DATA1002 Week 4 Tutorial

Monday 25/08/25

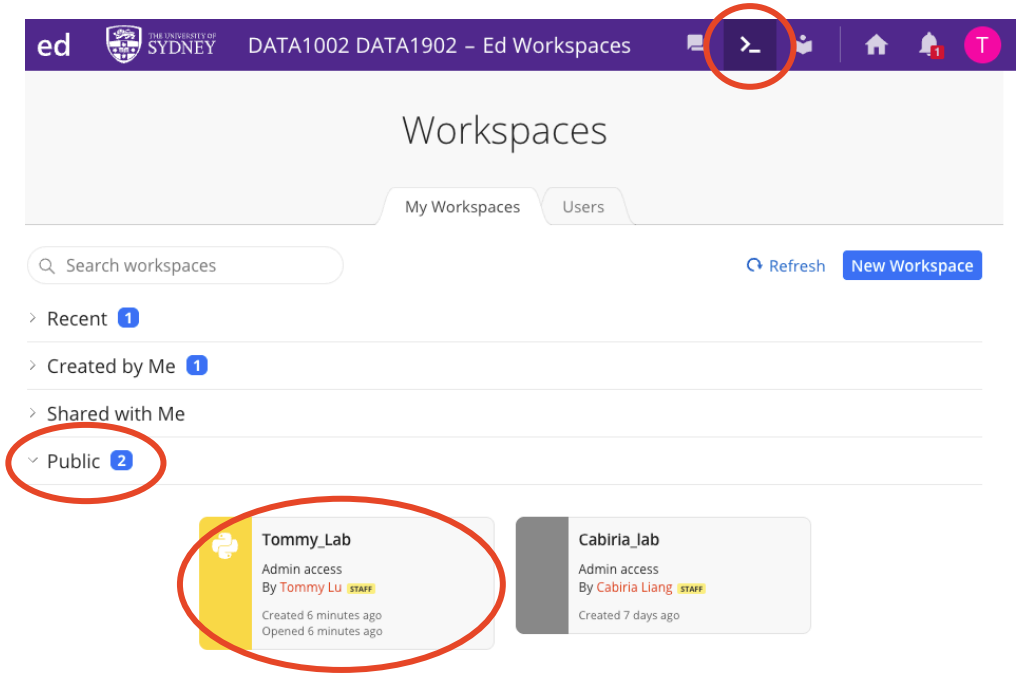
Tutorial Outline

- Content revision (Lists), Python Demo
- Content revision (Aggregation), Excel Demo
- Work on Assignment 1
  - Group Formation
  - Initial Research

THE UNIVERSITY OF SYDNEY

Tutor: *Tommy Lu*

Access the material for this tutorial through **Ed Workspaces**



Housekeeping

Assignment 1 out now!

1st hour we'll be revising content.

2nd hour we'll be working on the Assignment.

You'll get a break from me next week!

Content Revision

Lists

Lists

- A built-in collection type

way of storing data

Lists

Tuples, Dictionaries, etc.

- A built-in collection type

way of storing data

Lists

Tuples, Dictionaries, etc.

- A built-in collection type
- An object, that is made up of other objects arranged in a sequence
  - E.g. `["me", "I", "it"]`
- The elements can be of mixed types, even also a (nested) list
  - E.g. `["me", 5, False, [3.0, "it", "I"]]`
- Repetition is allowed
  - E.g. `["me", "I", "me"]`
- Empty list `[]`
- `range(6)` is the list of ints `[0,1,2,3,4,5]`

List Indexation

```
alist = ["me", "I", "it"]
```

What is the index of "I" in the list, alist?

```
alist = ["you", ["I", 0], 6]
```

What about now?

Searching Lists

- To find the index of an element:
  - `alist.index(value)`
    - `alist = ["me", "I", 5, "it"]`
    - `alist.index("I") = 1`
  - `alist.count(value)`
    - Returns the number of occurrences in the list which are equal to the value

Slices

- Gives another list, made of some of the elements for the original list

```
List[Initial : End : IndexJump]
```

  - `alist = ["me", "I", 5, "it"]`
  - `alist[1:3]`, make a list from items at index 1, 2
    - `["I", 5]`
  - `alist[0:3:2]`, make a list from items at index 1, 2
    - `["me", 5]`
  - `alist[::-1]`, make a reverse list
    - `["it", 5, "I", "me"]`

Mutation

- Modify the content of lists by assigning to a valid offset in the list
  - `alist = ["me", "I", 5, "it"]`
  - `alist[2] = "hi"`
  - `alist is now ["me", "I", "hi", "it"]`

Numbers and strings are immutable!

("I", "bee", [1, "c"], "a")

List Comprehensions

List comprehensions in Python offer a concise and efficient way to create new lists based on existing lists.

List Comprehensions

List comprehensions in Python offer a concise and efficient way to create new lists based on existing lists.

```
ylist = []
for x in xlist:
    if condition-on-x:
        ylist.append(expr-with-x)
```

Has an equivalent outcome to

```
ylist = [expr-with-x for x in xlist if condition-on-x]
```

List Comprehensions

List comprehensions in Python offer a concise and efficient way to create new lists based on existing lists.

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

newlist = [x for x in fruits if "a" in x]

print(newlist)
```

List Comprehensions

List comprehensions in Python offer a concise and efficient way to create new lists based on existing lists.

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

newlist = [x for x in fruits if "a" in x]

print(newlist)
```

```
newlist = [expression for item in iterable if condition == True]
```

Python Exercises

Lists

Dealing With Lists

```python
## WEEK 4 EXERCISES
# TIP: When focussing on just one exercise, comment out the other exercises
# TIP: Block comment by highlighting a section, then hit ctrl/cmd + /

# Exercise 1: Debugging
# The list below contains a series of numbers. The script should create a list of numbers where each
# number must be greater than the sum of all previous numbers in the original list. Fix the errors!
# For example, with an original list: [1, 5, 6, 13, 20, 50]
# The output list would be:           [1, 5, 13, 50]

ls = [1, 5, 6, 13, 20, 50]

for i in range(len(ls) + 1):
    if sum(ls[0:i+1]) <= ls[i]:
        output_list += ls[i]
print(output_list)

# BONUS Exercise (if already familiar with Python): Code Cracker
# You've received a hidden message
# By using the chr() function, loops, continue, and break
# Skip all numbers less than 50 and all numbers divisible by 7
# Reports have come in however if you go past 999 you will trigger an alert
# Translate the numbers posthaste!

nums = [
    42, 103, 39, 10, 100, 97, 49, 2, 70, 121, 21,
    84, 693, 13, 33, 999, 116, 111, 111, 102, 97, 114
]

message = ""

# FILL IN HERE!
```

Bonus Bonus:
Try redoing with list comprehensions!

Content Revision

Aggregation

Data Aggregation Patterns

Simple Aggregation

Combine all items into one value (e.g. sum, max, min, mean, count etc.)

Filtered Aggregation

Summarise only items meeting a condition (i.e. filter, then aggregate)

Grouped Aggregation (Bucketing)

Split into groups, then summarise each (i.e. group, then aggregate)

Aggregation Over Groups

Summarise the summaries (i.e. group, then aggregate, then aggregate again)

Data Aggregation Patterns

Simple Aggregation

Combine all items into one value (e.g. sum, max, min, mean, count etc.)

Filtered Aggregation

Summarise only items meeting a condition (i.e. filter, then aggregate)

Name	Age	Height (cm)	Student Type
Ben	84	160	International
Chen	46	175	Domestic
Darcie	41	155	International
Jose	22	184	Domestic
Kim	23	156	Domestic
Vinitha	89	141	International

Grouped Aggregation (Bucketing)

Split into groups, then summarise each (i.e. group, then aggregate)

Aggregation Over Groups

Summarise the summaries (i.e. group, then aggregate, then aggregate again)

Data Aggregation Patterns

Simple Aggregation

Combine all items into one value (e.g. sum, max, min, mean, count etc.)

Name	Age	Height (cm)	Student Type
Ben	84	160	International
Chen	46	175	Domestic
Darcie	41	155	International
Jose	22	184	Domestic
Kim	23	156	Domestic
Vinitha	89	141	International

Filtered Aggregation

Summarise only items meeting a condition (i.e. filter, then aggregate)

Grouped Aggregation (Bucketing)

Split into groups, then summarise each (i.e. group, then aggregate)

Aggregation Over Groups

Summarise the summaries (i.e. group, then aggregate, then aggregate again)

Data Aggregation Patterns

Simple Aggregation

Combine all items into one value (e.g. sum, max, min, mean, count etc.)

Filtered Aggregation

Summarise only items meeting a condition (i.e. filter, then aggregate)

Name	Age	Height (cm)	Student Type
Ben	84	160	International
Chen	46	175	Domestic
Darcie	41	155	International
Jose	22	184	Domestic
Kim	23	156	Domestic
Vinitha	89	141	International

Grouped Aggregation (Bucketing)

Split into groups, then summarise each (i.e. group, then aggregate)

Aggregation Over Groups

Summarise the summaries (i.e. group, then aggregate, then aggregate again)

Data Aggregation Patterns

Simple Aggregation

Combine all items into one value (e.g. sum, max, min, mean, count etc.)

Name	Age	Height (cm)	Student Type
Ben	84	160	International
Chen	46	175	Domestic
Darcie	41	155	International
Jose	22	184	Domestic
Kim	23	156	Domestic
Vinitha	89	141	International

Filtered Aggregation

Summarise only items meeting a condition (i.e. filter, then aggregate)

Grouped Aggregation (Bucketing)

Split into groups, then summarise each (i.e. group, then aggregate)

Aggregation Over Groups

Summarise the summaries (i.e. group, then aggregate, then aggregate again)

Excel Exercises

Aggregation

Produce One of Each Aggregation

using climate_data_2017.csv (Find on Ed Workspace)

Simple Aggregation

Combine all items into one value (e.g. sum, max, min, mean, count etc.)

Filtered Aggregation

Summarise only items meeting a condition (i.e. filter, then aggregate)

Grouped Aggregation (Bucketing)

Split into groups, then summarise each (i.e. group, then aggregate)

Aggregation Over Groups

Summarise the summaries (i.e. group, then aggregate, then aggregate again)

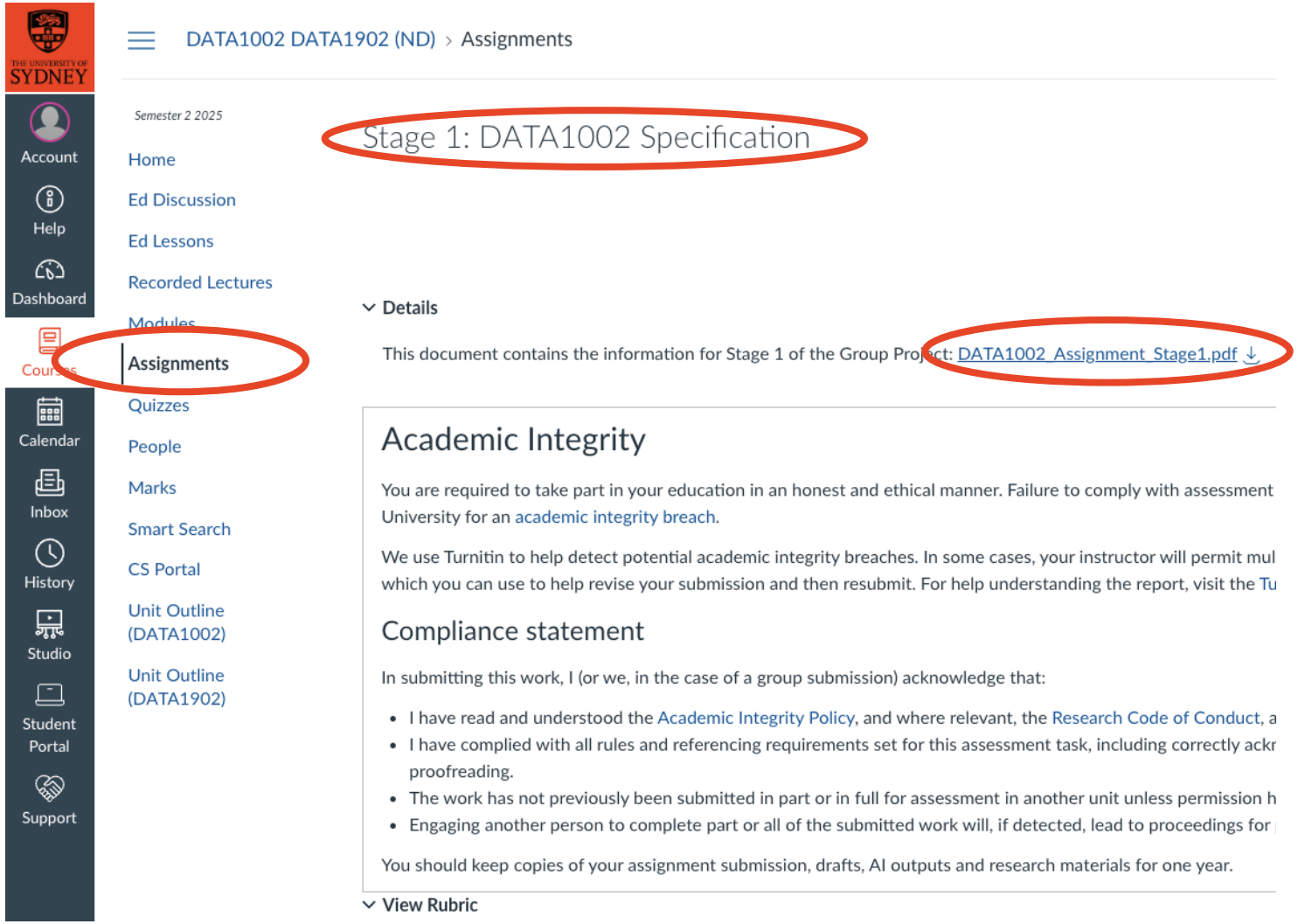
Let’s Take a Short Break!

Assignment 1

What does it actually involve for you?

NB: Where do you get information about the Assignment?

Where is it?


THE UNIVERSITY OF SYDNEY
Account
Help
Dashboard
Courses
Calendar
Inbox
History
Studio
Student Portal
Support
DATA1002 DATA1902 (ND) › Assignments
Semester 2 2025
Home
Ed Discussion
Ed Lessons
Recorded Lectures
Modules
Assignments
Quizzes
People
Marks
Smart Search
CS Portal
Unit Outline (DATA1002)
Unit Outline (DATA1902)
Stage 1: DATA1002 Specification
Details
This document contains the information for Stage 1 of the Group Project: DATA1002_Assignment_Stage1.pdf
Academic Integrity
You are required to take part in your education in an honest and ethical manner. Failure to comply with assessment
University for an academic integrity breach.
We use Turnitin to help detect potential academic integrity breaches. In some cases, your instructor will permit mul
which you can use to help revise your submission and then resubmit. For help understanding the report, visit the Tu
Compliance statement
In submitting this work, I (or we, in the case of a group submission) acknowledge that:
I have read and understood the Academic Integrity Policy, and where relevant, the Research Code of Conduct, a
I have complied with all rules and referencing requirements set for this assessment task, including correctly ackr
proofreading.
The work has not previously been submitted in part or in full for assessment in another unit unless permission h
Engaging another person to complete part or all of the submitted work will, if detected, lead to proceedings for
You should keep copies of your assignment submission, drafts, AI outputs and research materials for one year.
View Rubric

What is it?

Group Project Stage 1

Due: 17:00 pm on Sunday at the end of week 8 (Sep 28th)

Value: 20% of Total Mark

Note: Get started your project ASAP. Discuss with your tutors and make use of Ed to ask questions.

1 Purpose

The Stage 1 Project is a collaborative data science investigation completed in groups of 3 or 4. It assesses your ability to identify a meaningful question, prepare and clean data, summarise and analyse it using Python.

2 Group Formation

This is your source of truth!

- Groups of 3–4 students.
- All group members must be enrolled in the same lab.
- The same mark is awarded to all members unless otherwise specified.
- Tutor approval is required for any group changes.

3 The Project Work for Stage 1

3.1 Define a Topic or a Question

The group should define questions or issues that are not simply a factual matter, but instead examine relationships where insights might be impactful for some stakeholder groups. We realise that you may not find data that completely resolves the issue you are targeting, but all the data should at least be helpful to provide some insights.

- Choose a topic that explores **relationships** (not just factual reporting).
- Justify the importance of your question.
- Include stakeholder relevance and real-world impact.

What is it?

Group Project Stage 1

Due: 17:00 pm on Sunday at the end of week 8 (Sep 28th)

Value: 20% of Total Mark

Note: Get started your project ASAP. Discuss with your tutors and make use of Ed to ask questions.

What you'll do:
1. Define Topic or Question

Define Topic or Question (max 1 page)	18%	1) Research question is clearly defined, relational (not factual). 2) Importance and relevance to stake-holders are explicitly justified. 3) Real-world impact is described.

What you'll do:
2. Select & Describe Data

Select and Describe Data (max 3 pages)	18%	1) At least 3 datasets with $\geq$300 records total are selected. 2) Each dataset is documented with schema (data dictionary), provenance (source chain + date), and noted limitations. 3) Original raw data is preserved and referenced.

What you'll do:
3. Ensure Data Quality

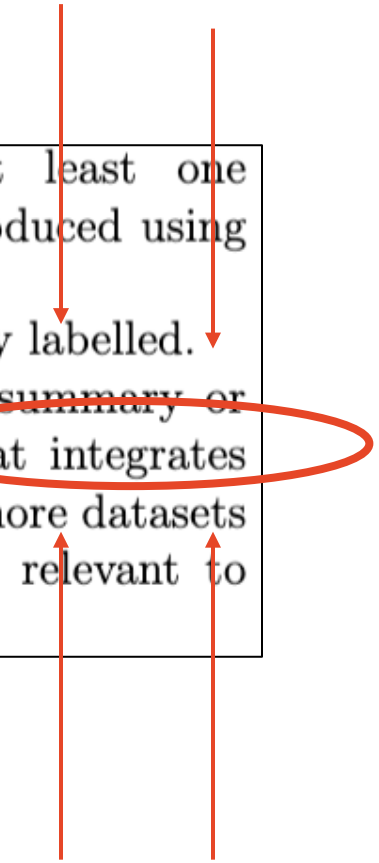
Ensure Data Quality (max 5 pages)	18%	1) Python used to check and clean each dataset (missing values, formatting issues, duplicates addressed). 2) Clear explanation of data transformations (if any). 3) Final dataset is of high quality and ready for analysis.

What you'll do:
4. Perform Simple Analysis

Perform Simple Analysis (max 5 pages)	22%	1) For each dataset, at least one meaningful summary is produced using Python. 2) All summaries are clearly labelled. 3) At least one combined summary or comparison is provided that integrates information across two or more datasets to highlight a relationship relevant to the research question.

What you'll do:
4. Perform Simple Analysis

Perform Simple Analysis (max 5 pages)	22%	1) For each dataset, at least one meaningful summary is produced using Python. 2) All summaries are clearly labelled. 3) At least one combined summary or comparison is provided that integrates information across two or more datasets to highlight a relationship relevant to the research question.



What you'll do:
5. Conclusion

Conclusion (max 1 page)	6%	1) Concise non-technical summary of findings. 2) Contributions of each group member are explicitly listed.

What you'll do:
6. Formatting & References

Formatting & References	6%	1) All datasets and literature are cited in APA 7th style. 2) Report formatting is professional and consistent with academic standards.

What you'll do:
7. Code & Data

Code and Data	12%	1) Provide all python code. 2) The code runs successfully within a reasonable time. 3) Code is well-structured, clearly commented, and properly documented to ensure readability and reproducibility. 4) Both the original raw data and the cleaned, processed datasets are included and appropriately organised.

Lab Activities

Working on Assignment 1

Group Formation

Get into groups of 3 – 4, all group members must:

- Come from the same lab *
- Be able to find time to meet outside of scheduled lab
- Be able to agree on the domain/topic to analyse for the project

Once you are happy with your group, let your tutor know!

Activity

Download specifications and start planning for the Assignment!

Exam-Style Questions

Question 1:

Discuss the concept of data aggregation and its significance in data science. What are some common aggregation functions, and how are they applied?

Exam-Style Questions

Data aggregation involves summarizing multiple data points into a single value, providing insights into the dataset. Common aggregation functions include **sum, mean, count, maximum, and minimum**. For example, calculating the average temperature from a list of daily temperatures helps understand climate trends.

Aggregation is significant in data science for reducing data complexity, identifying patterns, and informing decision-making. Functions like `sum()`, `max()`, and `mean()` facilitate these operations, enabling efficient data analysis and summarization.

Exam-Style Questions

Question 2:

How do data scientists handle corner cases in data aggregation? Provide an example and explain its importance.

Exam-Style Questions

Data scientists handle corner cases by **defining rules for edge scenarios**, ensuring robustness in aggregation functions. For instance, aggregating an empty list could return a default value or raise an error. Handling corner cases, like defining the sum of an empty list as 0, prevents program crashes and ensures meaningful results.

Addressing these cases is crucial for **reliable data analysis, maintaining data integrity, and providing accurate insights under all conditions.**

That’s it folks!

Remaining Ed Lessons, Questions, Assignment etc.