

Mugs Data Engineer internship

ASSIGNMENT

NAME- Khushi Tolani

EMAIL ID- [kkhushitolanii@gmail.com](mailto:kkhushitolanii@gmail.com)

PHONE NO.- 8318094789

```
!wget https://physionet.org/static/published-projects/mimic-iv-fhir-demo/mimic-iv-clinical-database-demo-on-fhir-2.0.zip
```

```
--2024-05-07 22:20:54-- https://physionet.org/static/published-projects/mimic-iv-fhir-demo/mimic-iv-clinical-database-demo-on-fhir-2.0.
Resolving physionet.org (physionet.org)... 18.13.52.205
Connecting to physionet.org (physionet.org)|18.13.52.205|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 121863908 (116M) [application/zip]
Saving to: 'mimic-iv-clinical-database-demo-on-fhir-2.0.zip'

mimic-iv-clinical-d 100%[=====] 116.22M  269KB/s   in 7m 25s

2024-05-07 22:28:20 (267 KB/s) - 'mimic-iv-clinical-database-demo-on-fhir-2.0.zip' saved [121863908/121863908]
```

```
!unzip mimic-iv-clinical-database-demo-on-fhir-2.0.zip -d mimic_fhir_dataset
```

```
Archive: mimic-iv-clinical-database-demo-on-fhir-2.0.zip
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/LICENSE.txt
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/SHA256SUMS.txt
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Condition.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Encounter.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/EncounterICU.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Location.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Medication.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/MedicationAdministration.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/MedicationAdministrationICU.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/MedicationDispense.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/MedicationRequest.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationChartevents.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationDatetimeevents.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationLabevents.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationMicroOrg.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationMicroSusc.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationMicroTest.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ObservationOutputevents.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Organization.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Patient.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Procedure.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/ProcedureICU.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/Specimen.ndjson
  inflating: mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/SpecimenLab.ndjson
```

```
import json
import pandas as pd
import os
```

```
def read_ndjson(file_path):
    data = []
    with open(file_path, 'r') as file:
        for line in file:
            data.append(json.loads(line))
    return data
```

```
dataset_dir = '/content/mimic_fhir_dataset/mimic-iv-clinical-database-demo-on-fhir-2.0/mimic-fhir/'
```

```
patients = read_ndjson(dataset_dir + 'Patient.ndjson')
```

```
conditions = read_ndjson(dataset_dir + 'Condition.ndjson')
```

```
encounters = read_ndjson(dataset_dir + 'Encounter.ndjson')
```

```
encounters_icu = read_ndjson(dataset_dir + 'EncounterICU.ndjson')
```

```
from datetime import datetime
import csv
```

```
def get_condition_timestamp(condition, encounters_data):
    encounter_id = condition.get('encounter', {}).get('reference', '').split('/')[-1]
    for encounter in encounters_data:
        if encounter['id'] == encounter_id:
            start_time = encounter.get('period', {}).get('start')
            if start_time:
                timestamp = datetime.fromisoformat(start_time.replace('Z', '+00:00')).timestamp()
                return int(timestamp)
    return None
```

```
patient_conditions = {}
for condition in conditions:
    patient_id = condition.get('subject', {}).get('reference', '').split('/')[-1]
    if patient_id not in patient_conditions:
        patient_conditions[patient_id] = []
    condition_timestamp = get_condition_timestamp(condition, encounters)
    if condition_timestamp is not None:
        condition_data = {
            'pid': patient_id,
            'time': condition_timestamp,
            'code': condition.get('code', {}).get('coding', [{}])[0].get('code', ''),
            'description': condition.get('code', {}).get('coding', [{}])[0].get('display', '')
        }
        patient_conditions[patient_id].append(condition_data)
```

```
print("Step 2: Patient Conditions:")
for patient_id, conditions in patient_conditions.items():
    print(f"Patient ID: {patient_id}")
    for condition in conditions:
        print(condition)
```

```
Patient ID: 94abdf17-f13a-5eae-aac0-eca407bbfadd
{'pid': '94abdf17-f13a-5eae-aac0-eca407bbfadd', 'time': 6243609900, 'code': 'I214', 'description': 'Non-ST elevation (NSTEMI) myocardial infarction'}
{'pid': '94abdf17-f13a-5eae-aac0-eca407bbfadd', 'time': 6243609900, 'code': 'Z7982', 'description': 'Long term (current) use of aspirin'}
{'pid': '94abdf17-f13a-5eae-aac0-eca407bbfadd', 'time': 6243609900, 'code': 'Z7902', 'description': 'Long term (current) use of antiplatelet therapy'}
{'pid': '94abdf17-f13a-5eae-aac0-eca407bbfadd', 'time': 6243609900, 'code': 'I7102', 'description': 'Dissection of abdominal aorta'}
{'pid': '94abdf17-f13a-5eae-aac0-eca407bbfadd', 'time': 6243609900, 'code': 'Z23', 'description': 'Encounter for immunization'}
{'pid': '94abdf17-f13a-5eae-aac0-eca407bbfadd', 'time': 6243609900, 'code': 'K219', 'description': 'Gastro-esophageal reflux disease'}
Patient ID: 5ddeb201-5de6-5177-a116-fa82ce8ad2f2
{'pid': '5ddeb201-5de6-5177-a116-fa82ce8ad2f2', 'time': 5545862220, 'code': 'Z85850', 'description': 'Personal history of malignant neoplasm of thyroid gland'}
{'pid': '5ddeb201-5de6-5177-a116-fa82ce8ad2f2', 'time': 5545862220, 'code': 'G40909', 'description': 'Epilepsy, unspecified, not intractable, without current drug therapy'}
{'pid': '5ddeb201-5de6-5177-a116-fa82ce8ad2f2', 'time': 5545862220, 'code': 'D1802', 'description': 'Hemangioma of intracranial structures'}
```

```
csv_filename = 'patient_conditions.csv'
csv_columns = ['pid', 'time', 'code', 'description']
with open(csv_filename, 'w', newline='') as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=csv_columns)
    writer.writeheader()
    for patient_id, conditions in patient_conditions.items():
        for condition in conditions:
            writer.writerow(condition)
```

```
print("CSV File Generated:", csv_filename)
```

CSV File Generated: patient\_conditions.csv

```
import pandas as pd

df = pd.read_csv(csv_filename)
df
```

	pid	time	code	description
0	b410dd44-7d65-56f9-974f-2751e8aa80e2	5616018000	Z8546	Personal history of malignant neoplasm of pros...
1	b410dd44-7d65-56f9-974f-2751e8aa80e2	5387190060	V1072	Personal history of hodgkin's disease
2	b410dd44-7d65-56f9-974f-2751e8aa80e2	5639393940	Z7902	Long term (current) use of antithrombotics/ant...
3	b410dd44-7d65-56f9-974f-2751e8aa80e2	5426582400	49390	Asthma, unspecified type, unspecified
4	b410dd44-7d65-56f9-974f-2751e8aa80e2	5426582400	2724	Other and unspecified hyperlipidemia
...	...	...	...	...
4176	94abdf17-f13a-5eae-aac0-eca407bbfadd	6243609900	Z23	Encounter for immunization
4177	94abdf17-f13a-5eae-aac0-eca407bbfadd	6243609900	K219	Gastro-esophageal reflux disease without esoph...
4178	5ddeb201-5de6-5177-a116-fa82ce8ad2f2	5545862220	Z85850	Personal history of malignant neoplasm of thyroid
4179	5ddeb201-5de6-5177-a116-fa82ce8ad2f2	5545862220	G40909	Epilepsy, unspecified, not intractable, withou...
4180	5ddeb201-5de6-5177-a116-fa82ce8ad2f2	5545862220	D1802	Hemangioma of intracranial structures

4181 rows x 4 columns

Next steps: [Generate code with df](#) [View recommended plots](#)

```
df.head()
```

	pid	time	code	description
0	b410dd44-7d65-56f9-974f-2751e8aa80e2	5616018000	Z8546	Personal history of malignant neoplasm of pros...
1	b410dd44-7d65-56f9-974f-2751e8aa80e2	5387190060	V1072	Personal history of hodgkin's disease
2	b410dd44-7d65-56f9-974f-2751e8aa80e2	5639393940	Z7902	Long term (current) use of antithrombotics/ant...
3	b410dd44-7d65-56f9-974f-2751e8aa80e2	5426582400	49390	Asthma, unspecified type, unspecified
4	b410dd44-7d65-56f9-974f-2751e8aa80e2	5426582400	2724	Other and unspecified hyperlipidemia

Next steps: [Generate code with df](#) [View recommended plots](#)

```
df.tail()
```

	pid	time	code	description
4176	94abdf17-f13a-5eae-aac0-eca407bbfadd	6243609900	Z23	Encounter for immunization
4177	94abdf17-f13a-5eae-aac0-eca407bbfadd	6243609900	K219	Gastro-esophageal reflux disease without esoph...
4178	5ddeb201-5de6-5177-a116-fa82ce8ad2f2	5545862220	Z85850	Personal history of malignant neoplasm of thyroid
4179	5ddeb201-5de6-5177-a116-fa82ce8ad2f2	5545862220	G40909	Epilepsy, unspecified, not intractable, withou...
4180	5ddeb201-5de6-5177-a116-fa82ce8ad2f2	5545862220	D1802	Hemangioma of intracranial structures

df.shape

(4181, 4)

df.info

`pandas.core.frame.DataFrame.info`  
def info(verbose: bool | None=None, buf: WriteBuffer[str] | None=None, max\_cols: int | None=None, memory\_usage: bool | str | None=None, show\_counts: bool | None=None) -> None

[/usr/local/lib/python3.10/dist-packages/pandas/core/frame.py](#)  
Print a concise summary of a DataFrame.

This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.

from google.colab import files

files.download(csv\_filename)  
print("Successfully downloaded the csv file :", csv\_filename)

Successfully downloaded the csv file : patient\_conditions.csv