**HỌC VIỆN KỸ THUẬT QUÂN SỰ**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO MÔN HỌC**

**NHẬP MÔN CƠ SỞ DỮ LIỆU LỚN**

**ĐỀ TÀI:...................**

|  |  |
| --- | --- |
| **Sinh viên thực hiện:** | **Phan Trung Kiên** |
|  | **Nguyễn Xuân Sơn** |

*Hà Nội, 12/2020*

# **PHÂN CÔNG CÔNG VIỆC**

|  |  |
| --- | --- |
| **Công việc** | **Sinh viên thực hiện** |
| Xây dựng base front end | Nguyễn Xuân Sơn |
| Xây dựng base back end | Phan Trung Kiên, Nguyễn Xuân Sơn |
| Xây dựng chức năng CRUD quản lý thông tin phim | Nguyễn Xuân Sơn |
| Xây dựng chứng năng vote phim + Tích hợp học máy gợi ý phim | Phan Trung Kiên |
| Trực quan hóa dữ liệu | Phan Trung Kiên |
| Phan Tích chất lượng dữ liệu | Nguyễn Xuân Sơn , Phan Trung Kiên |
| Đề xuất giải pháp làm sạch dữ liệu | Phan Trung Kiên, Nguyễn Xuân Sơn |

# **Xây dựng chức năng CRUD sử dụng hệ quản trị cơ sở dữ liệu MongoDB**

## **Tổng quan về MongoBD**

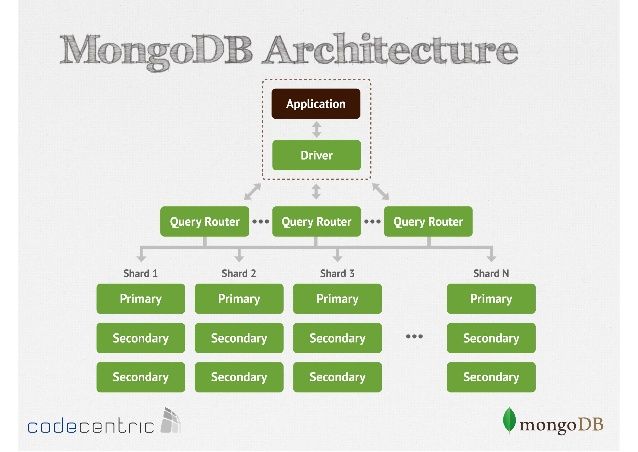
NoSQL là 1 dạng CSDL mã nguồn mở không sử dụng Transact-SQL để truy vấn thông tin. NoSQL viết tắt bởi: None-Relational SQL, hay có nơi thường gọi là Not-Only SQL. CSDL này được phát triển trên Javascript Framework với kiểu dữ liệu JSON. (Cú pháp của JSON là “key:value”) NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu xót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS về tốc độ, tính năng, khả năng mở rộng, memory cache,..*.*

MongoDB là một cơ sở dữ liệu mã nguồn mở và là cơ sở dữ liệu NoSQL hàng đầu, được hàng triệu người sử dụng. MongoDB được viết bằng C++. MongoDB là một cơ sở dữ liệu đa nền tảng, hoạt động trên các khái niệm Collection và Document, nó cung cấp hiệu suất cao, tính khả dụng cao và khả năng mở rộng dễ dàng.

**Các thuật ngữ sử dụng trong MongoDB:**

* **\_id** – Là trường bắt buộc có trong mỗi document. Trường \_id đại diện cho một giá trị duy nhất trong document MongoDB. Trường \_id cũng có thể được hiểu là khóa chính trong document. Nếu bạn thêm mới một document thì MongoDB sẽ tự động sinh ra một \_id đại diện cho document đó và là duy nhất trong cơ sở dữ liệu MongoDB.
* **Collection** – Là nhóm của nhiều document trong MongoDB. Collection có thể được hiểu là một bảng tương ứng trong cơ sở dữ liệu RDBMS (Relational Database Management System). Collection nằm trong một cơ sở dữ liệu duy nhất. Các collection không phải định nghĩa các cột, các hàng hay kiểu dữ liệu trước.
* **Cursor** – Đây là một con trỏ đến tập kết quả của một truy vấn. Máy khách có thể lặp qua một con trỏ để lấy kết quả.
* **Database** – Nơi chứa các Collection, giống với cơ sở dữ liệu RDMS chúng chứa các bảng. Mỗi Database có một tập tin riêng lưu trữ trên bộ nhớ vật lý. Một mấy chủ MongoDB có thể chứa nhiều Database.
* **Document** – Một bản ghi thuộc một Collection thì được gọi là một Document. Các Document lần lượt bao gồm các trường tên và giá trị.
* **Field** – Là một cặp name – value trong một document. Một document có thể có không hoặc nhiều trường. Các trường giống các cột ở cơ sở dữ liệu quan hệ.
* **JSON** – Viết tắt của JavaScript Object Notation. Con người có thể đọc được ở định dạng văn bản đơn giản thể hiện cho các dữ liệu có cấu trúc. Hiện tại JSON đang hỗ trợ rất nhiều ngôn ngữ lập trình.
* **Index** – Là những cấu trúc dữ liệu đặc biệt, dùng để chứa một phần nhỏ của các tập dữ liệu một cách dễ dàng để quét. Chỉ số lưu trữ giá trị của một fields cụ thể hoặc thiết lập các fields, sắp xếp theo giá trị của các fields này. Index hỗ trợ độ phân tích một cách hiệu quả các truy vấn. Nếu không có chỉ mục, MongoDB sẽ phải quét tất cả các documents của collection để chọn ra những document phù hợp với câu truy vấn. Quá trình quét này là không hiệu quả và yêu cầu MongoDB để xử lý một khối lượng lớn dữ liệu.

**Hoạt động của MongoDB:**



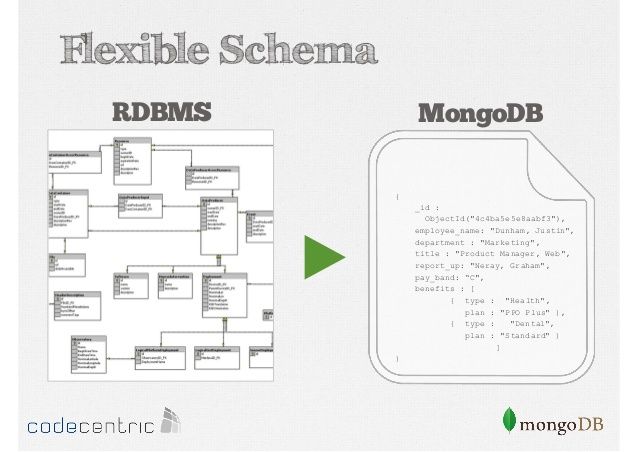
* MongoDB hoạt động dưới một tiến trình ngầm service, luôn mở một cổng (Cổng mặc định là 27017) để lắng nghe các yêu cầu truy vấn, thao tác từ các ứng dụng gửi vào sau đó mới tiến hành xử lý.
* Mỗi một bản ghi của MongoDB được tự động gắn thêm một field có tên “\_id” thuộc kiểu dữ liệu ObjectId mà nó quy định để xác định được tính duy nhất của bản ghi này so với bản ghi khác, cũng như phục vụ các thao tác tìm kiếm và truy vấn thông tin về sau. Trường dữ liệu “\_id” luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất.
* Mỗi khi có một truy vấn dữ liệu, bản ghi được cache (ghi đệm) lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.
* Khi có yêu cầu thêm/sửa/xóa bản ghi, để đảm bảo hiệu suất của ứng dụng mặc định MongoDB sẽ chưa cập nhật xuống ổ cứng ngay, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng.

**Từ đây có thể nhìn thấy *nhược điểm* của Mongodb như sau:**

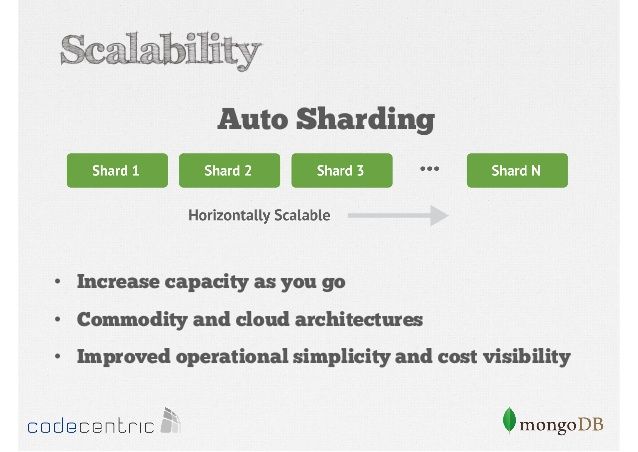
* Dữ liệu được caching, lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn
* Như đã giới thiệu ở trên, mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ổ cứng ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao.

**Ưu điểm của MongoDB:**

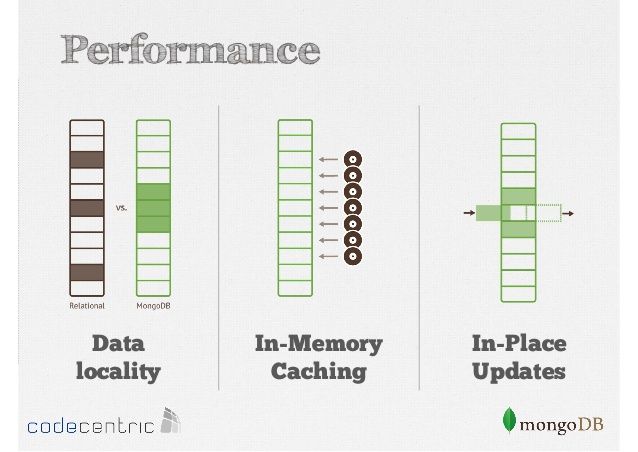
* + Ít schema hơn: Vì schema được sinh ra là để nhóm các đối tượng vào 1 cụm, dễ quản lý. Ví dụ như tạo 1 schema tên là Students chẳng hạn thì chỉ có những gì liên quan đến student thì mới được cho vào schema này. Trong khi đó trong mongodb thì chỉ 1 collection ta có thể chứa nhiều document khác nhau . Với mỗi document thì số trường, nội dung, kích thước lại có thể khác nhau.



* Cấu trúc của một đối tượng rõ ràng.
* Không có các Join phức tạp.
* Khả năng mở rộng cực lớn: việc mở rộng dữ liệu mà không phải lo đến các vấn đề như khóa ngoại, khóa chính, kiểm tra ràng buộc, ... MongoDB cho phép thực hiện replication và sharding nên việc mở rộng cũng thuận lợi hơn.

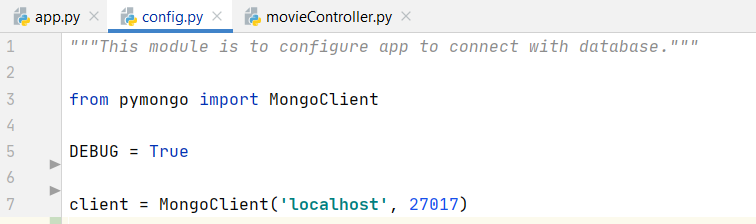


* Sử dụng bộ nhớ trong để lưu giữ cửa sổ làm việc cho phép truy cập dữ liệu nhanh hơn. Việc cập nhật được thực hiện nhanh gọn nhờ update tại chỗ (in-place).

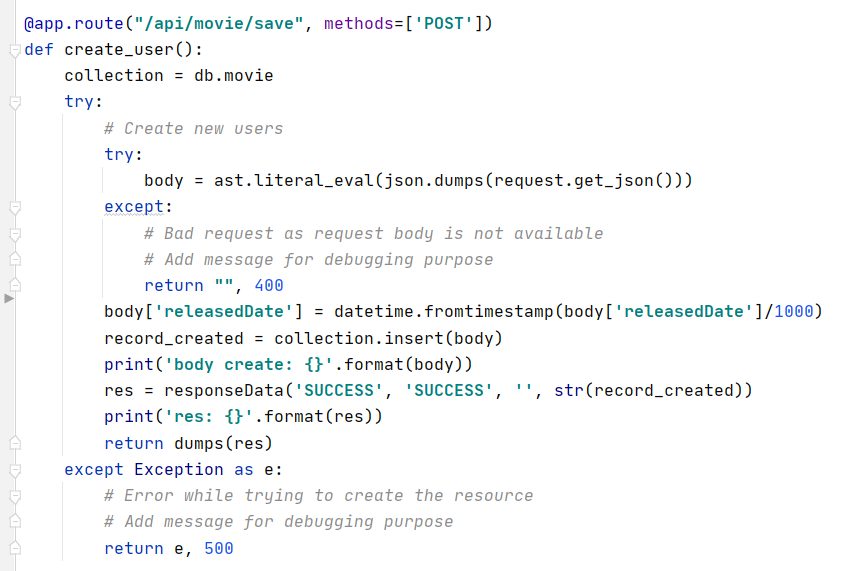


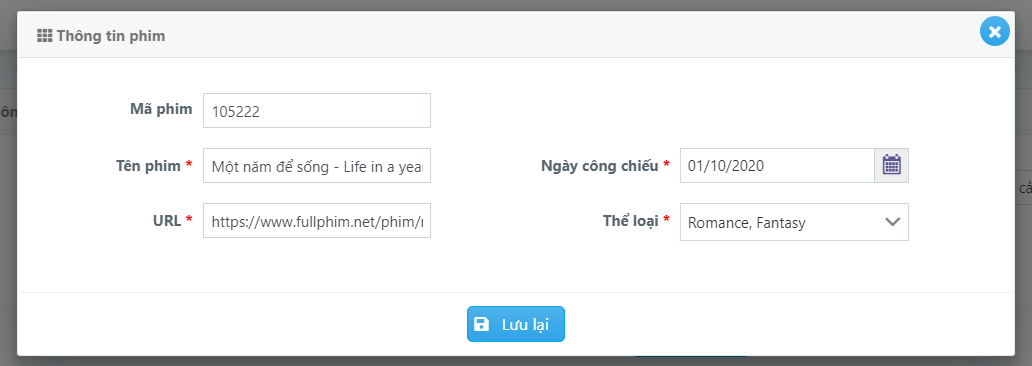
## **Xây dựng chức năng quản lý thông tin phim sử dụng hệ quản trị cơ sở dữ liệu MongoDB kết hợp với ngôn ngữ lập trình Python**

* Cấu hình kết nối tới cơ sở dữ liệu MongoDB

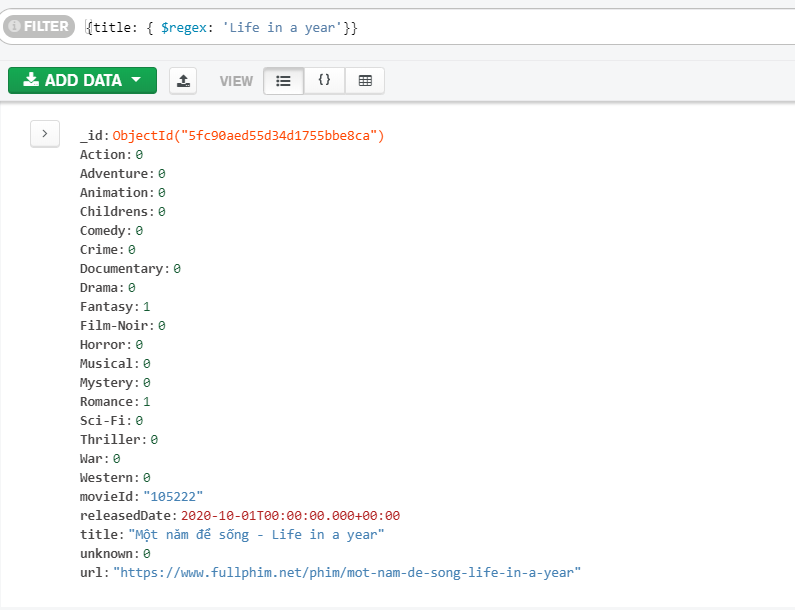


* Chức năng thêm mới dữ liệu



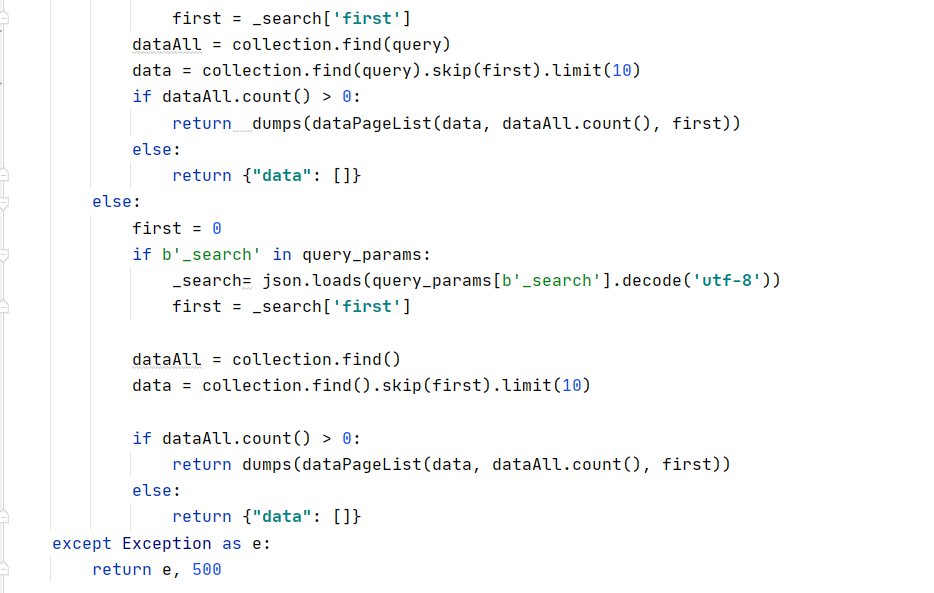


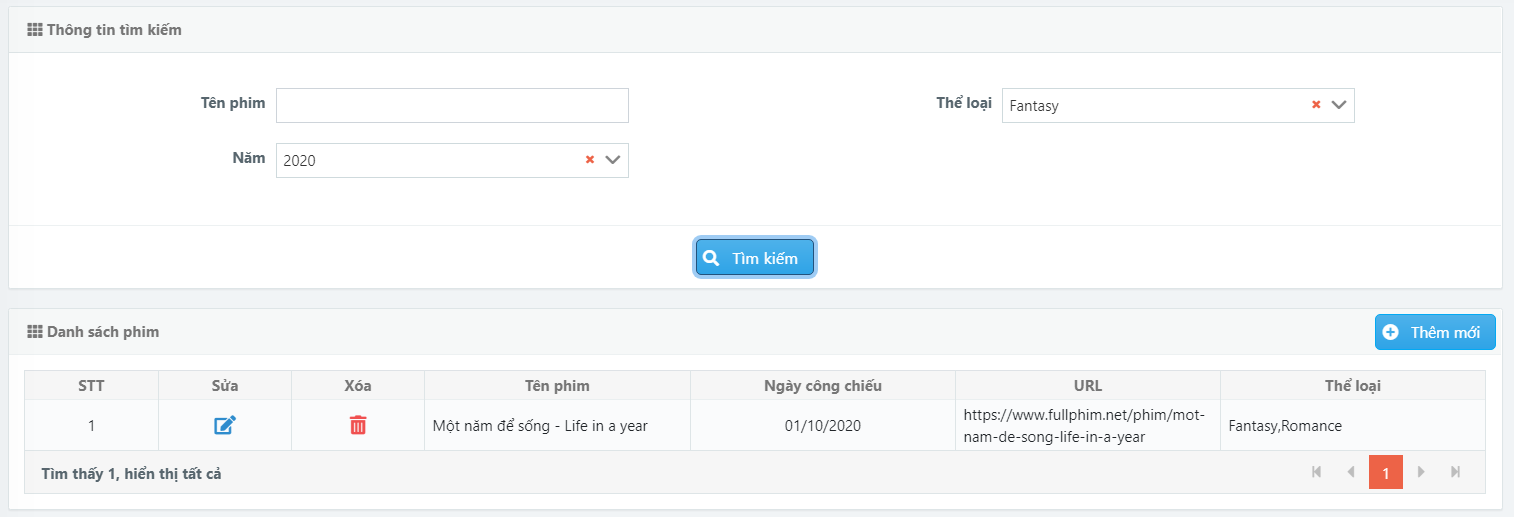
Dữ liệu được lưu vào cơ sở dữ liệu



* **Chức năng tìm kiếm/lọc dữ liệu**



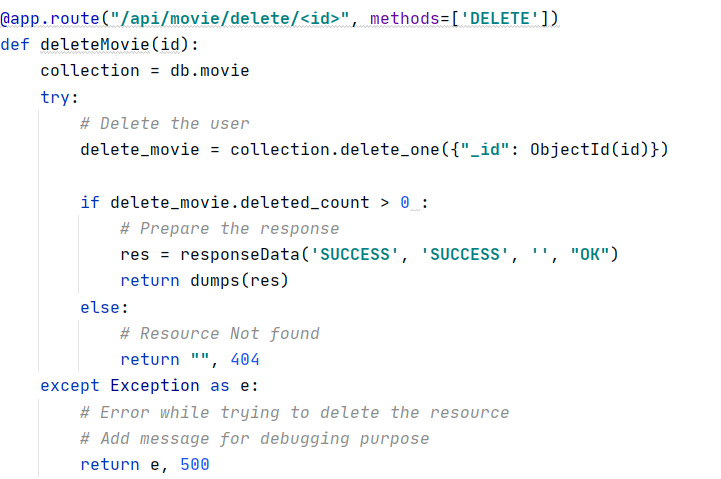




* **Chức năng cập nhật thông tin phim**



* **Chức năng xóa phim**



# **Tích hợp thư viện học máy thực gợi ý**

**1. Mô tả bài toán**

* Bài toán: Xây dựng hệ thống dự đoán phim cho người dùng dựa vào cơ sở dữ liệu MovieLens 100k và hướng tiếp cận lọc theo nội dung (Content-based filtering)
* Mô tả tập dữ liệu: Bộ cơ sở dữ liệu MovieLens 100k được công bố năm 1998 bởi GroupLens. Bộ cơ sở dữ liệu này bao gồm 100,000 (100k) ratings từ 943 users cho 1682 bộ phim. Trong đó:

+ ua.base, ua.test: là 2 tập dữ liệu cho training, một cho test. Gồm các cột thông tin lần lượt là: id user, id phim, điểm rating, thời gian phát hành.

+ u.item: thông tin về mỗi bộ phim, mỗi thông tin ngăn cách bới dấu “|” bao gồm: id phim, tên phim, ngày công chiếu, URL, và 19 thể loại.

**2. Kỹ thuật sử dụng**

a, Mô hình Hồi quy tuyến tính

- Giả sử, với mỗi user, ta có thể tìm được mộ mô hình cho mỗi user, minh họa bởi vector cột hệ số w và độ chênh lệch b sao cho rating (y) của user tới một item (x) được tính bằng một hàm tuyến tính:

Y = x\*w+b

- Với mỗi User, rating n bộ phim. Ta có hàm trung bình sai số của rating thật so với rating dự đoán là:

🡺 tham số w, b phù hợp nhất cho hàm tuyến tính chính là tham số sao cho hàm trung bình MSE có giá trị nhỏ nhất.

🡺 MSE nhỏ nhất khi đạo hàm của nó bằng 0.

🡺 Tính đạo hàm của MSE theo:

- w:

- b:

🡺 Áp dụng phương pháp Gradient descent cho hai đạo hàm trên để tìm tham số w, b phù hợp:

+ Khởi tạo w, b ngẫu nhiên, α, ε > 0 đủ nhò.

+ Lặp:

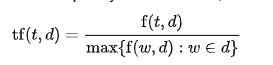
- w = w – α\*df/dw

- b = b – b\*df/db

+ Đến khi |df/dw| < ε và |df/db| < ε thì dừng lặp và w và b khi đó là hai tham số cần tìm.

b, TF-IDF: viết tắt của thuật ngữ tiếng Anh term frequency – inverse document frequency, của một từ là một con số thu được qua thống kê thể hiện mức độ quan trọng của từ này trong một văn bản, mà bản thân văn bản đang xét nằm trong một tập hợp các văn bản, trong đó:

**- TF- term frequency** – tần số xuất hiện của 1 từ trong 1 văn bản. Cách tính:



* **IDF** – *inverse document frequency.* Tần số nghịch của 1 từ trong tập văn bản (corpus)



- Giá trị **TF-IDF**:



🡺 Từ tập dữ liệu u.item ta có thể lọc ra được ma trận thể loại phim từ 19 giá trị nhị phân ở cuối mỗi dòng

🡺 Ma trận này chính là tần số xuất hiện của thể loại ở mỗi phim

🡺 Áp dụng TF-IDF cho ma trận này ta được feature vector cho mỗi item. Vector gồm 19 chiều tương ứng với 19 thể loại, độ dài mỗi chiều biểu thị cho mức độ ảnh hưởng của thể loại đó.

**3. Xây dựng mô hình**

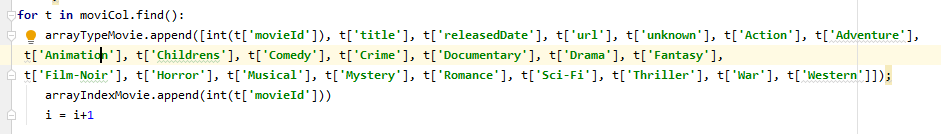
- B1: Tiền xử lý dữ liệu: Do tập data này đã được làm sạch nên sẽ không cần qua bước làm sạch hay tiền xử lý dữ liệu nữa.

- B2: Đọc dữ liệu đầu vào.

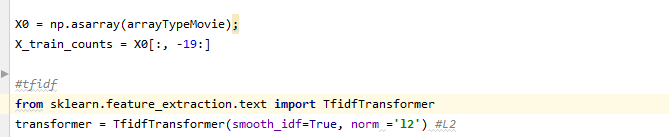


- B3: Xây dựng item profile chính là mà trận thể loại phim

+ Lấy thông tin phim kèm thể loại từ db

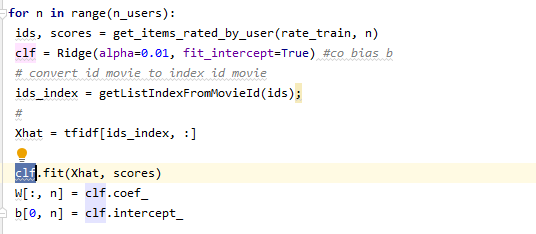


+ Xây dựng ma trận tf-idf

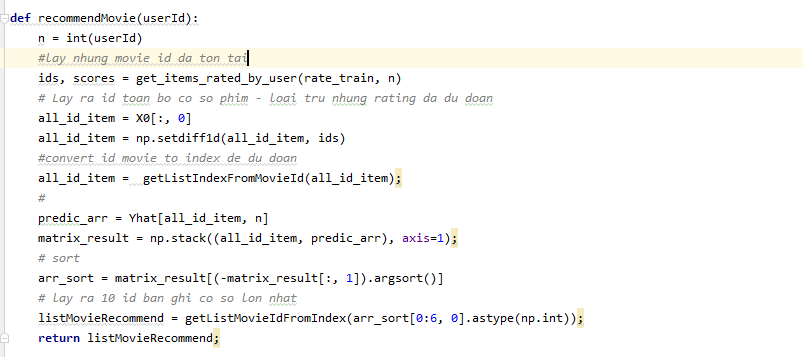


🡺 Mỗi hàng của **tfidf** tương ứng với feature vector của một bộ phim.

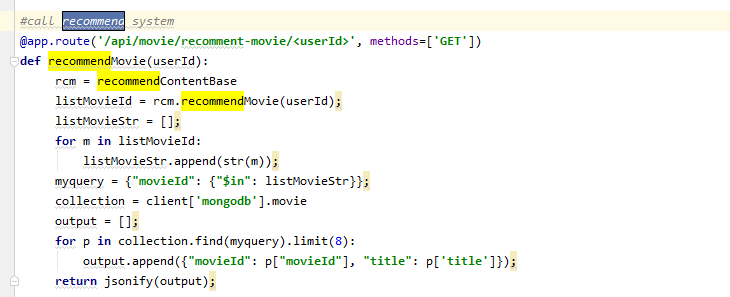
- B4: Tìm mô hình cho mỗi user.



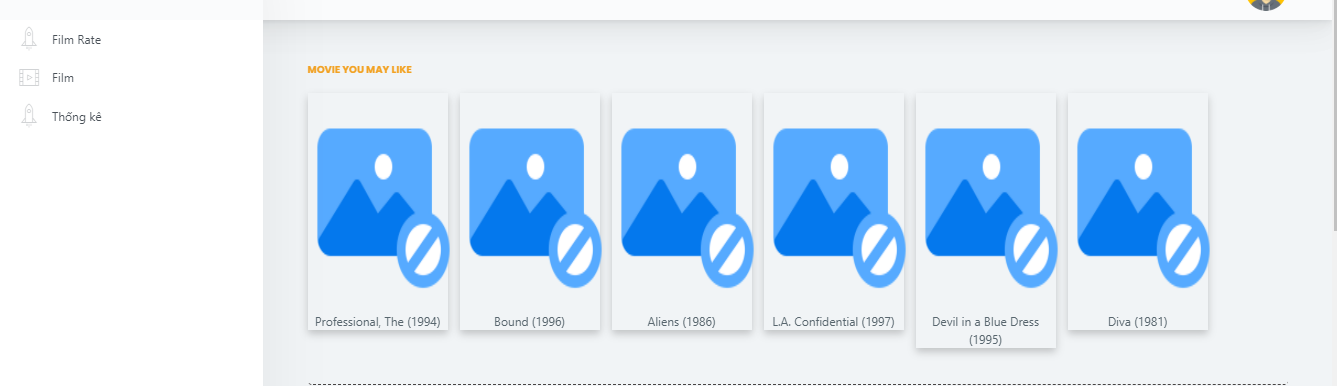
- B5: Xây dựng hàm dự đoán rate phim từ mô hình vừa xây dựng 🡺 Lấy ra top 6 bộ phim có rate cao nhất trả về:



**-** B6: Xây dựng API gọi ra hàm dự đoán:

****

* **Kết quả:**

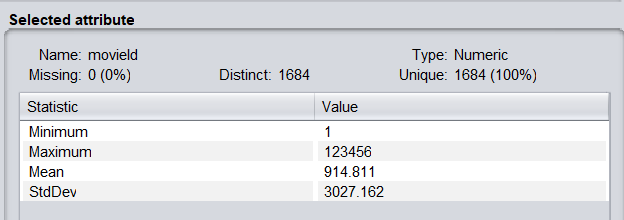
****

# **Phân tích chất lượng dữ liệu**

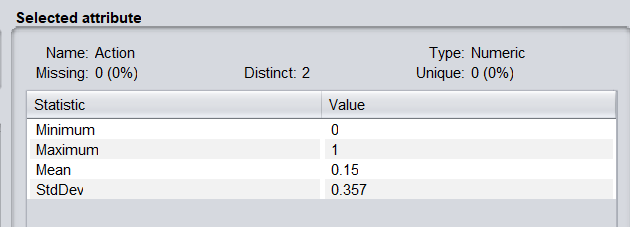
# \* Sử dụng WEKA, lần lượt mở 2 file dữ liệu movie chứa thông tin phim và rate chứa thông tin rate phim.

# - Với dữ liệu thông tin phim (movie), ta được:

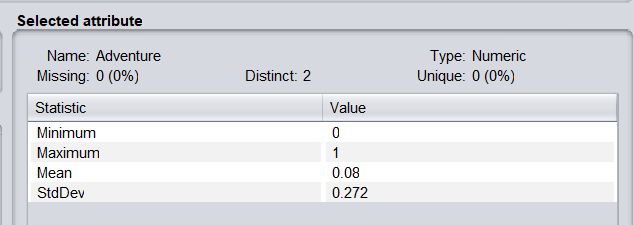
+ Cột movieId

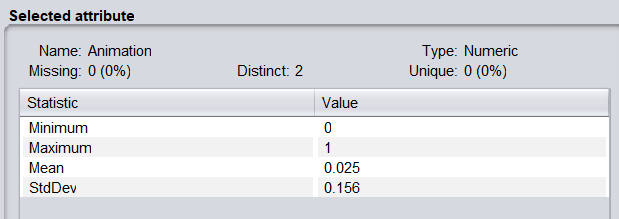


+ Cột Action

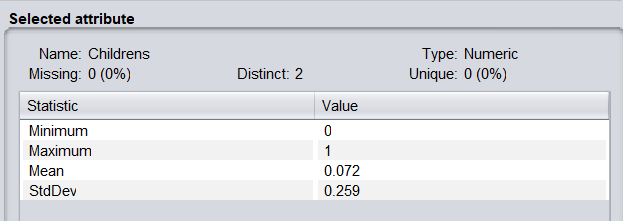


+ Cột Adventure

  
+ Cột Animation



+ Cột Childrens



🡺 Các cột dữ liệu đều không có giá trị bị mất (Missing = 0%)

🡺 Dữ liệu của các cột thể loại đều nằm ở giá trị hợp lệ là 0 hoặc 1 (Min: 0. Max: 1)

# - Với dữ liệu rating, ta được thông tin các cột: + Cột userId

# 

# + Cột movieId

# 

+ Cột rating:

# 

# **🡺 Dữ liệu ở các cột đều không bị mất ( Missing = 0%)**

# **🡺 Dữ liệu ở cột rate đều nằm trong phạm vi hợp lệ là từ 1 đến 5 (Min: 1; Max: 5)**

# **Tích hợp công cụ trực quan hóa dữ liệu.**

\* Tiến hành viết các API thống kê trả về số liệu và sử dụng module ng-char của Angular 7 để trực quan hóa dữ liệu về:

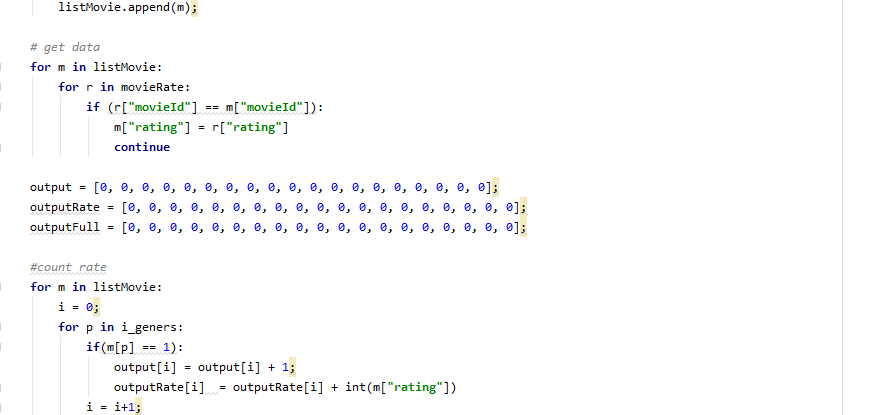
- Số lượng rate của user login theo từng thể loại phim

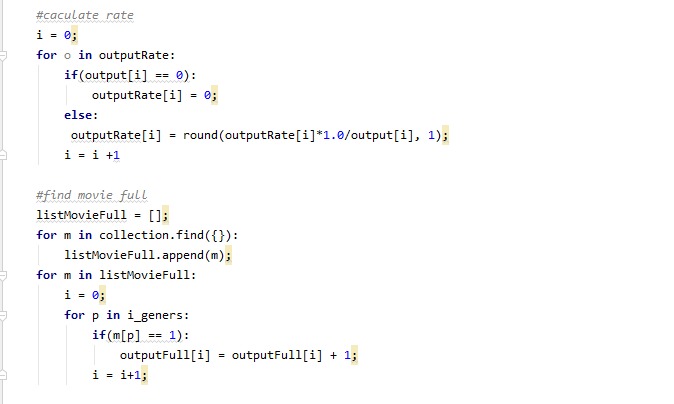
- Giá trị trung bình rating theo từng thể loại phim của user login

- Tỉ lệ phim theo thể loại

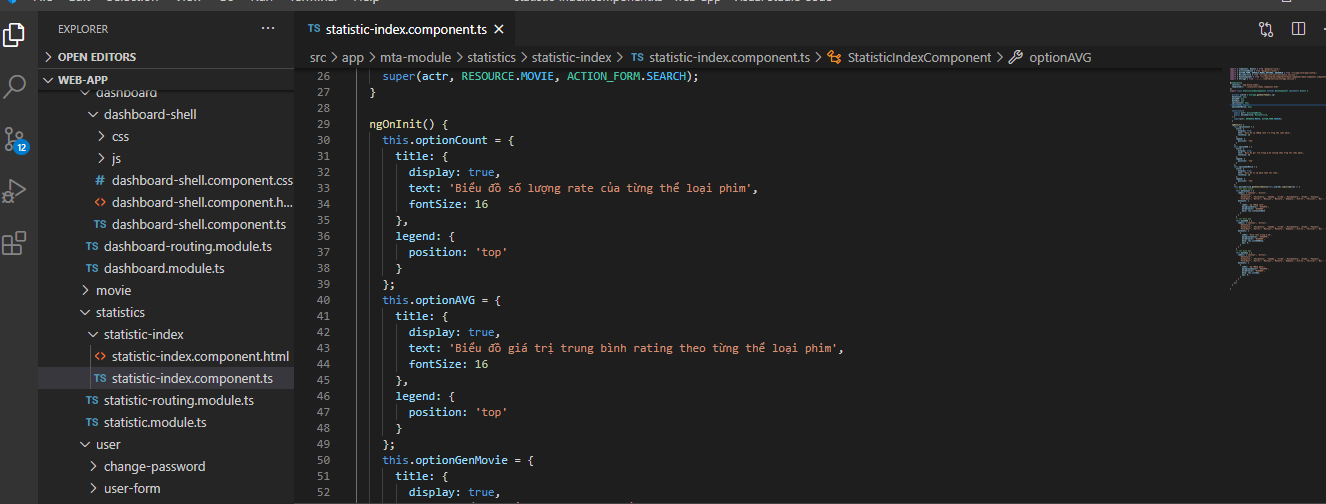
\* Xây dựng API:

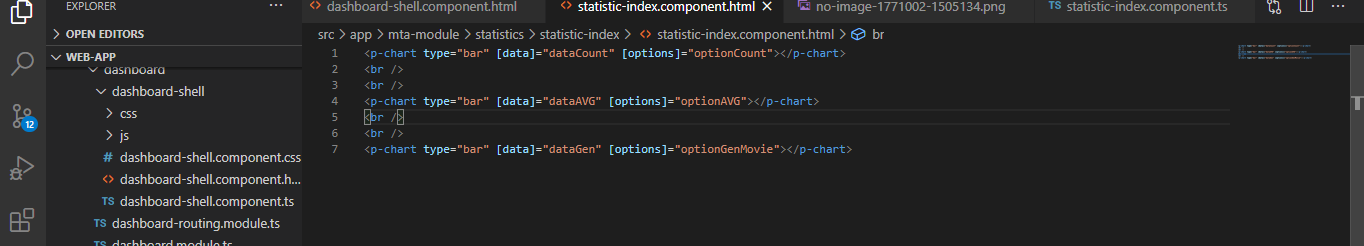






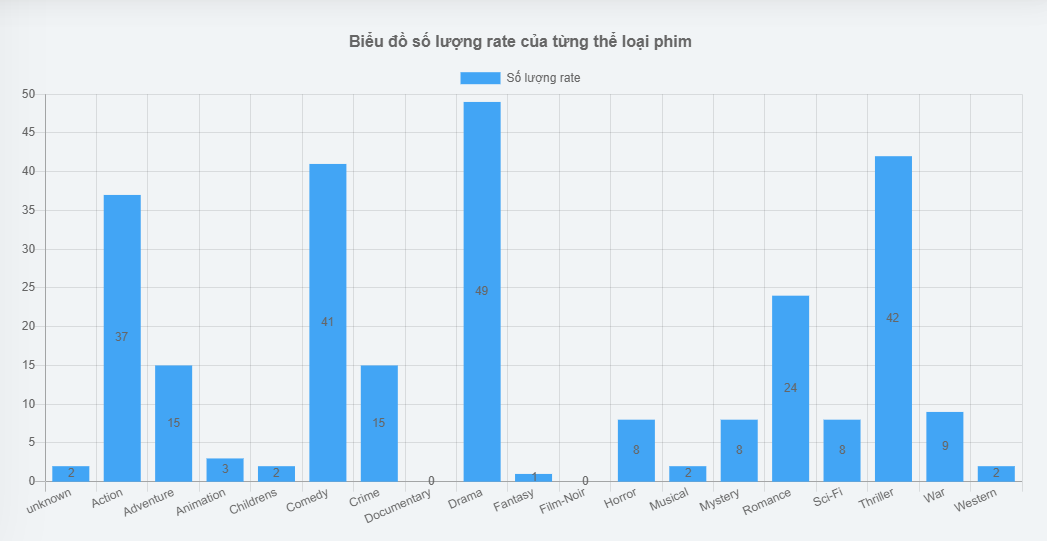
\* Xây dựng Front-end (Sử dụng module ng-char):



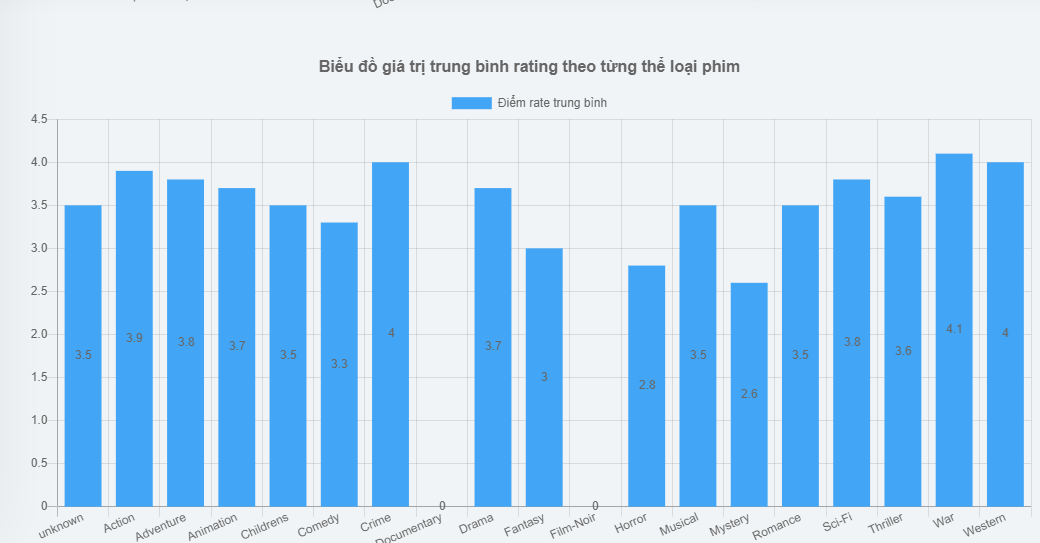


\* Kết quả:

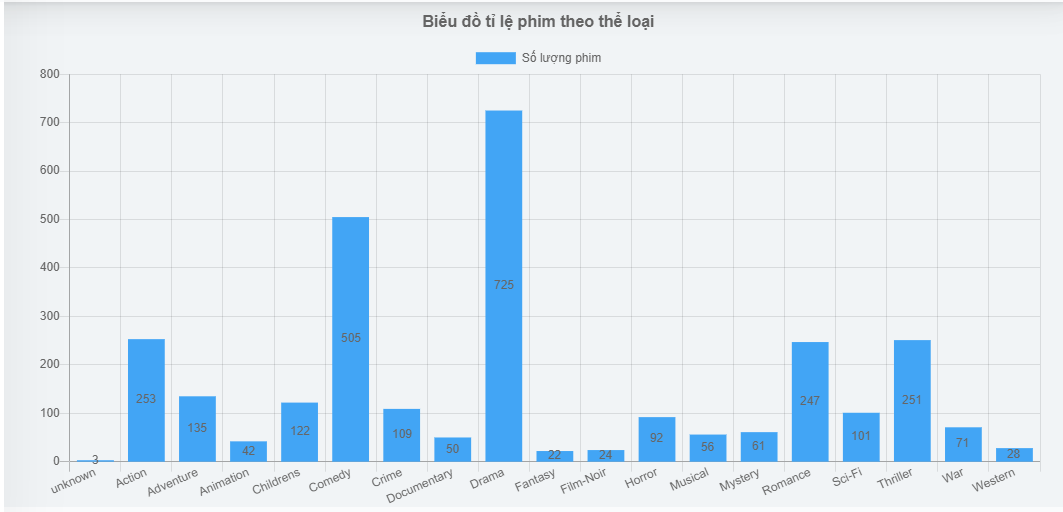
- Biểu đồ lượng rate của user login theo từng thể loại phim



- Biểu đồ giá trị trung bình rating của user login theo từng thể loại phim



- Biểu đồ tỉ lệ phim theo thể loại



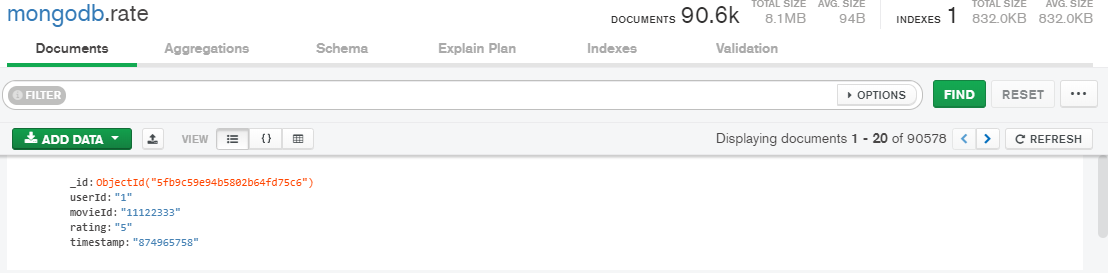
# **Đề xuất giải phải làm sạch dữ liệu**

* Dề xuất phương pháp làm sạch lần lượt dựa theo 5 tiêu chí:

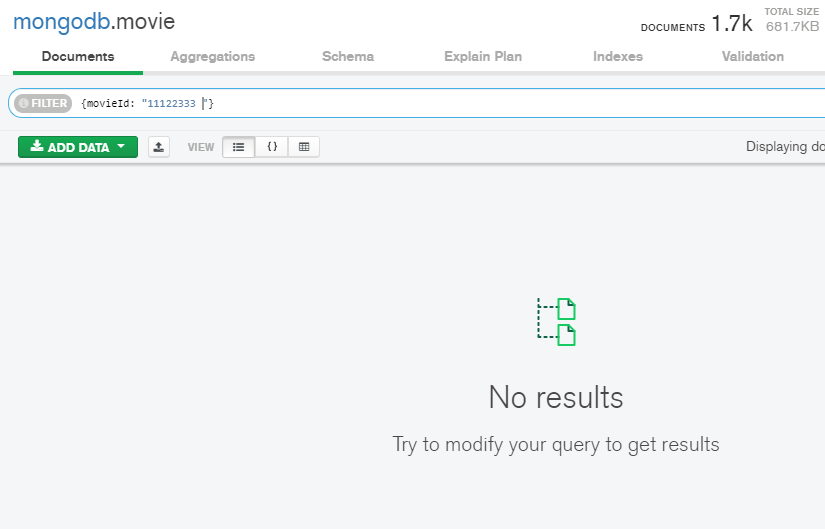
1. Data consistency: Tính thống nhất của dữ liệu

* Xử lý giá trị không nhất quán: Do dataset bao gồm 2 bảng là thông tin phim (movie) và thông tin rating (rate) của các phim nên dữ liệu thu thập có thể xảy ra trường hợp có dữ liệu ở bảng rate nhưng không có dữ liệu ở bảng movie. Ví dụ:

+ Rating cho movie id “11122333” ở bảng rate:



+ Nhưng lại không có movie này ở bảng movie:



🡺 Khi thực hiện training gặp bộ dữ liệu này sẽ bị lỗi.

🡺 Thực hiện xóa những bộ dữ liệu không nhất quán ở bảng rate bằng câu lệnh mongodb:

db.rate.distinct("movieId").forEach(function(event) {

var movie = db.movie.findOne({movieId:event});

if(movie == null) {

db.rate.remove({movieId: event},{justOne: false});

};

})

1. Completenese information: Sự hoàn thiện của dữ liệu

* Xử lý các giá trị thiếu:

+ Xử lý giá trị thiếu cho bảng movie:

* Bảng movie dùng để hình thành nên vector feature từ thể loại phim 🡺 Không thể thiếu thông tin ở các trường: id phim, các trường biểu thị thể loại phim. Cụ thể:

+ Đối với các bản ghi thiếu trường id phim 🡺 Thực hiện loại bỏ bằng câu lệnh mongodb:

db.movie.remove(

{movieId: null},

{

justOne: false,

}

)

+ Dối với bản ghi có trưởng dữ liệu biểu thị thể loại bị rỗng (null) 🡺 Thực hiện điền giá trị vào trường trống đó giá trị mặc định là 0 (Tức là phim không có thể loại này). Thực hiện việc fill dữ liệu bằng câu lệnh mongodb:

db.movie.update(

{Comedy: null},

{$set:{Comedy: 0}},

{

multi: true,

}

)

+ Xử lý giá trị thiếu cho bảng rate:

* Bảng rate chứa thông tin để xác định rating của từng user cho từng movie 🡺 Dữ liệu không thể thiếu là: id phim, id user và giá trị rate.

🡺 Tiến hành xóa những bản ghi thiếu một trong ba thông tin này bằng câu query:

db.rate.remove(

{ &or: [{movieId: null}, {rating: null}, {userId: null}]},{justOne: false}

)

1. Data accurercy : Độ chính xác của dữ liệu:

+ Kiểm tra dữ liệu trường rating trong bảng rate phải thỏa mãn hai điều kiện:

* Là kiểu số.

Nếu dữ liệu không là kiểu số 🡺 Loại bỏ bản ghi đó. Việc loại bỏ đó thực hiện bằng câu lệnh:

db.rate.remove(

{rating:{$not:{$type: 16}}},{justOne: false}

)

* Trong khoảng (1 -> 5).

Nếu dữ liệu ngoài khoảng 1 -> 5 🡺 Loại bỏ bản ghi đó. Thực hiện bằng câu lệnh:

db.rate.remove(

{$or :[{rating:{$gt: 5}}, {rating:{$lt: 1}}]},{justOne: false}

)