# Rock 7 Database processing task

The task is to process a JSON file containing details of a number of boats in a yacht race and the positions that have been provided at intervals during the race.

## MySQL database

Boat –containing details of the name and id of each boat.

Position – containing details of each of the position reports along with a generated field providing the hour of the report.

Visible – generated database to contain the list of and total number of boats visible for each boat and day.

## Processing

1. Read the JSON file and put the information into the Boat and Position database tables.
2. Use the Position database table to find all of the possible visible pairs of boats and add this information to the Visible table.
3. Query the Visible table to find the resulting values required.

## Assumptions made

That the original JSON file is ordered by boat and time of position report.

The maximum visible distance is 12nm.

The haversine formula for calculating the distance between 2 points is slow to execute. I have used a quicker Pythagoras equation to find the distance between 2 x,y points even though this will not be accurate.

I have made a quick assumption that boats that are more than 0.2 degrees of latitude or longitude apart will not be visible to each other. I realise that this is not quite true for longitudinal distances.

I have assumed that there was one report per hour for each boat and used only one position report per hour.

Where the position reports are not made every hour, I have assumed that the boat will stay in the same position until the next report. This is obviously not true, but should be sufficient for the average value calculation required.

Where a whole day of position reports is missing for a boat, then this boat is assumed to be not visible to anyone on that day.

## Problems encountered

I noticed that some of the time the position reports are hourly, this is not always the case. Some are reported multiple times an hour and some only a few times per day. My original checks where only matching boats that had a report at the same hour.

I added extra rows to the Positions table to fill in the hours that were not reported with the assumption that the boat will not move until the next actual report.

I removed extra rows where any boat position was reported more than once per hour.

## Improvements that could be made

It would be better to have a database interface class so that all of the SQL database processing could be restricted to be in one place.

I think that there must be a mathematical formula that would be better for finding the distances between 2 moving objects. This might make the processing time quicker.

The processing time for this program is quite long. Depending on the constraints for accuracy against time, some loss of accuracy could be made for an improvement in time taken.

## Database Schema

CREATE TABLE `boat`  
( `idboat` int(11) NOT NULL,  
 `name` varchar(45) DEFAULT NULL,  
 PRIMARY KEY (`idboat`) )

CREATE TABLE `position`   
( `idposition` int(11) NOT NULL,  
 `time` datetime DEFAULT NULL,  
 `latitude` float(20,6) DEFAULT NULL,  
 `longitude` float(20,6) DEFAULT NULL,  
 `cog` float(20,6) DEFAULT NULL,  
 `speed` float(20,6) DEFAULT NULL,  
 `boatid` int(11) NOT NULL,  
 `timemillis` bigint(20) DEFAULT NULL,  
 `hour` bigint(20) DEFAULT NULL )

CREATE TABLE `visible`   
( `idboat` int(11) NOT NULL,  
 `visibleboat` int(11) NOT NULL,  
 `visibleday` date DEFAULT NULL,  
 `dailycount` int(11) DEFAULT NULL )

## Results

|  |  |
| --- | --- |
| Average number of other boats visible to each boat | Date |
| 185.0000 | 2017-11-19 |
| 71.9351 | 2017-11-20 |
| 12.5341 | 2017-11-21 |
| 4.2805 | 2017-11-22 |
| 2.8299 | 2017-11-23 |
| 2.0544 | 2017-11-24 |
| 2.2623 | 2017-11-25 |
| 2.4272 | 2017-11-26 |
| 2.6111 | 2017-11-27 |
| 2.4228 | 2017-11-28 |
| 2.4000 | 2017-11-29 |
| 2.2742 | 2017-11-30 |
| 1.9286 | 2017-12-01 |
| 1.7474 | 2017-12-02 |
| 1.8476 | 2017-12-03 |
| 2.1441 | 2017-12-04 |
| 2.0357 | 2017-12-05 |
| 1.8679 | 2017-12-06 |
| 2.4444 | 2017-12-07 |
| 8.2124 | 2017-12-08 |
| 13.3714 | 2017-12-09 |
| 25.7500 | 2017-12-10 |
| 24.0923 | 2017-12-11 |
| 22.1304 | 2017-12-12 |
| 14.4000 | 2017-12-13 |
| 8.9565 | 2017-12-14 |
| 17.8947 | 2017-12-15 |
| 11.3333 | 2017-12-16 |
| 7.0000 | 2017-12-17 |
| 2.5000 | 2017-12-18 |
| 3.0000 | 2017-12-19 |