1. markdown :Esc M
2. code :Esc y
3. for pupup: .tab
4. help : shift tab
5. data.read\_csv / data.read\_excel to read the files
6. data.head() : displays the first five records
7. data.tail() : displays the last five records
8. data.head(2)/data.tail(2) displays first 2 / last 2 records
9. data.columns : displays the columns in the file
10. list(data.columns) : displays columns in the list format
11. data.info() : displays the fields and datatypes
12. data.describe() : displays count, std, mean, max, min, 25%, 50%, 75% for integer columns and for string columns it shows count, unique, top, freq

**Day :6**

1. data['column name']: displays records of the mentioned column
2. data[['address\_2', 'city', 'state\_province']] : you can give more than one column in a list
3. data[['address\_2', 'city', 'state\_province']].head() : displays first five rows of the mentioned columns
4. data1 = pd.read\_csv("addresses.csv", header = None, names = [1,2,3,4,5,6,7,8]) : gives numbers in place of column names
5. data1.to\_excel("myAddress.xlsx", index = False) : copy data of data1. in excel file named myaddress.xlsx
6. data.shape : displays size as no. of rows and columns
7. data.dtypes : displays all column names and its datatype
8. data.dtypes[data.dtypes == 'int64'] : displays the columns having datatype int64
9. list(data.dtypes[data.dtypes == 'int64'].index) : displays the columns in a list format having datatype int64
10. to filter the records from table based on some condition

data[data['Embarked']=='S']

1. For filtering data ( using add)

filter1 = data['Survived'] == 0

filter2 = data['Sex'] == 'male'

data[filter1 & filter2]

**or**

data[(data['Survived'] == 0) & (data['Sex'] == 'male')]

1. To print the count of filtered data

data[filter1 & filter2].count()

1. To print the count of filtered data

len(data[filter1 & filter2])

1. For filtering data ( using or)

len(data[(data['Survived']==0) | (data['Sex']=='male')])

1. To find unique values

data['Embarked'].unique()

1. To find unique values in list format

list(data['Embarked'].unique())

1. To find the count of unique values for a column

data['Embarked'].value\_counts()

it gives the output as

Embarked

S 644

C 168

Q 77

1. Find the passenger paid max fair

data[data['Fare'] == max(data['Fare'])]

1. Find the passenger paid max fair

data[data['Fare']==min(data['Fare'])]

1. To find the null values count for a column

data[data['Cabin'].isnull()==False]

1. To create a new column

data['newColumn']='India'

1. Assigning a value of another column to the new column

data['newColumn2'] =data['Fare']

1. To drop a column

data = data.drop(newColumn, axis=1) can be used only once, if we run it again it will give error as it is already deleted

To drop a column but to keep it to the same memory location

data.drop('newColumn', axis =1, inplace=True)

To delete more than one columns, we have to write in a list

data.drop(['newColumn2','newColumn3'], axis =1, inplace=True)

1. To access single record

data.iloc[3] // index location

1. To find the data wrt row and column position

data.iat[3,4] // row 3 column 4

1. To find the data wrt row and column position using column name

data.loc[2,'Ticket']

1. To filter the range of records by giving rows range and columns range

data.loc[50:55,'PassengerId':'Sex']

1. We can give random row number and column names in a list

data.loc[[1,2,3,8,4,90,80],['PassengerId','Survived','Cabin','Name','Fare']]

1. Filter based on condition

If age <20

Data[data[‘Age’] < 20]

If age >15 and <20

temp = data[(data['Age'] > 15) & (data['Age']<20)]

To find how many persons are there of age 18

len(data[data['Age']==18])

1. To set a column as an index

data.set\_index('Name')

1. To remove the rows having missing values

temp = data.dropna()

1. To place some value in a rows having null value

temp = data.fillna(555555)

temp =data.isna()

temp

1. Grouping

data.groupby('Cabin').sum('Fare')

data.groupby('Sex').sum('Fare')

data.groupby('Sex').sum()['Fare']

1. To ignore warnings

Import warnings

warnings.filterwarnings('ignore')

1. if want to apply some function on some column

data[‘new\_column’] = data[‘Fare’].apply(funname)

def apply\_group(x):

if x<=100:

return 'A'

elif x>100 and x<=200:

return 'B'

else:

return 'C'

data['Fare\_Group'] = data['Fare'].apply(apply\_group)

1. Create dataframe

Use dictionary

my\_dict={

"key1" : [1,2,3,4,9,8],

"key2" : [25,36,14,58,69,8],

"key3" : [96,85,41,25,36,96]

}

my\_dict

output:

{'key1': [1, 2, 3, 4, 9, 8],

'key2': [25, 36, 14, 58, 69, 8],

'key3': [96, 85, 41, 25, 36, 96]}

df1 = pd.DataFrame(my\_dict)

df1

key1 key2 key3

0 1 25 96

1 2 36 85

2 3 14 41

3 4 58 25

4 9 69 36

5 8 8 96

type(df1)

output

pandas.core.frame.DataFrame

**Day 7**

1.

import mysql.connector as conn

if it is giving error like

**ModuleNotFoundError** Traceback (most recent call last)

Cell **In[1], line 1**

**----> 1** **import** **mysql.connector** **as** **conn**

**ModuleNotFoundError**: No module named 'mysql'

Then try this command as

pip install mysql-connector-python

execute this command

then use this command

pip install mysql-connector

execute this command

after installing both this packages, restart the Kernel

2. **myconn = conn.connect(host='localhost', user='root', passwd='root')**

If this is giving error like

**NotSupportedError**: Authentication plugin 'caching\_sha2\_password' is not supported

ALTER USER ‘bhagyalaxmi'@'your\_host' IDENTIFIED WITH 'mysql\_native\_password' BY 'your\_password';

If this not works, try this

UPDATE mysql.user SET Password=PASSWORD('1234') WHERE User=’bhagyalaxmi’;

If this not works, try this

ALTER USER 'bhagyalaxmi'@'localhost' IDENTIFIED BY '1234';

If this not works, try this

default\_authentication\_plugin=1234

version = mysql.connector.\_\_version\_\_

If this not works, try this

**pip install mysql-connector-python==8.0.32 –user # this works**

**restart the kernel**

3. query ='show tables'

cursor.execute(query)

cursor.fetchall()

4. query = "select \* from city"

data = pd.read\_sql\_query(query,myconn)

data

5. query = "select CountryCode,Language from countrylanguage"

data = pd.read\_sql\_query(query,myconn)

data.head(2)

6. Extract unique values from table column

query = "select distinct CountryCode from countrylanguage"

data = pd.read\_sql\_query(query,myconn)

data

check with

data['IsOfficial'].unique()

7. extract records based on mathematical conditions