

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP HCM
KHOA CÔNG NGHỆ THÔNG TIN**



**MÔN: TƯƠNG TÁC DỮ LIỆU
TRỰC QUAN
BÁO CÁO ĐỒ ÁN**

ĐỀ TÀI:

**TÌM HIỂU VỀ SUPERSET
VÀ ỨNG DỤNG LÀM ĐỀ TÀI CUỐI KỲ**

GVHD: HUỲNH XUÂN PHỤNG

SVTH:

LƯU GIA BẢO 19133008

NGUYỄN DUY PHƯỚC 19133003

NGUYỄN QUỐC VIỆT 19133068

Tp.HCM, tháng 6/2022 -

HỌ VÀ TÊN SINH VIÊN THỰC HIỆN ĐỀ TÀI:

- | | | |
|---------------------|---|----------|
| 1. Lưu Gia Bảo | - | 19133008 |
| 2. Nguyễn Duy Phước | - | 19133003 |
| 3. Nguyễn Quốc Việt | - | 19133068 |

ĐIỂM:

NHẬN XÉT CỦA GV:

.....

.....

.....

.....

.....

.....

.....

.....

.....

GV KÝ TÊN

Mục lục

MỞ ĐẦU	1
1.1 Lý do chọn đề tài	1
1.2 Phạm vi đề tài	1
1.3 Mục tiêu đề tài	1
NỘI DUNG	2
Chương 1 Tìm hiểu và cài đặt Apache Superset	2
1.1 Giới thiệu superset	2
1.1.1 Khái niệm.....	2
1.1.2 Chức năng.....	2
1.2 Cài đặt Superset	3
Chương 2 Xây dựng dashboard công nghiệp điện ảnh.....	4
2.1 Dữ liệu	4
2.1.1 Giới thiệu dữ liệu	4
2.1.2 Mô tả dữ liệu.....	4
2.1.3 Mục tiêu	5
2.2 Xây dựng database	5
2.3 Cập nhập dữ liệu bằng python.....	8
2.4 Tạo và vẽ các biểu đồ trên Dashboard	9
2.4.1 Các loại biểu đồ sử dụng trong dashboard.....	9
2.4.2 Dashboard.....	12
2.5 Sử dụng jinja template trong superset	15
2.5.1 Giới thiệu JINJA TEMPLATE	15
2.5.2 Tích hợp JINJA TEMPLATE	15
2.6 Sử dụng API trong superset	17
ĐÁNH GIÁ VÀ KẾT LUẬN	26
1.1 Kết quả đạt được	26
1.2 Hạn chế.....	26
1.3 Hướng phát triển	26
TÀI LIỆU THAM KHẢO	27

MỞ ĐẦU

1.1 Lý do chọn đề tài

Giống như các phát minh lớn khác như ô tô, điện, hóa chất và máy bay, điện ảnh xuất hiện ở hầu hết các nước trên thế giới. Từ những năm 1910 trở đi, mỗi năm hàng tỷ vé xem phim được bán ra và trở nên cực kỳ phổ biến.

Có rất nhiều câu hỏi được đặt ra xoay quanh chủ đề này như Số lượng các nhà làm phim trên thế giới? Chi phí sản xuất của mỗi bộ phim qua từng năm? Doanh thu của phim, điểm đánh giá? Số lượng người tham gia đánh giá phim? Để trả lời cho những câu hỏi đó nhóm em đã chọn và phân tích bộ dữ liệu “Movie Industry” được thu thập từ trang [idmb](#) là một trang web trực tuyến lưu trữ những thông tin chi tiết nhất về các bộ phim như thông tin về nhà sản xuất, diễn viên, đạo diễn... cùng những chủ đề điện ảnh, truyền hình và video game.

1.2 Phạm vi đề tài

Phạm vi đề tài sẽ bao gồm:

- Các nước được thu thập trong bộ dữ liệu
- Thời gian: 1986-2021
- Gồm 8068 bộ phim
- Sử dụng trên công cụ Apache Superset

1.3 Mục tiêu đề tài

Mục tiêu của nhóm:

- Sử dụng thành thạo công cụ hỗ trợ Apache Superset
- Trực quan hóa dữ liệu thành các biểu đồ để trả lời cho các câu hỏi
- Xây dựng và thiết kế dashboard hoàn chỉnh cho đề tài
- Tìm hiểu thêm các tính năng nâng cao của Apache Superset

NỘI DUNG

Chương 1 Tìm hiểu và cài đặt Apache Superset

1.1 Giới thiệu superset

1.1.1 Khái niệm

Apache Superset là một ứng dụng web thông minh kinh doanh hiện đại, sẵn sàng cho doanh nghiệp. Nó có tốc độ xử lý nhanh, trực quan và có các tùy chọn giúp người dùng có thể dễ dàng khám phá và hình dung dữ liệu của họ, từ biểu đồ hình tròn đơn giản đến biểu đồ không gian địa lý.

1.1.2 Chức năng

Superset cung cấp:

- Giao diện dễ dàng sử dụng và các bảng điều khiển để tương tác
- Một loạt các hình ảnh trực quan đẹp mắt để giới thiệu dữ liệu của bạn
- Trình tạo trực quan hóa không cần mã để trích xuất và trình bày bộ dữ liệu
- Cú pháp đơn giản cho phép các nhà phân tích dữ liệu nhanh chóng xác định và tùy chỉnh
- Hỗ trợ out-of-the-box cho hầu hết các cơ sở dữ liệu nổi tiếng SQL
- Bộ nhớ đệm và truy vấn không đồng bộ liền mạch, trong bộ nhớ
- Một mô hình bảo mật có thể mở rộng cho phép cấu hình các quy tắc rất phức tạp về việc ai có thể truy cập các tính năng và bộ dữ liệu của sản phẩm.
- Tích hợp với các phụ trợ xác thực chính (cơ sở dữ liệu, OpenID, LDAP, OAuth, v.v.)
- Khả năng thêm các plugin trực quan hóa tùy chỉnh
- Cung cấp API để tương tác từ bên ngoài

- Một kiến trúc gốc đám mây được thiết kế từ đầu cho quy mô

1.2 Cài đặt Superset

Một số cách để bạn có thể bắt đầu với Superset :

- Cài đặt Docker
- Cài đặt Docker Compose
- Clone superset từ trên github về
- Chạy superset trên Docker Compose

Chương 2 Xây dựng dashboard công nghiệp điện ảnh

2.1 Dữ liệu

2.1.1 Giới thiệu dữ liệu

Ngành công nghiệp điện ảnh là một trong những nguồn giải trí lớn nhất trên thế giới. Ngành công nghiệp sản xuất hàng ngàn bộ phim hàng năm và thu về hàng tỷ đô la doanh thu.

Những câu hỏi về doanh thu, xếp hạng của bộ phim, hay công ty nào sản xuất, quốc gia nào đứng đầu về ngành công nghiệp điện ảnh. Và bây giờ, bất kỳ ai có kinh nghiệm (bạn) đều có thể hỏi các câu hỏi cụ thể về ngành công nghiệp điện ảnh và nhận được câu trả lời.

2.1.2 Mô tả dữ liệu

Có 6820 phim trong tập dữ liệu (220 phim mỗi năm, 1986-2016).

Mỗi phim có các thuộc tính sau:

1. budget: the budget of a movie.
2. company: the production company
3. country: country of origin
4. director: the director
5. genre: main genre of the movie.
6. gross: revenue of the movie
7. name: name of the movie
8. rating: rating of the movie (R, PG, etc.)
9. released: release date (YYYY-MM-DD)
- 10.runtime: duration of the movie
- 11.score: IMDb user rating

12.votes: number of user votes

13.star: main actor/actress

14.writer: writer of the movie

15.year: year of release

16.id

2.1.3 Mục tiêu

- Phân tích những ảnh hưởng và thành công của ngành phim ảnh
- + Số lượng các nhà làm phim trên thế giới
- + Chi phí sản xuất của mỗi bộ phim qua từng năm
- + Doanh thu của phim , điểm đánh giá
- + Số lượng người tham gia đánh giá phim
- Phân tích xu hướng của ngành phim ảnh
- + Những phim được đánh giá cao nhất
- + Thể loại phim phổ biến
- + Diễn viên, đạo diễn, nhà làm phim được yêu thích .

2.2 Xây dựng database

Trước khi vẽ biểu đồ thì chúng ta cần có database để lưu trữ dữ liệu .Ở đây nhóm em sử dụng mysql . Đầu tiên vào máy đã cài superset tạo file yaml để tạo container mysql và chạy file trên

```
docker-compose up
```



```
version: '3.8'

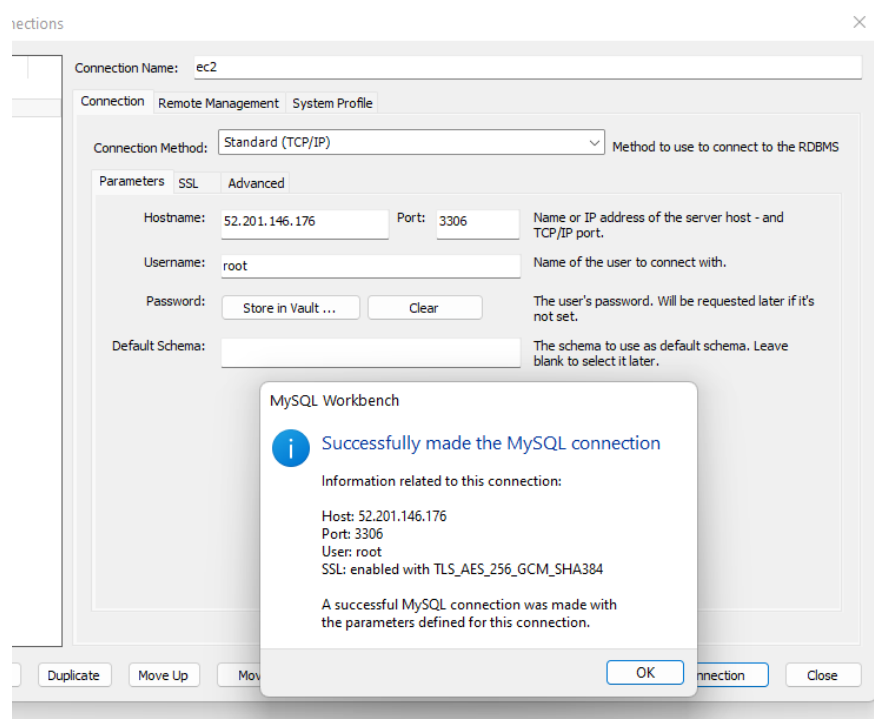
services:
  db:
    image: mysql:latest # use latest version of mysql
    container_name: db # add a name for the container
    command: --default-authentication-plugin=mysql_native_password
    restart: unless-stopped
    environment: # add default values, see docs for more info.
      MYSQL_USER: user
      MYSQL_ROOT_PASSWORD: root
      MYSQL_PASSWORD: root
      MYSQL_DATABASE: testdb # create this database on startup
    volumes:
      - my-db:/var/lib/mysql
    ports:
      - '3306:3306'

volumes: # add persistent data even if container is removed.
  my-db:
```

Kiểm tra các container đang chạy

```
:~ $ docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED
STATUS        PORTS                    COMMAND                  NAME
6477e87dc433   mysql:latest                       "docker-entrypoint.s..." 3 days
ago          Up 7 minutes            0.0.0.0:3306->3306/tcp, 33060/tcp   db
b73b8f5e074b   apache/superset:latest-dev         "/app/docker/docker-..." 11 day
s ago       Up 7 minutes (healthy)  0.0.0.0:8088->8088/tcp             sup
erset_app
553ea3207f16   apache/superset:latest-dev         "/app/docker/docker-..." 11 day
s ago       Up 7 minutes (unhealthy) 8088/tcp                         sup
erset_worker
3af044803f9f   apache/superset:latest-dev         "/app/docker/docker-..." 11 day
s ago       Up 6 minutes (unhealthy) 8088/tcp                         sup
erset_worker_beat
df903f87ad9d   redis:latest                       "docker-entrypoint.s..." 11 day
s ago       Up 7 minutes            6379/tcp                         sup
erset_cache
3453a32a82a9   postgres:10                       "docker-entrypoint.s..." 11 day
s ago       Up 7 minutes            5432/tcp                         sup
erset_db
```

Sau đó vào mysql workbench kết nối với mysql vừa tạo



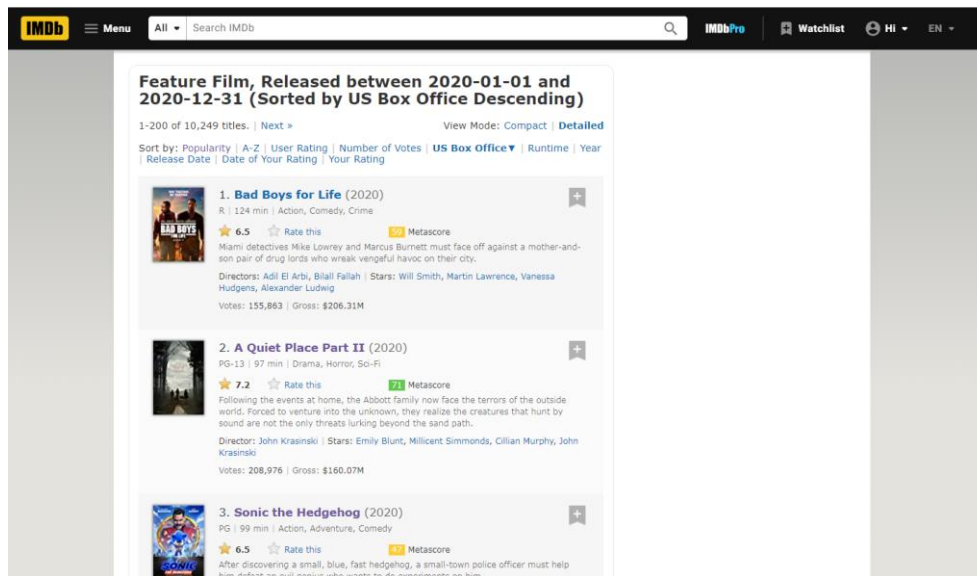
Tạo cơ sở dữ liệu và thêm dữ liệu theo yêu cầu

```
CREATE TABLE movie(  
  id INT NOT NULL auto_increment primary key  
  ,name    VARCHAR(83) NOT NULL  
  ,rating  VARCHAR(9)  
  ,genre   VARCHAR(9) NOT NULL  
  ,year    INTEGER NOT NULL  
  ,released DATE  
  ,score   NUMERIC(3,1)  
  ,votes   NUMERIC(9,1)  
  ,director VARCHAR(32) NOT NULL  
  ,writer  VARCHAR(32)  
  ,star    VARCHAR(27)  
  ,country VARCHAR(30)  
  ,budget  NUMERIC(11,1)  
  ,gross   NUMERIC(12,1)  
  ,company VARCHAR(63)  
  ,runtime NUMERIC(5,1)  
);  
  
INSERT INTO movie(name,rating,genre,year,released,score,votes,director,writer,star,country)  
INSERT INTO movie(name,rating,genre,year,released,score,votes,director,writer,star,country)  
INSERT INTO movie(name,rating,genre,year,released,score,votes,director,writer,star,country)  
INSERT INTO movie(name,rating,genre,year,released,score,votes,director,writer,star,country)
```

2.3 Cập nhập dữ liệu bằng python

Nhóm em sử dụng python để cào dữ liệu từ trang

https://www.imdb.com/search/title/?count=200&view=simple%27%27&title_type=feature&release_date=2020&sort=boxoffice_gross_us,desc



Đây là phần code sử dụng thư viện BeautifulSoup

```
#####
return details

def write_csv(data):
    '''Write list of dicts to csv.'''
    df = pd.DataFrame(data)
    df.to_csv('movies.csv', index=False)

def main():
    all_movie_data = []

    for year in range(2020, 2022):
        movies = get_movies(year)
        for movie_url in movies:
            movie_data = {}
            movie_html = go_to_movie(movie_url)
            soup = BeautifulSoup(movie_html, 'html.parser')
            movie_data.update(scrap_titlebar(soup, year))
            movie_data.update(scrap_crew(soup))
            movie_data.update(scrap_details(soup))
            all_movie_data.append(movie_data)
            time.sleep(1)
        print(year, 'done.')
    write_csv(all_movie_data)

''' '''
```

Kết quả sau khi chạy sẽ được lưu ở file csv từ đó có thể cập nhập được dữ liệu mới

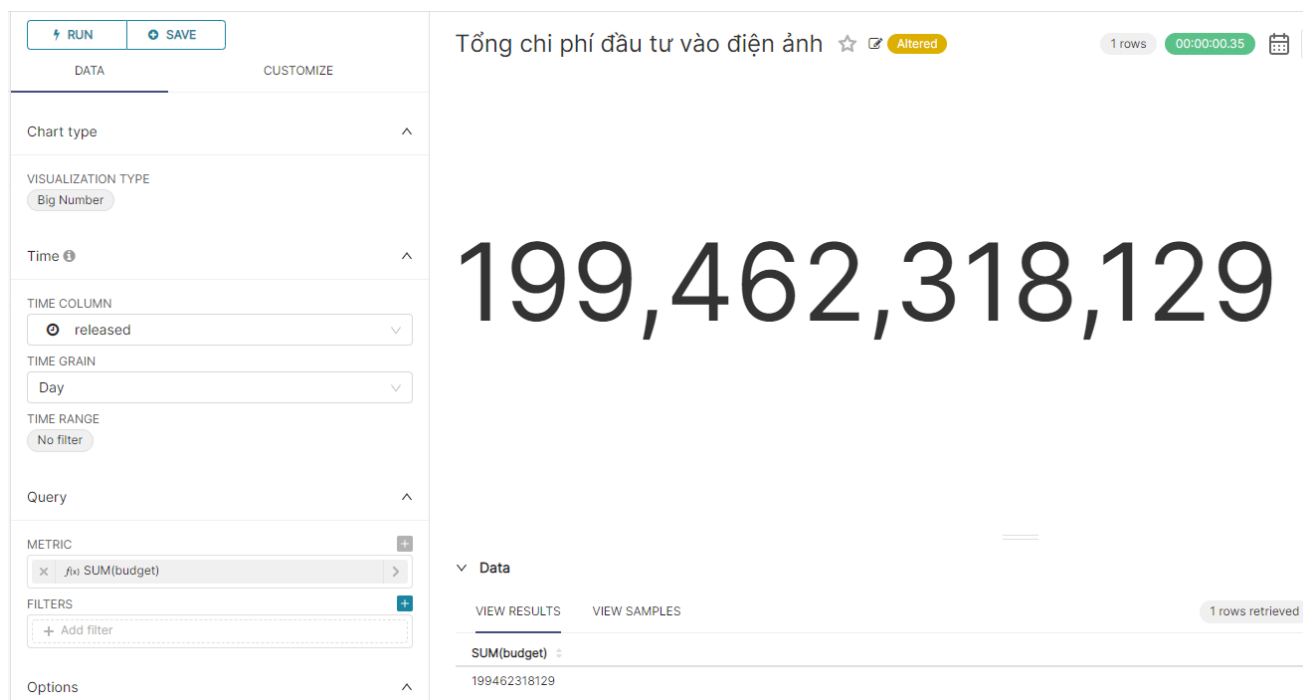
```
if __name__ == '__main__':
    main()

{'name': 'Bad Boys for Life', 'rating': 'R', 'genre': None, 'year': 2020, 'released': 'January 17, 2020 (United States)', 'score': 6.5, 'votes': 155000.0}
{'director': 'Adil El Arbi', 'writer': 'Peter Craig', 'star': 'Will Smith'}
{'country': 'United States', 'budget': 90000000.0, 'gross': 426505244.0, 'company': 'Columbia Pictures', 'runtime': 124}
{'name': 'A Quiet Place Part II', 'rating': 'PG-13', 'genre': None, 'year': 2020, 'released': 'May 28, 2021 (United States)', 'score': 7.2, 'votes': 207000.0}
{'director': 'John Krasinski', 'writer': 'John Krasinski', 'star': 'Emily Blunt'}
{'country': 'United States', 'budget': None, 'gross': 297372261.0, 'company': 'Buffalo FilmWorks', 'runtime': 97}
{'name': 'Sonic the Hedgehog', 'rating': 'PG', 'genre': None, 'year': 2020, 'released': 'February 14, 2020 (United States)', 'score': 6.5, 'votes': 125000.0}
{'director': 'Jeff Fowler', 'writer': 'Pat Casey', 'star': 'Ben Schwartz'}
{'country': 'United States', 'budget': 85000000.0, 'gross': 319715683.0, 'company': 'Paramount Pictures', 'runtime': 99}
{'name': 'Birds of Prey', 'rating': 'R', 'genre': None, 'year': 2020, 'released': 'February 7, 2020 (United States)', 'score': 6.0, 'votes': 228000.0}
```

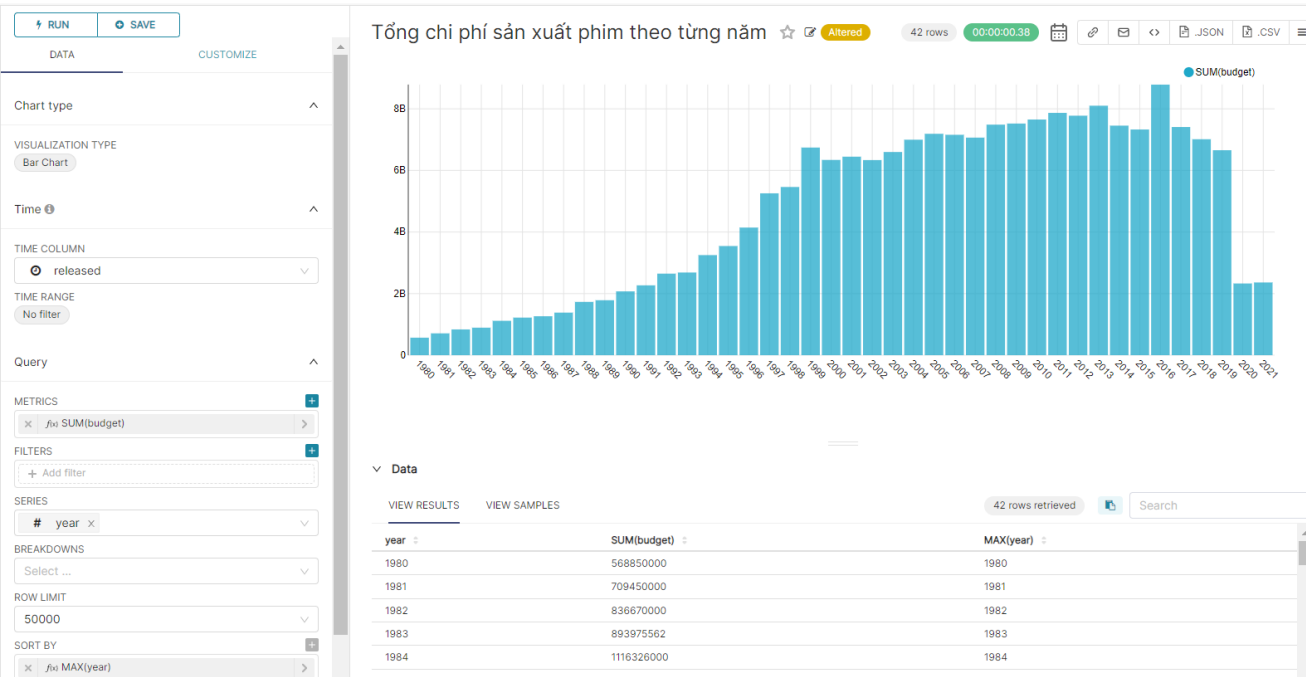
2.4 Tạo và vẽ các biểu đồ trên Dashboard

2.4.1 Các loại biểu đồ sử dụng trong dashboard

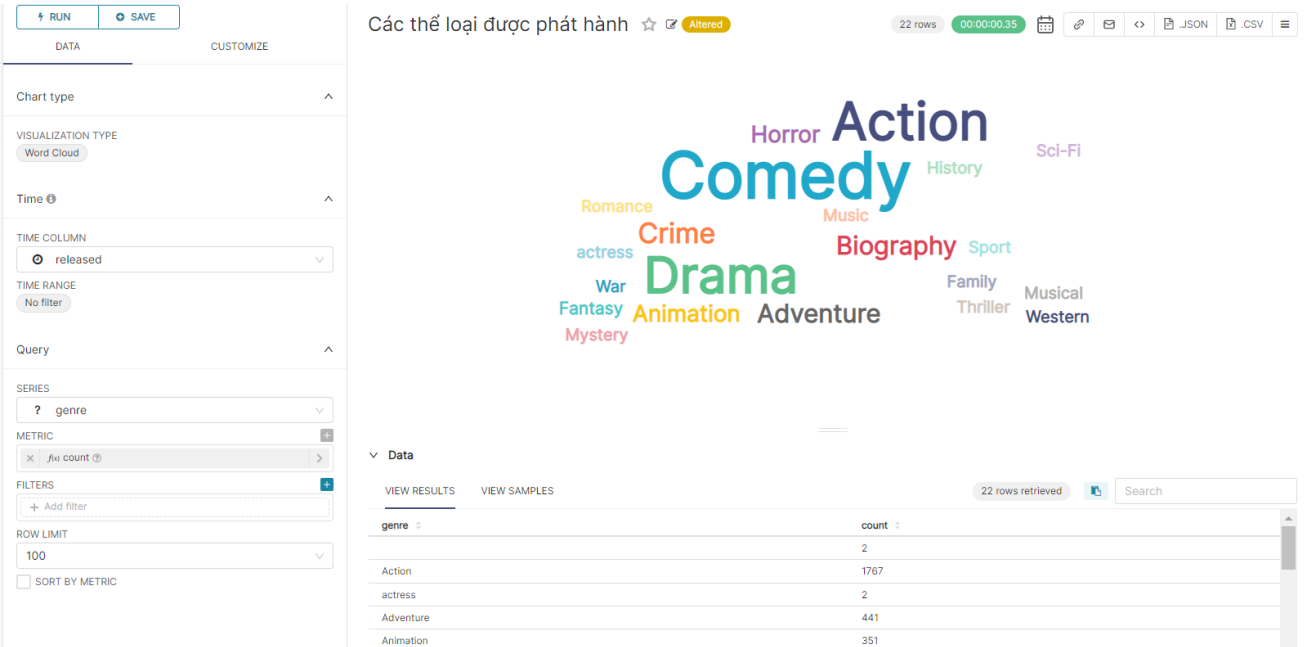
a. Biểu đồ Big number



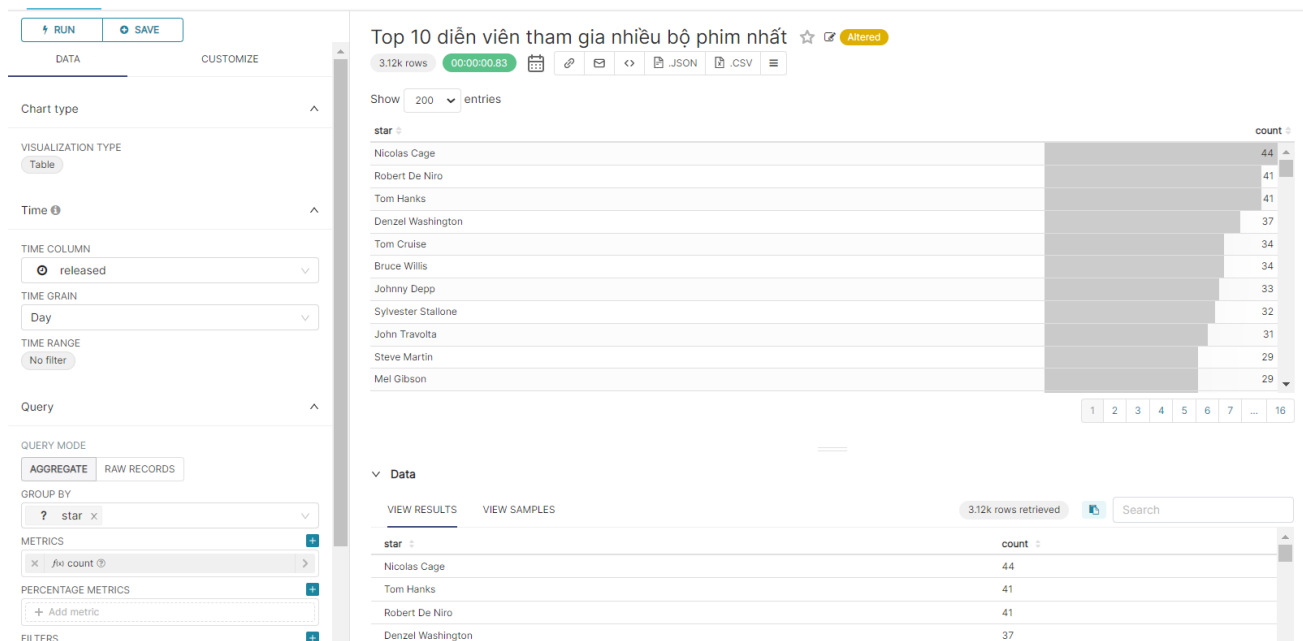
d. Biểu đồ Bar Chart



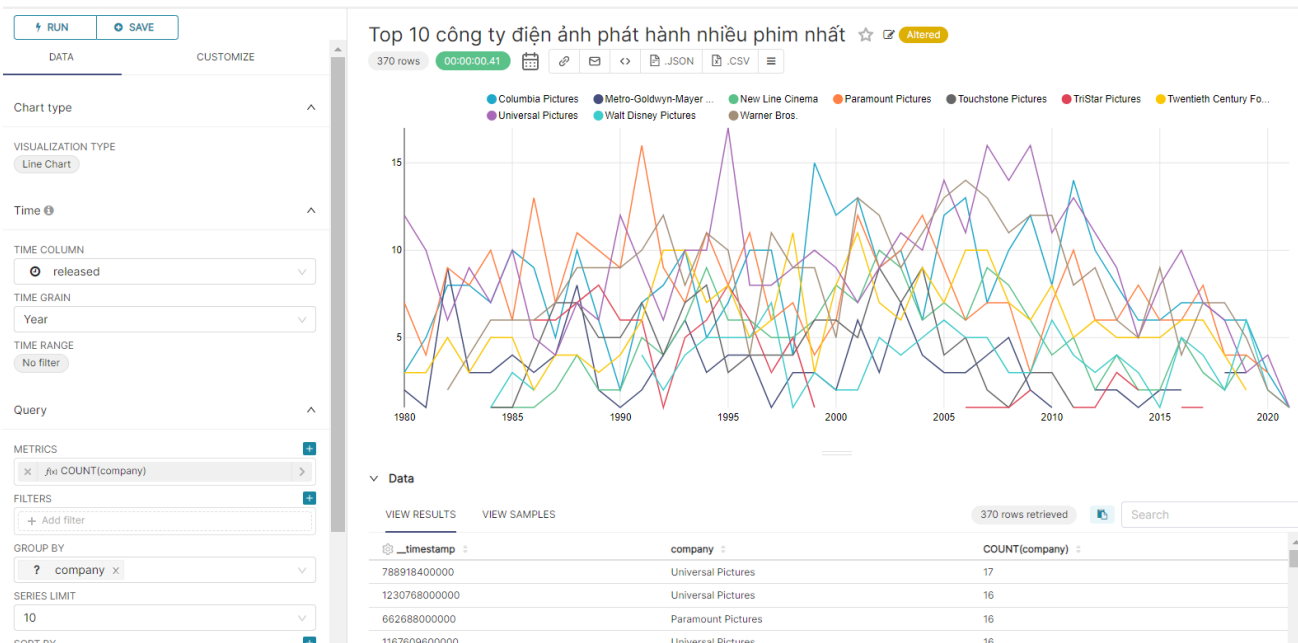
e. Biểu đồ Word Cloud



f. Biểu đồ Table



g. Biểu đồ Line Chart

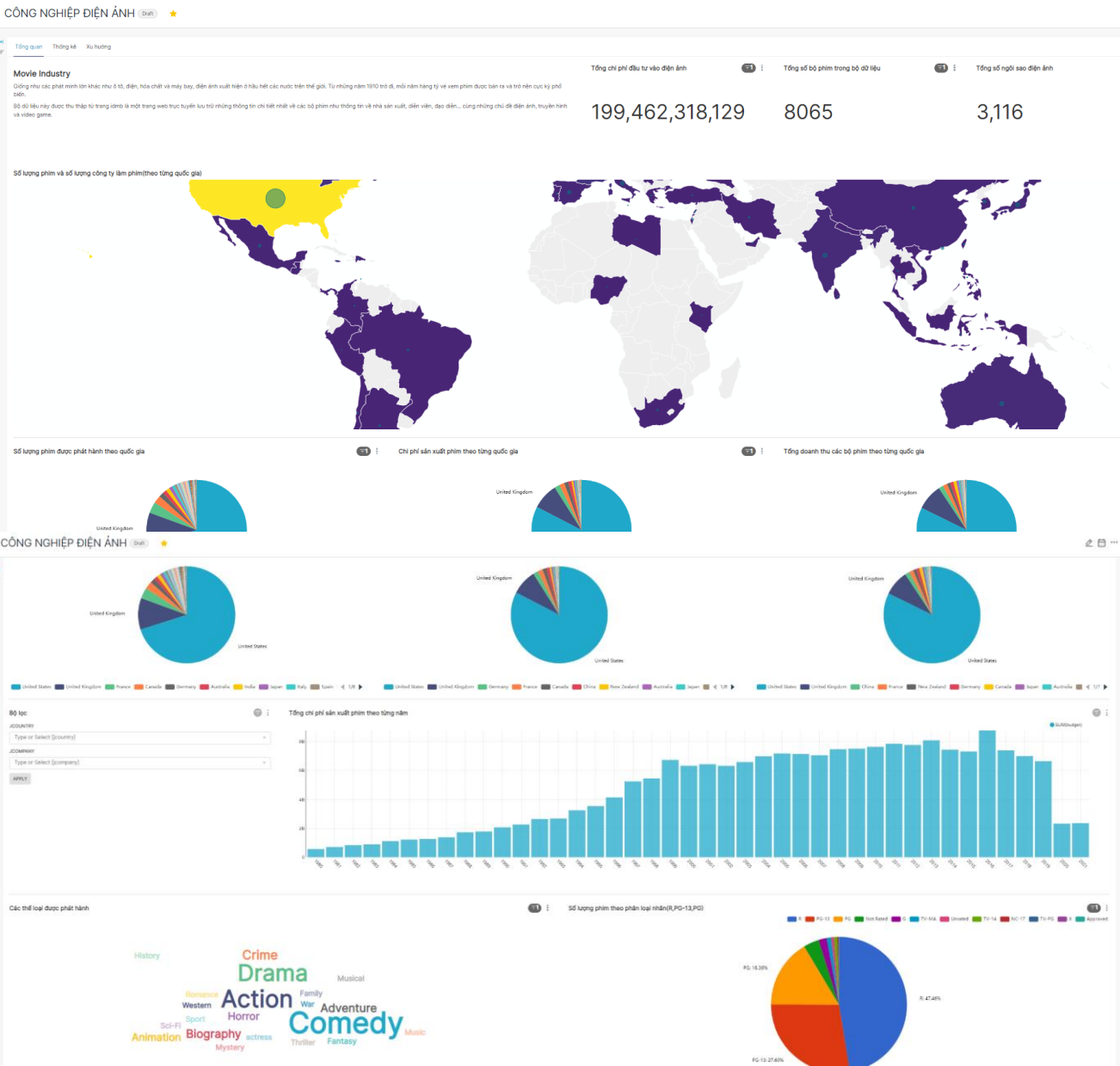


2.4.2 Dashboard

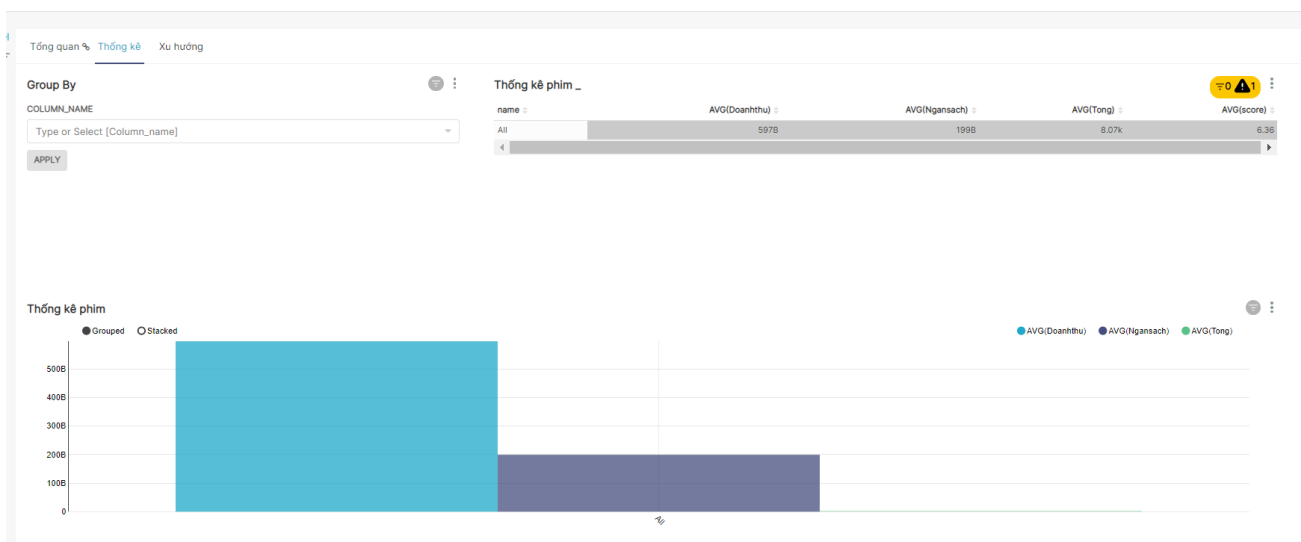
Địa chỉ web: <http://54.237.122.89:8088/>

Gồm 3 phần:

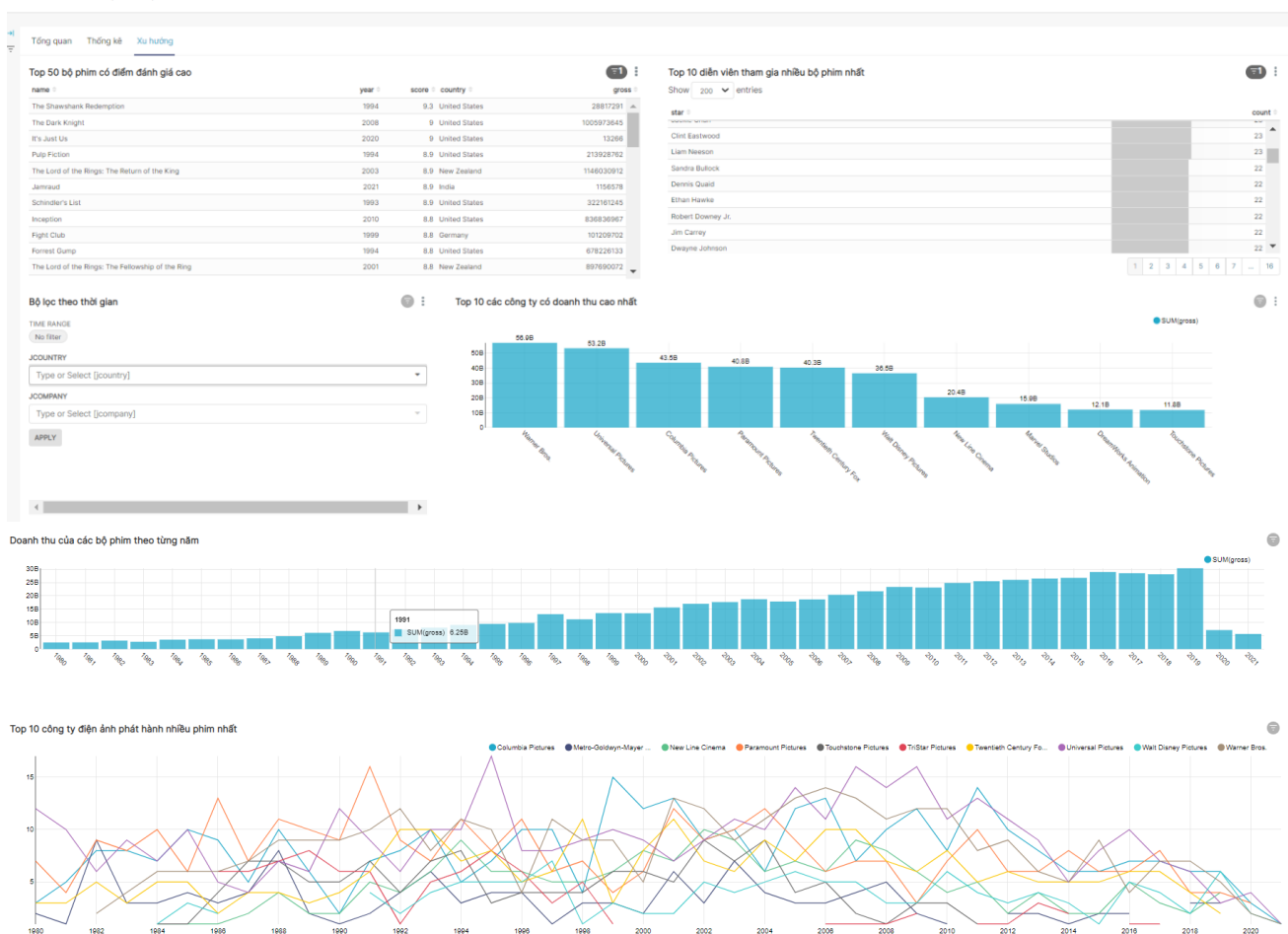
i. Tổng quan



ii. Thống kê

CÔNG NGHIỆP ĐIỆN ẢNH Draft ★

iii. Xu hướng

CÔNG NGHIỆP ĐIỆN ẢNH Draft ★

2.5 Sử dụng jinja template trong superset

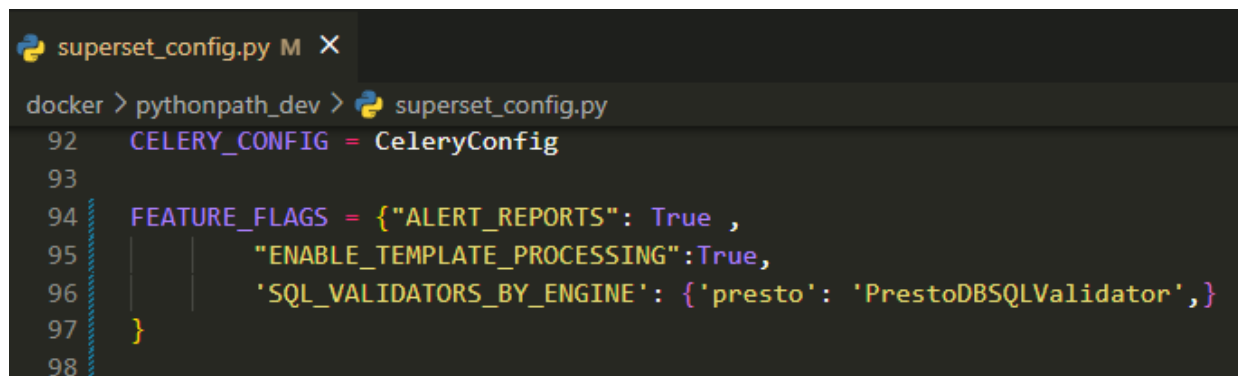
2.5.1 Giới thiệu JINJA TEMPLATE

Jinja2 là một ngôn ngữ tạo template giàu tính năng được sử dụng rộng rãi trong hệ sinh thái Python. Nó có thể được sử dụng trực tiếp trong các chương trình Python của bạn và rất nhiều ứng dụng lớn hơn sử dụng nó làm công cụ kết xuất mẫu của họ.

Ngôn ngữ tạo template cho phép tạo các tài liệu dựa trên văn bản trong đó một số nội dung có thể được tạo động. Các tệp kết quả có thể là HTML, JSON, XML hoặc bất kỳ thứ gì sử dụng văn bản thuần túy làm mã hóa..

2.5.2 Tích hợp JINJA TEMPLATE

Nhóm em sử dụng jinja template để làm filter box để mang lại kết quả tốt hơn khi sử dụng filter của superset. Đầu tiên để kích hoạt được jinja ta vào file superset và cấu hình như sau



```
superset_config.py M X
docker > pythonpath_dev > superset_config.py
92 CELERY_CONFIG = CeleryConfig
93
94 FEATURE_FLAGS = {"ALERT_REPORTS": True,
95                 "ENABLE_TEMPLATE_PROCESSING": True,
96                 'SQL_VALIDATORS_BY_ENGINE': {'presto': 'PrestoDBSQLValidator'},
97             }
98
```

Vào edit dataset và thêm dòng lệnh dưới đây để có thể lọc theo thời gian ,country và company

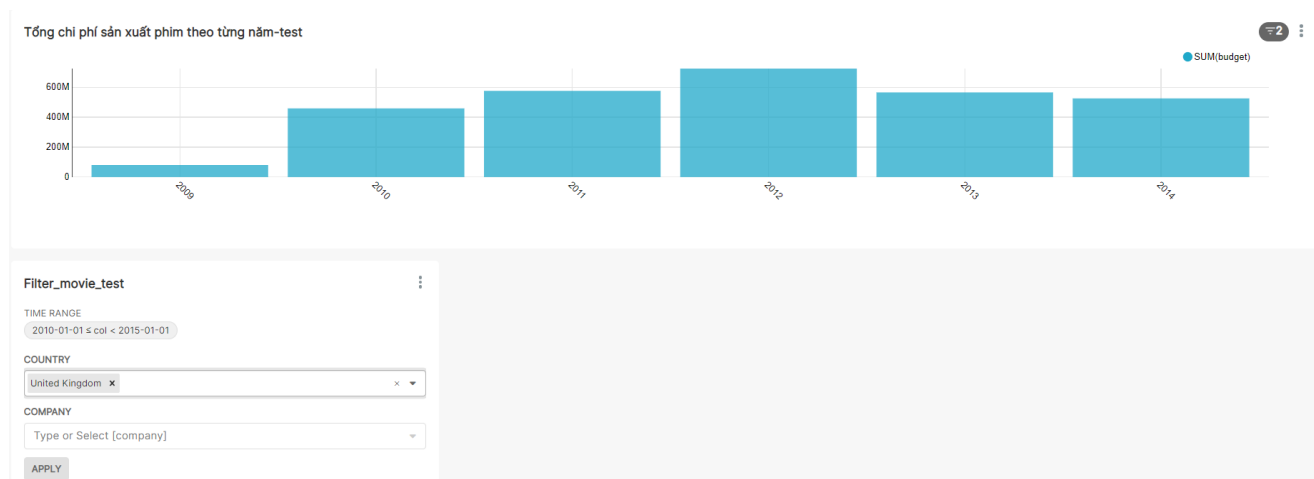
SQL ⓘ

```

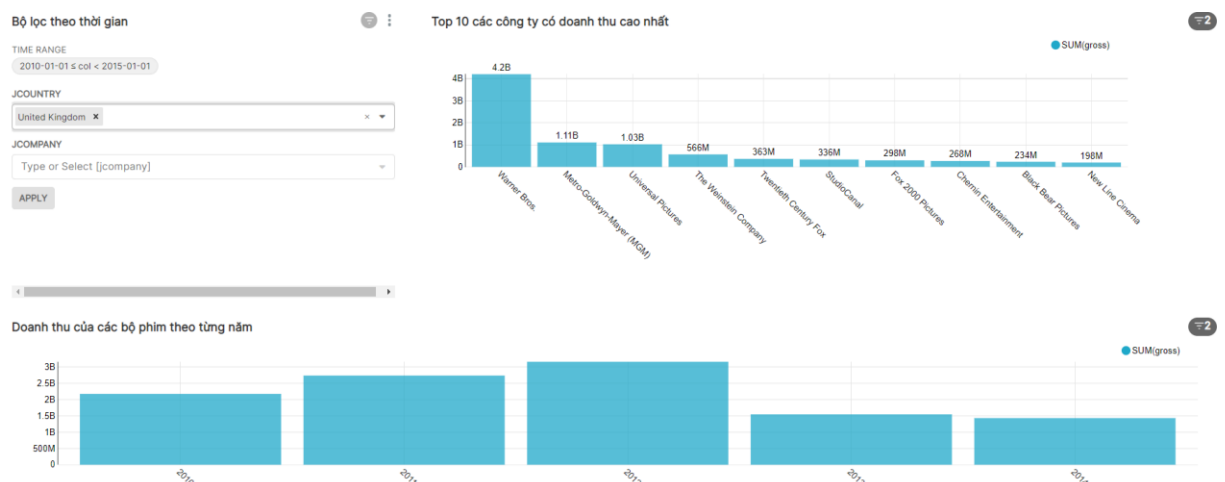
1  -- Note: Unless you save your query, these tabs will NOT persist if you clear your cookies or change browsers.
2
3  SELECT * FROM DoAn4.movie
4
5
6  {% if filter_values('jcountry')|length ==0 and filter_values('jcompany')|length ==0 %}
7
8      where country LIKE '%'
9  {% else %}
10     where (country in ({{filter_values('jcountry')|join('','')+ "'"}}))
11     or company in ({{filter_values('jcompany')|join('','')+ "'"}}))
12  {% endif %}
13
14
15  {% if from_dttm != None %}
16  and ( year >= year('{{from_dttm}}') and year < year('{{to_dttm}}'))
17  {% endif %}

```

Khi không áp dụng jinja bộ lọc sẽ cho ra kết không như yêu cầu



Khi áp dụng cho ra kết quả đúng yêu cầu



Tiếp theo áp dụng jinja để là filter groupby theo các cột yêu cầu. Vào trong sqllab để tạo ra dataset là các cột trong bộ dữ liệu

```

1 -- Note: Unless you save your query, these tabs will NOT persist if you clear your cookies or change browsers.
2 select column_name
3 from information_schema.columns
4 where Table_name like 'movie' and Column_name in ('country','company','director','writer','start','year')
5

```

RUN LIMIT: 1000 00:00:01.13

RESULTS QUERY HISTORY

EXPLORE DOWNLOAD TO CSV COPY TO CLIPBOARD Filter results

5 rows returned

Column_name
year
director
writer
country
company

Vào edit dataset trong chart

```

1 -- Note: Unless you save your query, these tabs will NOT persist if you clear your cookies or change browsers.
2 {% if filter_values('Column_name')|length !=0 %}
3 SELECT count(*) as Tong ,sum(budget) as Ngansach,sum(gross) as Doanhthu,avg(score) as score , {{""+"","".join(filter_va
4 from movie
5 {% if from_dttm != None %}
6 Where year >= year('{{from_dttm}}') and year < year('{{ to_dttm }}')
7 {% endif %}
8 GROUP BY {{""+"","".join(filter_values('Column_name'))+ ""}}
9 {% else %}
10 SELECT count(*) as Tong ,sum(budget) as Ngansach,sum(gross) as Doanhthu,avg(score) as score , 'All' as name
11 from movie
12 {% if from_dttm != None %}
13 Where year >= year('{{from_dttm}}') and year < year('{{ to_dttm }}')
14 {% endif %}
15 {% endif %}
16

```

Kết quả

Group By	Thống kê phim _
COLUMN_NAME	name
country	AVG(Doanhthu)
APPLY	AVG(Ngansach)
	AVG(Tong)
	AVG(score)
	United States 104B 33B 713 6.4
	United Kingdom 11.9B 3.03B 116 6.76
	New Zealand 3B 699M 6 7.42
	France 2.63B 577M 39 6.58
	Canada 630M 380M 21 6.44
	Australia 590M 300M 14 6.69
	Germany 573M 150M 7 6.34
	Spain 417M 72M 7 7.14

2.6 Sử dụng API trong superset

Trong superset có cung cấp các api để người dùng thực hiện các yêu cầu của mình như thêm xóa sửa các dashboard,chart,..

Superset Documentation Community

Introduction
Installation and Configuration
Connecting to Databases
Creating Charts and Dashboards
Miscellaneous
Contributing
Frequently Asked Questions
API
Security

API

Superset's public **REST API** follows the [OpenAPI specification](#), and is documented here. The docs bellow are generated using [Swagger React UI](#).

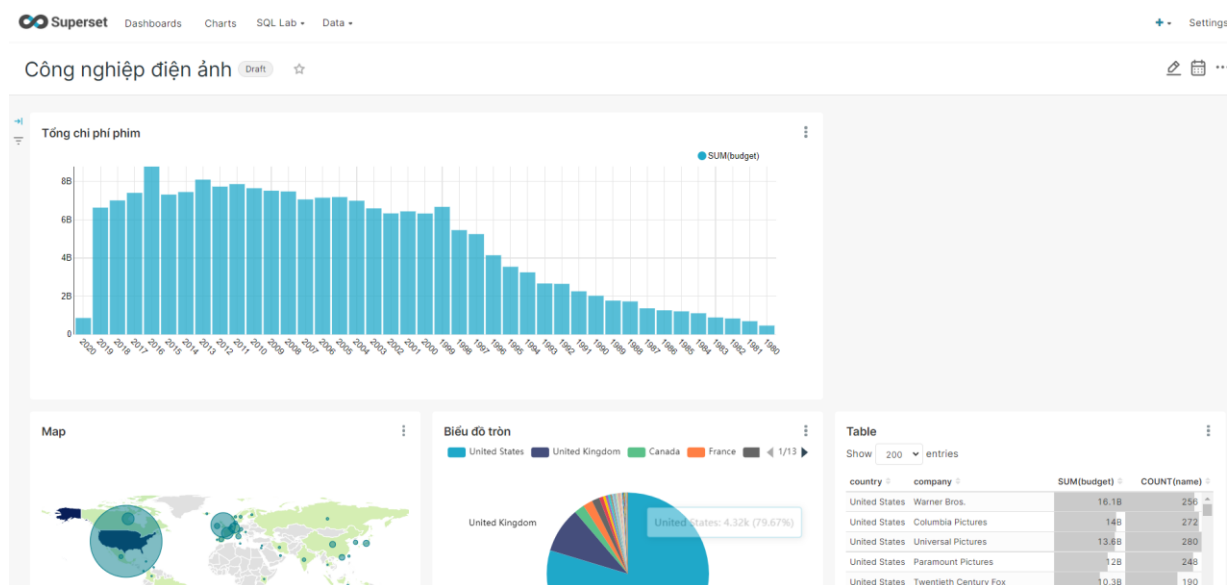
NOTE! You can find an interactive version of this documentation on your local Superset instance at `/swagger/v1` (unless disabled)

Servers
http://localhost:8088

Annotation Layers

- DELETE** /api/v1/annotation_layer/
- GET** /api/v1/annotation_layer/
- POST** /api/v1/annotation_layer/
- GET** /api/v1/annotation_layer/_info

Ở đề tài này nhóm em sẽ sử dụng API để đăng nhập vào superset và đọc , thêm , xóa , sửa các chart trong dashboard .



Ở đây trong dashboard có các chart cơ bản như bar , map ,pie , table sử dụng api để xem thông tin chi tiết các chart . Các thư viện hỗ trợ

```
import pandas as pd
import json
import requests
from IPython.display import JSON
```

✓ 1.8s

Để xem thông tin chart thì đầu tiên ta phải đăng nhập để được cấp quyền . Đăng nhập bằng tài khoản admin

```
payload = {
    'username': 'admin',
    'password': 'admin',
    'provider': 'db'
}

r = requests.post(base_url + '/api/v1/security/login', json=payload)
access_token = r.json()
headersAuth = {
    'Authorization': 'Bearer ' + access_token['access_token']
}
```

✓ 0.7s

Xem thông tin các dashboard có trong superset qua phương thức GET

```
r3 = requests.get(base_url + '/api/v1/dashboard/', headers = headersAuth)
resp_dashboard = r3.json()
for result in resp_dashboard['result']:
    print(result['dashboard_title'], result['id'])
```

```
Công nghiệp điện ảnh 12
Unicode Test 10
Slack Dashboard 9
FCC New Coder Survey 2018 8
Sales Dashboard 7
Video Game Sales 6
COVID Vaccine Dashboard 5
deck.gl Demo 4
Misc Charts 3
USA Births Names 2
World Bank's Data 1
[ untitled dashboard ] 13
```

Ở đây dashboard dashboard chúng ta cần có id = 12 , truy cập vào dashboard trên để xem thông tin các chart có trong dashboard thông qua phương thức GET

```
r2 = requests.get(base_url + '/api/v1/dashboard/12/charts', headers = headersAuth)
resp_chart = r2.json()
for result in resp_chart['result']:
    print(result['slice_name'], result['form_data']['slice_id'])
# resp_chart['result']
```

✓ 0.5s

Tổng chỉ phí phim 841
Map 846
Biểu đồ tròn 847
Table 848

Xem thông tin chi tiết chart

```
r3 = requests.get(base_url + '/api/v1/chart/841', headers = headersAuth)
resp_chart = r3.json()
# with open('pie.json', 'w') as f:
#     json.dump(resp_chart, f)
resp_chart
# for result in resp_chart['result']:
#     print(result['slice_name'], result['id'])
```

✓ 0.5s

```
{'description_columns': {},
 'id': 841,
 'label_columns': {'cache_timeout': 'Cache Timeout',
 'certification_details': 'Certification Details',
 'certified_by': 'Certified By',
 'dashboards.dashboard_title': 'Dashboards Dashboard Title',
 'dashboards.id': 'Dashboards Id',
 'dashboards.json_metadata': 'Dashboards Json Metadata',
 'description': 'Description',
 'is_managed_externally': 'Is Managed Externally',
 'owners.first_name': 'Owners First Name',
 'owners.id': 'Owners Id',
 'owners.last_name': 'Owners Last Name',
 'owners.username': 'Owners Username',
 'params': 'Params',
 'query_context': 'Query Context',
 'slice_name': 'Slice Name',
 'viz_type': 'Viz Type'},
 'result': {'cache_timeout': None,
 'certification_details': None,
 'certified_by': None,
 'dashboards': [{'dashboard_title': 'Công nghiệp điện ảnh',
 'id': 12,
 'json_metadata': '{"show_native_filters": true, "color_scheme": "", "refresh_frequency": 0, "timed_refresh_immune_slices": [], "default_filters": "{}", "chart_configuration": {}}'}],
 'timed_refresh_immune_slices': [], 'default_filters': "{}", 'chart_configuration': {}}
```

Nhóm em đã lưu ra một file json để tiện theo dõi. Mỗi cái có các thông tin cần lưu ý như id, id dashboard (chart đó thuộc dashboard nào), owner (người sở hữu), param (tham số của chart).

```
    },
    "result": {
      "cache_timeout": null,
      "certification_details": null,
      "certified_by": null,
      "dashboards": [
        {
          "dashboard_title": "C\u00f4ng nghi\u1ec7p \u0111i\u1ec7n \u1ea3nh",
          "id": 12,
          "json_metadata": "{\n  \"show_native_filters\": true, \"color_scheme\":\n",
        }
      ],
      "description": null,
      "is_managed_externally": false,
      "owners": [
        {
          "first_name": "Superset",
          "id": 1,
          "last_name": "Admin",
          "username": "admin"
        }
      ],
      "params": "{\n  \"adhoc_filters\": [],\n  \"bottom_margin\": \"auto\",\n  \"\n",
      "query_context": "{\n  \"datasource\": {\n    \"id\": 24, \"type\": \"table\", \"force\": f",
      "slice_name": "T\u1ed5ng chi ph\u00ed",
      "viz_type": "dist_bar"
    },
    "viz_type": "dist_bar"
  },
  "viz_type": "dist_bar"
}
```

Nhóm em sẽ sử dụng các thông tin trên để tạo ra một bar chart với tên bar_chart_test, với metric sum(budget) (tổng chi phí phim qua các năm).


```
with open('bar/data1.json') as file:
    data1 = json.load(file)
with open('bar/params.json') as file:
    param = json.load(file)
with open('bar/metric.json') as file:
    metric = json.load(file)
with open('bar/metric.json') as file:
    metric2 = json.load(file)
print(type(data1))
print(type(param))
print(type(metric))

# data
data1['dashboards'][0]=12
# data1['dashboards'].append(200)
data1['datasource_id']=24
data1['datasource_type']="table"
data1['slice_name']="bar_chart_test"
data1['params']=" "
# data1['datasource_name']="abc"
# metric
metric['column']['id']=774
metric['column']['column_name']="budget"
metric['aggregate']="SUM"
metric['label']="SUM(budget)"
metric['expressionType']='SIMPLE'

metric2['column']['id']=766
metric2['column']['column_name']="year"
metric2['aggregate']="MAX"
metric2['label']="MAX(year)"
metric2['expressionType']='SIMPLE'
```

```
# Bar chart

# Params
param['groupby'][0]="year"
# param['groupby'].append("aaa")
param['datasource']="abcd"
param['granularity_sqla']="released"
param['order_desc']=True
param['row_limit']=10000
param['viz_type']='dist_bar'
data1['viz_type']='dist_bar'

param['metrics'].append(metric)
param['timeseries_limit_metric']=metric2
data1['params']=string_param
# Test
```

Khi ta đã lấy được thông tin các chart thì dựa vào đó để tạo chart mới . Thay đổi các tham số trong param để phù hợp với yêu cầu .

Dashboard[0] : id dashboard mà chart sử dụng

Datasource_id : dataset sử dụng để vẽ chart

Slice_name : tên chart

Column _ id : id cột trong dataset để sử dụng metric

Column_name : tên cột

Aggregate : hàm tổng hợp (SUM,COUNT,MAX,..)

Labe :nhãn

Groupby: Group by theo cột

Datasource : tên datasource

Granularity_sqla : time column

Order_desc : sắp xếp giảm dần

Row_limit : giới hạn dòng

Viz_type : kiểu chart muốn vẽ

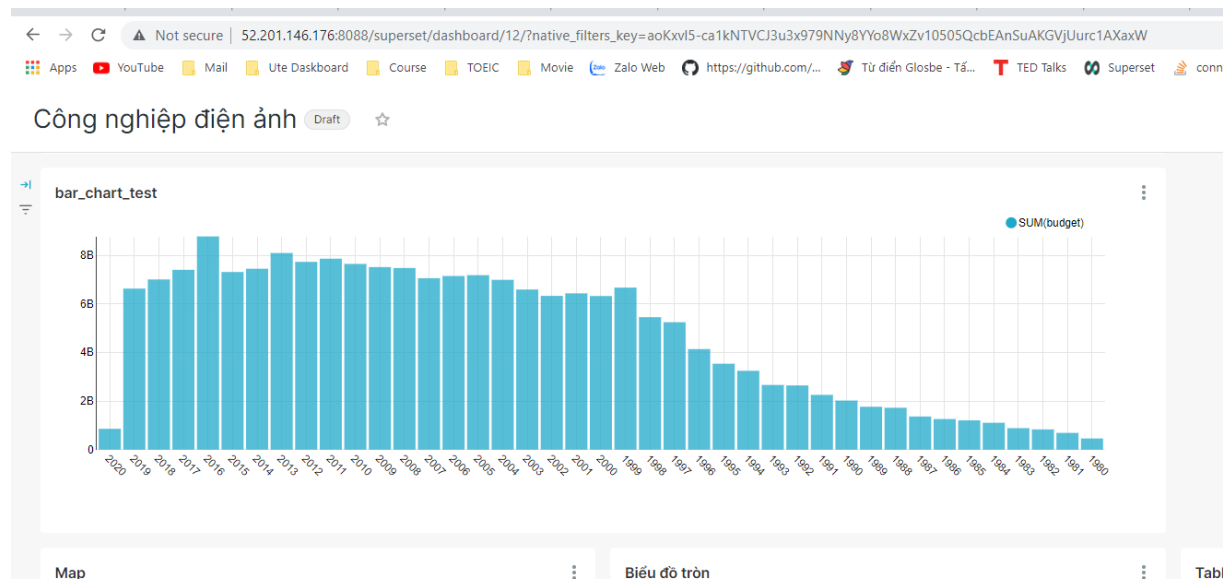
Sau khi khai báo các tham số cần thiết thì ta sử dụng api với phương thức POST để tạo chart:

```
r4 = requests.post(base_url + '/api/v1/chart/', headers = headersAuth, json= data1)
print(r4.status_code)
resp_chart = r4.json()
resp_chart["id"]

# for item in param:
#     print(item)
✓ 0.5s

<class 'dict'>
<class 'dict'>
<class 'dict'>
{
  "cache_timeout": 0,
  "certification_details": null,
  "certified_by": null,
  "dashboards": [
    12
  ]
}
```

Kết quả trên superset



Để cập nhập chart thì ta sẽ thay đổi các tham số cần thiết . Ở đây ta sẽ đổi tên thành Số lượng phim , và metric là SUM(name)

```
data1['dashboards'][0]=12
# data1['dashboards'].append(200)
data1['datasource_id']=24
data1['datasource_type']="table"
data1['slice_name']="Số lượng phim"
data1['params']=""
# data1['datasource_name']="abc"

# metric
metric['column']['id']=763
metric['column']['column_name']="name"
metric['aggregate']="COUNT"
metric['label']="Số lượng phim"
metric['expressionType']='SIMPLE'

metric2['column']['id']=766
metric2['column']['column_name']="year"
metric2['aggregate']="MAX"
metric2['label']="MAX(year)"
metric2['expressionType']='SIMPLE'
```

Để cập nhập chart ta dùng phương thức put với id chart (1669) cần cập nhập

```
r4 = requests.put(base_url + '/api/v1/chart/1669',headers = headersAuth,json= data1)
print(r4.status_code)
resp_chart = r4.json()
resp_chart["id"]
```

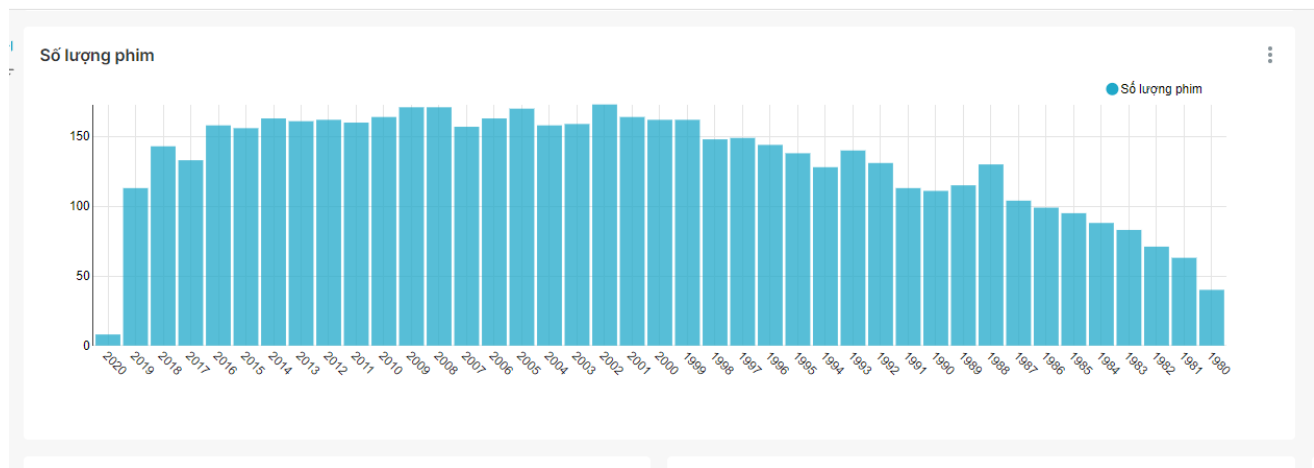
✓ 0.5s

200

1669

Kiểm tra kết quả

Công nghiệp điện ảnh Draft ☆



Để xoá chart ta dùng phương thức delete với id chart cần xoá (1669)

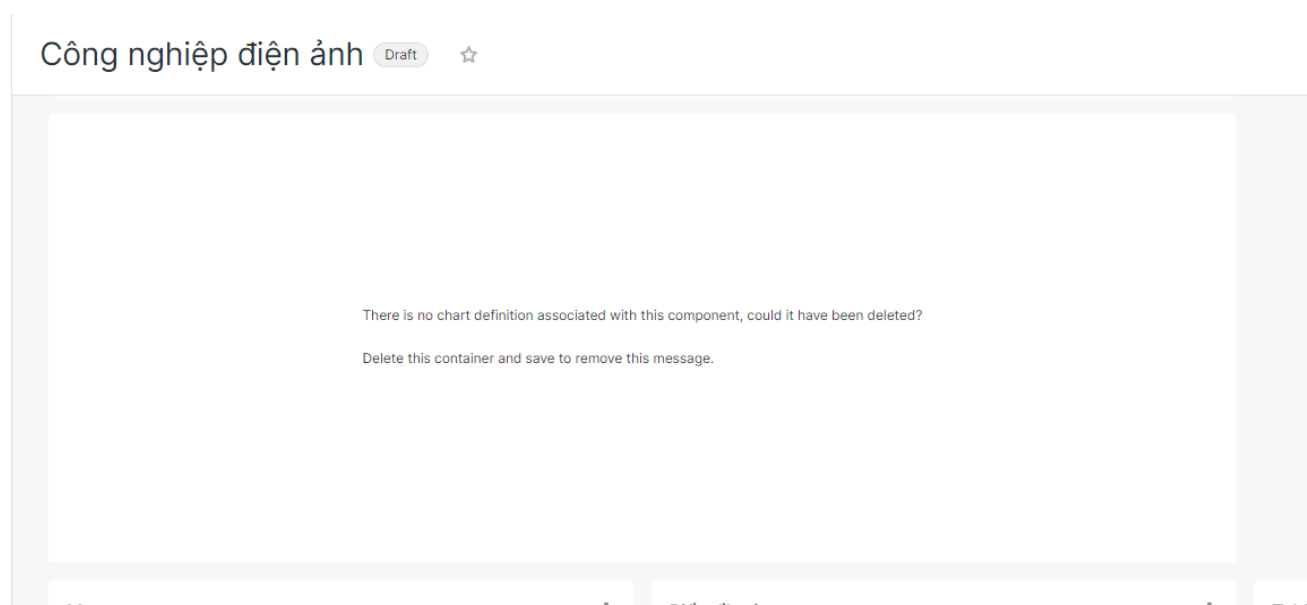
```
r5 = requests.delete(base_url + '/api/v1/chart/1669', headers = headersAuth)

resp_chart = r5.json()
resp_chart
```

[29] ✓ 0.5s

```
{'message': 'OK'}
```

Kết quả trên superset



ĐÁNH GIÁ VÀ KẾT LUẬN

1.1 Kết quả đạt được

- Tìm hiểu được superset và các khái niệm liên quan
- Áp dụng được những kiến thức tìm hiểu được để xây dựng dashboard và vẽ biểu đồ.
- Hiểu được jinja template và áp dụng làm filter box
- Tích hợp được api để thực hiện các yêu cầu CRUD

1.2 Hạn chế

- Do thời gian có hạn, một số chức năng của ứng dụng vẫn chưa hoạt động đúng mong đợi
- Chưa xây dựng frontend cho phần api

1.3 Hướng phát triển

- Tìm hiểu và xây dựng thêm các chart để có nhiều thông tin hơn
- Xây dựng web ui cho phần api để người dùng dễ tương tác hơn
- Tích hợp tự động cập nhập dữ liệu

TÀI LIỆU THAM KHẢO

<https://jinja.palletsprojects.com/>.

<https://superset.apache.org/docs/installation/installing-superset-using-docker-compose>

<https://superset.apache.org/docs/installation/sql-templating>

<https://superset.apache.org/docs/api>

<https://viblo.asia/p/real-time-analytics-airflow-kafka-druid-superset-1Je5EAYj5nL>