1. Create a Pandas data frame for empdata.csv

```
import pandas as pd
df = pd.read_csv("/content/empdata.csv")
```

```
df.head()
```

	Empid	Ename	Salary	DOJ
0	1001	Ganesh	1000.00	10-10-2000
1	1002	Anil	23000.50	3/20/2002
2	1003	Gaurav	NaN	03-03-2002
3	1004	Hema Chandra	16500.50	09-10-2000
4	1005	Laxmi Prasanna	12000.75	10-08-2000

```
df.tail(2)
```

	Empid	Ename	Salary	DOJ
4	1005	Laxmi Prasanna	12000.75	10-08-2000
5	1006	Anant	9999.99	09-09-1999

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Empid   6 non-null      int64
 1   Ename   6 non-null      object
 2   Salary  5 non-null      float64
 3   DOJ     6 non-null      object
dtypes: float64(1), int64(1), object(2)
memory usage: 320.0+ bytes
```

```
df.describe()
```

	Empid	Salary
count	6.000000	5.000000
mean	1003.500000	12500.348000
std	1.870829	8139.622234

```
df.shape
```

```
(6, 4)
```

```
type(df)
```

```
pandas.core.frame.DataFrame
```

2. To retrieve column names

```
df.columns
```

```
Index(['Empid', 'Ename', 'Salary', 'DOJ'], dtype='object')
```

4. To retrieve column data

```
df.Empid
```

```
0    1001
1    1002
2    1003
3    1004
4    1005
5    1006
Name: Empid, dtype: int64
```

5. To retrieve a set of columns

```
df[['Empid','Ename']]
```

	Empid	Ename
1	1002	Anil

Check for duplicates and remove them

```
df1 = df.append(df)
print('Dimensions of the original frame', df.shape)
print('Dimensions of the frame with duplicates', df1.shape)
#remove the duplicates
df1 = df1.drop_duplicates()   #or use this statement df1.drop_duplicates(inplace = True)
print('Dimensions of the frame after removing duplicates', df1.shape)
```

```
Dimensions of the original frame (6, 4)
Dimensions of the frame with duplicates (12, 4)
Dimensions of the frame after removing duplicates (6, 4)
<ipython-input-22-b62acb36e572>:1: FutureWarning: The frame.append method is deprecat
  df1 = df.append(df)
```

```
#change all column names to Upper case
df.columns = [i.upper() for i in df]
print(df.columns)
```

```
Index(['EMPID', 'ENAME', 'SALARY', 'DOJ'], dtype='object')
```

Handling missing values

```
df.isna().sum()
#df.isnull().sum()

#print(df.isnull())
print('The no. of nulls in each column is \n',df.isnull().sum())
df.dropna(axis = 1, inplace = False)
```

```
The no. of nulls in each column is
 EMPID     0
```

```
df.isna().sum()
```

```
EMPID     0
ENAME     0
SALARY    1
DOJ       0
dtype: int64
```

1 1002 Anil 3/20/2002

```
df
```

	EMPID	ENAME	SALARY	DOJ
0	1001	Ganesh	1000.00	10-10-2000
1	1002	Anil	23000.50	3/20/2002
2	1003	Gaurav	NaN	03-03-2002
3	1004	Hema Chandra	16500.50	09-10-2000
4	1005	Laxmi Prasanna	12000.75	10-08-2000
5	1006	Anant	9999.99	09-09-1999

6. Find the highest and lowest salary

```
print('Highest Salary is',df['SALARY'].max())
print('Lowest Salary is', df['SALARY'].min())
```

```
Highest Salary is 23000.5
Lowest Salary is 1000.0
```

```
 # This is formatted as code
```

Display the details of employees whose salary is above 20000

```
df[df.SALARY > 20000]
```

	EMPID	ENAME	SALARY	DOJ
1	1002	Anil	23000.5	3/20/2002

Display only the id and names of employees whose salary is greater than 20000

```
df[['EMPID', 'ENAME']] [df.SALARY > 20000]
```

	EMPID	ENAME
1	1002	Anil

Display the Eid and name of the highest paid employee

```
df[['EMPID','ENAME']] [df.SALARY == df.SALARY.max()]
```

	EMPID	ENAME
1	1002	Anil

Display the enames whose salary is above the average salary

```
print('Average Salary is', df.SALARY.mean())
df['ENAME'][df.SALARY > df.SALARY.mean()]
```

```
Average Salary is 12500.348
1            Anil
3    Hema Chandra
Name: ENAME, dtype: object
```

Sort in ascending order of DOJ and store the result in another frame

```
df1['DOJ'] = pd.to_datetime(df1['DOJ'])    #convert DOJ to date type
df1.info()
print('Frame before sorting\n', df1)
df1.sort_values("DOJ", inplace = True)
print('Frame after sorting\n', df1)
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 6 entries, 0 to 5
Data columns (total 4 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   Empid   6 non-null      int64
 1   Ename   6 non-null      object
 2   Salary  5 non-null      float64
 3   DOJ     6 non-null      datetime64[ns]
dtypes: datetime64[ns](1), float64(1), int64(1), object(1)
memory usage: 240.0+ bytes
Frame before sorting
    Empid           Ename    Salary        DOJ
0   1001          Ganesh   1000.00 2000-10-10
1   1002            Anil  23000.50 2002-03-20
2   1003          Gaurav       NaN 2002-03-03
3   1004    Hema Chandra  16500.50 2000-09-10
4   1005  Laxmi Prasanna  12000.75 2000-10-08
5   1006           Anant    9999.99 1999-09-09
Frame after sorting
    Empid           Ename    Salary        DOJ
```

```
5   1006          Anant    9999.99 1999-09-09
3   1004    Hema Chandra  16500.50 2000-09-10
4   1005  Laxmi Prasanna  12000.75 2000-10-08
0   1001          Ganesh   1000.00 2000-10-10
2   1003          Gaurav       NaN 2002-03-03
1   1002            Anil  23000.50 2002-03-20
```

Sort in descending order of dates

```
df1.sort_values("DOJ", ascending = False, inplace = True)
df1
```

	Empid	Ename	Salary	DOJ
1	1002	Anil	23000.50	2002-03-20
2	1003	Gaurav	NaN	2002-03-03
0	1001	Ganesh	1000.00	2000-10-10
4	1005	Laxmi Prasanna	12000.75	2000-10-08
3	1004	Hema Chandra	16500.50	2000-09-10
5	1006	Anant	9999.99	1999-09-09

Sort DOJ in descending and salary in ascending order

```
df1.sort_values(by = ['DOJ', 'Salary'] , ascending = [False, True], inplace = True)
df1
```

	Empid	Ename	Salary	DOJ
1	1002	Anil	23000.50	2002-03-20
2	1003	Gaurav	NaN	2002-03-03
0	1001	Ganesh	1000.00	2000-10-10
4	1005	Laxmi Prasanna	12000.75	2000-10-08
3	1004	Hema Chandra	16500.50	2000-09-10
5	1006	Anant	9999.99	1999-09-09