# COMP90024 Cluster and Cloud Computing Assignment 2

# City Analytics on the Cloud

## Team 2

Sin I Vong 904974

Wei Liem Tan 900826

Zilong Deng 1020483

Huiyu Yang 976700

Shiyang Chen 931880

# Contents

## 1 Introduction

The aim of this assignment is to improve our knowledge of life in the cities of Australia with the help of Twitter harvesting, AURIN data and CouchDB database. The final solution should be cloud-based and deployed on a multitude of virtual machines across the UniMelb Research Cloud for harvesting and storing tweets through the Twitter APIs. In this assignment, Stream API is chosen for harvesting and the harvesting area is the whole Australia. The IP addresses of three nodes are 172.26.132.17, 172.26.133.152, 172.26.130.38, which respectively corresponds to three scenarios. Similarly, the CouchDB cluster is deployed on these three nodes.

In this assignment, the first scenario is “How many tweets mention nightclub or bar? Are these clustered in certain areas?”. The second is “How many tweets mention covid-19 or coronavirus? Are these clustered in certain areas?”. The last is “How many tweets mention covid-19, financial pressure or pressure? Are these clustered in certain areas?”.

Our work allocation is shown in table 1.1.

| **Work Allocation** | Description | Responsible teammate |
|---|---|---|
| **Scenario 1** | Harvesting tweets, downloading AURIN data and simple MapReduce | Zilong Deng |
| **Scenario 2** | Harvesting tweets, downloading AURIN data and simple MapReduce | Huiyu Yang |
| **Scenario 3** | Harvesting tweets, downloading AURIN data and simple MapReduce | Shiyang Chen |
| **CouchDB Cluster Backend** | Deploying the CouchDB cluster on three nodes | Zilong Deng |
| **Web-based Visualization and Video** | Further data process and visualize the finished data on the front web page. | Sin I Vong and Wei Liem Tan |

Table 1.1 Work Allocation

GitHub repository:
https://github.com/Starry326/ccc_2020

Frontend server address:
http://172.26.132.17/index.html

Youtube demonstration video:
https://www.youtube.com/watch?v=-aX_NztU-S8&feature=youtu.be

## 2 System architecture

This tweets analysis system makes great use of the Melbourne Research Cloud to harvest tweets through Twitter API and shows stories in three different scenarios from the data. Because of the distributed architecture, the system has both flexibility, fault tolerance and efficiency features. It is distributedly deployed in three VMs (172.26.132.17, 172.26.133.152, 172.26.130.38) in the cloud. Moreover, these three nodes can communicate with each other through appropriate security groups settings.

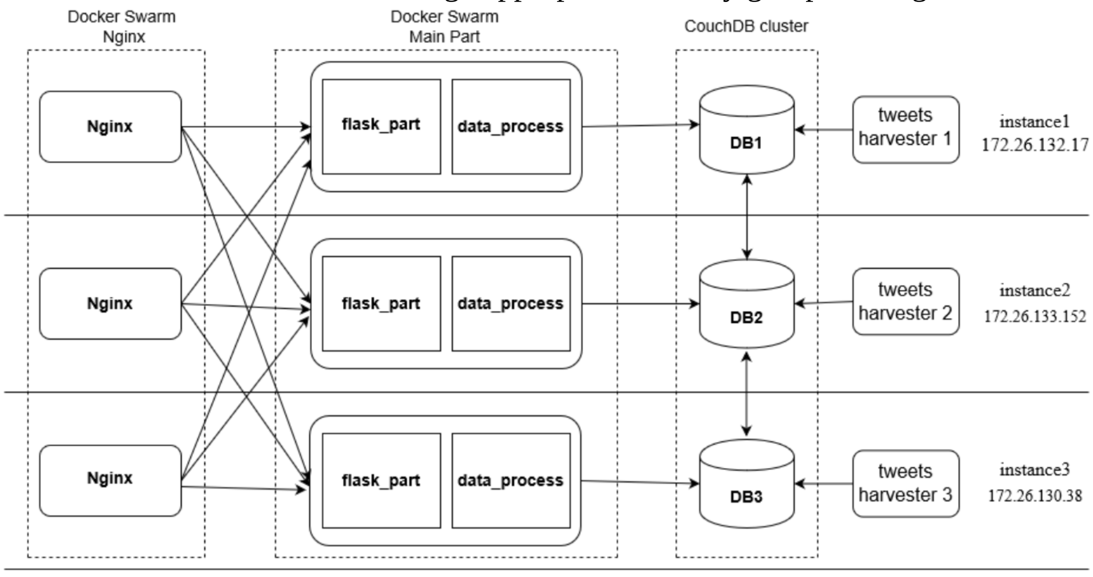


Fig 2.1 System Architectures

### 2.1 CouchDB Cluster

This system uses a CouchDB cluster to store both tweets data and AURIN data. As CouchDB has availability and partition tolerance features, our system can also cope with such kinds of difficulties. The CouchDB services are deployed by docker which can help conquer the complexity of app deployment in different environments. There is complete data in each CouchDB container and these three containers can communicate with each other through port 4369. If the data are updated in one node, the other nodes will also update their data sequentially. Since the data have three copies in three nodes, as long as one node can work well, the breakdown of the databases in other nodes will not cause the breakdown of the whole system.

As the fig 2.1 shows, there are three different tweets harvesters in three instances respectively. They are also deployed by docker and each of them correspond to one scenario and gain tweets through Twitter Stream API. The data in databases are updated in real-time. The more time the harvesters have worked, the more useful data will be stored in databases.

### 2.2 APP Main Body

Our system uses flask as the main technology to build the backend part. As a micro web framework written in Python, flask is easy to use without requirements for other tools and libraries.

In detail, the main body consists of two parts including flask_part and data_process part (Fig 2.1). Flask part comprises routes for different ways to get data from databases as well as routes for html pages. The flask_part should have different responses when gaining different requests. If the requests are asking for data, the flask_part should invoke corresponding data process methods to process and gain data from databases. The processed data should be responded as json format to front end which can be easily analyzed.

The data_process part provides all the necessary methods to count and make calculations based on the data from databases. The information with great formats will be responded to flask_part waiting for further integrations.

The main body services are deployed by Docker Swarm, which ensures the flexibility and tolerance of the system. The service named AppMainBody has three replicas. When one of them breaks down, the others will still work and the Swarm self-healing mechanism will fix the down container as soon as possible. In addition, the Docker Swarm also have load balance mechanism which cooperates with the Nginx to provide more efficient allocations.

## 2.3 Load Balance

For a complex distributed service, load balancers are necessary which make great use of the replicas of the service and markedly improve the efficiency of the system especially when a large number of users visit the app simultaneously. The load balancer will allocate the requests to different modules of the system. Some of them are in one node and some of them are distributed in different nodes.

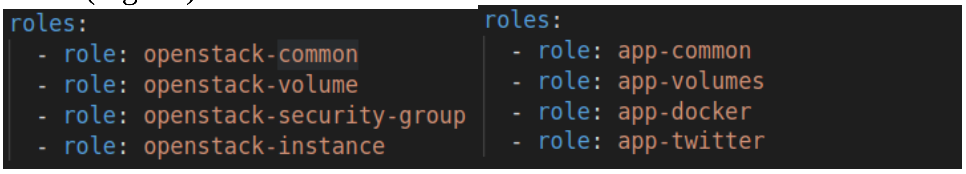
For this system, we use a two-layer load balance mechanism which ensures the efficiency when facing high data throughput situations. The fist layer is Nginx (Fig 3.1), which are deployed by Swarm. Since the Nginx service has three replicas, users can visit the app through any one IP among the three VMs and the exposed port is 80. Round robin is the algorithm to determine which main body of the app should be visited. When the Nginx receives new requests, it will allocate the requests to these three nodes in order. Besides, when one node is down and cannot make valid response, the Nginx will choose another working node.

The second layer is Docker Swarm load balance mechanism. The same as Nginx, it also uses a round robin algorithm to allocate requests to different replicas. The cooperation of these two load balance layers guarantees the availability and efficiency of our system.

# 3 Deployment

## 3.1 Environment Building

For a distributed system, the deployment of each instance is difficult and time wasting. Some operations including the creation of instances, volumes, security groups and basic environment building should be repeated many times. If the environment of each instance has similar parts, Ansible is an excellent technology helping us build the basic environments efficiently and accurately. The basic environment building can be achieved by one ansible playbook. There are a total eight roles each of them has different responsibilities (Fig 3.1).



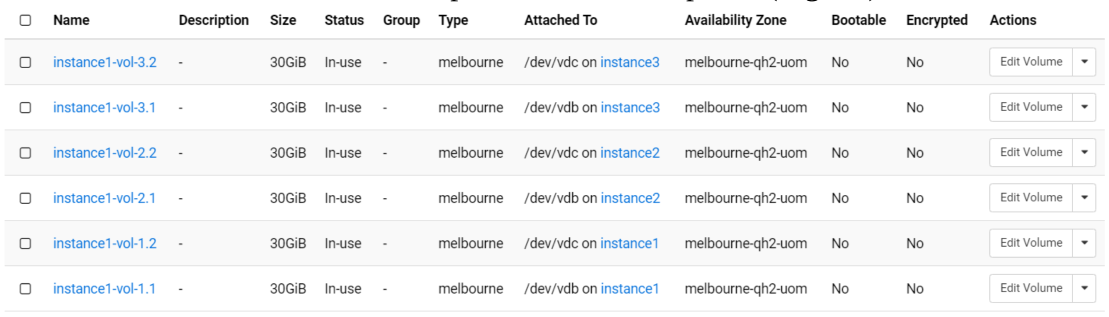
```
roles:
  - role: openstack-common
  - role: openstack-volume
  - role: openstack-security-group
  - role: openstack-instance
```

```
roles:
  - role: app-common
  - role: app-volumes
  - role: app-docker
  - role: app-twitter
```

Fig 3.1 Roles in Ansible Playbook

*Openstack-common* role will do some basic preparatory works in localhost including updating necessary packages such as ‘pip’ and ‘openstack’.

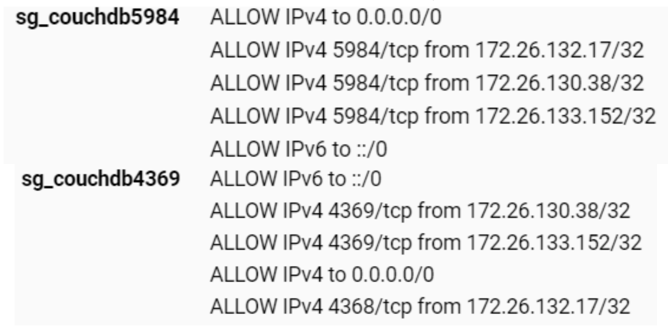
*Openstack-volume* role will create 6 volume with 30 GB storage for each and each instance will be attached two volumes later in openstack-instance phase (Fig 3.2).



| ☐ | Name | Description | Size | Status | Group | Type | Attached To | Availability Zone | Bootable | Encrypted | Actions |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ☐ | instance1-vol-3.2 | - | 30GiB | In-use | - | melbourne | /dev/vdc on instance3 | melbourne-qh2-uom | No | No | Edit Volume |
| ☐ | instance1-vol-3.1 | - | 30GiB | In-use | - | melbourne | /dev/vdb on instance3 | melbourne-qh2-uom | No | No | Edit Volume |
| ☐ | instance1-vol-2.2 | - | 30GiB | In-use | - | melbourne | /dev/vdc on instance2 | melbourne-qh2-uom | No | No | Edit Volume |
| ☐ | instance1-vol-2.1 | - | 30GiB | In-use | - | melbourne | /dev/vdb on instance2 | melbourne-qh2-uom | No | No | Edit Volume |
| ☐ | instance1-vol-1.2 | - | 30GiB | In-use | - | melbourne | /dev/vdc on instance1 | melbourne-qh2-uom | No | No | Edit Volume |
| ☐ | instance1-vol-1.1 | - | 30GiB | In-use | - | melbourne | /dev/vdb on instance1 | melbourne-qh2-uom | No | No | Edit Volume |

Fig 3.2 Volumes List

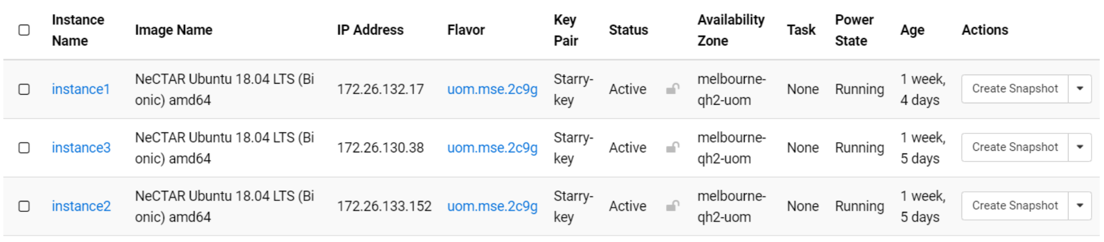
*Openstack-security-group* role will create basic security groups, which both guarantee the communication between these three VMs and protect the databases being visited by users directly. Hence, the security rules for CouchDB should be set to private, which means that only the IP from specific nodes can visit the databases.



| | |
|---|---|
| **sg_couchdb5984** | ALLOW IPv4 to 0.0.0.0/0 |
| | ALLOW IPv4 5984/tcp from 172.26.132.17/32 |
| | ALLOW IPv4 5984/tcp from 172.26.130.38/32 |
| | ALLOW IPv4 5984/tcp from 172.26.133.152/32 |
| | ALLOW IPv6 to ::/0 |
| **sg_couchdb4369** | ALLOW IPv6 to ::/0 |
| | ALLOW IPv4 4369/tcp from 172.26.130.38/32 |
| | ALLOW IPv4 4369/tcp from 172.26.133.152/32 |
| | ALLOW IPv4 to 0.0.0.0/0 |
| | ALLOW IPv4 4368/tcp from 172.26.132.17/32 |

Fig 3.3 Security Groups List

*Openstack-instance* role will create three instances in Melbourne-qh2-uom zone (Fig 3.4). All of them are based on Ubuntu1.8 system and the created volumes will be attached to these instances in order and the security groups will be added. After them are created, the IP addresses will be stored in localhost and be used to do further remote controls.



| ☐ | Instance Name | Image Name | IP Address | Flavor | Key Pair | Status | | Availability Zone | Task | Power State | Age | Actions |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ☐ | instance1 | NeCTAR Ubuntu 18.04 LTS (Bionic) amd64 | 172.26.132.17 | uom.mse.2c9g | Starry-key | Active | 🔓 | melbourne-qh2-uom | None | Running | 1 week, 4 days | Create Snapshot ▾ |
| ☐ | instance3 | NeCTAR Ubuntu 18.04 LTS (Bionic) amd64 | 172.26.130.38 | uom.mse.2c9g | Starry-key | Active | 🔓 | melbourne-qh2-uom | None | Running | 1 week, 5 days | Create Snapshot ▾ |
| ☐ | instance2 | NeCTAR Ubuntu 18.04 LTS (Bionic) amd64 | 172.26.133.152 | uom.mse.2c9g | Starry-key | Active | 🔓 | melbourne-qh2-uom | None | Running | 1 week, 5 days | Create Snapshot ▾ |

Fig 3.4 Instances List

*App-common* role will create some basic folder structures and install some basic tools in these three instances such as 'vim', 'curl' and python related packages. Furthermore, the instance proxy and docker proxy will be added to theses nodes in order to communicate with public addresses.

*App-volume* role will mount the attached two volumes to appropriate folders. One volume will be mounted to the folder of docker, the other one will be mounted to the folder of database. The total 60 GB storage will be enough for data storage and docker modules.

*App-docker* role will install the latest Docker and docker-compose in each VMs.

*App-twitter* role is the last role in playbook. It just creates folders for apps which can help with further app deployments.

## 3.2 System Deployment

All the system modules are deployed by docker. Docker is an open platform for developing, shipping, and running applications. It enables us to separate our applications from our infrastructure so that we can deliver software quickly. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, we can significantly reduce the delay between writing codes and running it in production. The Nginx, AppMainBody and tweets_harvesters are built as images by Dockerfiles and uploaded to DockerHub. When they are deployed in VMs, they are pulled from DockerHub.

- **Database**

The containers of CouchDB in the CouchDB cluster are deployed in each instance respectively. Ports mapping includes 4369:4369, 5984:5984 and 9100-9200:9100-9200 ensure that they can communicate with each other well and cooperate with harvesters and app main bodies well (Fig 3.5). For security reasons, these ports are only opened between these three instances which means that the users cannot visit the databases directly from other IP addresses.

Because we have a large number of tweets harvested every day by the harvesters, the storage of the CouchDB should be considered. One of the mount points of 30 GB volume is the folder of CouchDB, which ensures that we can store a large number of tweets.

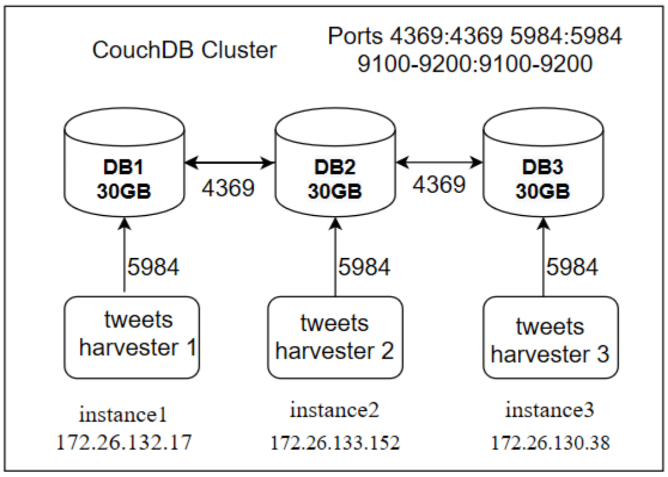


Fig 3.5 Nginx Architecture

- **APP Main Body**

The main body of this system is also deployed by docker swarm (Fig 3.6). In order to communicate with databases in their own VMs, the network of AppMainBody service is set as ‘host’. This means that the main bodies share the same ports with their own VMs and can communicate with database through 127.0.0.1:5984. Although the host network model may not be an excellent solution in complex systems because it limits the available numbers of ports. Instead, overlay network of Swarm may be a good choice. However, in this system, it can work well without considering the limitation of ports and make it easy for this service to communicate with database service. Besides, the Nginx can also communicate with this part through the IP of the instances since the containers of main body share the port 5000 with VMs.

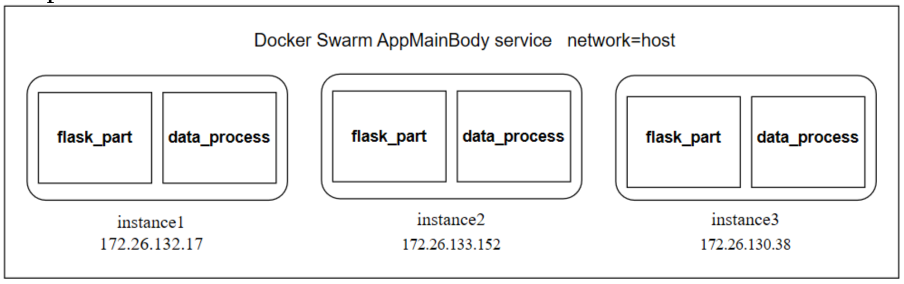


Fig 3.6 App main body Architecture

- **Nginx**

Similar to the deployment of app main body, Nginx containers are deployed by Swarm. The difference is that they use port mapping instead of host network model. As the AppMainBody service uses host network model, Nginx can easily transfer requests to each of main body containers through their remote IP addresses. Some parts of the Nginx configuration are shown in fig 3.8. The exposed port of Nginx is 80, and it will deliver the received requests to the three instances in order.

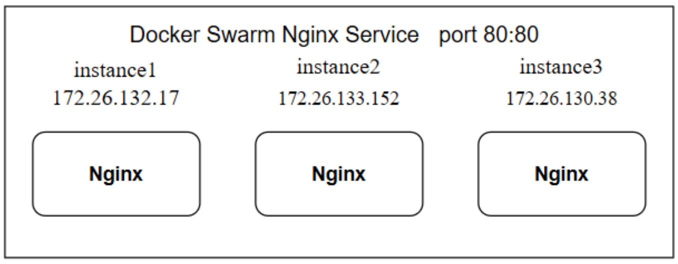
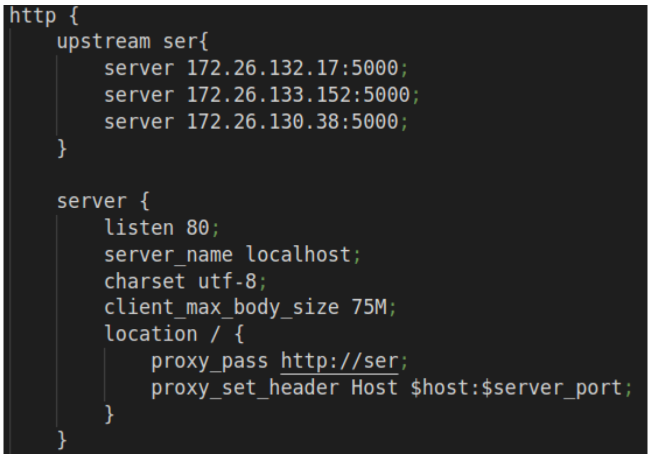


Fig 3.7 Nginx Architecture



```
http {
    upstream ser{
        server 172.26.132.17:5000;
        server 172.26.133.152:5000;
        server 172.26.130.38:5000;
    }

    server {
        listen 80;
        server_name localhost;
        charset utf-8;
        client_max_body_size 75M;
        location / {
            proxy_pass http://ser;
            proxy_set_header Host $host:$server_port;
        }
    }
```

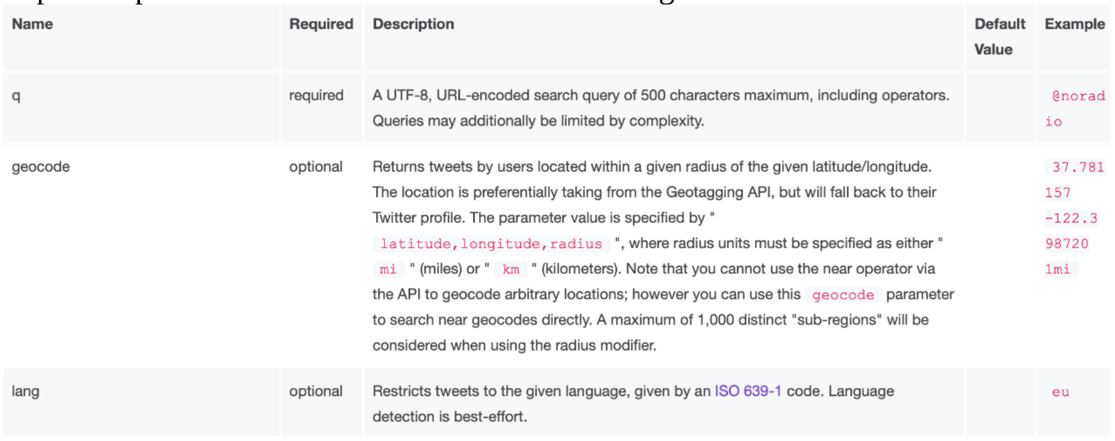
Fig 3.8 Nginx Configuration

## 3.3 Tweet Harvesting

In this assignment, the tweepy library based on Python language is used for harvesting. There are major APIs in this library, search API and stream API.

- **Search API**

Search API returns a collection of relevant Tweets matching a specified query with the largest limitation of 100 tweets. The parameters since_id and max_id record the start and the end of each query. With the help of a loop, harvester can send multiple queries. Some important parameters of search API are shown in fig 3.9

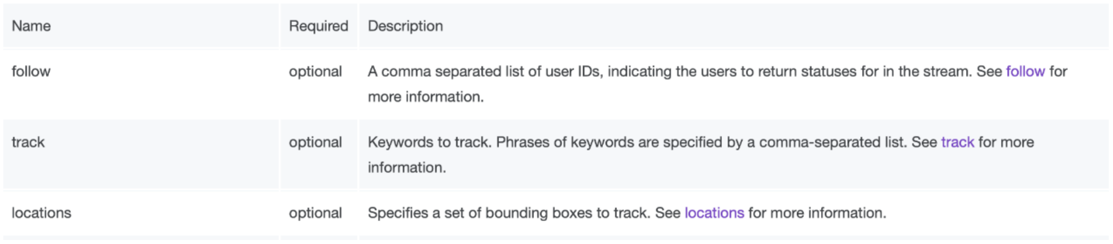


| Name | Required | Description | Default Value | Example |
|---|---|---|---|---|
| q | required | A UTF-8, URL-encoded search query of 500 characters maximum, including operators. Queries may additionally be limited by complexity. | | @noradio |
| geocode | optional | Returns tweets by users located within a given radius of the given latitude/longitude. The location is preferentially taking from the Geotagging API, but will fall back to their Twitter profile. The parameter value is specified by " latitude,longitude,radius ", where radius units must be specified as either " mi " (miles) or " km " (kilometers). Note that you cannot use the near operator via the API to geocode arbitrary locations; however you can use this geocode parameter to search near geocodes directly. A maximum of 1,000 distinct "sub-regions" will be considered when using the radius modifier. | | 37.781157 -122.398720 1mi |
| lang | optional | Restricts tweets to the given language, given by an ISO 639-1 code. Language detection is best-effort. | | eu |

Fig 3.9 Parameters of Search API

- **Stream API**

Stream API is real-timely updated and returns public statuses that match one or more filter predicates. Multiple parameters may be specified which allows most clients to use a single connection to the stream API. Some important parameters of stream API are shown in fig 3.10.



| Name | Required | Description |
|---|---|---|
| follow | optional | A comma separated list of user IDs, indicating the users to return statuses for in the stream. See follow for more information. |
| track | optional | Keywords to track. Phrases of keywords are specified by a comma-separated list. See track for more information. |
| locations | optional | Specifies a set of bounding boxes to track. See locations for more information. |

Fig 3.10 Parameters of Stream API

Finally, the stream API is chosen for this assignment. Compared to search API, stream API can acquire real-time tweet data continuously. This is more convenient on a large demand of data. Besides, the geolocation of search API is based on the user location. However, the geolocation of stream API is based on the real-time place of each tweet, which is more accurate and reliable. Apart from that, a bounding box of each area in stream API is easy to be found in AURIN. Most importantly, the track keywords match not only the text attribute of each tweet, but also some other attributes, such as text for hashtags, media, expanded_url and display_url for links and screen_name for user mentions. This can provide extra useful tweets for analysis.

## 3.4 Web-based Visualization

Flask leverages Jinja2 which is one of the most used template engines for Python as its template engine. Jinja2 is a Python template engine used to create templates such as HTML files which are sent to the client via HTTP response.

In this project, Jinja template is used to achieve the separation between the logic of the application and the presentation of the webpages. All the templates are placed in templates folder at the application root level (the app folder) as what Flask expects by default, so that Flask can read the HTML files by making the templates folder available for use with render_template() method, which returns the template (HTML file) from the templates folder with the defined variables (the variables that should be available in the context of the template). For instance, the server will send the template and the relevant data from Twitter and AURIN when it receives the HTTP request from the client. In order to illustrate the meaningful data obtained from Twitter API and AURIN platform, we used a charting framework called AnyChart which helps data visualization.

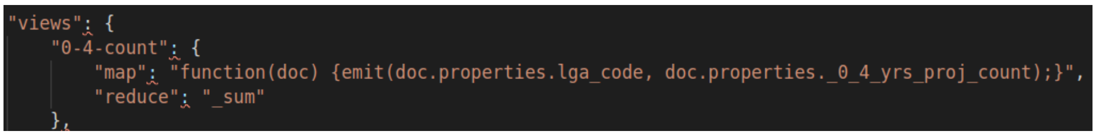
# 4 Scenarios

## 4.1 Scenario 1

This scenario is to analyze the relation between age distribution and the ratio of tweets related to nightclub and bar. The data of this scenario are stored in two databases. One is population database and the other one is tweets database.

Tweets database stores all the tweets located in Australia and contain keywords 'nightclub' or 'bar'. Since the tweets are from Twitter Stream API, if the tweets harvester continuously works, the data in this database will be updated continuously. The Stream API gains data located in Australia by setting the location parameter to locations = [113.080648, -44.134922, 155.252163, -10.768672], which will only gain data in a rectangle zone with specific longitude and latitude of left top point and right down point. This zone is exactly Australia and some sea areas, so that almost all the tweets can be considered as from Australia. Furthermore the gained tweets will be divided by the location feature , which can tell the location that the tweets are published. Another parameter in filter is 'track' which limits the API to gain tweets containing keywords 'nightclub' and 'bar'. Before the tweets are stored in database, their unique ID format should be changed to <region>:id, which is used to make advantage of the CouchDB partition function. When the data are large in volume, partition will help process the data efficiently. It means that when gaining the tweets in a specific region, the tweets in other regions will not be processed. As for this scenario, in order to gain a large number of data in each zone to analyze, the regions are divided by territories and states namely nsw(New South Wales), vic(Victoria), ws(Western Australia), act(Australian Capital Territory), qld(Queensland), nt(Northern Territory), sa(South Australia) and tas(Tasmania).

As for population database, the data are also partitioned by regions like tweets data. The population dataset named 'LGA15 Projected People Age Distribution -2020' is from AURIN and grouped by 5-year age ranges from 0-4 to over 85. It is customized projections prepared for the Australian Government Department of Social Services by the Australian Institute of Health and Welfare, which is rather reliable and official. In order to count the population in each age slot and specific zones, map reduce technique is used to sum the number of people of specific age slot in one specific zone. Like the view of 0-4-count (Fig 4.1), 18 views in total are created to cover the whole range of ages. With these views, it is easy to sum the population in a specific zone and age range based on partition with reduce level 0.



```
"views": {
    "0-4-count": {
        "map": "function(doc) {emit(doc.properties.lga_code, doc.properties._0_4_yrs_proj_count);}",
        "reduce": "_sum"
    },
```

Fig 4.1 View for 0-4 Slot

The data will be displayed in webpages through different graphs. As shown in fig 4.2, there is an Australia Map when cursor is moved to a specific area, it will show the number of gained tweets in this zone.

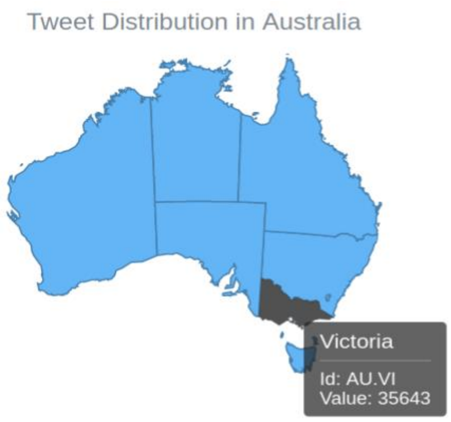


Fig 4.2 Data Display

Besides, a bar chart will show the ratio of tweets containing ‘nightclub’ and ‘bar’ in each zone (Fig 4.3). The high ratio means that with the same population, the people in this zone are more likely to refer to ‘nightclub’ and ‘bar’ in their tweets. It is obvious that Capital Territory has the highest ratio which is 0.0066 and Western Australia has lowest ratio which is 0.00309. The ratio in other zones are New South Wales (0.00489), Victoria (0.00550), Queensland (0.00329), Northern Territory (0.00439), South Australia (0.00414) and Tasmania (0.00369) respectively.

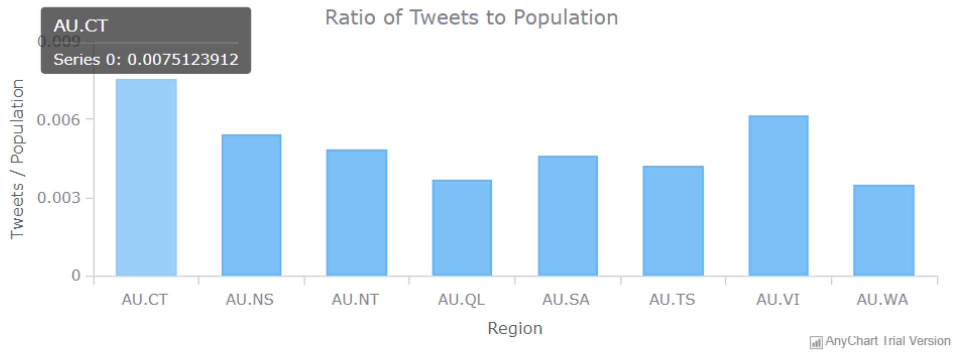


Fig 4.3 Tweets/Population Ratio Chart

As for the bar chart part, users can pick specific zones’ data to be shown in the graph by clicking the legends of zones (Fig 4.4). The relation between tweets and population distribution is far away from our original guess. Before we analyze the data, we thought that the highest ratio will appear in New South Wales or Victoria since metropolises such as Sydney and Melbourne may have more bars and people’s nightlife are more abundant. Besides, the zones which have more young people may have a high ratio of tweets containing keywords ‘nightclub’ and ‘bar’.

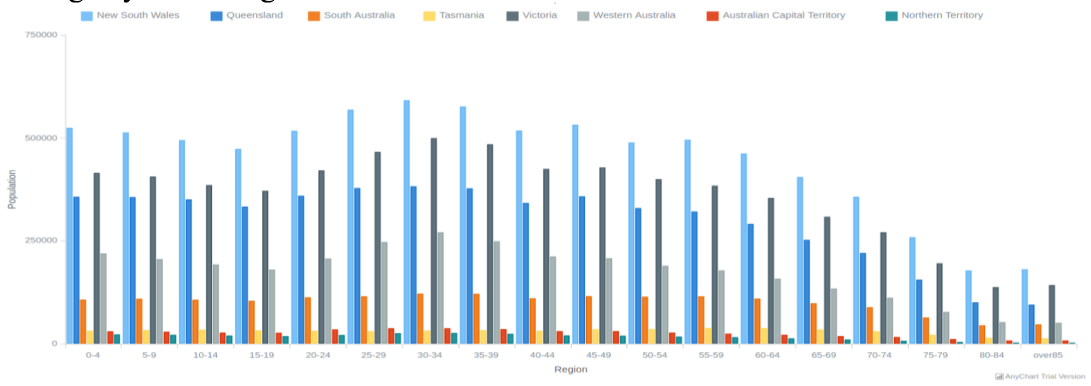


Fig 4.4 Population Display

The bar charts of age distribution of each zone show that except South Australia and Tasmania, all the other zones have similar graph shapes that the peak appear in the age 25-

35 range. Especially, the peak is more obvious in the graphs of Victoria, Western Australia, Australian Capital Territory and Northern Territory.

Compared with the tweets ratios, it is reasonable that Victoria, Australian Capital Territory and Northern Territory have high ratios since the people with age around 30 may be more likely to go to nightclubs and bars. Even though Melbourne and Sydney are both metropolis cities, New South Wales has lower keywords related tweets ratio than Victoria. However, the Western Australia is an exception. It has an obvious young population but has the lowest tweets ratio compared with other zones.

After analysis, we find that some reasons may cause the trend. Firstly, because of the COVID-19, people are all isolated in their home and because we use the stream API to gain real-time data, all the tweets are published during this special period. They may not have a chance to go to bars and nightclubs so that the tweets containing these keywords may not represent that they go to such venues. Instead, they may just refer to such keywords in their tweets, which cannot represent their activities accurately. Secondly, the zone may be too large to represent a kind of age distribution. For example, except Sydney, New South Wales also has a lot of places which are in suburb area. These areas may not have so many nightclubs and bars like big cities. If we can divide the areas in smaller zones, it may have a more clear relation between age distribution and tweets containing keywords like 'bars' and 'nightclubs'.

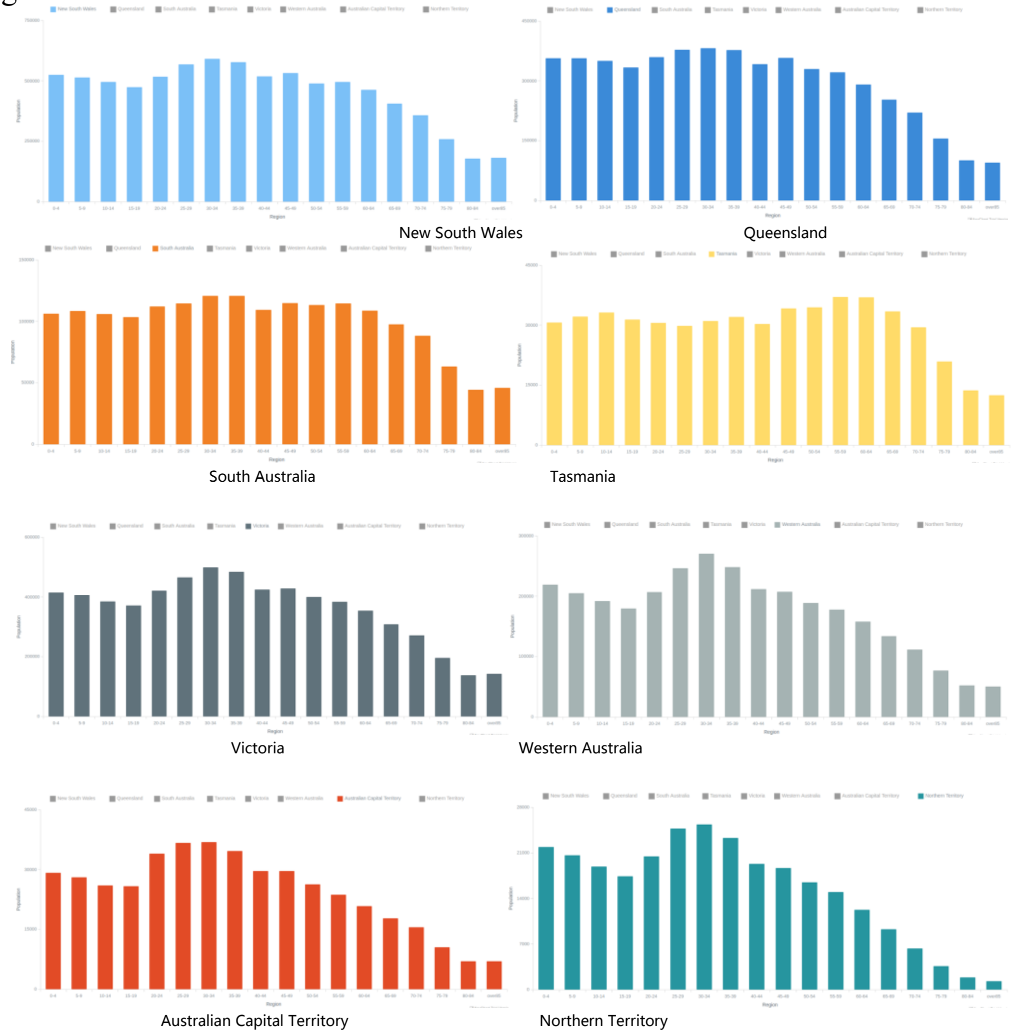


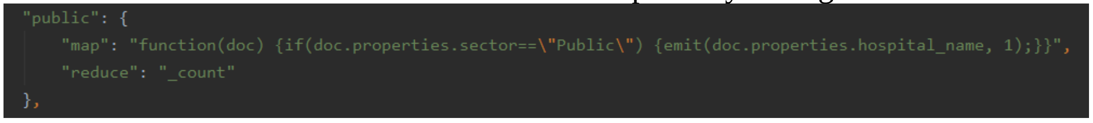
Fig 4.5 Age Distribution of Each Zone

## 4.2 Scenario2

This scenario is to analyze the relation between the number of tweets which mention covid-19 or coronavirus and the local hospital/income situation. Three databases are used in this scenario. One is to store the harvested tweets containing the keywords. One is to store the hospital data downloaded from AURIN, which includes the name, geolocation, size and category of each hospital. Another is to store the AURIN income data aggregated according to states, which records the income of the great city and the rest part of each state.

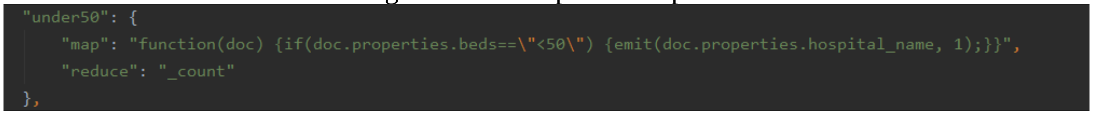
Similar to scenario1, the Stream API is used in this scenario, with the location parameter of the filter set to locations = [113.080648, -44.134922, 155.252163, -10.768672]. The track parameter of the filter is changed to 'covid-19' and 'coronavirus'. In order to be consistent with other scenarios, database partition is used in this scenario. The unique ID of each tweet is transformed to <region>:id before uploaded to the database. The partition is the same as that of scenario1. Each partitioned database is named with nsw(New South Wales), vic(Victoria), ws(Western Australia), act(Australian Capital Territory), qld(Queensland), nt(Northern Territory), sa(South Australia) and tas(Tasmania).

To reduce the work on MapReduce, for hospital and income databases, the data is also partitioned according to regions. The population dataset named 'MyHospitals Profile Data-Number of Beds' is downloaded from AURIN. This database records the name, state, size and category of each hospital. To be consistent with tweet data, the hospital data is partitioned by states. However, the database also includes detailed geolocation of each hospital for further analysis. MapReduce is used to count the number of hospitals according to size and category, such as the views of public hospitals (Fig 4.6) and hospitals with less than 50 beds (Fig 4.7). When someone sends a query to the database. These MapReduce views are efficient to return the actual number of hospitals by setting the reduce level to 0.



```
"public": {
    "map": "function(doc) {if(doc.properties.sector==\"Public\") {emit(doc.properties.hospital_name, 1);}}",
    "reduce": "_count"
},
```

Fig 4.6 View for public hospitals



```
"under50": {
    "map": "function(doc) {if(doc.properties.beds==\"<50\") {emit(doc.properties.hospital_name, 1);}}",
    "reduce": "_count"
},
```

Fig 4.7 View for hospital with less than 50 beds

As shown in fig 4.8, an Australia Map divided by states is designed to show the number of gained tweets in this zone. The top three states which mention covid-19 or coronavirus most frequently are New South Wales (41180), Victoria (36503) and Queensland (17803).

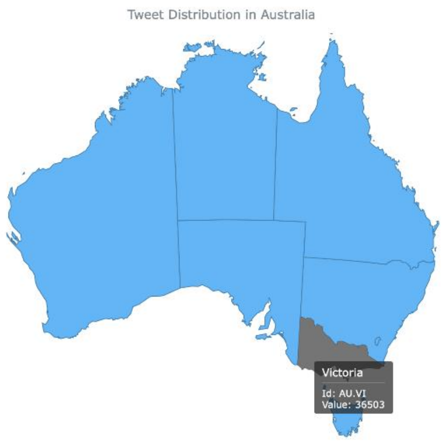


Fig 4.8 Tweets Distribution

Besides, a bar chart shows the ratio of tweets containing ‘covid-19’ and ‘coronavirus’ to hospitals in each zone (Fig 4.9). The higher ratio means that with the same number of hospitals, more Twitter users have mentioned these two keywords in their tweets. Shown in the bar chart, the Capital Territory has the highest ratio (393.75), followed by Northern Territory (199.00), Victoria (155.99), New South Wales (128.69), West Australia (87.90), South Australia (73.19), Queensland (68.47) and Tasmania (67.17).

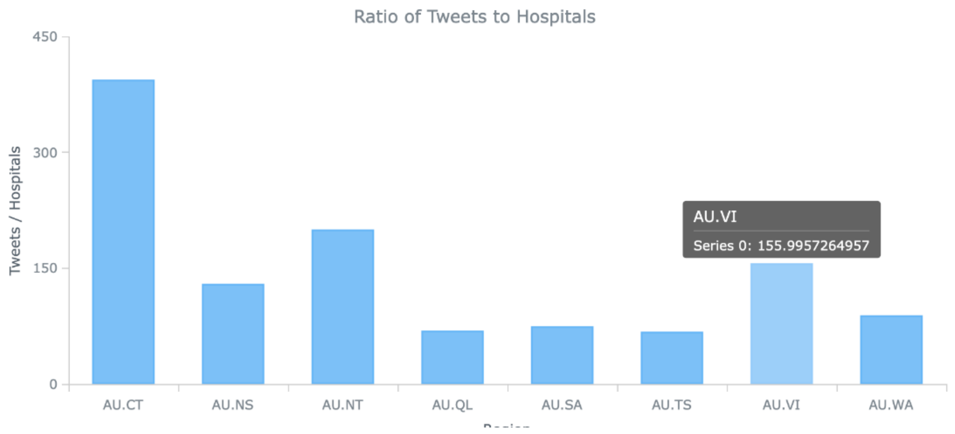


Fig 4.9 Tweets/Hospitals Ratio Chart

The following bar charts (Fig 4.10 and 4.11) show the size distribution of hospitals in each state according to the number of beds. The capability of hospital admission is important in this covid-19 emergency. Most hospitals in Australia have small sizes (less than 50 beds). It is obvious that New South Wales, Victoria and Queensland have the best medical resources in Australia regardless of the hospital size. This is consistent with the numbers of tweets in these three states.

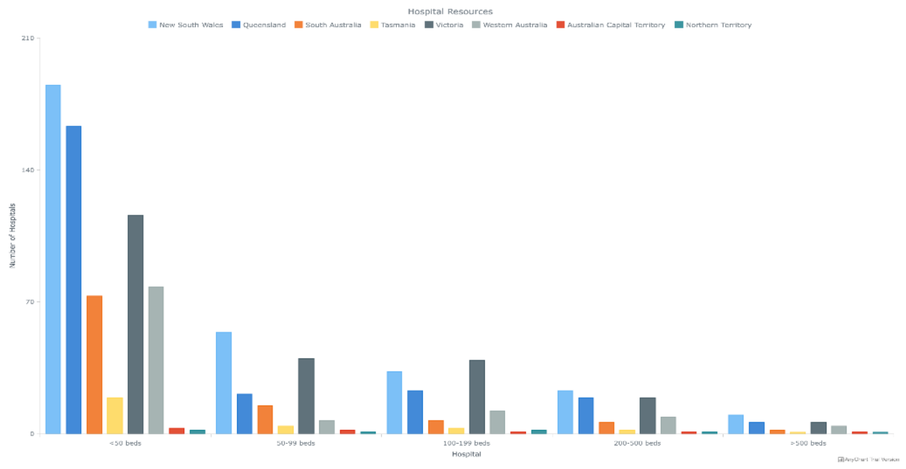
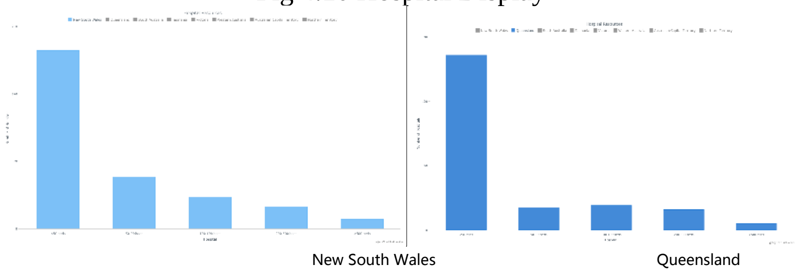


Fig 4.10 Hospital Display



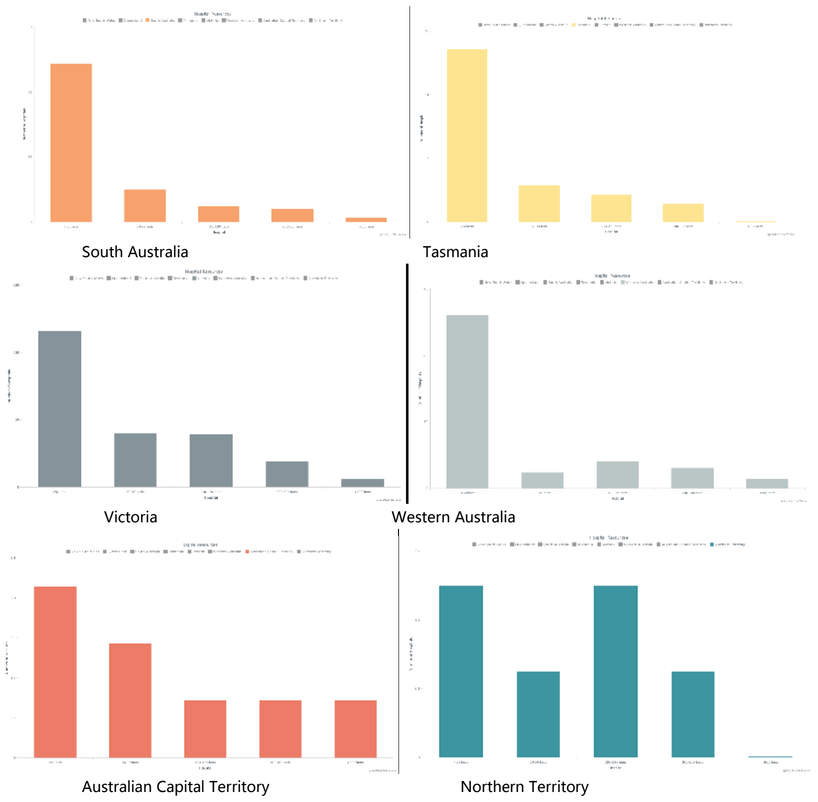


Fig 4.11 Hospital Display of Each State

The following bar chart (Fig 4.12) shows the distribution of public and private hospitals in each state. Except for Capital Territory, the percentage of public hospitals are obviously higher than that of private hospitals. However, the category of hospitals seems to have little relationship with the number of tweets which mention covid-19 and coronavirus.

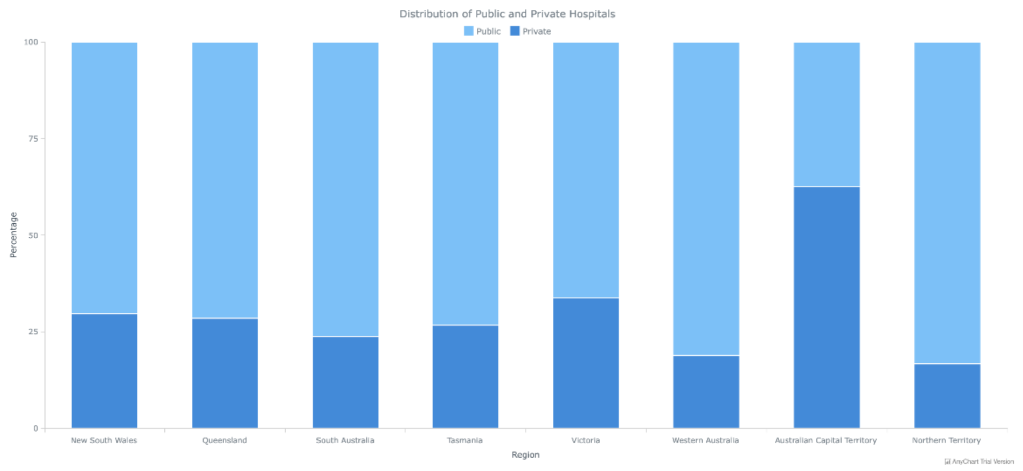


Fig 4.12 Distribution of Public/Private Hospitals

The following two bar charts (Fig 4.13 and 4.14)show the mean and median income across the whole Australia. Focusing on the greater city income, the top three states which have the highest mean income are Western Australia, New South Wales and Capital Territory. The top three states which have the highest median income are Capital Territory, Northern Territory and Western Australia.

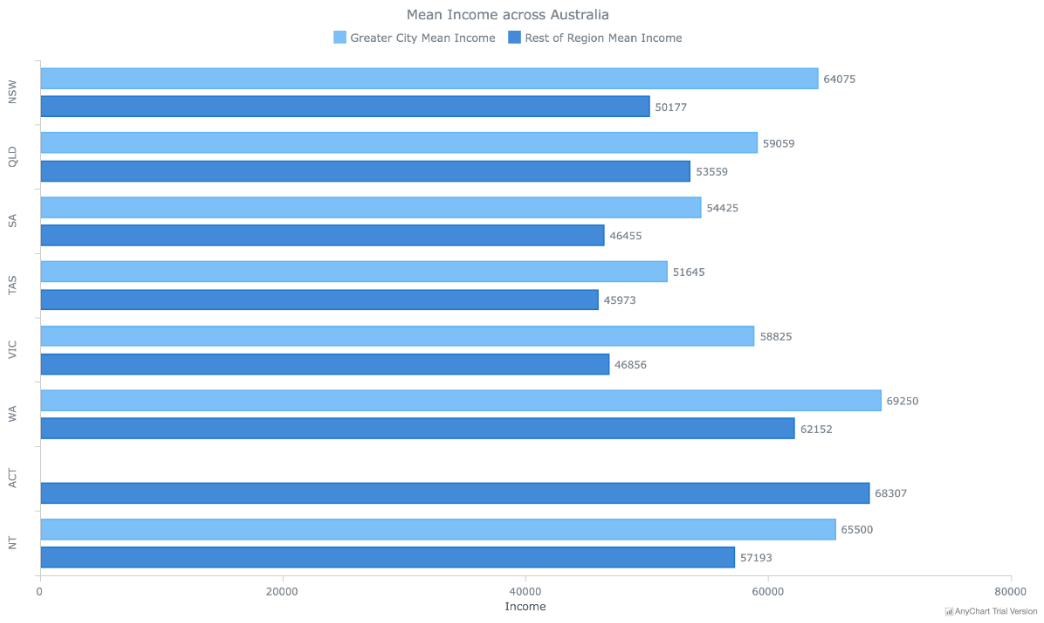


Fig 4.13 Mean Income across Australia

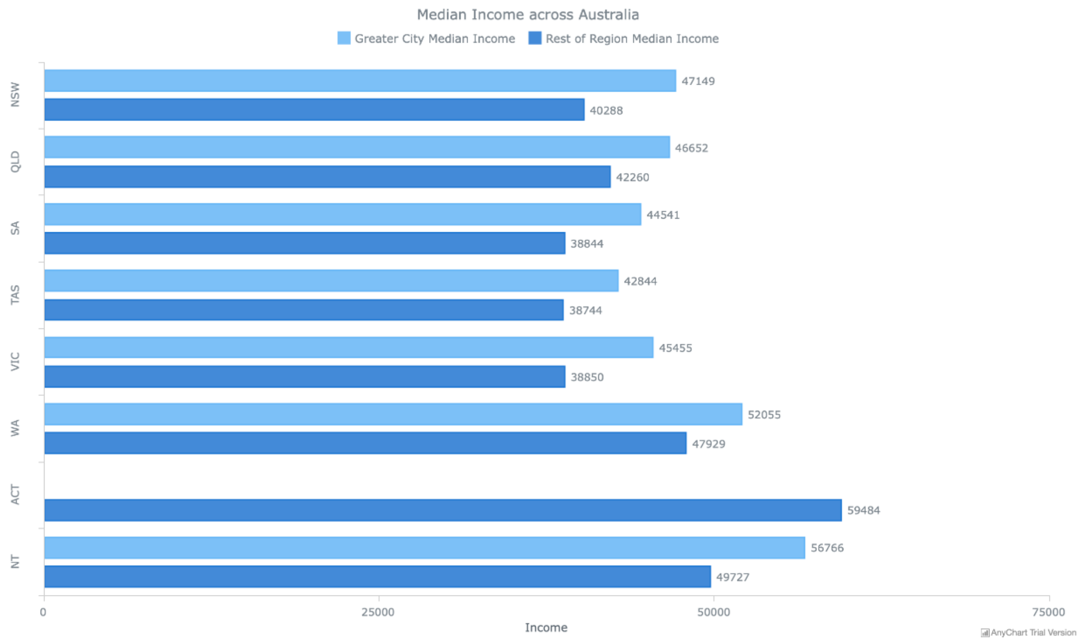


Fig 4.14 Median Income across Australia

The conclusion is that New South Wales, Victoria and Queensland have the largest number of tweets mentioning covid-19 or coronavirus and hospitals. More hospitals seem to mean a larger number of Twitter users who are concerned about the covid-19. Taking the ratio of tweets to hospitals into consideration, this relationship becomes less obvious. The Capital Territory comes to the first place. The possible reason is that this is the capital of Australia. More politicians living there and government institutions prefer to use Twitters to inform the latest news of the covid-19. Besides, due to the area of the Capital Territory, there are fewer hospitals compared to other states. Similarly, the Northern Territory comes to the second place because of the fewer hospitals. Apart from hospitals, the number of tweets mentioning covid-19 and coronavirus seems to have no obvious relationship with income. Based on the experience in real life, instead of hospitals and income, people are more concerned with the covid-19 in those areas where this virus becomes more serious, such as the number of the killed patients by this virus.

## 4.3 Scenario3

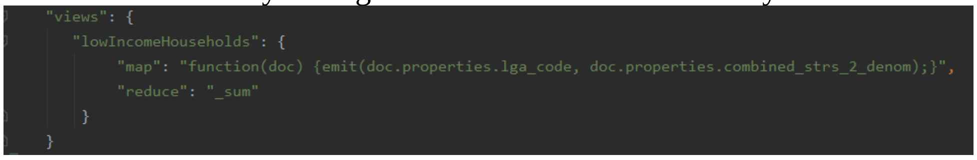
This scenario is to analyze the relation between the number of low-income households of Australia and the total numbers of tweets related to financial pressure, pressure and covid-19. There are three, in total, databases put into use to store the data of this scenario. The first one of the databases is about the population of each state of Australia. The second one is the number of low-income households database. And the final one is a tweets database.

The main purpose of this scenario is to find out whether there is a direct relationship between the financial pressure caused by the covid-19 epidemic and the number of low-income households in each state. Furthermore, it is analyzed whether the number of tweets containing keywords is related to the age structure of the population in each region. For example, whether the more people who are suitable for working age, such as 20-60 years old, the greater the recent financial pressure due to covid-19.

All of the tweets containing keywords 'financial pressure', 'pressure', 'covid-19' in Australia are stored in tweets database. In this project, the Twitter Stream API gains the tweets data from resources continuously. As long as the tweets harvester is working non-stop, the data in this database will be continuously updated. Some Tweet API may obtain data in a circular range. When using this kind of API to obtain data from multiple adjacent areas, due to the characteristics of the circular pattern, there must be overlap between the data. However, the selected Streaming API will only obtain data in a rectangular area with specific longitude and latitude of the upper left point to the lower right point. And it is the same as scenario 1, the location parameter was set by the Stream API to obtain the Australian data, which equals [113.080648, -44.134922, 155.252163, -10.768672]. The square area determined by the above parameters contains only Australian territory and part of the sea area, so the data obtained from this area is basically all from the Australian. And the parameter in filter is 'track' to limit the Stream API to obtain the tweets only containing the keywords above, 'financial pressure', 'pressure', 'covid-19'.

In addition, the obtained data are from different regions. In this example, the data resources were distinguished by state. To make full advantages of CouchDB partition function, the system will change the unique ID format to <region>:id before the tweets are stored in the database. In this scenario, due to handle large volume of the data, partitioning of data processing can be more efficient. According to the geographical characteristics of Australia, it is divided into eight parts by state to analyze the data.

As for low-income households databases, the data are partitioned by 8 states too, just like the tweets data. The dataset of low-income households is one of the LGA (Local Government Area) data from Aurin, which records the number of low-income households in each region according to the names of LGA and state. The data was compiled by PHIDU based on the ABS which is strongly reliable. In order to obtain the number of the households of each region, MapReduce technology is used to count the number of each region. And with the view as following figure, the number of low-income households of each region can be returned by setting the reduce level to 0 efficiently.



```
"views": {
  "lowIncomeHouseholds": {
      "map": "function(doc) {emit(doc.properties.lga_code, doc.properties.combined_strs_2_denom);}",
      "reduce": "_sum"
  }
}
```

Fig 4.15 View for low-income households

As shown in the fig 4.16, the front end has designed a map of Australia separated by state to show the number of tweets obtained in the zone. The top three states with the discussed keywords, 'financial pressure', 'pressure' and 'covid-19', are New South Wales (21549), Victoria (19238) and Queensland (9460). Of course, the amount of tweet data is also directly related to the population of each state.

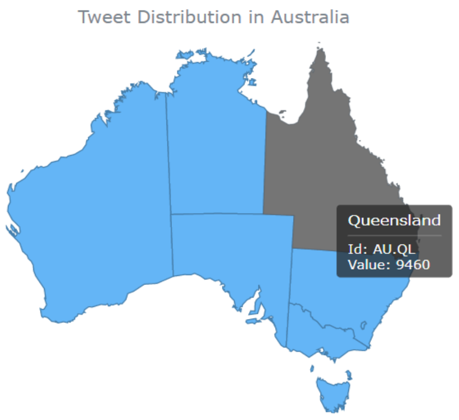


Fig 4.16 Tweets Distribution of Scenario3

In addition, the bar chart fig 4.17 below shows the ratio of tweets to low-income households of each region. A higher ratio means that more tweets are accompanied by fewer low-income households. In other words, low-income households in high-ratio states have recently been more affected about the economic situation due to the keywords 'financial pressure', 'pressure', 'covid-19'. So the ratio of the eight states from high to low is Australian Capital Territory, Northern Territory, Victoria, New South Wales, Western Australia, South Australia, Queensland and Tasmania.

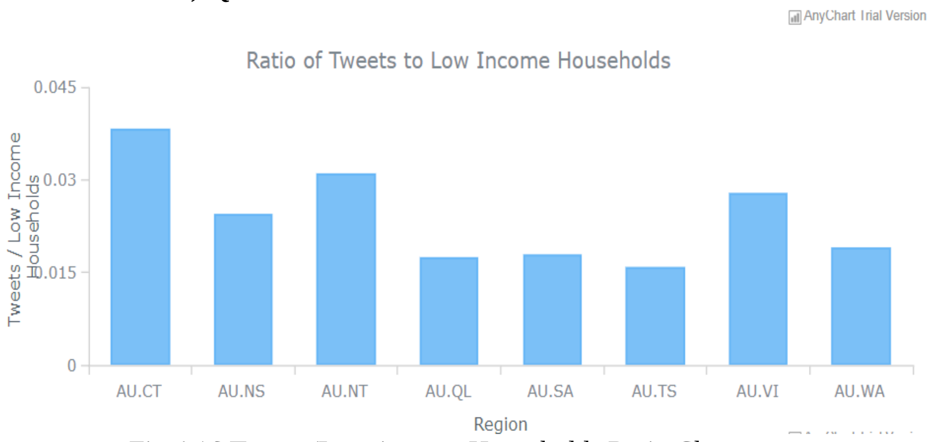


Fig 4.16 Tweets/Low-income Households Ratio Chart

The following bar chart Fig 4.17 illustrates the population of different age groups across Australia. Due to the legal working age limit set in Australia, the data of the 20-60 age groups will be used as the key reference object. Members of these age groups are all working age and are more susceptible to financial pressure because of covid-19.

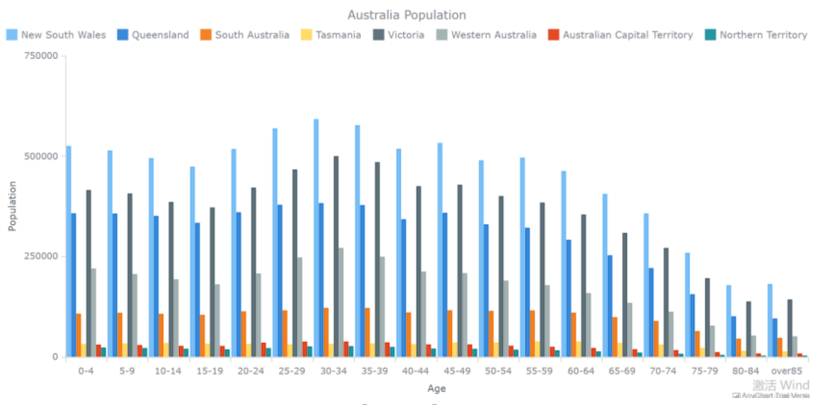


Fig 4.17 Population by Age Groups

Based on the data in the above chart, some speculative judgments can be made. The top4 states with the ratio of tweets to low-income households of each region are Australian Capital Territory, Northern Territory, Victoria and New South Wales. Among low-income households per unit, these states have collected more tweets data on the keywords ‘financial pressure’, ‘pressure’ and ‘covid-19’, which means the financial problem could occur more frequently in the low-income households of these states because that the epidemic has caused a large number of job opportunities to disappear, the economy has reversed, and more people are facing financial pressure. Furthermore, compared to other states, Australian Capital Territory and Victoria have a higher proportion of the working age groups (20-24, 25-29, …55-59). The income of low-income households themselves is unstable which could make the population ratio of these two states be one of the most crucial reason for the low-income households in these two states to face greater financial challenges in covid-19.

## 5 Challenges

In order to gain useful tweets data, choosing the proper Twitter API is important. At the beginning, we chose to use normal search API which seems to search tweets with specific keywords easily. However, as we have tried it in many different ways, we found many problems with this API. Firstly, it can only gain up to 100 tweets once, which are not enough for our analysis. Secondly, although it can limit the search zones, the search zone must be a circle which means that it is difficult to define the zones without overlaps between two adjacent search zones. The last problem is fatal and it lets us to choose stream API. If we use the same commands of search API, such as searching the same keywords tweets in same zones, it may return the same data. It means that there are bunch of repeated data and we have no way to determine whether the returned data are repeated.

Compared with normal search API, stream API has many advantages. Firstly, it can gain tweets in real-time. Hence, if we deploy the system in cloud, it will gain data continuously. It is easy to gain enough tweets for analysis in several days. Secondly, the tweets search zone can be a rectangle, which is better than circle since it can help avoid overlapping zones. Thirdly, because the data are gained in real-time, there will not be repeated tweets and the proportion of the number of tweets will be corresponded to the populations in different areas. Because of these advantages, we choose stream API to gain data which can make great advantage of the clouds.

The security of the system should also be considered. Since the database is the key of this system, it is necessary to prohibit others visiting the databases directly. Hence, appropriate security groups should be chosen. For the sake of communication between these nodes, the ports for CouchDB cluster are opened to specific IP addresses so that only these three nodes can communicate with others in port 4369 and 5984. Besides, the port 80 is exposed to all the IP addresses for users to visit this app.

Since the Melbourne Research Cloud has limitation to VMs, the proxy should be added to make sure the VMs can visit public IPs. Except the instance proxy, the docker containers also need a proxy or the tweets_harvesters will not be able to gain tweets.