Data Aggregator Requirements v1.2

The intent of this code is to provide a flexible basis for grouping GPS Client data from the raw client/server database into a more usable format for analysis work. This will be accomplished by

1. Defining the boundaries of the area of interest to be analyzed.
2. Computing the elements of the grid that will form the individual cells which will form the "containers" in which the data elements will be tallied
3. Filtering and iterating over the GPS client data and assigning to those containers based on location
4. Writing to a database table.

This can be implemented via an interpreted language such as PERL or Python. All modules used should be available on both Linux and Windows implementations of the language.

As the application will need to be passed a large amount of parameters at runtime, it should have the ability to connect to the reports database (specified in OUTPUTS) and have it invoked with a report number. It will then re-run the specified report (creating a unique new report number in the process).

The application should receive a configuration file as a parameter which at a minimum has the database connection information. In the absence of a report number to model after passed to the application, the application will also read this configuration file for the same parameters as would be specified in a database job number.

INPUT

1. DATASET
    a. This is the primary SQL table from the GPS Client/Server application. In initial development, the credentials and database connect information can be placed as easily identifiable variables shortly after the code headers
    b. For the purpose of this exercise the table is not altered from last state and said documentation can be referenced for format and usage
    c. For the purpose of this exercise it will be assumed that Postgres is the database in use
2. DATE RANGE
    a. Specified when application is invoked. Application will convert at runtime from local time to UTC time to match the GPS time.
    b. In a from-to format in local time (i.e. from 2014-06-01 00:00 to 2014-06-02 23:59), OR
    c. Last X hours, in which the current time serves as the end time and the start time is the current time minus X hours; for example 72 would give the last 72 hours, ending at the current time

3. BOUNDARY
    a. Latitude and longitude for all four corners, assumed to form a four-sided box which defines the outer coordinates of the area to evaluate out of the data set.
    b. This will be a regular rectangle with lines along north south and east west parallels.
    c. All adjacent points must share one coordinate value with its neighbor
    d. If points in the dataset are outside of this boundary box, they will be ignored and not calculated
    e. If not specified, the applications should use all points in the DATASET (with the caution that some error checking needs to be included to avoid GPS errors causing unreasonably large output grids)
4. CLIENT DEVICES
    a. List of the client devices, comma separated, to perform the analysis on. If specified, all others in the dataset will be ignored and only these devices will be tallied for the output
    b. If not specified, all devices in the dataset should be evaluated
5. GRID SIZE
    a. This is a required input and should be specified as S where S is in meters and defines the length of the sides of the grid
6. MOTIONLESS DATA MAX
    a. The maximum number of seconds of motionless data to include for a client
    b. If a client does not move in this number of seconds, its data (EVENT_TYPE 0, EVENT_TYPE 1 and EVENT_TYPE 4) is disregarded until it goes into motion again (this is to prevent an idling machine from skewing statistics)
    c. If not specified, defaults to evaluating all data without regard to if the client was in motion or stopped
7. PRUNING INCLUSION
    a. Should be a boolean value which indicates whether or not a client's location data (EVENT_TYPE 0) should be included after it has been pruned (EVENT_TYPE 9) by the server.
    b. If set to false, once a prune record (EVENT_TYPE 9) for a clientis discovered, all EVENT_TYPE 0 records should be discarded until an EVENT_TYPE 7 for the client is registered.
    c. If not specified, defaults to TRUE which means all data is evaluated, even following a pruning

OUTPUT

The output of this application is updates in two database tables:

1. REPORT ID DATABASE
    a. Establishes a unique number for the report with which all output will be associated

b. In the row identified by the unique number is all of the data (INPUT 2 through INPUT 7) that was used to build the report such that a report of the same type could be executed again with all of the same information
c. Includes time and date of execution

2. REPORT DATA
    a. Each row of this database contains the unique REPORT ID referenced in 1.a.
    b. Each row of this database must also have a unique ID number
    c. Each row of this database correlates directly to a grid generated from the boundary divided by a box of S size (see INPUT 3 and 5)
    d. Each row contains data as defined with data types as appropriate:

REPORT_ID	The report ID used to build this data as referenced in OUTPUT 1.a.
BOX_SIZE	The size, in meters, of the grid section this row represents. This is directly from INPUT 5.a.
NW_CORNER	The latitude and longitude of the northwest corner of the grid section that this row represents.
LOCATION_TOTAL	The total number of location events (EVENT_TYPE 0 from GPSDATA database) which occurred in this grid section
UNICAST_TOTAL	As in LOCATION_TOTAL, except a tally of EVENT_TYPE 1
BROADCAST_TOTAL	As in LOCATION_TOTAL, except a tally of EVENT_TYPE 2
MULTICAST_TOTAL	As in LOCATION_TOTAL, except a tally of EVENT_TYPE 3
ACK_TOTAL	As in LOCATION_TOTAL, except a tally of EVENT_TYPE 4

TABLE 1: REPORT DATA ROW DEFINITION

Essentially, the end result is a tally of all of the events of different types which occurred in multiple user-defined boundary boxes over the data set. For example, if the size S is set to 10, the area defined by the boundary box (A, B, C, D in the diagram following) then the area within that boundary box will be divided up into numerous 10m squares. The GPS coordinates of the

data set will be evaluated for which square it falls into, and then those events in the square will be tallied and eventually written as a row to the database when all data has been evaluated.

FIGURE 1 below shows how this would work given an evaluation plot with BOUNDARY A, B, C, D and a BOX_SIZE of S.

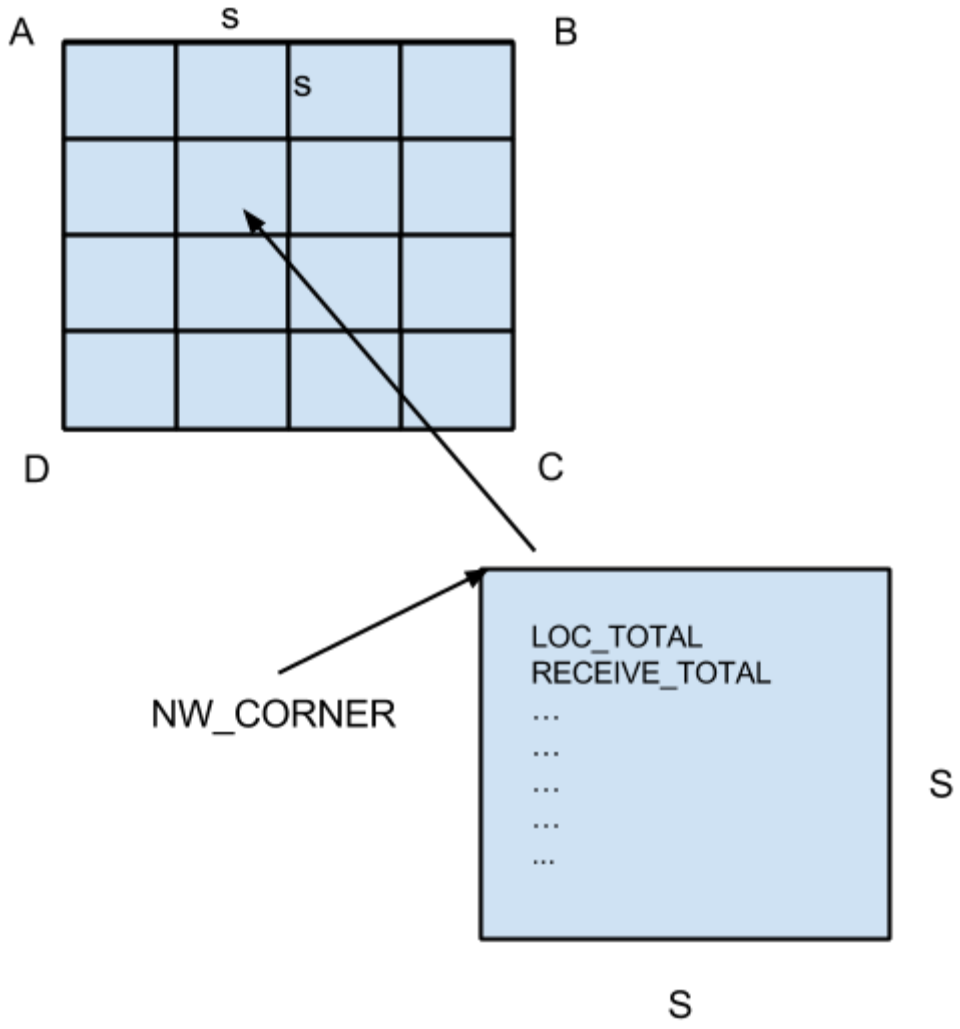

FIGURE 1: GEOSPATIAL DIAGRAM OF DB REPRESENTATION

PROCESS

What follows is a suggested process and may not be particularly efficient. Given the above requirements, anything which improves it would be welcome.

1. The data set must be derived from the raw data in the GPS database by executing a query against the database using the parameters specified in the input section. The resultant data will be the data to evaluate. It is recommended that this be ordered by

client_timestamp, uid in order to make assessing whether or not the clients have moved simpler (in the event MOTIONLESS DATA MAX is set).

2. Appropriate entry should be made in the REPORT ID table to indicate what report is being generate and assign a unique ID
3. If a BOUNDARY is not specified, the data must be examined to find the four corners (A, B, C, D) of the dataset. Otherwise the BOUNDARY should be used.
    a. NOTE, caution should be taken to validate that there are not extreme outliers in this dataset. Some checking needs to be done. For example, it is possible that a GPS may have a poor fix and report data hundreds of kilometers away; these need to be identified and discarded otherwise the processing could be huge.
4. The application should break the area represented by BOUNDARY up into boxes of size S on a side by an efficient algorithm.
    a. If the BOUNDARY is not evenly divisible by size S, the easternmost and southernmost boxes should expand in size to contain the remainder
    b. The application should now have a suitable list of coordinates which form the boundaries of the boxes of height and width S within BOUNDARY
5. A data structure should be created with each element representing a box and the associated elements that must be tracked for each box.
6. The data set should be iterated over and for each EVENT_TYPE, the geolocation should be assessed to determine in which box the event occurred. The appropriate _TOTAL should be incremented by one for the grid box in which the event fell.
    a. For example, if there was an EVENT_TYPE 1 at 55.67232, 12.2342333, UNICAST_TOTAL for the box that contains 55.67232, 12.2342333 should be incremented by 1.
    b. Note that if a MOTIONLESS DATA MAX value is specified, each entry should be checked to see if it differs in geolocation from the previous entry for that client. If it does not, a counter should be incremented with the number of seconds that client has remained motionless. Once that counter reaches MOTIONLESS DATA MAX, the client should be disregarded until it moves, at which point the counter should be reset. The point of this is to provide the ability to eliminate skewing data, for example if a client sits in one area of poor coverage or excellent coverage for an hour while being repaired, this provides a mechanism to remove that data.
7. If the PRUNING INCLUSION is set to false, when an EVENT_TYPE 9 occurs, all subsequent EVENT_TYPE 0 from the client should be ignored until an EVENT_TYPE 7 occurs.
8. When the data has been iterated over and all boxes are complete, the tally for each box should be written as a row to the database table as defined in table 1 in this document.