PAHO is the specialized international health agency for the Americas which also works with United Nations World Health Organization.

It works with countries throughout the region to improve and protect people's health. PAHO engages in technical cooperation with its member countries to fight communicable and noncommunicable diseases and their causes, to strengthen health systems, and to respond to emergencies and disasters.

We have collected reports from PAHO website on Chikungunya. PAHO’s data sources are: Data source: Cases reported by IHR NFPs to PAHO/WHO and/or through Member States websites.

In total we have collected 204 reports on Chikungunya from PAHO website. These were in pdf formats.

The reports contained the organization Names, logo, Data table and Notes. The data tables contained information of Chikungunya for the following parameters: Country/Region, Epidemiological Weeks, Suspected Cases, Confirmed Cases, Incident rates, Deaths, Imported Cases and Population.

We converted these files to excel using <https://pdftoxls.com> 20 files at each batch conversion. After the conversion we selected all the tables from the converted files and saved only the tables. The files were converted to Microsoft Excel format (.xls) by the pdftoxls website. Now we used the following command to convert the files to Open Spread Sheet format(.ods) in order to import them to MySql Database using phpMyAdmin.

for i in \*.xlsx;

do

libreoffice --convert-to ods $i;

done

After the conversion to ods format we imported the files to MySql using phpMyAdmin. After that we filtered the data depending on some constrains. Some of the data tables of the reports didn’t had all the entries. Which may lead to a dataset with errors. We removed all the entries which didn’t had the following values: Epidemiological Weeks, Suspected Cases, Confirmed Cases.

Due to conversion to excel from pdf, some of the Country names had garbage characters at the end. For example, some Country names contained #,^,g or & and the end of the names, Epidemiological Weeks contained the word “Week”, “WEEk” with the number. We removed al the unnecessary characters from the end of the names. We used the following codes to filter the dataset.

$a = array( '>', '\*', '(1)', '(2)', '(^)', '()', '#', '^', '?', '$', '/', '$', '&' );

$filteredName = rtrim(rtrim(ltrim(str\_replace( $a,"",$row["Country"]))), 'g');

$week = rtrim(ltrim(str\_replace( "WEEK","", $row["Epidemiological Weeks"])));

$week = rtrim(ltrim(str\_replace( "Week","", $week)));

$population = rtrim(ltrim(str\_replace( ",","", $row["Population X 1000"])));

Then, we used the following SQL query “SELECT DISTINCT Country

FROM data” to get the distinct values of Countries from the Country column and saved them in a table. Later we used the Goolg egeocode api to get the latitude and longitude of the location. We wrote a php script that retrieved these values of the distinct Country names from the database and called the google geo code api for each entry in order to get the latitude and longitude of the places. The values were also saved in the database to the corresponding places. A google secret key was required to call the google geo code api, which was retrieved from the google cloud console by enabling google geo code api for the project. The block of code that did the mentioned job is given below.

$sql = "SELECT DISTINCT Country FROM data";

$result = $conn->query($sql);

if ($result->num\_rows > 0) {

while($row = $result->fetch\_assoc()) {

$country = $row['Country'] ;

$service\_url = 'https://maps.googleapis.com/maps/api/geocode/json?address='.urlencode ($country ).'&key=<api key>';

$curl = curl\_init($service\_url);

curl\_setopt($curl, CURLOPT\_RETURNTRANSFER, true);

$curl\_response = curl\_exec($curl);

if ($curl\_response === false) {

$info = curl\_getinfo($curl);

curl\_close($curl);

die('error occured during curl exec. Additioanl info: ' . var\_export($info));

}

curl\_close($curl);

$decoded = json\_decode($curl\_response, true );

if (isset($decoded->response->status) && $decoded->response->status == 'ERROR') {

die('error occured: ' . $decoded->response->errormessage);

}

$latitude = $decoded['results'][0]['geometry']['location']['lat'];

$longitude = $decoded['results'][0]['geometry']['location']['lng'];

$insertSql = 'INSERT INTO city\_info VALUES( "'.$country.'", "'. $latitude .'","'. $longitude . '")';

$result2 = $conn->query($insertSql);

}

Now we have all the geo co ordinates of the Countries collected from the PAHO reports.

Now we need to retrieve the weather information of the Countries of the reported Epidemiological Weeks. After searching for an API that will give us the weather information we chose “Dark Sky API” which provided weather information such as Temperature, Wind speed, Humidity, dew point, summary, Air pressure for first 1000 call for free for each day. Other APIs for getting historical weather data were not free, which included: OpenWeatherMap, Forcast.io, Weather Underground etc.

The Dark Sky API requires a api key, latitude, longitude of the place and the unix time for which the weather data needs to be retrieved.

We used the php DateTime() method to get the exact date of the 'Epidemiological Week of the given year.

$year = $row['Year'];

$week = $row['Epidemiological Weeks'];

$id = $row['id'];

$dto = new DateTime();

$dto->setISODate($year, $week);

$startDate = $dto->format('Y-m-d');

$startDateTimeStamp = $dto->getTimestamp();

Later, we used the Google Timezone API to get the time zones of the countries using their geo coordinates and correct the dates which were converted from Epidemiological week. The code block that executed the mentioned process is given below.

$latitude = $row['lat'];

$longitude = $row['lon'];

//lets get the time zone

$service\_url = 'https://maps.googleapis.com/maps/api/timezone/json?location='.$latitude.','.$longitude.'&timestamp='.$startDateTimeStamp.'&key=<api key>';

$curl = curl\_init($service\_url);

curl\_setopt($curl, CURLOPT\_RETURNTRANSFER, true);

$curl\_response = curl\_exec($curl);

if ($curl\_response === false) {

$info = curl\_getinfo($curl);

curl\_close($curl);

die('error occured during curl exec. Additioanl info: ' . var\_export($info));

}

curl\_close($curl);

$decoded = json\_decode($curl\_response, true );

if (isset($decoded->response->status) && $decoded->response->status == 'ERROR') {

die('error occured: ' . $decoded->response->errormessage);

}

echo "<br><br>";

$timeZoneId = $decoded['timeZoneId'];

$dto->setTimeZone(new DateTimeZone($timeZoneId));

$startDate = $dto->format('Y-m-d');

$startDateTimeStamp = $dto->getTimestamp();

Now we used this unix timestamp and geo coordinates to get the weather information for each entries. The php code that was used to get the weather information is as follows.

$darksky = "https://api.darksky.net/forecast/<api key>/".$latitude.",".$longitude.",".$startDateTimeStamp;

$curl = curl\_init($darksky);

curl\_setopt($curl, CURLOPT\_RETURNTRANSFER, true);

$curl\_response = curl\_exec($curl);

if ($curl\_response === false) {

$info = curl\_getinfo($curl);

curl\_close($curl);

die('error occured during curl exec. Additioanl info: ' . var\_export($info));

}

curl\_close($curl);

$decoded = json\_decode($curl\_response, true );

if (isset($decoded->response->status) && $decoded->response->status == 'ERROR') {

die('error occured: ' . $decoded->response->errormessage);

}

echo "<br>";

$temperature = $decoded['currently']['temperature'];

$summary = $decoded['currently']['summary'];

$dewPoint = $decoded['currently']['dewPoint'];

$humidity = $decoded['currently']['humidity'];

$pressure = $decoded['currently']['pressure'];

$windSpeed = $decoded['currently']['windSpeed'];

$updateSql = "UPDATE data3 SET

weather\_windSpeed = '".$windSpeed."',

weather\_pressure = '".$pressure."',

weather\_temperature = '".$temperature."',

weather\_summary = '".$summary."',

weather\_dewPoint = '".$dewPoint."',

weather\_humidity = '".$humidity."'

WHERE id=".$id;

$conn->query($updateSql);

Some entries such as, rows with country name Peru, Venezuela etc failed to get weather data from the Dark Sky API for the given date. We didn’t used those entries in our final dataset.