PURPOSE: To pre-generate movement data into a format which allows for faster retrievals based on multiple primary keys.

Columns:

1. MOVEMENT\_ID: Unique ID of the movement
   1. Not used as primary key. Purpose created for grouping of latlon pairs for polyline. However final result did not implement this. Next team can consider use for this or remove column.
   2. INT
2. UNIX\_START: Start time of the movement data
   1. PRIMARY KEY
   2. Start time of the frame the movement data is for
   3. INT(10)
3. UNIX\_END: End time of the movement data
   1. PRIMARY KEY
   2. End time of the frame the movement data is for
   3. INT(10)
4. START\_LOCATION: Starting point of the movement data
   1. PRIMARY KEY
   2. The location which the movement data row started from
   3. VARCHAR
5. DEST\_LOCATION: Ending point of the movement data
   1. PRIMARY KEY
   2. The location which the movement data row ended (destination). WILL NOT be the same as START\_LOCATION, otherwise it will not be a movement
   3. VARCHAR
6. COUNT: Movement count
   1. Number of subscribers recorded / detected making this same movement at same time, from same place to same destination.
   2. Will not be 0, which will mean no movement. If no movement will not be recorded.
   3. INT

# **Creating Movement data**

1. Obtain S\_ID, location from **Master table** where time = 0900hrs

a. SELECT S\_ID, LOCATION FROM MASTER WHERE UNIX\_TIME = UNIX\_TIMESTAMP(‘2015-05-20 09:00:00’);

2. For each S\_ID we now have to go through **Master table** and look for them at 1000hrs. Select only entries with movement. Get it’s new location.

a. SELECT LOCATION FROM MASTER WHER UNIX\_TIME = UNIX\_TIMESTAMP(‘2015-05-20 10:00:00’) AND S\_ID = s\_id AND LOCATION != 0900\_location

3. Now we have detected movement with both start and end points, proceed to insert data into **Movement table.**

# **How to create a complete set of ‘OLAP cube’?**

The cube will be very large in size.

To speed up initial process we generate data for per hour pairs. Point A will contain groups of S\_ID from 0000hrs to 2300hrs (Step 1).

Then for each groups we compare them with Point B, 0100hrs to 0000hrs (D+1) (Step 2), then create new movement data (Step 3). However if we are to generate the complete set of OLAP cube with time pairs of variable granularity(minutes) it will be very time, disk space consuming, and unneeded data generation.

Perhaps we will only generate data on demand based on user selection on UI. But user may end up waiting for a while for data to be created.

# **Problem with this approach**

Point markers and lines will only show traffic count as subscriber data is not included in the movement table. We can implement separate function to obtain S\_ID from **Master table**. But since we may only be interested to get traffic data having subscriber ID may be unneeded.

## **Possible issue: Losing of cluster size of Point A (solution)**

Total amount of subscribers originating from Point A may be lost due to losing the data during transformation. However it can still be calculated by summing the **count** number of each destination within the same time pairs, and point A.