

**TRƯỜNG ĐẠI HỌC SƯ PHẠM TP.HCM**

**Khoa Công Nghệ Thông Tin**

**Môn Chuyên đề Oracle**

# **BÀI BÁO CÁO**

**Giảng viên: Võ Tiến An**

**Sinh viên thực hiện:**

- 1. Trần Thị Hồng Hà**
- 2. Nguyễn Thái Sang**
- 3. Bang Minh Kiệt**
- 4. Hoàng Phi Long**

Thành phố Hồ Chí Minh 2020

## Mục Lục

<b>Lời cam đoan.....</b>	<b>1</b>
<b>Danh mục hình ảnh .....</b>	<b>2</b>
<b>Mở đầu.....</b>	<b>3</b>
<b>CHƯƠNG 1 - Tổng quan .....</b>	<b>4</b>
1.1. Mô tả dự án .....	4
1.2. Yêu cầu .....	4
1.2.1. Yêu cầu chức năng.....	4
1.2.2. Yêu cầu phi chức năng.....	4
<b>CHƯƠNG 2 - Cơ sở lý thuyết.....</b>	<b>5</b>
2.1. Dữ liệu.....	5
2.2. Các phương pháp trích xuất dữ liệu .....	5
2.3. Kỹ thuật nâng cao: Database Sharding .....	6
<b>CHƯƠNG 3 - Mô hình và cách giải quyết .....</b>	<b>10</b>
3.1. Cơ sở dữ liệu .....	10
3.2. Mô hình và cách giải quyết .....	14
3.3. Lên lịch cập nhật dữ liệu.....	21
<b>CHƯƠNG 4 - Kết quả thực hiện.....</b>	<b>24</b>
4.1. Oracle Database .....	24
4.2. Kết quả màn hình hiển thị .....	24
<b>CHƯƠNG 5 - Kết luận và khuyến nghị .....</b>	<b>26</b>
<b>Tài liệu tham khảo.....</b>	<b>27</b>

## **Lời cam đoan**

Nhóm em xin cam đoan: Đề tài “Phân tích đại dịch Covid-19” được tiến hành công khai, dựa trên sự cố gắng, nỗ lực của nhóm.

Nhóm em hoàn toàn chịu trách về tính trung thực của các nội dung khác trong đề tài của mình.

Hồ Chí Minh, ngày 23 tháng 6 năm 2020

Tác giả đề tài

Nhóm 9

## Danh mục hình ảnh

Hình 2. 1 Sơ đồ minh họa phân vùng theo chiều ngang và dọc .....	7
Hình 3. 1 Table Oracle Database .....	10
Hình 3. 2 Connect Database.....	10
Hình 3. 3 Lỗi khi insert .....	11
Hình 3. 4 Gán các quyền cho bảng COUNTRIES.....	12
Hình 3. 5 Gán các quyền cho bảng GLOBALS.....	12
Hình 3. 6 Gán các quyền cho bảng COVID_COUNTRIES .....	13
Hình 3. 7 Insert COUNTRIES thành công .....	13
Hình 3. 8 Insert COVID_COUNTRIES thành công.....	14
Hình 3. 9 Insert GLOBALS thành công .....	14
Hình 3. 10 Mô hình MVC .....	15
Hình 3. 11 Hiển thị View .....	15
Hình 3. 12 Lấy dữ liệu từ GLOBALS đề xuất dưới dạng JSON .....	16
Hình 3. 13 Lấy dữ liệu từ bảng 'COVIDS' đề xuất ra dưới dạng json.....	17
Hình 3. 14 Lấy dữ liệu từ bảng COUNTRIES đề xuất ra dưới dạng JSON .....	17
Hình 3. 15 Hiển thị index trong View.....	21
Hình 3. 16 Tạo Task Schedule .....	22
Hình 3. 17 Tạo mới một trigger .....	22
Hình 3. 18 Tạo mới một action .....	23
Hình 4. 1 Tổng quan màn hình .....	24
Hình 4. 2 Hiển thị kết quả với dữ liệu .....	25
Hình 4. 3 Tổng quan .....	25

# Mở đầu

Trong thời điểm hiện tại, đại dịch covid vẫn luôn là chủ đề nóng trong xã hội Việt Nam, và rộng ra là toàn thế giới. Chỉ vài tuần trước các việc này dường như không thể xảy ra – áp đặt và thực thi các biện pháp kiểm dịch nghiêm ngặt và cách ly hàng triệu người – hiện đang là một thực tế ở nhiều quốc gia. Mọi người trên khắp thế giới sẽ phải thích nghi và tạo ra lối sống phù hợp trước một sự kiện phức tạp nhất kể từ Chiến tranh thế giới thứ II.

Nhân loại đang đối mặt với một căn bệnh chưa được hiểu rõ ràng nhưng lại đe dọa nghiêm trọng và gây tử vong cho con người. Hệ thống chăm sóc sức khỏe bị quá tải. Chưa có phương pháp điều trị đặc hiệu nào được chứng minh có hiệu quả, vaccine sẽ không có sẵn trong thời gian sắp tới.

Với mục đích tìm hiểu về đại dịch, cho ra những kết quả cái nhìn sơ lược về tỉ lệ mắc dịch bệnh trong nước nói riêng và toàn thế giới nói chung. Qua một thời gian học tập và tìm hiểu, nhóm em đã hoàn thành báo cáo tổng hợp của mình. Bài báo cáo gồm những chương sau:

## **Chương 1- Tổng quan**

## **Chương 2- Cơ sở lý thuyết**

## **Chương 3- Mô hình và các giải quyết**

## **Chương 4- Kết quả thực hiện**

## **Chương 5- Kết luận và khuyến nghị**

Mặc dù đã cố gắng hết khả năng của mình nhưng do trình độ kiến thức và kinh nghiệm còn hạn chế, nên không tránh khỏi có những sơ sót. Em rất mong được sự nhận xét, đánh giá, đóng góp ý kiến của các đề án này được hoàn thiện hơn.

# CHƯƠNG 1 - Tổng quan

## 1.1. Mô tả dự án

Phân tích đại dịch Covid để có cái nhìn tổng quan nhất về những gì đại dịch đã gây ra.

Bài làm dựa trên dữ liệu được lấy từ các API với link sau:

<https://documenter.getpostman.com/view/10808728/SzS8rjbc?version=latest>

Dựa trên dữ liệu được lấy về và sử dụng Oracle Database và ngôn ngữ PHP để cho thấy cái nhìn tổng quan nhất về tình hình dịch bệnh của các nước trên thế giới, trang web của nhóm em sẽ cho thấy được bản đồ các nước trên thế giới có số ca nhiễm, số ca tử vong, số ca khỏi bệnh từ thời điểm tháng một năm 2020 đến hiện nay.

## 1.2. Yêu cầu

### 1.2.1. Yêu cầu chức năng

Đề tài phân tích dịch covid, xây dựng trang web kết nối cơ sở dữ liệu Oracle để lưu trữ thông tin.

- Vẽ ra được bản đồ thế giới cùng với tổng hợp những gì liên quan tới đại dịch của từng nước.
- Cho thấy tỉ lệ tử vong do đại dịch gây ra cho từng nước.
- Hiện thị dữ liệu dạng bảng về tổng số người bị bệnh, số người khỏi bệnh, số người chết cho từng nước.
- Khai thác tính năng dữ liệu lớn của Oracle là dữ liệu liên quan tới dịch Covid có liên tục, thay đổi theo thời gian.

### 1.2.2. Yêu cầu phi chức năng

- Trang web đảm bảo tính khách quan cho mọi người
- Trang web hỗ trợ độ phân giải trên từng thiết bị

# CHƯƠNG 2 - Cơ sở lý thuyết

## 2.1. Dữ liệu

Hiện nay dịch Covid đang là một vấn đề nóng bỏng, các trang web có rất nhiều API cho vấn đề này. Nhóm em đã chọn API trên trang web phù hợp với ý tưởng đề tài theo đường link: <https://documenter.getpostman.com/view/10808728/SzS8rjbc?version=latest>. Theo như document của trang web thì dữ liệu được cập nhật sau 2 tiếng.

## 2.2. Các phương pháp trích xuất dữ liệu

Phương pháp trích xuất dữ liệu, thì như đã nói trên mục dữ liệu là được lấy API từ trang web.

Cơ sở dữ liệu được tạo ra gồm ba bảng chính: COUNTRIES, COVID\_COUNTRIES, và GLOBALS

Sử dụng ngôn ngữ PHP để trích xuất dữ liệu ra, thì gồm các trình tự sau:

- Connect PHP với cơ sở dữ liệu Oracle
- Sau khi connect thì chúng ta thực hiện lấy dữ liệu từ các đường link
- Thực hiện insert dữ liệu vào cơ sở dữ liệu bằng PHP, insert các dữ liệu có đường link sau:

+ <https://api.covid19api.com/countries>

+ <https://api.covid19api.com/total/dayone/country/south-africa/status/confirmed>

Đường link đầu là lấy các giá trị như là country\_name, slug, ISO2 ứng với bảng dữ liệu được tạo ra trong cơ sở dữ liệu có tên COUNTRIES. Đường link này có các giá trị có ý nghĩa là lấy ra tên quốc gia, chữ cái viết tắt của quốc gia. Đây là code demo:

```
public function getAll() {  
    return $this->_map($this->db->query("SELECT * FROM sys.Countries  
ORDER BY ID")->result_array());  
}
```

Đường Link thứ hai là lấy các giá trị như là confirmed, deaths, recovered, date ứng với bảng dữ liệu được tạo ra trong cơ sở dữ liệu có tên COVID\_COUNTRIES.

Đây là code demo:

```
public function insert ($countryId, $confirmed, $deaths, $recovered, $active,  
$date)  
{
```

```

$query = "EXEC SP_INSERT_COVID_COUNTRIES
($countryId,$confirmed,$deaths,$recovered,$active,TO_DATE('".date_format(date_create($date),"Y-m-d H-i-s")."', 'DD-MM-YYYY HH24:MI:SS'))";
$this->db->query($query);
return $this->_map ( $this->db->row_affect() );
}

```

Đường link này có các giá trị có ý nghĩa là lấy các trường hợp bị nhiễm bệnh, tử vong, hồi phục. Các giá trị này được cập nhật theo từng ngày và theo từng quốc gia.

Còn lại bảng GLOBALS, dữ liệu để trích xuất vào bảng này là ta lấy tổng các dữ liệu từng ngày của từng nước. Đây là code demo:

```

public function getTotalByCountry()
{
    $query = "SELECT * FROM sys.COUNTRIES
    LEFT JOIN (SELECT sys.covid_countries.country_id, MAX(confirmed)
    confirmed,MAX(deaths) DEATHES,MAX(recoverd)
    RECOVERED,MAX(active) ACTIVE
    FROM sys.COVID_COUNTRIES GROUP BY country_id) covid
    ON covid.country_id = sys.countries.id
    ORDER BY sys.COUNTRIES.id";

    return $this->_map($this->db->query($query)->result_array());
}

```

## 2.3. Kỹ thuật nâng cao: Database Sharding

Bất kỳ ứng dụng hoặc trang web nào có sự tăng trưởng đáng kể, cuối cùng đều cần phải mở rộng quy mô để phù hợp với sự gia tăng lưu lượng. Đối với các ứng dụng và trang web có nhiều dữ liệu, điều quan trọng là việc chia tỷ lệ được thực hiện theo cách đảm bảo tính bảo mật và tính toàn vẹn của dữ liệu của dữ liệu. Có thể khó dự đoán mức độ phổ biến của một trang web hoặc ứng dụng hoặc thời gian duy trì mức độ phổ biến đó, đó là lý do tại sao một số tổ chức chọn kiến trúc cơ sở dữ liệu cho phép họ mở rộng quy mô cơ sở dữ liệu của họ một cách linh hoạt.

Định nghĩa về sharding: Sharding là một mẫu kiến trúc cơ sở dữ liệu liên quan đến phân vùng ngang - thực tế tách một hàng bảng Bảng thành nhiều bảng khác nhau, được gọi là partitions. Mỗi partitions có cùng schema và cột, nhưng cũng có các hàng hoàn



toàn khác nhau. Tương tự, dữ liệu được giữ trong mỗi partitions là duy nhất và độc lập với dữ liệu được giữ trong các partitions khác.

Có thể hữu ích khi so sánh suy nghĩ về phân vùng ngang theo cách nó liên quan đến phân vùng dọc. Trong một bảng được phân vùng theo chiều dọc, toàn bộ các cột được tách ra và đưa vào các bảng mới, riêng biệt. Dữ liệu được giữ trong một phân vùng dọc độc lập với dữ liệu trong tất cả các phân vùng khác và mỗi dữ liệu chứa cả các hàng và cột riêng biệt. Sơ đồ sau minh họa cách bảng có thể được phân vùng theo cả chiều ngang và chiều dọc:

**Original Table**

CUSTOMER ID	FIRST NAME	LAST NAME	FAVORITE COLOR
1	TAEKO	OHNUKI	BLUE
2	O.V.	WRIGHT	GREEN
3	SELDA	BAGCAN	PURPLE
4	JIM	PEPPER	AUBERGINE

**Vertical Partitions**

**VP1**

CUSTOMER ID	FIRST NAME	LAST NAME
1	TAEKO	OHNUKI
2	O.V.	WRIGHT
3	SELDA	BAGCAN
4	JIM	PEPPER

**VP2**

CUSTOMER ID	FAVORITE COLOR
1	BLUE
2	GREEN
3	PURPLE
4	AUBERGINE

**Horizontal Partitions**

**HP1**

CUSTOMER ID	FIRST NAME	LAST NAME	FAVORITE COLOR
1	TAEKO	OHNUKI	BLUE
2	O.V.	WRIGHT	GREEN

**HP2**

CUSTOMER ID	FIRST NAME	LAST NAME	FAVORITE COLOR
3	SELDA	BAGCAN	PURPLE
4	JIM	PEPPER	AUBERGINE

*Hình 2. 1 Sơ đồ minh họa phân vùng theo chiều ngang và dọc*

Sharding liên quan đến việc chia dữ liệu thành hai hoặc nhiều phần nhỏ hơn, được gọi là logical shard (phân đoạn logic). Các logical shard sau đó được phân phối trên các database node riêng biệt, được gọi là physical shard (phân đoạn vật lý), có thể chứa nhiều logical shard.

Thông thường, sharding được triển khai ở cấp ứng dụng, nghĩa là ứng dụng bao gồm mã xác định shard nào sẽ truyền đọc và ghi vào. Tuy nhiên, một số hệ thống quản lý cơ sở dữ liệu có các khả năng tích hợp sẵn tính năng sharding, bạn có thể triển khai sharding trực tiếp ở cấp cơ sở dữ liệu.

Lợi ích của Sharding:

+ Sự hấp dẫn chính của việc bảo vệ cơ sở dữ liệu là nó có thể giúp tạo điều kiện mở rộng theo chiều ngang, còn được gọi là nhân rộng. Chia tỷ lệ theo chiều ngang là cách thực hành thêm nhiều máy vào ngăn xếp hiện có để phân tán tải và cho phép lưu lượng truy cập nhiều hơn và xử lý nhanh hơn

+ Nó tương đối đơn giản để có một cơ sở dữ liệu quan hệ chạy trên một máy duy nhất và mở rộng nó khi cần thiết bằng cách nâng cấp tài nguyên máy tính của nó.

+ Đối với một ứng dụng có cơ sở dữ liệu lớn, nguyên khối, các truy vấn có thể trở nên chậm chạp. Tuy nhiên, bằng cách chia một bảng thành nhiều bảng, các truy vấn phải đi qua ít hàng hơn và các tập kết quả của chúng được trả về nhanh hơn nhiều.

Hạn chế của Sharding:

+ Khó khăn đầu tiên mà mọi người gặp phải với sharding là sự phức tạp tuyệt đối của việc thực hiện đúng một kiến trúc cơ sở dữ liệu bị phân mảnh.

+ Nếu được thực hiện không chính xác, có một rủi ro đáng kể rằng quá trình sharding có thể dẫn đến mất dữ liệu hoặc các bảng bị hỏng. Tuy nhiên, ngay cả khi được thực hiện một cách chính xác, việc bảo vệ có thể có tác động lớn đến quy trình làm việc nhóm của bạn. Thay vì truy cập và quản lý một dữ liệu từ một điểm nhập cảnh duy nhất, người dùng phải quản lý dữ liệu trên nhiều vị trí phân đoạn, điều này có khả năng gây gián đoạn cho một số nhóm.

+ Một vấn đề mà người dùng đôi khi gặp phải sau khi hủy cơ sở dữ liệu là các mảnh vỡ cuối cùng trở nên mất cân bằng.

Các Kiến trúc Sharding:

- Key Based Sharding
- Range Based Sharding
- Directory Based Sharding

Do sự phức tạp trong hoạt động mà sharding được thêm vào, sharding thường chỉ được thực hiện khi xử lý một lượng dữ liệu rất lớn. Một số tình huống phổ biến trong đó có thể có ích để bảo vệ cơ sở dữ liệu:

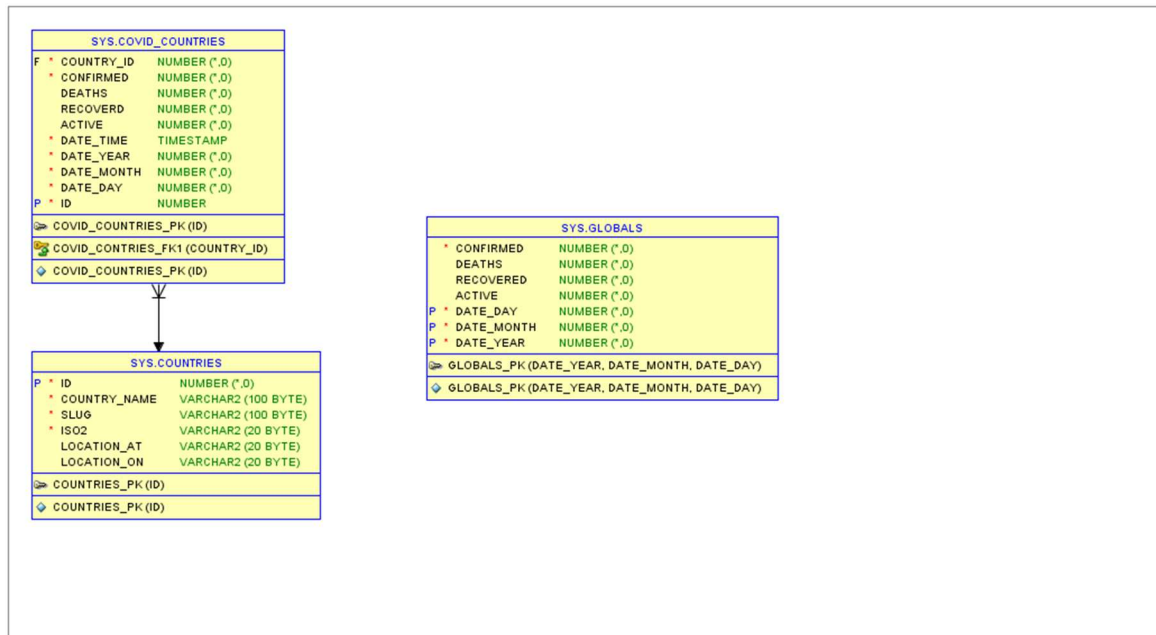
- Lượng dữ liệu ứng dụng lên vượt quá dung lượng lưu trữ của một nút cơ sở dữ liệu.

- Khối lượng ghi hoặc đọc vào cơ sở dữ liệu vượt quá những gì một nút hoặc bản sao đọc của nó có thể xử lý, dẫn đến thời gian phản hồi hoặc thời gian chờ bị chậm.
  - Băng thông mạng được ứng dụng yêu cầu vượt quá băng thông có sẵn cho một nút cơ sở dữ liệu và bất kỳ bản sao đọc nào, dẫn đến thời gian phản hồi hoặc thời gian chờ bị chậm.
  - Trước khi sharding, bạn nên sử dụng tất cả các tùy chọn khác để tối ưu hóa cơ sở dữ liệu của bạn. Một số tối ưu hóa bạn có thể muốn xem xét bao gồm:
    - + **Thiết lập cơ sở dữ liệu từ xa.** Nếu bạn làm việc với một ứng dụng nguyên khối trong đó tất cả các thành phần của nó nằm trên cùng một máy chủ, bạn có thể cải thiện hiệu suất cơ sở dữ liệu của mình bằng cách chuyển nó sang máy của chính nó. Điều này không có thêm sự phức tạp như sharding vì các bảng cơ sở dữ liệu còn nguyên vẹn. Tuy nhiên, nó vẫn cho phép bạn mở rộng quy mô theo chiều dọc cơ sở dữ liệu của bạn ngoài phần còn lại của cơ sở hạ tầng.
    - + **Thực hiện lưu trữ.** Nếu ứng dụng của bạn, thì hiệu suất đọc của bạn là thứ khiến cho bạn gặp rắc rối, bộ nhớ đệm là một chiến lược có thể giúp cải thiện nó. Bộ nhớ đệm liên quan đến việc lưu trữ tạm thời dữ liệu đã được yêu cầu trong bộ nhớ, cho phép bạn truy cập dữ liệu nhanh hơn sau này.
    - + **Tạo một hoặc nhiều bản sao đọc.** Một chiến lược khác có thể giúp cải thiện hiệu suất đọc, điều này liên quan đến việc sao chép dữ liệu từ một máy chủ cơ sở dữ liệu (máy chủ chính) sang một hoặc nhiều máy chủ thứ cấp. Theo đó, mỗi lần ghi mới sẽ chuyển đến bản chính trước khi được sao chép sang phần thứ hai, trong khi các lần đọc được thực hiện riêng cho các máy chủ thứ cấp. Phân phối đọc và ghi như thế này giữ cho bất kỳ một máy nào không phải chịu quá nhiều tải, giúp ngăn chặn sự chậm lại và sự cố.
    - + **Nâng cấp lên một máy chủ lớn hơn.** Trong hầu hết các trường hợp, việc mở rộng một máy chủ cơ sở dữ liệu trên máy chủ thành một máy có nhiều tài nguyên hơn đòi hỏi ít nỗ lực hơn so với sharding. Cũng như việc tạo các bản sao đọc, một máy chủ được nâng cấp với nhiều tài nguyên hơn có thể sẽ tốn nhiều tiền hơn. Theo đó, bạn chỉ nên thực hiện thay đổi kích thước nếu nó thực sự là lựa chọn tốt nhất của bạn.
- ⇒ Sharding có thể là một giải pháp tuyệt vời cho những người muốn mở rộng quy mô cơ sở dữ liệu của họ theo chiều ngang. Tuy nhiên, nó cũng thêm rất nhiều phức tạp và tạo ra nhiều điểm thất bại tiềm năng cho ứng dụng của bạn. Sharding có thể cần thiết cho một số người, nhưng thời gian và tài nguyên cần thiết để tạo và duy trì một kiến trúc bị che khuất có thể vượt xa lợi ích cho những người khác.

## CHƯƠNG 3 - Mô hình và cách giải quyết

### 3.1. Cơ sở dữ liệu

Thiết kế cơ sở dữ liệu trong Oracle có dạng như sau:



Hình 3. 1 Table Oracle Database

Tiếp theo chúng ta connect PHP với Database mình vừa tạo ra:

```
<?php
$db = "(DESCRIPTION=(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521)))(CONNECT_DATA=(SID=orcl)))";

if($c = OCILogon("system", "12345", $db))
{
    echo "Successfully connected to Oracle.\n";
    OCILogout($c);
}
else
{
    $err = OCIError();
    echo "Connection failed." . $err[text];
}

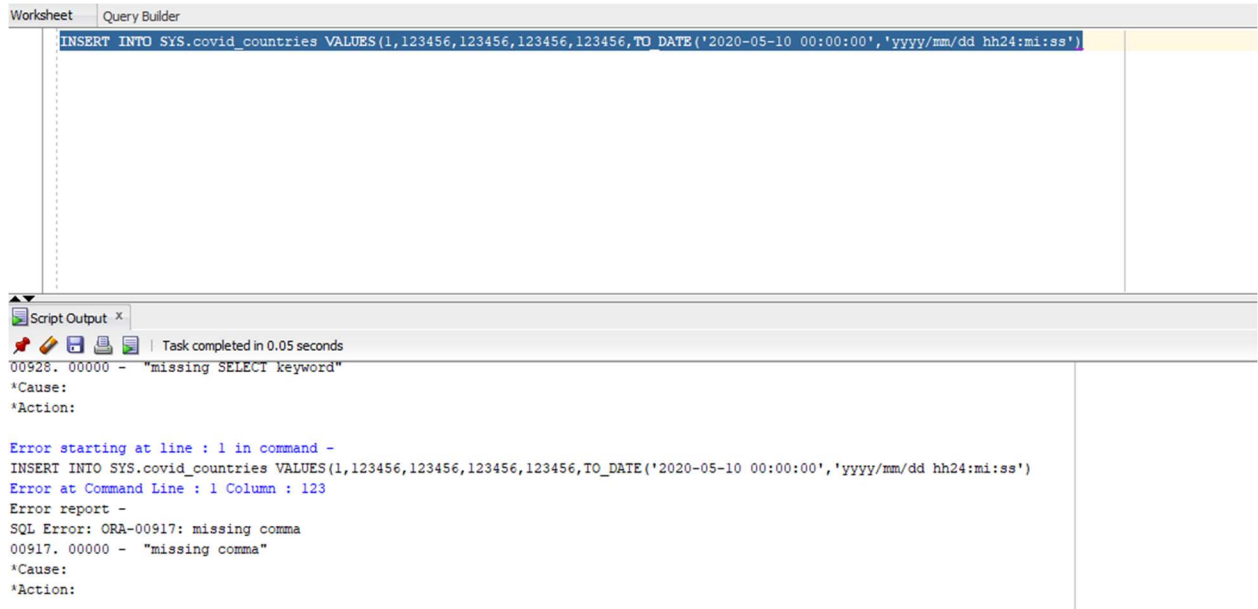
$conn = oci_connect('system', '12345', 'localhost:1521/orcl');
if (!$conn) {
    $e = oci_error();
    trigger_error(htmlentities($e['message'], ENT_QUOTES), E_USER_ERROR);
}

else {
    echo "SUCCESS";
}
```

Hình 3. 2 Connect Database

Sau khi connect Database chúng ta bắt đầu insert dữ liệu vào các bảng.

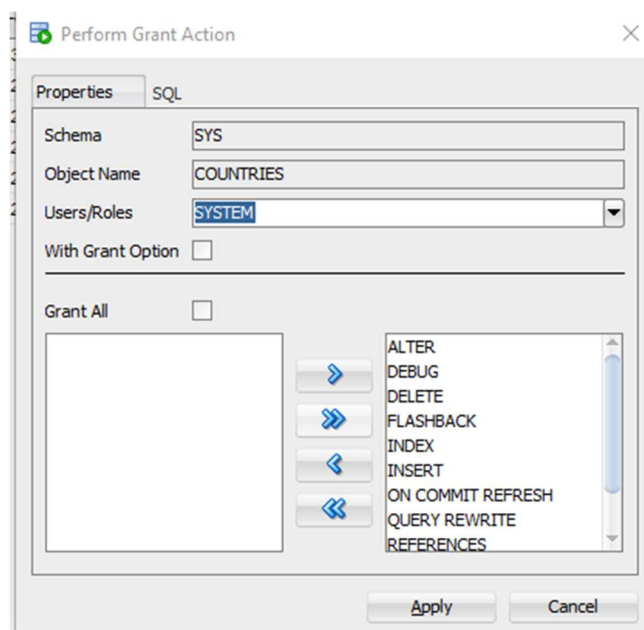
Insert vào một bảng sẽ bị lỗi như sau:



*Hình 3. 3 Lỗi khi insert*

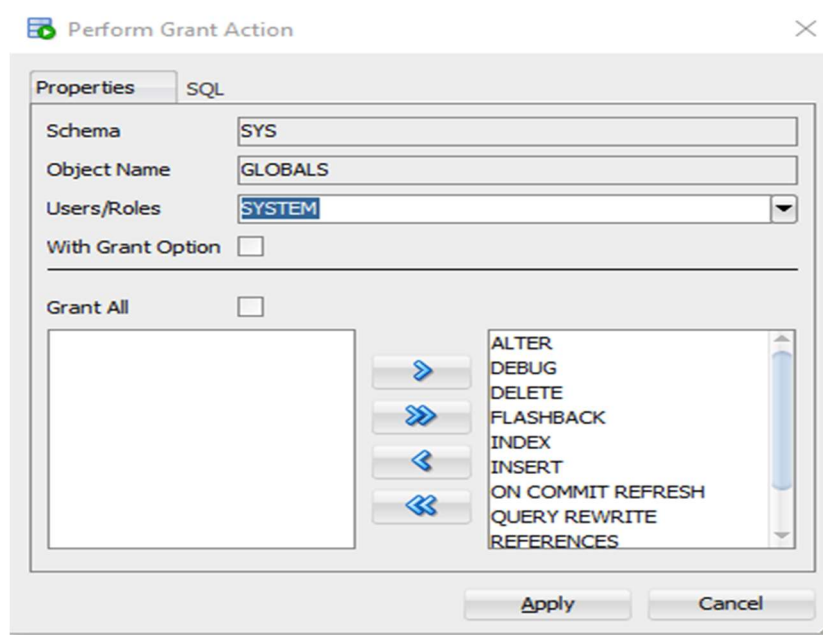
Lỗi trên là do chúng ta chưa phân quyền cho các bảng khởi tạo trong cơ sở dữ liệu Oracle, vậy để insert được thì chúng ta sẽ thực hiện bước phân quyền cho 3 bảng mình vừa tạo:

+ Phân quyền cho bảng COUNTRIES



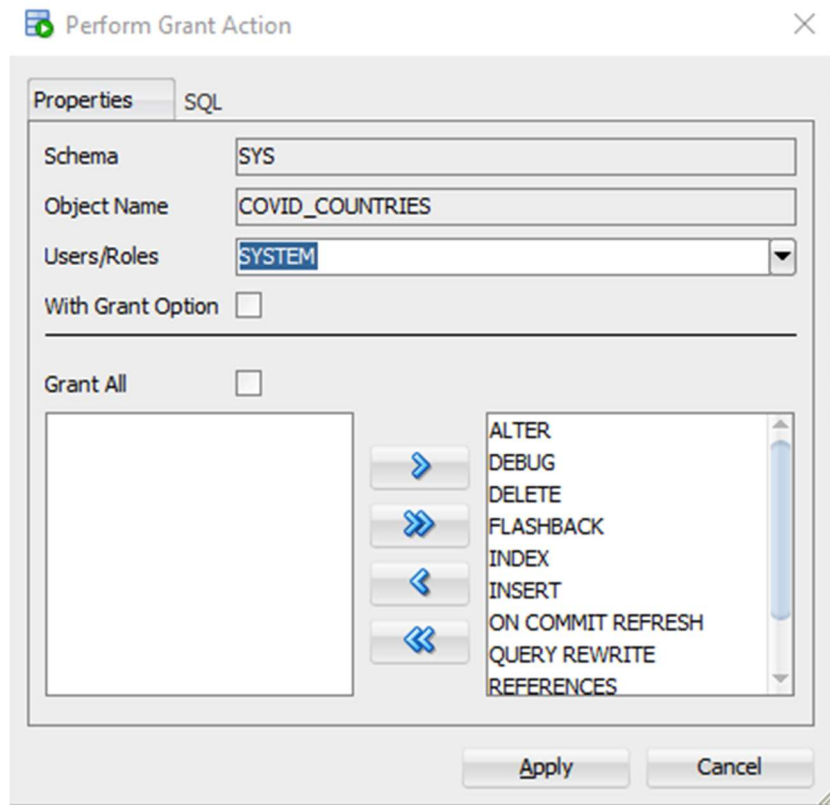
*Hình 3. 4 Gán các quyền cho bảng COUNTRIES*

+ Phân quyền cho bảng GLOBALS



*Hình 3. 5 Gán các quyền cho bảng GLOBALS*

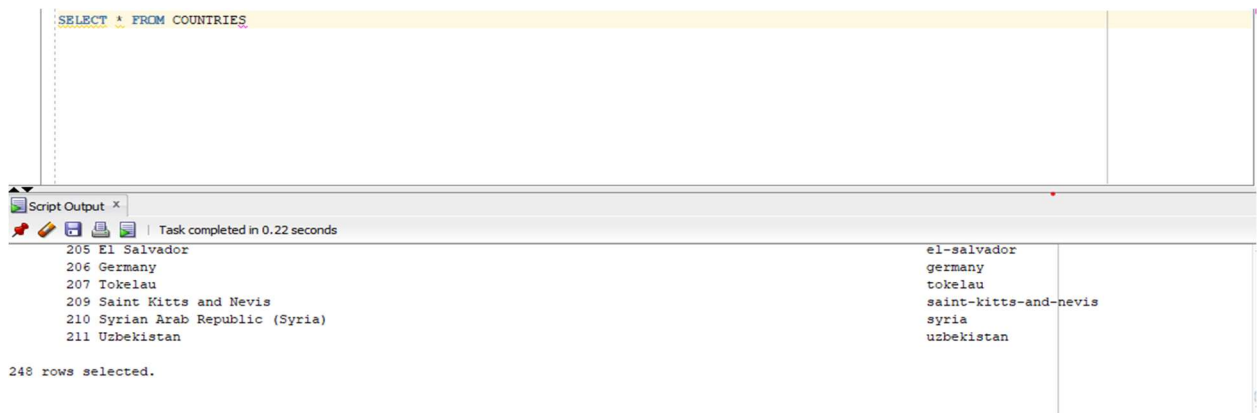
+ Phân tất cả các quyền cho bảng COVID\_COUNTRIES



*Hình 3. 6 Gán các quyền cho bảng COVID\_COUNTRIES*

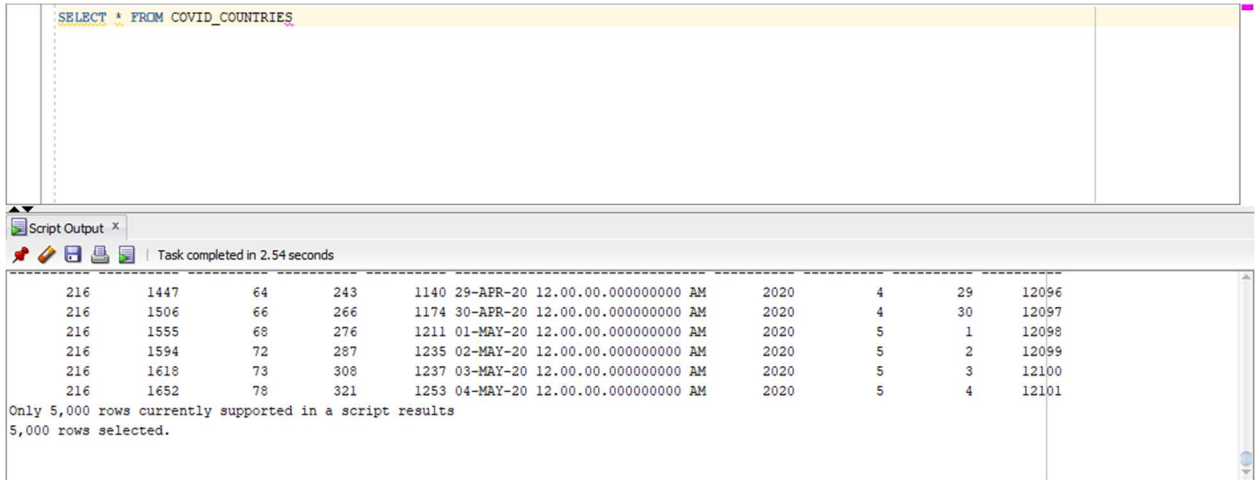
Sau đó ta sẽ insert thành công:

Dữ liệu cho bảng COUNTRIES:



*Hình 3. 7 Insert COUNTRIES thành công*

Dữ liệu cho bảng COVID\_COUNTRIES



```
SELECT * FROM COVID_COUNTRIES
```

Script Output x

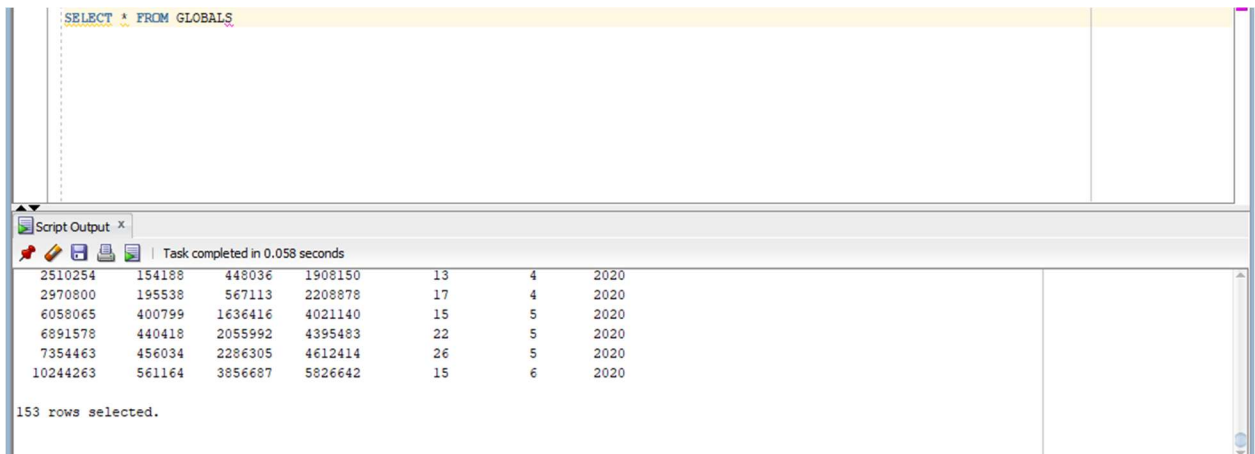
Task completed in 2.54 seconds

216	1447	64	243	1140	29-APR-20	12.00.00.000000000 AM	2020	4	29	12096
216	1506	66	266	1174	30-APR-20	12.00.00.000000000 AM	2020	4	30	12097
216	1555	68	276	1211	01-MAY-20	12.00.00.000000000 AM	2020	5	1	12098
216	1594	72	287	1235	02-MAY-20	12.00.00.000000000 AM	2020	5	2	12099
216	1618	73	308	1237	03-MAY-20	12.00.00.000000000 AM	2020	5	3	12100
216	1652	78	321	1253	04-MAY-20	12.00.00.000000000 AM	2020	5	4	12101

Only 5,000 rows currently supported in a script results  
5,000 rows selected.

Hình 3. 8 Insert COVID\_COUNTRIES thành công

Dữ liệu cho bảng GLOBALS



```
SELECT * FROM GLOBALS
```

Script Output x

Task completed in 0.058 seconds

2510254	154188	448036	1908150	13	4	2020
2970800	195538	567113	2208878	17	4	2020
6058065	400799	1636416	4021140	15	5	2020
6891578	440418	2055992	4395483	22	5	2020
7354463	456034	2286305	4612414	26	5	2020
10244263	561164	3856687	5826642	15	6	2020

153 rows selected.

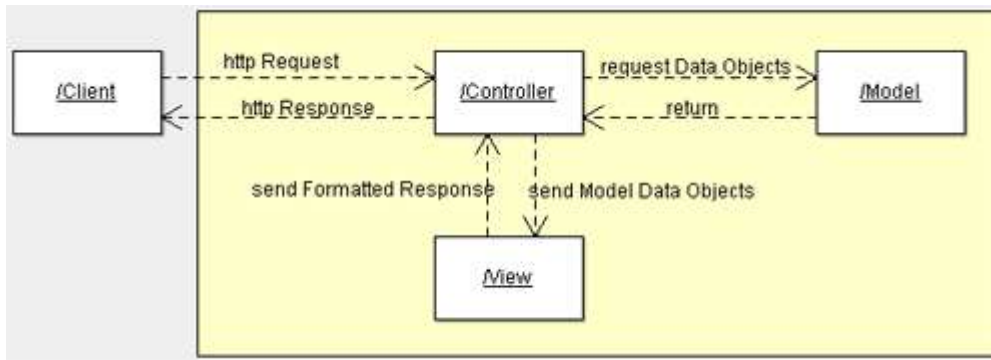
Hình 3. 9 Insert GLOBALS thành công

### 3.2. Mô hình và cách giải quyết

Sử dụng Framework CodeIgniter 3 và ngôn ngữ PHP

Mô hình MVC được chọn làm đề giải quyết cho bài toán của nhóm.





Hình 3. 10 Mô hình MVC

- MVC

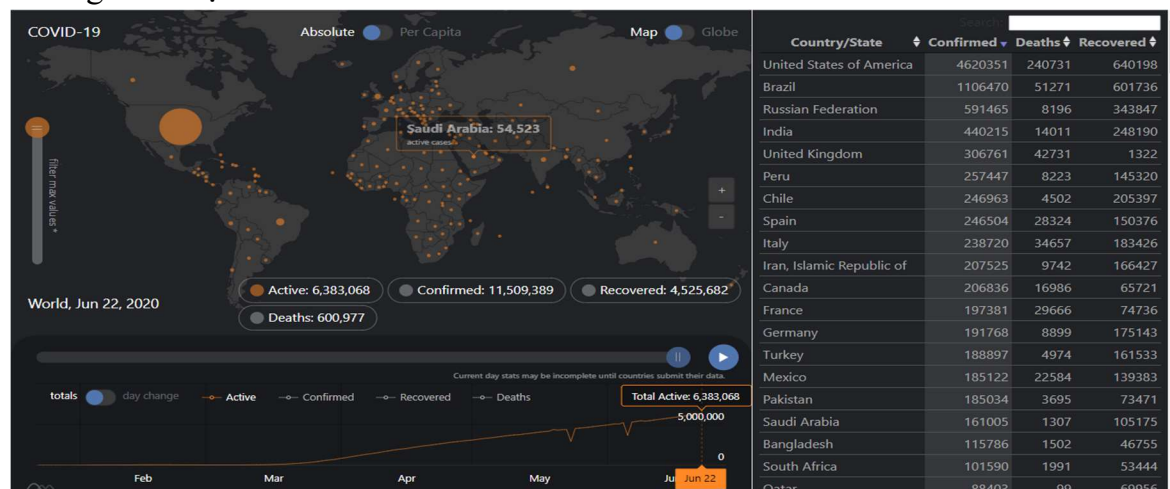
- Controller:

- 1) Home:

```

<?php defined('BASEPATH') or exit('No direct script access allowed');
class Home extends CI_Controller {
    public function __construct() {
        parent::__construct();
    }
    public function index() {
        $this->load->view("index");
    }
}
  
```

Tương hiển thị view:



Hình 3. 11 Hiển thị View

- 2) Globals:

```
<?php defined("BASEPATH") or exit("No direct script access allowed");
class Globals extends CI_Controller
{
    public function __construct() {
        parent::__construct();
        $this->load->model("Global_Model","repo");
    }

    public function getAll() {
        if(isset(getallheaders()["HTTP_X_REQUESTED_WITH"]))
            echo json_encode($this->repo->getAll());
    }
}
```

Lấy dữ liệu từ bảng ‘GLOBALS’ để xuất ra dưới dạng json

Name	x	Headers	Preview	Response	Initiator	Timing	Cookies
<input type="checkbox"/> Oracle_Covid/				▼ [{"confirmed": "555", deaths: "17", recovered: "28", active: "510", date_day: "22", date_month: "1",-,,-}]			
<input type="checkbox"/> core.js				▶ [0 - 99]			
<input type="checkbox"/> charts.js				▼ [100 - 152]			
<input type="checkbox"/> maps.js				▶ 100: {confirmed: "4501208", deaths: "307038", recovered: "1051770", active: "3142544", date_day: "1",-,,-}			
<input type="checkbox"/> dark.js				▶ 101: {confirmed: "4615167", deaths: "313800", recovered: "1092467", active: "3209011", date_day: "2",-,,-}			
<input type="checkbox"/> animated.js				▶ 102: {confirmed: "4722611", deaths: "318886", recovered: "1124591", active: "3279275", date_day: "3",-,,-}			
<input type="checkbox"/> worldLow.js				▶ 103: {confirmed: "4822699", deaths: "324398", recovered: "1162079", active: "3336390", date_day: "4",-,,-}			
<input type="checkbox"/> countries2.js				▶ 104: {confirmed: "4927538", deaths: "332426", recovered: "1198187", active: "3397071", date_day: "5",-,,-}			
<input type="checkbox"/> jquery-3.3.1.min.js				▶ 105: {confirmed: "5045875", deaths: "341561", recovered: "1244768", active: "3459802", date_day: "6",-,,-}			
<input type="checkbox"/> jquery.dataTables.min.css				▶ 106: {confirmed: "5166077", deaths: "349700", recovered: "1284096", active: "3532539", date_day: "7",-,,-}			
<input type="checkbox"/> select.dataTables.min.css				▶ 107: {confirmed: "5286711", deaths: "356705", recovered: "1321405", active: "3608854", date_day: "8",-,,-}			
<input type="checkbox"/> jquery.dataTables.min.js				▶ 108: {confirmed: "5399508", deaths: "362859", recovered: "1374979", active: "3661960", date_day: "9",-,,-}			
<input type="checkbox"/> dataTables.select.min.js				▶ 109: {confirmed: "5498079", deaths: "367155", recovered: "1408329", active: "3722887", date_day: "10",-,,-}			
<input type="checkbox"/> dark.css				▶ 110: {confirmed: "5593671", deaths: "372053", recovered: "1455558", active: "3766358", date_day: "11",-,,-}			
<input type="checkbox"/> app.js				▶ 111: {confirmed: "5700759", deaths: "379543", recovered: "1492763", active: "3828779", date_day: "12",-,,-}			
<input type="checkbox"/> GetAll				▶ 112: {confirmed: "5808010", deaths: "386665", recovered: "1547896", active: "3873779", date_day: "13",-,,-}			
<input type="checkbox"/> GetAll				▶ 113: {confirmed: "5931709", deaths: "393832", recovered: "1587242", active: "3950923", date_day: "14",-,,-}			
<input type="checkbox"/> GetAll				▶ 114: {confirmed: "6050065", deaths: "400799", recovered: "1636416", active: "4021140", date_day: "15",-,,-}			
<input checked="" type="checkbox"/> 128.png				▶ 115: {confirmed: "6176028", deaths: "406259", recovered: "1692546", active: "4077502", date_day: "16",-,,-}			
<input type="checkbox"/> sort_both.png				▶ 116: {confirmed: "6275331", deaths: "410573", recovered: "1733312", active: "4131770", date_day: "17",-,,-}			
<input type="checkbox"/> sort_desc.png				▶ 117: {confirmed: "6385770", deaths: "414711", recovered: "1786224", active: "4185145", date_day: "18",-,,-}			
				▶ 118: {confirmed: "6502590", deaths: "421154", recovered: "1867968", active: "4393804", date_day: "19",-,,-}			
				▶ 119: {confirmed: "6625860", deaths: "427624", recovered: "1896815", active: "4301761", date_day: "20",-,,-}			
				▶ 120: {confirmed: "6758208", deaths: "433812", recovered: "1948088", active: "4376604", date_day: "21",-,,-}			
				▶ 121: {confirmed: "6891578", deaths: "440418", recovered: "2055992", active: "4395403", date_day: "22",-,,-}			
				▶ 122: {confirmed: "7034790", deaths: "448546", recovered: "2111534", active: "4394712", date_day: "23",-,,-}			
				▶ 123: {confirmed: "7132807", deaths: "449228", recovered: "2167912", active: "4515975", date_day: "24",-,,-}			
				▶ 124: {confirmed: "7240324", deaths: "451022", recovered: "2231087", active: "4558500", date_day: "25",-,,-}			
				▶ 125: {confirmed: "7354463", deaths: "456034", recovered: "2286305", active: "4612414", date_day: "26",-,,-}			

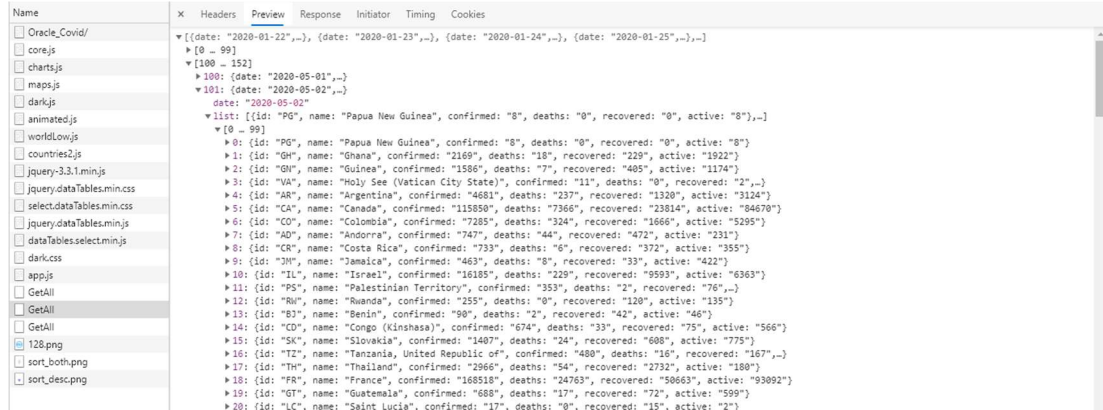
Hình 3. 12 Lấy dữ liệu từ GLOBALS để xuất dưới dạng JSON

3) Coids:

```
<?php defined('BASEPATH') or exit("No direct script access allowed");
class Coids extends CI_Controller {
    public function __construct() {
        parent::__construct();
        $this->load->model("Covid_Model","repo");
    }

    public function getAll() {
        if(isset(getallheaders()["HTTP_X_REQUESTED_WITH"]))
            echo json_encode($this->repo->getAll());
    }
}
```

## Lấy dữ liệu từ bảng ‘COVIDS’ để xuất ra dưới dạng json

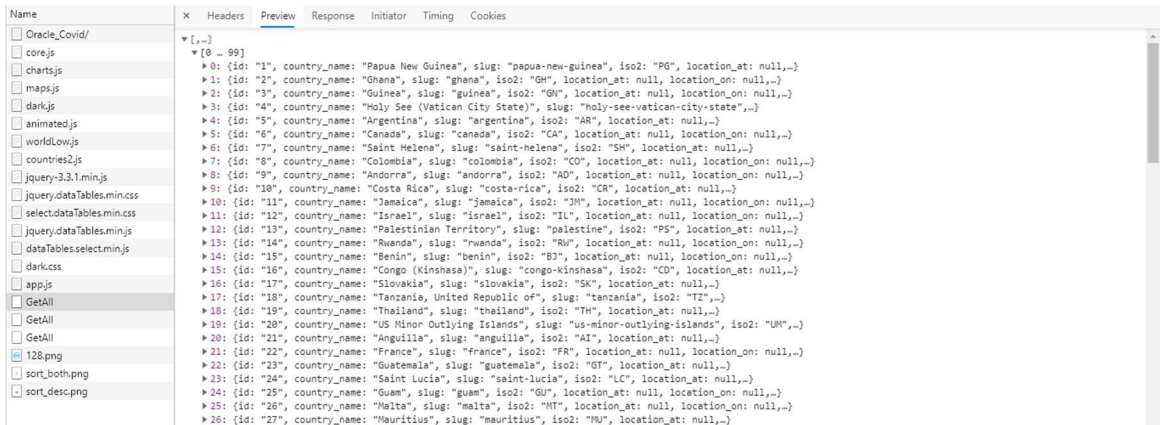


Hình 3.13 Lấy dữ liệu từ bảng ‘COVIDS’ để xuất ra dưới dạng json

### 4) Countries

```
<?php defined("BASEPATH") or exit("No direct script access allowed");
class Countries extends CI_Controller {
    public function __construct() {
        parent::__construct();
        $this->load->model("Country_Model", "repo");
    }
    public function getAll() {
        if(isset(getallheaders()["HTTP_X_REQUESTED_WITH"]))
            echo json_encode($this->repo->getAll());
    }
}
```

## Lấy dữ liệu từ bảng ‘COUNTRIES’ để xuất ra dưới dạng json



Hình 3.14 Lấy dữ liệu từ bảng COUNTRIES để xuất ra dưới dạng JSON

### ➤ Models

## 1) Global

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Global_Model extends CI_Model {
    public function __construct() {
        parent::__construct();
    }
    private function _map($array) {
        return array_map([$this, "_mapDate"], $array);
    }
    private function _mapDate($entity) {
        $entity["date"] = date_format(date_create("$entity[DATE_YEAR]
]-$entity[DATE_MONTH]-$entity[DATE_DAY]"), "Y-m-d");
        return array_change_key_case($entity, CASE_LOWER);
    }
    public function getAll() {
        $query = "SELECT * FROM sys.GLOBALS ORDER BY DATE_YEAR,DATE_
MONTH,DATE_DAY";
        return $this->_map($this->db->query($query)-
>result_array());
    }
    public function getTotalByCountry() {
        $query = "SELECT * FROM sys.COUNTRIES
        LEFT JOIN (SELECT sys.covid_countries.country_id, MAX(confir
med) confirmed,MAX(deaths) DEATHES,MAX(recoverd) RECOVERED,MAX(activ
e) ACTIVE
                FROM sys.COVID_COUNTRIES GROUP BY country_id) co
vid
        ON covid.country_id = sys.countries.id
        ORDER BY sys.COUNTRIES.id";
        return $this->_map($this->db->query($query)-
>result_array());
    }
}
```

## 2) Covid

```
<?php
defined('BASEPATH') or exit('No direct script access allowed');
class Covid_Model extends CI_Model {
    public function __construct() {
        parent::__construct();
    }
    private function _map($array) {
        $_array = [];
        $array = array_map([$this, "_mapDate"], $array);
    }
}
```

```

        foreach($array as $value) {
            $date = date_format(date_create("$value[date_year]-$value[date_month]-$value[date_day]"), "Y-m-d");

            $_find = array_search($date,array_column($_array,"date"))
        );

        if($_find === FALSE) {
            $_find = count($_array);
            $_array[] = [ "date" => $date, "list" => [] ];
        }
        $_array[$_find]["list"][] = [
            "id" => $value["id"],
            "name" => $value["name"],
            "confirmed" => $value["confirmed"],
            "deaths" => $value["deaths"],
            "recovered" => $value["recoverd"],
            "active" => $value["active"]
        ];
    }
    return $_array;
}

private function _mapDate($entity) {
    return array_change_key_case($entity, CASE_LOWER);
}

public function getAll() {
    $query = "
        SELECT sys.COUNTRIES.ISO2 ID, sys.COUNTRIES.COUNTRY_NAME NAME, confirmed,deaths,recoverd,active, DATE_YEAR, DATE_MONTH, DATE_DAY
        FROM (
            SELECT COUNTRY_ID, sum(confirmed) confirmed,sum(deaths)
            deaths,sum(recoverd) recoverd,sum(active) active, DATE_YEAR, DATE_MONTH, DATE_DAY
            FROM SYS.COVID_COUNTRIES GROUP BY DATE_YEAR, DATE_MONTH, DATE_DAY, COUNTRY_ID
            ORDER BY DATE_YEAR, DATE_MONTH, DATE_DAY, COUNTRY_ID
        ) covid
        JOIN sys.COUNTRIES ON sys.countries.id = covid.COUNTRY_ID
    ";
    return $this->_map($this->db->query($query)->result_array());
}

```

```

        public function insert($countryId,$confirmed,$deaths,$recovered,
$active,$date) {
            $query = "EXEC SP_INSERT_COVID_COUNTRIES ($countryId,$confir
med,$deaths,$recovered,$active,TO_DATE('".date_format(date_create($d
ate),"Y-m-d H-i-s")."', 'DD-MM-YYYY HH24:MI:SS'))";
            $this->db->query($query);
            return $this->_map( $this->db->row_affect() );
        }
    }
}

```

### 3) Country

```

<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class Country_Model extends CI_Model
{
    public function __construct() {
        parent::__construct();
    }

    private function _map($array) {
        return array_map([$this,"_mapDate"], $array);
    }

    private function _mapDate($entity) {
        return array_change_key_case($entity,CASE_LOWER);
    }

    public function getAll() {
        return $this->_map($this->db->
query("SELECT * FROM sys.Countries ORDER BY ID")->result_array());
    }
}

```

➤ View:

#### 1) Index

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Covid-19</title>
7   <script src="=> base_url("public/vendors/amcharts4/core.js") ?>"</script>
8   <script src="=> base_url("public/vendors/amcharts4/charts.js") ?>"</script>
9   <script src="=> base_url("public/vendors/amcharts4/maps.js") ?>"</script>
10  <script src="=> base_url("public/vendors/amcharts4/themes/dark.js") ?>"</script>
11  <script src="=> base_url("public/vendors/amcharts4/themes/animated.js") ?>"</script>
12  <script src="=> base_url("public/vendors/amcharts4-geodata/worldLow.js") ?>"</script>
13  <script src="=> base_url("public/vendors/amcharts4-geodata/data/countries2.js") ?>"</script>
14  <script src="=> base_url("public/vendors/jquery/jquery-3.3.1.min.js") ?>"</script>
15  <link rel="stylesheet" media="all" href="=> base_url("public/vendors/datatables/css/jquery.dataTables.min.css") ?>" />
16  <link rel="stylesheet" media="all" href="=> base_url("public/vendors/datatables/css/select.dataTables.min.css") ?>" />
17  <script src="=> base_url("public/vendors/datatables/js/jquery.dataTables.min.js") ?>"</script>
18  <script src="=> base_url("public/vendors/datatables/js/dataTables.select.min.js") ?>"</script>
19  <link rel="stylesheet" media="all" href="=> base_url("public/dark.css") ?>" />
20 </head>
21 <body>
22   <div class="flexbox">
23     <div id="chartdiv"></div>
24     <div id="list">
25       <table id="areas" class="compact hover order-column row-border">
26         <thead>
27           <tr>
28             <th>Country/State</th>
29             <th>Confirmed</th>
30             <th>Deaths</th>
31             <th>Recovered</th>
32           </tr>
33         </thead>
34         <tbody>
35           <tbody>
36         </tbody>
37       </table>
38     </div>
39   </div>
40 </body>
41 <script src="=> base_url("public/app.js") ?>"</script>
42 <script>
43   function finishLoading() {
44     if (!window.countries || !window.covid_world_timeline || !window.covid_total_timeline)
45       return;
46     Start();
47   }
48   function callData(url, callback) {
49     fetch(url, { headers: { "HTTP_X_REQUESTED_WITH": "AJAX" } })
50       .then(b => b.json()).then(b => { callback(b); finishLoading(); });
51   }
52   window.addEventListener("DOMContentLoaded", async () => {
53     callData("=> base_url("Countries/GetAll") ?>", e => window.countries = e)
54     callData("=> base_url("Covids/GetAll") ?>", e => window.covid_world_timeline = e)
55     callData("=> base_url("Globals/GetAll") ?>", e => window.covid_total_timeline = e)
56   })
57 </script>
58 </html>

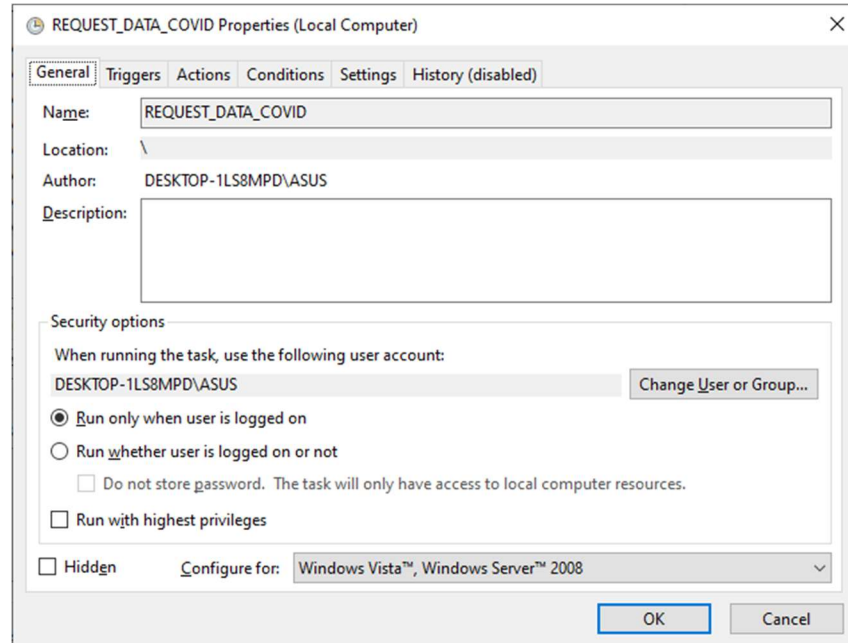
```

Hình 3. 15 Hiện thị index trong View

### 3.3. Lên lịch cập nhật dữ liệu

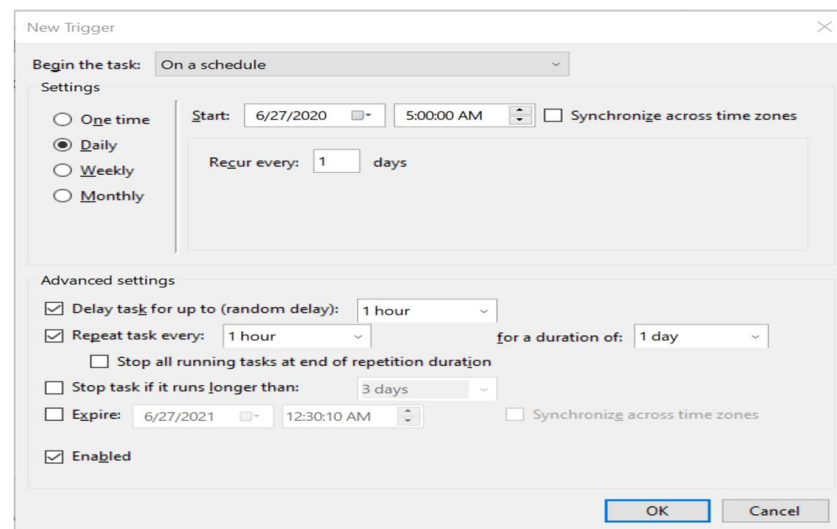


- Tạo một task schedule mới



*Hình 3. 16 Tạo Task Schedule*

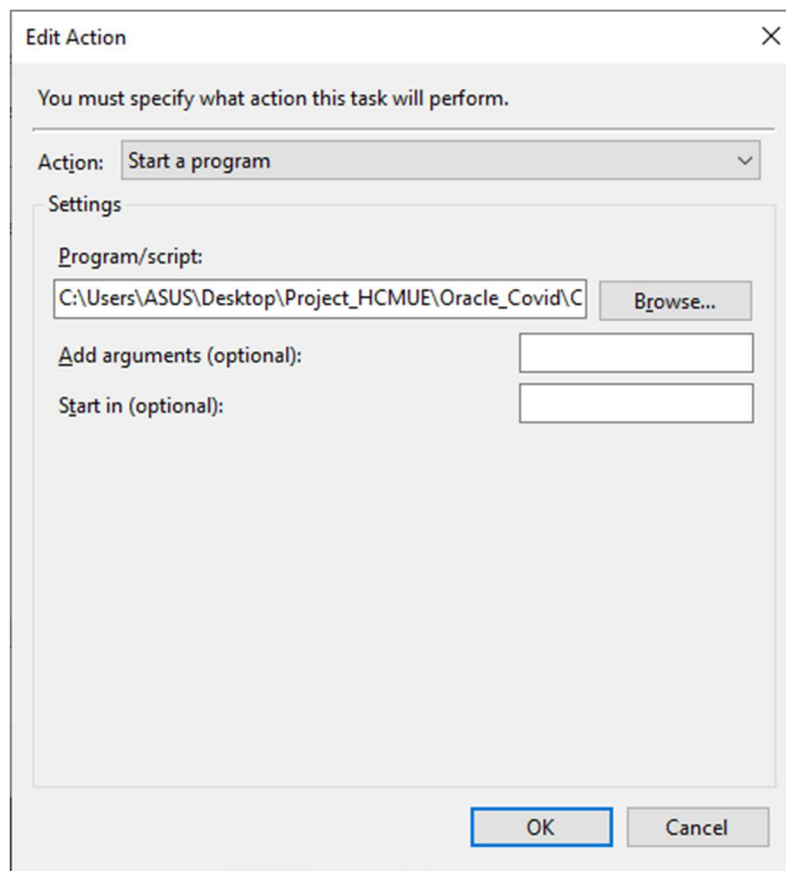
- Tạo một trigger cho



*Hình 3. 17 Tạo mới một trigger*



- Tạo một action dùng để update dữ liệu



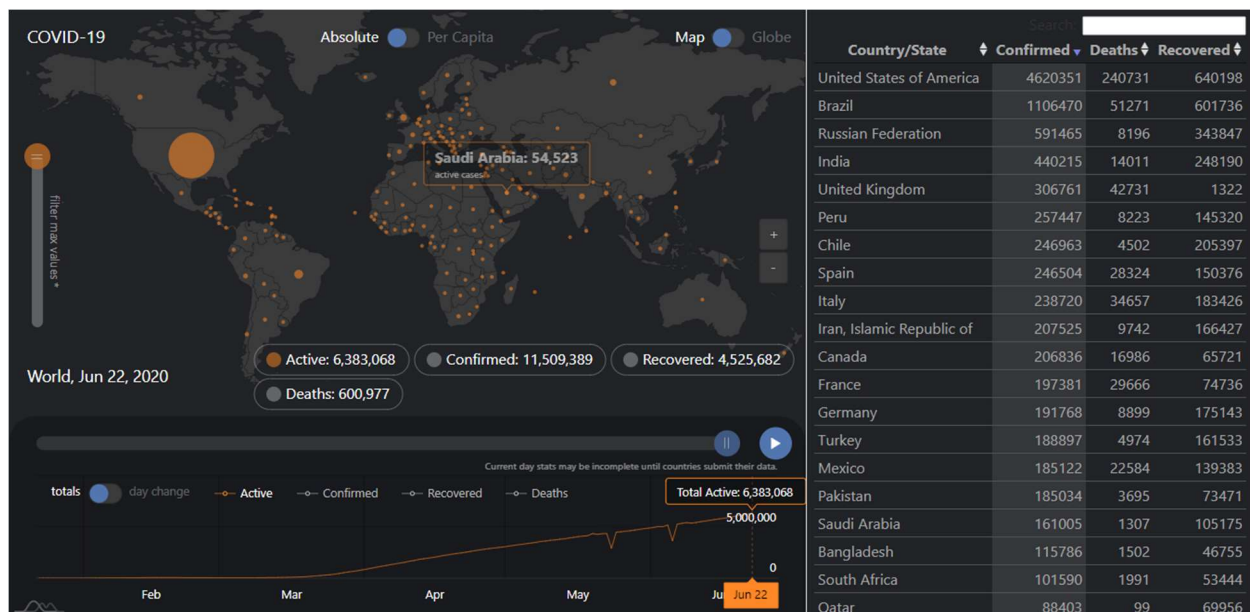
*Hình 3. 18 Tạo mới một action*

## CHƯƠNG 4 - Kết quả thực hiện

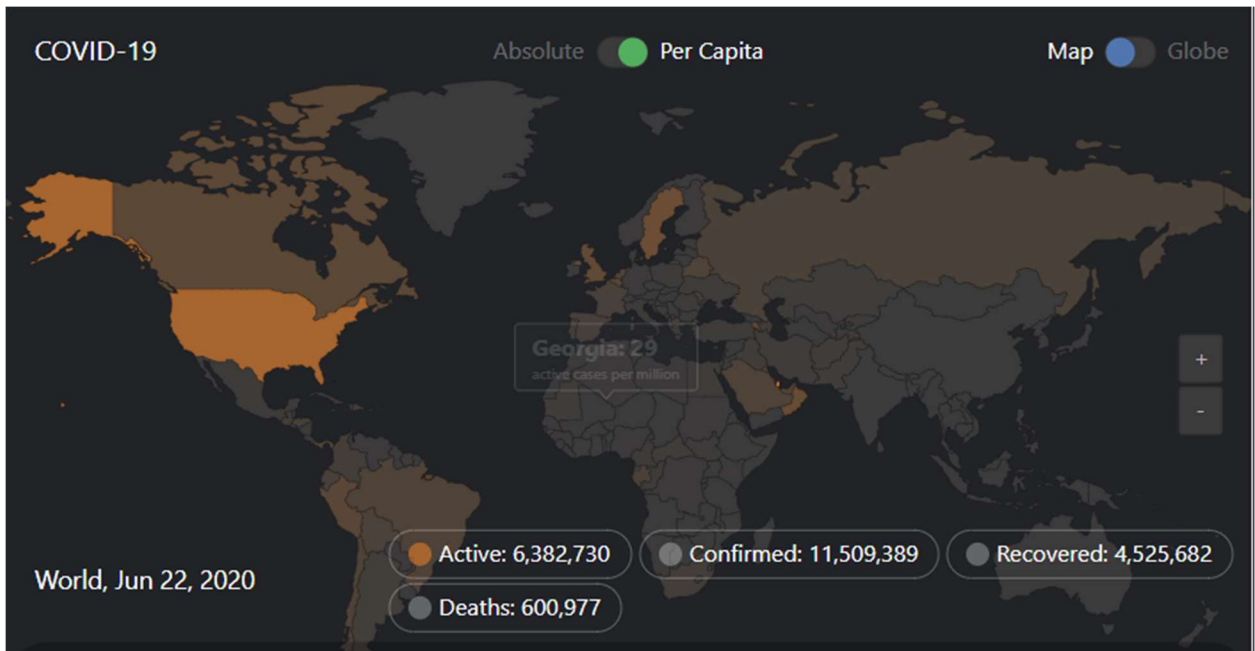
### 4.1. Oracle Database

- Đã lấy được dữ liệu từ các API liên quan đến đề tài vào trong cơ sở dữ liệu oracle
- Sử dụng các kỹ thuật nâng cao như là:
  - + Khóa chính tự động tăng
  - + Store Procedure: để insert thông tin vào bảng COVID\_COUNTRIES với mục tiêu là giảm bớt việc truyền ngày tháng năm từ code vào cơ sở dữ liệu. Store sẽ thực hiện việc lấy ngày tháng năm từ DATE để insert vào bảng covid\_countries.
  - + Trigger sẽ được sử dụng sau khi insert thành công dữ liệu vào bảng COVID\_COUNTRIES. Trigger sẽ kiểm tra dữ liệu theo ngày tháng năm vừa insert vào có tồn tại ở trong bảng GLOBALS hay chưa để tiến hành lựa chọn việc update hay insert mới vào bảng globals

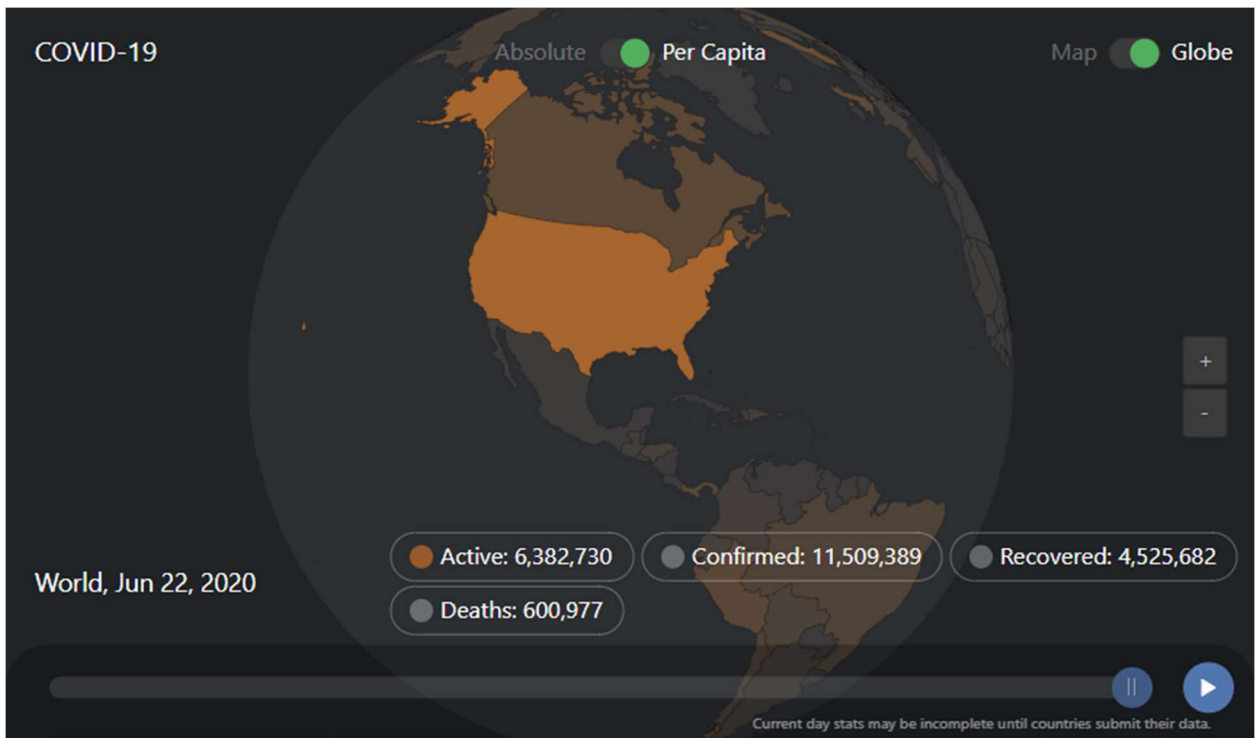
### 4.2. Kết quả màn hình hiển thị



Hình 4. 1 Tổng quan màn hình



Hình 4. 2 Hiện thị kết quả với dữ liệu



Hình 4. 3 Tổng quan

## **CHƯƠNG 5 - Kết luận và khuyến nghị**

Dựa vào dữ liệu lấy được từ API, đã hoàn thành việc vẽ ra bản đồ các nước với số ca mắc bệnh, số ca chết vì dịch bệnh, và số ca hồi phục.

Cho ra cái nhìn tổng quan về dịch bệnh hiện tại của thế giới.

Xác định đề tài, khai thác được tính năng dữ liệu lớn của oracle

Sử dụng được một số xử lý nâng cao cho bài toán nhóm

Tuy nhiên còn mặt hạn chế về mặt kỹ thuật nên bài làm có thể chưa hoàn thiện hẳn.

## Tài liệu tham khảo

- <https://docs.oracle.com/en/>
- <https://codeigniter.com/userguide3/index.html>
- <https://www.amcharts.com/docs/v3/tutorials/using-react/>