I-DataMig: Uma Ferramenta Inteligente para Migração Eficiente e Segura de Bancos de Dados*

Victor Misael B.F. Carneiro[1], **Gustavo Moraes**[1], **Angelo Brayner**[1]

[1]Departamento de Computação – Universidade Federal do Ceará (UFC)
Campus do Pici – Fortaleza – CE – Brasil

{victor.misael, gustavo.moraes}@alu.ufc.br, brayner@dc.ufc.br

Abstract. *Efficient database migration between different systems is crucial in a constantly evolving technological landscape, where adaptability and minimizing downtime are critical. This paper presents the I-DataMig tool, a solution developed to automate the database migration process, providing an intuitive interface, as well as a fault-tolerant and auditable process. The I-DataMig allows detailed control of the migration steps, from the transfer of schemas and table instances to the migration of integrity constraints and indexes between different database systems. This work details the architecture, overview, and performance analysis of the tool.*

Resumo. *A migração eficiente de bancos de dados entre diferentes sistemas é fundamental no cenário atual, onde a adoção de novas tecnologias é um imperativo. Assim, este trabalho apresenta a ferramenta I-DataMig*[1]*, uma solução desenvolvida para automatizar o processo de migração de bancos de dados, proporcionando uma interface intuitiva, além de um processo tolerante a falhas e auditável. O I-DataMig permite um controle detalhado dos passos do processo de migração, desde a transferência de esquemas e instâncias de tabelas até a migração de restrições de integridade e índices entre diferentes sistemas de bancos de dados. Este trabalho apresenta uma visão geral da ferramenta, bem como detalha sua arquitetura e avaliação de desempenho.*

1. Introdução

Com a evolução tecnológica constante e as mudanças dinâmicas nas demandas comerciais, os processos de migração de banco de dados emergem como elementos cruciais no desenvolvimento de aplicações de software. Impulsionadas pelo ritmo acelerado do progresso tecnológico, pelas demandas comerciais em constante mutação e pela busca por otimização do desempenho e redução de custos, as organizações se veem diante da necessidade de constantemente migrar seus dados entre plataformas. Seja na migração de sistemas locais para a nuvem ou na atualização para bancos de dados mais robustos em busca de recursos expandidos, esse desafio demanda uma condução precisa.

O processo de migração de banco de dados pode gerar riscos significativos para a integridade e segurança dos dados. Entre os principais desafios enfrentados durante esse processo, destaca-se a perda ou comprometimento desses dados, provocada por exemplo, pela incompatibilidade de tipos de dados [Matthes et al. 2011].

*Vídeo da demonstração da ferramenta disponível em: https://youtu.be/09pV-lPnTug

[1]Código da ferramenta disponível em: https://github.com/VicMisael/IDataMig

No cenário de migração de bancos de dados, o uso de uma ferramenta intuitiva, adaptável e auditável, com a geração de arquivo de log das operações executadas na migração, torna-se essencial. Além disso, um maior controle sobre os processos de migração permite ao usuário escolher quais estruturas do banco de dados serão migradas, como tabelas, tuplas, restrições de integridade e estruturas de índice. Por exemplo, a possibilidade de migrar tuplas antes de migrar índices e restrições tende a ser reduzido, já que elimina o tempo de inserir novas entradas na estrutura de índice, uma árvore B^+ , e o tempo de verificar restrições. Outro aspecto fundamental durante o processo de migração é a capacidade de auditar, corrigir e recuperar-se de eventuais falhas.

Com o objetivo de automatizar o processos de migração de bancos de dados, apresentamos a ferramenta *I-DataMig*, uma ferramenta Source-to-Conceptual-Target [Elamparithi and Anuratha 2015]. A ferramenta proposta executa, sem intervenção humana, todo o processo de migração de bancos de dados de diferentes sistemas de banco de dados (SGBD), como de MySQL para PostgreSQL e vice-versa. Assim, basta o DBA disparar o processo na ferramenta e acompanhar todo o processo através da interface de acompanhamento da migração da *I-DataMig*. Vale destacar que a *I-DataMig* garante a recuperação do processo de migração após a ocorrência de falhas.

Para avaliar a ferramenta proposta, experimentos foram realizados, utilizando-se o TPC-H, fatores de escala 1 e 10. Os resultados obtidos evidenciam a eficiência do *I-DataMig*, com relação à correção da migração e ao baixo tempo para executar todo o processo. Destaca-se ainda a utilização bem-sucedida ferramenta em uma empresa, que precisava migrar um banco de dados legado MySQL para PostgreSQL com replicação e gerenciamento de dados espaciais. O banco de dados legado possuía 120 tabelas, contendo aproximadamente 30 milhões de tuplas, 460 índices. O espaço de armazenamento do banco de dados legado era de 5,60 GB.

Este artigo está estruturado da seguinte forma: a Seção 2 revisa os trabalhos relacionados. A Seção 3 detalha a arquitetura da ferramenta *I-DataMig*. A Seção 4 descreve a o uso da ferramenta, bem como a interface do usuário. A Seção 5, por sua vez, analisa o desempenho da ferramenta com a execução de experimento, utilizando o *benchmark* TPC-H. Por fim, a Seção 6 apresenta as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Por vários anos, diversas ferramentas de migração de banco de dados têm sido propostas para auxiliar no processo de migração. No trabalho de [Neto et al. 2013], é apresentado um catálogo de requisitos que pode ser usado para facilitar o desenvolvimento de novas ferramentas. Estes requisitos incluem a capacidade de migração entre banco de dados diferentes (heterogêneos), migração paralela ou distribuída, migração completa ou incremental, reengenharia de dados, além de testabilidade e usabilidade.

Ferramentas baseadas no modelo ETL (Extract, Transform, Load), como *Oracle Data Service Integrator* e *Pentaho Data Integration*, têm se mostrado eficazes em processos de migração. Adicionalmente, ferramentas que aproveitam ambientes na nuvem, como *Azure Migrate* e *AWS Database Migration Service*, oferecem escalabilidade e flexibilidade, facilitando a migração em larga escala e a integração com serviços de nuvem.

Nos trabalhos de [Neto et al. 2013] e [Elamparithi and Anuratha 2015], apresentam uma vasta coleção de ferramentas de migração. Estas ferramentas são discuti-

das em termos de suas capacidades e aplicações específicas. Adicionalmente, os estudos de [Sarmah 2018] e [Matthes et al. 2011] abordam perspectivas sobre diversas estratégias de migração e como utilizar as ferramentas de forma inteligente, e por fim [Kurako and Orlov 2023] demonstram um processo de migração de Oracle para Postgres, comparando ferramentas e exemplificando o mapeamento de tipos. Um aspecto comum nessas estratégias é o planejamento em etapas.

Em linha com essas estratégias, nossa abordagem oferece a vantagem de uma migração em etapas, melhorando a usabilidade ao dividir o processo e permitindo uma migração incremental. Além disso, nosso diferencial está no mecanismo de tolerância a falhas, que retoma a migração do ponto onde foi interrompida, sem a necessidade de reiniciar o processo. Essa metodologia atende aos requisitos de continuidade e rastreabilidade de erros, permitindo ajustes e garantindo um registro detalhado das etapas concluídas.

3. Arquitetura do *I-DataMig*

O *I-DataMig* é uma ferramenta desenvolvida em Python capaz de automatizar a migração de bancos de dados de SGBDs relacionais, independentemente se os SGBDs sejam diferentes ou o mesmo. Sua interface (ver Seção 4) possibilita uma migração incremental na qual o usuário seleciona objetos específicos do banco de dados a serem migrados.

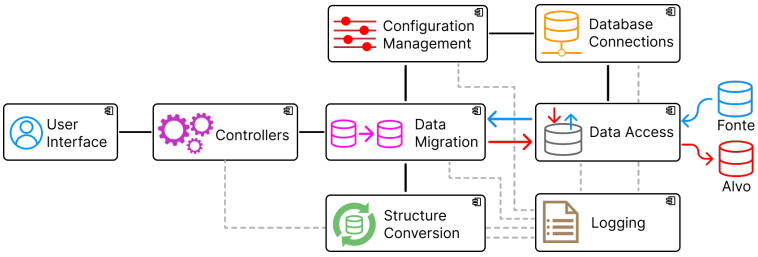

Figura 1. Representação Conceitual do *I-DataMig*

A Figura 1 apresenta a Representação Conceitual do *I-DataMig* O usuário interage com o componente *User Interface*, que se comunica por meio de requisições web com os *Controllers*, responsáveis por coordenar a migração. O componente *Logging* registra eventos, erros e atualizações de status durante o processo de migração, fornecendo informações detalhadas para solução de problemas, auditoria e monitoramento.

O componente *Database Connections* estabelece e gerencia conexões com os bancos de dados, lidando com autenticação e pooling de conexão. O *Data Access* fornece métodos para leitura e escrita nos bancos de dados, lidando com consultas, transações e processamento de conjuntos de resultados. O *Data Migration* é responsável por transferir os dados com segurança, garantindo a integridade e consistência durante o processo. O *Configuration Management* gerencia as configurações do sistema de migração, permitindo que os usuários especifiquem credenciais e outros parâmetros (ver Seção 5.2). Por fim, o *Structure Conversion* converte a estrutura e definições de objetos, lidando com diferenças em tipos de dados, restrições e indexação entre os dois SGBDs.

Inicialmente o *I-DataMig* lê diversas informações sobre o banco de dados fonte usando o *Data Access*. Baseada na abordagem *Source-to-Conceptual-To-Target* descrita por [Elamparithi and Anuratha 2015], a ferramenta cria um formato intermediário em um

objeto JSON, no qual armazena informações do banco de dados fonte, como tipos de dados, chaves primárias, chaves estrangeiras, cardinalidades e índices, conforme descritas na modelagem. Dessa forma, o componente *Structure Conversion* adapta estes objetos intermediários para a linguagem SQL específica do banco de dados alvo. Devido ao grande número de tipos de dados diferentes entre bancos de dados, em alguns casos o *I-DataMig* pode não fornecer uma conversão adequada. Por isso, a ferramenta também oferece uma área para o usuário implementar funções de conversão em Python.

Durante a migração de tuplas (por meio do *Data Migration*), a ferramenta permite a transferência de dados diretamente entre os SGBDs ou por meio de uma abordagem focada na tolerância a falha. Essa abordagem armazena os dados em arquivos indexados por blocos e mantém um log de controle dos blocos migrados, útil para recuperação em caso de falha, pois registra o último bloco salvo com sucesso. As transações de inserção de cada bloco são enviadas ao banco de dados alvo. Em caso de falha dos SGBDs ou da ferramenta, o usuário é imediatamente informado, e o *I-DataMig* consulta o log de controle para continuar a migração a partir do último bloco migrado com sucesso.

4. Conhecendo o *I-DataMig*

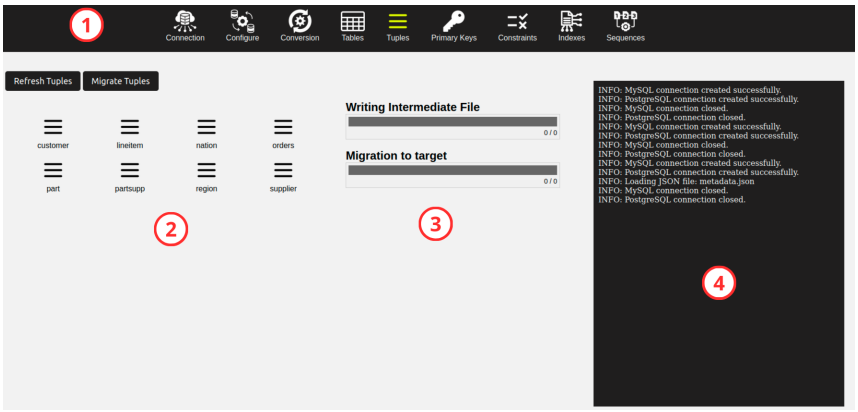

Figura 2. Interface de Usuário

A Figura 2 ilustra a interface do *I-DataMig*. Em **(1)** vemos o painel superior onde podemos selecionar as etapas de migração. A opção *Connection* verifica o status das conexões. Em *Configure*, inserimos o nome do esquema alvo e os tamanhos dos blocos de leitura e escrita (ver Seção 5.2). A etapa *Conversion* realiza as devidas conversões de estrutura. Em *Tables*, selecionamos as tabelas a serem migradas, e em *Tuples*, migramos as tuplas para as tabelas escolhidas. Os botões seguintes tratam da migração dos objetos correspondentes. Após a configuração, o usuário pode escolher a ordem de migração dos objetos, como optar por migrar chaves primárias e restrições antes de inserir as tuplas.

A cada opção de migração selecionada no painel superior **(1)**, o painel **(2)** assume diferentes configurações, oferecendo flexibilidade ao usuário para definir quais objetos específicos devem ser migrados. Por exemplo, podemos optar por migrar uma tabela, mas ao mesmo tempo não migrar seus índices. Isso pode ser usado para economizar tempo e recursos, especialmente quando certos objetos não são críticos para o funcionamento imediato do sistema e podem ser migrados posteriormente. O painel **(3)** exibe informações de status das operações. Por exemplo, ao selecionar a opção de migração de tuplas, é

possível acompanhar o progresso das inserções nos arquivos intermediários e no banco de dados de destino. Durante todo o processo de migração, o painel **(4)** apresenta o log de execução, permitindo monitorar o progresso e identificar possíveis erros.

5. Avaliação Experimental

Nesta seção, apresentamos os experimentos conduzidos e os resultados obtidos.

5.1. Ambiente de Avaliação

Para avaliar o *I-DataMig* foi usado o benchmark TPC-H. Para testar o tempo de migração de um banco de dados MySQL para PostgreSQL, usamos a ferramenta Benchbase [Difallah et al. 2013] para criar o esquema e popular os dados do banco de dados de origem (MySQL). A Tabela 1 apresenta as características de cardinalidade e o tamanho ocupado das tabelas inseridas no banco de dados de origem para o fator de escala 1 e 10.

Tabela 1. Métricas TPCH Fator de Escala 1 e 10

	customer	lineitem	nation	orders	part	partsupp	region	supplier
Rows (F1)	147,940	5,463,772	25	1,386,862	198,300	915,396	5	10,037
Total MB (F1)	33.58	2,107.34	0.05	260.33	34.08	270.88	0.03	2.91
Rows (F10)	1,481,039	59,457,290	25	14,182,066	1,853,948	7,792,243	5	99,026
Total MB (F10)	315.83	19,230.86	0.05	2,383.77	310.30	2,498.94	0.03	21.55

Os experimentos foram executados em uma máquina de 64 GB de RAM e 480 GB de SSD Kingston SA400S37480G com Ubuntu Linux 18.04 LTS, Intel i7-9700k 3,60 GHz, ambas as instâncias dos bancos de dados estavam em rede local.

5.2. Experimentos com Tamanho do armazenamento

A aplicação *I-DataMig* permite ajustar os parâmetros de leitura e armazenamento de dados. No contexto deste teste, foram analisadas duas configurações principais: o tamanho do bloco de leitura (*Batch Size*) na tabela fonte e o número de tuplas inseridas por transação no bloco de escrita (*Bulk Insert*) na tabela alvo.

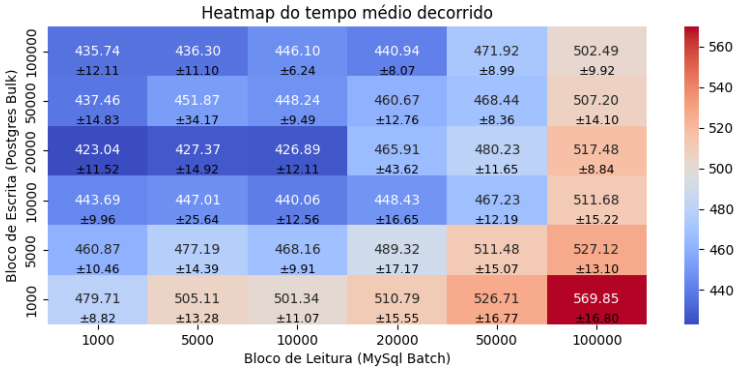

Figura 3. Heatmap dos tamanhos de *Batch* e *Bulk*

Para o fator de escala 1, conduzimos experimentos para avaliar o desempenho dos procedimentos de leitura e escrita. O objetivo foi identificar a combinação ideal entre o tamanho do lote de leitura e o tamanho do buffer de escrita (*Bulk Size*) para otimizar os tempos de migração. A Figura 3 apresenta um mapa de calor com a média de tempo em

segundos e a margem de erro com confiança de 97,5 % dos experimentos, correlacionando o tamanho do *Batch* de leitura do MySQL no eixo x com o tamanho do *Bulk* de inserção do PostgreSQL no eixo y, onde cada célula representa o tempo médio de execução da migração de tuplas em segundos, calculado a partir de 10 execuções.

A análise indica que combinar tamanhos menores de blocos de leitura (1000, 5000, 10000) com blocos de escrita moderados (20000) melhora significativamente o tempo de migração. Uma abordagem equilibrada entre leitura e escrita otimiza a performance. Em contraste, blocos maiores de leitura combinados com blocos menores de escrita resultaram nos maiores tempos de migração. Isso pode ser atribuído ao aumento do *overhead* de gerenciamento de memória durante a leitura e à ineficiência de realizar múltiplas operações de escrita menores em vez de aproveitar a capacidade de escrita em blocos maiores de uma só vez.

O experimento com fator de escala 10, com 15 execuções, usando 24 GB de dados e 84 milhões de tuplas migradas do MySQL para PostgreSQL com parâmetros de *Batch* e *Bulk* fixos em 1000 e 20000, respectivamente, indicam um tempo médio de migração de 5393 segundos (cerca de 90 minutos) com uma margem de erro de 494 segundos (cerca de 8 minutos) dentro de um intervalo de confiança de 97,5%. A margem de erro calculada nos tempos de migração sugere consistência e previsibilidade dado o cenário.

6. Conclusão

Neste artigo, apresentamos a ferramenta *I-DataMig* destinada à migração de bancos de dados. A ferramenta organiza a migração em etapas opcionais. Os dados são armazenados em formato JSON e posteriormente convertidos para um formato compatível com o SGBD de destino. O *I-DataMig* oferece uma solução eficiente e tolerante a falhas para realizar migrações, minimizando o tempo de inatividade e a perda de dados. Para trabalhos futuros, pretendemos superar as desvantagens atuais, como o suporte restrito à migração entre diferentes esquemas e à compatibilidade de tipos de dados entre origem e destino, migração de outros elementos como *triggers* e *stored procedures*, além de abordar requisitos de segurança, como o tratamento de dados criptografados.

Referências

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

Elamparithi, M. and Anuratha, V. (2015). A review on database migration strategies, techniques and tools. *World Journal of Computer Application and Technology*, 3(3):41–48.

Kurako, E. A. and Orlov, V. L. (2023). Database migration from oracle to postgresql. *Programming and Computer Software*, 49(5):455–463.

Matthes, F., Schulz, C., and Haller, K. (2011). Testing & quality assurance in data migration projects. In *2011 27th IEEE International Conference on Software Maintenance (ICSM)*, pages 438–447.

Neto, P. S., Neto, J. R., Júnior, F. R., and Oliveira, P. (2013). Requisitos para ferramentas de migração de dados. In *Anais do IX Simpósio Brasileiro de Sistemas de Informação*, pages 887–898, Porto Alegre, RS, Brasil. SBC.

Sarmah, S. S. (2018). Data migration. *Science and Technology*, 8(1):1–10.