OLTPBenchAdmin: Uma Ferramenta para o Gerenciamento Distribuído do OLTPBenchmark

Bruno P. Sabino[1], **Leonardo O. Moreira**[1], **Emanuel F. Coutinho**[1]

[1]Instituto Universidade Virtual (UFC Virtual)
Universidade Federal do Ceará (UFC) – Fortaleza, CE – Brasil

bruno.p.sabino@gmail.com, {leoomoreira,emanuel}@virtual.ufc.br

Abstract. *Performance evaluation in database systems are important, because in general, most of the time spent by the applications is associated with processing in the DBMS. In this sense, there are many benchmarks to achieve such performance experiments in various scenarios. OLTPBenchmark is a framework that provides various databases schemes as ways of testing in centralized databases. However, it was designed for centralized environment and its configuration is quite complex and involves expertise. This work aims to propose a tool that simplifies the handling of OLTPBenchmark while allowing the execution of distributed experiments.*

Resumo. *Avaliação de desempenho em sistemas de bancos de dados são importantes, pois em geral, a maioria do tempo gasto pelas aplicações está relacionado ao processamento no SGBD. Neste sentido, existem diversos benchmarks para realizar tais experimentos de desempenho, em diversos cenários. OLTP-Benchmark é um framework que fornece diversos esquemas de bancos de dados como formas de testes em bancos centralizados. No entanto, ele foi projetado para ambiente centralizado e sua configuração é bastante complexa e envolve conhecimentos específicos. Este trabalho tem como objetivo propor uma ferramenta que simplifique a manipulação do OLTPBenchmark ao mesmo tempo que permite a execução de experimentos distribuídos.*

1. Introdução e Motivação

Um banco de dados pode ser definido como um repositório para uma coleção de arquivos de dados dispostos em uma estrutura que facilita o armazenamento e a recuperação dos dados nestes arquivos [Date 2004]. As aplicações utilizam bancos de dados como um modo de melhorar o desempenho e simplificar o acesso aos dados. Além disso, favorecendo os aspectos de segurança, confiabilidade e integridade dos dados diante de múltiplos acessos simultâneos a grandes volumes de dados. Essas características e vantagens dos bancos de dados são disponibilizadas pelo uso de um Sistema Gerenciador de Banco de Dados (SGBD) que é composto por um conjunto de programas que permite aos usuários criarem e manterem os bancos de dados [Elmasri e Navathe 2005]. Um SGBD pode gerenciar diversos bancos de dados e também é responsável por permitir que as aplicações manipulem os dados existentes no banco de dados.

Várias aplicações, em diversos contextos, são orientadas a dados [Hauser et al. 2010] [Matos et al. 2010] [Agrawal et al. 2011] [Hemel e Visser 2011]. Tais aplicações se servem dos bancos de dados para permitirem manipulações de dados

com as características oferecidas pelos SGBDs. Em [Sousa et al. 2012] foi apresentada uma questão sobre desempenho as aplicações *web* orientadas a dados que utilizam bancos de dados como mecanismo de persistência. Neste estudo, foi enfatizado que a maior parte do tempo de resposta gasto para o processamento das requisições das aplicações *web* está relacionado ao desempenho do SGBD. Com isso, surge a necessidade de uma avaliação de desempenho nos SGBDs e nas estruturas dos bancos de dados, objetivando melhorar o desempenho das aplicações orientadas a dados.

Bancos de dados distribuídos é uma alternativa para melhorar a disponibilidade dos dados, aumentar o desempenho no acesso aos dados de forma concorrente e também tornar as aplicações mais tolerantes a falhas. Pode-se definir um banco distribuído como uma coleção de vários bancos de dados interligados por meio de uma rede de computadores [Ozsu 2007]. Assim como os bancos de dados tradicionais são mantidos por um SGBD, um banco de dados distribuído é mantido por um Sistema Gerenciador de Banco de Dados Distribuído (SGBDD). Um SGBDD é responsável pela distribuição dos dados na rede, realizar consultas/armazenamento em diversos fragmentos/cópias dos dados na rede e lidar com toda a gestão de comunicação e sincronização dos dados na rede. Com os avanços dos sistemas distribuídos e dos meios de comunicação, os bancos de dados distribuídos estão cada vez sendo mais utilizados.

Um *benchmark* é um aplicativo que executa no intuito de avaliar o desempenho de algum recurso computacional [Baumann e Stamerjohanns 2014]. Para isso, um *benchmark* realiza um conjunto de testes sobre tal recurso. Estes testes são orientados pelos conjuntos de entradas, denominadas cargas de trabalho, para utilizá-las no recurso. Além disso, os *benchmarks* fornecem métodos comparativos de desempenho, visando uma melhor eficácia dos testes. Para cada recurso avaliado, existem *benchmarks* apropriados, pois cada um necessita de métricas particulares de avaliação, cargas de trabalho e comparações. No contexto de bancos de dados, resumidamente, as entradas são conjuntos de transações, tamanhos dos dados, taxas de execução das transações e a duração do experimento. Por fim, são expostas informações de desempenho que ilustram a vazão do banco de dados, os tempos de respostas das transações e o número de transações executadas com sucesso e falha.

O OLTPBenchmark [Difallah et al. 2013] é um *framework* para avaliar o desempenho de diferentes sistemas gerenciadores de banco de dados relacionais diante de configurações de cargas de trabalho no processamento de transações em tempo real (OLTP). O *framework* possui diversos *benchmarks* que representam diferentes aplicações, tais como TPC-C, Twitter, YCSB, AuctionMark e Wikipedia. No entanto, o OLTPBenchmark foi projetado apenas para experimentos que envolvam no máximo uma máquina, dificultando sua aplicabilidade em ambientes que os bancos de dados estão distribuídos na rede. Dada essa restrição, alguns trabalhos [Moreira et al. 2014] [Sousa e Machado 2012] tiveram que realizar adaptações do OLTPBenchmark para utilizá-lo para execução de experimentos em ambientes distribuídos. Como a utilização do *framework* OLTPBenchmark exige conhecimentos em comandos Linux e utilização de *scripts*, surge a necessidade de uma estratégia que facilite a execução de experimentos com tal *framework*. Toda a execução e informações geradas pelo OLTPBenchmark é feita por meio de comandos no modo terminal, o que dificulta interpretação do estado da execução dos experimentos.

Neste sentido, uma aplicação gráfica que permitisse a interação com ambientes

distribuídos simplificaria, neste contexto, o processo de manipulação do *framework* e interpretação dos dados resultantes dos experimentos. Além disso, essa aplicação tornaria possível gerenciar e acompanhar experimentos elaborados pelo OLTPBenhmark em diversas máquinas que se comunicam por meio de uma rede. Com isso, permitindo a manipulação de experimentos distribuídos com o OLTPBenchmark.

Este artigo apresenta o OLTPBenchAdmin [Moreira et al. 2016], uma ferramenta para o gerenciamento distribuído do *framework* OLTBenchmark, servindo-se de mecanismos de comunicação distribuída e sincronização. Com isso, criando uma abstração das complexidades do OLTPBenchmark ou mesmo tempo que contribui para se consiga realizar experimentos distribuídos que utilize tal *framework*. Além disso, o OLTPBenchAdmin utiliza uma interface gráfica, o que permite que uma melhor interatividade com o OLTPBenchmark.

O objetivo geral deste trabalho é motivado pelo desenvolvimento de uma aplicação gráfica que permita o gerenciamento distribuído do OLTPBenchmark. Para alcançá-lo, traçamos quatro objetivos específicos: (i) elaborar uma arquitetura, voltada para ambientes distribuídos, para servir como base estrutural de implementação; (ii) especificar as funcionalidades e estratégias de implementação de cada componente arquitetural; (iii) elencar um conjunto de funcionalidades do OLTPBenchmark que serão implementadas pela ferramenta proposta; e (iv) implementar as funcionalidades na ferramenta gráfica. O trabalho está organizado da seguinte forma. O referencial teórico do presente trabalho é mostrado na Seção 2. A Seção 3 apresenta, resumidamente, o OLTPBenchmark. Na Seção 4, são apresentadas as características e funcionalidades da ferramenta proposta. Finalmente, a Seção 5 contém as conclusões e os direcionamentos futuros deste ensaio.

2. Referencial Teórico

Nesta seção serão apresentados os conceitos e conhecimentos necessários para o entendimento do presente trabalho. Para isso, será discutida a avaliação de bancos de dados sob a perspectiva de desempenho. Em seguida, serão mostrados os conceitos e as características de *benchmarks* para uso geral. Por fim, serão apresentados alguns *benchmarks* voltados para bancos de dados.

2.1. Avaliação de Bancos de Dados

Date (2004) e Elmasri e Navathe (2005) dicutem alguns fatores que influenciam no desempenho do banco de dados do ponto de vista dos ambientes centralizados [Date 2004][Elmasri e Navathe 2005]. O fator carga de trabalho define o conjunto de operações que deve ser submetido ao banco de dados. Já o fator vazão indica a capacidade do processamento de operações no banco de dados por unidade de tempo. Um outro fator, chamado de concorrência, denota o número de conexões concorrentes em mesmo banco de dados, onde tais conexões submetem uma carga de trabalho a esse banco de dados. O tamanho do banco de dados é outro fator que pode influenciar no desempenho, pois quanto maior o volume de dados, maior será a complexidade para gerenciá-lo. Outro fator importante está relacionado aos algoritmos ou técnicas de otimização do próprio SGBD. Por exemplo, reescrita de consultas são utilizadas para encontrar uma consulta que produza os mesmos resultados, mas que possua uma complexidade menor. Por fim, os parâmetros de configuração do SGBD são fatores que podem ser ajustados para alterar o comportamento do SGBD para determinados cenários.

No ambiente distribuído existem outros fatores que influenciam no desempenho dos bancos de dados. Nos bancos de dados distribuídos é comum a utilização de técnicas de distribuição de dados na rede para obter um melhor desempenho [Ozsu 2007]. A replicação [Sousa e Machado 2012] é uma técnica que utiliza a redundância dos dados para distribuição na rede. Neste caso, a redundância ajuda tanto no aumento da disponibilidade dos dados quanto no aumento da tolerância a falhas. Portanto, em ambientes replicados, se uma cópia falhar, outras cópias poderão assumir. Na avaliação de desempenho de bancos de dados distribuídos, os fatores de desempenho de rede e quantidade de mensagens de comunicação/sincronização são levados em consideração e podem ter um impacto significativo no desempenho [Ozsu 2007].

2.2. Benchmark

Um *benchmark*, no âmbito computacional, pode ser definido com um conjunto de programas que são executados no intuito de avaliar o desempenho de um dado objeto [Baumann e Stamerjohanns 2014]. Em um típico cenário do uso de algum *benchmark*, existem alguns passos a serem definidos. O primeiro passo é a definição do ambiente experimental de comparação para destacar quais as condições de hardware e software onde a comparação será realizada. Outro passo importante é caracterização da carga computacional a ser aplicada no objeto avaliado. Além disso, em um cenário de comparação com outros objetos, é importante ressaltar também que o mesmo ambiente experimental e a mesma carga de trabalho são utilizados no conjunto de objetos a serem comparados.

Ainda no contexto da computação existem vários *benchmarks* que são utilizados e voltados para características distintas tanto de hardware quanto de software. Em hardware existem *benchmarks* para avaliar arquiteturas de processadores [Reguly et al. 2015], discos rígidos [Skendžić et al. 2016], dispositivos e infraestruturas de rede [Vőneki et al. 2016] [Galpin et al. 2014], etc. Já em software é possível avaliar sistemas operacionais [Chen et al. 2013], linguagens de programação [Cardoso et al. 2013] e programas aplicativos de uso geral.

Existem duas variações ou tipos de *benchmarks* no âmbito computacional. A primeira são os *benchmarks* sintéticos que possuem programas que foram desenvolvidos para simular um determinado comportamento ao objeto de avaliação em um dado cenário. Esses programas possuem interfaces para que se possa configurar determinadas situações, especificar métricas a serem comparadas e até mesmo os valores correspondentes ao que deve ser executado no objeto de avaliação. Já os *benchmarks* de aplicação executam programas do mundo real e oferecem medidas de situação reais ou de execuções não simuladas. Com isso, *benchmarks* de aplicação oferecem dados mais realistas do desempenho real em contraste com o uso de ambientes simulados.

3. OLTPBenchmark

O OLTPBenchmark [Difallah et al. 2013] é um *framework* para avaliar o desempenho de diferentes sistemas gerenciadores de banco de dados relacionais diante de configurações de cargas de trabalho OLTP. O *framework* possui diversos *benchmarks* que representam diferentes aplicações, tais como TPC-C, Twitter, YCSB, AuctionMark e Wikipedia. O OLTPBenchmark possibilita configurar a taxa de transações, definir o percentual de cada tipo de transação por tempo de experimento, obter informações sobre vazão, média do

tempo de resposta e informações sobre utilização dos recursos do sistema gerenciador de banco de dados e do sistema operacional. Este *framework* utiliza o padrão JDBC (*Java Database Connectivity*) para a conexão com o sistema gerenciador de banco de dados.

Para realizar um experimento com o OLTPBenchmark, deve-se primeiro, criar, manualmente, o banco de dados no SGBD. Após a criação do banco de dados, utiliza-se um comando do OLTPBenchmark para criar o esquema do *benchmark* almejado no experimento. Depois da criação do esquema, é possível parametrizar o tamanho do banco de dados a ser utilizado pelo experimento. Neste caso, o OLTPBenchmark gera dados sintéticos para agregar dados nos bancos. O próximo passo é a criação do roteiro de execução, onde especifica-se o número de conexões com o banco, as transações a serem utilizas e o tempo de experimento. Além disso, é possível configurar os instantes de tempo em que a taxa de transação é alterada, podendo aumentar ou diminuir.

4. OLTPBenchAdmin

O OLTPBenchAdmin é uma ferramenta, implementada em Java, que disponibiliza uma interface gráfica de interação com o OLTPBenchmark. Com isso, automatizando e simplificando a invocação dos comandos de funcionamento do OLTPBenchmark. Além disso, o OLTPBenchAdmin permite a execução de outras funcionalidades e a interação de diversas instâncias do OLTPBenchmark que podem estar em diferentes máquinas da rede.

4.1. Arquitetura

O OLTPBenchAdmin utiliza os componentes arquiteturais que implementam o modelo de agentes distribuídos cooperando com um coordenador central [Tanenbaum e van Steen 2007]. Uma visão geral da arquitetura proposta para o OLTPBenchAdmin pode ser visualizada na Figura 1.

A interface gráfica do OLTPBenchAdmin tem como objetivo abstrair a complexidade da invocação dos *scripts* de execução do OLTPBenchmark, por meio de uma interface de janelas e controles. Essa interface serve de mecanismo de submissão de requisições ao OLTPBenchmark e visualização dos resultados. Neste sentido, a interface gráfica mantém conexões *sockets* com as diversas máquinas que possuem instalações do OLTPBenchmark. Por fazer a gestão de várias conexões sockets, o OLTPBenchAdmin permite a execução e controle de experimentos, utilizando o OLTPBenchmark, em diversas instâncias/máquinas, podendo assim realizar experimentos em paralelo.

A comunicação entre a interface gráfica e as máquinas é feita por um componente que está implantado em cada máquina que possui uma instância do OLTPBenchmark, denominado agente. Para isso, deve ser informado endereço de rede e porta da máquina que possui este agente. Com isso, estabelecendo um canal de comunicação com máquina alvo do experimento. O agente possui três principais funcionalidades: se comunicar com os SGBDs instalados na máquina, executar os *scripts* do OLTPBenchmark e devolver os resultados produzidos. Esses resultados devem ser visualizados pelo usuário do OLTPBenchAdmin.

4.2. Aspectos Funcionais

Esta seção descreve as funcionalidades que foram implementadas no OLTPBenchAdmin, tanto na parte gráfica quanto no agente que fica hospedado em cada máquina que possui

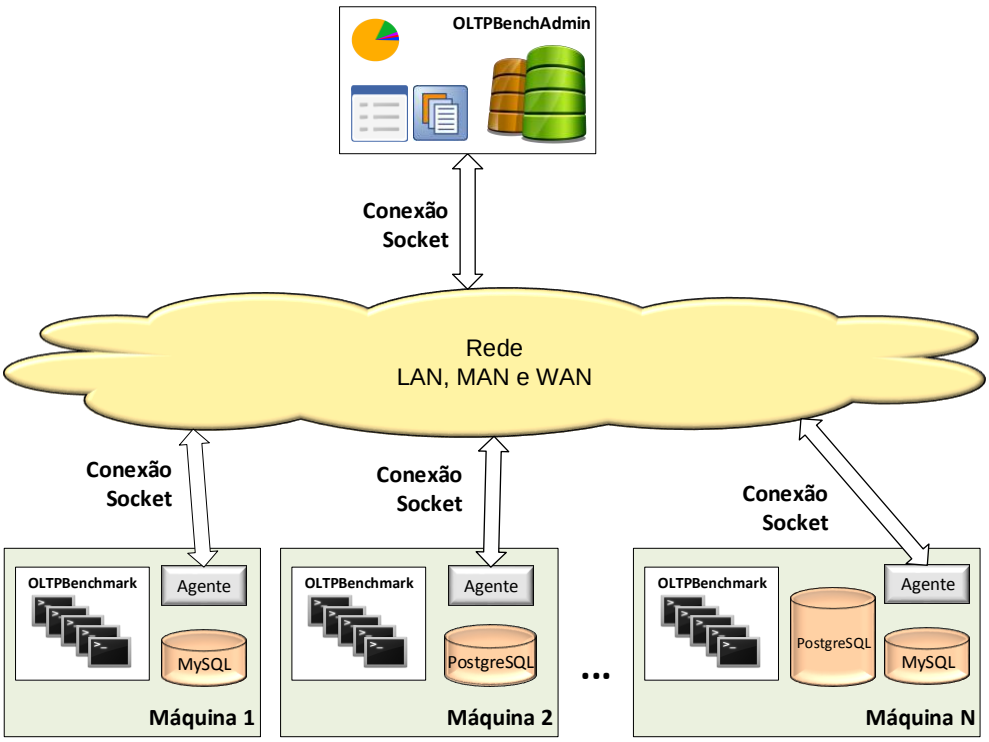

Figura 1. Arquitetura do OLTPBenchAdmin

instâncias do OLTPBenchmark. A Figura 2 demonstra a tela inicial da ferramenta gráfica, o OLTPBenchAdmin. Nela, é possível identificar quatro áreas: os menus, uma área a esquerda que mostra as máquinas envolvidas nos experimentos, uma área principal que serve para anexar experimentos a serem disparados nas máquinas e uma área que mostra informações sobre a execução dos comandos da ferramenta gráfica.

O OLTPBenchmark foi implementado com alguns *scripts* de comandos Linux, neste caso, sua execução é direcionada para máquinas Linux. No entanto, como mostra a Figura 2, onde o OLTPBenchAdmin está sendo executado em ambiente Windows. É possível, por meio do OLTPBenchAdmin, disparar e controlar experimentos em qualquer sistema operacional que suporte Java. Entretanto, as máquinas que possuem o OLTPBenchmark devem ser máquinas Linux. Cabe ao OLTPBenchAdmin administrar e coordenar todas as máquinas Linux envolvidas nos experimentos distribuídos.

Uma funcionalidade do OLTPBenchAdmin é a permitir a conexão com diversas máquinas que possuem instâncias do OLTPBenchmark. Para isso, existe um opção para conectar a uma máquina da rede, como mostra a Figura 3. Como parâmetros, é necessário informar o endereço da máquina e a porta onde o agente está executando. Neste sentido, além de instalar o agente na máquina é necessário configurar a porta que ele deve ativar seu *socket* servidor. Além disso, é necessário configurar um arquivo XML com as informações dos usuários administradores dos SGBDs instalados na máquina, pois serão mostradas funcionalidades que necessitam destas informações.

Quando a conexão for estabelecida com uma máquina, as informações sobre quais são os roteiros de experimentos do OLTPBenchmark, os SGBDs instalados e os bancos

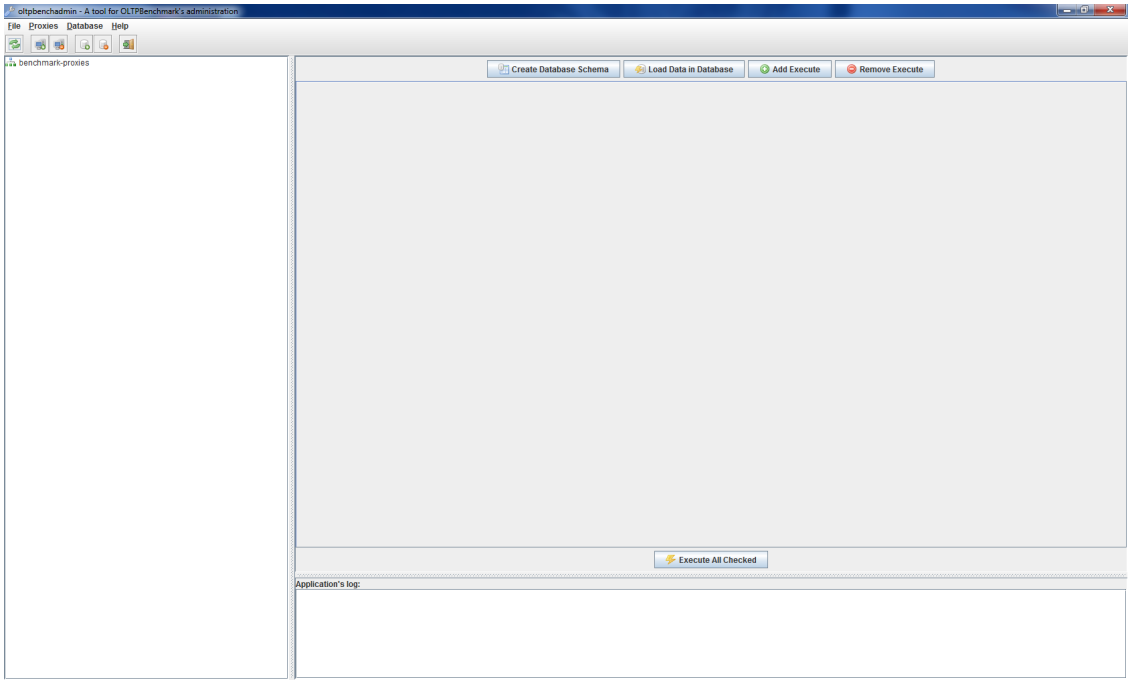

Figura 2. Tela inicial do OLTPBenchAdmin

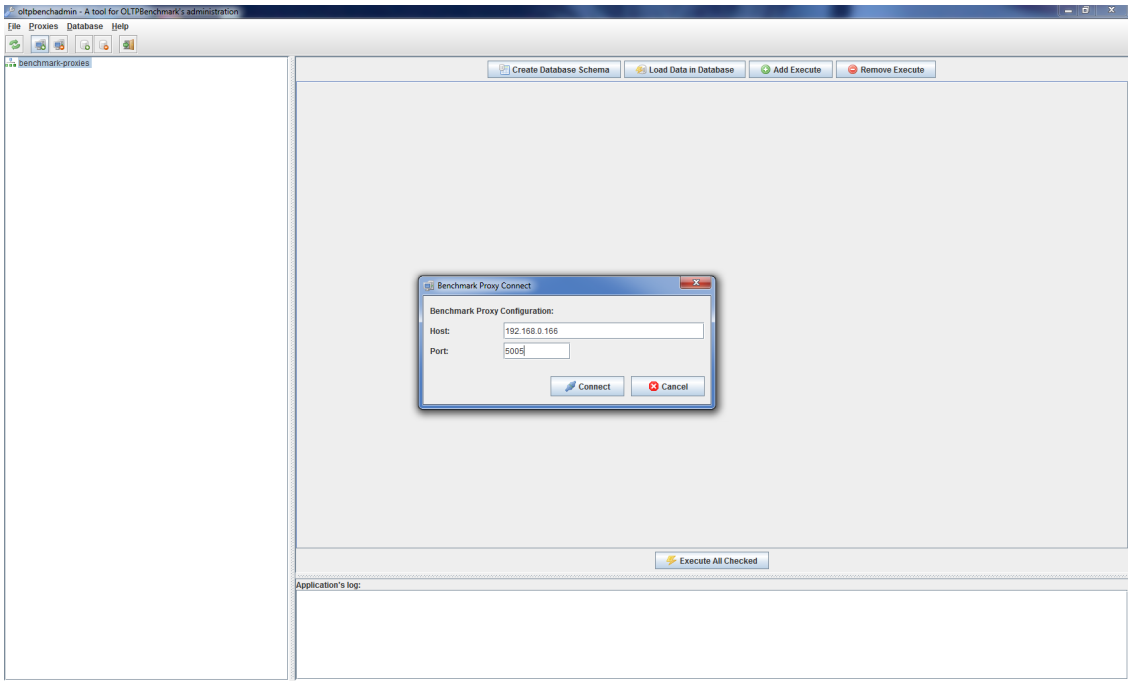

Figura 3. Tela de conexão com uma máquina

que estão nestes SGBDs, conforme a Figura 4. De posse destas informações o OLTPBenchAdmin pode interagir com os SGBDs, funções que seriam feitas manualmente pelos usuários das máquinas. Por exemplo, para criar ou remover um banco de dados, o código SQL tinha que ser elaborado pelo usuário. Já com o OLTPBenchAdmin, basta selecionar o SGBD e criar um banco de dados ou selecionar um banco de dados e mandar removê-lo. Outra funcionalidade do OLTPBenchAdmin é transferir os arquivos listados para a

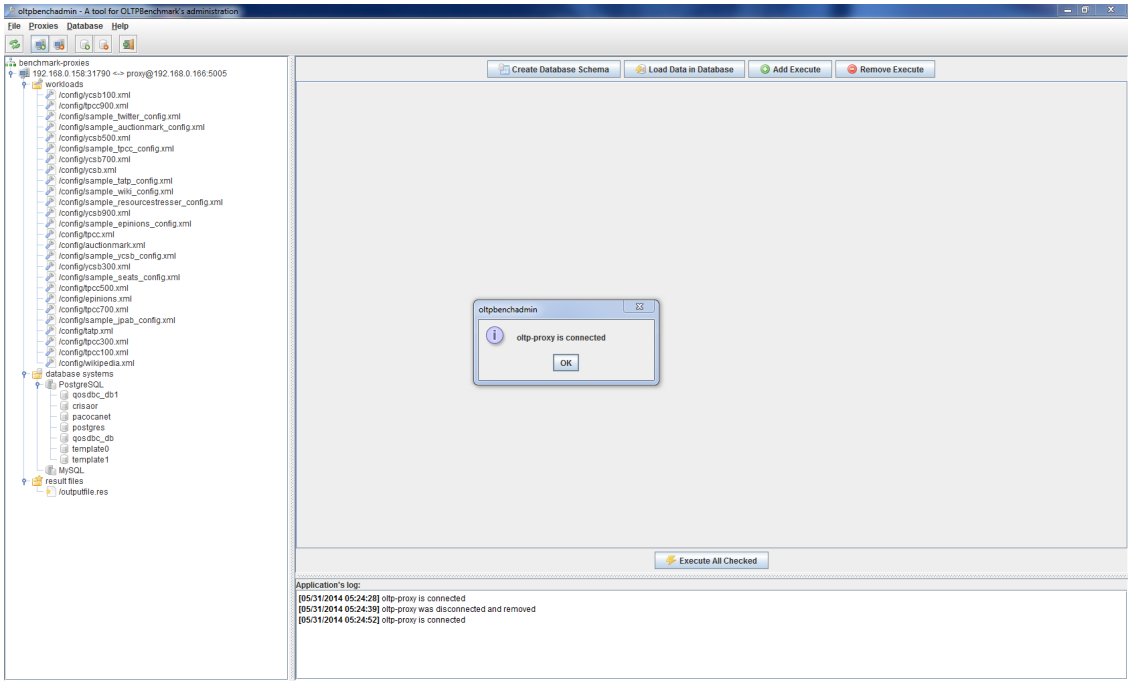

Figura 4. Tela após a conexão com um máquina

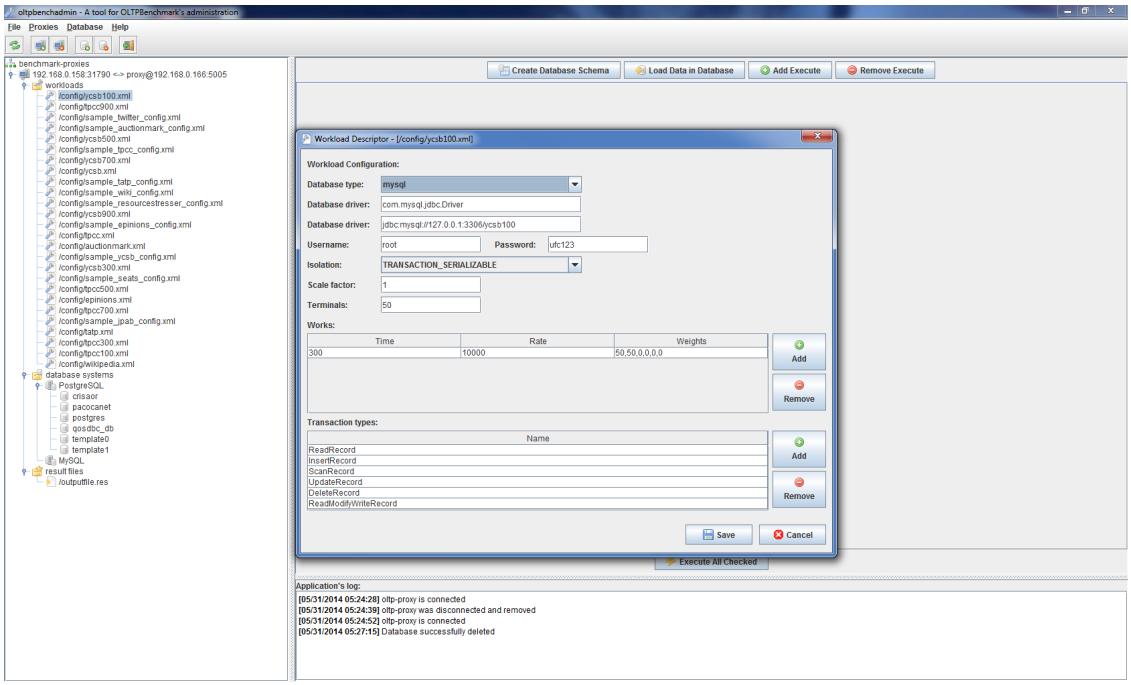

Figura 5. Formulário de um roteiro de experimento

máquina onde está operando o OLTPBenchAdmin, basta selecionar o arquivo e efetuar o *download*.

Os usuários do OLTPBenchAdmin podem criar, remover ou alterar roteiros de experimentos. A Figura 5 ilustra a alteração de um roteiro de experimento. Todos os campos do documento XML utilizado pelo OLTPBenchmark foram postos em um for-

mulário. Ao salvar, as informações são sincronizadas e propagadas para a máquina onde o arquivo está fisicamente armazenado. Basicamente, esses roteiros de experimentos possuem informações de acesso ao banco de dados, tipos de isolamento, tamanho da base, número de conexões, sazonalidade para mudança da taxa de transações e quais os tipos de transações que podem ser utilizadas no experimento. É válido ressaltar que, por meio destes roteiros, o OLTPBenchAdmin permite que seja criado o esquema do *benchmark* escolhido em um banco de dados e carregar dados sintéticos para gerar bancos de dados conforme o tamanho descrito no roteiro.

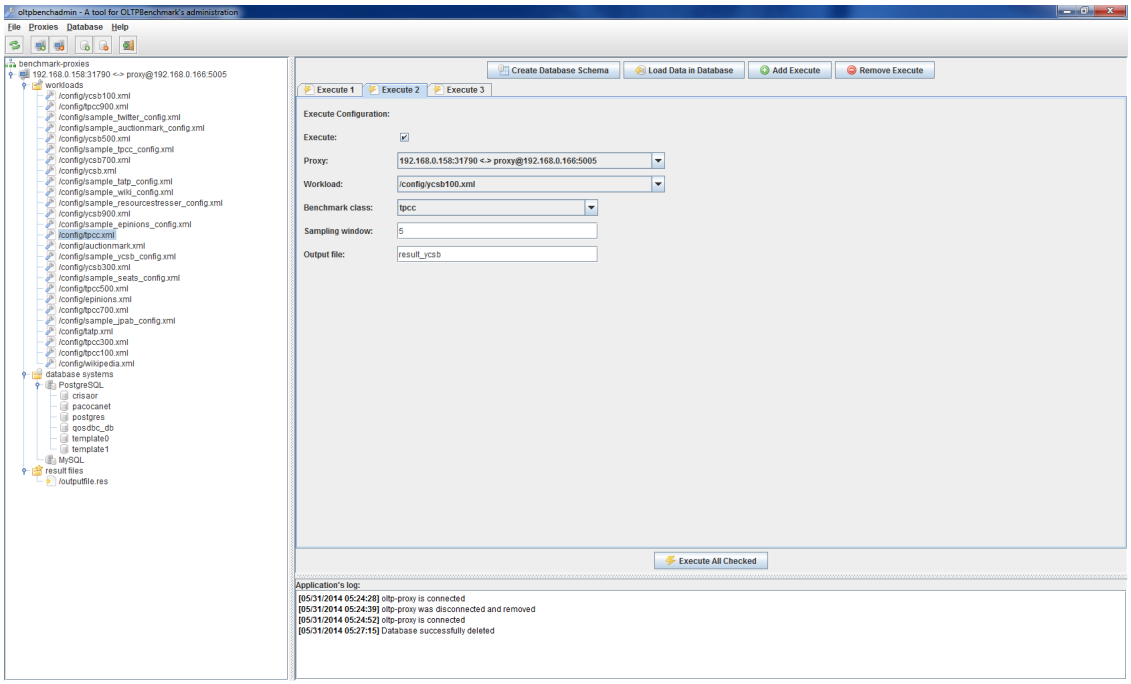

Figura 6. Especificação dos roteiros a serem executados de forma concorrente

Após de ter elaborado todos os roteiros de execução, o OLTPBenchAdmin, conectados com várias máquinas, possui a funcionalidade de especificar quais os roteiros de experimentos devem ser executados de forma concorrente. Para isso, basta clicar no roteiro e adicioná-lo na área principal da ferramenta, onde cada aba é um roteiro de experimento, conforme a Figura 6. Em cada aba especifica-se o nome do arquivo a ser gerado com os resultados do experimento, qual a máquina a ser executado o experimento, o roteiro, o tipo de *benchmark* e o instante de tempo que o OLTPBenchmark deve alimentar o arquivo de resultado do experimento.

Por fim, basta sinalizar a execução de todos experimentos. O OLTPBenchAdmin, por meio da sua área de informações, destaca o andamento dos experimentos. Ao final, na área reservada para os conteúdos das máquinas, possui um item que mostra todos os arquivos resultantes de experimentos, como pode ser observado no último item da árvore na Figura 6. Para que os usuários possam ver manipular esses arquivos, basta selecionar e efetuar o *download*

5. Conclusão e Trabalhos Futuros

Este ensaio apresentou o OLTPBenchAdmin, uma ferramenta de interface gráfica que simplifica a gestão do OLTPBenchmark e permite que o usuário controle experimentos distribuídos. Além disso, a ferramenta permite os usuários disparem seus experimentos a partir de máquinas que não possuem Linux, pois a interface gráfica foi desenvolvida em uma linguagem portável, no caso, Java. A arquitetura utilizada para implementar o OLTPBenchAdmin foi discutida e os aspectos funcionais demonstrados.

Como trabalhos futuros, pretende-se estudar outras funcionalidades do OLTPBenchmark e adicioná-las a nossa ferramenta. Além disso, entendemos que, para uma melhor facilidade e amplo acesso, seria interessante desenvolver uma versão *web* do OLTPBenchAdmin. Com isso, permitindo que o OLTPBenchAdmin seja utilizando tanto em PCs quanto em *tablets* e *smartphones*. Por fim, pretende-se realizar uma melhor visualização dos arquivos de retorno dos experimentos, elaborados pelo OLTPBenchmark. Neste sentido, colocar esses resultados de experimentos em uma forma gráfica, visando facilitar sua interpretação.

Referências

Agrawal, D., El Abbadi, A., Das, S., e Elmore, A. J. (2011). *Database Scalability, Elasticity, and Autonomy in the Cloud*, pages 2–15. Springer Berlin Heidelberg, Berlin, Heidelberg.

Baumann, P. e Stamerjohanns, H. (2014). Towards a systematic benchmark for array database systems. In Rabl, T., Poess, M., Baru, C., e Jacobsen, H.-A., editors, *Specifying Big Data Benchmarks*, volume 8163 of *Lecture Notes in Computer Science*, pages 94–102. Springer Berlin Heidelberg.

Cardoso, R. C., Zatelli, M. R., Hübner, J. F., e Bordini, R. H. (2013). Towards benchmarking actor- and agent-based programming languages. In *Proceedings of the 2013 Workshop on Programming Based on Actors, Agents, and Decentralized Control*, AGERE! 2013, pages 115–126, New York, NY, USA. ACM.

Chen, S.-H., Lin, H.-M., Chen, K.-Y., Chang, Y.-H., Yew, P.-C., e Ho, C.-C. (2013). A systematic methodology for os benchmark characterization. In *Proceedings of the 2013 Research in Adaptive and Convergent Systems*, RACS '13, pages 404–409, New York, NY, USA. ACM.

Date, C. (2004). *Introdução a sistemas de bancos de dados*. Campus.

Difallah, D. E., Pavlo, A., Curino, C., e Cudré-Mauroux, P. (2013). Oltp-bench: An extensible testbed for benchmarking relational databases. *PVLDB*, 7(4):277–288.

Elmasri, R. e Navathe, S. (2005). *Sistemas de banco de dados*. ADDISON WESLEY BRA.

Galpin, I., Stokes, A. B., Valkanas, G., Gray, A. J. G., Paton, N. W., Fernandes, A. A. A., Sattler, K.-U., e Gunopulos, D. (2014). Sensorbench: Benchmarking approaches to processing wireless sensor network data. In *Proceedings of the 26th International Conference on Scientific and Statistical Database Management*, SSDBM '14, pages 21:1–21:12, New York, NY, USA. ACM.

Hauser, K., Olsen, D., e Fadel, K. (2010). An integrated approach to teaching web development. *Review of Business Information Systems (RBIS)*, 14(1).

Hemel, Z. e Visser, E. (2011). Declaratively programming the mobile web with mobl. In *Proceedings of the 2011 ACM International Conference on Object Oriented Programming Systems Languages and Applications*, OOPSLA '11, pages 695–712, New York, NY, USA. ACM.

Matos, T. B., Brayner, A., e Maia, J. E. B. (2010). Towards in-network data prediction in wireless sensor networks. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 592–596, New York, NY, USA. ACM.

Moreira, L. O., Farias, V. A. E., Sousa, F. R. C., Santos, G. A. C., Maia, J. G. R., e Machado, J. C. (2014). Towards improvements on the quality of service for multi-tenant rdbms in the cloud. In *Proceedings of the 6th International Workshop on Cloud Data Management*, CloudDB '14, pages 162–169, Chicago, IL, USA. IEEE.

Moreira, L. O., Sousa, F. R. C., e Sabino, B. P. (2016). *OLTPBenchAdmin is a tool for managing the OLTPBenchmark*. `https://code.google.com/p/oltpbenchadmin/`.

Ozsu, M. T. (2007). *Principles of Distributed Database Systems*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition.

Reguly, I. Z., Keita, A.-K., e Giles, M. B. (2015). Benchmarking the ibm power8 processor. In *Proceedings of the 25th Annual International Conference on Computer Science and Software Engineering*, CASCON '15, pages 61–69, Riverton, NJ, USA. IBM Corp.

Skendžić, A., Kovačić, B., e Tijan, E. (2016). Effectiveness analysis of using solid state disk technology. In *2016 39th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1587–1591.

Sousa, F. R. C. e Machado, J. C. (2012). Towards elastic multi-tenant database replication with quality of service. In *Proceedings of the 2012 IEEE/ACM Fifth International Conference on Utility and Cloud Computing*, UCC '12, pages 168–175, Washington, DC, USA. IEEE Computer Society.

Sousa, F. R. C., Moreira, L. O., Santos, G. A. C., e Machado, J. C. (2012). Quality of service for database in the cloud. In Leymann, F., Ivanov, I., van Sinderen, M., e Shan, T., editors, *CLOSER*, pages 595–601. SciTePress.

Tanenbaum, A. e van Steen, M. (2007). *Distributed systems: principles and paradigms*. Pearson Prentice Hall.

Vőneki, B., Valat, S., Schwemmer, R., Neufeld, N., Machen, J., e Pérez, D. H. C. (2016). Evaluation of 100 gb/s lan networks for the lhcb daq upgrade. In *2016 IEEE-NPSS Real Time Conference (RT)*, pages 1–3.