Data Archive Manual

Local Storage for Data Archive

The system will keep up to 30-day data, and the outdated data will be removed by the LASS/AS team periodically. If the participating party is interested in archiving the dataset, it is suggested to run a local database on their own. The data will be available in a separated keyspace inside the Cassandra system.

The Cassandra Archive Table

The keyspace (means "database" in Cassandra) for data archive is called "**archive**" with 1 tables named "**all**". This table has the following attributes:

id	• The PRIMARY KEY of the table. • The format is YYYYMMDDhhmmssdevice_id, 14-bit date and time concatenated with device ID. • Data format: text • Not necessary for quering the data.
source	• The source of the data • Currently, there are 10 sources available, including **AirBox**, **AsusAirBox**, **MAPS**, **LASS**, **LASS4U**, **Indie, ProbeCube, WEBDUINO**, **EPA**, and **CWB**
device_id	• The device ID of the sensor. • Data format: **text** [Note] Since the variety of the self-design LASS PM2.5 sensors, the device ID could be various for LASS devices. Not all the devices has the ID in unified format.
day	• The 2-bit day of the measurement data recorded. • Data format: **int**
month	• The 2-bit month of the measurement data recorded. • Data format: **int**
year	• The 4-bit year of the measurement data recorded. • Data format: **int**
data	The sensing data in **JSON** format. (a JSON string)

Data Source Introduction

AirBox	The AirBox Project in Taipei City. (Sensors made by EDIMAX)
AsusAirBox	The AirBox Project. (Sensors made by ASUS CLOUD)
MAPS	The sensor designed by IIS-NRL, Academia Sinica.
LASS	• The PM2.5 sensors from the LASS maker community. • URL: https://www.facebook.com/groups/1607718702812067/
LASS4U	• The LASS community's LASS4U Project. • Project URL: http://lass-net.org/2016/05/lass4u/
Indie	• The g0v Indie Project • Project URL: http://beta.hackfoldr.org/g0vairmap/g0v--8MQ03kk9wwD
ProbeCube	• The g0v ProbeCube Project • Project URL: https://github.com/Lafudoci/ProbeCube
WEBDUINO	• 高雄市校園空氣品質監測 • Project URL: http://marty5499.github.io/air-schools/
EPA	• The sensing station deployed by **Environmental Protection Administration, Taiwan**. • The data is **hourly** provided.
CWB	• The weather data from the **Central Weather Bureau, Taiwan**. • The data is **hourly** provided.

Data Query

- The account to communicate with Cassandra is provided in the sample python program (in **config.py** file). Please refer to the file for the account information, thank you!

- To login to Cassandra using the `cqlsh` (the Cassandra Query Language Shell), use the command: `$ cqlsh <server IP> -u <Cassandra account>` , then enter the password.

To query the Cassandra Archive Table, please use the following command:

```
SELECT data FROM archive.all WHERE source='' and year=<int> and month=<int> and day=<int> ALLOW FILTERING;
```

- Due to Cassandra mechanism, we need to provide the **source name** (AirBox, LASS, etc.) to query the data. And please **DO NOT** remove the "**ALLOW FILTERING**" option in the command.
- Example:

```
# Querying the data from source "AirBox" for 2017/02/13
SELECT data FROM archive.all WHERE source='AirBox' and year=2017 and month=02 and day=13 ALLOW FILTERING;
```

Sample Data

The followings are the sample value of the "data" column:

- The meaning of the key value could be found on the LASS HACKPAD: https://lass.hackpad.com/LASS-Data-specification-1dYpwINtH8R
- Please note that the content of the data could be various among different data sources.

1. AirBox


```
{"gps_num": 9, "app": "AirBox", "s_d1": 0, "fmt_opt": 1, "device": "kdps", "s_d2": 0, "s_d0": 11, "gps_alt": 2, "s_h0": 69, "loc": {"type": "Point", "coordinates": [25.062, 121.451]}, "SiteName": "\u53f0\u5317\u5e02\u95dc\u6e21\u570b\u5c0f", "gps_fix": 1, "ver_app": 1.0, "s_t0": 27.62, "date": "2017-01-01", "tick": 1483237459, "device_id": "28C2DDDD45AA", "s_1": 100, "s_0": 102, "s_3": 0, "s_2": 1, "ver_format": 3, "time": "02:24:19"}
```


2. AsusAirBox


```
{"ver_format":3, "fmt_opt":1, "temperature":18.0, "power":100.0, "SiteName":"bapspm", "timestamp":"2017-02-01T15:58:35Z", "app":"AsusAirBox_COPY", "humidity":88.0, "time":"15:58:35", "date":"2017-02-01", "pm25":6.0, "device_id":"Asus_28C2DDDD4579", "loc":{"type":"Point","coordinates":[25.035,121.571]}}
```


3. MAPS


```
{"ver_format":3, "FAKE_GPS":1, "app":"MAPS", "ver_app":"5.1.1", "device_id":"04000413", "tick":0, "device":"LinkIt_Smart_7688_Duo", "s_d0":21, "s_d1":23, "s_t4":15.6, "s_h4":83.4, "gps_fix":1, "gps_num":15, "date":"2017-02-01", "time":"13:11:14", "SiteName":"MAPS0008", "loc":{"type":"Point","coordinates":[25.019722,121.548889]}}
```


4. LASS


```
{"ver_format":3, "Fake_GPS":1, "app":"PM25", "ver_app":"0.7.13", "device_id":"WF_1735326", "tick":-1125956175, "date":"2017-02-01", "time":"23:00:35", "device":"WF8266R", "s_d0":11.00, "s_t0":15.1, "s_h0":94.5, "s_d1":11.00, "gps_fix":1, "gps_num":9, "gps_alt":2, "loc":{"type":"Point","coordinates":[24.413394,121.459744]}}
```


5. LASS4U


```
{"ver_format":3, "FAKE_GPS":1, "app":"LASS4U", "ver_app":"beta", "device_id":"FT2_0036", "date":"2017-02-01", "time":"15:12:54", "device":"Ameba", "s_t2":19.89, "s_h2":70.54, "s_d0":14.00, "s_g8":418.00, "s_d2":11.00, "s_d1":14.00, "loc":{"type":"Point","coordinates":[24.092882,120.887993]}}
```


6. Indie


```
{"ver_format":3, "fmt_opt":1, "app":"Indie_COPY", "PM2_5":15.6, "Temperature":20.0, "s_0":6038, "SiteName":"Home Sense", "app":"Indie_COPY", "Humidity":75.0, "time":"16:29:55", "date":"2017-01-17", "device_id":"Indie_206063", "loc":{"type":"Point","coordinates":[25.069879,121.632872]}}
```


7. ProbeCube


```
{"ver_format":3, "fmt_opt":1, "app":"ProbeCube_COPY", "PM2_5":58, "Temperature":19.6, "s_0":370652, "SiteName":"lemon", "app":"ProbeCube_COPY", "Humidity":0.2, "time":"16:14:42", "date":"2017-02-19", "device_id":"PC_101039", "loc":{"type":"Point","coordinates":[24.768058,120.967176]}}
```


8. WEBDUINO

```
# Currently, we do not receive data from any WEBDUINO sensor.
```

The Sample Python Program

Along with this manual, we provided a sample Python program which can retrieve the data from the Cassandra Archive Table.

The program includes the following files:

1. **config.py**: the parameter settings for the program including the account information for accessing Cassandra cluster (should be place in the same directory with data_archive.py)
2. **data_archive.py**: the program that needs to be routinely executed

Configuration

- It is recommended to use the address "**127.0.0.1**" to connect to Cassandra system.
- **To dump one month's data from Cassandra system could trigger tremendous server loading. This situation might raise up the probability for causing the "Operation Timeout Exception" when trying to retrieve the data. Therefore, we suggest to dump the data weekly or dump it in few days.**