Pontifícia Universidade Católica
do Rio de Janeiro

Otávio Freitas Teixeira

Auto-Sintonia para Sistemas de Bancos de Dados na Nuvem

Dissertação de Mestrado

Dissertação apresentada como requisito parcial para obtenção do título de Mestre pelo Programa de Pós-Graduação em Informática da PUC-Rio.

Orientador: Prof. Sérgio Lifschitz

Rio de Janeiro

Março de 2016

Pontifícia Universidade Católica
do Rio de Janeiro

Otávio Freitas Teixeira

Auto-Sintonia paraSistemas de Bancos de Dados na Nuvem

Dissertação apresentada como requisito parcial para obtenção do título de Mestre pelo Programa de Pós-Graduação em Informática da PUC-Rio. Aprovada pela Comissão Examinadora abaixo assinada.

Prof. Sérgio Lifschitz
Orientador
Departamento de Informática - PUC-Rio

Prof.ª Ana Carolina Brito de Almeida
Departamento de Informática – UERJ

Prof. Rogério Luis de Carvalho Costa
Departamento de Informática – PUC-Rio

Prof. Mário da Silveira Carvalho
Coordenador Setorial do Centro
Técnico Científico – PUC-Rio

Rio de Janeiro, 31 de março de 2016

Otávio Freitas Teixeira

Otávio Freitas Teixeira graduou-se em Análise de Sistemas pela PUC-RIO em 2009. Desde então trabalha como administrador de banco de dados na Fundação Getúlio Vargas, se envolvendo em diversos projetos Possui interesse acadêmico e profissional nas áreas ligadas Banco de dados, Computação na Nuvem e sintonia fina de banco de dados.

Ficha Catalográfica

Teixeira, Otávio Freitas

Auto-Sintonia para Sistemas de Bancos de Dados na Nuvem / Otávio Freitas Teixeira ; orientador: Sérgio Lifschitz. – 2016.
63 f. : il. color. ; 30 cm

Dissertação (mestrado)–Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2016.
Inclui bibliografia

1. Informática – Teses. 2. Auto-sintonia. 3. Auto-gerência. 4. DBaaS. 5. Multi-inquilino. 6. Nuvem. I. Lifschitz, Sérgio. II. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. III. Título.

CDD: 004

Dedico este trabalho a meus pais Rogério e Silvana. Que sempre me ensinaram que o futuro escreve com estudo.

Agradecimentos

A Deus por ser um porto seguro nas horas de aflição e dificuldades.

Aos meus pais, Rogério Felgueiras Teixeira e Silvana Paes de Freitas Teixeira, que sempre estiveram ao meu lado independente de qualquer coisa, dispostos a ajudar no que fosse preciso para proporcionar à minha família o melhor ambiente para o desenvolvimento desse e de outros trabalhos, meus pais dedicam todo o seu tempo a mim e a meus irmãos, Mateus Freitas Teixeira e Pedro Freitas Teixeira, a quem também devo agradecer por toda a paciência e apoio durante o período do mestrado.

Aos meus saudosos avôs, Alcebíades Teixeira Filho e Felizardo Duarte de Freitas, exemplos de pessoas determinadas que nunca desistiram de seus sonhos, apesar dos inúmeros obstáculos que encontraram ao longo da vida.

A toda minha família, avós, tios e primos que, de uma forma ou de outra, sempre prestaram sua solidariedade e seu apoio a cada novo desafio que me propus.

A todos os professores que compartilharam seus conhecimentos comigo e em especial ao meu orientador, Sérgio Lifschitz, pelo apoio, incentivo e orientação para desenvolvimento dessa dissertação.

A amiga, Marcia Lucas Pesce, pelas palavras de incentivo, apoio e ombro amigo nos momentos de dificuldades.

Aos meus amigos do laboratório BioBD, Ema Molina, Liester, Julio Omar, Alain, Rafael Pereira, Antony Seabra, que me ajudaram, com incentivando, conversas e conselhos durante a construção deste trabalho.

A todos os funcionários, em particular a Regina Zanon, que desde sempre me incentivou a ingressar no mestrado e, como uma segunda mãe, nunca mediu esforços para me auxiliar a resolver meus problemas burocráticos.

Resumo

Teixeira, Otávio Freitas; Lifschitz, Sergio. **Auto-Sintonia para Sistemas de Bancos de Dados na Nuvem.** Rio de Janeiro, 2016. 63p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

A computação na nuvem vem modificando a maneira pela qual os usuários acessam e usufruem de serviços computacionais. Um sistema gerenciador de banco de dados é um dos principais recursos deste novo ambiente de trabalho. Entretanto, os grandes volumes de dados devem ser adequadamente gerenciados e disponibilizados, de acordo com as oscilações das cargas de trabalho e em função de novos parâmetros existentes. Pelas dimensões do problema neste novo ambiente da nuvem, não há como dispor de um DBA que consiga, manualmente, administrar, manter disponibilidade e desempenho de maneira aceitável. Em particular, há necessidade de sintonia fina (tuning) automática pois o sistema na nuvem deve cumprir requisitos contratuais de operação e, para o usuário, deve-se oferecer recursos como se fossem ilimitados ao mesmo tempo que com excelente desempenho. Nesta dissertação são explicitadas e comparadas as atividades de (auto) sintonia fina em SGBDs que operam em ambientes convencionais e em ambientes de nuvem. Enfatizam-se as diferenças observadas na visão do provedor do serviço de nuvem e dos usuários em um contexto de DBaaS. Nesta pesquisa será proposta uma nova extensão da ontologia de domínio desenvolvida e aprimorada por [Almeida, 2013] e [Oliveira, 2015], a fim de abranger a sintonia fina em banco de dados na nuvem.

Palavras-Chave

Auto-sintonia; Auto-gerência; DBaaS; Multi-inquilino; Nuvem; Ontologia

Abstract

Teixeira, Otávio Freitas; Lifschitz, Sergio (Advisor). **Database Self-Tuning in the Cloud**. Rio de Janeiro, 2016. 63p. MSc. Dissertation – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

Cloud computing is changing the way users access and benefit from computer services. A database manager is one of the main features of this new working environment. However, large volumes of data must be properly managed and made available, according to the fluctuations in workloads and function of new and existing parameters. Because of dimensions problems in this new cloud environment, it is very difficult to have a DBA who can manually manage, maintain availability and performance acceptably. In particular, the necessity of a tuning process automatic in the cloud system to meet contractual operation requirements and the necessity of offering to the user resources as if they were unlimited while with excellent performance. This thesis explains and compares the activities of (self)-tuning database systems operating in conventional and cloud environments, emphasizing the differences observed in the cloud service provider's view and users in a context of DBaaS. In particular, it is proposed to extend of tuning ontology in order to automate actions to tuning the Database as a Service.

Keywords

Self-Tuning; Self-management; DBaaS; Multi-tenant; Cloud; Ontology

Sumário

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

BD Banco de Dados

DBA Data Base Administrator

DBaaS Data Base as a Service

MV Máquina Virtual

VM Virtual Machine

SGBD Sistema Gerenciador de Banco de Dados

SGBDs Sistemas Gerenciadores de Banco de dados

1
Introdução

A computação na nuvem é considerada por muitos pesquisadores uma revolução no mundo da tecnologia de informação, por proporcionar maior rapidez para incrementar e remover recursos de CPU, Memória, por exemplo, de acordo com a carga de trabalho e minimizar os custos de manutenção [Soror et al., 2017].

A utilização dos serviços na nuvem, como por exemplo, de banco de dados como serviço, atrai clientes de diferentes setores do mercado, desde pequenas empresas com objetivo de reduzir o custo total, com o uso de infraestrutura e sistemas de terceiros, até grandes organizações que buscam soluções para gerenciar milhares de máquinas e permitir o atendimento do aumento inesperado da carga de trabalho [Abadi, 2009] [Moreira, 2014]. Desta forma, existe a necessidade de ferramentas automáticas para realizar a administração dos serviços disponibilizados nestes ambientes.

Agrawal, 2012, afirma que a grande razão para o sucesso do sistema de banco de dados no ambiente de nuvem está relacionado com: (i) a capacidade de adequar-se a diferentes tipos de aplicação, pelo fato de utilizar o modelo relacional; (ii) pela consistência, performance e segurança dos dados; e (iii) pela persistência dos mesmos nos diferentes tipos de falhas.
Por muitos anos, SGBDs eram considerados não compatíveis à computação na nuvem, tendo em vista a dificuldade do SGBD em escalar [Agrawal et al, 2011]. Dessa forma, é crescente o número de estudos a fim de adequar os sistemas gerenciadores de banco de dados à essa nova tecnologia.

Nesse sentido, o presente trabalho tem como objetivo o estudo e a apresentação das possíveis soluções para adaptação do SGBD na nuvem, utilizando-se do conceito de multi-inquilino e as técnicas de live migration [Moreira, 2014]. Além disso, discutir-se-á os novos parâmetros adotados na sintonia fina dos bancos de dados na nuvem. Por fim, sugere-se a extensão da

ontologia [Almeida, 2013] de sintonia fina afim de tornar a auto-sintonia dos bancos de dados adaptada ao conceito de banco de dados na nuvem.

1.1 Motivação

A computação na nuvem impõe novos desafios, tendo em vista a crescente utilização de tais serviços de banco de dados na nuvem, não apenas por grandes companhias, mas também por empresas de pequeno e médio porte, além de usuários autônomos.

Com efeito, a sintonia automática dos sistemas de bancos de dados na nuvem representa um dos principais desafios a serem solucionados. Espera-se dos bancos de dados como serviço (DBaaS) que estes apresentem funcionalidades similares aos tradicionais bancos de dados [Chen et al, 2011] e, também, maior desempenho quando necessário, tendo em vista a possibilidade de ser escalável. Sendo assim, nota-se uma transformação no objetivo da sintonia fina quando se faz uso do DBaaS, pois, além de minimizar o tempo de resposta e elevar a vazão de dados, existirá a preocupação de que não se extrapole o investimento estimado, seja pela inclusão de mais recursos (CPU, memória, etc.), ou por eventual limite imposto pelo provedor de serviço na nuvem [Florescu et al., 2009; Silva, 2010; Shankaranarayana et al, 2013].

1.2 Objetivo

O presente trabalho tem como objetivo discutir as principais diferenças identificadas na sintonia fina, automática ou não, entre os banco de dados tradicionais e os bancos de dados como serviço (DBaaS). Para isto, propõe-se a extensão da ontologia de domínio desenvolvida pelo grupo de pesquisa BioBD da PUC-Rio [Almeida, 2013][Oliveira, 2015].

1.3 Organização do trabalho

A presente exposição foi dividida em quatro capítulos. No primeiro capítulo, objetiva-se expor a fundamentação teórica do tema proposto. Em seguida, no

segundo capítulo, expõe-se os trabalhos relacionados a este estudo. Logo após, no terceiro capítulo, é apresentado o estudo realizado sobre a sintonia fina em banco de dados tradicionais e na nuvem. Por fim, no quarto e último capítulo, apresenta-se a extensão da ontologia de sintonia fina [Almeida, 2013] e [Oliveira, 2015], como forma alcançar maior adequação e precisão no ambiente de nuvem.

2
Fundamentos

2.1
Sintonia fina de banco de dados

Sintonia de banco de dados abrange atividades como ajuste de suas configurações, parâmetros e projeto físico como: (i) seleção de estruturas de acesso, (ii) duplicação de estruturas físicas, (iii) determinar os objetos que devem ser particionados além dos tipos de particionamento a serem utilizados. Ainda em [Shasha and Bonnet, 2002], define-se sintonia fina como ações adotadas para permitir que o Sistema Gerenciador de Banco de Dados (SGBD) execute de forma mais eficiente, alcançando desta forma, melhor desempenho da carga de trabalho.

Em banco de dados a otimização de consultas e a sintonia fina são ações diferentes. Otimização de consulta é o processo do SGBD estabelecer os possíveis caminhos para acesso aos dados podendo utilizar estruturas de dados. Sintonia fina necessita da intervenção de um administrador do sistema (DBA) ou de um especialista em desempenho para fazer ajustes físicos e lógicos do banco de dados objetivando a melhora de performance [Milanés et al, 2004]. A atividade de sintonia fina pode influenciar nas alternativas de planos de execução gerados pelo otimizador, mas não determina qual deles será utilizado para responder uma determinada requisição, pois isso é uma função do otimizador [Almeida, 2013].

Os DBAs são capazes de criar novas estruturas de acesso ou modificar parâmetros de configuração, que influenciem na obtenção de novos planos de execução, deste modo, deixando o otimizador decidir qual seria a melhor forma de executar a consulta, selecionando ou não o novo plano.

Os SGBDs, por sua vez, oferecem diversos parâmetros, como por exemplo, quantidade de memória, nível de concorrência etc. O ajuste destes parâmetros pode contribuir para maior eficiência em seu funcionamento [Bruno, 2011] [Shasha etal, 2003] [Ramakrishnan et al, 2003].

Modificar parâmetros não é uma atividade trivial. Exige dos DBAs profundo conhecimento dos algoritmos envolvidos no SGBD além do impacto que a alteração

pode ocasionar [Bruno, 2011]. Dessa forma, um ajuste realizado deve ser cuidadosamente realizado uma vez que podem surtir efeito contrário e acabar degradando o desempenho do Banco de Dados (BD) [Almeida, 2013].

Em vista dos argumentos apresentados, entende-se que realizar a sintonia fina de banco de dados não é uma atividade simples exigindo consciência sobre as razões das recomendações existentes.

2.2
Sintonia fina automática em banco de dados

Realizar a sintonia fina de banco de dados manualmente não é uma tarefa tão simples [Bruno, 2011], uma vez que os SGBD estão cada vez mais complexos. Sendo assim, é necessário a elaboração de sistemas que tornam a sintonia fina automáticas. Sistemas automáticos de sintonia fina também devem ser aplicados à infraestrutura (Servidores, Sistemas Operacionais entre outros), que cada vez mais torna-se transparente para o usuário, e pode influenciar na performance do SGBD. Todas as ferramentas devem levar em consideração o dinamismo da carga de trabalho para assim tomar decisões de sintonia [Bruno, 2011] [Lifschitz et al, 2004].

Visto a complexidade e a clara necessidade de automação, é apresentado o conceito de auto-sintonia de banco de dados, sintonia fina automática de banco de dados ou self-tuning de banco de dados [Almeida, 2013]. O significado da auto-sintonia de banco de dados é o ajuste automático de configurações que visam melhorar o desempenho por meio da diminuição do tempo de resposta sobre as transações ou comandos submetidos ao banco de dados.

Em [Bruno, 2013], é argumentado que as ferramentas de auto-sintonia de banco de dados envolvem três componentes principais: espaço de busca, modelo de custo e estratégia de enumeração.

- **Espaço de busca:** para a sintonia fina é o conjunto de estruturas de acesso candidatas para uma base de dados de acordo com uma determinada carga de trabalho.
- **Modelo de custo:** é elaborado para avaliar qual seria o custo de execução dos comandos que fazem parte de uma determinada carga de trabalho juntamente com uma determinada configuração durante a sintonia fina. Esse modelo é criado fora do otimizador.

- **Estratégia de enumeração:** a partir das limitações encontradas no ambiente em que o SGBD está instalado, é feito a enumeração das configurações candidatas. Como exemplo, a estratégia pode considerar o espaço limitado de disco ao criar uma determinada estrutura de acesso.

A elevação da sofisticação das máquinas de consultas, cargas de trabalho e ambientes em que o SGBD está posto, fez com que a técnica de sintonia fina fosse aperfeiçoada ao ponto de utilizar heurísticas para simular configurações relevantes para o espaço de busca analisado.

Portanto, quanto mais complexo é o ambiente e o SGBD, maior é a complexidade no processo de sintonia fina de banco de dados. Assim, para que as ferramentas de automação possam realizar a sintonia, sem que haja a intervenção humana, existe a necessidade que seu raciocínio seja explicado. E por isso existem ferramentas que utilizam técnicas de ontologia como proposta na tese de doutorado de [Almeida, 2013] e desenvolvida por [Oliveira, 2015].

2.3
Gerenciador de banco de dados na nuvem

Cloud Storage, Data as a Service (DaaS) e Database as a Service (DBaaS) são termos utilizados para gerenciamento de dados na nuvem [Arora et al, 2012]. No entanto, os termos definem formas diferentes de gerenciar o armazenamento dos dados. Cloud storage permite aos usuários armazenar documentos, fotos e demais objetos em um armazenamento virtual. Exemplos desse tipo de serviços de armazenamento podem ser destacados Dropbox, iCloud e Google Drive. Com DaaS os usuários são capazes de armazenar dados em discos remotos disponibilizados através da internet. Na maioria das vezes esse serviço é utilizado para backup e gerenciamento básico de dados. O banco de dados com serviço (DBaaS), que será pesquisado, pretende oferecer aos usuários todos os recursos dos bancos de dados além de permitir acessar e armazenar seu banco em discos remotos, a qualquer hora e de qualquer lugar por meio da internet [Oracle, 2011] [Arora et al, 2012].

Banco de dados na nuvem proporciona escalabilidade, alta disponibilidade, alocação otimizada de recursos, performance, flexibilidade em custo baixo além da implementação de bancos de dados multi-inquililos (multi-tanancy). Outra

funcionalidade importante que se nota no DBaaS é permitir ao usuário criar e "administrar" seus bancos de dados por eles mesmos. Bancos de dados na nuvem diferenciam-se dos SGBDs tradicionais pelo fato de permitirem o armazenamento de dados não estruturados, semi-estruturado ou estruturados, enquanto os tradicionais são projetados para aceitar os dados estruturados [Arora et al, 2012].

Dessa forma, os SGBDs da nuvem devem suportar as funcionalidades da computação na nuvem assim como as dos bancos de dados tradicionais. Dentre os grandes desafios podem ser destacados:

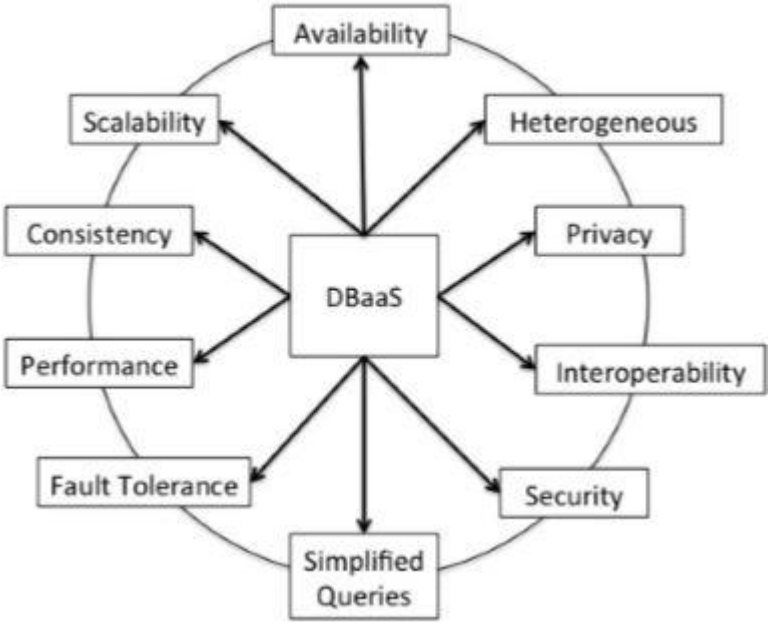

Figura 1 Possíveis desafios dos gerenciadores de bancos de dados na nuvem [Arora et al, 2012]

Escalabilidade

A escalabilidade é uma das principais características do paradigma de serviços na nuvem. Possibilita que os recursos possam ser adicionados e retirados a medida da necessidade sem que haja interrupção no serviço. Os desenvolvedores dos sistemas de gerenciadores de banco de dados na nuvem são desafiados a tornar os bancos de dados capazes de suportar a inserção de mais recursos como memória, CPU, além de novos servidores. Dessa maneira, é possível tolerar um grande número de usuários concorrente e a o aumento dos dados.

Alta disponibilidade e tolerante a falha

Os bancos de dados devem estar disponíveis sempre. Para que sejam tolerantes às possíveis falhas, os bancos são replicados geograficamente para, assim, garantir a alta disponibilidade e ser tolerante às falhas que são tidas como normais [Mohammad et al, 2013].

Ambiente heterogêneo

Esses bancos de dados devem ser acessados por meio de diferentes dispositivos como, smatphones, tables, notebooks e computadores. Assim como os dados e a forma como os usuários vão acessar esses dados é difícil pré-definir como os usuários vão usar o sistema.

Integridade e Consistência de dados

Para bancos relacionais na nuvem, consideram-se propriedades chamados de BASE (Basically Available, Soft state, Eventually Consistent), ou seja, em contraste com ACID, para bancos de dados na nuvem é esperado que os dados estejam eventualmente consistentes pois os dados são replicados por muitos locais. Por isso, torna-se complicado manter a consistência de transações de banco de dados que possuem mudanças muito rápidas de dados.

Consulta Simplificada

Bancos de dados na nuvem podem ser distribuídos. Consultas nesses bancos podem acessar muitos nós. Assim recomenda-se a padronização e simplificação das consultas que são realizadas no banco de dados.

Privacidade e Segurança de banco de dados

Os arquivos de dados são fisicamente armazenados em algum país, portanto, estão submetidos às leis e regulamentos específicos. Com por exemplo, nos Estados Unidos, o governo americano possui permissão para acessar os dados. Para evitar o risco de captura de informação armazenada nos bancos de dados, os dados são criptografados. O que desafia os fabricantes dos SGBDs a desenvolver algoritmos de criptografia que não interfiram tanto na performance das consultas.

Portabilidade de dados e interoperabilidade

Os usuários possuem a liberdade de movimentar seus bancos de dados pelos diferentes provedores do serviço de banco de dados na nuvem. Sendo assim, o

desafio está em desenvolver APIs que auxilie na migração dos bancos de dados entre os provedores de serviço.

2.4
Banco de dados como serviço (DBaaS)

O expressivo aumento de dados no ambiente de nuvem deve-se à crescente demanda das aplicações baseadas na web, uma vez que as empresas necessitam armazenar os dados e/ou logs de cada interação realizada pelo usuário. Desse modo, essa tendência resultou em um crescimento significativo da quantidade de dados associada a essas aplicações. A forma como tradicionalmente é feita a armazenagem e a recuperação dos dados impõe às empresas em geral, principalmente, às empresas de pequeno porte, um custo muito elevado, pois o valor do gerenciamento dos dados é cerca de 5 (cinco) ou 10 (dez) vezes maior que o custo em obtê-los, segundo [Agrawal, 2011]. Além disso, a gestão de dados na forma tradicional, requer dos profissionais um alto nível de conhecimento das tecnologias de armazenamento de dados, planejamento de capacidade e métodos de tolerância à falhas, buscando garantir, em caso de desastre, a confiabilidade e a recuperação dos dados perdidos. Sem descartar a necessidade de sustentação do DBMS (Database Management System) e a atualização dos sistemas operacionais.

Dessa forma, a terceirização do armazenamento de dados, utilizando-se do banco de dados como um serviço, surge como uma solução para a gestão de dados, ou seja, um prestador de serviços hospeda o banco de dados e fornece o suporte de software e hardware associados [Agrawal, 2011]. Nesse conceito, a responsabilidade pelo gerenciamento de hardware e software, onde os bancos de dados estarão hospedados, será do provedor de serviço de banco de dados. Com isso, busca-se a diminuição dos custos de armazenamento dos dados, sendo muito atraente para as empresas que possuem um elevado volume de dados. Esse serviço propicia, ainda, um acesso seguro e confiável através da internet.

Portanto, o banco de dados como serviço aplica o paradigma da computação na nuvem de Software como Serviço (SaaS), em que permite-se que softwares possam ser utilizados via Internet por usuários ou por outros sistemas de informação. Dessa forma, o provedor de serviço na nuvem escolhe o software,

como o é caso de sistemas de gerenciamento de banco de dados, realiza a instalação, administra e o gerencia, além de permitir o acesso, via internet, a muitos outros usuários. Assim, o consumidor do SaaS deixaria de se preocupar com o licenciamento, administração e manutenção do software contratado.

Nesse sentido, ao disponibilizar sistemas de gerenciamento de banco de dados (SGBD) em uma estrutura de nuvem, esse serviço passa a ser chamado de banco de dados como serviço (DBaaS).

Os bancos de dados como serviço propiciam aos usuários uma significativa diminuição dos custos operacionais. Isto se explica, pois tais bancos apresentam maior facilidade de configuração, de escalabilidade, performance e backup, além de diminuir a preocupação com o isolamento dos bancos de dados, tendo em vista que toda essa administração ficará à cargo do provedor do serviço. Como exemplos de grandes provedores de serviço de banco de dados, citam-se: Amazon com o RDS e SimpleDB, a Microsof com o SQL Azure e a Google com o Google Cloud SQL.

As plataformas de nuvem oferecem seus serviços de banco de dados em diversos tipos de arquiteturas. Dentre elas, pode-se citar a arquitetura em camadas que é considerada a mais clássica, tida esta como ponto de partida para o desenvolvimento do banco de dados na nuvem. Logo em seguida surgiram as arquiteturas de replicação, de particionamento e de distribuição.

A seguir serão expostas as arquiteturas adotadas pelos principais provedores de banco de dados [Cheng et al., 2016].

Clássico

Atualmente, a arquitetura de banco de dados clássica é utilizada pela maioria das aplicações. Nessa arquitetura, os usuários enviam suas requisições ao pool de servidores da aplicação e este envia os comandos SQL necessários ao servidor de banco de dados. Em seguida, o servidor de banco de dados processa as consultas e retorna o resultado para a aplicação [Cheng et al., 2016].

A principal característica desta arquitetura é possuir muitos servidores de aplicação conectados em um único servidor de banco de dados que faz uso de um ou mais discos presentes nas áreas de armazenamento.

Escalar essa arquitetura é bastante simples no que diz respeito aos servidores localizados nas camadas de aplicação e storage. Essas duas camadas podem ser escaladas horizontalmente, por exemplo, para escalar a camada de aplicação basta

acrescentar mais servidores ao conjunto. Assim como, para a camada de storage, pode-se adicionar mais discos semelhantes ou mais rápidos.

Já a camada do servidor de banco de dados, que fica entre as camadas de storage e de aplicação, apresenta maior dificuldade de escalar. Uma possível forma de escalar o servidor de banco de dados seria adquirir um que possua maior capacidade, porém esbarra na seguinte questão: o quanto maior deve ser esse servidor? Essa limitação faz com que a arquitetura clássica não seja a mais recomendada para DBaaS [Cheng et al., 2016].

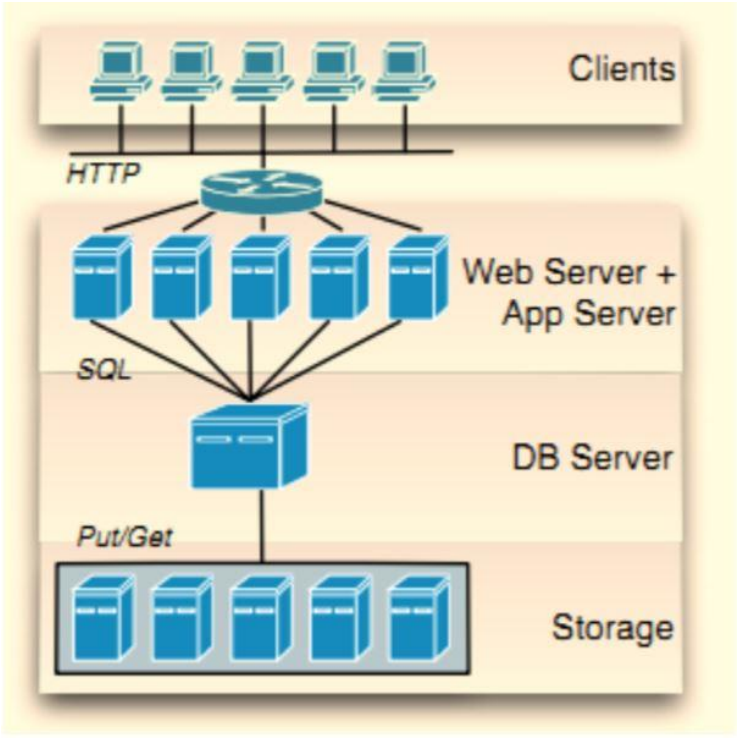

Figura 2 Ilustração da arquitetura clássica [Cheng et al., 2015]

Particionado

A arquitetura particionada, ao invés de utilizar apenas um servidor de banco de dados, divide o banco de dados em diferentes partições e cada uma dessas partições passa a ser gerenciada por um servidor exclusivo. Com isso, essa arquitetura consegue amenizar a dificuldade de existir um único servidor para suportar o banco de dados, tornando essa arquitetura adequada à sua utilização em um ambiente de nuvem.

O problema da arquitetura particionada é justamente a sua complexidade. Uma vez que, para adicionar ou remover um servidor de banco de dados, será necessário reavaliar as partições para que sejam redistribuídas entre os servidores

de banco de dados que permanecem ativos. Além disso, os servidores de aplicação também sofrem modificações, pois eles devem identificar quais partições estão alocadas em quais servidores de banco.

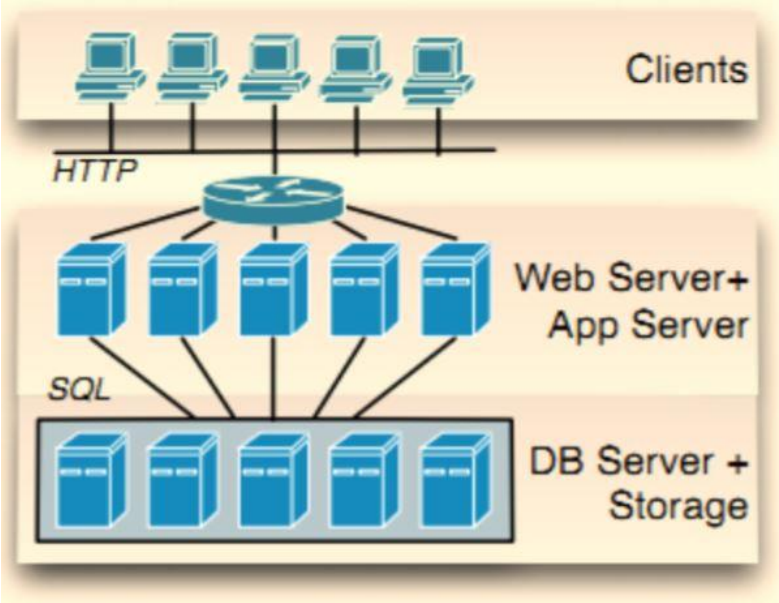

Figura 3 Ilustração da arquitetura de partição [Cheng et al., 2015]

Replicação

A arquitetura de replicação, diferentemente da arquitetura de partição que espalha o banco de dados, em forma de partição pelos servidores, mantém uma cópia do banco em cada um dos servidores 'escravos' do banco de dados presente no servidor master (ou principal).

Dessa forma, as operações de leitura de dados ("SELECT") podem ser processadas por qualquer um dos servidores, o que torna apropriado para cargas de trabalho que possuem operações intensas de leitura. No que se refere às operações de escrita ("INSERT", "UPDATE", "DELETE"), essas são realizadas exclusivamente na master. Isso faz com que, a cada commit de uma determinada transação, o servidor de banco de dados principal propague os dados para os servidores 'escravos' [Cheng et al., 2016].

Sendo assim, a adoção dessa arquitetura é bastante recomendada em sistemas que possuem elevado índice de operações de leitura, uma vez que, com índice de escrita de dados maior, as operações estarão direcionadas para apenas um banco de dados.

O ponto positivo dessa arquitetura é o fato de cada um dos servidores 'escravos' tornarem-se um backup, quase que em tempo real do banco de dados

principal. Em sistema de larga escala pode ser utilizado uma mescla entre a arquitetura de partição e a de replicação. Ou seja, o dado é primeiramente separado em partições e, assim, cada partição é configurada como um master que possuirá um conjunto de ‘escravos’ [Cheng et al.,2015].

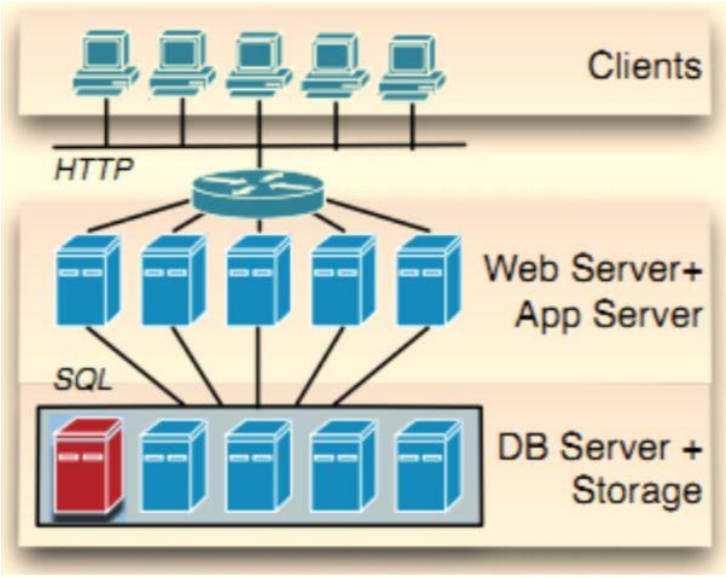

Figura 4 Ilustração da arquitetura de replicação [Cheng et al., 2015]

Controle Distribuído

A arquitetura de controle distribuído possui baixo acoplamento entre os nós, a fim de alcançar maior escalabilidade.

Basicamente, nessa arquitetura, a camada de armazenamento está separada da camada de banco de dados que, por sua vez, junta com a camada da aplicação. Dessa forma, os servidores de banco de dados possuem permissão para ler e escrever no storage compartilhado, utilizando protocolos de distribuição a fim de garantir diferentes níveis de consistência. Desse modo, todas as camadas possuiriam capacidade elástica, o que leva a ser uma arquitetura fortemente indicada para banco de dados na nuvem.

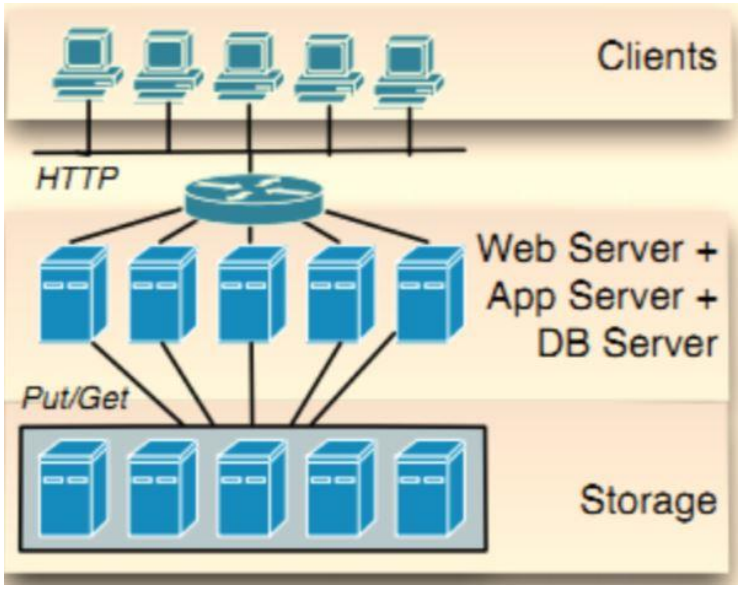

Figura 5 Ilustração da arquitetura de controle distribuído [Cheng et al., 2015]

Vantagens do SGBD na nuvem

A partir do surgimento da computação na nuvem, a indústria da tecnologia de informação vem se transformando. As companhias estão em processo de adoção dos serviços em nuvem ao invés de investirem na aquisição de infraestrutura para sistemas de gerenciamento de banco de dados [Snehal et al., 2015].

São muitas as vantagens de adotar essa nova tecnologia e esse novo conceito de sistema de gerenciamento de banco de dados.

O serviço de banco de dados no ambiente de nuvem permite aos desenvolvedores de software uma forma mais rápida e eficiente para criação de novos bancos de dados a fim de auxiliar na resolução das demandas dos usuários. No modelo tradicional, a elaboração de um plano de capacidade e a aquisição de infraestrutura para o projeto demandava bastante tempo [Snehal et al., 2015].

Adotar um sistema de gerenciamento de banco de dados na nuvem contribui na diminuição dos custos operacionais que as empresas possuem, como, por exemplo, ao adquirirem licenças dos SGBDs. Esse modelo maximiza, ainda, a utilização da infraestrutura, de modo que não fique subutilizada ou mesmo hiperutilizada.

As características de elasticidade e escalabilidade da computação na nuvem permite que os bancos de dados sejam mais adaptáveis aos picos das cargas de trabalho.

A utilização de técnicas de live migration [Moreira, 2014] visa, em casos de falha da infraestrutura migrar o banco de dados afim de não causar indisponibilidades ou garantir o cumprimento do acordo de nível de serviço nos casos em que haja um pico da carga de trabalho, mover o banco de dados automaticamente e de forma transparente para outra infraestrutura que possa suportar o aumento.

Por fim, para garantir a segurança dos dados do contratante, estudos sobre criptografia dos bancos de dados estão cada vez mais em voga. Ressalta-se que todos os SGBDs que estão na nuvem implementam algum algoritmo de criptografia de dados [Curino et al., 2011].

Desvantagens do SGBD na nuvem

As empresas que contratam o serviço de banco de dados na nuvem pagam pelo que utilizam, ou seja, pagam pelo espaço em disco ocupado pelo banco, pelo processamento e também pelo volume de dados que é trafegado. Assim, se a carga de trabalho for elevada e, por consequência, o trafego de dados na rede aumentar, a empresa terá de pagar por esse uso além da expectativa [Snehal et al., 2015].

Uma outra desvantagem é o fato de que no banco de dados de nuvem não se tem acesso total aos servidores onde o banco está hospedado. Dessa maneira, não se tem controle sobre o software que está instalado nos servidores. Não é permitido ao usuário verificar ou modificar a forma como a segurança foi implementada. Portanto, o usuário precisa confiar inteiramente no provedor do serviço de nuvem. Esse ponto gera um grande problema para a adoção de banco de dados em um ambiente de nuvem [Snehal et al., 2015].

Nesse sentido, os dados que estão armazenados na nuvem são totalmente dependentes do provedor de serviço. Sendo assim, as políticas de confidencialidade e garantias de que os dados não serão compartilhados, devem ser respeitadas, pois, além dos dados estarem em um ambiente compartilhado, estes também estão disponíveis na internet. Se as informações de uma determinada empresa forem distribuídas para as demais, esta sofrerá grandes perdas [Snehal et al., 2015].

A transferência de dados entre o banco de dados na nuvem e a estação de trabalho é o mais crítico, visto que a velocidade da internet influenciará no processo. Com isso é importante que se tenha uma internet de alta velocidade, se não, os bancos de dados tradicionais terão sempre maior taxa de transferência de dados [Snehal et al., 2015].

A escolha por um provedor de DBaaS tem que ser feita de maneira cautelosa. Pois, se for necessária a transferência de um banco para outro provedor, isso pode ocasionar grandes problemas, uma vez que os provedores de serviços na nuvem podem adotar arquiteturas diferentes e, em algumas situações, incompatíveis [Snehal et al., 2015].

Por fim, o fato dos dados serem acessados pela internet pode ocasionar alguns problemas pois, caso o servidor esteja indisponível, seja por estar desligado ou por problemas na internet, tornará as informações inacessíveis, gerando grandes perdas para as empresas [Snehal et al., 2015].

2.5
Banco de dados Multi-Inquilino

Os provedores de serviço na nuvem têm investido cada dia mais em modelo de gestão e armazenamento de dados na nuvem, visto que possuem um desafio constante de gerenciar e armazenar dados que são oriundos de diferentes tipos de aplicações, cada qual com suas características quando se trata da carga de trabalho. Estas aplicações são conhecidas como Inquilinos [Curino et al., 2011]. Porém, tornar um servidor de banco de dados específico para cada um dos inquilinos resulta em um desperdício computacional. Uma solução que está sendo adotada pelas empresas para consolidar os bancos de dados e assim minimizar os recursos computacionais é a implantação de bancos de dados multi-inquilinos ou multi-tenant, o que permite aos SGBDs na nuvem gerenciar grandes volumes de inquilinos que possam suportar cargas de trabalhos distintas. Ao aplicar essa técnica é possível que aplicações de múltiplos usuários possam ser consolidadas em um mesmo hardware, evitando a necessidade de servidores separados para cada inquilino. Multi-tenancy pode ser aplicado a todas as camadas de um software na nuvem: na camada web ou aplicação, camada de caching e na camada de banco de dados [Agrawal et al.,2012].

As aplicações que utilizam a plataforma de nuvem podem apresentar uma grande variação quanto a carga de trabalho, resultante da utilização por um elevado número de usuários de maneira relâmpago, além, é claro, da variação que pode ocorrer pelo uso dos recursos computacionais, como CPUs e memória. Portanto, como consequência, os provedores de serviço na nuvem que hospedam essas aplicações enfrentam grandes desafios por conta da imprevisibilidade para oferecer

e administrar os dados destas aplicações. Dentre esses desafios, ressalta-se a instalação e administração de um SGBD para: suportar os inquilinos, ser tolerante a falhas, permitir o particionamento dinâmico do banco de dados, realizar o balanceamento elástico da carga, além de assegurar o uso efetivo dos recursos e otimizar o custo [Agrawal et al.,2012].

O conceito de multi-inquilino em banco de dados tem sido utilizado predominantemente no contexto de Software como Serviço (SaaS). Com isso, identificam-se diferentes modelos de multi-inquilinos na camada do banco de dados e diversas formas de interação com o compartilhamento de recursos nos vários paradigmas da nuvem [Agrawal et al.,2012].

2.5.1
Modelos de bancos multi-inquilinos

Modelo Shared-table

O modelo Shared-table aplica o compartilhamento, não apenas de uma instância de banco de dados, mas de tabelas entre os inquilinos. Ou seja, cada tabela possui uma coluna que é responsável por identificar a qual inquilino uma determinada tupla pertence. Dessa forma, todos os inquilinos estarão compartilhando as mesmas tabelas [Albach, 2011].

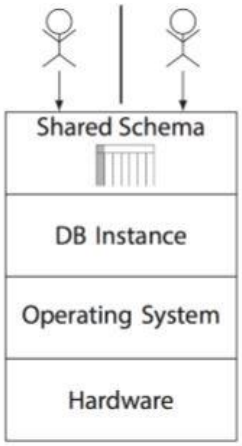

Figura 6 Ilustração do modelo Shared-table [Albach, 2011]

Account			
Tenant	Aid	Name	...
17	1	Acme	...
17	2	Gump	...
35	1	Ball	...
42	1	Big	...

Figura 7 Exemplo de tabela [Albach, 2011]

Para [Agrawal et al., 2012], [Schiller et al., 2011] e [Albach, 2011] o modelo shared-table apresenta problemas em sua abordagem. Os dados dos inquilinos são armazenados todos em um mesmo conjunto de tabelas o que torna a administração da instância problemática, pois cada ajuste realizado é refletido para todos os inquilinos como por exemplo, (i) ao realizar o processo de live migration todos os inquilinos são migrados e a configuração da nova infraestrutura pode influenciar negativamente em um inquilino. (ii) A atualização do SGBD é refletida para todos os inquilinos. (iii) A criação de um índice em uma determinada coluna significa que todos os inquilinos também farão uso desse índice. (iv) A segurança dos dados deve ser feita por tupla uma vez que o nível de isolamento é baixo.

Modelo Shared-process

Já no Shared-process, os inquilinos que lá estão instalados partilham uma mesma instância do banco de dados, o que permite compartilhar de forma efetiva recursos entre os inquilinos, além do fato de o SGBD conseguir gerenciar disco e memória, possibilitando um número maior de inquilinos de maneira consolidada e a utilização dos mesmos recursos físicos sem que haja problemas com o desempenho.

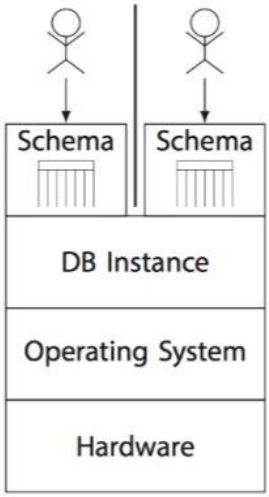

Figura 8 Modelo shared-process [Albach, 2011]

No modelo shared-process todos os inquilinos compartilham a mesma instância de SGBD e consequentemente os recursos disponibilizados para ela. Nessa configuração, a alocação de recursos para um inquilino deve ser controlada pela instância. Quanto ao isolamento, direitos de usuários devem ser mantidos com cuidado para que os inquilinos acessem apenas seus dados individuais [Agrawal et al., 2012]. O torna o isolamento dos dados, nesse modelo não tão eficiente quanto ao shared-hardware, já que estarão alocados com os dados de outros inquilinos.

Assim como no shared-table a instalação de nova versão ou ajustes em parâmetros da instância do SGBD pode afetar a todos os inquilinos. Assim a administração não é tão isolada quanto o shared-hardware [Albach, 2011].

Modelo Shared-hardware

Para implementação do modelo de compartilhamento de hardware, Shared-hardware, cada inquilino é responsável por uma instancia do SGBD [Albach, 2011]. Para esse modelo, quando não é utilizado a virtualização, existe um processo do SGBD por inquilino compartilhando o mesmo sistema operacional em um servidor compartilhado. Por outro lado, quando utilizado máquinas virtuais (MV) os recursos de hardware são compartilhados entre os inquilinos. Assim, cada inquilino possui uma instância do SGBD e um Sistema Operacional [Albach, 2011]. A figura abaixo ilustra o modelo de virtualização de hardware sem o recurso da virtualização.

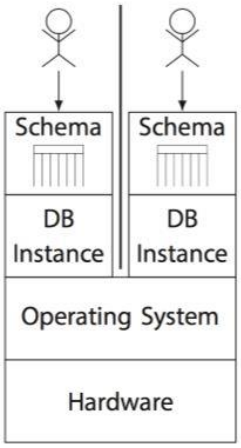

Figura 9 Modelo Shared-hardware [Albach, 2011]

Para a variante sem virtualização, a alocação de memória não é compartilhada pelos inquilinos. Para cada sistema de gerenciamento de banco de dados é feita a alocação de uma determinada quantidade de memória. Desta maneira, o modelo em questão não permite grandes volumes de inquilinos compartilhando o mesmo servidor [Lazarov, 2007].

A atualização dos SGBDs, é realizado em cada um dos inquilinos, o que onera as atividades administrativas durante o processo de upgrade porém permite que haja diferentes versões para diferentes inquilinos [Albach, 2011]. Além disso, apesar do longo processo de atualização, o tempo de indisponibilidade durante esse processo é sentido por apenas um dos inquilinos.

Análise dos modelos

O shared-hardware permite o uso da virtualização. Dessa forma, cada máquina virtual abrigará um único processo de banco de dados que deverá suportar todos os dados de uma determinada aplicação [Agrewal et al, 2012].

O modelo shared-table armazena os múltiplos inquilinos, utilizando-se das tabelas compartilhadas, o que influencia no menor nível de isolamento. Por sua vez, esse modelo exige mudanças na construção dos esquemas de bancos de dados, tendo em vista a adoção de uma única tabela compartilhada entre todos os inquilinos [Agrewal et al, 2012].

O modelo shared-process permite esquemas independentes por inquilinos que compartilham o mesmo processo do banco de dados entre os múltiplos inquilinos. Esse modelo é o que fornece o um maior nível de isolamento quando comparado

com o shared-table. Portanto, o modelo de compartilhamento de processo pode ser considerado o meio termo entre os modelos de multi-inquilinos.

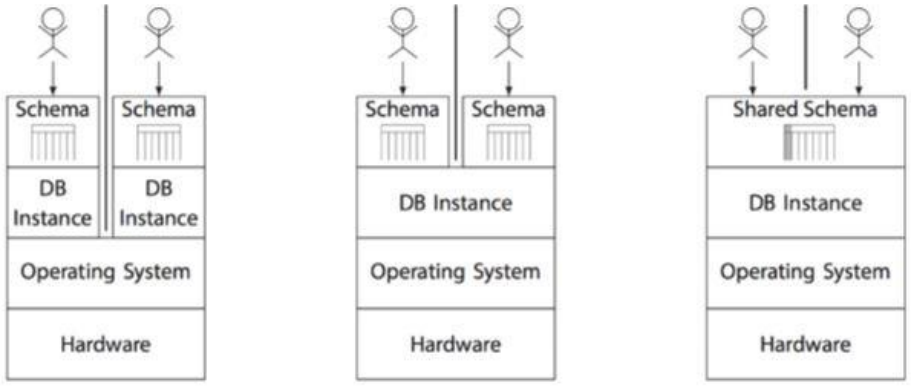

Figura 10 Modelos de multi-inquilinos [Albach, 2011]

Desse modo, identificam-se três principais paradigmas na computação em nuvem: IaaS (Infraestrutura como Serviço), PaaS (Plataforma como Serviço) e SaaS (Software como serviço). Para cada um destes paradigmas, o inquilino é exposto a um nível de abstração.

No paradigma de IaaS, o provedor do serviço disponibiliza uma máquina virtual, implementando o modelo *shared-hardaware*, por sua vez, tornam-se responsáveis pelos seus próprios SGBDs, esquemas, projeto físico do banco de dados, backup, etc. [Agrawal et al, 2012].

Já no paradigma de plataforma como serviço (PaaS) é verificado maior nível de abstração, ou seja, toda a interação do inquilino com o banco é realizada de maneira lógica e, não possuem controle sobre o layout físico dos dados e sobre a replicação.

Por fim, no software como serviço (SaaS), o inquilino interage apenas com o software (nível lógico), não tendo conhecimento do layout dos dados, por possuir acesso restrito ao SGBD [Agrawal et al, 2012], possuindo, assim, o maior nível de abstração em comparação aos modelos anteriores.

Para este trabalho, será adotado o paradigma IaaS, que é mais adotado pelas organizações e implementa o modelo *shared-hardware*.

2.6
Live Migration

O Live Migration tem o intuito de balancear a carga elástica e escalar sem que haja indisponibilidade, atuando com interrupção mínima em banco de dados multi-inquilinos, que estão alocados em um ambiente de computação em nuvem [Moreira, 2014]. O live migration pode ser implementado de algumas maneiras. No presente trabalho serão estudadas três delas, de modo a expor suas principais características e formas de funcionamento. Por último, será discutida a aplicabilidade das técnicas.

Implementações de live migration

O primeiro método abordado para implementação da live migration é baseado no conceito de virtualização na camada de banco de dados. O objetivo deste método é mover a representação lógica de dados, de metadados e do estado que representa um inquilino no banco de dados, a partir do SGBD que o hospeda. Assim, é utilizada a técnica de interative copy, cuja intenção é transferir o estado de memória principal, a fim de que no destino estes dados e metadados reiniciem [Moreira, 2014].

Conceitualmente, o estado de memória principal é composto pelo estado do banco de dados em cache, além do estado da transação. Dessa forma, o estado do banco de dados é constituído de páginas de dados armazenadas em cache e buffer [Moreira, 2014]. Já o estado de transação engloba a transação ativa e, possivelmente, um subconjunto de transações efetivadas, indispensáveis para a validação das transações ativas. O método de interative copy permite a mínima interrupção do serviço de banco de dados. O Live Migration com interative copy é composto por duas fases principais ilustradas na figura abaixo.

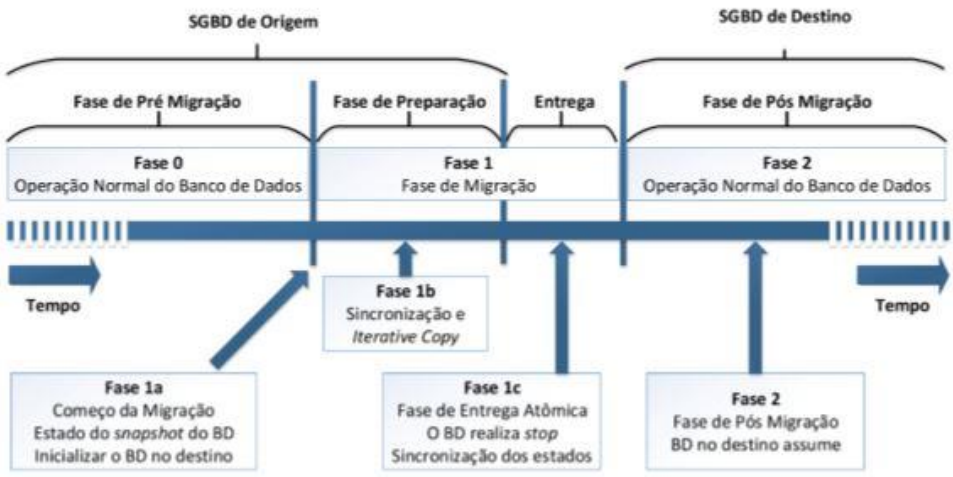

Figura 11 Fases da live migration [Moreira, 2014]

A fase 0 é compreendida como o operação normal do banco de dados, executando suas transações. Na fase1, princípio da migração, será realizada a notificação da fonte e o destino do início do processo de movimentação. A migração inicia-se com a realização do snapshot do estado de banco de dados. Assim, a migração é iniciada a partir desta fotografia do banco para o novo destino. Posteriormente, é iniciada a interative copy com o snapshot sendo migrado e a origem do dado, ainda ativa, recebendo transações. O processo de cópia interativa tenta recuperar o atraso e sincronizar os estados entre a fonte e o destino do banco de dados. Feita a sincronização, é realizada, logo em seguida, a migração completa do banco de dados, deixando de funcionar a fonte e transferindo o estado final do banco de dados para o destino. A fase 2, pós-migração, é entendida como o início da operação no novo destino.

O segundo método é o Stop and copy, uma técnica considerada tradicional de migração de banco de dados, que consiste em interromper o funcionamento do banco de dados, copiar e migrar em seguida para o novo destino, iniciando o serviço novamente. Por último, a On demand migration realiza a transferência do mínimo de informações, consistindo em uma parada rápida e cópia de migração [Moreira, 2014].

Estas características tornam o modelo de Live Migration on demand migration mais atraente, apesar de complexa implementação [Moreira, 2014]. Com relação a implementação, o stop and copy, conforme analisado, é considerada a técnica mais simples, apesar de apresentar desempenho inferior às demais técnicas,

pois realiza a suspensão do serviço, deixando a migração mais lenta, o que para um ambiente de nuvem, é indesejado. Conforme afirmado ao logo deste trabalho, pelos conceitos da computação na nuvem, todo serviço oferecido deve ter o mínimo de indisponibilidade.

2.7 Ontologia

O termo ontologia é definido na computação como um conjunto de primitivas relacionais responsáveis por modelar um domínio de conhecimento. Sendo esses primitivos objetos (conjunto de classes), propriedades (ou atributos) e as relações entre os objetos (ou entre membros de classe) [Gruber, 2009].

Segundo [Noy, 2004], o uso da ontologia tem como finalidade compartilhar entre pessoas e agentes de software um entendimento comum da estrutura de informação, proporcionar a reutilização de elementos do domínio de conhecimento. Explicitar as premissas e suposições do domínio. Distinguir o conhecimento de domínio do conhecimento operacional; e, por fim, possibilitar a análise do conhecimento do domínio.

Ontologias são classificadas de acordo com sua generalidade, definindo assim seus quatro tipos básicos:

- **Ontologia de fundamentação:** é o mais geral, ou seja, independentes de problemas particular ou domínio específico. Dessa maneira as ontologias de fundamentação podem ser utilizadas na elaboração de novas ontologias.
- **Ontologia de domínio:** utilizado para descrever classes e relacionamentos de conceitos que definem um determinado domínio.
- **Ontologia de tarefa:** classe para definir quais são os conceitos e relacionamentos utilizados para a realização de uma tarefa genérica.
- **Ontologia de aplicação:** considerada uma especialização da ontologia de domínio ou tarefa, mas é específica por ser utilizada dentro de aplicações. Tem como objetivo descrever conceitos utilizando tanto um domínio particular, quanto uma tarefa específica.

No desenvolvimento dessa pesquisa, foi considerada a utilização de uma ontologia de aplicação para sintonia fina [Almeida, 2013] [Oliveira, 2015], a qual

foi estendida para considerar sintonia fina na nuvem. A mesma é formada pelas ontologias de domínio, responsável pela descrição de conceitos, classes e relacionamentos envolvidos na tarefa de sintonia fina; e a ontologia de tarefa, definindo como cada heurística de sintonia fina deve ser executada.

Ontologia de sintonia fina

A ontologia, que pode ser utilizada por um DBA ou aplicação, poderia inferir as ações de sintonia fina para criação, remoção e reindexação de índices em bancos de dados relacionais. A ontologia se divide em duas partes: a primeira está relacionada ao domínio, que tem como principal função descrever os elementos envolvidos durante o processo de sintonia fina e a segunda se relaciona com a tarefa, que possui heurísticas capazes de analisar a carga de trabalho instanciada na ontologia e inferir quais são as ações que poderiam ser tomadas nesse contexto [Almeida, 2013].

O uso da ontologia proposta pode contribuir com a melhora na eficácia da prática de sintonia fina. Desse modo, a introdução da ontologia ao processo de sintonia fina de banco oferece mais transparência e confiabilidade sobre qual ação adotar mediante diferentes situações enfrentadas pelo SGBD. Portanto, a ontologia permite a criação automática de novas práticas de sintonia, que são inferidas a partir de práticas já existentes. Além de permitir a combinação de heurísticas por meio da ontologia de aplicação.

A primeira versão desenvolvida por [Almeida, 2013] focava na estrutura de índices de banco de dados. Este fato ocasionou a criação de duas heurísticas na ontologia de tarefa sendo elas: (i) a criação de índices hipotéticos baseando-se na carga de trabalho; e, (ii) dentre os índices hipotéticos criados, quais deles podem ser índices persistidos no banco de dados.

Na dissertação de mestrado do Rafael Pereira de Oliveira [Oliveira, 2015] é realizada uma extensão, consequentemente, uma nova versão dessa ontologia. [Oliveira, 2015] introduziu na ontologia a capacidade de inferir sobre a seleção e criação de visões materializadas. Dessa forma, a ontologia de domínio foi extendida para inclusão dos conceitos envolvidos na tarefa de seleção e criação de visões materializadas. Também na ontologia de tarefas foram adicionadas três novas heurísticas de sintonia: (i) heurística para criação de visão materializada hipotéticas; (ii) e (iii) para tomada de decisão sobre quais das visões materializadas hipotéticas poderia persistir no banco de dados.

2.8
Conclusão

Nesse capitulo foram apresentados conceitos para fundamentar e contextualizar a pesquisa apresentada. Foram apresentados conceitos de gerenciamento de banco de dados na nuvem, bancos de dados como serviço apresentando suas arquiteturas, vantagens e desvantagens; bancos de dados multi inquilinos com modelos e análise dos mesmos. Além da apresentação de sintonia fina tradicional e automática e a atividade de live migration, que será considerada como atividade de sintonia fina.

No capítulo 3 são descritas pesquisas de sintonia fina realizadas no grupo de pesquisa BioBD da PUC e demais trabalhos relacionados.

3
Trabalhos Relacionados

Neste capítulo será apresentada as principais pesquisas de sintonia fina do grupo de pesquisa BioBD da PUC, além das pesquisas em bancos de dados na nuvem.

3.1
Auto-sintonia fina do grupo BioBD.

O BioBD é o grupo de pesquisas em sintonia fina automática e semiautomática da PUC-Rio, cuja orientação é feita pelo professor Sérgio Lifschitz. Com o objetivo de realizar sintonia fina sem que haja mudanças no nível de hardware, diversos trabalhos realizados pelo grupo fazem uso de estruturas de acesso, como índices e visões materializadas, além de técnicas e ferramentas de engenharia de software na construção de sistemas, como agentes para sintonia e o uso de ontologias especificas, dentre outras formas de alcançar o mesmo objetivo.

Desde 2002, muitos trabalhos foram realizados. Ressalta-se aqui o trabalho do Marcos Antônio Vaz Salles, apresentado em 2004 [Salles, 2004], onde foram propostas duas arquiteturas que permitem automatizar a sintonia de índices, além de ter sido exposta, pela primeira vez, a heurística de benefícios. Uma das arquiteturas de índice foi implementada no SGBD PostgreSQL versão 7 e os resultados do seu uso foram analisados utilizando a carga de trabalho transacional TPCC15. Destaca-se, ainda, que o professor Rogério Luís de Carvalho Costa em seu trabalho [Costa et al, 2005] foi o precursor do desenvolvimento das ideias envolvendo a heurística de benefícios.

José Maria da Silva Filho em 2008 [Monteiro, 2008] adotou em seu trabalho uma abordagem não intrusiva, fazendo uso de índices no projeto físico do banco de dados para manutenção automática e on-the-fly. De acordo com [Monteiro, 2008], a partir de uma abordagem generalista, pode-se realizar a manutenção automática de índices e de clusters alternativos de dados sem estar vinculado a um único SGBD.

3.2
Ontologia e sintonia fina

No ano de 2013, Ana Carolina Brito de Almeida [Almeida, 2013], em sua tese de doutorado, reconheceu o uso de ontologia para a sintonia fina (automática ou não), formalizando decisões e inferências relativas à atividades de sintonia fina, principalmente, à seleção, criação e manutenção de índices. Com o uso da ontologia pode-se oferecer transparência e confiabilidade acerca das alternativas disponíveis para possíveis cenários no SGBD, além de respaldar as decisões que foram tomadas. Ainda em seu trabalho, foi proposto um framework chamado Outer-Tuning que executaria as heurísticas de sintonia fina por intermédio da ontologia apresentada.

Rafael Pereira de Oliveira [Oliveira, 2015] em seu trabalho de mestrado realizou o projeto e implementação do framework Outer-Tuning, definindo sua arquitetura de software para atender a requisitos específicos. Tanto a ontologia de domínio quanto a ontologia de tarefa foram estendidas por [Oliveira, 2015] para contemplar soluções de sintonia fina a partir do uso de visões materializadas. Com isso, foi possível propor o uso das heurísticas para realizar atividades de sintonia fina tanto para índice quanto para visão materializada.

Nessa pesquisa será proposta uma nova extensão da ontologia de domínio desenvolvida e aprimorada por [Almeida, 2013] e [Oliveira, 2015], a fim de abranger a sintonia fina na nuvem.

3.3
Framework para assegurar a QoS de banco de dados multi-inquilinos

Leonardo Oliveira Moreira [Moreira, 2014] em sua tese de doutorado propõem a utilização de técnicas de predição para garantir a qualidade de serviço (QoS) respeitando o nível de serviço acordado (SLA) entre o usuário do serviço e o provedor.

Por tanto, em sua tese foi proposto e elaborado o framework Performace Multi-tenant Databases (PMDB). Este adota o modelo multi-inquilino de SGBD compartilhado. Utiliza a técnica de live migration que garante a migração de banco de dados quase que imediata e com o mínimo de interrupção do serviço de banco de dados.

Neste trabalho, propomos como sintonia fina em banco de dados na nuvem a utilização de migração de bancos de dados (live migration) caso os mesmos estejam perdendo performance.

3.4 Conclusão

Neste capítulo foram apresentadas as pesquisas que serão utilizadas como base para o desenvolvimento dessa dissertação. Foi exposto as principais pesquisas do grupo de pesquisa BioBD da PUC-Rio. Em seguindo das pesquisas que consideravam o uso da ontologia de sintonia fina na automatização da sintonia fina de banco de dados. E por fim, apresentou-se o framework responsável por realizar a live migration de maneira preventiva de bancos de dados na nuvem.

No capítulo 4 será realizado o estudo da sintonia fina em banco de dados tradicionais e na nuvem.

4
Sintonia fina de banco de dados na nuvem

As pesquisas em banco de dados tradicionais têm como foco a criação de sistemas de auto-sintonia como: (i) a criação automática de índices; (ii) as visões materializadas ou criação de estatísticas; (iii) além de modificações em parâmetros de configuração com o objetivo de obter o melhor desempenho das consultas. No entanto, essas questões não se alteram quando o banco de dados está localizado em uma plataforma de bancos de dados como serviço [Mukerjee et al, 2011].

Em seu artigo, [Mukerjee et al, 2011] afirma que o custo exerce influência no ajuste de desempenho, de modo automático ou não. Portanto, para a efetivação da auto-sintonia na computação em nuvem, o ambiente, no qual o banco de dados está inserido, sofre um acréscimo de mais um parâmetro a ser levado em consideração, qual seja o custo. Por exemplo, ao se criar um índice automaticamente, este ocupa espaço em disco, por menor que seja, podendo ocasionar a mudança do plano contratado pelo usuário do serviço de nuvem. Por conseguinte, para alguns autores, o incremento de itens como CPU, memória e disco pode ser considerada como medida de sintonia fina do sistema. Porém, quanto mais o sistema de banco de dados permanecer em sua configuração original, melhor será.

Do ponto de vista do usuário, cujo banco de dados é o inquilino dos serviços da nuvem, existiria incentivos para redução dos custos de prestação de serviço. Dessa forma, promover-se-ia a reavaliação de itens computacionais custosos, como a compilação de consultas com a intenção de reduzir o desempenho gerado pelos planos de execução, bem como a atividade de encontrar planos que tenham um melhor desempenho em um ambiente multi inquilino.

Para o provedor do serviço de nuvem, o importante é realizar ajustes de desempenho quando há elevada utilização do hardware. Além de aplicar correções a um determinado inquilino, sem que esta influencie na queda de desempenho de outros bancos de dados. Por isso, os provedores de serviços de banco de dados devem investir em mecanismos que possibilitem o uso dos equipamentos de maneira mais equilibrada, de modo a garantir que uma sintonia aplicada em um

determinado banco de dados, não possa influenciar negativamente o desempenho de outros sistemas.

Assim, além de pequenos ajustes, que devem ser realizados, o módulo de auto-sintonia deve levar em consideração que o banco de dados está inserido em um servidor que compartilha os seus recursos com outros inquilinos. Dessa forma, o nível de isolamento destes bancos de dados deve ser garantido para que as mudanças realizadas não afetem o desempenho de outros bancos.

Com isso, realizar ajustes de desempenho se torna cada vez mais complexo, principalmente utilizando técnicas de multi-tenancy [Mukerjee et al, 2011], pois, com o hardware partilhado por muitos clientes, isolar o banco de dados com problemas pode tornar-se muito difícil e, se não forem isolados uns dos outros, pode prejudicar o desempenho dos demais. Dessa maneira, o provedor do serviço, não garantirá o cumprimento do SLA acordado.

Para tentar minimizar os efeitos que uma determinada carga de trabalho causa sobre os demais bancos de dados, pesquisadores como [Moreira, 2014], implementaram técnicas de live migration para movimentar esse banco para outra infraestrutura e, assim, garantir o cumprimento dos acordos sobre o nível de serviço.

Ao depositar um banco de dados em uma estrutura de nuvem, além dos fatores de implementação e dos recursos elásticos, outro aspecto importante e que pode influenciar no desempenho de um banco de dados, está relacionado à rede de dados a qual está depositada. Por se tratar de um provedor de serviços de nuvem pública, como Amazon, por exemplo, a comunicação da aplicação com o banco de dados será feita pela Internet. Com isso, deve-se estar atento quanto a conectividade entre os componentes do sistema.

Ao utilizar um banco de dados como serviço, os usuários possuem uma sensação de que seus servidores possuem recursos infinitos (como memória, CPU, disco rígido, etc) e também de que estão em um ambiente de alto poder de processamento, quando se trata de grandes volumes de dados. A atividade de sintonia fina tem como objetivo, além de maximizar o desempenho e minimizar o uso de recursos físicos.

Na nuvem tanto a QoS (qualidade de serviço), o SLA (Service Level Agrrement), quanto a possibilidade de expansão a qualquer momento dos recursos computacionais, tornam-se fatores importantes que devem ser levados em consideração ao se tocar no tema de desempenho de banco de dados como serviço.

Como a tendência da maioria das plataformas de nuvem que se dispõem a hospedar banco de dados e/ou prover recursos de banco de dados como serviço é possuir muitos bancos de dados, torna-se muito importante que a sintonia fina nesse ambiente seja automático ou semi-automático. A seguir serão discutidos os novos desafios de sintonia fina encontrados em um ambiente de nuvem.

4.1
Os novos desafios da auto-sintonia na nuvem

Na computação em nuvem é observada a presença de dois importantes agentes: o provedor de serviço de nuvem e o usuário do serviço. Cada um destes possui interesses distintos quanto à sintonia automática neste ambiente. A seguir será abordada a auto-sintonia na visão de cada um deles.

O provedor de serviço de nuvem

A computação em nuvem promove uma nova experiência para o usuário. Experiência essa, que envolve, além de outros fatores, a possibilidade de se aumentar ou diminuir os recursos utilizados pelos bancos de dados. Com isso, há a impressão de recursos infinitos, do cumprimento do acordo de nível de serviço (SLA), do backup do banco de dados, da administração, da tolerância à falhas, além da não preocupação com a aplicação de upgrade do software que está sendo oferecido como serviço. Como frisado anteriormente, tudo isso ficará a cargo do provedor do serviço de nuvem.

No entanto, no ambiente de nuvem, a tendência é que haja um número incontável de bancos de dados ativos ao mesmo tempo. Este fator torna muito difícil a administração de tais bancos de maneira manual. Sendo assim, o uso de sistemas de auto-sintonia e auto-gerenciamento torna-se indispensável.

A auto-sintonia na visão do provedor de nuvem requer, além da verificação do desempenho de cada um dos bancos de dados, a capacidade de otimização, no sentido de balancear a carga de trabalho de cada um dos servidores físicos responsáveis por sustentar os serviços e bancos de dados lá hospedados e oferecidos.

O paradigma da computação na nuvem, IaaS, que permite ao usuário a criação de máquinas virtuais (MVs) para suportar seus bancos de dados, introduz na sintonia fina um novo conceito que pode influenciar na performance do banco de

dados. Como cada uma das MVs compartilham uma porcentagem dos recursos do servidor físico (CPU, memória e largura de banda de I/O), dependendo das máquinas virtuais que estão no hospedeiro físico, pode ocorrer uma saturação dos recursos e, indiretamente, influenciar na performance de outros inquilinos.

Para isso, investe-se em pesquisas de live migration para, por exemplo, garantir o cumprimento dos acordos de serviços contratados pelos usuários. Ou seja, em caso de um ou vários inquilinos estarem com elevada carga de trabalho, os mesmos podem influenciar na baixa de desempenho dos demais inquilinos que compartilham, junto a eles, o mesmo servidor físico.

Além do uso de técnicas de migração com os bancos ativos, a auto-sintonia executada pelo provedor deve utilizar também técnicas de predição. Em [Moreira, 2014] é proposto o uso de técnicas de predição e aprendizagem de máquinas para prever a sazonalidade da carga de trabalho e, com isso, realizar a migração do banco de dados para outro servidor que esteja com baixa utilização. Dessa forma, conseguiria garantir a qualidade de serviço (QoS) prestada pelo provedor dos serviços de nuvem.

Usuário de nuvem

A sintonia fina na nuvem, de acordo com a visão do usuário possui forte ligação com o paradigma adotado, IaaS, PaaS, SaaS, além de admitir, no processo de sintonia fina (automática ou não), a possibilidade de adicionar e retirar os recursos de hardware.

Na infraestrutura como serviço (IaaS), conforme visto, o provedor do serviço de nuvem disponibiliza para o usuário acesso completo à máquina virtual. Nessa máquina, quando pronta, o único software instalado é o sistema operacional.

Após a obtenção dos acessos ao servidor virtual, o usuário pode instalar o SGBD. Sendo assim, todos os processos de atualização de software, aplicação de correções e backup são realizados pelos próprios usuários. A sintonia fina, neste caso, sofrendo limitação apenas pelo desconhecimento da carga de trabalho do hospedeiro ao qual a VM (Virtual Machine) está localizada. Portanto, ao utilizar um banco de dados como serviço no paradigma IaaS, o usuário pode fazer uso dos mesmos softwares de sintonia fina automáticos ou não que são utilizados em um ambiente tradicional.

Na IaaS, o usuário usufrui os recursos da computação na nuvem, sendo um deles o de adicionar e retirar os recursos utilizados de maneira horizontal ou

vertical. Ou seja, é possível incrementar o número de CPU, a quantidade de memória, os volumes de disco, a fim de suportar a carga de trabalho (vertical). Na horizontal, o usuário pode clonar a mesma máquina virtual e tornar os novos servidores parte integrante do cluster, como é feito no cluster MySQL, por exemplo, em que o novo servidor é adicionado ao cluster como uma instância 'escrava' e essa, por sua vez, após realizado o sincronismo, passa a responder às transações e, consequentemente, a balancear as requisições, de consultas a dados que são aplicados ao sistema de banco de dados. Os comandos de inserção, atualização e eliminação de dados são efetivados na instância master e depois sincronizados nas demais instâncias (horizontal).

Dessa maneira, é possível aumentar o throughput mediante o aumento repentino da carga de trabalho. Ressalta-se também a possibilidade de provisionamento de cópias da máquina de banco de dados de maneira rápida. Essa atividade em um ambiente não virtualizado, ou dentro da estrutura de virtualização de uma empresa, pode ser custoso e demorado. Com a normalização da carga de trabalho, o usuário pode desligar os servidores clonados e voltar a configuração original.

Na PaaS, o usuário está mais limitado. A sintonia fina do banco de dados torna-se limitada às mudanças locais, com a criação de estruturas de auxílio (índices e visões materializadas). As sintonias no âmbito global, como valores de parâmetros, não são permitidas. No entanto, o usuário não se preocuparia com a replicação dos bancos de dados para ser tolerante a falhas, assim como com o backup e a aplicação de atualização, que será realizada pelo provedor do serviço.

Por fim, o usuário ao contratar o serviço de software como serviço, percebe total limitação quanto à sintonia fina do banco de dados. No SQL Server Azure, por exemplo, a Microsoft permite que o usuário conecte ao banco utilizando a ferramenta própria de administração e utilize, de forma limitada, as ferramentas presentes de auxílio a sintonia fina. A microsoft também permitem a utilização, no portal do Azure, da ferramenta de auxílio a criação de índice. A ferramenta semiautomática de sintonia [Microsoft, 2016] fornece ao usuário sugestões para criação de índice, baseado nos comandos SQL de maior custo no banco de dados.

Tanto no PaaS, quanto no SaaS, é permitido que o usuário, mediante a queda de performance, ou para aumentar a vazão dos dados para uma determinada carga de trabalho, realize a migração em tempo real para uma estrutura de cluster cujos

servidores possuem mais recursos físicos. Por exemplo, na nuvem da Oracle, o usuário pode criar o banco de dados em um servidor Exadata Quarter Rack e solicitar sua migração para o Exadata Half Rack [Oracle, 2016].

Para o usuário, a expansão dos recursos de um servidor e a melhora no desempenho da consulta leva ao aumento do custo para ele. Por exemplo, ao aumentar a vazão eleva-se a troca de dados pela rede, o que na nuvem é taxado, além disso, o incremento de recursos leva ao provedor a cobrar pelo tempo de utilização do serviço.

Limites entre provedor do serviço e usuário

Ao realizar a atividade de sintonia fina em um banco de dados esteja ele na nuvem ou não, o maior interessado em possuir um banco de dados com indicadores de performance positivos, é o DBA.

No entanto, na computação na nuvem a figura do DBA, muitas vezes, se confunde com a do usuário autônomo, ou seja, uma pessoa que não possua os mesmo conhecimentos técnicos que um administrador de banco de dados possui. Mas a necessidade de saber e fazer com que o banco de dados esteja com uma boa performance continua sendo do usuário. Além disso, pelo fato da computação na nuvem ter como dogma o pagamento pelo que é consumido, será também do interesse do usuário saber se o que está sendo investido está realmente atendendo ou não às suas necessidades [Sulivan, 2012].

Para o provedor do serviço na nuvem o interesse em realizar sintonia fina dos bancos de dados que ele hospeda está em maximizar o uso dos servidores físicos, além de primar pelo cumprimento do acordo de nível de serviço e garantir que na necessidade de um inquilino escalar seus recursos o mesmo será atendido.

4.2
Sintonia fina na nuvem versus ambientes convencionais.

A tabela 1 tem como objetivo demonstrar as diferenças entre a sintonia fina tradicional e a sintonia fina realizada em um banco de dados como serviço.

As atividades de sintonia fina foram divididas em 3 (três) categorias: (i) banco de dados, que contempla as atividades realizadas no banco de dados ou esquema, como: (a) a criação de estruturas auxiliares (índice completos e parciais, visão materializada); (b) particionamento de tabelas (vertical e horizontal); (c) reescrita de consulta; e (d) live migration do banco de dados; (ii) as atividades que podem

ser feitas no sistema gerenciador de banco de dados, como: (a) distribuição de banco de dados; (b) balanceamento do I/O; (c) Otimização do plano de execução; (d) gerenciamento de buffer (cache); (c) gerenciamento de shared-pool; (d) configuração de parâmetros e (iii) as novas atividades de sintonia fina no ambiente de nuvem, como: (a) escalar verticalmente (por exemplo, CPU, memória); (b) escalar horizontalmente (por exemplo, master-slave); (c) Live migration da máquina virtual.

Conforme verificado anteriormente, a partir do paradigma adotado, além do provedor do serviço de nuvem, a atividade de sintonia fina pode ser realizada pelo usuário, pelo provedor ou por ambos.

Conclui-se, portanto, que ao adotar o modelo IaaS não foi identificado mudanças na atividade de sintonia fina. No paradigma de PaaS, para alguns provedores do serviço, o usuário está limitado a sintonias finas de banco de dados, para outros lhe é permitido realizar sintonia a nível de banco de dados, SGBD e nuvem. No SaaS o usuário pode realizar sintonia fina apenas de banco de dados e otimização do plano de execução.

Sintonia Fina	Tradicional	IaaS	PaaS	SaaS
BD	Índices completos	Usuário [5]	Usuário[5]	Usuário[5]
	Índices parciais	Usuário [5]	Usuário[5]	Usuário[5]
	Particionamento vertical	Usuário [5]	Usuário[5]	Usuário[5]
	Particionamento horizontal	Usuário [5]	Usuário[5]	Usuário[5]
	Reescrita de consulta	Usuário [5]	Usuário[5]	Usuário[5]
	Live Migration (BD)	Usuário [5]	Usuário[5] Provedor[1],[2],[3]	Provedor[5]
SGBD	Distribuição	Usuário [5]	-	-
	Balanceamento de I/O	Usuário [5]	Usuário[4] Provedor[1],[2],[3]	Provedor[5]
	Otimização plano de execução	Usuário [5]	Usuário [5]	Usuário [5]
	Gerenciamento de Buffer (cache)	Usuário [5]	Usuário[4] Provedor [1],[2],[3]	Provedor [5]
	Gerenciamento de shared-pool	Usuário [5]	Usuário[4] Provedor[1],[2],[3]	Provedor [5]
	Configuração de parâmentros	Usuário [5]	Usuário[4] Provedor[1],[2],[3]	Provedor [5]
Nuvem	Escalar verticalmente	Usuário [5]	Usuário[4] Provedor[1],[2],[3]	Usuário [5]
	Escalar horizontalmente	Usuário [5]	Provedor [5]	-
	Live Migration (VM)	Usuário[5] Provedor [5]	Provedor [5]	-

Tabela 1 Quadro comparativo entre sintonia fina de banco de dados tradicional e na nuvem.

[1] Amazon: https://aws.amazon.com

[2] Azure: https://azure.microsoft.com

[3] Google cloud: https://cloud.google.com

[4] Oracle Cloud: https://cloud.oracle.com

[5] Todos

4.3
DBA ainda é necessário?

Os serviços de bancos de dados na nuvem, oferecidos por diferentes provedores e sendo cada vez mais necessários sistemas automático de sintonia fina, os quais visam cada vez mais tornar a administração dos bancos transparentes e fáceis, é natural que a necessidade do administrador de banco de dados (DBA) seja questionada.

Para [Sulivan, 2012] ainda é necessário o DBA por mais que os serviços de banco de dados na nuvem venham a aliviar algumas tarefas que tradicionalmente cabia ao administrador do banco de dados.

Dentre as atividades que o DBA exerce nos ambientes tradicionais podem ser destacado: (a) dimensionar os novos bancos de dados, (b) instalar e configurar os sistemas de gerenciamento de dados, (c) verificar as novas versões dos mesmos e (d) identificar e retirar falhas de instalação, além de (e) determinar a melhor configuração do tamanho do bloco para leitura e gravação, (f) monitorar a performance, (g) identificar as consultas que possuem problemas e (h) garantir a segurança dos dados.

Quando o banco de dados é migrado para a plataforma da computação em nuvem, a tarefa de dimensionar o banco pode ser sucumbida por ser permitido incrementar a infraestrutura quando necessário. Assim como as atividades de instalação e manutenção do SGBD passa a ser responsabilidade do provedor do serviço de nuvem.

O DBA será cada vez mais necessário para atividades de sintonia fina, ou seja, as atividades serão cada vez menores de configuração e focado mais na modelagem física do banco e sintonia fina de consultas. Administradores de banco de dados podem identificar consultas de longa duração e identificar possíveis problemas de desempenho.

Além disto, novos bancos de dados com novas formas de modelagem de dados como por exemplo, bancos de dados orientados a objetos e bancos de dados analíticos, estão sendo cada vez mais utilizados. Seu uso não é para substituir o modelo relacional, mas sim complementar. Desse modo, o DBA com seu forte conhecimento da modelagem relacional passa a auxiliar o entendimento de quando e onde utilizar essas alternativas de armazenamento.

Portanto, mover o banco de dados para a nuvem não retira muitas das tarefas dos DBAs. Seus conhecimentos serão necessários para contribuir com: (i) a sintonia fina de consultas e modelagem física do banco, (ii) desenvolver aplicações que escalem e (iii) entender como utilizar alternativas de armazenamento de dados.

4.4
Conclusão

Nesse capítulo foram levantados os novos desafios da sintonia fina em bancos de dados como serviço. Apresentando um quadro comparativo das possíveis atividades realizadas pelos provedores do serviço e pelo usuário. Foi levantado o questionamento dos papéis de usuário e provedor e seus interesses quanto a sintonia fina de banco de dados. Por fim, levantamos a necessidade do administrador de banco de dados (DBA) quando adotado o banco de dados na nuvem.

O capítulo 5 será apresentado, como mais uma contribuição desse trabalho, a extensão da ontologia de sintonia fina, elaborada por [Almeida, 2013] e estendida por [Oliveira, 2015], para abranger a sintonia fina automática de bancos de dados como serviço.

5
Ontologia de sintonia fina em banco de dados como serviço.

Neste capítulo será apresentada a extensão da ontologia de sintonia fina elaborada por [Almeida, 2013] e estendida por [Oliveira, 2015], para abranger a sintonia fina automática de bancos de dados como serviço.

Extensão da ontologia de domínio

A extensão da ontologia de domínio proposta em [Oliveira, 2015], insere novos conceitos, atributos e relações. A figura abaixo exibe a referida ontologia de sintonia fina e a proposta de extensão a fim de contemplar a sintonia fina em banco de dados na nuvem.

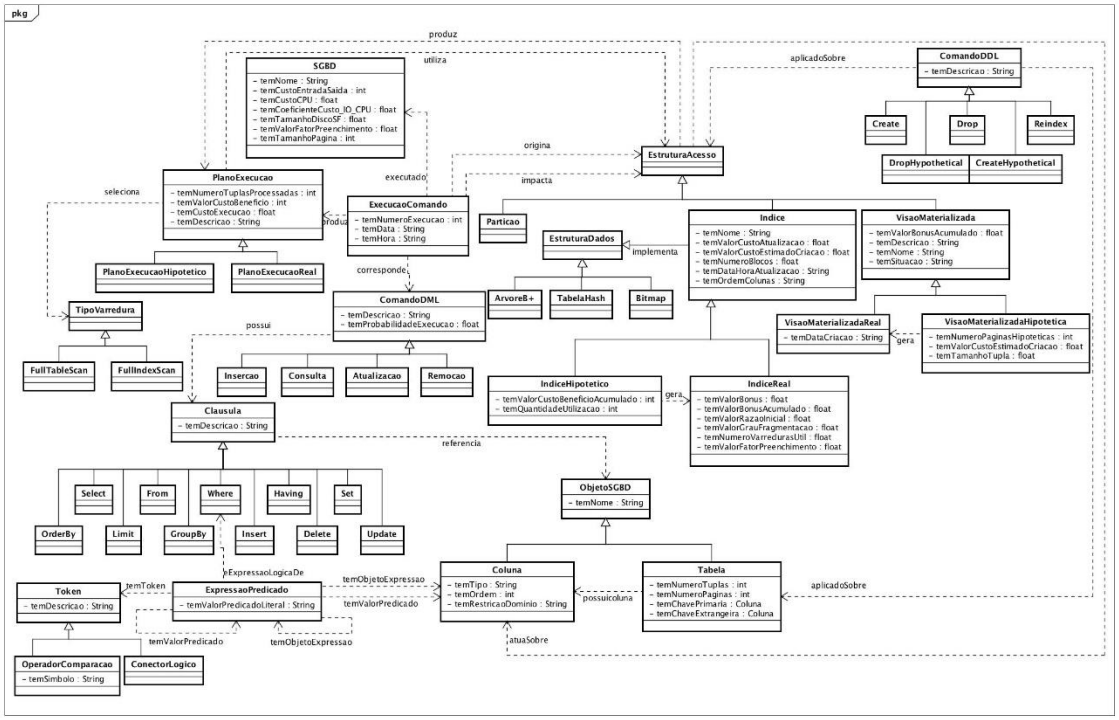

Figura 12 Modelo da ontologia

A extensão da ontologia de domínio proposta em [Oliveira, 2015], insere novos conceitos, atributos e relações. A figura abaixo exibe a referida ontologia estendida a fim de contemplar a sintonia fina com a qualidade de experiência em bancos de dados na nuvem.

Para representar o ambiente do banco de dados na nuvem, foram criados os conceitos: PlanoPagamentoCloud e InstanciaComputacional cujas especializações são ModeloIaaS, ModeloPaaS e ModeloSaaS (Figura 12). Esses conceitos foram introduzidos para que a ontologia de tarefa possa manipular indivíduos desses conceitos na durante a execução de novas heurísticas.

Foram criados na ontologia de domínio os conceitos PlanoPagamentoCloud, InstanciaComputacional, ModeloIaaS, ModeloPaaS e ModeloSaaS (Figura 13) com objetivo de instanciar os conceitos necessários para a tarefa de sintonia fina no banco de dados como serviço nuvem. Para permitir mapear na ontologia o plano de pagamento contratado pelo usuário assim com sua capacidade computacional foram criados os conceitos PlanoPagamentoCloud e InstanciaComputacional. A instancia computacional por sua vez, é especializada nos conceitos ModeloIaaS que representa os bancos de dados que utilizam o modelo de infraestrutura como serviço; Modelo PaaS que representa o conceito dos bancos de dados que utilizam o modelo de plataforma como serviço; e ModeloSaaS que representa o conceito de banco de dados que utilizam o modelo software como serviço. A herança entre os conceitos dos modelos computacionais e o conceito da instancia computacional se justifica pelos atributos temDisco, temCPU, temMemoria, já que todos os modelos possuem essas mesmas informações. Os atributos de cada conceito estão descritos nas tabelas:

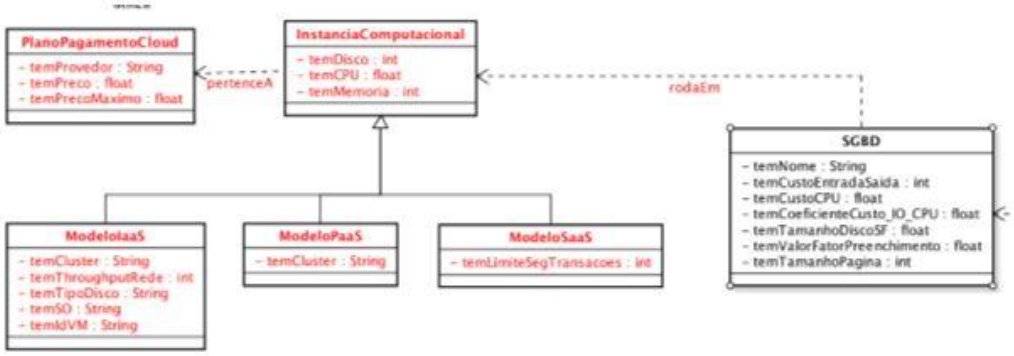

Figura 13 Extensão da ontologia

PlanoPagamentoCloud	
temProvedor	Armazena o nome do provedor do serviço de nuvem.
temPreco	Armazena o preço do plano contratado.
temPrecoMaximo	Armazena o preço máximo aceito pelo usuário para extensão do plano.

Tabela 2 PlanoPagamentoCloud

InstanciaComputacional	
temDisco	Armazena o tamanho do disco disponível para o banco de dados.
temCPU	Armazena o número de CPU disponível para o banco de dados.
temMemória	Armazena a quantidade de memória disponível para o banco de dados.

Tabela 3 InstanciaComputacional

ModeloIaaS	
temCluster	Armazena o nome do cluster que a infraestrutura está localizada
temThroughputRede	Armazena o throughput de rede contratado
temTipoDisco	Armazena o tipo do disco que está disponibilizado para a infraestrutura (por exemplo, SSD,HDD)
temSO	Nome do Sistema Operacional que está sendo executado na máquina virtual (MV)
temIdVM	Identificador de máquina virtual utilizado para distinguir quais MVs o banco de dados está distribuído

Tabela 4 ModeloIaaS

ModeloPaaS	
temCluster	Armazena o nome do cluster onde a plataforma está localizada

Tabela 5 ModeloPaaS

ModeloSaaS	
temLimiteSegTransacoes	Armazena o limite máximo de transações por segundo que podem ser realizadas.

Tabela 6 ModeloSaaS

Como mostra a figura 12, foram inseridos dois novos relacionamentos: (i) rodaEm que determina o relacionamento entre os conceitos SGBD e InstanciaComputacional e como nome sugere determina que um SGBD roda em uma ou mais InstanciasComputacionais. (ii) pertenceA que determina o relacionamento entre os conceitos PlanoPagamentoCloud e InstanciaComputacional e como sugerido pela relação uma InstanciaComputacional está relacionada com um PlanoPagamentoCloud.

6
Conclusão

Como visto, a computação na nuvem vem modificando a experiência dos usuários com relação às soluções tecnológicas. Os serviços de gestão e armazenamento de dados em nuvem passaram a ser extremamente atrativos [Curino et al., 2011], seja pela redução dos custos operacionais ou pela expectativa de um banco de dados elástico e escalável quando a carga de trabalho está elevada. Porém, os SGBDs eram considerados não compatíveis com a computação na nuvem, pelo fato de não escalar muito facilmente. Dessa maneira, é crescente o número de estudos que apresentam como objetivo a adequação dos sistemas gerenciadores de banco de dados a essa nova tecnologia.

Sendo assim, provedores de serviço na nuvem têm investido cada dia mais na gestão e armazenamento de dados, de modo a otimizar o uso dos servidores que hospedam os bancos de dados pelo uso das técnicas de multi-inquilinos, além de cumprir com o SLA contratado pelo usuário e permitir ao usuário a possibilidade de escalar o serviço contratado.

Conforme visto, a inserção do sistema de banco de dados na nuvem acrescentou novos paradigmas à sintonia fina destes sistemas, como a possibilidade de adição e remoção tanto de recursos (CPU, memória) quanto a criação de réplicas, de forma rápida, para balancear a carga de trabalho. Esses são exemplos de novas atividades que podem ser adotadas pelos sistemas que automatizam a sintonia de bancos de dados na nuvem, uma vez que as atividades de sintonia fina tradicionais não sofrem influência.

6.1
Contribuições

Esse trabalho teve como contribuição apontar novos paradigmas de sintonia fina de banco de dados como serviço na nuvem. Além disso, o presente trabalho propôs-se a estender a ontologia de sintonia fina desenvolvida pelo grupo de

pesquisa BioBD da PUC-Rio, para contemplar os sistemas de banco de dados como serviço de nuvem.

6.2
Limitações

Por se tratar de uma tecnologia muito recente, foram identificadas poucas pesquisas sobre sintonia fina e auto sintonia fina em banco de dados na nuvem. Os sistemas de gerenciamento de banco de dados na nuvem ainda estão em processo de adaptação a esse novo ambiente. Portanto, as pesquisas que envolvem segurança, formas de armazenamento e migração automática de bancos de dados foram mais exploradas.

Desse modo, neste trabalho, buscou-se, de forma pioneira, reunir informações sobre os novos paradigmas da sintonia fina e, especificamente, de sintonia fina em bancos de dados na nuvem.

6.3
Trabalhos Futuros

- Expandir a ontologia de tarefa que é utilizada na ferramenta Outer-tuning, para que incorpore ações pertinentes à sintonia fina na nuvem.
- Incluir ao Outer-tuning a possibilidade de criar réplicas do banco de dados para suprir a carga de trabalho de um banco de dados na nuvem, a fim de utilizar o processo de live migration do banco.
- Criar as heurísticas de sintonia fina para o ambiente de nuvem e que utilize a extensão da qualidade de experiência.

7
Referências bibliográfica

Abadi, D. J. (2009). Data management in the cloud: Limitations and opportunities. *IEEE Data Eng. Bull.*, 32(1):3-12.

AGRAWAL, D. et al. Database scalability, elasticity, and autonomy in the cloud (Extended abstract). Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v. 6587 LNCS, n. PART 1, p. 2–15, 2011.

Agrawal, D., Das, S., and El Abbadi, A. (2012). Data Management in the Cloud: Challenges and Opportunities. Synthesis Lectures on Data Management. Morgan & Claypool Publishers.

Alashqur, A. M., Su, S. Y. W., and Lam, H. (1989). Oql: a query language for manipulating object-oriented databases. In Proceedings of the 15th international conference on Very large databases, VLDB '89, pages 433–442, San Fran- cisco, CA, USA. Morgan Kaufmann Publishers Inc.

ALMEIDA, A. C. B. D..Framework para apoiar a sintonia fina de banco de dados. PhD thesis, Pontifícia Universidade Católica do Rio de Janeiro - PUC-RIO, 2013. (Document), 1.2, 1.3, 2.10, 3.1, 3.3, 3.4, 4, 4.1, 4.2.1, 4.3, 4.2.1, 4.2.2, 5.1, 7.2.2

Amazon Web Services (2016). Amazon web services (aws). Dispon´ıvel em: http://aws.amazon.com/, acessado em 17/01/2016.

Astrahan, M. M. and Chamberlin, D. D. (1975). Im- plementation of a structured english query language. Commun. ACM, 18:580–588.

Windows Azure: Cloud Computing — Cloud Services — Cloud Application Development. Dispon´ıvel em: http://www.windowsazure.com/en-us/, acessado em 17/01/2016.

Babcock, B. and Chaudhuri, S. (2005). Towards a robust query optimizer: a principled and practical approach. In Proceedings of the 2005 ACM SIGMOD international conference on Management of data, SIGMOD '05, pages 119– 130, New York, NY, USA. ACM

Baker, J., Bond, C., Corbett, J. C., Furman, J., Khorlin, A., Larson, J., Leon, J.-M., Li, Y., Lloyd, A., and Yushprakh, V. (2011). Megastore: Providing scal- able, highly available storage for interactive services. In Proceedings of the Conference on Innovative Data system Research (CIDR), pages 223–234.

Bennett, K. (1995). Legacy systems: Coping with success. IEEE Softw., 12(1):19–23.

Bini, T. A., Lange, A., Suny´e, M. S., Silva, F., and Almeida, E. C. d. (2011). Non-exhaustive Join Ordering Search Algorithms for LJQO. In ICEIS (1), pages 151–156.

Brodie, M. L. and Stonebraker, M. (1995). Migrating Legacy Systems: Gateways, Interfaces, and the Incremental Approach. Morgan Kauf- mann.

Cattell, R. (2011). Scalable sql and nosql data stores. SIGMOD Rec., 39(4):12–27.

Cheng et al., 2016. A Survey of Cloud Databases. Disponível em: http://www.hurui.info/files/A%20Survey%20of%20Cloud%20Databases.pdf, acessado 17/01/2016.

Clustrix (2016). Clustrix: Speed. Scale. Simplicity. Disponível em: http://www.clustrix.com/, acessado em 17/01/2016.

Codd, E. F. (1970). A relational model of data for large shared data banks. Commun. ACM, 13:377–387.

Codd, E. F. (1972). Relational completeness of data base sublanguages. In: R. Rustin (ed.): Database Systems: 65-98, Prentice Hall and IBM Research Report RJ 987, San Jose, California.

Costa, R. L. C. C. (2009), Quality of Experience in Database Systems. PhD Thesis submitted to the University of Coimbra, Portugal.

Curino, C., Jones, E., Popa, R. A., Malviya, N., Wu, E., Madden, S., Balakrishnan, H., and Zeldovich, N. (2011). Relational cloud: a database service for the cloud. In CIDR, pages 235–240.

Das, S., Agrawal D., Abbadi, A. E., (2013). ElasTraS: An Elastic, Scalable, and Self-Managing Transactional Database for the Cloud. ACM Trans. Database Syst., 38, Pg 5:1-5:45

Delimitrou, C., Sankar, S., Khessib, B., Vaid, K., and Kozyrakis, C. (2012). Time and cost-efficient modeling and generation of large-scale tpcc/tpce/tpch workloads. In Proceedings of the Third TPC Technology conference on Topics in Performance Evaluation, Measurement and Characterization, TPCTC'11, pages 146–162, Berlin, Heidelberg. Springer-Verlag.

Floratou, A., Teletia, N., DeWitt, D. J., Patel, J. M., and Zhang, D. (2012). Can the elephants handle the nosql onslaught? Proc. VLDB Endow. 5(12):1712–1723.

Garcia-Molina, H., Ullman, J. D., and Widom, J. (2008). Database Systems: The Complete Book, chapter The Query Compiler, pages 759–841. Prentice Hall Press, Upper Saddle River, NJ, USA, 2 edition.

Gulati, A., Kumar, C., and Ahmad, I. (2010). Modeling workloads and devices for IO load balancing in virtualized environments. SIGMETRICS Perform. Eval. Rev., 37(3):61–66.

Gupta, D., Cherkasova, L., Gardner, R., and Vahdat, A. (2006). Enforcing performance isolation across virtual machines in Xen. In Proceedings of the ACM/IFIP/USENIX 2006 International Conference on Middleware, Middleware '06, pages 342–362, New York, NY, USA. Springer-Verlag New York, Inc.

Han, J., Du, J., Survey on NoSQL Database, Pervasive Computing and Applications (ICPCA), 2011, 6th International Conference, Port Elizabeth,26-28 Oct. 2011, pages.363-366.

Harizopoulos, S., Abadi, D. J., Madden, S., and Stonebraker, M. (2008). Oltp through the looking glass, and what we found there. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, SIGMOD '08, pages 981–992, New York, NY, USA. ACM.

Hsu, W. W., Smith, A. J., and Young, H. C. (2001). I/o reference behavior of production database workloads and the tpc benchmarks an analysis at the logical level. ACM Trans. Database Syst., 26(1):96–143

Kossmann, D., Kraska, T., &Loesing, S. (2010, June). An evaluation of alternative architectures for transaction processing in the cloud.In Proceedings of the 2010 international conference on Management of data (pp. 579-590). ACM

Lakshman, A., Malik, P. 2009, Cassandra-A Structured Storage System on a P2P Network, http://cassandra.apache.org/

Lange, A. (2010). Uma avaliação de algoritmos não exaustivos para a otimização de junções. Dissertação de mestrado, Departamento de Informa´tica, UFPR.

Lazarov, V. (2007). Comparison of Different Implementations of Multi- Tenant Databases. PhD thesis, Technische Universit at Munchen.

Lifschitz, S., Oliveira, R., P., Haeusler, E., H., Almeida, A., C. 2015, Projeto e implementação do framework Outer-tuning: auto sintonia e ontologia para bancos de dados relacionais, XI Brazilian Symposium on information System, SBSI, Goiânia, GO, maio 26-29,2015.

Microsoft, 2016., http://www.microsoft.com acessado em: 17/01/2016

Milanés, A. Y., Lifschitz S., Salles, M. A. V. Estado da Arte em Auto-sintonia de Sistemas de Banco de Dados Relacionais, Monografia em ciência de computaçãoo, MCC, 2004, PUC-RIO.

MongoDB, 2016., https://www.mongodb.org/ acessado em 17/01/2016

Moreira, L. O., Abordagem para qualidade de serviço em banco de dados multi-inquilinos em nuvem, tese de doutorado, Universidade Federal do Ceará, UFC, 2014.

MySQL Cluster (2016). MySQL: MySQL Cluster CGE. Disponıvel em: http://www.mysql.com/products/cluster/, acessado em 17/01/2016.

MySql (2016). MySQL: The world's most popular open source database. Disponıvel em: http://www.mysql.com, acessado em 17/01/2016.

Mukerjee, K., Talius, T., Kalhan, A., Ellis, N., Cunningham, C., SQL Azure as a Self-Managing Databas Service: Lessons Learned and Challenges Ahead, IEEE, 2011.

Narayanan, D., Thereska, E., and Ailamaki, A. (2005). Contin- uous resource monitoring for self-predicting dbms. In Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, MASCOTS '05, pages 239–248, Washington, DC, USA. IEEE Computer Society.

Oliveria, R., P., Sintonia fina baseada em ontologia: o caso de visões materializadas. Dissertação de mestrado, Rio de Janeiro, 2015, pontifícia universidade do Rio de janeiro, PUC-RIO.

Oralce, 2016., http://www.oracle.com acesssado em 17/01/2016

Ramakrishnan, R. and Gehrke, J. (2008a). Sistema de Banco de Dados, chapter Bancos de Dados Paralelos e Distribu´ıdos, pages 604–636. McGraw-Hill.

Ramakrishnan, R. and Gehrke, J. (2008b). Sistema de Banco de Dados, chapter A´lgebra e Ca´lculos Rela´cionais, pages 83–107. McGraw- Hill.

Ramakrishnan, R. and Gehrke, J. (2008c). Sistema de Banco de Dados, chapter Um Otimizador de Consultas Relacionais T´ıpico, pages 399–431. McGraw-Hill.

Schiller, O., Schiller, B., Brodt, A., and Mitschang, B. (2011). Na- tive support of multi-tenancy in rdbms for software as a service. In EDBT, pages 117–128.

Shasha, D. and Bonnet, P. (2002). Database Tuning: Princi- ples, Experiments, and Troubleshooting Techniques. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science.

Silberschatz, A., Korth, H., and Sudarshan, S. (2010a). Database Systems Concepts, chapter Concurrency Control, pages 661–720. McGraw- Hill, Inc., New York, NY, USA, 6 edition.

Silberschatz, A., Korth, H., and Sudarshan, S. (2010b). Database Systems Concepts, chapter Query Processing, pages 537–577. McGraw-Hill, Inc., New York, NY, USA, 6 edition.

Snehal B. Shende., Prajakta P. Chapke, (2015). Cloud Database Management System (CDBMS). In Compusoft An internetional jornal of Advanced Computer Technology, Volume-IV, Janeiro., 2015.

Soror, A. A., Aboulnaga, A., and Salem, K. (2007). Database virtu- alization: A new frontier for database tuning and physical design. In Proceedings of the 2007 IEEE 23rd International Conference on Data Engineering Workshop, ICDEW '07, pages 388–394, Washington, DC, USA. IEEE Computer Society.

Soror, A. A., Minhas, U. F., Aboulnaga, A., Salem, K., Kokosielis, P., and Kamath, S. (2008). Automatic virtual machine configuration for database workloads. In Proceedings of the 2008 ACM SIGMOD international conference on Management of data, SIGMOD '08, pages 953–966, New York, NY, USA. ACM.

Sousa, F. R. C., Moreira, L. O., Macˆedo, J. A. F. d., and Machado, J. C. (2011). Gerenciamento de Dados em Nuvem: Conceitos, Sistemas e Desafios. Technical report, Universidade Federal do Cear´a - UFC.

Stonebraker, M. (2010). SQL databases v. NoSQL databases. Communications of the ACM, 53(4):10.

Sullivan, D. G., Seltzer, M. I., and Pfeffer, A. (2004). Using probabilistic reasoning to automate software tuning. SIGMETRICS Perform. Eval. Rev., 32(1):404–405.

Sullivan, D., Do you need a DBA for Databases in the cloud, Tom`sIT Pro, 2012. http://www.tomsitpro.com/articles/dba-database_administrator-hadoop-PaaS-cloud_computing,2-300.html Acessado em 17/01/2016

VoltDB (2016). VoltDB: Lightning Fast, Rock Solid. Disponıvel em: voltdb.com/, acessado em 17/01/2016.