Estudo de caso sobre o processamento de consultas com a *engine FederatedX*

Samuel L. B. Bispo[1], **Raqueline R. M. Penteado**[1]

[1]Departamento de Informática – Universidade Estadual de Maringá (UEM)
87.020-900 – Maringá – PR – Brasil

{ra103643,rrmpenteado}@uem.br

Abstract. *This article presents a case study on the federated query processing in the FederatedX engine of the MariaDB DBMS. The central focus of the study was query optimization, specifically the minimization of the execution time. The study considered two aspects: i) the definition of the logical model, and ii) the data indexing mechanism. The study's results demonstrated that the definition of the logical model was crucial in reducing processing time, while the same was not observed about indexing.*

Resumo. *Esse artigo apresenta um estudo de caso sobre o processamento federado de consultas na engine FederatedX do SGBD MariaDB. O ponto central do estudo foi a otimização de consultas, mais especificamente, a minimização do tempo de processamento. O estudo considerou dois pontos, sendo eles: i) a definição do modelo lógico e ii) o mecanismo de indexação de dados. Os resultados do estudo mostraram que a definição do modelo lógico foi essencial para a redução do tempo de processamento. Em contrapartida, o mesmo não ocorreu com a indexação.*

1. Introdução

Na última década o volume de dados armazenado e processado por Sistemas de Informação vem crescendo de forma exponencial. Além do volume, a variedade de modelos de dados também tem aumentado a fim de atender diversos tipos de sistemas. A possibilidade do processamento de consultas envolvendo bases de dados heterogêneas mostrou-se interessante para o processo de tomada de decisões em diversos contextos, tais como indústrias, instituições de ensino e *e-commerce*.

Uma alternativa para o processamento dessas consultas é a utilização de uma abordagem federada que provê uma visão unificada de todos os dados, alocados fisicamente em seus respectivos servidores, através de uma camada virtual [Arh 2022].

Um Sistema Federado de Banco de Dados é uma coleção de sistemas autônomos de bancos de dados que cooperam entre si para a execução de tarefas [Sheth and Larson 1990]. Os bancos de dados estão distribuídos (fisicamente ou logicamente) em *clientes* e um *servidor* contém a representação lógica dos dados alocados,nos clientes, conforme mostra a Figura 1. Sendo assim, os usuários do sistema federado enviam requisições de consultas ao servidor que resolve as consultas trocando mensagens como os clientes.

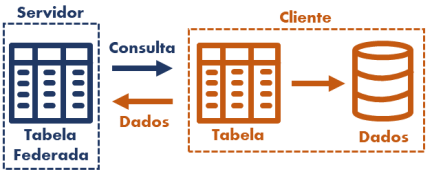

Figura 1. Sistema federado: comunicação entre servidor e cliente

Diversos sistemas foram propostos para atender a demanda de processamento federado de consultas. O MariaDB[1] admite somente bases de dados relacionais. Outros, tais como o Dremio[2], o DynamoDB[3] [Azevedo et al. 2020], admitem bases com modelos de dados variados.

Este estudo é uma continuação do estudo realizado em [Bispo et al. 2022] que trata da integração semântica de dados explorando ontologias por meio do *Ontop*. O objetivo deste trabalho é explorar a questão da distribuição física de dados, já que no estudo anterior os dados foram alocados em um único servidor. O experimento realizado no estudo utilizou o MariaDB, uma vez que o *Ontop* dá suporte ao mesmo. O *MariaDB* é um sistema *open source* que adota a *engine FederatedX* para o processamento federado de consultas. Os resultados do estudo mostraram que a *engine* possui uma alta facilidade de uso, porém apresenta limitação no quesito otimização de consultas.

2. Materiais e Métodos

O ambiente do experimento contou com três máquinas, um servidor e dois clientes. Quanto aos dados, duas bases, disponíveis no portal do DATASUS, com informações sobre procedimentos de cirurgia bariátrica realizados através do Sistema Único de Saúde Brasileiro (SUS) foram utilizadas no experimento. A primeira é a ABOPR, que contém dados do SIASUS (Sistema de Informações Ambulatoriais do SUS) sobre laudos de APAC (Autorização de Procedimento de Alta Complexidade) de acompanhamento a cirurgia bariátrica. A segunda é a RDPR, que possui dados do SIHSUS (Sistema de Informações Hospitalares do SUS) sobre registros de AIH (Autorização de Internação Hospitalar) de todos os atendimentos provenientes de internações hospitalares financiados pelo SUS. Ambas as bases consideraram dados de Janeiro de 2020 do Paraná ficando a RDPR com 3.161 registros e a ABOPR com 67.011 registros. Cada cliente alocou uma base de dados.

O estudo analisou a execução de uma consulta que envolve as duas bases do experimento. O objetivo da consulta é verificar se *as comorbidades dos pacientes implicam no custo da cirurgia bariátrica?"*.

3. Estudo de caso

O estudo de caso pode ser dividido em duas etapas, a definição do modelo lógico e a indexação de dados visando reduzir o tempo de processamento de consultas. A definição do modelo lógico envolveu a criação de tabelas federadas no servidor para a viabilização

[1]https://mariadb.com/
[2]https://www.dremio.com/
[3]https://aws.amazon.com/pt/dynamodb/

do processamento federado de consultas. A indexação envolveu a criação de índices para a execução da consulta.

3.1. Definição do modelo lógico

Após a alocação das bases de dados nos clientes, o servidor foi configurado no *FederatedX* e conectado aos clientes. A configuração do servidor envolveu a criação de duas tabelas federadas no mesmo, uma para cada base alocada nos clientes. A criação das tabelas federadas é semelhante à criação de tabelas locais nos clientes. Logo, a curva de aprendizagem é baixa para os usuários já familiarizados com SGBDs relacionais.

A Figura 2 mostra o *script* utilizado no servidor para a criação de uma tabela federada no servidor. No *FederatedX*, além do *Create Table* usual, faz-se necessário a definição da *engine FEDERATED*, do *CHARSET* e dos parâmetros de conexão que contêm: o *SGBD*; o usuário e a sua senha; e, o endereço IP, a porta, o nome do esquema e da tabela do cliente.

```
CREATE TABLE federated_rdpr (
  `n_aih` char(13) DEFAULT NULL,
  `val_tot` decimal(14,2) DEFAULT NULL
)
ENGINE='FEDERATED'
DEFAULT CHARSET='utf8mb4'
CONNECTION='mysql://root:senha@192.168.1.19:3306/ontop/rdpr';
```

Figura 2. *Script* SQL para criação da tabela federada referente a base RDPR

De acordo com a documentação do *FederatedX*[4], a tabela federada deve ser idêntica à tabela do cliente. Porém, no estudo realizado foi possível a criação de tabelas federadas com somente alguns atributos das tabelas remotas. A Figura 2 mostra apenas os atributos da base RDPR utilizados na consulta do experimento. Logo, o estudo considerou dois cenários, no *Cenário 1* as tabelas federadas são idênticas às tabelas alocadas nos clientes e, no *Cenário 2*, somente os atributos envolvidos na consulta são considerados nas tabelas federadas criadas no servidor. Enquanto que no *Cenário 1* as tabelas federadas referentes à RDPR e ABOPR possuem, respectivamente, 113 atributos e 78 atributos. No *Cenário 2*, as tabelas RDPR e ABOPR possuem, respectivamente, 02 e 07 atributos.

3.2. Execução de consultas

Dada a requisição de uma consulta, o *FederatedX* detecta as bases envolvidas na consulta. Na sequência, o servidor envia uma consulta SQL aos clientes solicitando dados das bases. Por fim, o servidor processa os dados recebidos gerando o resultado da consulta requisitada. Um ponto que vale a pena destacar é a solicitação de dados do servidor ao cliente. O *script* da consulta SQL depende da estrutura da tabela federada criada no servidor, uma vez que todos os atributos da tabela federada são considerados na operação *SELECT* da consulta enviada ao cliente.

A consulta do experimento envolveu uma operação de junção entre a ABOPR e a RDPR, além de um filtro na base ABOPR. As bases de dados foram indexadas tomando como base as operações envolvidas na consulta, ou seja, um índice envolvendo os atributos da operação de junção e um índice envolvendo os atributos do filtro. O *MariaDB* não permite a criação de índices no servidor.

[4]https://mariadb.com/kb/en/about-federatedx/

Cenário 1		Cenário 2	
Sem indexação	Com indexação	Sem indexação	Com indexação
139.97 s	138.86 s	36.29 s	36.28 s

Tabela 1. Tempo médio da consulta nos Cenários 1 e 2, com indexação ou não.

A Tabela 1 mostra os tempos de processamento da consulta variando o modelo lógico do servidor, *Cenário 1* e *Cenário 2*, e a indexação de dados nos clientes. O tempo médio (em segundos) de processamento da consulta foi de 139.97 s no *Cenário 1* e de 36.29 s no *Cenário 2*. É possível notar que a redução do número de atributos nas tabelas federadas implicou diretamente no tempo de processamento da consulta, uma vez que o servidor recuperou e processou um volume de dados maior no *Cenário 1* quando comparado com o *Cenário 2*. Já, em relação à indexação foi possível notar que a mesma não afetou o tempo médio de execução da consulta. A operação de junção não foi beneficiada com a indexação considerando que as tabelas envolvidas na operação foram recuperados de lugares distintos. Já, a operação do filtro nem foi considerada no *script* enviado do servidor para o cliente, uma vez que o servidor simplesmente requisita todos os dados das bases envolvidas na consulta.

4. Considerações Finais

A facilidade de uso da *engine FederatedX* é alta para usuários familiarizados com SGBDs relacionais. Porém, o estudo mostrou que a *engine* adota uma abordagem muito simples para o processamento federado de consultas comprometendo o desempenho de consultas que envolvem bases alocadas em clientes distintos. O ponto de destaque do estudo é que a definição do modelo lógico no servidor pode implicar diretamente no tempo de processamento federado de consultas. No experimento realizado o tempo de processamento da consulta reduziu em 74.07% quando somente os atributos relevantes para a consulta foram considerados no modelo lógico do servidor.

Como trabalho futuro vislumbra-se propor alterações para o código fonte do *FederatedX* a fim de compará-lo com o desempenho de outras ferramentas existentes, como o *Dremio* e o *Denodo*.

5. Agradecimentos

Ao PIC-UEM que possibilitou a participação do primeiro autor na pesquisa.

Referências

Arh, M. (2022). What is data federation, and why is it important to your organization?

Azevedo, L., F. de S. Soares, E., Souza, R., and Ferreira Moreno, M. (2020). Modern federated database systems: An overview. pages 276–283.

Bispo, S., Fukace, V., Mazur, A., Penteado, R., and Teixeira, H. (2022). Integração de dados abertos em saúde com o modelo obda: Um estudo de caso na Área de cirurgia bariátrica. In *Anais do XXII Simpósio Brasileiro de Computação Aplicada à Saúde*, pages 401–412, Porto Alegre, RS, Brasil. SBC.

Sheth, A. P. and Larson, J. A. (1990). Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Comput. Surv.*, 22(3):183–236.