VI Simpósio de Iniciação Científica 2023
Ciência como ferramenta de transformação da sociedade

METAPROGRAMAÇÃO DE LINGUAGEM ORIENTADA A OBJETOS EM BANCO DE DADOS ORIENTADO A GRAFOS

Giuseppe B. Scassiotti[1] (IC), Enzo Seraphim (PQ)[1]
[1]*Universidade Federal de Itajubá.*

Palavras-chave: Armazenamento. Grafos. Metaprogramação.

Introdução

No desenvolvimento de aplicações que usam banco de dados é necessário a definição de um modelo conceitual de dados que pode ser modelo entidade-relacionamento, modelo orientado a objetos, modelo multidimensional, modelo hierárquico e modelo de grafos.

O modelo conceitual de dados baseado em grafo representa uma categoria conhecida como NoSQL que são amplamente reconhecidos por sua facilidade de desenvolvimento, funcionalidade e desempenho em escala (Elmasri, Navathe, 2018).

Definido o modelo conceitual dos dados deve-se codificar o sistema utilizando a linguagem e seu paradigma de programação que pode ser procedural, orientada a objetos, funcional, genérica ou multiparadigma.

O paradigma de programação orientada a objetos é suportado por diversas linguagens e possibilita a construção de sistema muito mais complexo com ciclo de vida mais longo.

O modelo conceitual de dados baseado em grafo apresenta um nível conceitual de abstração diferente do modelo de programação orientada a objetos, sendo necessário um mapeamento entre os modelos.

Este trabalho tem por objetivo usar de metaprogramação para o mapeamento entre modelos: do orientado a objeto para o grafo; do orientado a objeto para o relacional.

Para atingir essa meta, foi estabelecido os seguintes objetos específicos para este trabalho:

- Implementar carregamento de base de dados real no banco de dados orientado a grafo;
- Implementar carregamento de base de dados real no banco de dados relacional;
- Gerar um conjunto de consultas em grafos na linguagem Cypher e em SQL;
- Avaliar o desempenho das consultas no banco de dados orientado a grafo.

A natureza da pesquisa explicativa deste projeto, visa conectar as ideias das disciplinas de linguagens de programação, grafos e banco de dados para indicar elementos na modelagem em grafo que proporcionam melhor desempenho.

Metodologia

O desenvolvimento deste trabalho utilizou-se da base de dados Dota 2(ANZELMO, 2021) e a metodologia foi dividida em 4 etapas:

1. Modelagem Orientada a Grafos;
2. Metaprogramação para Armazenamento no Neo4j;
3. Modelagem Orientada a Objetos;
4. Metaprogramação para Armazenamento no MySQL.

A figura 1 representa o modelo orientado a grafo proposto neste trabalho para a base de dados.

Figura 1 - Diagrama do grafo proposto.

Para representar os nós do grafo da figura 1 utilizou-se um OGM (Mapeador objeto-grafo do inglês *Object Graph Mapper*) que mapeia classes elementos de grafos. Todos os 22 nós foram implementados em Java (Horstmann,2010) usando de metaprogramação por anotações para sinalizar as relações e os nós na classe, como pode ser visto no Código 1.

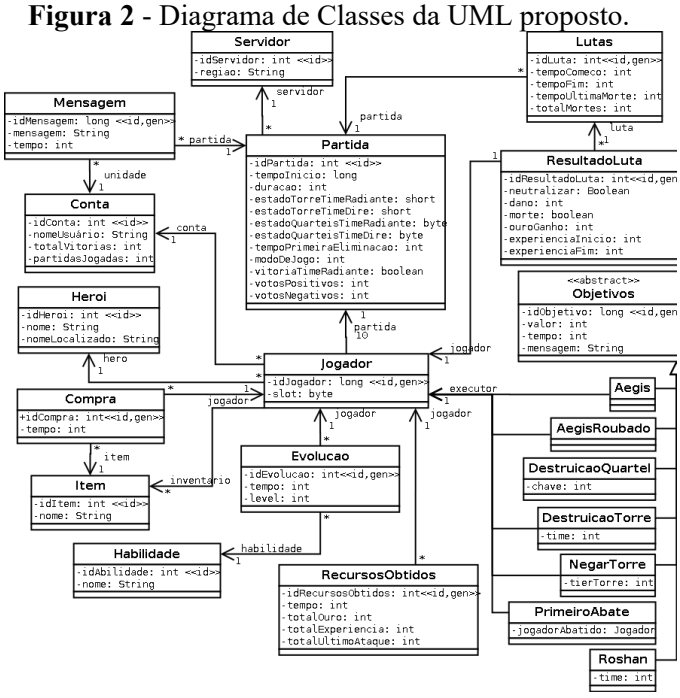
Código 1 - Metaprogramação Java para Nó Objetivos.

```
@NodeEntity @Data
public abstract class Objetivos
    implements Serializable {
  private static final long serialVersionUID =
    -3448439245119357736L;
  @Id @GeneratedValue
   private Long idObjetivo;
  @Relationship(type = "SUCEDE",
    direction = Relationship.INCOMING)
  private Partida partida;
  private Integer valor;
  private Integer tempo;
  private String mensagem;
}
```

Para realizar a carga dos dados do modelo orientado a grafos do bando de dados Neo4j ((Vukotic; et al., 2015) foi utilizado da ferramenta neo4j-admin import. Esse programa fornecido pelo próprio Neo4j ignora algumas abstrações do modelo orientado a objetos e manipula os dados de forma pura para inseri-los de forma mais rápida.

Para realizar o armazenamento em banco de dados relacional proposto o seguinte diagrama de classe em UML, que pode ser visto na Figura 2.

Figura 2 - Diagrama de Classes da UML proposto.



Para representar os objetos da figura 1 em tabelas utilizou-se um ORM (Mapeador objeto-relacional do inglês *Object Relational Mapper*). As classes foram implementadas em Java (Horstmann,2010) usando de metaprogramação por anotações para sinalizar as relações e os nós na classe (código 2).

Código 2 - Metaprogramação da Classe Objetivos.

```
@Entity @Data
public abstract class Objetivos
    implements Serializable{
  private static final long serialVersionUID =
    353678714172686910L;
  @Id @GeneratedValue
  private Long idObjetivo;
  @ManyToOne
  private Partida partida;
  private Integer valor;
  private Integer tempo;
  private String mensagem;
}
```

Para realizar a carga dos dados no bando de dados Relacional foi utilizado o mapeamento Objeto-Relacional com a biblioteca Hibernate (Bauer; King, 2006) com o banco de dados MySQL (Tahaghoghi;Williams, 2006).

Adicionalmente foi implementada uma aplicação que tem a interface visual mostrada pela figura 3.

Figura 3 - Aplicativo para Carga da Base.

Cada caixa de seleção (*chechbox*) apresenta um algoritmo de inserção que varre o arquivo (extensão .csv) da base de dados.

A Tabela 1, apresenta os arquivos, descrição e número de registros da base que são as caixas de seleção da Figura 3.

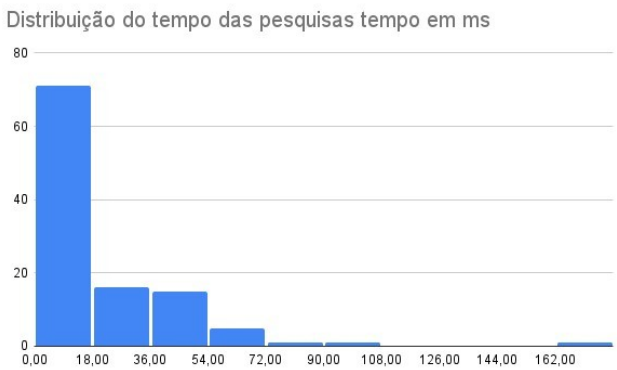
Arquivo	Descrição	Registros
cluster_ regions	Os diversos servidores e suas localizações	53
item_ids	Os itens compráveis durante a partida	189
ability_ids	As Habilidades dos diversos herois	688
hero_names	O nome dos Herois com suas traduções	112
match	As Partidas ranqueadas que foram auditadas	50.000
players	Os 10 jogadores de cada uma das partidas	500.000
teamfights	Confrontos entre jogadores	539.047
teamfights_ players	Participação dos jogadores nos confrontos	5.390.470
purchase_ log	Registro da compra dos itens	18.193.745
player_time	Recursos obtidos por jogador a cada minuto	2.209.778
ability_ upgrades	Tempo em que cada habilidade foi obtida	8.939.599
objectives	Objetivos executados	1.173.396
chat	Registro das mensagens enviadas	1.439.488

Tabela 1 – Total de Registros da Base.

Resultados e discussão

Os experimentos executados em uma máquina com a seguinte configuração: Sistema Operacional: Windows 10 Education; Processador: Core i7-9750H, 2.60 GHz, Hexa-core (6 núcleos); Memória: 8GB de RAM; Armazenamento: SSD com leitura/escrita sequencial até 545 MB/s.Depois da execução de cada consulta foram organizadas em um histograma da distribuição dos tempos de consulta como pode ser visto na Figura 4.

Figura 4 - Gráfico da execução das 221 consultas Cypher



Conclusões

Este trabalho apresentou a implementação de modelo no paradigma de programação orientado à objetos e seu mapeamento para base de dados baseado em grafos e relacional.

Análises foram realizadas usaram consultas na linguagem *Cypher* através de um gerador consultas implementadas neste trabalho.

Análise de desempenho das consultas das bases de dados orientado a grafo, mostram que os tempos foram inferiores à 1 segundo.

Como contribuições secundárias, listam-se os seguintes itens: modelo de grafo demonstrado na Figura 1; Metaprogramação Java para mapeamento Objeto-Grafo; modelo de classe da UML demonstrado na Figura 2; e Metaprogramação Java para mapeamento Objeto-Relacional.

Uma extensão para este trabalho seria comparar consultas em outras bases de dados, e utilizar outro paradigma de banco de dados, como por exemplo, orientados a documentos implementado no banco de dados mongoDB.

Agradecimento

O presente trabalho foi realizado com apoio da Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Referências

ANZELMO, D. Dota 2 Matches: Explore player behavior and predict match outcomes. 2021, Acessado em: 16/05/2021, https://www.kaggle.com/devinanzelmo/dota-2-matches.

Bauer, C.; King, G. (2006). Java Persistence with Hibernate. Manning. ISBN: 1932394885.

Elmasri, R.; Navathe, S. B.. (2018) Sistemas de Banco de Dados. Fundamentos e Aplicações. 7.ed. Pearson Education do Brasil, ISBN 9788543025001.

Horstmann, Cay S. (2010). Core Java, volume 1: fundamentos. Pearson Education do Brasil, ISBN 9788576053576.

Tahaghoghi, S.M.M.; Williams, H.E. (2006). Learning MySQL: Get a Handle on Your Data. O'Reilly Media, ISBN 9781449303969.

Vukotic, A., Watt, N., Abedrabbo, T., Fox, D., Partner, J. (2015). Neo4j in Action. Manning. ISBN: 9781617290763.