**A picture containing text

Description automatically generated**

**A02 – REPORT ON ADDITIONAL TRANSPORTATION DATASET.**

Use of public transport has lots of benefit from reducing toxic emission from vehicle to not having the burden of paying parking charge & extra tax for owning a vehicle.

DATASET:

The additional transport dataset I have chosen is **the Dublinbikes** **dataset**. Dublinbikes is a subscription-based and short-term hire basis hybrid/electric bike-sharing system. It has many bikes stand from where one can hire a cycle and explore the city. It provides a convenient option for commuting a short distance in the city. The dataset is located at [https://data.smartdublin.ie/dataset/dublinbikes-api#](https://data.smartdublin.ie/dataset/dublinbikes-api). It has bike usage data for **12 quarters** from the 3rd quarter (2018/07/01-2018/10/01) to the 2nd quarter of 2021(2021/04/01 – 2021/07/01). The dataset is available in **CSV format** and is **one file per quarter**. The total size 12 of CSV files is **3.37GB**. The real-time usage of bikes can also be accessed through JCDecaux API.

There are roughly 115 bikes stand which has a fixed number of bikes slot for parking the bikes. When someone takes a bike, the total available count of bikes reduces by one and the available bike stand count increases by one. The CSV file has two types of data, static and dynamic. Each row of the dataset has the following fields:

* STATION ID
* TIME
* LAST UPDATED
* NAME
* BIKE STANDS
* AVAILABLE BIKE STANDS
* AVAILABLE BIKES
* STATUS
* ADDRESS
* LATITUDE
* LONGITUDE

*Station Id, Name, Bike stand, Address, Latitude, and longitude* are **static field** which never changes with time. *Time, Last updated Available bikes, Available bike stand, and status* are **dynamic field**.

Station Id: An integer assigned to each bike stand.

Example: 2

Time: String representing timestamp in format <%Y-%m-%d %H: %M:%S>

Example: 01/04/2021 00:25:02

Last Updated: String representing timestamp in format <%Y-%m-%d %H: %M:%S> when the dynamic fields were last updated

Example: 01/04/2021 00:25:01

Name: String representing name of bike stand

Example: BLESSINGTON STREET

Bike Stands: An integer representing number of bikes slot for the stand

Example: 40

Available Bike Stand: An integer representing number of available bike parking slot.

Example:10

Available Bikes: An integer representing number of available bikes now

Example: 4

Status: A string representing status of bike station

Example: Open

Address: String representing address of bike stand

Latitude: A float representing latitude position of the bike station

Example: 53.35677

Longitude: A float representing longitude position of the bike station

Example: -6.26814

Dublin bus data cater to public transportation for long-distance. It does have static and dynamic information. But all the bus has a fixed schedule and can be availed only as per schedule even for short travel. Whereas Dublin bike sharing can be used at any time (without waiting) and have penetration even the inner part of the street which is lacking with the bus. So even though these two datasets are transport-related, but they have a different use case. Each dataset has one similarity in that it can depict people movement time windows. Most travel happens during the daytime.

As part of the assignment, we used bus data set to find the bus schedule, delay, the maximum number of buses stopping at the station and total travel distance by bus on a single day. A person can take a bus to commute for some distance and then take the shared bike to reach the nearest point of his destination. So, having a bike station near the bus station will improve the user experience. Availability of sufficient bikes at bike stations all the time especially during peak hours will make public transport reliable. When commuter drops bike at the station, bike filling vehicle picks some bikes and put on the station where bike count is low. This balances the available bike count per station.

TASK:

As part of the assignment, I am proposing a task to find out the number of bikes available per hour windows per station during most commutable time 7 AM until 8 PM. As bike availability data is updated every 5 secs, I am taking an average of available bikes count per hour per station. Also, I am considering only weekday data for the analysis. The aim here is to make bike filling service efficient and able to serve people better. This exercise will also give clarity about during which time windows bike available count becomes low and the station where people are not using bike-sharing service often.

SPARK core API & SPARK SQL both can be used to analyse the historical dataset here, I am choosing SPARK SQL to do analysis.

* SQL has advantage that its query planner makes better optimisation before converting everything to spark core API.
* The temporary view created during intermediate stages makes debugging better.

Implementation:

Here after 1st step, the bike counts mean running average number of bike present at the bike station. Bike refers to bike/cycle.

1. Filter out row from weekend data and when bike stand is closed.



1. Select rows where Time is between 7AM until 7.59 PM and group by the row based on hours window and apply aggregation function min and max on Available Bikes



1. At this point have two data-frame, having min & max bike count per hours per station id.

As these max and min bike count belongs to some station for hour windows. Joining it with original data frame before doing grouping will give the station id for the min and max bike counts. So doing two joins separately to get station id where available bike was low and high.



1. Now we have two data frame having stations id where bike count was low and bike count was high. Joining these two will give data-frame having minimum and maximum bike count per hours windows and their station id.



I ran the code on complete dataset except



The results indicate there are some stations where bikes are not utilised by much and this can be transferred to other station where bike count is low.

As part of future work, I would join this with original data-frame to with the solution data frame here to get latitude, longitude to route the bikes between station where bikes are used frequently.

This task is challenging, and it involved filtering, multiple joins, aggregation, and transformation. As I coded this fully, I have attached the code with the report.