NAME: ASIF ERFAN KHAN

ROLL NUMBER: 546

COURSE: MSc CS

SUBJECT: FUNDAMENTALS OF DATA SCIENCE

TOPIC: PRACTICAL 1 DATA COLLECTION, MODELLING AND COMPILATION

Data Collection: Data collection is defined as the procedure of collecting, measuring and analyzing accurate insights for research using standard validated techniques. A researcher can evaluate their hypothesis on the basis of collected data. In most cases, data collection is the primary and most important step for research, irrespective of the field of research. The approach of data collection is different for different fields of study, depending on the required information. The most critical objective of data collection is ensuring that information-rich and reliable data is collected for statistical analysis so that data-driven decisions can be made for research.

Data Collection and Datasets

From .csv Files From Excel Files From SQL Files

```
my_dict={'Name':["a","b","c","d","e","f","g"],
        'age':[20,27,35,45,55,43,35],
        'designation':["VP","CEO","CFO","VP","VP","CEO","MD"]}

import pandas as pd
import numpy as np
df=pd.DataFrame(my_dict)
df
```

	Name	age	designation
0	a	20	VP
1	b	27	CEO
2	c	35	CFO
3	d	45	VP
4	e	55	VP
5	f	43	CEO
6	g	35	MD

	Name	age	designation
0	a	20	VP
1	b	27	CEO

```
df_csv=pd.read_csv('Csv example')
df_csv
```

	Unnamed: 0	Name	age	designation
0	0	a	20	VP
1	1	b	27	CEO
2	2	c	35	CFO
3	3	d	45	VP
4	4	e	55	VP
5	5	f	43	CEO
6	6	g	35	MD

```
df.to_csv('CSV Ex',index=False)
df_csv=pd.read_csv('CSV Ex')
df_csv
```

	Name	age	designation
0	a	20	VP
1	b	27	CEO
2	c	35	CFO
3	d	45	VP
4	e	55	VP
5	f	43	CEO

	0	1	2	3	4	5	6	7	8	9	...	23
0	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel
1	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4
2	GP	F	17	U	GT3	T	1	1	at_home	other	...	5
3	GP	F	15	U	LE3	T	1	1	at_home	other	...	4

```
import pandas as pd
Location = "/content/drive/MyDrive/Colab Notebooks/student-mat.csv"
df = pd.read_csv(Location)
df.head()
```

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famre
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	
3	GP	F	15	U	GT3	T	4	2	health	services	...	
4	GP	F	16	U	GT3	T	3	3	other	other	...	

5 rows × 33 columns

```
import pandas as pd
Location = "/content/drive/MyDrive/Colab Notebooks/student-mat.csv"
# To add headers as we load the data...
df = pd.read_csv(Location, names=['RollNo','Names','Grades'])
# To add headers to a dataframe
df.columns = ['RollNo','Names','Grades']
df.head()
```

school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	reason	guardia

```
bsdegrees = [1,1,0,0,1]
msdegrees = [2,1,0,0,0]
phddegrees = [0,1,0,0,0]
Degrees = zip(names,grades,bsdegrees,msdegrees,phddegrees)
columns = ['Names','Grades','BS','MS','PhD']
df = pd.DataFrame(data = Degrees, columns=columns)
df
```

	Names	Grades	BS	MS	PhD
0	Bob	76	1	2	0
1	Jessica	95	1	1	1
2	Mary	77	0	0	0
3	John	78	0	0	0
4	Mel	99	1	0	0

```
import pandas as pd
Location = "/content/drive/MyDrive/Colab Notebooks/gradedata.xlsx"
df = pd.read_excel(Location)

#Changing column Names
df.columns = ['first','last','sex','age','exer','hrs','grd','addr']
df.head()
```

	first	last	sex	age	exer	hrs	grd	addr
0	Marcia	Pugh	female	17	3	10	82.4	7379 Highland Rd. , Dublin, GA 31021
1	Kadeem	Morrison	male	18	4	4	78.2	8 Bayport St. , Honolulu, HI 96815
2	Nash	Powell	male	18	5	9	79.3	Encino, CA 91316, 3 Lilac Street
3	Noelani	Wagner	female	14	2	7	83.2	Riverview, FL 33569, 9998 North Smith Dr.
4	Noelani	Cherry	female	18	4	15	87.4	97 SE. Ocean Street , Bethlehem, PA 18015

```
names = ['Bob','Jessica','Mary','John','Mel']
grades = [76,95,77,78,99]
GradeList = zip(names,grades)
df = pd.DataFrame(data = GradeList,columns=['Names','Grades'])

writer = pd.ExcelWriter('dataframe.xlsx', engine='xlsxwriter')
df.to_excel(writer, sheet_name='Sheet1')
writer.save()
```

```
import sqlite3
con = sqlite3.connect("/content/drive/MyDrive/Colab Notebooks/portal_mammals.sqlite")
cur = con.cursor()


for row in cur.execute('SELECT * FROM species;'):
    print(row)

con.close()
```

```
('AB', 'Amphispiza', 'bilineata', 'Bird')
('AH', 'Ammospermophilus', 'harrisi', 'Rodent')
('AS', 'Ammodramus', 'savannarum', 'Bird')
('BA', 'Baiomys', 'taylori', 'Rodent')
('CB', 'Campylorhynchus', 'brunneicapillus', 'Bird')
('CM', 'Calamospiza', 'melanocorys', 'Bird')
('CQ', 'Callipepla', 'squamata', 'Bird')
('CS', 'Crotalus', 'scutalatus', 'Reptile')
('CT', 'Cnemidophorus', 'tigris', 'Reptile')
('CU', 'Cnemidophorus', 'uniparens', 'Reptile')
('CV', 'Crotalus', 'viridis', 'Reptile')
('DM', 'Dipodomys', 'merriami', 'Rodent')
('DO', 'Dipodomys', 'ordii', 'Rodent')
('DS', 'Dipodomys', 'spectabilis', 'Rodent')
('DX', 'Dipodomys', 'sp.', 'Rodent')
('EO', 'Eumeces', 'obsoletus', 'Reptile')
('GS', 'Gambelia', 'silus', 'Reptile')
('NL', 'Neotoma', 'albigula', 'Rodent')
('NX', 'Neotoma', 'sp.', 'Rodent')
('OL', 'Onychomys', 'leucogaster', 'Rodent')
('OT', 'Onychomys', 'torridus', 'Rodent')
('OX', 'Onychomys', 'sp.', 'Rodent')
```

```
('RM', 'Reithrodontomys', 'megalotis', 'Rodent')
('RO', 'Reithrodontomys', 'montanus', 'Rodent')
('RX', 'Reithrodontomys', 'sp.', 'Rodent')
('SA', 'Sylvilagus', 'audubonii', 'Rabbit')
('SB', 'Spizella', 'breweri', 'Bird')
('SC', 'Sceloporus', 'clarki', 'Reptile')
('SF', 'Sigmodon', 'fulviventer', 'Rodent')
('SH', 'Sigmodon', 'hispidus', 'Rodent')
('SO', 'Sigmodon', 'ochrognathus', 'Rodent')
('SS', 'Spermophilus', 'spilosoma', 'Rodent')
('ST', 'Spermophilus', 'tereticaudus', 'Rodent')
('SU', 'Sceloporus', 'undulatus', 'Reptile')
('SX', 'Sigmodon', 'sp.', 'Rodent')
('UL', 'Lizard', 'sp.', 'Reptile')
('UP', 'Pipilo', 'sp.', 'Bird')
('UR', 'Rodent', 'sp.', 'Rodent')
('US', 'Sparrow', 'sp.', 'Bird')
('ZL', 'Zonotrichia', 'leucophrys', 'Bird')
('ZM', 'Zenaida', 'macroura', 'Bird')
```

```
import sqlite3

# Create a SQL connection to our SQLite database
con = sqlite3.connect("/content/drive/MyDrive/Colab Notebooks/portal_mammals.sqlite")

cur = con.cursor()

# Return all results of query
cur.execute('SELECT plot_id FROM plots WHERE plot_type="Control"')
print(cur.fetchall())

# Return first result of query
cur.execute('SELECT species FROM species WHERE taxa="Bird"')
print(cur.fetchone())

# Be sure to close the connection
con.close()
```

```
[(2,), (4,), (8,), (11,), (12,), (14,), (17,), (22,)]
('bilineata',)
```

```
   record_id  month  day  year  plot_id species_id sex  hindfoot_length  \
0          1      7   16  1977        2         NL   M             32.0
1          2      7   16  1977        3         NL   M             33.0
2          3      7   16  1977        2         DM   F             37.0
3          4      7   16  1977        7         DM   M             36.0
4          5      7   16  1977        3         DM   M             35.0

   weight
0     NaN
1     NaN
2     NaN
3     NaN
4     NaN
```

```
from pandas import DataFrame
Cars={'Brand':['Honda Civic','Toyota Corolla','Ford Focus','Audi A4'],
      'Price':[22000,25000,27000,35000]
      }
df=DataFrame(Cars,columns=['Brand','Price'])
print(df)
```

```
              Brand  Price
0     Honda Civic  22000
1  Toyota Corolla  25000
2      Ford Focus  27000
3         Audi A4  35000
```

```
import sqlite3
conn=sqlite3.connect('TestDB1.db')
c=conn.cursor()
```

```
c.execute('CREATE TABLE CARS2(Brand text, Price number)')
conn.commit()
```

```
df.to_sql('CARS2',conn,if_exists='replace',index=False)
df
```

```
''')
```

```
<sqlite3.Cursor at 0x7f39bd9e1ce0>
```

```
df=DataFrame(c.fetchall(),columns=['Brand','Price'])
df
```

	Brand	Price
0	Audi A4	35000

Example1

```
import pandas as pd
import os
import sqlite3 as lite
from sqlalchemy import create_engine
```

```
studentId=["rj101","rj150","rj134","rj70"]
SName=["Saurabh","Giftson","Vikas","Radha"]
LName=["Chavan","Paul","Bisoi","Rai"]
Department=["Bms","Bcom","BscCS","BScIT"]
Email=["100rabh@gmail.com","gift01@gmail.com","vik21@gmail.com","rad01@gmail.com"]
```

```
studata = zip(studentId,SName,LName,Department,Email)
```

```
df = pd.DataFrame(data =studata, columns=['StudentId','SName','LName','Department','Email'])
df
```

	StudentId	SName	LName	Department	Email
0	rj101	Saurabh	Chavan	Bms	100rabh@gmail.com
1	rj150	Giftson	Paul	Bcom	gift01@gmail.com
2	rj134	Vikas	Bisoi	BscCS	vik21@gmail.com

```
db_filename = r'studentdata.db'
con = lite.connect(db_filename)
df.to_sql('student',
con,
schema=None,
if_exists='replace',
index=True,
index_label=None,
chunksize=None,
dtype=None)
con.close()

db_file = r'studentdata.db'
engine = create_engine(r"sqlite:///{}" .format(db_file))
sql = 'SELECT * from student '


studf = pd.read_sql(sql, engine)
studf
```

	index	StudentId	SName	LName	Department	Email
0	0	rj101	Saurabh	Chavan	Bms	100rabh@gmail.com
1	1	rj150	Giftson	Paul	Bcom	gift01@gmail.com
2	2	rj134	Vikas	Bisoi	BscCS	vik21@gmail.com
3	3	rj70	Radha	Rai	BScIT	rad01@gmail.com

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

```
state=pd.read_csv("/content/drive/MyDrive/Colab Notebooks/US_violent_crime.csv")
state.head()
```

	State	Murder	Assault	UrbanPop	Rape

```
  return x*2
state.apply(some_func) #update each entry of dataframe without any loop
state.apply(lambda n: n*2) #lambda also works the same
```

	State	Murder	Assault	UrbanPop	Rape
0	AlabamaAlabama	26.4	472	116	42.4
1	AlaskaAlaska	20.0	526	96	89.0
2	ArizonaArizona	16.2	588	160	62.0
3	ArkansasArkansas	17.6	380	100	39.0
4	CaliforniaCalifornia	18.0	552	182	81.2
5	ColoradoColorado	15.8	408	156	77.4
6	ConnecticutConnecticut	6.6	220	154	22.2
7	DelawareDelaware	11.8	476	144	31.6
8	FloridaFlorida	30.8	670	160	63.8
9	GeorgiaGeorgia	34.8	422	120	51.6
10	HawaiiHawaii	10.6	92	166	40.4
11	IdahoIdaho	5.2	240	108	28.4
12	IllinoisIllinois	20.8	498	166	48.0
13	IndianaIndiana	14.4	226	130	42.0
14	IowaIowa	4.4	112	114	22.6
15	KansasKansas	12.0	230	132	36.0
16	KentuckyKentucky	19.4	218	104	32.6
17	LouisianaLouisiana	30.8	498	132	44.4
18	MaineMaine	4.2	166	102	15.6
19	MarylandMaryland	22.6	600	134	55.6
20	MassachusettsMassachusetts	8.8	298	170	32.6
21	MichiganMichigan	24.2	510	148	70.2

```
state.transform(func = lambda x : x * 10)
```

	State	Murder	Assault	UrbanPop	R
0	AlabamaAlabamaAlabamaAlabamaAlabamaAlabamaAlab...	132.0	2360	580	2
1	AlaskaAlaskaAlaskaAlaskaAlaskaAlaskaAlaskaAlas...	100.0	2630	480	4
2	ArizonaArizonaArizonaArizonaArizonaArizonaAriz...	81.0	2940	800	3
3	ArkansasArkansasArkansasArkansasArkansasArkans...	88.0	1900	500	1
4	CaliforniaCaliforniaCaliforniaCaliforniaCalifo...	90.0	2760	910	4
5	ColoradoColoradoColoradoColoradoColoradoColora...	79.0	2040	780	3
6	ConnecticutConnecticutConnecticutConnecticutCo...	33.0	1100	770	1
7	DelawareDelawareDelawareDelawareDelawareDelawa...	59.0	2380	720	1
8	FloridaFloridaFloridaFloridaFloridaFloridaFlor...	154.0	3350	800	3
9	GeorgiaGeorgiaGeorgiaGeorgiaGeorgiaGeorgiaGeor...	174.0	2110	600	2
10	HawaiiHawaiiHawaiiHawaiiHawaiiHawaiiHawaiiHawa...	53.0	460	830	2
11	IdahoIdahoIdahoIdahoIdahoIdahoIdahoIdahoIdahoI...	26.0	1200	540	1
12	IllinoisIllinoisIllinoisIllinoisIllinoisIllino...	104.0	2490	830	2
13	IndianaIndianaIndianaIndianaIndianaIndianaIndi...	72.0	1130	650	2
14	IowaIowaIowaIowaIowaIowaIowaIowaIowaIowa	22.0	560	570	1
15	KansasKansasKansasKansasKansasKansasKansasKans...	60.0	1150	660	1
16	KentuckyKentuckyKentuckyKentuckyKentuckyKentuc...	97.0	1090	520	1
17	LouisianaLouisianaLouisianaLouisianaLouisianaL...	154.0	2490	660	2
18	MaineMaineMaineMaineMaineMaineMaineMaineMaineM...	21.0	830	510	
19	MarylandMarylandMarylandMarylandMarylandMaryla...	113.0	3000	670	2
20	MassachusettsMassachusettsMassachusettsMassach...	44.0	1490	850	1
21	MichiganMichiganMichiganMichiganMichiganMichig...	121.0	2550	740	3
22	MinnesotaMinnesotaMinnesotaMinnesotaMinnesotaM...	27.0	720	660	1

30	New MexicoNew MexicoNew MexicoNew MexicoNew Me...	114.0	2850	700	3
31	New YorkNew YorkNew YorkNew YorkNew YorkNew Yo...	111.0	2540	860	2
32	North CarolinaNorth CarolinaNorth CarolinaNort...	130.0	3370	450	1
33	North DakotaNorth DakotaNorth DakotaNorth Dako...	8.0	450	440	
34	OhioOhioOhioOhioOhioOhioOhioOhioOhioOhio	73.0	1200	750	2
35	OklahomaOklahomaOklahomaOklahomaOklahomaOklaho...	66.0	1510	680	2
36	OregonOregonOregonOregonOregonOregonOregonOreg...	49.0	1590	670	2
37	PennsylvaniaPennsylvaniaPennsylvaniaPennsylvan...	63.0	1060	720	1
38	Rhode IslandRhode IslandRhode IslandRhode Isla...	34.0	1740	870	
39	South CarolinaSouth CarolinaSouth CarolinaSout...	144.0	2790	480	2
40	South DakotaSouth DakotaSouth DakotaSouth Dako...	38.0	860	450	1
41	TennesseeTennesseeTennesseeTennesseeTennesseeT...	132.0	1880	590	2
42	TexasTexasTexasTexasTexasTexasTexasTexasTexasT...	127.0	2010	800	2
43	UtahUtahUtahUtahUtahUtahUtahUtahUtahUtah	32.0	1200	800	2

```
#usinggroupby
mean_purchase =state.groupby('State')["Murder"].mean().rename("User_mean").reset_index()
print(mean_purchase)
```

```
          State  User_mean
0       Alabama       13.2
1        Alaska       10.0
2       Arizona        8.1
3      Arkansas        8.8
4    California        9.0
5      Colorado        7.9
6   Connecticut        3.3
7      Delaware        5.9
8       Florida       15.4
9       Georgia       17.4
10       Hawaii        5.3
11        Idaho        2.6
```

25	Montana	6.0
26	Nebraska	4.3
27	Nevada	12.2
28	New Hampshire	2.1
29	New Jersey	7.4
30	New Mexico	11.4
31	New York	11.1
32	North Carolina	13.0
33	North Dakota	0.8
34	Ohio	7.3
35	Oklahoma	6.6
36	Oregon	4.9
37	Pennsylvania	6.3
38	Rhode Island	3.4
39	South Carolina	14.4
40	South Dakota	3.8
41	Tennessee	13.2
42	Texas	12.7
43	Utah	3.2
44	Vermont	2.2
45	Virginia	8.5
46	Washington	4.0
47	West Virginia	5.7
48	Wisconsin	2.6
49	Wyoming	6.8

```
mer=state.merge(mean_purchase)
mer
```

	State	Murder	Assault	UrbanPop	Rape	User_mean
0	Alabama	13.2	236	58	21.2	13.2
1	Alaska	10.0	263	48	44.5	10.0
2	Arizona	8.1	294	80	31.0	8.1
3	Arkansas	8.8	190	50	19.5	8.8
4	California	9.0	276	91	40.6	9.0
5	Colorado	7.9	204	78	38.7	7.9
6	Connecticut	3.3	110	77	11.1	3.3
7	Delaware	5.9	238	72	15.8	5.9
8	Florida	15.4	335	80	31.9	15.4
9	Georgia	17.4	211	60	25.8	17.4
10	Hawaii	5.3	46	83	20.2	5.3
11	Idaho	2.6	120	54	14.2	2.6
12	Illinois	10.4	249	83	24.0	10.4
13	Indiana	7.2	113	65	21.0	7.2
14	Iowa	2.2	56	57	11.3	2.2
15	Kansas	6.0	115	66	18.0	6.0
16	Kentucky	9.7	109	52	16.3	9.7
17	Louisiana	15.4	249	66	22.2	15.4
18	Maine	2.1	83	51	7.8	2.1
19	Maryland	11.3	300	67	27.8	11.3
20	Massachusetts	4.4	149	85	16.3	4.4
21	Michigan	12.1	255	74	35.1	12.1
22	Minnesota	2.7	72	66	14.9	2.7

30	New Mexico	11.4	285	70	32.1	11.4
31	New York	11.1	254	86	26.1	11.1
32	North Carolina	13.0	337	45	16.1	13.0
33	North Dakota	0.8	45	44	7.3	0.8
34	Ohio	7.3	120	75	21.4	7.3
35	Oklahoma	6.6	151	68	20.0	6.6

```
#checking for missing values
print(state.isnull().sum())
```

```
State       0
Murder      0
Assault     0
UrbanPop    0
Rape        0
dtype: int64
```

42	Texas	12.7	201	80	25.5	12.7

EXAMPLE2

44	Vermont	2.2	48	32	11.2	2.2

```
import pandas as pd
import numpy as np
cols=['col0', 'col1', 'col2', 'col3', 'col4']
rows=['row0', 'row1', 'row2', 'row3', 'row4']
data=np.random.randint(0, 100, size=(5,5))
df=pd.DataFrame(data, columns=cols, index=rows)
df.head()
```

	col0	col1	col2	col3	col4
row0	23	19	47	30	65
row1	85	4	34	64	33
row2	98	14	4	40	11
row3	34	12	42	22	28

```
df['col5']=0
df['col6']=np.nan
df.head()
```

	col0	col1	col2	col3	col4	col5	col6
row0	23.0	19	47.0	30	65	0	NaN
row1	85.0	4	NaN	64	33	0	NaN
row2	98.0	14	4.0	40	11	0	NaN
row3	34.0	12	42.0	0	28	0	NaN
row4	NaN	52	57.0	64	9	0	NaN

```
df.loc[:,df.all()]
```

	col0	col1	col2	col4	col6
row0	23.0	19	47.0	65	NaN
row1	85.0	4	NaN	33	NaN
row2	98.0	14	4.0	11	NaN
row3	34.0	12	42.0	28	NaN
row4	NaN	52	57.0	9	NaN

```
df.loc[:,df.any()]
```

	col0	col1	col2	col3	col4
row0	23.0	19	47.0	30	65
row1	85.0	4	NaN	64	33
row2	98.0	14	4.0	40	11
row3	34.0	12	42.0	0	28

	col0	col2	col6
row0	23.0	47.0	NaN
row1	85.0	NaN	NaN

```
df.loc[:,df.notnull().all()]
```

	col1	col3	col4	col5
row0	19	30	65	0
row1	4	64	33	0
row2	14	40	11	0
row3	12	0	28	0
row4	52	64	9	0

```
df.dropna(how="all",axis=0)
```

	col0	col1	col2	col3	col4	col5	col6
row0	23.0	19	47.0	30	65	0	NaN
row1	85.0	4	NaN	64	33	0	NaN
row2	98.0	14	4.0	40	11	0	NaN
row3	34.0	12	42.0	0	28	0	NaN
row4	NaN	52	57.0	64	9	0	NaN

```
df.fillna(df.sum())
```

	col0	col1	col2	col3	col4	col5	col6
row0	23.0	19	47.0	30	65	0	0.0
row1	85.0	4	150.0	64	33	0	0.0

```
    'C' : [random.choice(('a','b','c')) for i in range(1000000)],
    'A' : [random.randint(1,10) for i in range(1000000)],
    'B' : [random.randint(1,10) for i in range(1000000)]

})
data
```

	C	A	B
0	c	4	4
1	a	7	10
2	b	2	4
3	a	10	7
4	a	8	2
...	...	...	...
999995	a	1	9
999996	a	7	9
999997	a	3	4
999998	c	5	9
999999	a	9	9

1000000 rows × 3 columns

```
v=data.groupby('C')["A"].mean
v
```

```
<bound method GroupBy.mean of <pandas.core.groupby.generic.SeriesGroupBy object at
0x7f39ba052b90>>
```

```
mean=data.groupby('C')["A"].mean().rename("D").reset_index()
mean
```

	C	A	B	D
0	c	4	4	5.498086
1	c	3	4	5.498086
2	c	5	10	5.498086
3	c	3	3	5.498086
4	c	9	6	5.498086
...	...	...	...	...
999995	b	2	3	5.495739
999996	b	3	8	5.495739
999997	b	10	10	5.495739
999998	b	10	6	5.495739
999999	b	10	5	5.495739

1000000 rows × 4 columns