Banco de Dados NoSQL

Lieverton Silva 17/0039251
Welison Regis 17/0024121

DATASET

MODELAGEM

Tópicos:

- Base de dados escolhida;
- Diagrama ER;
- Diagrama lógico;

Base de Dados

A base de dados escolhida se trata de um dataset com filmes que tiveram lançamento anterior ao ano de 2017. O dataset é composto por filmes com descrições sobre título, gênero, visão geral, elenco, equipe, data de lançamento, duração, popularidade etc. A base de dados possui **26 milhões de registros de classificações de 270.000 usuários para os 45.000 filmes cadastrados**. As classificações estão em uma escala de 1 a 5.

Diagrama Entidade-Relacionamento

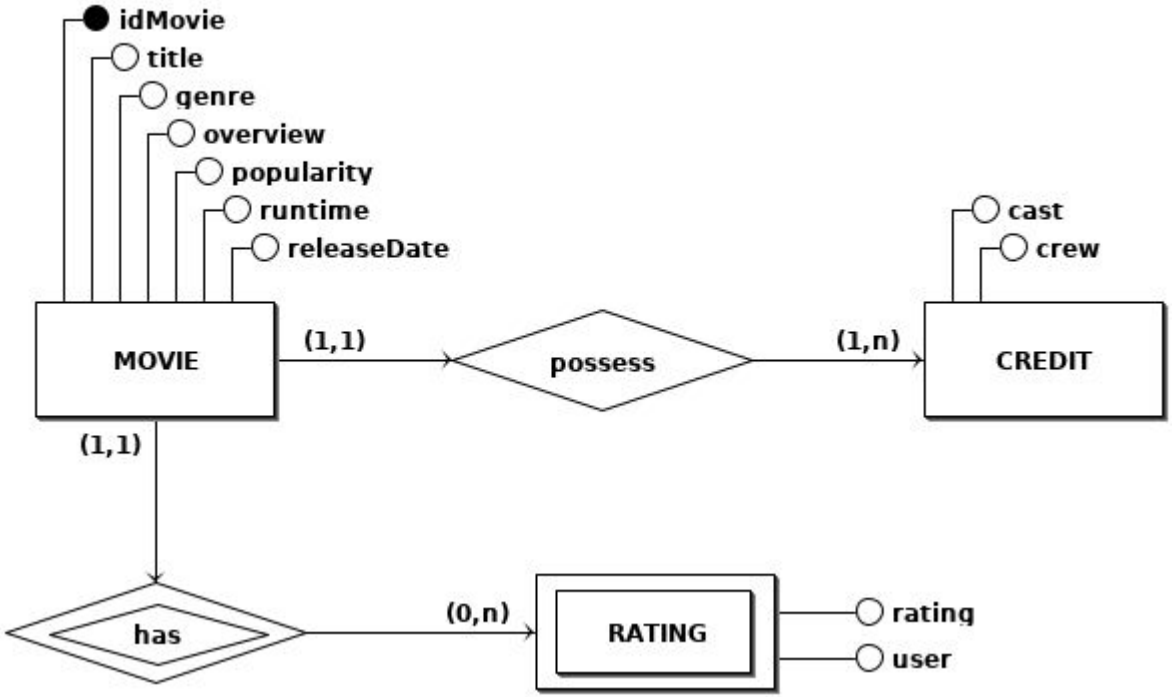


Figura 1 - Diagrama entidade-relacionamento do projeto. Fonte: do autor, 2019.

Diagrama Lógico

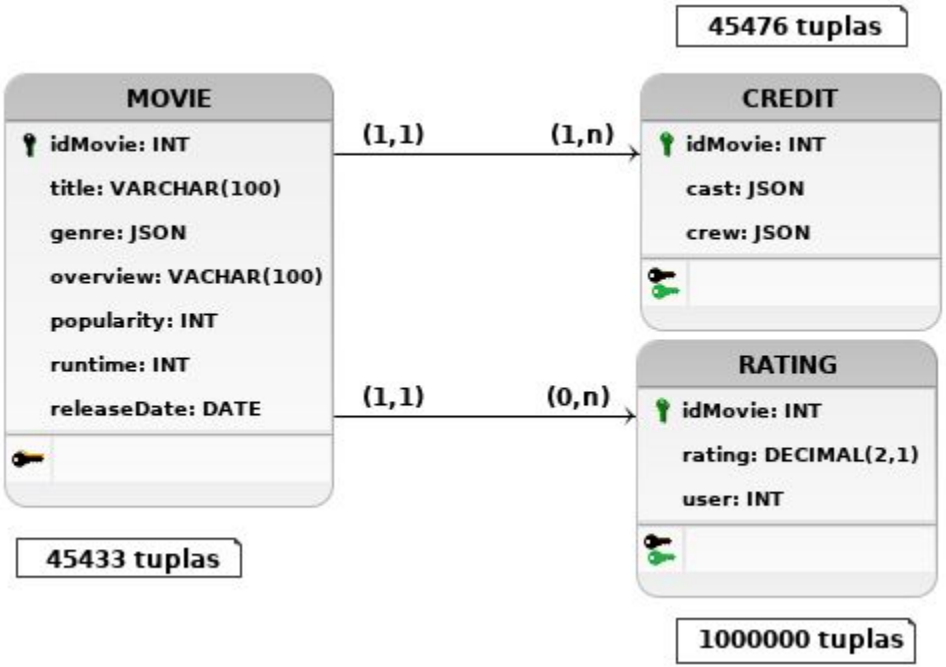


Figura 2 - Diagrama lógico do projeto. Fonte: do autor, 2019.

BASES DE DADOS

Geração das bases de dados MySQL e MongoDB

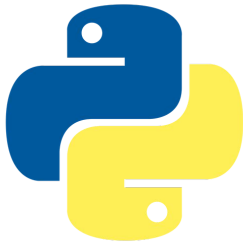
Tópicos:

- Leitura e escrita de dados;
- Bases de dados.

LEITURA E ESCRITA DE DADOS

Para a leitura e o armazenamento de dados na base, utilizou-se:

- Linguagem: **Python** no ambiente **Jupyter notebook**;
- Principais pacotes: **MySQLdb** e **MongoClient**.
- Scripts responsáveis por popular as bases MySQL e MongoDB:
  - **movie.ipynb:** popula os filmes;
  - **credits.ipynb**: popula o elenco e equipe (credits);
  - **ratings.ipynb**: popula as avaliações dos usuários.



LEITURA E ESCRITA DE DADOS

Método de execução dos scripts:

- Realiza-se a **conexão com a base de dados**;
- Os dados (csv) são lidos e tratados pelo método de classe **read_data.**
- Armazena-se os dados no MySQL pelo método de classe: **save_mysql**.
- Armazena-se os dados no MongoDB pelo método de classe: **save_mongo.**

Conexão com a base no MySQL

```
# Conexão com o MySQL
db = MySQLdb.connect("localhost","root","SENHA","tf_f_dupla1_fim")
cursor = db.cursor()
```

Figura 3 - Conexão com MySQL em python. Fonte: do autor, 2019.

Conexão com a base no MongoDB

```
# Conexão com o MongoDB
client = MongoClient()
client = MongoClient('localhost', 27017)
# Cria uma base de dados no MongoDB
db_mongo = client['tf_f_dupla1_fim']
```

Figura 4 - Conexão com MongoDB em python. Fonte: do autor, 2019.

BASES DE DADOS

Características:

- **Esquema definido**;
- Baseado em tuplas;
- Realiza commits;
- Foco nas propriedades ACID.

#	tf_f_dupla1_fim	DB Size in MB
10	tf_f_dupla1_fim	390.9

Características:

- **Esquema livre**: facilidade de inserção;
- Baseado em documentos (json);
- Busca disponibilidade do serviço;
- Foco nas propriedades BASE.

```
> show dbs;
tf_f_dupla1_fim  0.124GB
```

BASES DE DADOS

idMovie	title	genre	overview	popularity	release Date	runtime
862	Toy Story	"[{'id': 16, 'name': 'Animation'}, {'id': 35, 'name': 'Comedy'}, {'id': 10751, 'name': 'Family'}]"	Led by Woody, Andy's toys live happily in his room until [..]	21.9469430	1995-10-30	81


```
{
  '_id': ObjectId('5de40ae8d0942e0a7932d18d'),
 'genres': "[{'id': 16, 'name': 'Animation'}, {'id':
   35, 'name': 'Comedy'}, "
           "{'id': 10751, 'name': 'Family'}]",
 'id': 862,
 'overview': "Led by Woody, Andy's toys live happily
   in his room until Andy's "
             'birthday brings Buzz Lightyear onto the
               scene. Afraid of losing '
             "his place in Andy's heart, Woody plots
               against Buzz. But when "
             'circumstances separate Buzz and Woody
               from their owner, the duo '
             'eventually learns to put aside their
               differences.',
 'popularity': 21.946943,
 'release_date': '1995-10-30',
 'runtime': 81.0,
 'title': 'Toy Story'
}
```

BASES DE DADOS

Desempenho
MongoDB x NoSQL

Tópicos:

- Consultas realizadas;
- Comparação de desempenho para consulta, inserção e exclusão de dados.

DESEMPENHO EM CONSULTA SIMPLES

Realização de uma busca simples em MySQL e MongoDB:

```
# Condição de filtro na coleção de filmes
query = { "title": "Toy Story" }

# Computa o tempo de resposta para a procura acima
start_time = time.time()
result = db_mongo['movie'].find(query)
end_time = time.time()

# Apresenta os resultados de tempo e tupla recuperada
print('Tamanho da base: ', db_mongo['movie'].count_documents({}), 'documentos\n')
print('Tempo em segundos (MongoDB): ', end_time-start_time, '\n')
for value in result:
    print(value)
```

```
Tamanho da base:  45433 documentos

Tempo em segundos (MongoDB):  0.0001697540283203125

{'_id': ObjectId('5de40ae8d0942e0a7932d18d'), 'genres': "[{'id': 1
6, 'name': 'Animation'}, {'id': 35, 'name': 'Comedy'}, {'id': 1075
1, 'name': 'Family'}]", 'id': '862', 'overview': "Led by Woody, An
dy's toys live happily in his room until Andy's birthday brings Bu
zz Lightyear onto the scene. Afraid of losing his place in Andy's
heart, Woody plots against Buzz. But when circumstances separate B
uzz and Woody from their owner, the duo eventually learns to put a
side their differences.", 'popularity': 21.946943, 'release_date':
'1995-10-30', 'runtime': 81.0, 'title': 'Toy Story'}
```

```
# Computa o tempo de resposta para o select abaixo
start_time = time.time()
cursor.execute("SELECT * FROM MOVIE WHERE title='Toy Story';")
result = cursor.fetchall()
end_time = time.time()

# Conta o número de filmes na base
cursor.execute('SELECT COUNT(*) FROM MOVIE;')
movies_quantity = cursor.fetchall()

# Apresenta os resultados de tempo e o documento recuperado
print('Tamanho da base: ', *movies_quantity, 'tuplas\n')
print('Tempo em segundos (MySQL): ', end_time-start_time, '\n')
for value in result:
    print(x)
```

```
Tamanho da base:  (45433,) tuplas

Tempo em segundos (MySQL):  0.04988598823547363

(862, 'Toy Story', '"[{\'id\': 16, \'name\': \'Animation\'}, {\'id
\': 35, \'name\': \'Comedy\'}, {\'id\': 10751, \'name\': \'Family
\'}]"', "Led by Woody, Andy's toys live happily in his room until
Andy's birthday brings Buzz Lightyear onto the scene. Afraid of lo
sing his place in Andy's heart, Woody plots against Buzz. But when
circumstances separate Buzz and Woody from their owner, the duo ev
entually learns to put aside their differences.", Decimal('21.9469
430'), datetime.date(1995, 10, 30), 81)
```

DESEMPENHO EM ESCRITA DE DADOS

Testou-se a aplicação no seguinte cenário:

- Inserção de **1 milhão de tuplas** com **3 atributos cada**.

Possíveis causas da diferença:

- Integridade de chaves;
- Constraints;
- Commits.

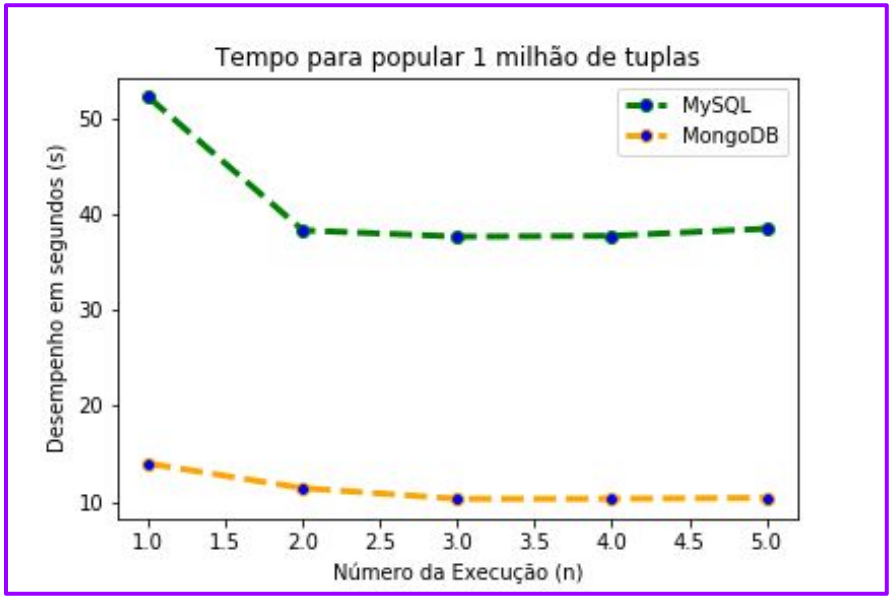


Figura 4 - Desempenho em Inserção de dados. Fonte: do autor, 2019.

```
cursor.executemany(self.insert, dataframe)
db.commit()
```

```
ratings = db_mongo['rating']
ratings.insert_many(json_ratings)
```

DESEMPENHO EM DELETAR DADOS

Verificou-se o desempenho dos bancos MySQL e MongoDB no seguinte cenários:

- Exclusão de **1 milhão de tuplas** com **3 atributos cada**.

Resultados : MongoDB performou 5x mais rápido.

```
print('Tamanho da base: ', db_mongo['rating'].count_documents({}), 'documentos\n')

# Computa o tempo para excluir as tuplas de rating
start_time = time.time()
db_mongo['rating'].remove({})
end_time = time.time()

# Tempo para excluir 1 milhão de tuplas
print('Tempo em segundos (MongoDB): ', end_time-start_time, '\n')
```

```
Tamanho da base:  1000000 documentos


Tempo em segundos (MongoDB):  6.839916229248047
```

```
cursor.execute('SELECT COUNT(*) FROM RATING;')
ratings = cursor.fetchall()
print('Tamanho da base: ', *ratings, 'tuplas\n')

# Computa o tempo para excluir as tuplas de rating
start_time = time.time()
cursor.execute("DELETE FROM RATING;")
end_time = time.time()

# Tempo para excluir 1 milhão de tuplas
print('Tempo em segundos (MySQL): ', end_time-start_time, '\n')
```

```
Tamanho da base:  (1000000,) tuplas

Tempo em segundos (MySQL):  32.65891408920288
```

CONSULTAS ESTRATÉGICAS

Consultas na base de dados

Tópico:

- Consultas estratégicas feitas em MySQL.

CONSULTAS ESTRATÉGICAS - MYSQL

Consulta 1: Tem como objetivo oferecer ao usuário o nome dos **10 filmes mais bem avaliados**. Para entrar nesse ranking o filme deve ter sido avaliado no mínimo 100 vezes.

```
CREATE OR REPLACE VIEW v_MOVIE_RATING AS
SELECT title, sumRating/`number of ratings` AS rating, `number of ratings`
   FROM MOVIE AS M
      INNER JOIN
      (
         SELECT idMovie, count(idMovie) AS `number of ratings`, sum(rating) AS
sumRating
            FROM RATING
            GROUP BY idMovie
      ) AS R
         ON M.idMovie = R.idMovie
   WHERE `number of ratings` >= 100
   ORDER BY rating DESC, `number of ratings` DESC;

SELECT * FROM v_MOVIE_RATING LIMIT 10;
```

Tempo de execução:

- 4,50 segundos

CONSULTAS ESTRATÉGICAS - MYSQL

Consulta 1: Tem como objetivo oferecer ao usuário o nome dos **10 filmes mais bem avaliados**. Para entrar nesse ranking o filme deve ter sido avaliado no mínimo 100 vezes.

title	rating	number of ratings
The Million Dollar Hotel	4.43993	7924
Sleepless in Seattle	4.33034	4963
Hard Target	4.28089	1230
Once Were Warriors	4.27259	5921
Baise-moi	4.24675	154
A Woman, a Gun and a Noodle Shop	4.24011	758
License to Wed	4.23986	5253
Torrente 2: Mission in Marbella	4.23626	546
Dead Man	4.22214	664
The Thomas Crown Affair	4.21737	2551

10 rows in set (4,50 sec)

Figura 5 - Filmes mais bem avaliados Fonte: do autor, 2019.

CONSULTAS ESTRATÉGICAS - MYSQL

Consulta 2: Tem como objetivo oferecer ao usuário o nome dos **20 filmes mais populares, além das respectivas avaliações**.

```
CREATE INDEX popularity_idx ON MOVIE (popularity);

SELECT M.title, popularity, rating
   FROM MOVIE AS M
      INNER JOIN v_MOVIE_RATING AS V
         ON M.title = V.title
   ORDER BY popularity DESC, rating DESC
   LIMIT 20;
```

Tempo de execução:

- 3,20 segundos

CONSULTAS ESTRATÉGICAS - MYSQL

Consulta 2: Tem como objetivo oferecer ao usuário o nome dos **20 filmes mais populares, além das respectivas avaliações**.

title	popularity	rating
Pulp Fiction	140.9502360	3.52174
The Dark Knight	123.1672590	3.43913
Blade Runner	96.2723740	3.24561
Fight Club	63.8695990	3.08871
The Shawshank Redemption	51.6454030	2.99010
Forrest Gump	48.3071940	3.38924
Pirates of the Caribbean: The Curse of the Black Pearl	47.3266650	3.34627
Star Wars	42.1496970	3.67330
Schindler's List	41.7251230	2.63014
The Godfather	41.1092640	3.00000
Spirited Away	41.0488670	3.27778
Life Is Beautiful	39.3949700	2.95146
Harry Potter and the Philosopher's Stone	38.1872380	3.62061
Avengers: Age of Ultron	37.3794200	3.00000
Psycho	36.8263090	3.54089
The Godfather: Part II	36.6293070	2.80435
One Flew Over the Cuckoo's Nest	35.5295540	2.81757
The Matrix	33.3663320	2.82759
Once Upon a Time in America	32.1828510	2.11111
The Lord of the Rings: The Fellowship of the Ring	32.0707250	3.02778

20 rows in set (3,20 sec)

Figura 6 - Filmes mais populares. Fonte: do autor, 2019.

CONSULTAS ESTRATÉGICAS - MYSQL

Consulta 3: Tem como objetivo oferecer ao usuário o nome dos **10 filmes melhores avaliados com a participação de um determinado ator.**

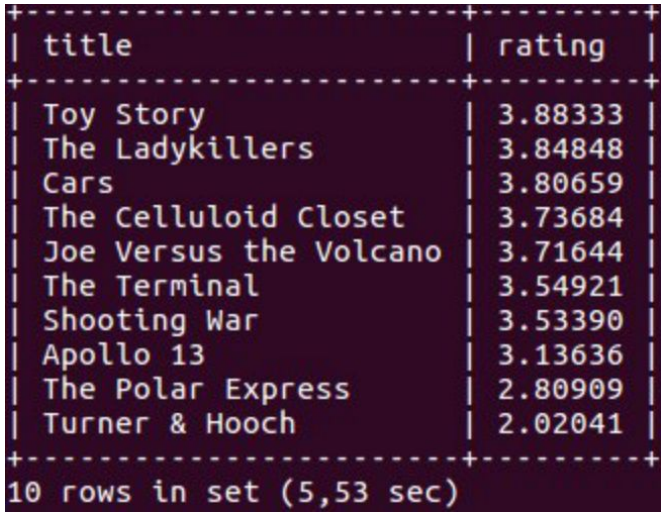
```
SELECT title, rating
   FROM CREDIT AS C
      LEFT JOIN v_MOVIE_RATING AS V
         ON C.idMovie = V.idMovie
   WHERE JSON_SEARCH(cast, 'one', "Tom Hanks") IS NOT NULL
   ORDER BY rating DESC
   LIMIT 10;
```

Tempo de execução:

- 5,53 segundos

CONSULTAS ESTRATÉGICAS - MYSQL

Consulta 3: Tem como objetivo oferecer ao usuário o nome dos **10 filmes melhores avaliados com a participação de um determinado ator.**



title	rating
Toy Story	3.88333
The Ladykillers	3.84848
Cars	3.80659
The Celluloid Closet	3.73684
Joe Versus the Volcano	3.71644
The Terminal	3.54921
Shooting War	3.53390
Apollo 13	3.13636
The Polar Express	2.80909
Turner & Hooch	2.02041

10 rows in set (5,53 sec)

Figura 7 - Filmes mais populares do ator Tom Hanks. Fonte: do autor, 2019.