ARCOS Data with Dask

Exercise 1

Here, I downloaded the 2GB zipped ARCOS data

Exercise 2

In [ ]:

```
import pandas as pd
import numpy as np
import zipfile
```

```
c:\Users\kbagh\miniconda3\Lib\site-packages\numpy\_distributor_init.py:30: UserWarni
ng: loaded more than 1 DLL from .libs:
c:\Users\kbagh\miniconda3\Lib\site-packages\numpy\.libs\libopenblas64__v0.3.21-gcc_1
0_3_0.dll
c:\Users\kbagh\miniconda3\Lib\site-packages\numpy\.libs\libopenblas64__v0.3.23-246-g
3d31191b-gcc_10_3_0.dll
  warnings.warn("loaded more than 1 DLL from .libs:"
```

In [ ]:

```
zip_file_name = "arcos_2011_2012.tsv.zip"
tsv_file_name = "arcos_2011_2012.tsv"

with zipfile.ZipFile(zip_file_name, "r") as zip_file:
    with zip_file.open(tsv_file_name) as tsv_file:
        pd_df = pd.read_csv(tsv_file, sep="\t", nrows=100_000)

print(
    "The following is a sample of the first 100,000 rows of the dataset, read in wi
)
pd_df.sample(10)
```

```
The following is a sample of the first 100,000 rows of the dataset, read in with Pan
das.
```

```
C:\Users\kbagh\AppData\Local\Temp\ipykernel_8948\1164597319.py:6: DtypeWarning: Colu
mns (4,6,27) have mixed types. Specify dtype option on import or set low_memory=Fals
e.
  pd_df = pd.read_csv(tsv_file, sep="\t", nrows=100_000)
```

Out[ ]:

	Unnamed: 0	REPORTER_DEA_NO	REPORTER_BUS_ACT	REPORTER_NAME	REPORT
13371	47799	PD0029567	DISTRIBUTOR	MCKESSON CORPORATION	
38079	1315	PG0149650	DISTRIBUTOR	AMERICAN SALES COMPANY	
44982	42312	PK0070297	DISTRIBUTOR	KINRAY INC	
98055	62560	PM0018425	DISTRIBUTOR	MCKESSON CORPORATION	
54233	77813	PL0032627	DISTRIBUTOR	AMERISOURCEBERGEN DRUG CORP	
42477	19134	PK0070297	DISTRIBUTOR	KINRAY INC	
63506	106750	PM0000771	DISTRIBUTOR	MCKESSON CORPORATION	
17120	62099	PD0029567	DISTRIBUTOR	MCKESSON CORPORATION	
11671	41422	PC0003044	DISTRIBUTOR	CARDINAL HEALTH 110, LLC	
91872	41432	PM0003094	DISTRIBUTOR	MCKESSON CORPORATION	

10 rows × 45 columns

In [ ]:

```
pd_df["total_opioids"] = (pd_df["MME_Conversion_Factor"] * 1000) * pd_df[
    "CALC_BASE_WT_IN_GM"
]
print("The following is an output on the highest values per company")
pd_df.groupby(["REPORTER_NAME"])["total_opioids"].sum().sort_values(
    ascending=False
)
```

```
The following is an output on the highest values per company
```

```
Out[ ]: REPORTER_NAME
        MCKESSON CORPORATION                 2.992663e+08
        CARDINAL HEALTH 110, LLC             5.435232e+07
        AMERISOURCEBERGEN DRUG CORP          3.456139e+07
        KINRAY INC                           2.862032e+07
        LOUISIANA WHOLESALE DRUG CO          1.478777e+07
        FRANK W KERR INC                     8.730016e+06
        H D SMITH WHOLESALE DRUG CO          6.399324e+06
        KAISER FOUNDATION HOSPITALS          3.891330e+06
        BURLINGTON DRUG COMPANY              3.889490e+06
        AMERICAN SALES COMPANY               3.432058e+06
        DIK DRUG CO                          3.278514e+06
        KPH HEALTHCARE SERVICES, INC.        1.988890e+06
        BLOODWORTH WHOLESALE DRUGS           1.782827e+06
        DISCOUNT DRUG MART                   1.272596e+06
        KAISER FNDTN HEALTH PLAN NW          1.159892e+06
        H J HARKINS COMPANY INC              3.523940e+05
        CAPITAL WHOLESALE DRUG & CO          1.883182e+05
        APOTHECA INC                         2.391330e+04
        HALS MED DENT SUPPLY CO., INC.       1.816200e+04
        CESAR CASTILLO INC                   3.027000e+03
        MERRITT VETERINARY SUPPLIES INC      1.816200e+03
        ACE SURGICAL SUPPLY CO INC           6.054000e+02
        Name: total_opioids, dtype: float64
```

From the input above, we can conclude that McKesson Corporation has shipped the most opioids within the timeframe of the dataset, and only within the first 10,000 rows

Exercise 3

```
In [ ]: import os

        print(f"I have {os.cpu_count()} logical cores.")
```

```
I have 8 logical cores.
```

```
In [ ]: from dask.distributed import Client

        client = Client()
        client
```

Out[ ]:

Client

Client-98a5d87d-938c-11ee-a2f4-204ef6e641b6

Connection method: Cluster object **Cluster type:** distributed.LocalCluster

Dashboard: http://127.0.0.1:8787/status

▸ Cluster Info

```
In [ ]: import dask.dataframe as dd
```

```
c:\Users\kbagh\miniconda3\Lib\site-packages\dask\dataframe\_pyarrow_compat.py:17: Fu
tureWarning: Minimal version of pyarrow will soon be increased to 14.0.1. You are us
ing 13.0.0. Please consider upgrading.
  warnings.warn(
```

In [ ]:

```
df = dd.read_csv(
    "arcos_2011_2012.tsv",
    sep="\t",
    dtype={
        "Unnamed : 0": "object",
        "REPORTED_DEA_NO": "object",
        "REPORTER_BUS_ACT": "object",
        "REPORTER_NAME": "object",
        "REPORTER_ADDL_CO_INFO": "object",
        "REPORTER_ADDRESS1": "object",
        "REPORTER_ADDRESS2": "object",
        "REPORTER_CITY": "object",
        "REPORTER_STATE": "object",
        "REPORTER_ZIP": "object",
        "REPORTER_COUNTY": "object",
        "BUYER_DEA_NO": "object",
        "BUYER_BUS_ACT": "object",
        "BUYER_NAME": "object",
        "BUYER_ADDL_CO_INFO": "object",
        "BUYER_ADDRESS1": "object",
        "BUYER_ADDRESS2": "object",
        "BUYER_CITY": "object",
        "BUYER_STATE": "object",
        "BUYER_ZIP": "object",
        "BUYER_COUNTY": "object",
        "TRANSACTION_CODE": "object",
        "DRUG_CODE": "object",
        "NDC_NO": "object",
        "DRUG_NAME": "object",
        "QUANTITY": "float64",
        "UNIT": "object",
        "ACTION_INDICATOR": "object",
        "ORDER_FORM_NO": "object",
        "CORRECTION_NO": "object",
        "STRENGTH": "object",
        "TRANSACTION_DATE": "object",
        "CALC_BASE_WT_IN_GM": "float64",
        "DOSAGE_UNIT": "float64",
        "TRANSACTION_ID": "object",
        "Product_Name": "object",
        "Ingredient_Name": "object",
        "Measure": "object",
        "MME_Conversion_Factor": "float64",
        "Combined_Labeler_Name": "object",
        "Revised_Company_Name": "object",
        "Reporter_family": "object",
        "dos_str": "object",
        "date": "object",
        "year": "object",
    },
)
```

```
temp_df = df[["MME_Conversion_Factor","CALC_BASE_WT_IN_GM","REPORTER_NAME"]]

temp_df["total_opioids"] = (temp_df["MME_Conversion_Factor"] * 1000) * temp_df["CAL
answer = temp_df.groupby(["REPORTER_NAME"])["total_opioids"].sum().compute()
answer.sort_values(ascending=False)
```

Out[ ]:

```
REPORTER_NAME
MCKESSON CORPORATION                                5.604679e+10
CARDINAL HEALTH                                     4.671958e+10
WALGREEN CO                                         4.185033e+10
AMERISOURCEBERGEN DRUG CORP                         2.553364e+10
CARDINAL HEALTH 110, LLC                            5.896801e+09
                                                        ...
QUIQ, INC                                           5.327520e+02
SOUTHERN MEDICAL LASERS DBA SML MEDICAL SALES       4.540500e+02
GAVIS PHARMACEUTICALS, LLC                          3.027000e+02
REMEDYREPACK                                        1.816200e+02
MIKART                                              1.695120e+02
Name: total_opioids, Length: 316, dtype: float64
```

When using Dask, we can see that Mckesson Coporation was the company with the most sold, which is in-line with what was found earlier when only looking at the first 10,000 rows of data

Exercise 4

In [ ]:

```
ex4_df = df[["DOSAGE_UNIT","REPORTER_NAME","BUYER_STATE"]]

answer_4 = ex4_df.groupby(["BUYER_STATE","REPORTER_NAME"])["DOSAGE_UNIT"].sum().com
```

In [ ]:

```
only_state = answer_4.reset_index().groupby("BUYER_STATE").apply(lambda x: x.loc[x[
```

In [ ]:

```
print("The following below is a table, which contains the company with the highest
only_state
```

Out[ ]:

	BUYER_STATE	REPORTER_NAME	DOSAGE_UNIT
BUYER_STATE			
AK	AK	CARDINAL HEALTH	12912712.0
AL	AL	MCKESSON CORPORATION	210395190.0
AR	AR	AMERISOURCEBERGEN DRUG CORPORATION	57196800.0
AZ	AZ	WALGREEN CO	176419710.0
CA	CA	AMERISOURCEBERGEN DRUG CORP	449992280.0
CO	CO	MCKESSON CORPORATION	74987840.0
CT	CT	CARDINAL HEALTH	56635720.0
DC	DC	CARDINAL HEALTH	9694400.0
DE	DE	WALGREEN CO	29274900.0
FL	FL	WALGREEN CO	459455250.0
GA	GA	MCKESSON CORPORATION	127935540.0
GU	GU	AMERISOURCEBERGEN DRUG CORP	964500.0
HI	HI	AMERISOURCEBERGEN DRUG CORP	27102040.0
IA	IA	WALGREEN CO	40055380.0
ID	ID	MCKESSON CORPORATION	34168720.0
IL	IL	WALGREEN CO	265412740.0
IN	IN	CVS INDIANA	193518900.0
KS	KS	MCKESSON CORPORATION	76247270.0
KY	KY	AMERISOURCEBERGEN DRUG CORP	149117060.0
LA	LA	WALGREEN CO	91262050.0
MA	MA	CARDINAL HEALTH	132415210.0
MD	MD	MCKESSON CORPORATION	142428820.0
ME	ME	CARDINAL HEALTH	44604490.0
MI	MI	MCKESSON CORPORATION	196841400.0
MN	MN	MCKESSON DRUG COMPANY	77370860.0
MO	MO	WALGREEN CO	128879010.0
MP	MP	AMERISOURCEBERGEN DRUG CORP	287900.0
MS	MS	AMERISOURCEBERGEN DRUG CORP	65602720.0
MT	MT	MCKESSON CORPORATION	34330100.0

	BUYER_STATE	REPORTER_NAME	DOSAGE_UNIT
BUYER_STATE			
NC	NC	CARDINAL HEALTH	186727600.0
ND	ND	MCKESSON DRUG COMPANY	10297650.0
NE	NE	MCKESSON CORPORATION	31205240.0
NH	NH	MCKESSON CORPORATION	21763450.0
NJ	NJ	MCKESSON CORPORATION	98708550.0
NM	NM	WALGREEN ARIZONA DRUG CO	31751330.0
NV	NV	WALGREEN CO	96501610.0
NY	NY	CARDINAL HEALTH 110, LLC	259680450.0
OH	OH	CARDINAL HEALTH	238501750.0
OK	OK	MCKESSON CORPORATION	121119950.0
OR	OR	MCKESSON CORPORATION	129109660.0
PA	PA	MCKESSON CORPORATION	250514560.0
PR	PR	DROGUERIA BETANCES, LLC	11419040.0
RI	RI	CARDINAL HEALTH	19344120.0
SC	SC	MCKESSON CORPORATION	237201700.0
SD	SD	MCKESSON CORPORATION	13952560.0
TN	TN	WALGREEN CO	131097140.0
TX	TX	WALGREEN CO	376538690.0
UT	UT	AMERISOURCEBERGEN DRUG CORP	58896360.0
VA	VA	CARDINAL HEALTH	113905462.0
VI	VI	CARDINAL HEALTH P.R. 120, INC.	622220.0
VT	VT	MCKESSON CORPORATION	12777210.0
WA	WA	MCKESSON CORPORATION	169188860.0
WI	WI	WALGREEN CO	171923190.0
WV	WV	CARDINAL HEALTH	70562930.0
WY	WY	MCKESSON CORPORATION	10089750.0

The following above is the reporter of the highest pill distriubtions per each state. While there is no one company that reigns supreme in for causing the opioid epidemic, it seems that there are a few distributors that were very involved.

Exercise 5

In [ ]:

```
ex5_df = df[["BUYER_STATE","BUYER_COUNTY","year","CALC_BASE_WT_IN_GM", "MME_Convers
ex5_df["total_opioid"] = (ex5_df["MME_Conversion_Factor"] * 1000) * ex5_df["CALC_BA

answer_5 = ex5_df.groupby(["BUYER_COUNTY","BUYER_STATE","year"])[["total_opioid"]].
```

In [ ]:

```
print("The following below shows the total morphine equivalent sent to each county
answer_5.reset_index()
```

```
The following below shows the total morphine equivalent sent to each county in the U
S during the timeframe of our project.
```

Out[ ]:

	BUYER_COUNTY	BUYER_STATE	year	total_opioid
0	ACADIA	LA	2011	3.254470e+07
1	ACADIA	LA	2012	2.702122e+07
2	ACCOMACK	VA	2011	8.488628e+06
3	ACCOMACK	VA	2012	7.968890e+06
4	ADA	ID	2011	1.660791e+08
...	...	...	...	...
6168	ARCHER	TX	2012	3.027000e+02
6169	OLDHAM	TX	2012	6.054000e+02
6170	DENALI	AK	2012	3.027000e+03
6171	THROCKMORTON	TX	2012	1.210800e+02
6172	ALPINE	CA	2012	3.027000e+02

6173 rows × 4 columns

Exercise 6

The following below is a link to my branch that accomplishes the Dask task for the entire arcos dataset

https://github.com/MIDS-at-Duke/opioid-2023-team-reps/tree/kian_dask