RMIT UNIVERSITY

COSC2820 Advanced Programming for Data Science

COSC 2820/2815
Assignment 1: Data Parsing, Cleansing and Integration

Assessment Type	Individual assignment. Submit online via Canvas→Assignments→Assignment 1. Marks awarded for meeting requirements as closely as possible. Clarifications/updates may be made via announcements/relevant discussion forums.
Due Date	Week 6, Sunday 29th August 2021, 11:59pm
Marks	25

1. Overview

Nowadays there are many job hunting websites including seek.com, Azuna.com, etc. These job hunting sites all manage a job search system, where job hunters could search for relevant jobs based on keywords, salary, and categories, etc. Job advertisement data analysis is becoming increasingly important and beneficial for job hunting sites, as they can be used to make improvements on the experience of users searching for jobs.

This assessment assumes that you, as a data analyst, are required to wrangle a large set of job advertisement records stored in xml format and with unknown data quality issues, you will also be required to integrate the given data set with another data source, identify and resolve conflicts in data integration. This assessment contains three major tasks that are specified as follows, which has to be completed in order:

- In Task 1, you will explore the first dataset, identify its format. You will then use appropriate Python tools and libraries to parse the data into a pandas dataframe;
- Once you successfully parse the data, in Task 2, you will need to explore the data further, identify and fix data problems in the dataset, and finally output the clean data as per required format.
- Then in Task 3, you will integrate the cleaned dataset (the output from Task 2) and a 2nd dataset. You will need to resolve any schema level conflicts, merge the data, and then identify and fix any data-level conflicts that may exist.

2. Learning Outcomes

This assessment relates to following learning outcomes of the course:

- CLO 1: Programmatically parse data in the required format;
- CLO 2: Programmatically identify and resolve data quality issues;
- CLO 3: Programmatically integrate data from various sources for data enrichment;
- CLO 5: Document and maintain an editable transcript of the data pre-processing pipeline for professional reporting.

3. Assessment details

The Assignment Folder

In this assignment, each student has one assignment folder, which is named with his/her student ID.
You can search and download your the assignment folder from **here** (note, you will need to login rmit account to be able to access this folder).

Each student assignment folder will contain two datasets, namely, '**<student_id>_dataset1.xml**' and '**<student_id>_dataset2.csv**', as well as two jupyter notebook templates, '**<student_id>_task1_2.ipynb**' and '**<student_id>_task3.ipynb**'

The datasets are different for each individual student. You should look for exactly the folder named with your student ID. We request you to double-check and ensure you work on the right datasets.

Note that you should work on your own datasets individually. Distributing, exchanging or comparing your assigned datasets with other students would breach the academic integrity policy.

The Data

In this assessment, you are given two job advertisement datasets.

- **<student_id>_dataset1.xml** is for Task 1 and 2, where you are required to parse and clean the data, and get it ready for Task 3.
- **<student_id>_dataset2.csv** is for Task 3, where you are required to integrate together with the output from Task 2, to create an integrated dataset of job advertisements.

Task 1. Parsing Data

In this task, you are required to parse the job advertisement data stored in '<student_id>_dataset1.xml'. The specific tasks you need to perform includes:

- Examine the structure and format of the provided dataset.
- Parse the data into a Pandas dataframe. After the data is parsed and loaded, you should have a DataFrame where each row is a job advertisement record, containing the following columns/attributes: Id, Title, Location, Company, ContractType, ContractTime, Category, Salary, OpenDate, CloseDate and SourceName. **Note, make sure all the columns are parsed with the corresponding attribute names.**

Table 1. Column Descriptions of the Pandas DataFrame

COLUMN	DESCRIPTION
Id	8 digit Id of the job advertisement
Title	Title of the advertised job position
Location	Location of the advertised job position
Company	Company (employer) of the advertised job position
ContractType	The contract type of the advertised job position
ContractTime	The contract time of the advertised job position
Category	The category of the advertised job position
Salary	Annual salary of the advertised job position

OpenDate	The opening time for the job application
CloseDate	The closing time for applying for the advertised job position
SourceName	The website where the job position is advertised

Note, for OpenDate and CloseDate, the format of the string in the xml is **YYYYMMDDThhmmss**, where **Y** indicates year, **M** indicates month, **D** indicates day, **T** is just a letter (means time), **h** indicates hour (24-hour format), **m** indicates minute, and **s** indicates second. For example, "20130312T150000" means 15:00:00 12th March 2013.

Task 2. Auditing and Cleansing Data

In this task, you are required to inspect and audit the parsed dataset to identify data problems and to fix those problems. The description of each column and its required format in the output cleaned dataset are shown in Table 2.

Table 2. Columns and the Required Format of the Cleaned Job Dataset DataFrame after Task 2

COLUMN	**[FORMAT] and Domain values**
Id	**[Integer]**
Title	**[String]**
Location	**[String]**
Company	**[String]** If there is no company information, the value should be 'non-specified'.
ContractType	**[String]** It could be 'full_time', 'part_time' or 'non-specified'.
ContractTime	**[String]** It could be 'permanent', 'contract' or 'non-specified'.
Category	**[String]** There are 8 possible categories: 'IT Jobs', 'Healthcare & Nursing Jobs', 'Engineering Jobs', 'Accounting & Finance Jobs', 'Sales Jobs', 'Hospitality & Catering Jobs', 'Teaching Jobs', 'PR, Advertising & Marketing Jobs'.
Salary	**[Float]** All the values need to be expressed to two decimal places, e.g., 80000.25. Also, all salary values must be valid float numbers and **not** null.
OpenDate	**[Datetime]** All the values need to be in the datetime format, e.g., 2013-03-12 15:00:00
CloseDate	**[Datetime]** All the values need to be in the datetime format, e.g., 2013-03-12 15:00:00
SourceName	**[String]**

Different generic and major data problems could be found in the data might include:

- Typos and spelling mistakes
- Irregularities, e.g., abnormal data values and data formats
- Violations of the Integrity constraint.
- Outliers
- Duplications
- Missing values
- Inconsistency, e.g., inhomogeneity in values and types in representing the same data

Hint: You might need to use non-graphical (e.g., statistics) and graphical (e.g., different plots) methods to explore the data in order to identify those problems.

Required Output for Task 1 and 2:

- After parsing and cleansing the dataset, you should output the clean dataset as '**<student_id>_dataset1_solution.csv**'
- All Python code related to Task 1 and 2 should be written in the jupyter notebook '**<student_id>_task1_2.ipynb**'
- Except for the code, you are also required to record **all the found errors as well as the way you handle them** in a CSV file '**<student_id>_errorlist.csv'**

The **<student_id>_errorlist.csv** should have the following columns and information:

Table 3. Error list table

COLUMN	DESCRIPTION
indexOfdf	the index of the record/row in the original dataset. If the data issue involves all rows, just put "ALL".
Id	the id of the job advertisement that has the data issue. If the data issue involves all job records, just put "ALL".
ColumnName	The name(s) of the column that the data issue locates. ● If the data issue involves more than one column, you can put multiple column names separated by a comma, e.g., "Cloname1,Colname2,Colname3". ● If the data issue involves all columns, just put "ALL".
Original	The original value of the cell. If the data issue involves all rows with different cell values, just put "ALL".
Modified	The modified value of the cell. If the data issue involves all rows with different modified cell values, just put "ALL".
ErrorType	The type of errors, for example, Missing Values, Violation of Integrity Constraint, Outliers, or any other errors you found.
Fixing	Describe how did you fix this problem

Below is the content of an example record in **<student_id>_errorlist.csv**. Note that values below are not indicative.

indexOfdf	Id	ColumnName	Original	Modified	ErrorType	Fixing
5	71528123	Location	Loden	London	Misspelling	change 'Loden' to 'London'

Important Notes:

- The way you describe the problem (i.e., ErrorType) or how you fix the problem (i.e., Fixing) in the **<student_id>_errorlist.csv** is flexible. However, this file is very important for marking, and you need to ensure the format you record the errors are as per requirement above. If you fail to record any errors in the file, you will lose those marks even if your jupyter notebook contains the relevant code.
- You will also need to record any errors/problems you found in the file, even for those you decide **not** to fix (e.g., if the found problem is due for a more detailed and careful analysis rather than handled by a simple replacement/deletion). For problems you found but not fixed (in which case, you can leave the "Modified" column empty), you will need to provide justification on why you choose not to fix them in the "Fixing" column as well as in your jupyter notebook.
- For missing values, there are multiple ways to handle it. If you decided to simply delete all records with missing values, you will have to provide a well justified reason on why you think that's a suitable way in this context.

Task 3. Integrating the Job datasets

In this task, you will be given a 2nd job advertisement dataset **<student_id>_dataset2.csv**. All data in this dataset are coming from another datasource www.jobhuntlisting.com. You are required to integrate this dataset with the output from Task 2, i.e., **<student_id>_dataset1_solution.csv**.

To complete this task successfully, you are required to do the following:

1. **Resolving schema conflicts and merging data:** Inspect and compare the schema of **<student_id>_dataset1_solution.csv** and **<student_id>_dataset2.csv** to identify and resolve any schema conflicts. You will need to write Python code to
    a. Resolve any schema conflicts. You will need to adopt the schema in **Table 1** as your global schema. Hint: **<student_id>_dataset2.csv** does not have 'Id' information, however, you can write your own id generator for records in this dataset. However, please do **NOT** change the job Ids in the first dataset **<student_id>_dataset1_solution.csv**.
    b. Implement the semantic mapping and integrate the two data sets **<student_id>_dataset1_solution.csv** and **<student_id>_dataset2.csv** to produce one unified table.

2. **Resolving data conflicts:** Inspect tuples/instances for data conflicts in the unified table. In this step, you are required to do the following:
    a. Use Pandas libraries to detect and resolve duplications in the unified table.
    b. Identify a proper global/unique key for the integrated job data and explain your chosen key in the notebook, i.e., why you think the chosen key can be used as a unique identifier of a job advertisement.
3. Finally, you should output the integrated dataset as **<student_id>_dataset_integrated.csv**

Note that all Python code related to Task 3 should be written in **<student_id>_task3.ipynb**.

Note also that you could assume the given data in **<student_id>_dataset2.csv** are clean, i.e., you don't need to clean data in this dataset.

Summary of Input and Output from the Tasks

Following is the summary of the input, output for the different tasks in this assignment:

Task	Input	Output	Jupyter notebook
Task 1	<student_id>_dataset1.xml	NA	
Task 2	Follow from Task 1	<student_id>_dataset1_solution.csv, <student_id>_errorlist.csv	<student_id>_task1_2.ipynb
Task 3	<student_id>_dataset1_solution.csv (from Task 2), <student_id>_dataset2.csv	<student_id>_dataset_integrated.csv	<student_id>_task3.ipynb

4. Marking Guidelines

Marking Criteria

- ***Mechanical pass****:* Your outputs will be compared against the expected output. Therefore, marking will be based on the similarity between what we expect (as discussed in the instructions) and what we receive from you. It is extremely important to carefully follow the instructions to produce the expected output. Otherwise, you may easily lose many points for simple mistakes (e.g. typos in the names of the attributes, format of the files, not loading essential libraries, different file names/path, etc).
- ***Expert pass****:* Your jupyter notebook will be checked by an expert to validate the logic and flow, proper use of libraries and functions, and clarity of codes, comments, structure and presentation.
- You need to ensure all the codes and files that are required to run your code are included in the submission. The expert will **NOT** fix your code's problem even if it is a simple typo in an attribute name or an imported library.

Mark Allocations

- Task 1 Data Parsing [7%]
    - Implementation [5%]
    - Notebook presentation [2%], proportional to the percentage of completion in implementation
- Task 2 Data Cleansing [12%]
    - Implementation [10%]
    - Notebook presentation [2%], proportional to the percentage of completion in implementation
- Task 3 Data Integration [6%]
    - Implementation [5%]
    - Notebook presentation [1%], proportional to the percentage of completion in implementation

For all Tasks 1, 2, and 3, you are required to maintain an auditable and editable transcript, and communicate any justification of methods/approaches chosen, results, analysis and findings through jupyter notebook. The presentation of the jupyter notebook accounts for certain percentages of the allocated mark for each task, proportional to the percentage of completion of the task, as per specified above. The rubric for Notebook Presentation (including code commenting and notebook content) is common across Task 1, 2 and 3. Please refer to the marking rubric.

5. Submission

The final submission of this milestone will consist of:

- Your student folder (named with your student id). This directory should contain:
    - The given datasets **<student_id>_dataset1.xml** and **<student_id>_dataset2.csv**
    - The required output from Task 2, including **<student_id>_dataset1_solution.csv** and **<student_id>_errorlist.csv**;
    - The required output from Task 3, **<student_id>_dataset_integrated.csv**;
    - The jupyter notebooks **<student_id>_task1_2.ipynb**, and **<student_id>_task3.ipynb**, which contains all your codes, descriptions and comments;
- Before submission, you should restart your kernel and rerun your code from beginning to the end to make sure everything works as expected. You will keep all the outputs in the notebook in submission. However, during the expert pass, the assessor will re-run your notebook. Therefore, please make sure everything required to run your code is included in the submission folder. If there are external libraries you used in your assignment, you can put a comment on the top of the jupyter notebook.
- Make sure the output files are properly named according to the instruction.
- Zip the folder with the same name (ie **<student_id>.zip**) and upload to Canvas for submission.

Assessment declaration:
When you submit work electronically, you agree to the assessment declaration:
https://www.rmit.edu.au/students/student-essentials/assessment-and-exams/assessment/assessment-declaration

Late Submission Penalty

Late submissions will incur a 10% penalty on the total marks of the corresponding assessment task per day or part of day late. Submissions that are late by 5 days or more are not accepted and will be awarded zero, unless special consideration has been granted. Granted Special Considerations with a new due date set more than 2 weeks after the original due will automatically result in an equivalent assessment in the form of a practical test with interview, assessing the same knowledge and skills of the assignment (location and time to be arranged by the course coordinator). Please ensure your submission is correct (all files are there, compiles etc), re-submissions after the due date and time will be considered as late submissions.

6. Academic integrity and plagiarism (standard warning)

Academic integrity is about honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge and ideas. You should take extreme care that you have:

- acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e. directly copied), summarised, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods,
- provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviours, including:

- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students

For further information on our policies and procedures, please refer to https://www.rmit.edu.au/students/student-essentials/rights-and-responsibilities/academic-integrity