

# BinderHub Snakemake 使用 mock-up

## I. 前言

Depositar為得讓所儲存的資料集達成FAIR data(Findable, Accessible, Interoperable, Reusable), 引進了BinderHub來讓資料得以在內部進行運作驗證(interoperable), 而Snakemake的加入則是為了省去複雜的資料驗證程序, 並讓資料更新可以只重新執行需要的部分, 降低資源使用, 而"省去複雜的資料驗證程序"的部分則可以彈性使用, 讓使用者也可以以自己的資料去驗證實驗結果(Reusable)。

## II. 簡介

snakemake 是一個處理工作流程的工具, 透過與其他軟件搭配(ex: samtools, bwa ...)或單純使用shell, python指令建立各種規則(rule)對資料進行更分析、產生結果報告, 類似於Makefile, 但有功能使用更使用者友善一些。

## III. 內容

利用Snakemake來讓作者將其研究方法(分析順序)的Snakefile、script.sh、資料上傳至depositar, 當其他使用者帶著自己的資料集來驗證時, 上傳至binder中, 在終端機中輸入"bash script.sh", 系統便會自動下載snakemake與snakedeploy或其他所需要的環境, 並直接執行Snakefile, 而根據不同需求, Snakefile可能會要取用到其他地方(ex: Github)的資源, 或就在本地端執行即可, 若想將資料的處理分為前置作業與資料分析, 可使用多個Snakefile來達到多層次的工作流程。而Snakemake除了產生出工作流程後產生的成果外, 也有內建的功能可以靠指令產生工作流程圖(DAG有向無環圖)或report(.zip,html...).

## IV. snakemake基本教學

- mamba Setup (來源:  
<https://snakemake.readthedocs.io/en/stable/tutorial/setup.html#step-1-installin-g-mambaforge>)
  - Linux 環境

```
$ curl -L https://github.com/conda-forge/miniforge/releases/latest/download/Mambaforge-Linux-x86_64.sh -o Mambaforge-Linux-x86_64.sh
$ bash Mambaforge-Linux-x86_64.sh
```

### ➤ 下載跟安裝Miniconda 3

```
Welcome to Mambaforge 24.3.0-0

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>> |
```

```
Mambaforge will now be installed into this location:
/home/[redacted]/mambaforge

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below

[/home/sheepy/mambaforge] >>> |
```

```

=====
=
Copyright (c) 2019-2022, conda-forge
All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, th
is
list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation
and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributor
s
may be used to endorse or promote products derived from this software without
specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIE
D
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY
,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE US
E
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Do you accept the license terms? [yes|no]
>>> |

```

➤ 一直按Enter繼續

```

To activate this environment, use:

    micromamba activate /home/[redacted]mambaforge

Or to execute a single command in this environment, use:

    micromamba run -p /home/[redacted]mambaforge mycommand

installation finished.
Do you wish to update your shell profile to automatically initialize conda?
This will activate conda on startup and change the command prompt when activ
ated.
If you'd prefer that conda's base environment not be activated on startup,
run the following command when conda is activated:

conda config --set auto_activate_base false

You can undo this by running 'conda init --reverse $SHELL'? [yes|no]
[no] >>> |

```

➤ yes

```

[redacted]:~/binder3$ mamba --version
mamba 1.5.8
conda 24.7.1
[redacted]:~/binder3$ conda activate base
(base) [redacted]:~/binder3$ |

```

➤ 可以使用mamba了

- 環境設置(environment.yaml)

```
name: environment                                # 環境名稱

channels:
  - conda-forge
  - bioconda                                     # 生物信息學軟體包的 channel

dependencies:                                    # 函式庫
  - snakemake-minimal >=8.4.4                  # 可強調版本
  - python
  - matplotlib
  - pandas
  - numpy
  - plotly
  - folium
  - pip                                           # 這裡是 Conda 自身的 pip 包

# 如果有需要從 PyPI 安裝的包, 可以這樣放在 pip 下 (conda沒有的)
# - pip:
#   - python_package_not_in_conda
```

- script(劇本)

```
#!/bin/bash

# Check if Snakemake is installed, if not, install it
if ! command -v snakemake &> /dev/null
then
    echo "Snakemake not found, installing..."

    conda activate base
    mamba create -c conda-forge -c bioconda --name snakemake
    snakemake snakedeploy -y
fi

# Initialize conda in the script
source $(conda info --base)/etc/profile.d/conda.sh

conda activate snakemake

# Run the Snakemake workflow
snakemake --cores all --use-conda
```

- 先確認目前有沒snakemake
  - ➔ 沒有就安裝
  - ➔ 有就啟動環境
- 因為在建立環境時會問y/n, 所以加 -y 回答

```
if ! command -v snakemake &> /dev/null
then
    echo "Snakemake not found, installing..."
    # conda install -n base -c conda-forge mamba
    conda activate base
    mamba env create -n snakemake --file environment.yaml || mamba
env update -n snakemake --file environment.yaml
    # mamba create -c conda-forge -c bioconda -n snakemake
    snakemake
fi
```

- ex2(視範例)的用法, 不是下載snakemake與snakedeploy, 而是安裝 environment.yaml裡所需的環境, 以 `||` 來判斷, 若已有環境, 則看是否更新

- rule (規則)

```
rule all:
    input:
        "new_report.html"

rule unzip_file:                                # rule 後接 名稱
    input:
        "resource.zip"
    output:
        "resource/tp-marriage.csv",
        "resource/tp-economy.csv"
    shell:
        "unzip {input} -d ./"
```

**rule all** 是 **\$ snakemake** 的時候跑的

- rule attribute (規則屬性)

- input:
  - 放入這個rule會用到的檔案(可以不用用到的都放, 但放較好), 為達成workflow, 會放入前一個rule的output來強調順序
  - 路徑名需加"
  - 多個輸出以","分開 (ex: "report.zip", directory("data"))

- 有放在input的, 在其他地方使用時, 就可以用{input}取代, 反之, 若沒有, 則得打出完整路徑(ex: `workflow/Snakefile`), 有多個input時, 在其他地方按順在input的順序以{input[0]},{input[1]}...使用
- 指定rule的輸出為input: `rules.rule_name,output`
  - output:
- 放入這個rule會輸出的檔案(可以不用都放, 但放較好), 為達成workflow, 駛下一個rule可以靠的此rule的output建立順序
- 路徑名需加" "
- 多個輸出以","分開 (ex: "report.zip", `directory("data")`)
- 有放在output的, 在其他地方使用時, 就可以用{output}取代, 反之, 若沒有, 則得打出完整路徑, 有多個output時, 在其他地方按順在output的順序以{output[0]},{output[1]}...使用

```
rule generate_report:
    input:
        rules.run_workflow.output
    output:
        "report.zip"
    shell:
        '''
        snakemake -s workflow/Snakefile --report {output}
        rm -rf {input}
        '''
```

- shell:
- 寫入所要執行的shell指令
- 單個指令用 "" (ex: "snakemake -s workflow/Snakefile --report {output}")
- 多個指令用 "" ""
- 指令與python程式碼都要用"""" """"包住全部, 僅python\_code要另外用 `python -c "..python code.."` 在包python程式碼, shell指令不用

```
rule data_update:
    input:
        rules.deploy.output[0],
        rules.deploy.output[1]
    output:
        "workflow/new_snakefile"
    shell: """
        python -c "
import re
# Read the file
/* more */
"
```

```
mv {input[0]} {output}      # shell 指令
"""
```

- run:

- 直接放入python程式碼, 不用" "
- 加入shell指令用 shell("/...shell command.../")

```
run:
    with open(input.report) as f:
        report_content = f.read()

    # 讀取 CSV 文件內容
    csv_content1 = open(input.m_csv).read()
    /* more */
```

- script:

- 放入要執行的python\_code\_file

```
rule ebar_chart:
    input:
        "resource/tp-economy.csv"
    output:
        "result/bar/e-bar.html"
    script:
        "scripts/make_bar_chart.py"
```

- log:

- 輸出日誌文件的路徑
- 路徑名需加" "
- 在shell的指令候用 ... 2> {log} 完成

- params:

- 傳遞額外的參數到 shell 指令或其他地方
- 以{param}使用

// toy example

```
params:
    extra_params="--option1 value1 --option2 value2"
shell:
    "process_data {input} {params.extra_params} > {output}"
```

- threads: 和resources:

- threads -> 規則所需的CPU數量
- resources -> 規則所需的如內存、磁碟空間...

- 配置文件 (config.yaml)

```
# config.yaml
samples:
  A: data/samples/A.fastq
  B: data/samples/B.fastq
  C: data/samples/C.fastq

# Snakefile
def get_bwa_map_input_fastqs(wildcards):
    return config["samples"][wildcards.sample]
rule bwa_map:
    input:
        "data/genome.fa",
        get_bwa_map_input_fastqs
        # "data/samples/{sample}.fastq"
    output:
        "mapped_reads/{sample}.bam"
    params:
        rg=r"@RG\tID:{sample}\tSM:{sample}"
    log:
        "logs/bwa_mem/{sample}.log"
    threads: 8
    shell:
        "(bwa mem -R '{params.rg}' -t {threads} {input} | "
        "samtools view -Sb - > {output}) 2> {log}"

configfile: "config.yaml" # config file 的路徑
    ../more../
rule bcftools_call:
    input:
        fa="data/genome.fa",
        bam=expand("sorted_reads/{sample}.sorted.bam",
sample=config["samples"]),
        bai=expand("sorted_reads/{sample}.sorted.bam.bai",
sample=config["samples"])
    ../more../
```

- "config["samples"][wildcards.sample]"取得 config 檔案裡sample下的路徑, 在shell中使用{input}時會將三個路徑都當輸入跑一遍
- {sample} 是通配符, 會得到樣本名稱(不是路徑, 所以logs/bwa\_mem/裡面是A,B,C的log檔)
- bam=expand("sorted\_reads/{sample}.sorted.bam", sample=config["samples"])會讓 bam = sorted\_reads/{A,B,C}.sorted.bam 三個檔案, 在shell用 {input.bam} 會三個檔案都跑

- 內建輸出

- 輸出工作流程圖(DAG有向無環圖)

```
$ snakemake --dag /target/ | dot -Tsvg > /file_output_name/.svg
```

- 輸出report (內建workflow, statistics, about)

```
$ snakemake -s(op) /target_snakefile/ --report /report_name/ (.zip or .html ...)
```

- Snakemake參數

- -n (--dry-run) : 模擬試跑workflow(for debug) ex: \$ snakemake -n
- -p : 列出所有紀錄(for debug) ex: \$ snakemake -p
- -f (-F) : 強制重新跑完整workflow ex: \$ snakemake -f
- -n (-name): 指定名稱 ex: \$ conda env create -n example\_env
- --file: 指定安裝環境 ex: \$ conda env create -n example\_env --file environment.yaml
- -s : 指定跑的Snakefile ex: \$ snakemake -s new\_snakefile --report report.html

## V. 範例









### EX1: Data analysis of *saccharomyces cerevisiae* (Brewer's yeast)

(credit to Catalog example:

<https://github.com/snakemake-workflows/rna-seq-star-deseq2>)

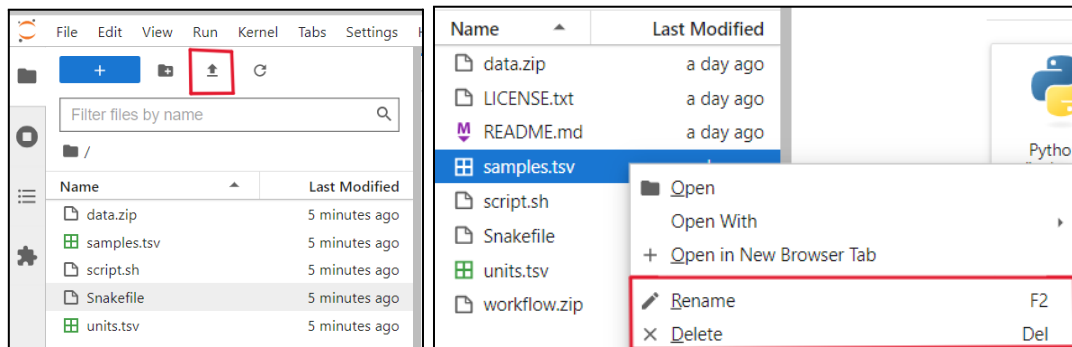
(1) 作者在depositar上資料集放入script,Snakefile,測資, workflow.zip(optional)

#### 資料與資源

	<b>LICENSE.txt</b> MIT LICENSE, provide by Johannes Köster	<a href="#">↻ 探索 ▾</a>
	<b>README.md</b> Instructions of how to reproduce and replicate	<a href="#">↻ 探索 ▾</a>
	<b>script.sh</b> Use: help install snakemake, snakedeploy and run workflow	<a href="#">↻ 探索 ▾</a>
	<b>workflow.zip</b> zip file of Snakefile and config.yaml	<a href="#">↻ 探索 ▾</a>
	<b>Snakefile</b> for running workflow	<a href="#">↻ 探索 ▾</a>
	<b>samples.tsv</b> Example: samples (can be removed to run other samples)	<a href="#">↻ 探索 ▾</a>
	<b>units.tsv</b> Example: units (can be removed to run other units)	<a href="#">↻ 探索 ▾</a>
	<b>data.zip</b> Example: data (can be removed to run other data) the path to data must be...	<a href="#">↻ 探索 ▾</a>

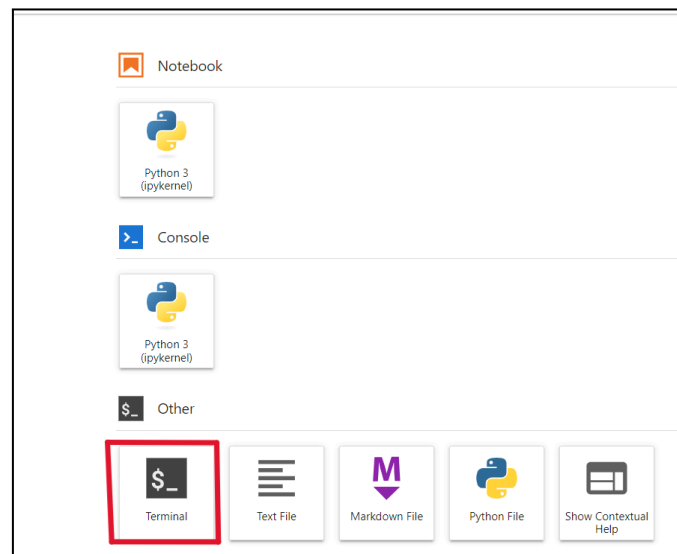


(2) 使用者在BinderHub中可將data.zip, sample, units更換為自己的資料, 或是直接跑測試資料



。上傳資料

。重新命名或刪除



。打開終端機

```
jovyan@jupyter-dataset-2dtest2-2ds4sht1vh:~$ bash script.sh
Snakemake not found, installing...
Channels:
- conda-forge
- bioconda
- defaults
Platform: linux-64
Collecting package metadata (repodata.json): -
```

。在終端機中輸入 `$ bash script.sh`

(3) Snakefile便會自動跑資料分析, 並將結果從zip檔解壓縮成資料夾, 使用者可在資料夾中的html檔中查看分析結果

```

Assuming unrestricted shared filesystem usage.
Building DAG of jobs...
Using shell: /usr/bin/bash
Provided cores: 16
Rules claiming more threads will be scaled down.
Singularity containers: ignored
Job stats:
-----
Job                                count
-----
align                               4
all                                 1
count_matrix                        1
cutadapt_pe                         4
cutadapt_pipe                       8
deseq2                              1
deseq2_init                        1
gene_2_symbol                      3
get_annotation                     1
get_genome                         1
multiqc                             1
pca                                 1
rseqc_gtf2bed                      1
rseqc_infer                        4
rseqc_innerdis                    4
rseqc_junction_annotation          4
rseqc_junction_saturation          4
rseqc_readdis                     4
rseqc_readdup                     4
rseqc_readgc                      4
rseqc_stat                         4
star_index                         1
total                              61

Select jobs to execute...
Execute 2 jobs...

```

## ➤ 秀出要執行的工作

```

Activating conda environment: .snakemake/conda/9e00c644315e1670fd769930d9ae25eb_
[Mon Aug 26 10:31:58 2024]
Finished job 24.
59 of 61 steps (97%) done
--- Attaching core tidyverse packages --- tidyverse 2.0.0 ---
✓dplyr      1.1.4      ✓readr      2.1.5
✓forcats    1.0.0      ✓stringr    1.5.1
✓ggplot2    3.5.1      ✓tibble     3.2.1
✓lubridate  1.9.3      ✓tidyr      1.3.1
✓purrr      1.0.2
--- Conflicts --- tidyverse_conflicts() ---
✗dplyr::filter() masks stats::filter()
✗dplyr::lag() masks stats::lag()
✗dplyr::select() masks biomaRt::select()
[Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors]
[Mon Aug 26 10:32:06 2024]
Finished job 1.
60 of 61 steps (98%) done
Select jobs to execute...
Execute 1 jobs...

[Mon Aug 26 10:32:06 2024]
localrule all:
  input: results/diffexp/treated-vs-untreated.diffexp.symbol.tsv, results/deseq2/normcounts.symbol.tsv, results/counts/all.symbol.tsv, results/qc/multiqc_report.html, results/pca.condition.svg, results/pca.condition.svg
  jobid: 0
  reason: Input files updated by another job: results/counts/all.symbol.tsv, results/qc/multiqc_report.html, results/pca.condition.svg, results/diffexp/treated-vs-untreated.diffexp.symbol.tsv, results/deseq2/normcounts.symbol.tsv
  resources: tmpdir=tmp
[Mon Aug 26 10:32:06 2024]
Finished job 0.
61 of 61 steps (100%) done

```

## ➤ 100% done



## ➤ reports的html檔

## EX2: 台北市金價與結婚人數分析 (以前面未提及過的內容為主)

➤ 利用而外的python code得到金價與結婚對數的bar chart

```
# make_bar_chart.py

import pandas as pd
import numpy as np
import plotly.graph_objects as go

# Access the input and output files from Snakemake
input_file = str(snakemake.input[0])
output_file = str(snakemake.output[0])

df = pd.read_csv(input_file)

if 'marriage' in input_file:
    col1 = '結婚對數/總計[對]'
    col2 = '年別'
    layout_title = '台北結婚對數變化圖'
    ytitle = '結婚對數/總計[對]'
    markcolor = 'blue'
elif 'economy' in input_file:
    col1 = '黃金[飾金]市價[元/臺錢]'
    col2 = '年底別'
    layout_title = '台北黃金市價變化圖'
    ytitle = '黃金市價[元/臺錢]'
    markcolor = 'gold'

bar = go.Bar(
    x=df[col2],
    y=df[col1],
    marker=dict(color=markcolor)
)

layout = go.Layout(
    title=layout_title,
    title_font_size = 30,
    xaxis=dict(title="年分"),
    yaxis=dict(title=ytitle)
)

fig = go.Figure(data=bar, layout=layout)
fig.write_html(output_file)
```

### ➤ 在snakefile裡產生html檔

```
# Snakefile
    ../../more../

rule mbar_chart:
    input:
        "resource/tp-marriage.csv"
    output:
        "result/bar/m-bar.html"
    script:
        "scripts/make_bar_chart.py"

    ../../more../

rule custom_report:

    ../../more../

    output:
        "new_report.html",
        "dag_graph.svg"
    run:
        # generate the DAG graph
        shell("snakemake --dag | dot -Tsvg > {output[1]}")

        # read report content

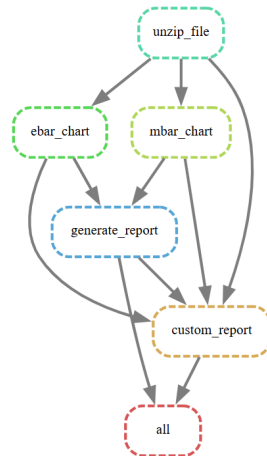
    ../../more../

    # generate the new HTML content
    new_content = f"""
    <!DOCTYPE html>
    <html>
        ../../more../

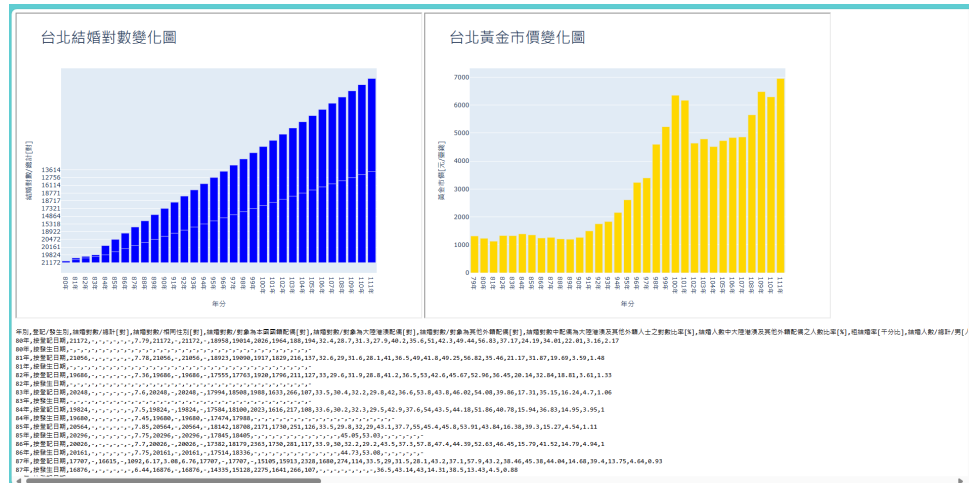
    </body>
    </html>
    """
    ../../more../
```

➤ output 出 三個檔案

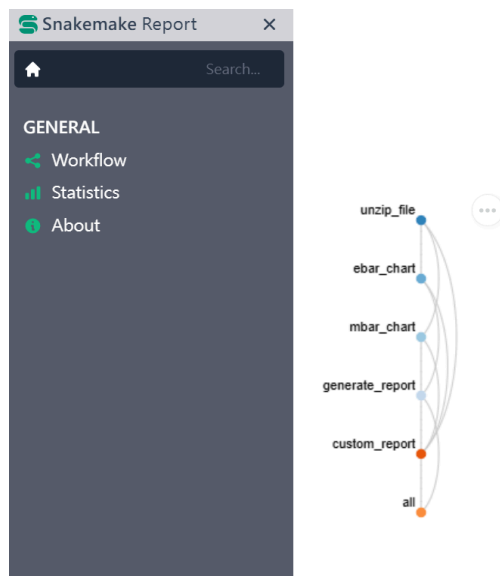
## 1. dag-graph



## 2. new\_report



## 3. report



(還不知道要怎麼改才可以把自己的資料加到snakemake內建的report裡)

depositar:

- snakemake paper report  
<https://pid.depositar.io/ark:37281/k5c4x9n7d>
- ex1: Data analysis of *saccharomyces cerevisiae* ( Brewer's yeast)  
<https://pid.depositar.io/ark:37281/k56253r3p>
- ex2: 台北市金價與結婚人數分析  
<https://pid.depositar.io/ark:37281/k5b8f3n3j>

webpage link: <https://hutakihare.github.io/2024AS-intern-snakemake/>

github link: <https://github.com/HutakiHare/2024AS-intern-snakemake.git>