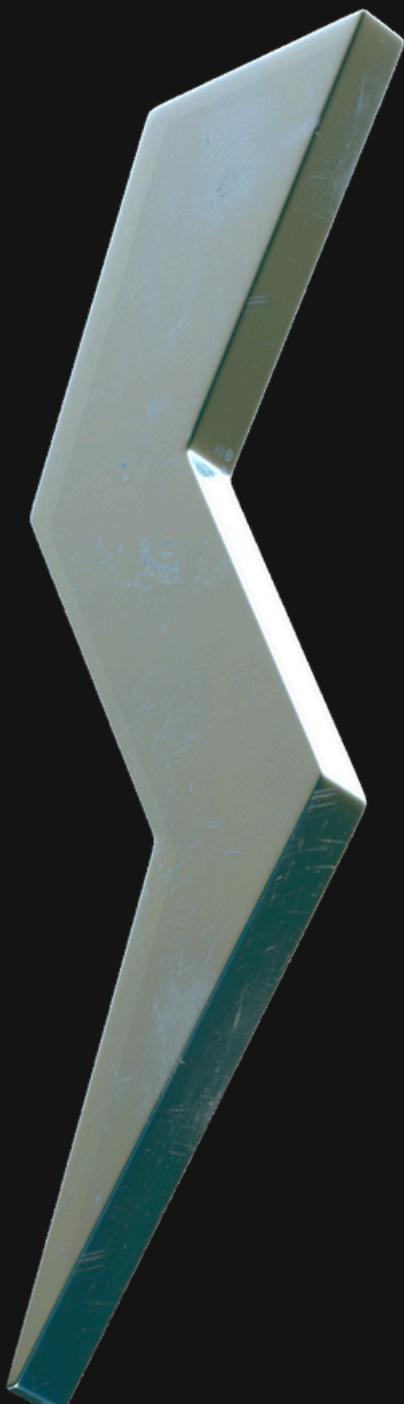


Mortal Date

DateBase



Criadores:



Flávio Albuquerque

Davio dos Santos

Breno Miguel



Projeto:

Tivemos a ideia de criar um projeto com a famosa franquia de jogos Mortal Kombat. Para isso, utilizamos a linguagem de programação Cassandra e armazenamos os dados no MySQL para gerar um banco de dados detalhado da lista de heróis.

2. TECNOLOGIAS UTILIZADAS
PARA A IMPLEMENTAÇÃO DO PROJETO, OPTAMOS POR UTILIZAR CASSANDRA E MYSQL DEVIDO
ÀS SUAS CARACTERÍSTICAS COMPLEMENTARES:

CASSANDRA: ESCOLHEMOS CASSANDRA COMO A LINGUAGEM DE PROGRAMAÇÃO DEVIDO À
SUA ROBUSTEZ E CAPACIDADE DE GERENCIAR GRANDES VOLUMES DE DADOS DE FORMA
DISTRIBUÍDA E EFICIENTE. CASSANDRA NOS PERMITIU ESTRUTURAR E PROCESSAR OS DADOS
DE FORMA ESCALÁVEL.

MYSQL: UTILIZAMOS MYSQL COMO SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS
RELACIONAL PARA ARMAZENAR A LISTA DE HERÓIS. MYSQL É CONHECIDO POR SUA
CONFIABILIDADE E FACILIDADE DE USO, FACILITANDO A ORGANIZAÇÃO E A CONSULTA
EFICIENTE DOS DADOS.



3. CRIAÇÃO DO BANCO DE DADOS DE HERÓIS

O BANCO DE DADOS FOI PROJETADO PARA INCLUIR INFORMAÇÕES DETALHADAS SOBRE CADA HERÓI DA FRANQUIA MORTAL KOMBAT. ENTRE OS DADOS ARMAZENADOS, ESTÃO:

PERSONAGENS
CATEGORIA
JOGOS

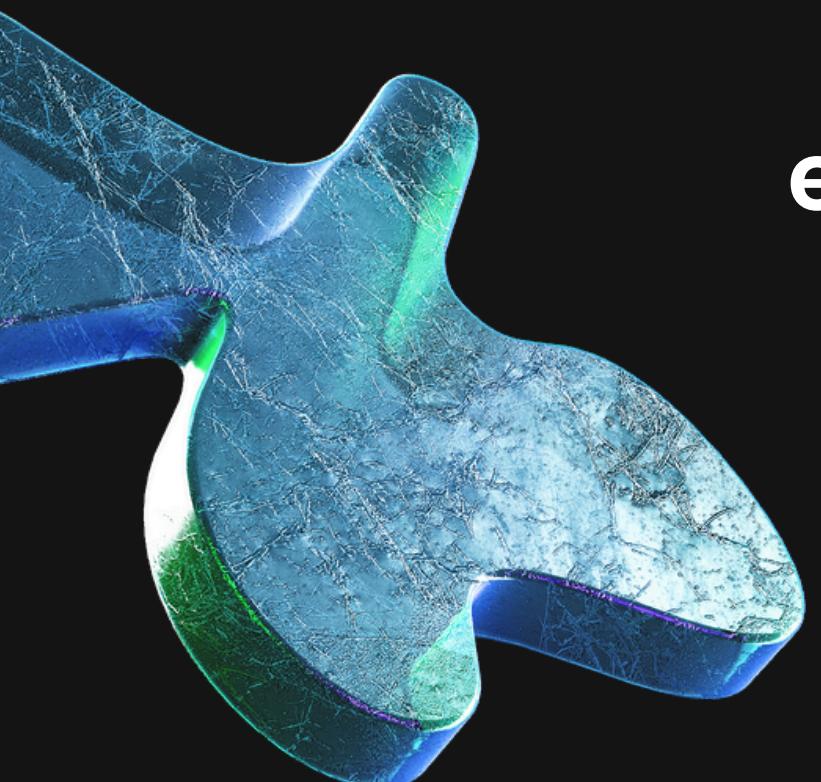


4. Integração e Resultados

A integração entre Cassandra e MySQL foi realizada de forma a aproveitar o melhor de cada tecnologia. Utilizamos Cassandra para a ingestão e processamento inicial dos dados, garantindo uma estrutura robusta e escalável. Posteriormente, esses dados foram transferidos para o MySQL, onde foram organizados em tabelas relacionais que facilitam a consulta e a análise. O resultado é um banco de dados abrangente e bem estruturado, que permite a fácil recuperação e análise de informações sobre os heróis de Mortal Kombat. Esse banco de dados pode ser utilizado para diversas finalidades, desde a criação de conteúdo adicional, como histórias e jogos, até análises detalhadas dos personagens.

5. Conclusão

A criação de um banco de dados dos heróis da franquia Mortal Kombat utilizando Cassandra e MySQL foi um projeto inovador que combinou tecnologias avançadas para atingir um objetivo claro. A escolha de Mortal Kombat como tema proporcionou um rico conjunto de dados, enquanto a utilização de Cassandra e MySQL garantiu a eficiência e a escalabilidade necessárias para gerenciar e explorar esses dados de maneira eficaz.



File Owner DB Run Export Import Client

SQLite MS SQL MariaDB PostgreSQL MS SQL 0.15.1 beta Table Categorias Jogos Personagens Personagens_Jogos

```
48
49 INSERT INTO Jogos (ID, Nome)
50 VALUES
51 (1, 'Mortal Kombat'),
52 (2, 'Mortal Kombat II');
53
54 INSERT INTO Personagens_Jogos (PersonagemID, JogoID)
55 VALUES
56 (1, 1),
57 (1, 2),
58 (2, 1),
59 (2, 2);
60
```

Plesk Sign up >

History Syntax | History

```
MS SQL 18:27:43
INSERT INTO Personagens_Jogos (Personag VALUES
(1, 1),
(1, 2),
(2, 1),
...
18:27:39
MS SQL
INSERT INTO Jogos (ID, Nome)
VALUES
(1, 'Mortal Kombat'),
(2, 'Mortal Kombat II');
```

18:27:08

MS SQL 18:27:00
CREATE TABLE Personagens (
ID INT PRIMARY KEY,
Nome VARCHAR(255),
Categoria INT,
FOR
... 18:27:00

MS SQL 18:26:19
CREATE TABLE Categorias (
ID INT PRIMARY KEY,
Nome VARCHAR(255)
); 18:26:19

MS SQL 2024 18:25:45
DROP table demo; 2024 18:25:45

sql

```
CREATE TABLE Categorias (
    ID INT PRIMARY KEY,
    Nome VARCHAR(255)
);
```

ID: Chave primária, um identificador único para cada categoria.
Nome: Nome da categoria (ex.: Lutador, Deus/Entidade, etc.).

```
sql

CREATE TABLE Personagens (
    ID INT PRIMARY KEY,
    Nome VARCHAR(255),
    Categoria INT,
    FOREIGN KEY (Categoria) REFERENCES Categorias(ID)
);
```

- **ID**: Chave primária, identificador único para cada personagem.
- **Nome**: Nome do personagem (ex.: Liu Kang, Johnny Cage, etc.).
- **Categoria**: Chave estrangeira que referencia a tabela Categorias, ligando o personagem a uma categoria específica.

```
sql
```

```
CREATE TABLE Jogos (
    ID INT PRIMARY KEY,
    Nome VARCHAR(255)
);
```

- ID: Chave primária, identificador único para cada jogo.
- Nome: Nome do jogo (ex.: Mortal Kombat, Mortal Kombat II, etc.).

sql

```
CREATE TABLE Personagens_Jogos (
    PersonagemID INT,
    JogoID INT,
    FOREIGN KEY (PersonagemID) REFERENCES Personagens(ID),
    FOREIGN KEY (JogoID) REFERENCES Jogos(ID)
);
```

- PersonagemID: Chave estrangeira que referencia a tabela Personagens, associando um personagem a um jogo específico.
- JogoID: Chave estrangeira que referencia a tabela Jogos, associando um jogo a um personagem específico.
- Esta tabela é usada para criar uma relação muitos-para-muitos entre personagens e jogos.

```
INSERT INTO Categorias (ID, Nome)
VALUES
(1, 'Lutador'),
(2, 'Deus/Entidade'),
(3, 'Edeniano'),
(4, 'Shokan'),
(5, 'Tarkatan'),
(6, 'Wraith'),
(7, 'Netherrealmiano'),
(8, 'Ciborgue'),
(9, 'Zaterran'),
(10, 'Chaosrealmer'),
(11, 'Kytinn'),
(12, 'Osh-Tekk'),
(13, 'Alienígena'),
(14, 'Outro');
```

- Inserindo diversas categorias que descrevem os tipos de personagens dentro do universo Mortal Kombat.

sql

```
INSERT INTO Personagens (ID, Nome, Categoria)
VALUES
(1, 'Liu Kang', 1),
(2, 'Johnny Cage', 1),
(3, 'Sonya Blade', 1);
```

- Inserindo personagens com seus nomes e associando-os a uma categoria específica (neste caso, todos são da categoria 'Lutador').

sql

```
INSERT INTO Jogos (ID, Nome)
VALUES
(1, 'Mortal Kombat'),
(2, 'Mortal Kombat II');
```

- Inserindo os nomes de alguns jogos da franquia.

sql

```
INSERT INTO Personagens_Jogos (PersonagemID, JogoID)
VALUES
    (1, 1),
    (1, 2),
    (2, 1),
    (2, 2);
```

- Associando personagens a jogos específicos.
Por exemplo, Liu Kang (ID 1) aparece em Mortal Kombat (ID 1) e Mortal Kombat II (ID 2).

Resumo

Este código estabelece um sistema relacional para armazenar informações sobre categorias de personagens, detalhes de personagens e jogos, e as associações entre personagens e os jogos nos quais aparecem. Utiliza chaves primárias para identificação única de registros e chaves estrangeiras para criar relacionamentos entre tabelas, permitindo consultas complexas e integridade referencial no banco de dados.