Python Programming Challenge

For more information:

Stathis Gkotsis

SW Engineering Director

Commsquare

Mobile: +30 695 648 91 88

Email: stathis.gkotsis@commsquare.com

Website: www.commsquare.com

Vasilis Vagenas

Principal SW Engineer

Commsquare

Mobile: +30 210 63 97 250

Email: vasilis.vagenas@commsquare.com

Website: www.commsquare.com

commsquare

Contents

1 Development environment

Any Linux system with python 2.x or python 3.x, MariaDB or MySQL, Django (optional). The recommended stack is: Python 3.6+, MariaDB and Django 2.2+.

2 Problem description

We are building a system for calculating mobile subscriber and network statistics from raw data based on open source software components. The raw data are produced by a 3rd party system and refer to the user activity in 5-minute time periods. They are produced in the form of CSV text files and contain the following fields:

Field	Description
interval_start_timestamp	Start time of the interval (Unix timestamp in milliseconds)
interval_end_timestamp	End time of the interval (Unix timestamp in milliseconds)
msisdn	MSISDN of mobile user (64bit integer). This is a unique identifier for the mobile user.
bytes_uplink	Number of uplink bytes (64bit integer)
bytes_downlink	Number of downlink bytes (64bit integer)
service_id	Identifier of traffic class (32bit integer). Traffic class can be facebook, youtube, instagram etc.
cell_id	Cell ID of mobile user (64bit integer)

The CSV files are produced every 5 minutes, i.e. at 00:00, 00:05, 00:10 etc. The content of each file refers to the mobile users' activity in the last 5-minute interval, so a file produced at 00:30 will refer to the activity recorded between 00:25 and 00:30. For each interval there can be many files produced and the filename format of each file is: ipflow_data.ts-*ts.id*.txt, where *ts* is the unix timestamp (in milliseconds) of the end of the interval and *id* is an auto incrementing id for the multiple files of the same interval.

We can assume that the generation of CSV files starts at the end of the 5-minute interval and finishes 1 minute later at the latest, so 1 minute after the end of the interval.

The purpose of the application is to calculate user and network KPIs (Key Performance Indicators) for 5-minute and 1-hour intervals and store them in the database. The KPIs to be calculated are the following:

- KPI1: Top 3 services by traffic volume: the top 10 services (as identified by service_id) which generated the largest traffic volume in terms of bytes (downlink_bytes + uplink_bytes) for the interval.
- KPI2: Top 3 cells by number of unique users: the top 10 cells (as identified by cell_id) which served the highest number of unique users (as identified by msisdn) for the interval.

The KPIs above should be calculated for all 5-minute intervals within the day, but also for all 1-hour intervals of the day. So, for each 5-minute KPI there should be calculations for the intervals: 00:00 – 00:05, 00:05 – 00:10, 00:10-00:15, etc. For each 1-hour KPI, this should be done for: 00:00 – 01:00, 01:00 – 02:00 etc.

The result should be stored in one database table for each KPI. For the 1st KPI, the table should contain the following fields:

Field	Description
interval_start_timestamp	Start time of the interval (Unix timestamp in milliseconds)
interval_end_timestamp	End time of the interval (Unix timestamp in milliseconds)
service_id	Identifier of traffic class (integer). Traffic class can be facebook, youtube, instagram etc.
total_bytes	Total number of bytes for the service
interval	5-minute or 1-hour

For the 2nd KPI, the table should contain:

Field	Description
interval_start_timestamp	Start time of the interval (Unix timestamp in milliseconds)
interval_end_timestamp	End time of the interval (Unix timestamp in milliseconds)
cell_id	Cell ID of mobile user
number_of_unique_users	Number of unique users for the cell
interval	5-minute or 1-hour

3 Deliverables

a. A high-level design of the system which will perform the KPI calculations based on the input raw files and store the result in database. The design should include the high-level blocks of the system and how these interact with each other. The design diagram can be done in a visualization tool or even sketched on paper and then scanned.
b. A simple implementation (source code) written in Python for the calculation and storage of the KPIs. The program should take several 5-minute raw files as the input and store the calculated KPIs in database.
c. Optionally: The design (URLs and response format) of a RESTful API which will provide the KPI data (e.g. for KPIs visualization in a web application). It is preferable that the response body is in JSON.
d. Optionally: A simple implementation of the above API in Python. This can be done in Django or another framework. No HTTP server software (e.g. Apache) is required, since the Django runserver can be used.

4 Sample data

Sample files are provided as input to deliverable 3b. For the interval ending at: 01/03/2017 09:05AM CET (refers to 09:00 – 09:05AM), we are given two 5-minute sample raw files: ipflow_data.ts-1488355500000.1.txt and ipflow_data.ts-1488355500000.2.txt (in attachment). The expected calculated KPIs for this 5-minute interval are in files: KPI1.5min.ts-1488355500000.txt and KPI2.5min.ts- 1488355500000.txt (in attachment).