial Attached SCSI*) de alto desempenho de 600 GB com 10.000 RPM cada, totalizando 2.3 TB e uma controladora de disco HBA com 1 GB *supercap-backed write cache*. Como o *Oracle Exadata* vem se tornando cada vez mais um banco de dados em memória, além dos discos SAS, o equipamento possui mais 2 placas F160 produzidas pela parceria *Oracle Intel* conectadas no barramento PCIe de 470 mil IOPS de *read* e 170 mil IOPS de *write*, ligadas a 4 placas de *Flashdisk* produzidas pela Intel que somadas entrega aproximadamente 180 TB de armazenamento. Um esquema técnico sobre esta arquitetura pode ser visto na Figura 3 (ORACLE, 2016).

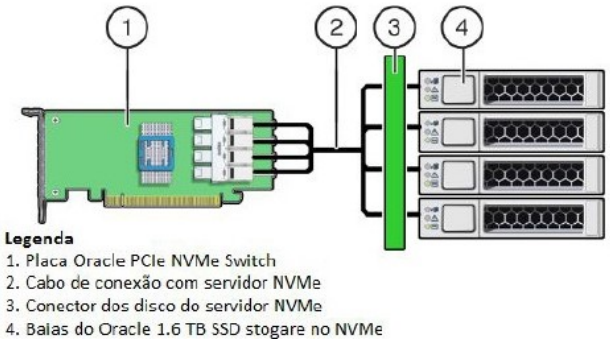

Figura 3: Placa de conexão Infiniband.

5.2 Exadata Software

O *Exadata Software* executa no S.O. Linux instalado no *Storage Server* da *Exadata Machine* de forma isolada, ou seja, cada *Storage Server* executa uma instância do *Exadata Software* sem saber que existe as demais unidades do *Storage Server*. A arquitetura do *Exadata Software* é dividida em processos: *Management Server, Restart Server e Cell Server.* O *Management Server* (MS) responde pelo gerenciamento do *hardware* que é disponibilizado para o *Exadata Software*. Uma analogia poderia ser feita como sendo um " *Driver Device*" criando uma interface do *software* com o *hardware*. Já o *Restart Server* (RS) responde pela monitoração dos processos e em casos de travamento é responsável por re-inicializá los. Por fim a *Cell Server* trata da parte principal do *Exadata Software*, e responde pelo gerenciamento de discos, comunicação e recursos com *offload* de SQL (OSBORNE,2015).

O *Cell Server* é a representação de um *Storage Server* e contém dados como nome do *storage* e endereço ip de acesso. Para isso, uma *cell* é composta de um *physicaldisk*, que por sua vez é composto por uma LUN (*Logical Unit Number*) que representa de forma lógica um *physicaldisk*. Em cada LUN existe o *celldisk* que pode ser acessado e gerenciado pela função *cellcli*. Por fim, o *celldisk* é composto pelos "n" *griddisk* que são agrupados nos *diskgroups* já no lado do servidor de banco de dados no ASM (OSBORNE,2015).

Existem por padrão 3 griddisk: DATA, RECO e DBFS. O DATA, como o próprio nome diz, guarda os dados. RECO é responsável pela área de recuperação ou *redo.* DBFS é responsável pela área dos arquivos de sistema. O *Exadata Software* distribui essas áreas de forma estratégica no disco, ficando a RECO na

parte central do disco, a DBFS na área interna e a DATA na parte externa. Isso é feito para aumentar a quantidade de IOPS uma vez que a cabeça de leitura fará movimentos menores para chegar ao dado buscado, conforme Figura 4 (ORACLE, 2016).

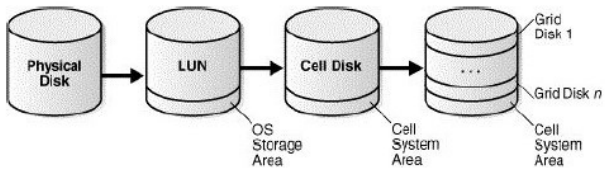

Figura 4: Sistema de discos do Storage Server.

Outra parte importante no *Exadata Software* é o protocolo iDB, que é responsável pela comunicação entre o servidor de banco de dados e o *Storage Server* através das *Infinibands*. O protocolo iDB desenvolvido pela *Oracle* é baseado no RDS (*Reliable Datagram Sockets* v3) que possibilita baixo *overhead* e latência utilizando o ZDP (*Zero-loss Zero-copy Datagram Protocol*) do *Infiniband* com o DMA (*direct memory access*). Isso torna o *Exadata Software* capaz de ler diretamente a memória do servidor de banco de dados e gravar na memória do *Storage Server* sem passar pelo processador (ORACLE, 2016). Para finalizar, o *Exadata software* dispõe das seguintes funcionalidades (OSBORNE,2015):

- Descentralização e escalabilidade: os dados não precisam mais ir para o servidor de banco de dados e voltar para o *Storage Server*. O banco de dados (*GridDisk*) pode informar à *Storage Server* destino qual a *Storage Server* origem e então realizar a leitura diretamente, o que diminui consideravelmente o consumo de rede Infiniband.
- *Smart Flash*: garante que os dados mais acessados ficarão tanto no disco quando na memória *flash* e em caso de atualização de dados, o valor é enviado para os dois ao mesmo tempo e o primeiro que for sucesso já informa ao servidor de banco de dados a operação concluída.
- *Resource Manager*: permite modificar em tempo real os recursos associados à configuração do banco de dados sem reiniciar o servidor.
- *Cell Offload*: o grande diferencial do *Exadata Software* está na função *cell offload*, pois nela o *software* é capaz de verificar na consulta SQL recebida no *storage*, quais colunas e condições são requisitados. Baseado nessas informações ao invés de retornar todo um bloco de dados referente à solicitação requisita e retorna apenas as linhas que respeitam a condição enviada. Este recurso fornece um ganho de 85% de redução do I/O entre o servidor de banco de dados e o *Storage Server* (WEISS, 2012).
- HCC: *Hybric Columnar Compression* é uma tecnologia híbrida da forma de compressão dos dados em colunas e linhas ao invés de apenas linhas, onde são agrupados os valores das colunas de um determinado grupo de linhas e compactados, isso baseado no resultado da análise do *Storage Server*.
- *Storage Index*: Os dados são indexados no próprio *storage* em pacotes de 1MB, de modo que são gravados os mínimos e máximos valores por coluna.

6 Estudo de Caso

Após descrever os recursos que o *Exadata Software* disponibiliza para os DBAs, será demonstrado um estudo de caso para se entender na prática como são aplicados os conceitos do *Exadata Software*.

O ambiente tradicional utilizado para comparação com o ambiente *Exadata* é composto por 3 servidores *Windows Server* 2008, com processadores *Intel Xeon* de 2.67GHz cada, onde individualmente cada processador possui 8 núcleos. Somados os 3 servidores dispõem de 16GB de memória RAM e 6TB para armazenamento dos dados, distribuídos em 3 discos HDD de 2TB com 7200RPM cada. Os servidores são conectados entre si e entre as estações através de redes *ethernet* 10/100.

Primeiramente, foi criado um cenário hipotético, relacionado com a tarefa de precificação dos produtos de um hipermercado. Neste cenário é necessário primeiro classificar o produto tributariamente através de uma NCM (Nomenclatura Comum do MERCOSUL), registrar um grupo de fornecedores / distribuidores que trabalha com o produto, e por fim indicar em que UFs o determinado produto será vendido. Nesse contexto são criadas 5 tabelas: NCM, Produto, Fornecedor, Tributação e Lojas. Estas tabelas e os seus relacionamentos estão ilustrados na Figura 5.

Com relação à volumetria das tabelas, os valores são os seguintes: a tabela NCM possui aproximadamente 8 mil registros, a tabela produto

http://dx.doi.org/10.25286/repa.v2i2.552

possui 50 mil registros, a tabela fornecedor possui 3 mil registros, a tabela lojas possui 190 registros e a tabela tributação, que é o resultado de produto x UF, possui 1,3 milhões de registros.

Executando uma consulta SQL para saber quais produtos são atendidos por fornecedores de regime tributário simples nacional e que são tributados de ICMS ST na UF de Pernambuco em um servidor de banco de dados tradicional seria realizado o *join* entre as tabelas Produto, Fornecedor e Tributação. Um *full table scan* seria aplicado na tabela de produto e na de fornecedor, pois não existe índice no atributo tipo de regime tributário, retornando 53 mil registros resultado da soma dos 50 mil registros da tabela de produtos mas os 3 mil registros da tabela de fornecedor, em 2 GB de blocos de dados e um tempo médio de 831 milissegundos para a tabela de produto, e 320 milissegundos para a tabela de fornecedor. Por fim, a tabela de tributação retorna um *hash-join* resultado produto do cartesiano de produtos x fornecedor. Esta última operação retorna 187 milhões registros com 15 GB de blocos de dados em um tempo médio de 1381 milissegundos.

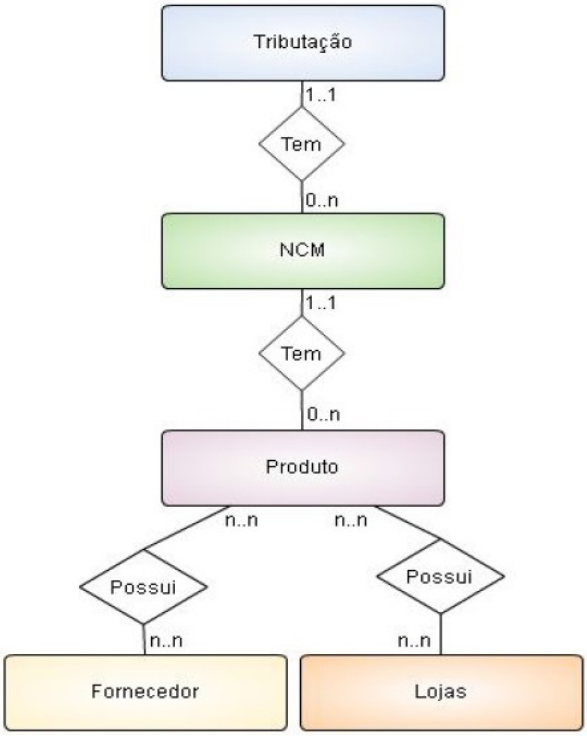

Figura 5: MER do cenário hipotético de precificação dos produtos.

Os 17 GB de bloco de dados são enviados do *storage server* para o servidor de banco de dados pela rede 10/100/1000 a 1 GB/sec, o que resulta em 17000 milissegundos para entregar o bloco de dados. O ASM no servidor de banco de dados aplicará as condições *where* para o fornecedor e a tributação dentro do resultado de uma consulta no tempo médio de 759 milissegundos, recuperando apenas os fornecedores de regime tributário simples nacional e as tributações com valor de ICMS ST, descartando os demais blocos de dados. A operação completa trafega 17 GB de dados e ocorre em 19532 milissegundos.

Essa mesma consulta acontecendo no *Exadata Software* seria interpretada dentro do *storage server* pelo *smart flash*, por meio do mapeamento do *storage index* e passando para o *cell offload*, que irá separar do bloco de 17 GB de dados apenas o que é solicitado. As operações de *join* entre as tabelas produto, fornecedor e tributação, o *full table scan* das tabelas de produto e fornecedor, além do *hash-join* da tabela de tributação resultado do cartesiano produtos x fornecedor tem um tempo médio de 1875 milissegundos de execução, o que é muito próximo ao ambiente tradicional que levou 2532 milissegundos para executar essa tarefa. O *Exadata Software* através da função *cell offload* trata os blocos de dados antes de transferir do *storage server* para o servidor de banco de dados. Ao invés de enviar as mais de 200 milhões linhas para o servidor de banco de dados, os blocos enviados são referentes apenas ao resultado da condição *where*, ou seja, apenas as 1300 linhas. Além disso, a taxa de transferência é de 40 GB/seg via *infiniband*, se no ambiente tradicional a operação ocorreu em 17000 milissegundos, no ambiente *Exadata* ficou próximo a zero milissegundo. Mesmo executando praticamente as mesmas tarefas, o fato de tudo acontecer apenas no lado do *storage server* reduz consumo de processamento, volume de dados trafegado e efetivamente reduz o tempo de resposta em 95%.

Para melhor exemplificar a diferença entre o desempenho nos dois ambientes, foram criadas variações do cenário básico, onde a quantidade de produtos envolvidos no cenário principal foi de 50 mil produtos e nas variações os testes foram realizados com 5, 10 e 20 mil produtos. A Tabela 1 resume os resultados obtidos quando a volumetria da tabela de produtos é alterada.

Tabela 1: Comparativo dos cenários variando a quantidade de produtos nos ambientes *Exadata* e Tradicional.

Volumetria (Produtos)	Exadata Média e desvio padrão (ms)	Tradicional Média e desvio padrão (ms)

5000	970 ± 95,03	9300 ± 963,91
10000	1310 ± 52,63	12100 ± 1441,77
20000	2030 ± 299,12	21300 ± 1254,52
50000	1875 ± 179,26	19532 ± 759,73

Analisando os dados da Tabela 1, é possível observar que as funcionalidades e a arquitetura desenvolvida no *Exadata Software* possibilita um ganho expressivo na redução de tempo das consultas SQLs, pois o tempo médio da consulta SQL no ambiente Exadata é 0,01% do tempo da mesma consulta no ambiente tradicional. Em contrapartida as vantagem do ambiente *Exadata*, existe a desvantagem do alto custo financeiro para implementação do seu ambiente.

7 Conclusões

Com esse trabalho é possível observar que o conceito de *appliance* criado pela *Oracle* para disponibilizar no mercado uma solução na qual o *hardware* e o *software* trabalham perfeitamente alinhados traz ganhos significativos. Além disso, o *hardware* do *Exadata Machine* alinhado ao *Exadata software* e todo o conjunto de funcionalidades que traz a inteligência do processamento das consultas SQLs, o servidor de banco de dados e o *Storage server*, oferecem ganhos significativos quando *software* e *hardware* específico trabalham integrados.

Quando comparado com o ambiente tradicional de servidor de banco de dados Oracle, os recursos do *Exadata software* se mostraram mais eficientes na operação das consultas SQLs.

O futuro do *Exadata Machine* caminha para um banco de dados em memória, com isso, trabalhos futuros poderão demonstrar ambientes com um amadurecimento maior tanto da tecnologia para se tornar financeiramente viável, como para que os banco de dados OLTP não sejam impactados pela taxa de IOPS das memórias *flash's* que resulta em um aumento da taxas de *deadlock*.

Referências

[1] KIMBALL, R; ROSS, M. The Data Warehouse Toolkit - Wiley - Kimball Group - 3ª Edition, 2013.

[2] CLARKE, J. Oracle Exadata Recipes - A Problem-Solution Approach, 2013.

[3] KOROTKEVITCH, D. Expert SQL Server In-Memory OLTP - Revolutionizing OLTP Performance in SQL Server, 2015.

[4] OSBORNE, K; BACH, M; ARAO, K; COLVIN, A; HOOGLAND, F; JOHNSON, R; PODER, T. Expert Oracle Exadata - 2ª Edition, 2015.

[5] SILVERS, F. Building and Maintaining a Data Warehouse - CRC Press Taylor & Francis Group, 2008.

[6] Oracle Exadata Guider, http://docs.oracle.com/cd/E50790_01/welcome.htm. Último acesso em 19/10/2016.

[7] Oracle Exadata Database Machine, https://www.oracle.com/engineered-systems/exadata/index.html. Último acesso em 16/10/2016.

[8] WEISS, R. Uma visão geral técnica do Oracle Exadata Storage Server da Sun X2-8 - Oracle do Brasil Sistemas Ltda, 2009.

[9] WEISS, R. Uma visão geral técnica do Oracle Exadata Storage Server da Sun X4-8 - Oracle do Brasil Sistemas Ltda, 2012.

[10] KEVIN, C. Hardware Eye for the database Guy - Revista NoCOUG Journal, v. 26, n. 3, 2012.

[11] PORTUGAL, Paulo (Principal Sales Consultant). Oracle OpenWorld Latin America 2016 - Oracle Exadata: Novidades e planos . Ministrado em 30/06/2016.

[12] PEDREGAL, Cris (Consulting Member of Technical Staff). Oracle OpenWorld Latin America 2016 - Uma visão técnica aprofundada do Oracle Exadata: Arquitetura e características intrínsecas . Ministrado em 28/06/2016.

[13] NASCIMENTO, J. S; PAULUS, G. B; RUBERT, D.L.V.G.; ANTONIAZZI, R.L. Comparação da Eficiência de Bancos de Dados Relacionais como Data Warehouse em um Contexto de BI. UNICRUZ de Cruz Alta no Rio Grande do Sul, 2016.

[14] ANZANELLO, C.A. OLAP Conceitos e Utilização . UFRGS - Instituto de Informática – Universidade Federal do Rio Grande do Sul, 2007.

[15] TAVARES, J.A; MORAES FILHO, J.A; BRAYNER, A. SCM Join - Um Algoritmo de Junção

http://dx.doi.org/10.25286/repa.v2i2.552

para Memória SCM. UNIFOR - Universidade de Fortaleza, 2012.

[16] COSTA, C.L. Avaliação noSQL para Indexação de Dados. Universidade de Caxias do Sul, 2016.

[17] FOSTER, E.C.; GODBOLE, S. Database Systems - A Pragmatic Approach, 2014.

[18] COUGO, P. Modelagem Conceitual e Projeto de Bancos de Dados , 2011.

[19] PETERSEN, W.P;ARBENZ, P. Introduction to Parallel Computing , 2003.

[20] SINCIC, M. Technet Microsoft: https://technet.microsoft.com/pt-br/library/jj681657.aspx. Último acesso em 08/11/2016.

[21] Teradata Technology, https://www.teradata.com. Último acesso em 07/11/2016.

[22] SINGH, M; LEONHARDI B. Introduction to the IBM Netezza warehouse appliance , 2011.

[23] DIANA, M; GEROSA, M.A. NOSQL na Web 2.0: Um Estudo Comparativo de Bancos Não-Relacionais para Armazenamento de Dados na Web 2.0. Departamento de Ciência da Computação – Universidade de São Paulo (USP), 2011.

[24] YU, Kai et al. Design and Architecture of Dell Acceleration Appliances for Database (DAAD) : A Practical Approach with High Availability Guaranteed. In: High Performance Computing and Communications (HPCC), 2015 IEEE 7th International Symposium on Cyberspace Safety and Security (CSS), 2015 IEEE 12th International Conferen on Embedded Software and Systems (ICESS), 2015 IEEE 17th International Conference on. IEEE, 2015. p. 430-435.

[25] CROWTHER, P; LAKE, P. Concise Guide To Databases - A Practical Introduction (Undergraduate Topics in Computer Science), 2013.