```
import pandas as pd
import numpy as np

# Load the data
df = pd.read_csv('https://raw.githubusercontent.com/Deepsphere-AI/DataAnalyticsTraining/main/Python/UNIT%20-5%20LVA-DSAI-EMP-DATA-SET-V
```

```
#1
df['COMMISSION_PTC'] = df['DEPARTMENT_ID'].apply(lambda x: 0.10 if x == 100 else (0.11 if x == 110 else 0.05))
print(df['COMMISSION_PTC'])
```

```
0     0.05
1     0.05
2     0.05
3     0.05
4     0.05
5     0.05
6     0.05
7     0.11
8     0.11
9     0.05
10    0.05
11    0.05
12    0.05
13    0.05
14    0.05
15    0.05
16    0.05
17    0.10
18    0.10
19    0.10
20    0.10
21    0.10
22    0.10
23    0.05
24    0.05
25    0.05
26    0.05
27    0.05
28    0.05
29    0.05
30    0.05
31    0.05
32    0.05
33    0.05
34    0.05
35    0.05
36    0.05
37    0.05
38    0.05
39    0.05
40    0.05
41    0.05
42    0.05
43    0.05
44    0.05
45    0.05
46    0.05
47    0.05
48    0.05
49    0.05
50    0.05
Name: COMMISSION, dtype: float64
```

```
#2
grouped = df.groupby('JOB_ID')

print(grouped)
```

```
<pandas.core.groupby.generic.DataFrameGroupBy object at 0x7fdd5185f610>
```

```
#3
wage = df.groupby('DEPARTMENT_ID')['SALARY'].agg(['max', 'min'])
print(wage)
```

```
                    max      min
DEPARTMENT_ID
10.0             4400.0   4400.0
20.0            13000.0   6000.0
30.0            11000.0   2500.0
40.0             6500.0   6500.0
50.0             8200.0   2100.0
60.0             9000.0   4200.0
70.0            10000.0  10000.0
90.0            24000.0  17000.0
```

```
100.0          12008.0   6900.0
110.0          12008.0   8300.0
```

```
#4
df['MANAGER_ID'].fillna(124)
```

```
0     124
1     124
2     101
3     100
4     201
5     101
6     101
7     101
8     205
9      -
10    100
11    100
12    102
13    103
14    103
15    103
16    103
17    101
18    108
19    108
20    108
21    108
22    108
23    100
24    114
25    114
26    114
27    114
28    114
29    100
30    100
31    100
32    100
33    100
34    120
35    120
36    120
37    120
38    121
39    121
40    121
41    121
42    122
43    122
44    122
45    122
46    123
47    123
48    123
49    123
50    124
Name: MANAGER_ID, dtype: object
```

5. Select specific columns:

```
#5
selected_column = df[['EMPLOYEE_ID', 'JOB_ID', 'SALARY', 'COMMISSION_PCT', 'MANAGER_ID', 'DEPARTMENT_ID']]
print(selected_column)
```

```
    EMPLOYEE_ID      JOB_ID   SALARY  COMMISSION_PCT MANAGER_ID  DEPARTMENT_ID
0         198.0    SH_CLERK   2600.0            0.05        124           50.0
1         199.0    SH_CLERK   2600.0            0.05        124           50.0
2         200.0     AD_ASST   4400.0            0.05        101           10.0
3         201.0      MK_MAN  13000.0            0.05        100           20.0
4         202.0      MK_REP   6000.0            0.05        201           20.0
5         203.0      HR_REP   6500.0            0.05        101           40.0
6         204.0      PR_REP  10000.0            0.05        101           70.0
7         205.0      AC_MGR  12008.0            0.11        101          110.0
8         206.0  AC_ACCOUNT   8300.0            0.11        205          110.0
9         100.0     AD_PRES  24000.0            0.05         -            90.0
10        101.0       AD_VP  17000.0            0.05        100           90.0
11        102.0       AD_VP  17000.0            0.05        100           90.0
12        103.0     IT_PROG   9000.0            0.05        102           60.0
13        104.0     IT_PROG   6000.0            0.05        103           60.0
14        105.0     IT_PROG   4800.0            0.05        103           60.0
15        106.0     IT_PROG   4800.0            0.05        103           60.0
16        107.0     IT_PROG   4200.0            0.05        103           60.0
17        108.0      FI_MGR  12008.0            0.10        101          100.0
18        109.0  FI_ACCOUNT   9000.0            0.10        108          100.0
19        110.0  FI_ACCOUNT   8200.0            0.10        108          100.0
```

```
20        111.0  FI_ACCOUNT   7700.0           0.10        108          100.0
21        112.0  FI_ACCOUNT   7800.0           0.10        108          100.0
22        113.0  FI_ACCOUNT   6900.0           0.10        108          100.0
23        114.0      PU_MAN  11000.0           0.05        100           30.0
24        115.0    PU_CLERK   3100.0           0.05        114           30.0
25        116.0    PU_CLERK   2900.0           0.05        114           30.0
26        117.0    PU_CLERK   2800.0           0.05        114           30.0
27        118.0    PU_CLERK   2600.0           0.05        114           30.0
28        119.0    PU_CLERK   2500.0           0.05        114           30.0
29        120.0      ST_MAN   8000.0           0.05        100           50.0
30        121.0      ST_MAN   8200.0           0.05        100           50.0
31        122.0      ST_MAN   7900.0           0.05        100           50.0
32        123.0      ST_MAN   6500.0           0.05        100           50.0
33        124.0      ST_MAN   5800.0           0.05        100           50.0
34        125.0    ST_CLERK   3200.0           0.05        120           50.0
35        126.0    ST_CLERK   2700.0           0.05        120           50.0
36        127.0    ST_CLERK   2400.0           0.05        120           50.0
37        128.0    ST_CLERK   2200.0           0.05        120           50.0
38        129.0    ST_CLERK   3300.0           0.05        121           50.0
39        130.0    ST_CLERK   2800.0           0.05        121           50.0
40        131.0    ST_CLERK   2500.0           0.05        121           50.0
41        132.0    ST_CLERK   2100.0           0.05        121           50.0
42        133.0    ST_CLERK   3300.0           0.05        122           50.0
43        134.0    ST_CLERK   2900.0           0.05        122           50.0
44        135.0    ST_CLERK   2400.0           0.05        122           50.0
45        136.0    ST_CLERK   2200.0           0.05        122           50.0
46        137.0    ST_CLERK   3600.0           0.05        123           50.0
47        138.0    ST_CLERK   3200.0           0.05        123           50.0
48        139.0    ST_CLERK   2700.0           0.05        123           50.0
49        140.0    ST_CLERK   2500.0           0.05        123           50.0
50          NaN         NaN      NaN           0.05        124            NaN
```

```
#6
df['SALARY'] = df['SALARY'].astype(float).fillna(0)
print(df['SALARY'])
df['COMMISSION_PCT'] = df['COMMISSION_PCT'].astype(float).fillna(0)
print(df['COMMISSION_PCT'])
```

```
0      2600.0
1      2600.0
2      4400.0
3     13000.0
4      6000.0
5      6500.0
6     10000.0
7     12008.0
8      8300.0
9     24000.0
10    17000.0
11    17000.0
12     9000.0
13     6000.0
14     4800.0
15     4800.0
16     4200.0
17    12008.0
18     9000.0
19     8200.0
20     7700.0
21     7800.0
22     6900.0
23    11000.0
24     3100.0
25     2900.0
26     2800.0
27     2600.0
28     2500.0
29     8000.0
30     8200.0
31     7900.0
32     6500.0
33     5800.0
34     3200.0
35     2700.0
36     2400.0
37     2200.0
38     3300.0
39     2800.0
40     2500.0
41     2100.0
42     3300.0
43     2900.0
44     2400.0
45     2200.0
46     3600.0
47     3200.0
48     2700.0
49     2500.0
50        0.0
```

```
Name: SALARY, dtype: float64
0     0.05
1     0.05
2     0.05
3     0.05
4     0.05
5     0.05
```

```
#7
df.tail(25)
```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB
26	117.0	Sigal	Tobias	STOBIAS	515.127.4564	24-Jul-05	PU_CLE
27	118.0	Guy	Himuro	GHIMURO	515.127.4565	15-Nov-06	PU_CLE
28	119.0	Karen	Colmenares	KCOLMENA	515.127.4566	10-Aug-07	PU_CLE
29	120.0	Matthew	Weiss	MWEISS	650.123.1234	18-Jul-04	ST_M
30	121.0	Adam	Fripp	AFRIPP	650.123.2234	10-Apr-05	ST_M
31	122.0	Payam	Kaufling	PKAUFLIN	650.123.3234	1-May-03	ST_M
32	123.0	Shanta	Vollman	SVOLLMAN	650.123.4234	10-Oct-05	ST_M
33	124.0	Kevin	Mourgos	KMOURGOS	650.123.5234	16-Nov-07	ST_M
34	125.0	Julia	Nayer	JNAYER	650.124.1214	16-Jul-05	ST_CLE
35	126.0	Irene	Mikkilineni	IMIKKILI	650.124.1224	28-Sep-06	ST_CLE
36	127.0	James	Landry	JLANDRY	650.124.1334	14-Jan-07	ST_CLE
37	128.0	Steven	Markle	SMARKLE	650.124.1434	8-Mar-08	ST_CLE
38	129.0	Laura	Bissot	LBISSOT	650.124.5234	20-Aug-05	ST_CLE
39	130.0	Mozhe	Atkinson	MATKINSO	650.124.6234	30-Oct-05	ST_CLE
40	131.0	James	Marlow	JAMRLOW	650.124.7234	16-Feb-05	ST_CLE
41	132.0	TJ	Olson	TJOLSON	650.124.8234	10-Apr-07	ST_CLE
42	133.0	Jason	Mallin	JMALLIN	650.127.1934	14-Jun-04	ST_CLE
43	134.0	Michael	Rogers	MROGERS	650.127.1834	26-Aug-06	ST_CLE
44	135.0	Ki	Gee	KGEE	650.127.1734	12-Dec-07	ST_CLE
45	136.0	Hazel	Philtanker	HPHILTAN	650.127.1634	6-Feb-08	ST_CLE
46	137.0	Renske	Ladwig	RLADWIG	650.121.1234	14-Jul-03	ST_CLE
47	138.0	Stephen	Stiles	SSTILES	650.121.2034	26-Oct-05	ST_CLE
48	139.0	John	Seo	JSEO	650.121.2019	12-Feb-06	ST_CLE
49	140.0	Joshua	Patel	JPATEL	650.121.1834	6-Apr-06	ST_CLE
50	NaN	NaN	NaN	NaN	NaN	NaN	N

```
#8
df['TOTAL_COMPENSATION'] = df['SALARY'] + df['COMMISSION_PCT']
print(df['TOTAL_COMPENSATION'] )
```

```
0      2600.05
1      2600.05
2      4400.05
3     13000.05
4      6000.05
5      6500.05
6     10000.05
7     12008.11
8      8300.11
9     24000.05
10    17000.05
11    17000.05
12     9000.05
13     6000.05
14     4800.05
15     4800.05
16     4200.05
17    12008.10
18     9000.10
19     8200.10
20     7700.10
21     7800.10
22     6900.10
23    11000.05
```

```
24     3100.05
25     2900.05
26     2800.05
27     2600.05
28     2500.05
29     8000.05
30     8200.05
31     7900.05
32     6500.05
33     5800.05
34     3200.05
35     2700.05
36     2400.05
37     2200.05
38     3300.05
39     2800.05
40     2500.05
41     2100.05
42     3300.05
43     2900.05
44     2400.05
45     2200.05
46     3600.05
47     3200.05
48     2700.05
49     2500.05
50        0.05
Name: TOTAL_COMPENSATION, dtype: float64
```

```
#9
df['EMPLOYEE_LOCATION'] = np.nan
df['EMPLOYEE_BIRTH_DATE'] = pd.NaT
df['EMPLOYEE_REFERRAL'] = np.nan
```

10. JSON (JavaScript Object Notation) is a lightweight data-interchange format that is easy for humans to read and write and easy for machines to parse and generate. JSON is often used when data is sent from a server to a web page. JSON is better than CSV when the data is hierarchical (values can be nested), and when data structure can vary from record to record.
11. For financial data, CSV might be a better choice because financial data is often tabular and not hierarchical, which fits well with the CSV format.
12. In Python, you can create a DataFrame from various data sources such as CSV files, Excel files, SQL queries, lists, dictionaries, and many others.

```
#13
def calculate_service_seniority(df):
    today = pd.to_datetime('today')
    df['SERVICE_YEARS'] = (today - pd.to_datetime(df['HIRE_DATE'])).dt.days // 365
    df['SENIORITY'] = df['SERVICE_YEARS'] // 5
    return df

df = calculate_service_seniority(df)
print(df)
```

```
    EMPLOYEE_ID   FIRST_NAME    LAST_NAME     EMAIL  PHONE_NUMBER  HIRE_DATE  \
0         198.0       Donald     OConnell  DOCONNEL  650.507.9833  21-Jun-07
1         199.0      Douglas        Grant    DGRANT  650.507.9844  13-Jan-08
2         200.0     Jennifer       Whalen   JWHALEN  515.123.4444  17-Sep-03
3         201.0      Michael    Hartstein  MHARTSTE  515.123.5555  17-Feb-04
4         202.0          Pat          Fay      PFAY  603.123.6666  17-Aug-05
5         203.0        Susan       Mavris   SMAVRIS  515.123.7777   7-Jun-02
6         204.0      Hermann         Baer     HBAER  515.123.8888   7-Jun-02
7         205.0      Shelley      Higgins  SHIGGINS  515.123.8080   7-Jun-02
8         206.0      William        Gietz    WGIETZ  515.123.8181   7-Jun-02
9         100.0       Steven         King     SKING  515.123.4567  17-Jun-03
10        101.0        Neena      Kochhar  NKOCHHAR  515.123.4568  21-Sep-05
11        102.0          Lex      De Haan   LDEHAAN  515.123.4569  13-Jan-01
12        103.0    Alexander       Hunold   AHUNOLD  590.423.4567   3-Jan-06
13        104.0        Bruce        Ernst    BERNST  590.423.4568  21-May-07
14        105.0        David       Austin   DAUSTIN  590.423.4569  25-Jun-05
15        106.0        Valli    Pataballa  VPATABAL  590.423.4560   5-Feb-06
16        107.0        Diana      Lorentz  DLORENTZ  590.423.5567   7-Feb-07
17        108.0        Nancy    Greenberg  NGREENBE  515.124.4569  17-Aug-02
18        109.0       Daniel       Faviet   DFAVIET  515.124.4169  16-Aug-02
19        110.0         John         Chen     JCHEN  515.124.4269  28-Sep-05
20        111.0       Ismael      Sciarra  ISCIARRA  515.124.4369  30-Sep-05
21        112.0  Jose Manuel        Urman   JMURMAN  515.124.4469   7-Mar-06
22        113.0         Luis         Popp     LPOPP  515.124.4567   7-Dec-07
23        114.0          Den     Raphaely  DRAPHEAL  515.127.4561   7-Dec-02
24        115.0    Alexander         Khoo     AKHOO  515.127.4562  18-May-03
25        116.0       Shelli        Baida    SBAIDA  515.127.4563  24-Dec-05
26        117.0        Sigal       Tobias   STOBIAS  515.127.4564  24-Jul-05
27        118.0          Guy       Himuro   GHIMURO  515.127.4565  15-Nov-06
28        119.0        Karen   Colmenares  KCOLMENA  515.127.4566  10-Aug-07
```

```
29        120.0      Matthew        Weiss    MWEISS  650.123.1234  18-Jul-04
30        121.0         Adam        Fripp    AFRIPP  650.123.2234  10-Apr-05
31        122.0        Payam     Kaufling  PKAUFLIN  650.123.3234   1-May-03
32        123.0       Shanta      Vollman  SVOLLMAN  650.123.4234  10-Oct-05
33        124.0        Kevin      Mourgos  KMOURGOS  650.123.5234  16-Nov-07
34        125.0        Julia        Nayer    JNAYER  650.124.1214  16-Jul-05
35        126.0        Irene  Mikkilineni  IMIKKILI  650.124.1224  28-Sep-06
36        127.0        James       Landry   JLANDRY  650.124.1334  14-Jan-07
37        128.0       Steven       Markle   SMARKLE  650.124.1434   8-Mar-08
38        129.0        Laura       Bissot   LBISSOT  650.124.5234  20-Aug-05
39        130.0        Mozhe     Atkinson  MATKINSO  650.124.6234  30-Oct-05
40        131.0        James       Marlow   JAMRLOW  650.124.7234  16-Feb-05
41        132.0           TJ        Olson   TJOLSON  650.124.8234  10-Apr-07
42        133.0        Jason       Mallin   JMALLIN  650.127.1934  14-Jun-04
43        134.0      Michael       Rogers   MROGERS  650.127.1834  26-Aug-06
44        135.0           Ki          Gee      KGEE  650.127.1734  12-Dec-07
45        136.0        Hazel   Philtanker  HPHILTAN  650.127.1634   6-Feb-08
46        137.0       Renske       Ladwig   RLADWIG  650.121.1234  14-Jul-03
47        138.0      Stephen       Stiles   SSTILES  650.121.2034  26-Oct-05
48        139.0         John          Seo      JSEO  650.121.2019  12-Feb-06
49        140.0       Joshua        Patel    JPATEL  650.121.1834   6-Apr-06
50          NaN          NaN          NaN       NaN           NaN        NaN

        JOB_ID   SALARY  COMMISSION_PCT MANAGER_ID  DEPARTMENT_ID  COMMISSION  \
0     SH_CLERK   2600.0            0.05        124           50.0        0.05
1     SH_CLERK   2600.0            0.05        124           50.0        0.05
2      AD_ASST   4400.0            0.05        101           10.0        0.05
3       MK_MAN  13000.0            0.05        100           20.0        0.05
```

```
#14
a = df[df['DEPARTMENT_ID'].between(1, 50)]
b = df[df['DEPARTMENT_ID'].between(51, 110)]
print(a)
print(b)
```

```
12             9000.05                NaN                NaT
13             6000.05                NaN                NaT
14             4800.05                NaN                NaT
15             4800.05                NaN                NaT
```

```
#15
df = df[df.duplicated(['FIRST_NAME', 'LAST_NAME'])]
print(df)
```

```
Empty DataFrame
Columns: [EMPLOYEE_ID, FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, DEPARTMENT
Index: []
```