SINGAPORE UNIVERSITY OF SOCIAL SCIENCES

ICT233

End-of-Course Assessment – January Semester 2022

Data Programming

INSTRUCTIONS TO STUDENTS:

1. This End-of-Course Assessment paper comprises of **TWO (2)** questions and **SEVEN (7)** pages (including the cover page).

2. You are to include the following particulars in your submission: Course Code, Title of the ECA, SUSS PI No., Your Name, and Submission Date.

3. Late submission will be subjected to the marks deduction scheme. Please refer to the Student Handbook for details.

IMPORTANT NOTE

ECA Submission Deadline: Saturday, 14 May 2022, 12 noon.

Please Read This Information before You Start Working on your ECA

This ECA carries 70% of the course marks and is a compulsory component. It is to be done individually and not collaboratively with other students.

Submission

You are to submit the ECA assignment in exactly the same manner as your tutor-marked assignments (TMA), i.e. using Canvas. Submission in any other manner like hardcopy or any other means will not be accepted.

Electronic transmission is not immediate. It is possible that the network traffic may be particularly heavy on the cut-off date and connections to the system cannot be guaranteed. Hence, you are advised to submit your assignment the day before the cut-off date in order to make sure that the submission is accepted and in good time.

Once you have submitted your ECA assignment, the status is displayed on the computer screen. You will only receive a successful assignment submission message if you had applied for the e-mail notification option.

ECA Marks Deduction Scheme

Please note the following:

*a) Submission Cut-off Time – Unless otherwise advised, the cut-off time for ECA submission will be at **12:00 noon** on the day of the deadline. All submission timings will be based on the time recorded by Canvas.*

*b) Start Time for Deduction – Students are given a grace period of 12hours. Hence calculation of late submissions of ECAs will begin at **00:00 hrs** the following day (this applies even if it is a holiday or weekend) after the deadline.*

*c) How the Scheme Works – From 00:00 hrs the following day after the deadline, **10 marks** will be deducted for each **24-hour block**. Submissions that are subject to more than 50 marks deduction will be assigned **zero mark**. For examples on how the scheme works, please refer to Section 5.2 Para 1.7.3 of the Student Handbook.*

Any extra files, missing appendices or corrections received after the cut-off date will also not be considered in the grading of your ECA assignment.

Plagiarism and Collusion

Plagiarism and collusion are forms of cheating and are not acceptable in any form of a student's work, including this ECA assignment. You can avoid plagiarism by giving appropriate references when you use some other people's ideas, words or pictures (including diagrams). Refer to the American Psychological Association (APA) Manual if you need reminding about quoting and referencing. You can avoid collusion by ensuring that your submission is based on your own individual effort.

The electronic submission of your ECA assignment will be screened through a plagiarism detecting software. For more information about plagiarism and cheating, you should refer to the Student Handbook. SUSS takes a tough stance against plagiarism and collusion. Serious cases will normally result in the student being referred to SUSS's Student Disciplinary Group. For other cases, significant marking penalties or expulsion from the course will be imposed.

Answer all questions. (Total 100 marks)

There are 5 datasets:

- humidity.csv
- rain.csv
- temperature.csv
- wind.csv
- others.csv

These 5 CSV files are created with some modifications from the original dataset (source: https://data.cityofchicago.org/Parks-Recreation/Beach-Weather-Stations-Automated-Sensors/k7hf-8y75). The necessary data dictionary is provided in the given link.

Each of them is a data stream from some IOT sensors. For example, temperature.csv is the past data stream from temperature sensors. We want to build a simple **digital twin** (https://en.wikipedia.org/wiki/Digital_twin) to centralize data from all sensors.

A record in each CSV file can be identified uniquely by (Station Name, Measurement Timestamp).

Question 1

Objectives:

- Understand dataset with Data Scientist mindset.
- Exposure to real-world dataset analysis.
- Understand and design computation logic and routines in Python.
- Assess use of Pandas and Dataframes to perform extract, load, transformation and calculation operations.
- Conduct visualization in an appropriate way.
- Structure code in appropriate methods (functions), looping and conditions.

(a) Load in the 5 CSV files into 5 dataframe respectively and develop (perform) the following **TWO (2)** steps:

Step 1: For each dataframe of sensor readings, we introduce a new column called "Sensor Type".

- For the humidity dataset, the field "Sensor Type" of all rows has the value of "humidity".
- For the rain dataset, the field "Sensor Type" of all rows has the value of "rain".
- For the temperature dataset, the field "Sensor Type" of all rows has the value of "temperature".
- For the wind dataset, the field "Sensor Type" of all rows has the value of "wind".
- For the others dataset, the field "Sensor Type" of all rows has the value of "others".

Step 2: Combine 5 dataframes into **ONE (1)** dataframe and sort it by *Measurement Timestamp* ascendingly

- The resulted data frame contains these columns: *Station Name, Measurement Timestamp, Sensor Type, Air Temperature, Wet Bulb Temperature, Humidity, Rain Intensity, Interval Rain, Total Rain, Precipitation Type, Wind Direction, Wind Speed, Maximum Wind Speed, Barometric Pressure, Solar Radiation, Heading,* and *Battery Life*.

(5 marks)

(b) We target to develop and enrich the dataframe outputted from **Q1(a)** to represent the digital twin with data from all sensors. You may find more information about digital twin from https://en.wikipedia.org/wiki/Digital_twin

PER station name and **PER** measurement timestamp **T** from **Q1(a)** dataframe, we enrich **each** row to contain all **latest** sensor values up until the timestamp **T inclusively** for that station name.

For example, (shown in Figure 1), for the row with index 0 (63rd Street Weather Station, 2021-05-17 14:59:59:380916…), the value is obtained from temperature dataset (indicated with value "temperature" under *Sensor Type* column in Figure 1) and its value 15.7 in column *Air Temperature* has the latest value of all Air Temperature sensor readings recorded before 14:59:59:380916 inclusively gathered by the station 63rd Street Weather Station.

For the row with index 1 (63rd Street Weather Station, 2021-05-17 14:59:59:380916…), the values are obtained from rain dataset (*Rain Intensity, Interval Rain, Total Rain* columns) and the value in *Air Temperature* is copied from previous row (row with index 0).

Another example is to look at the row with index 6, only the value in *Humidity* column is updated to 84.0 and the rest of values in the row are copied from previous row (row with index 5).

Apply this logic to enrich all rows of the dataframe that obtained from Q**1(a)**.

(10 marks)

	Station Name	Measurement Timestamp	Humidity	Sensor Type	Rain Intensity	Interval Rain	Total Rain	Air Temperature	Wet Bulb Temperature	Wind Direction	Wind Speed	Maximum Wind Speed	Heading	Precipitation Type	Barometric Pressure	Solar Radiation	Battery Life
0	63rd Street Weather Station	2021-05-17 14:59:59.380916	NaN	temperature	NaN	NaN	NaN	15.7	14.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	63rd Street Weather Station	2021-05-17 14:59:59.956473	NaN	rain	0.0	0.0	11.0	15.7	14.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	63rd Street Weather Station	2021-05-17 15:00:00.321877	NaN	wind	0.0	0.0	11.0	15.7	14.0	51.0	1.3	2.7	352.0	NaN	NaN	NaN	NaN
3	63rd Street Weather Station	2021-05-17 15:00:00.741304	NaN	others	0.0	0.0	11.0	15.7	14.0	51.0	1.3	2.7	352.0	0.0	1000.6	467.0	11.8
4	63rd Street Weather Station	2021-05-17 15:00:00.889681	83.0	humidity	0.0	0.0	11.0	15.7	14.0	51.0	1.3	2.7	352.0	0.0	1000.6	467.0	11.8
5	63rd Street Weather Station	2021-05-17 15:59:59.209392	83.0	rain	0.0	0.0	11.0	15.7	14.0	51.0	1.3	2.7	352.0	0.0	1000.6	467.0	11.8
6	63rd Street Weather Station	2021-05-17 15:59:59.424498	84.0	humidity	0.0	0.0	11.0	15.7	14.0	51.0	1.3	2.7	352.0	0.0	1000.6	467.0	11.8

Figure 1: Example output for Q1(b)

(c) **Per** Station Name, we want to formulate and analyse the enriched reading stream from Q**1(b)** at 1-hour interval.
For example, we break the day (on 2021-05-17) into 2021-05-17 00:00:00, 2021-05-17 01:00:00, ..., 2021-05-17 23:00:00. We break the day (on 2021-05-18) into 2021-05-18 00:00:00, 2021-05-18 01:00:00, etc.

Sometimes, there is a delay in sending back the readings by IOT sensors, we consider a 15 minutes 59 seconds' delay is acceptable in our analysis. Therefore, we split the global time clock into **non-overlapping** windows, e.g.: [2021-05-17 00:16:00, 2021-05-17 01:15:59], [2021-05-17 01:16:00, 2021-05-17 02:15:59], and so on. Data received within the window [2021-05-17 00:16:00, 2021-05-17 01:15:59] shall belong to the hourly timestamp 2021-05-17 01:00:00.

Per station, for records (rows) with *Measurement Timestamp* in between a window, we keep only the record (its sensor values) with the **latest** *Measurement Timestamp* to represent that window. For example, for a station, there are 10 records in between [2021-05-17 00:16:00, 2021-05-17 01:15:59], we keep only 1 record with the **latest** *Measurement Timestamp* and set its *Hourly Timestamp* to 2021-05-17 01:00:00, and discard the other 9 rows of this time window. As a result, we have a new stream of readings at 1 hour interval per Station Name.

(10 marks)

	Station Name	Hourly Timestamp	Measurement Timestamp	Humidity	Rain Intensity	Interval Rain	Total Rain	Air Temperature	Wet Bulb Temperature	Wind Direction	Wind Speed	Maximum Wind Speed	Heading	Precipitation Type	Barometric Pressure	Solar Radiation	Battery Life
0	63rd Street Weather Station	2021-05-17 15:00:00	2021-05-17 15:00:00.889681	83.0	0.0	0.0	11.0	15.7	14.0	51.0	1.3	2.7	352.0	0.0	1000.6	467.0	11.8
1	63rd Street Weather Station	2021-05-17 16:00:00	2021-05-17 16:00:00.452415	84.0	0.0	0.0	11.0	16.1	14.4	51.0	1.1	1.9	351.0	0.0	1000.3	283.0	11.9
2	63rd Street Weather Station	2021-05-17 17:00:00	2021-05-17 17:00:00.459542	81.0	0.0	0.0	11.0	15.7	13.8	85.0	1.0	2.3	351.0	0.0	1000.4	196.0	11.9

Figure 2: Example output for Q1(c)

(d) On a **SINGLE** diagram, design and visualize the air temperature of each station name along their hourly timestamp. Share **TWO (2)** insights which you may draw from the diagram.

(10 marks)

(e) Design and visualize the correlation between humidity and rain intensity when it is rainy. Share insights which you may draw from the diagram.

(10 marks)

Question 2

- Manipulate dataset with data scientist mindset.
- Exposure to real-world dataset analysis.
- Design computation logic and routines in Python.
- Design methods to extract and parse information from the internet.
- Assess use of Pandas and Dataframes to perform extract, load, transformation and calculation operations.
- Assess the Design and use of Database method to perform create and load operations.

(a) Scrape the family tree of British monarchs (https://en.wikipedia.org/w/index.php?title=Family_tree_of_British_monarchs&oldid=1043575587) by applying (using) **BeautifulSoup** library and store them into a SQLite table british_monarch_family_tree using **SQLite3** library with the following fields:

- id: the primary key
- name
- wiki_url

(15 marks)

(b) Apply **SQLite3** library to add two new fields into the british_monarch_family_tree table

- father_id: references to the field *british_monarch_family_tree.id*
- mother_id: references to the field *british_monarch_family_tree.id*

Then follow each Wikipedia URL stored in the table british_monarch_family_tree, which is scraped in the Q2(a), to scrape the father and mother information of each king/queen

- If there is no father / mother information, put null value under father_id / mother_id.
- If a father or mother does not exist in the british_monarch_family_tree table, insert him/her into the british_monarch_family_tree table
- Set the father_id and mother_id fields to represent the father and mother relationships

For example, there are 3 rows:

- (id: 1, name: A, wiki_url: A_url, father_id: null, mother_id: null)
- (id: 2, name: B, wiki_url: B_url, father_id: null, mother_id: null)
- (id: 3, name: C, wiki_url: C_url, father_id: 1, mother_id: 2)

These rows indicate that A is the father of C, and B is the mother of C. The null value of father_id / mother_id means that there is no father / mother information.

(15 marks)

(c) Apply **SQLite3** library to find all children of King "George III" (https://en.wikipedia.org/wiki/George_III).

(6 marks)

(d) Apply **SQLite3** library to find the father and mother of King "George III" (https://en.wikipedia.org/wiki/George_III).

(6 marks)

(e) Apply **SQLite3** library to find all siblings of King "George IV" (https://en.wikipedia.org/wiki/George_IV)

(6 marks)

(f) Apply **SQLite3/Dataframe** library to find all descendants of "Queen Victoria" (https://en.wikipedia.org/wiki/Queen_Victoria) from the table british_monarch_family_tree.

(7 marks)

----- END OF ECA PAPER -----