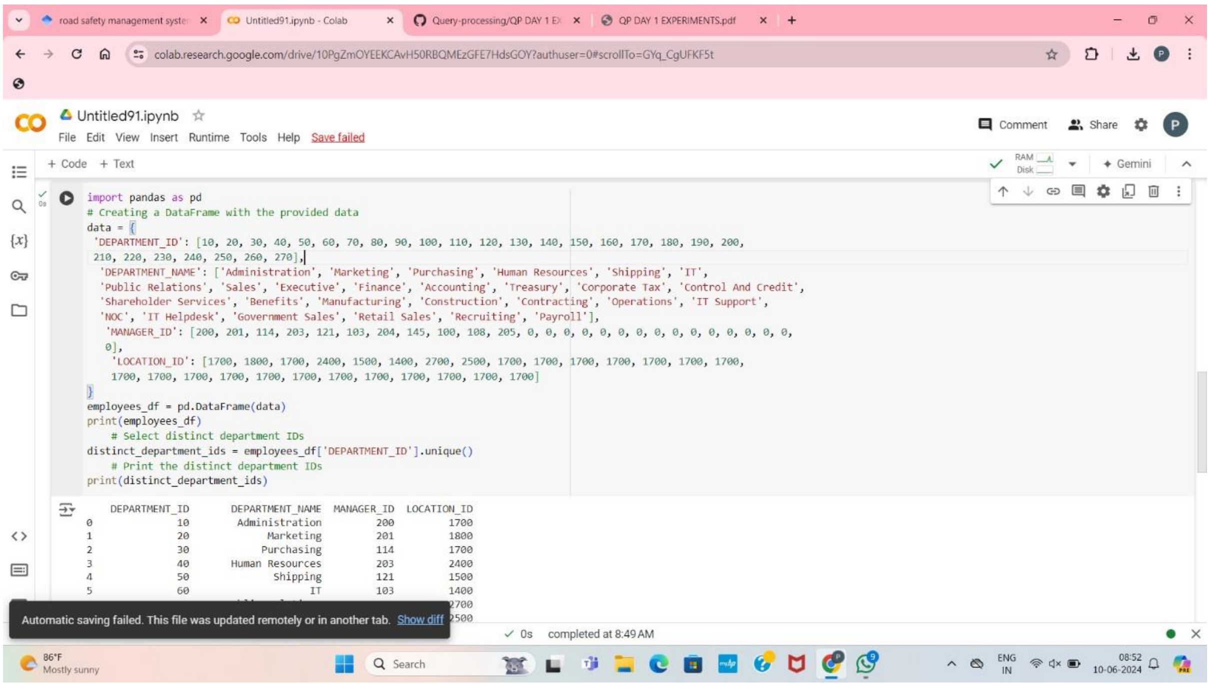
1.Write a Pandas program to select distinct department id from employees
file.

```
+---------------+----------------------+------------+-------------+
| DEPARTMENT_ID | DEPARTMENT_NAME      | MANAGER_ID | LOCATION_ID |
+---------------+----------------------+------------+-------------+
|            10 | Administration       |        200 |        1700 |
|            20 | Marketing            |        201 |        1800 |
|            30 | Purchasing           |        114 |        1700 |
|            40 | Human Resources      |        203 |        2400 |
|            50 | Shipping             |        121 |        1500 |
|            60 | IT                   |        103 |        1400 |
|            70 | Public Relations     |        204 |        2700 |
|            80 | Sales                |        145 |        2500 |
|            90 | Executive            |        100 |        1700 |
|           100 | Finance              |        108 |        1700 |
|           110 | Accounting           |        205 |        1700 |
|           120 | Treasury             |          0 |        1700 |
|           130 | Corporate Tax        |          0 |        1700 |
|           140 | Control And Credit   |          0 |        1700 |
|           150 | Shareholder Services |          0 |        1700 |
|           160 | Benefits             |          0 |        1700 |
|           170 | Manufacturing        |          0 |        1700 |
|           180 | Construction         |          0 |        1700 |
|           190 | Contracting          |          0 |        1700 |
|           200 | Operations           |          0 |        1700 |
|           210 | IT Support           |          0 |        1700 |
|           220 | NOC                  |          0 |        1700 |
|           230 | IT Helpdesk          |          0 |        1700 |
|           240 | Government Sales     |          0 |        1700 |
|           250 | Retail Sales         |          0 |        1700 |
|           260 | Recruiting           |          0 |        1700 |
|           270 | Payroll              |          0 |        1700 |
+---------------+----------------------+------------+------------
```

CODE:


road safety management syste
Untitled91.ipynb - Colab
Query-processing/QP DAY 1 E
QP DAY 1 EXPERIMENTS.pdf
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=GYq_CgUFKF5t
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
Comment
Share
+ Code + Text
RAM
Disk
Gemini
86°F
Mostly sunny
Search
ENG
IN
08:52
10-06-2024

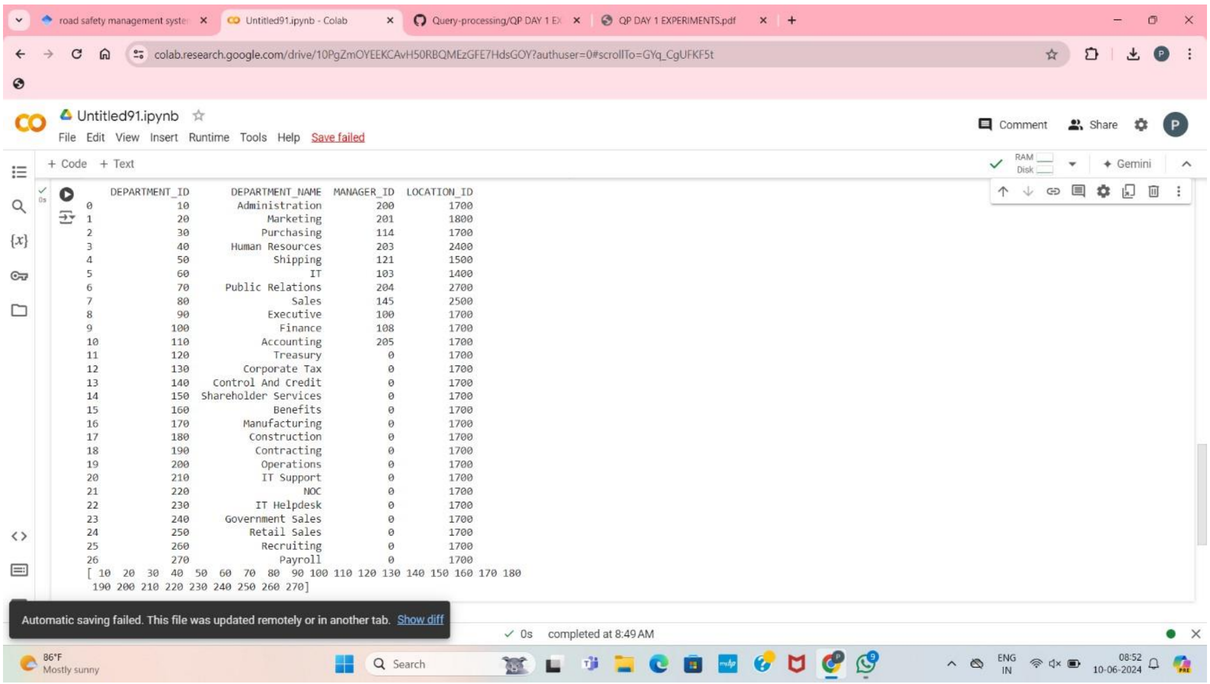
```
import pandas as pd
# Creating a DataFrame with the provided data
data = {
 'DEPARTMENT_ID': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200,
 210, 220, 230, 240, 250, 260, 270],
  'DEPARTMENT_NAME': ['Administration', 'Marketing', 'Purchasing', 'Human Resources', 'Shipping', 'IT',
  'Public Relations', 'Sales', 'Executive', 'Finance', 'Accounting', 'Treasury', 'Corporate Tax', 'Control And Credit',
  'Shareholder Services', 'Benefits', 'Manufacturing', 'Construction', 'Contracting', 'Operations', 'IT Support',
  'NOC', 'IT Helpdesk', 'Government Sales', 'Retail Sales', 'Recruiting', 'Payroll'],
   'MANAGER_ID': [200, 201, 114, 203, 121, 103, 204, 145, 100, 108, 205, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
   0],
    'LOCATION_ID': [1700, 1800, 1700, 2400, 1500, 1400, 2700, 2500, 1700, 1700, 1700, 1700, 1700, 1700, 1700,
    1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700, 1700]
}
employees_df = pd.DataFrame(data)
print(employees_df)
    # Select distinct department IDs
distinct_department_ids = employees_df['DEPARTMENT_ID'].unique()
    # Print the distinct department IDs
print(distinct_department_ids)
```

```
   DEPARTMENT_ID       DEPARTMENT_NAME  MANAGER_ID  LOCATION_ID
0             10        Administration         200         1700
1             20             Marketing         201         1800
2             30            Purchasing         114         1700
3             40       Human Resources         203         2400
4             50              Shipping         121         1500
5             60                    IT         103         1400
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 8:49 AM

OUTPUT:


road safety management syste
Untitled91.ipynb - Colab
Query-processing/QP DAY 1 E
QP DAY 1 EXPERIMENTS.pdf
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=GYq_CgUFKF5t
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
Comment
Share
+ Code + Text
RAM
Disk
Gemini
86°F
Mostly sunny
Search
ENG
IN
08:52
10-06-2024

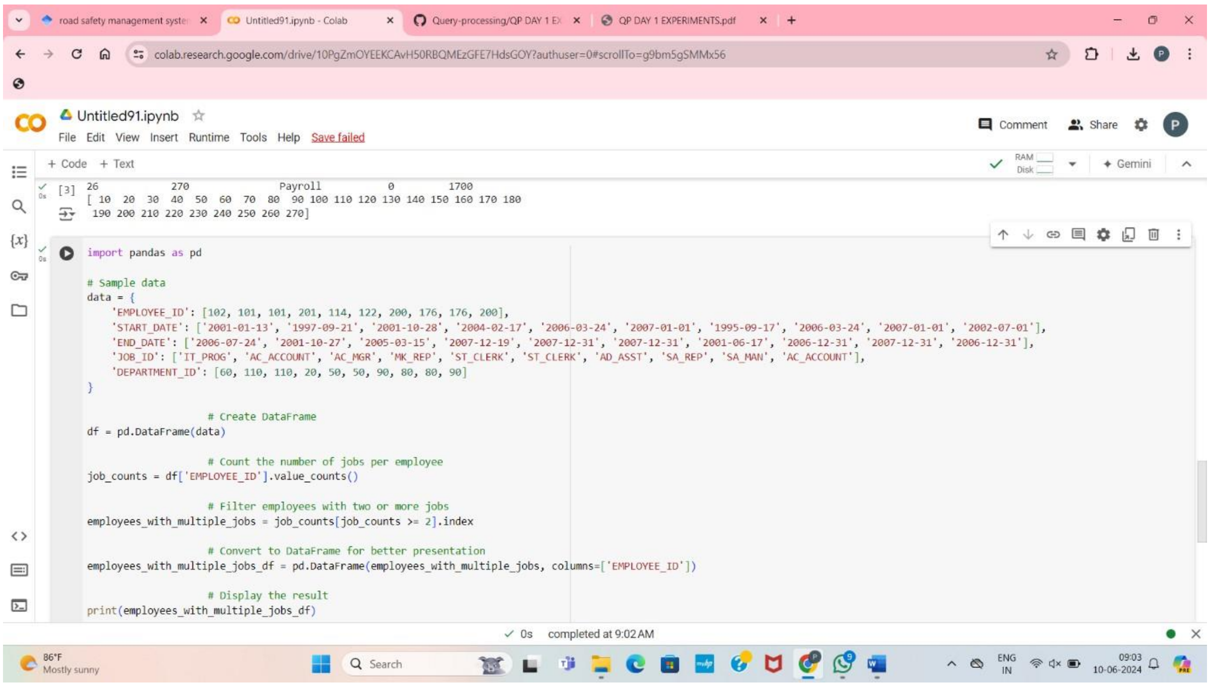
```
    DEPARTMENT_ID       DEPARTMENT_NAME  MANAGER_ID  LOCATION_ID
0              10        Administration         200         1700
1              20             Marketing         201         1800
2              30            Purchasing         114         1700
3              40       Human Resources         203         2400
4              50              Shipping         121         1500
5              60                    IT         103         1400
6              70      Public Relations         204         2700
7              80                 Sales         145         2500
8              90             Executive         100         1700
9             100               Finance         108         1700
10            110            Accounting         205         1700
11            120              Treasury           0         1700
12            130         Corporate Tax           0         1700
13            140    Control And Credit           0         1700
14            150  Shareholder Services           0         1700
15            160              Benefits           0         1700
16            170         Manufacturing           0         1700
17            180          Construction           0         1700
18            190           Contracting           0         1700
19            200            Operations           0         1700
20            210            IT Support           0         1700
21            220                   NOC           0         1700
22            230           IT Helpdesk           0         1700
23            240      Government Sales           0         1700
24            250          Retail Sales           0         1700
25            260            Recruiting           0         1700
26            270               Payroll           0         1700
[ 10  20  30  40  50  60  70  80  90 100 110 120 130 140 150 160 170 180
 190 200 210 220 230 240 250 260 270]
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 8:49 AM

2.Write a Pandas program to display the ID for those employees who did two or more jobs in the past.

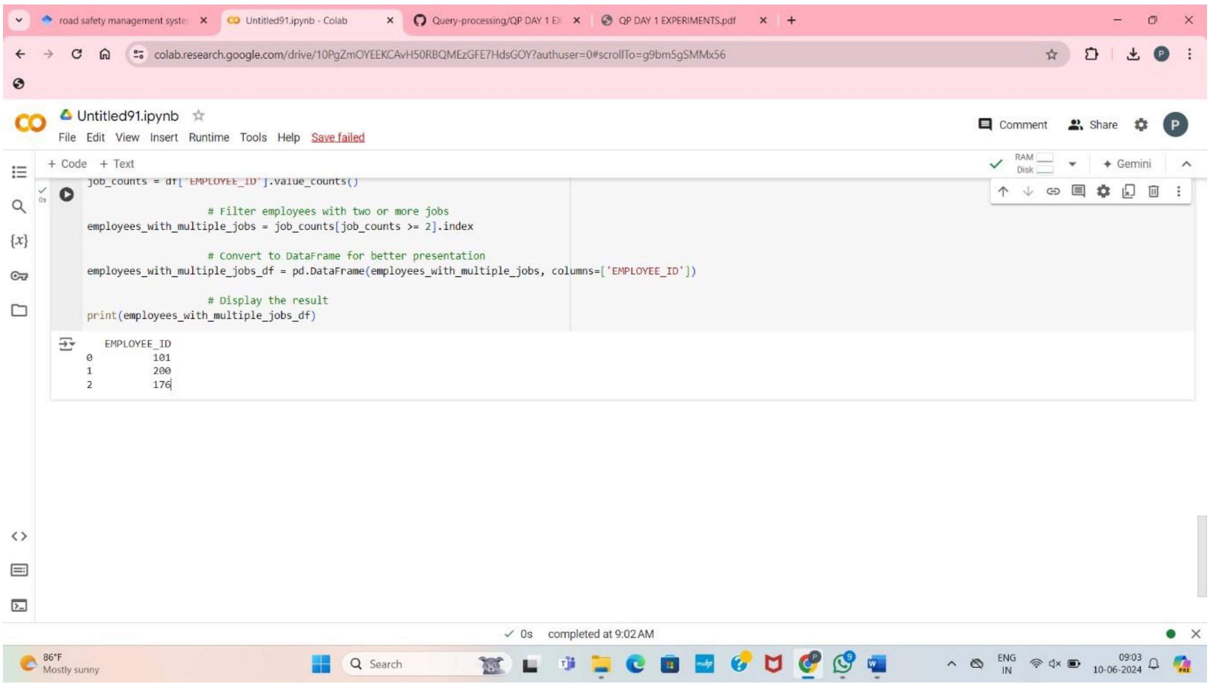
```
+-------------+------------+------------+------------+---------------+
| EMPLOYEE_ID | START_DATE | END_DATE   | JOB_ID     | DEPARTMENT_ID |
+-------------+------------+------------+------------+---------------+
|         102 | 2001-01-13 | 2006-07-24 | IT_PROG    |            60 |
|         101 | 1997-09-21 | 2001-10-27 | AC_ACCOUNT |           110 |
|         101 | 2001-10-28 | 2005-03-15 | AC_MGR     |           110 |
|         201 | 2004-02-17 | 2007-12-19 | MK_REP     |            20 |
|         114 | 2006-03-24 | 2007-12-31 | ST_CLERK   |            50 |
|          122 | 2007-01-01 | 2007-12-31 | ST_CLERK   |          50 |
|          200 | 1995-09-17 | 2001-06-17 | AD_ASST    |          90 |
|          176 | 2006-03-24 | 2006-12-31 | SA_REP     |          80 |
|          176 | 2007-01-01 | 2007-12-31 | SA_MAN     |          80 |
|          200 | 2002-07-01 | 2006-12-31 | AC_ACCOUNT |          90 | +---
----------+------------+------------+------------+--------------+
```



```
[3] 26            270              Payroll          0          1700
    [ 10  20  30  40  50  60  70  80  90 100 110 120 130 140 150 160 170 180
     190 200 210 220 230 240 250 260 270]
```

```python
import pandas as pd

# Sample data
data = {
    'EMPLOYEE_ID': [102, 101, 101, 201, 114, 122, 200, 176, 176, 200],
    'START_DATE': ['2001-01-13', '1997-09-21', '2001-10-28', '2004-02-17', '2006-03-24', '2007-01-01', '1995-09-17', '2006-03-24', '2007-01-01', '2002-07-01'],
    'END_DATE': ['2006-07-24', '2001-10-27', '2005-03-15', '2007-12-19', '2007-12-31', '2007-12-31', '2001-06-17', '2006-12-31', '2007-12-31', '2006-12-31'],
    'JOB_ID': ['IT_PROG', 'AC_ACCOUNT', 'AC_MGR', 'MK_REP', 'ST_CLERK', 'ST_CLERK', 'AD_ASST', 'SA_REP', 'SA_MAN', 'AC_ACCOUNT'],
    'DEPARTMENT_ID': [60, 110, 110, 20, 50, 50, 90, 80, 80, 90]
}

                    # Create DataFrame
df = pd.DataFrame(data)

                    # Count the number of jobs per employee
job_counts = df['EMPLOYEE_ID'].value_counts()

                    # Filter employees with two or more jobs
employees_with_multiple_jobs = job_counts[job_counts >= 2].index

                    # Convert to DataFrame for better presentation
employees_with_multiple_jobs_df = pd.DataFrame(employees_with_multiple_jobs, columns=['EMPLOYEE_ID'])

                    # Display the result
print(employees_with_multiple_jobs_df)
```

OUTPUT:



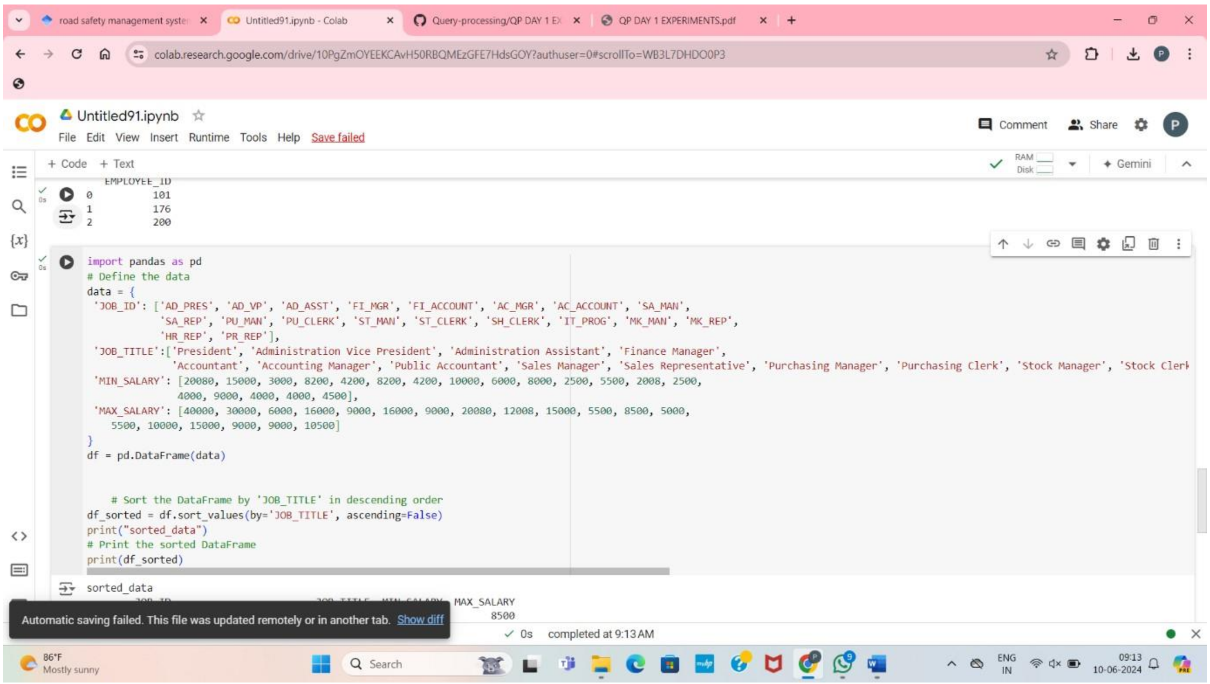
```python
job_counts = df['EMPLOYEE_ID'].value_counts()

                    # Filter employees with two or more jobs
employees_with_multiple_jobs = job_counts[job_counts >= 2].index

                    # Convert to DataFrame for better presentation
employees_with_multiple_jobs_df = pd.DataFrame(employees_with_multiple_jobs, columns=['EMPLOYEE_ID'])

                    # Display the result
print(employees_with_multiple_jobs_df)
```

```
   EMPLOYEE_ID
0          101
1          200
2          176
```

1. Write a Pandas program to display the details of jobs in descending sequence on job title.

```
+------------+-----------------------------------+------------+------------+
| JOB_ID     | JOB_TITLE                         | MIN_SALARY | MAX_SALARY |
+------------+-----------------------------------+------------+------------+
| AD_PRES    | President                         |      20080 |      40000 |
| AD_VP      | Administration Vice President     |      15000 |      30000 |
| AD_ASST    | Administration Assistant          |       3000 |       6000 |
| FI_MGR     | Finance Manager                   |       8200 |      16000 |
| FI_ACCOUNT | Accountant                        |       4200 |       9000 |
| AC_MGR     | Accounting Manager                |       8200 |      16000 |
| AC_ACCOUNT | Public Accountant                 |       4200 |       9000 |
| SA_MAN     | Sales Manager                     |      10000 |      20080 |
| SA_REP     | Sales Representative              |       6000 |      12008 |
| PU_MAN     | Purchasing Manager                |       8000 |      15000 |
| PU_CLERK   | Purchasing Clerk                  |       2500 |       5500 |
| ST_MAN     | Stock Manager                     |       5500 |       8500 |
| ST_CLERK   | Stock Clerk                       |       2008 |       5000 |
| SH_CLERK   | Shipping Clerk                    |       2500 |       5500 |
| IT_PROG    | Programmer                        |       4000 |      10000 |
| MK_MAN     | Marketing Manager                 |       9000 |      15000 |
| MK_REP     | Marketing Representative          |       4000 |       9000 |
| HR_REP     | Human Resources Representative    |       4000 |       9000 |
| PR_REP     | Public Relations Representative   |       4500 |      10500 |
+------------+-----------------------------------+------------+------------+
```

CODE:


road safety management syste
Untitled91.ipynb - Colab
Query-processing/QP DAY 1 EX
QP DAY 1 EXPERIMENTS.pdf
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=WB3L7DHDO0P3
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
Comment
Share
+ Code + Text
RAM
Disk
Gemini
86°F
Mostly sunny
Search
ENG
IN
09:13
10-06-2024

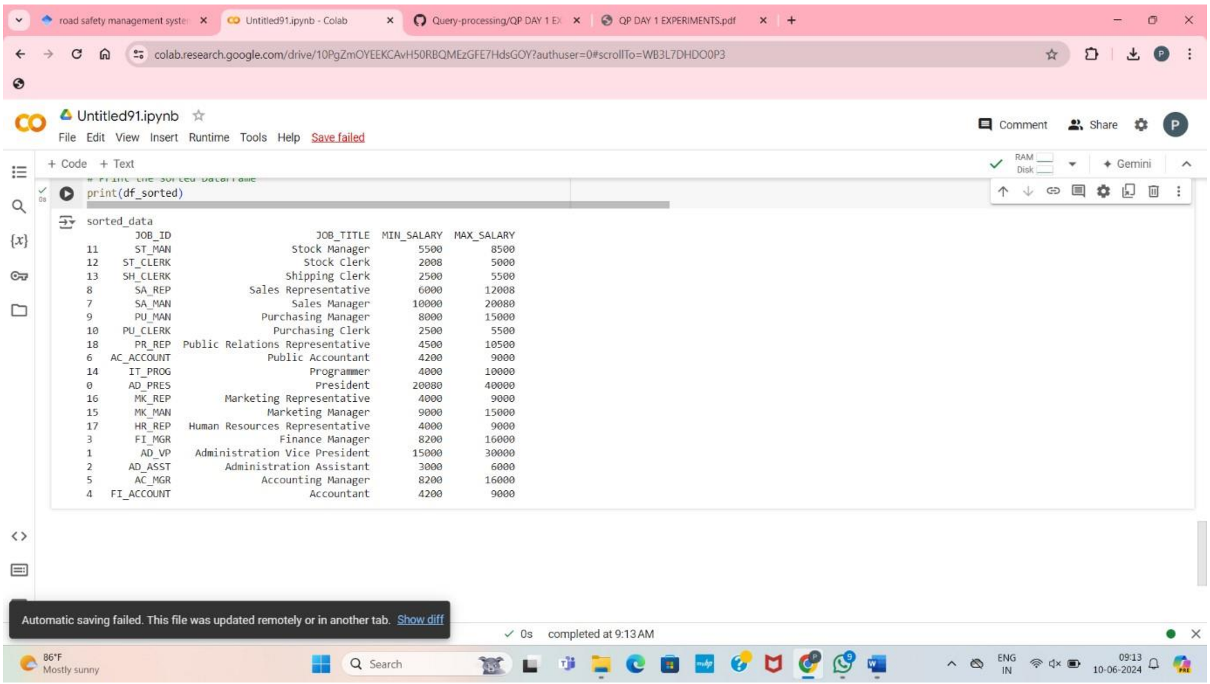
```
   EMPLOYEE_ID
0          101
1          176
2          200
```

```
import pandas as pd
# Define the data
data = {
 'JOB_ID': ['AD_PRES', 'AD_VP', 'AD_ASST', 'FI_MGR', 'FI_ACCOUNT', 'AC_MGR', 'AC_ACCOUNT', 'SA_MAN',
           'SA_REP', 'PU_MAN', 'PU_CLERK', 'ST_MAN', 'ST_CLERK', 'SH_CLERK', 'IT_PROG', 'MK_MAN', 'MK_REP',
           'HR_REP', 'PR_REP'],
 'JOB_TITLE':['President', 'Administration Vice President', 'Administration Assistant', 'Finance Manager',
             'Accountant', 'Accounting Manager', 'Public Accountant', 'Sales Manager', 'Sales Representative', 'Purchasing Manager', 'Purchasing Clerk', 'Stock Manager', 'Stock Clerk
 'MIN_SALARY': [20080, 15000, 3000, 8200, 4200, 8200, 4200, 10000, 6000, 8000, 2500, 5500, 2008, 2500,
              4000, 9000, 4000, 4000, 4500],
 'MAX_SALARY': [40000, 30000, 6000, 16000, 9000, 16000, 9000, 20080, 12008, 15000, 5500, 8500, 5000,
   5500, 10000, 15000, 9000, 9000, 10500]
}
df = pd.DataFrame(data)


    # Sort the DataFrame by 'JOB_TITLE' in descending order
df_sorted = df.sort_values(by='JOB_TITLE', ascending=False)
print("sorted_data")
# Print the sorted DataFrame
print(df_sorted)
```

```
sorted_data
                                          MAX_SALARY
                                                8500
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 9:13 AM

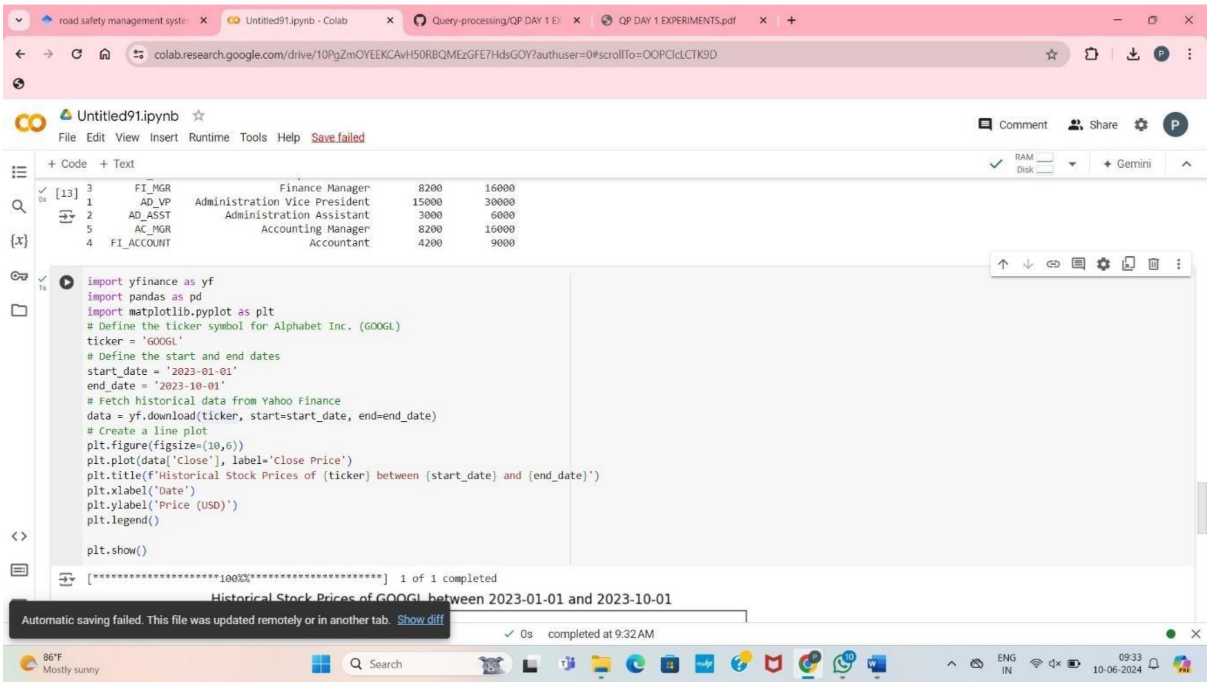


Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
+ Code + Text
Comment Share
Automatic saving failed. This file was updated remotely or in another tab. Show diff
completed at 9:13 AM

```
print(df_sorted)
```

```
sorted_data
       JOB_ID                        JOB_TITLE  MIN_SALARY  MAX_SALARY
11     ST_MAN                    Stock Manager        5500        8500
12   ST_CLERK                      Stock Clerk        2008        5000
13   SH_CLERK                   Shipping Clerk        2500        5500
8      SA_REP             Sales Representative        6000       12008
7      SA_MAN                    Sales Manager       10000       20080
9      PU_MAN               Purchasing Manager        8000       15000
10   PU_CLERK                 Purchasing Clerk        2500        5500
18     PR_REP  Public Relations Representative        4500       10500
6  AC_ACCOUNT                Public Accountant        4200        9000
14    IT_PROG                       Programmer        4000       10000
0     AD_PRES                        President       20080       40000
16     MK_REP         Marketing Representative        4000        9000
15     MK_MAN                Marketing Manager        9000       15000
17     HR_REP   Human Resources Representative        4000        9000
3      FI_MGR                  Finance Manager        8200       16000
1       AD_VP   Administration Vice President       15000       30000
2     AD_ASST         Administration Assistant        3000        6000
5      AC_MGR               Accounting Manager        8200       16000
4  FI_ACCOUNT                       Accountant        4200        9000
```

4.Write a Pandas program to create a line plot of the historical stock prices of Alphabet Inc. between two specific dates.

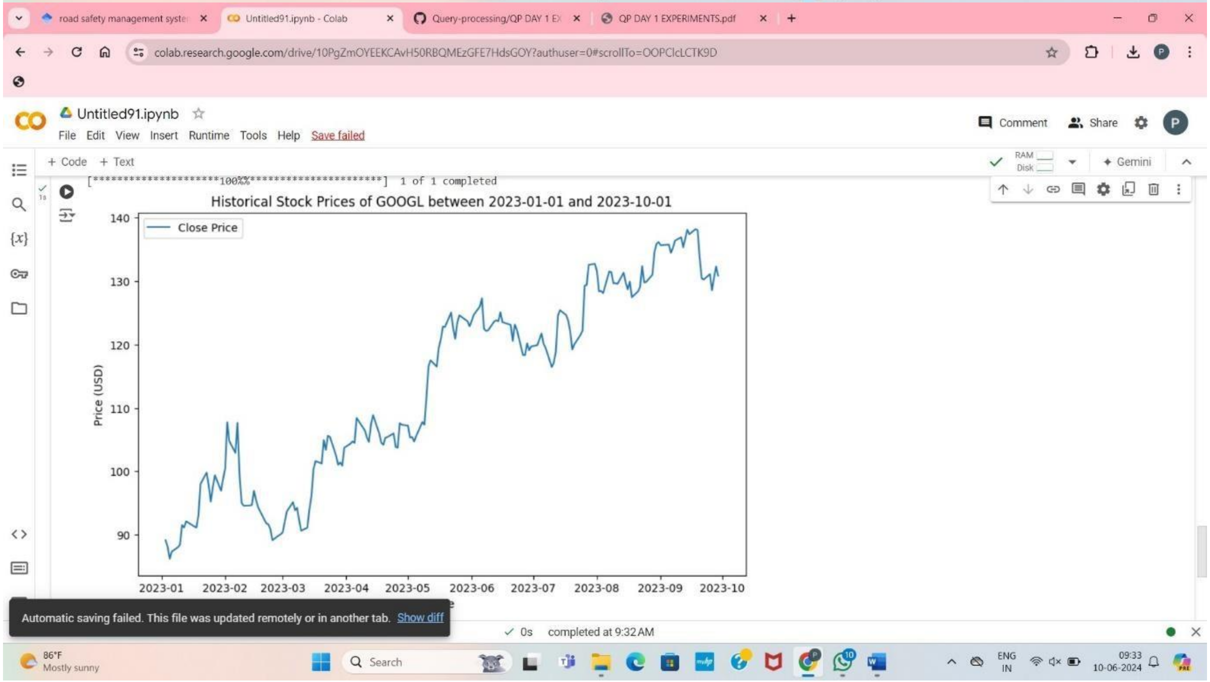


Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
+ Code + Text
Automatic saving failed. This file was updated remotely or in another tab. Show diff
completed at 9:32 AM

```
[13] 3     FI_MGR                  Finance Manager    8200   16000
     1      AD_VP   Administration Vice President   15000   30000
     2    AD_ASST        Administration Assistant    3000    6000
     5     AC_MGR              Accounting Manager    8200   16000
     4 FI_ACCOUNT                      Accountant    4200    9000
```

```
import yfinance as yf
import pandas as pd
import matplotlib.pyplot as plt
# Define the ticker symbol for Alphabet Inc. (GOOGL)
ticker = 'GOOGL'
# Define the start and end dates
start_date = '2023-01-01'
end_date = '2023-10-01'
# Fetch historical data from Yahoo Finance
data = yf.download(ticker, start=start_date, end=end_date)
# Create a line plot
plt.figure(figsize=(10,6))
plt.plot(data['Close'], label='Close Price')
plt.title(f'Historical Stock Prices of {ticker} between {start_date} and {end_date}')
plt.xlabel('Date')
plt.ylabel('Price (USD)')
plt.legend()

plt.show()
```

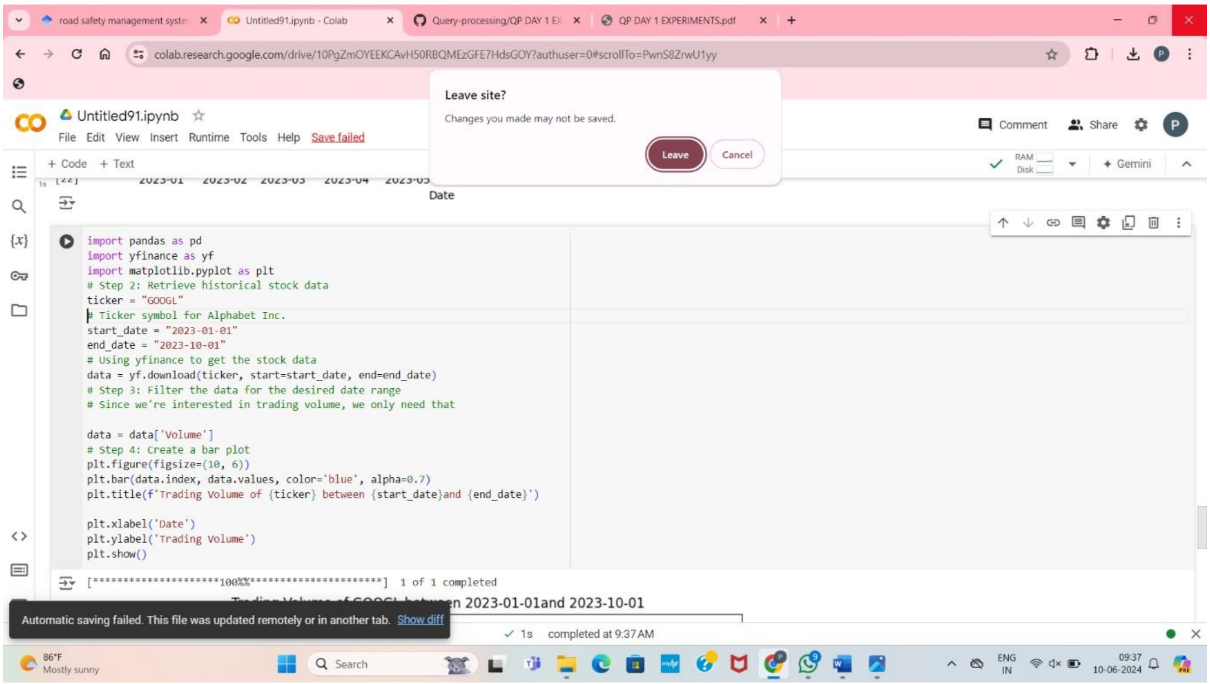
```
[*********************100%%**********************]  1 of 1 completed
```



Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
+ Code + Text
Historical Stock Prices of GOOGL between 2023-01-01 and 2023-10-01
Close Price
Price (USD)
140
130
120
110
100
90
2023-01
2023-02
2023-03
2023-04
2023-05
2023-06
2023-07
2023-08
2023-09
2023-10
Automatic saving failed. This file was updated remotely or in another tab. Show diff
completed at 9:32 AM

5.Write a Pandas program to create a bar plot of the trading volume of Alphabet Inc. stock between two specific dates.

CODE:



Leave site?

Changes you made may not be saved.

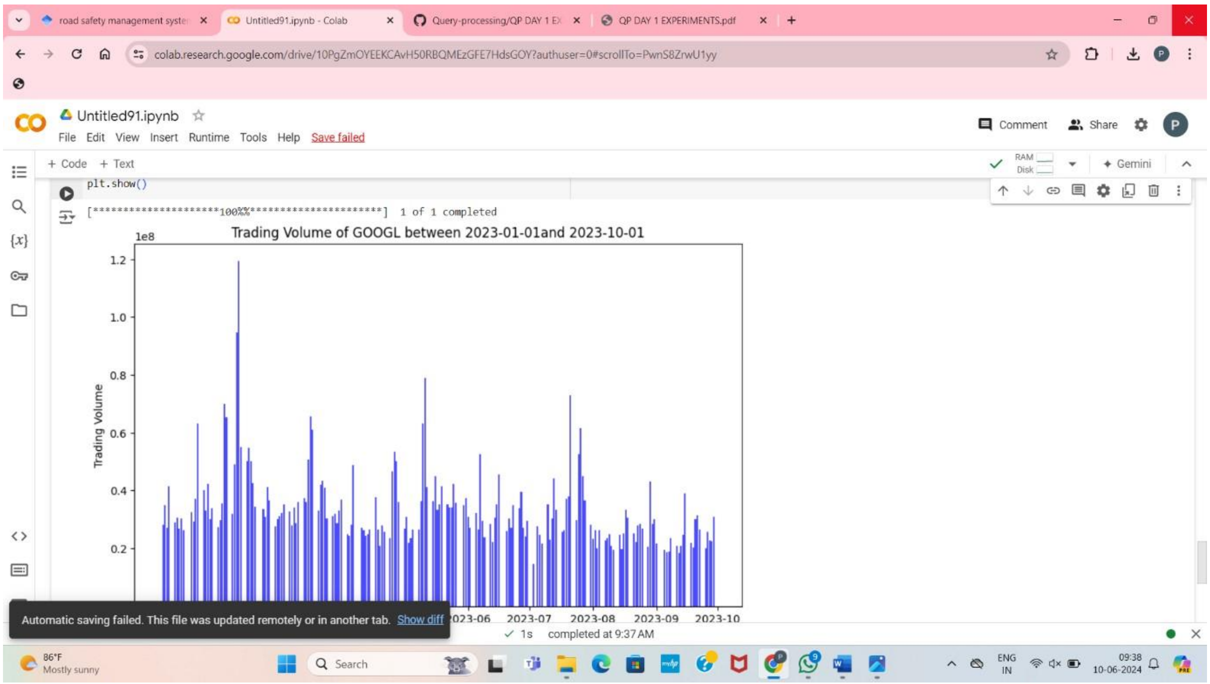
```
import pandas as pd
import yfinance as yf
import matplotlib.pyplot as plt
# Step 2: Retrieve historical stock data
ticker = "GOOGL"
# Ticker symbol for Alphabet Inc.
start_date = "2023-01-01"
end_date = "2023-10-01"
# Using yfinance to get the stock data
data = yf.download(ticker, start=start_date, end=end_date)
# Step 3: Filter the data for the desired date range
# Since we're interested in trading volume, we only need that

data = data['Volume']
# Step 4: Create a bar plot
plt.figure(figsize=(10, 6))
plt.bar(data.index, data.values, color='blue', alpha=0.7)
plt.title(f'Trading Volume of {ticker} between {start_date}and {end_date}')

plt.xlabel('Date')
plt.ylabel('Trading Volume')
plt.show()
```

```
[*********************100%%**********************]  1 of 1 completed
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

road safety management syste
Untitled91.ipynb - Colab
Query-processing/QP DAY 1 EX
QP DAY 1 EXPERIMENTS.pdf
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=PwnS8ZrwU1yy
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
Comment
Share
+ Code + Text
Leave
Cancel
Date
✓ 1s completed at 9:37 AM
86°F Mostly sunny
Search
ENG IN
09:37 10-06-2024

Output:



```
plt.show()
```

```
[*********************100%%**********************]  1 of 1 completed
```

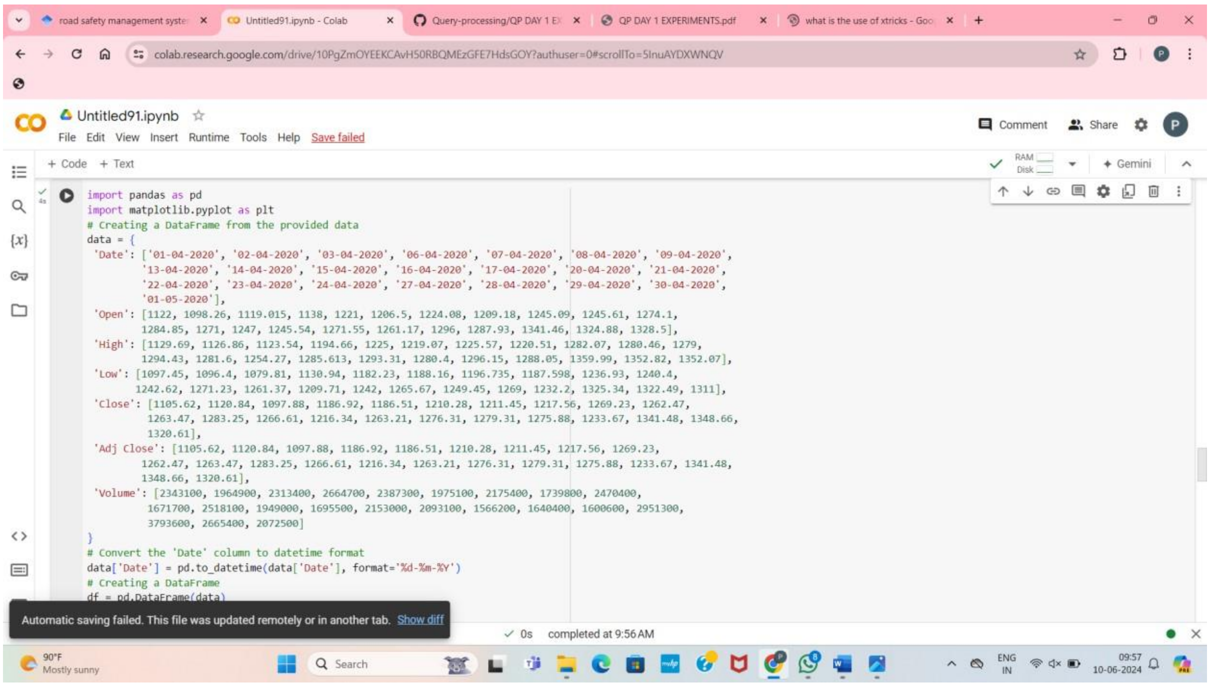
Automatic saving failed. This file was updated remotely or in another tab. Show diff

Trading Volume of GOOGL between 2023-01-01and 2023-10-01
1e8
1.2
1.0
0.8
0.6
0.4
0.2
Trading Volume
2023-06
2023-07
2023-08
2023-09
2023-10
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
✓ 1s completed at 9:37 AM
86°F Mostly sunny
09:38 10-06-2024

6.Write a Pandas program to create a scatter plot of the trading volume/stock prices of Alphabet Inc. stock between two specific dates.

alphabet_stock_data:

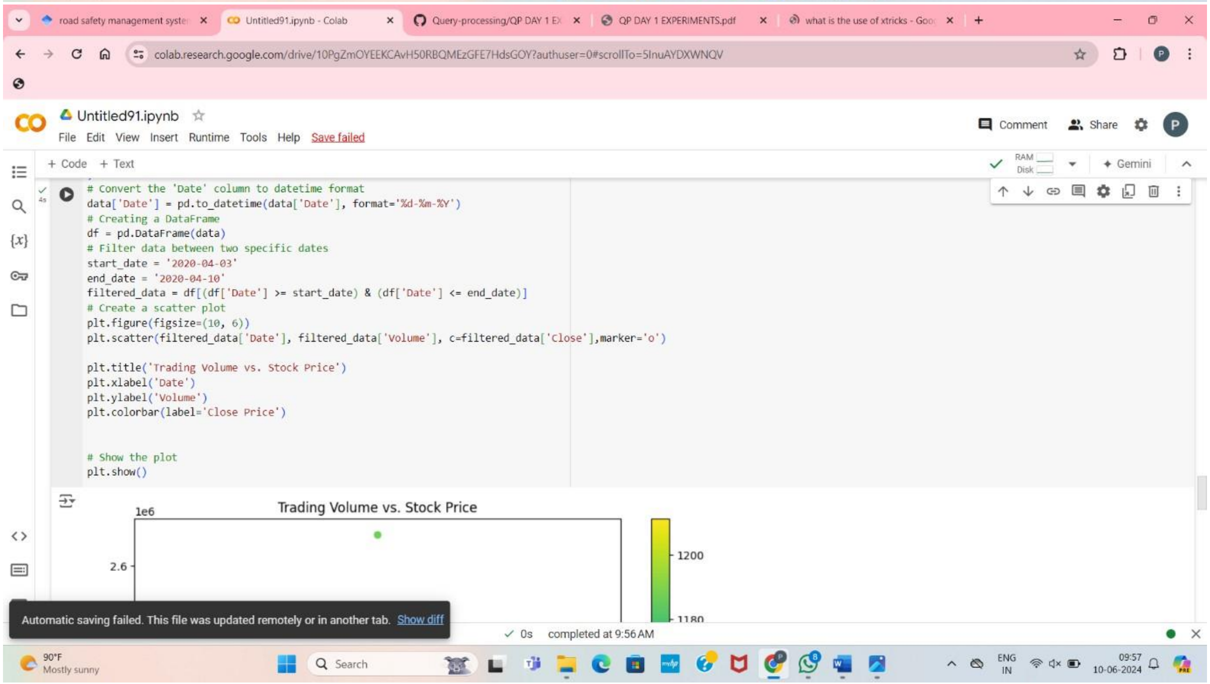
CODE:



```
import pandas as pd
import matplotlib.pyplot as plt
# Creating a DataFrame from the provided data
data = {
 'Date': ['01-04-2020', '02-04-2020', '03-04-2020', '06-04-2020', '07-04-2020', '08-04-2020', '09-04-2020',
         '13-04-2020', '14-04-2020', '15-04-2020', '16-04-2020', '17-04-2020', '20-04-2020', '21-04-2020',
         '22-04-2020', '23-04-2020', '24-04-2020', '27-04-2020', '28-04-2020', '29-04-2020', '30-04-2020',
         '01-05-2020'],
 'Open': [1122, 1098.26, 1119.015, 1138, 1221, 1206.5, 1224.08, 1209.18, 1245.09, 1245.61, 1274.1,
         1284.85, 1271, 1247, 1245.54, 1271.55, 1261.17, 1296, 1287.93, 1341.46, 1324.88, 1328.5],
 'High': [1129.69, 1126.86, 1123.54, 1194.66, 1225, 1219.07, 1225.57, 1220.51, 1282.07, 1280.46, 1279,
         1294.43, 1281.6, 1254.27, 1285.613, 1293.31, 1280.4, 1296.15, 1288.05, 1359.99, 1352.82, 1352.07],
 'Low': [1097.45, 1096.4, 1079.81, 1130.94, 1182.23, 1188.16, 1196.735, 1187.598, 1236.93, 1240.4,
        1242.62, 1271.23, 1261.37, 1209.71, 1242, 1265.67, 1249.45, 1269, 1232.2, 1325.34, 1322.49, 1311],
 'Close': [1105.62, 1120.84, 1097.88, 1186.92, 1186.51, 1210.28, 1211.45, 1217.56, 1269.23, 1262.47,
          1263.47, 1283.25, 1266.61, 1216.34, 1263.21, 1276.31, 1279.31, 1275.88, 1233.67, 1341.48, 1348.66,
          1320.61],
 'Adj Close': [1105.62, 1120.84, 1097.88, 1186.92, 1186.51, 1210.28, 1211.45, 1217.56, 1269.23,
         1262.47, 1263.47, 1283.25, 1266.61, 1216.34, 1263.21, 1276.31, 1279.31, 1275.88, 1233.67, 1341.48,
         1348.66, 1320.61],
 'Volume': [2343100, 1964900, 2313400, 2664700, 2387300, 1975100, 2175400, 1739800, 2470400,
          1671700, 2518100, 1949000, 1695500, 2153000, 2093100, 1566200, 1640400, 1600600, 2951300,
          3793600, 2665400, 2072500]
}
# Convert the 'Date' column to datetime format
data['Date'] = pd.to_datetime(data['Date'], format='%d-%m-%Y')
# Creating a DataFrame
df = pd.DataFrame(data)
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 9:56 AM

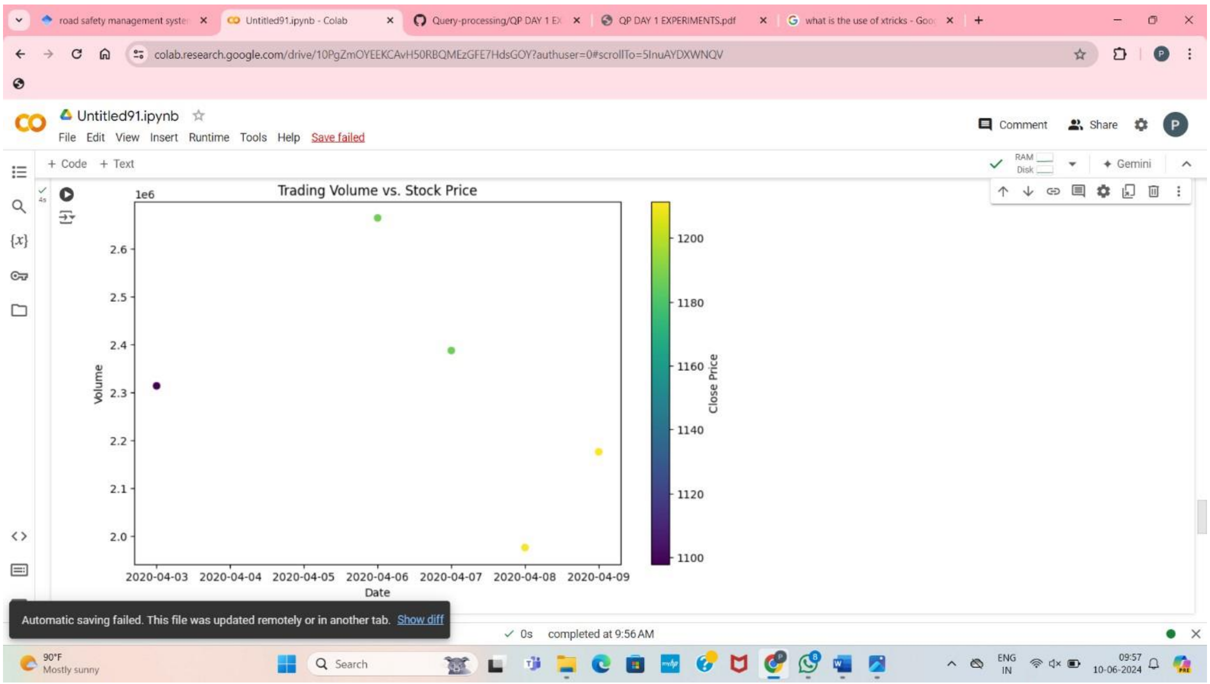


```
# Convert the 'Date' column to datetime format
data['Date'] = pd.to_datetime(data['Date'], format='%d-%m-%Y')
# Creating a DataFrame
df = pd.DataFrame(data)
# Filter data between two specific dates
start_date = '2020-04-03'
end_date = '2020-04-10'
filtered_data = df[(df['Date'] >= start_date) & (df['Date'] <= end_date)]
# Create a scatter plot
plt.figure(figsize=(10, 6))
plt.scatter(filtered_data['Date'], filtered_data['Volume'], c=filtered_data['Close'],marker='o')

plt.title('Trading Volume vs. Stock Price')
plt.xlabel('Date')
plt.ylabel('Volume')
plt.colorbar(label='Close Price')


# Show the plot
plt.show()
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 9:56 AM

OUTPUT:



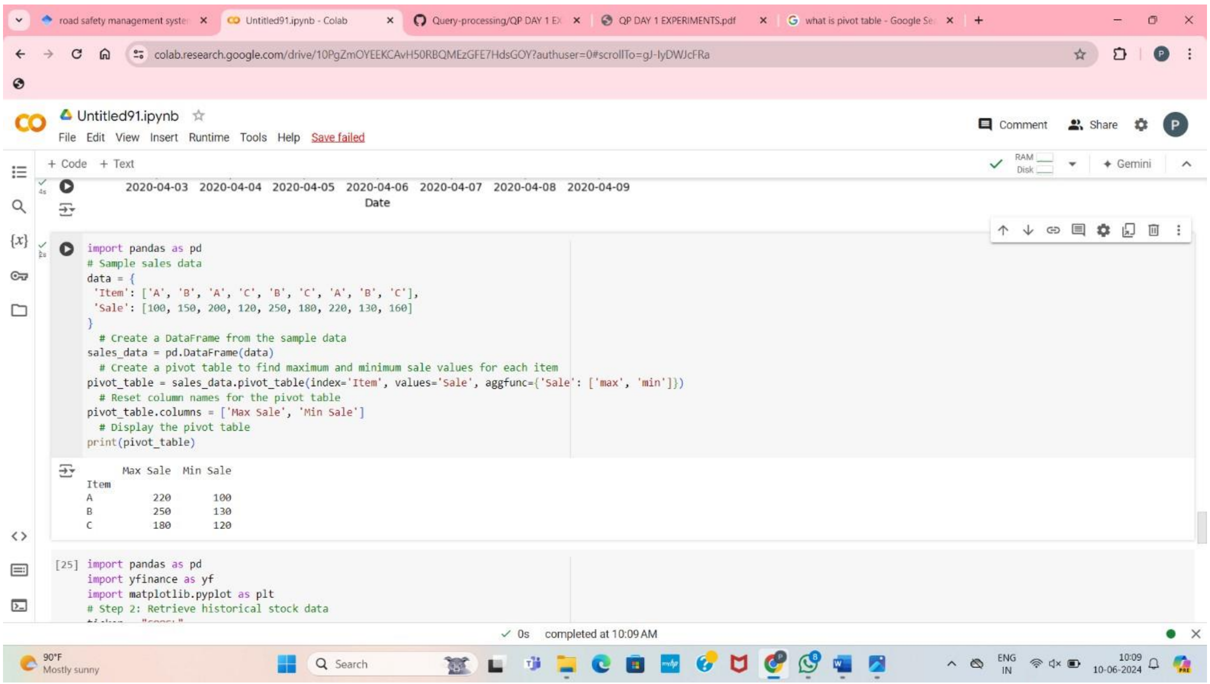
Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 9:56 AM

7.Write a Pandas program to create a Pivot table and find the maximum

and minimum sale value of the items.(refer sales_data table)

CODE &OUTPUT:



```
import pandas as pd
# Sample sales data
data = {
 'Item': ['A', 'B', 'A', 'C', 'B', 'C', 'A', 'B', 'C'],
 'Sale': [100, 150, 200, 120, 250, 180, 220, 130, 160]
}
  # Create a DataFrame from the sample data
sales_data = pd.DataFrame(data)
  # Create a pivot table to find maximum and minimum sale values for each item
pivot_table = sales_data.pivot_table(index='Item', values='Sale', aggfunc={'Sale': ['max', 'min']})
  # Reset column names for the pivot table
pivot_table.columns = ['Max Sale', 'Min Sale']
  # Display the pivot table
print(pivot_table)
```

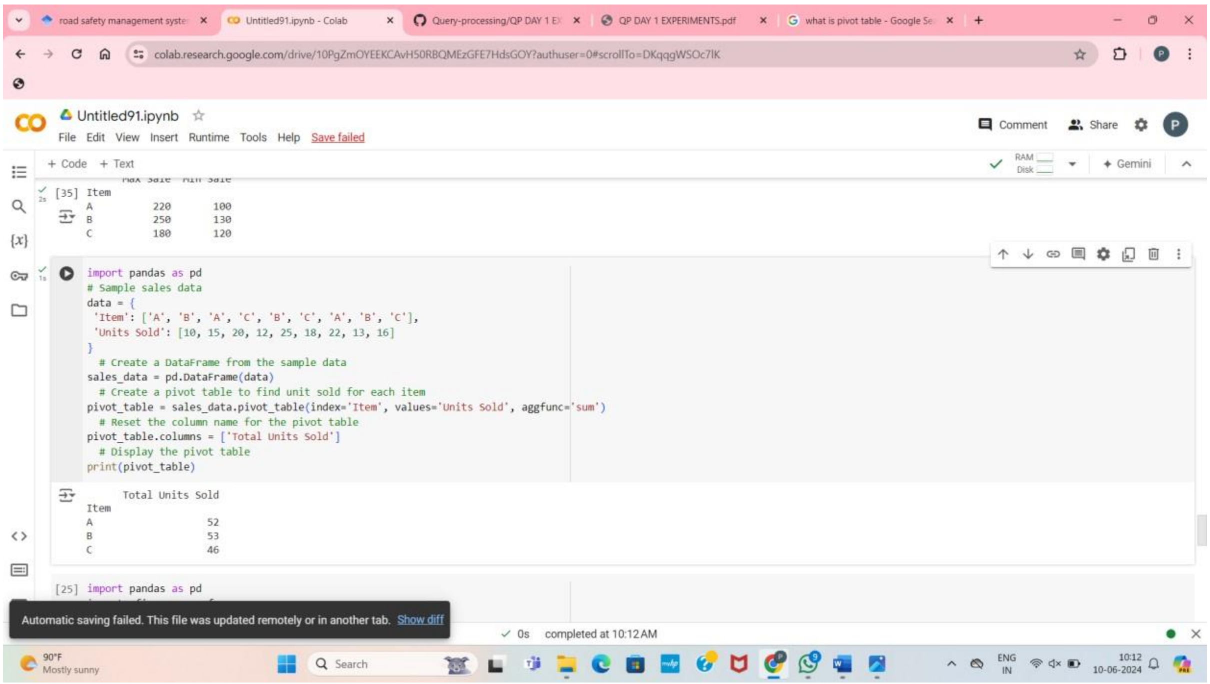
```
      Max Sale  Min Sale
Item
A          220       100
B          250       130
C          180       120
```

```
[25] import pandas as pd
     import yfinance as yf
     import matplotlib.pyplot as plt
     # Step 2: Retrieve historical stock data
```

✓ 0s completed at 10:09 AM

2. Write a Pandas program to create a Pivot table and find the item wise unit sold. .(refer sales_data table)

CODE AND OUTPUT:



```
       Max Sale  Min Sale
[35] Item
     A          220       100
     B          250       130
     C          180       120
```

```
import pandas as pd
# Sample sales data
data = {
 'Item': ['A', 'B', 'A', 'C', 'B', 'C', 'A', 'B', 'C'],
 'Units Sold': [10, 15, 20, 12, 25, 18, 22, 13, 16]
}
  # Create a DataFrame from the sample data
sales_data = pd.DataFrame(data)
  # Create a pivot table to find unit sold for each item
pivot_table = sales_data.pivot_table(index='Item', values='Units Sold', aggfunc='sum')
  # Reset the column name for the pivot table
pivot_table.columns = ['Total Units Sold']
  # Display the pivot table
print(pivot_table)
```

```
      Total Units Sold
Item
A                   52
B                   53
C                   46
```

```
[25] import pandas as pd
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

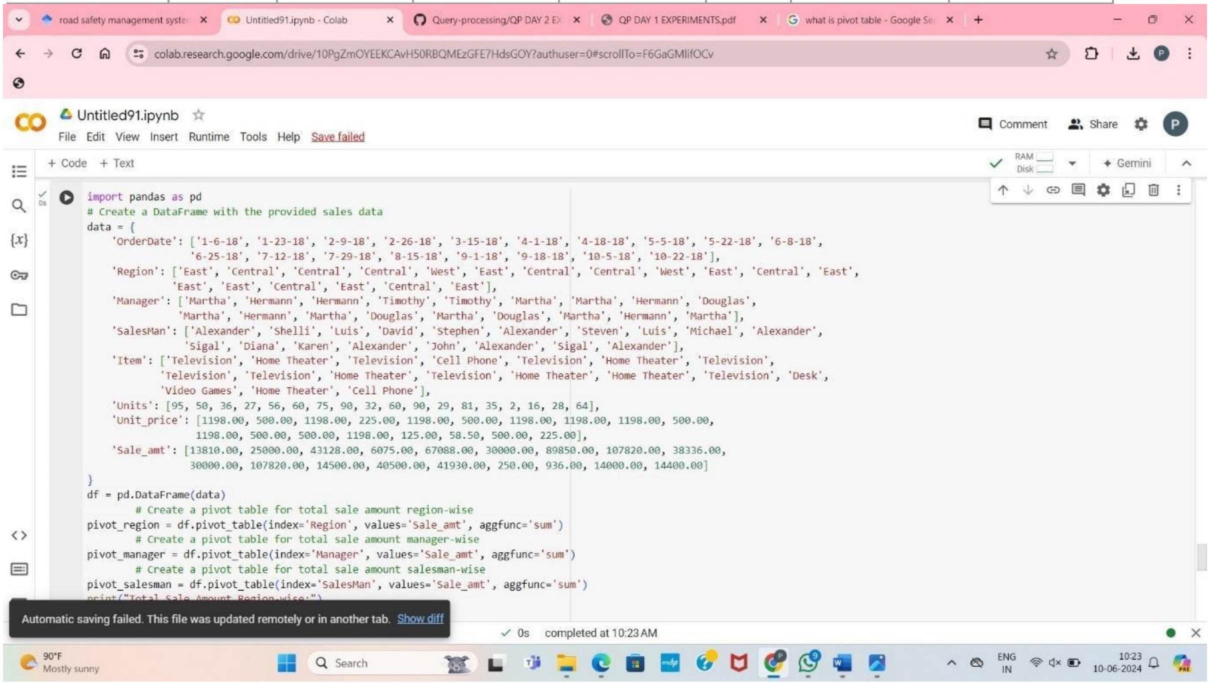
✓ 0s completed at 10:12 AM

3. Write a Pandas program to create a Pivot table and find the total sale amount region wise, manager wise, sales man wise. .(refer sales_data table)

Sales_data:

OrderDate	Region	Manager	SalesMan	Item	Units	Unit_price	Sale_amt
1-6-18	East	Martha	Alexander	Television	95	1,198.00	1,13,810.00
1-23-18	Central	Hermann	Shelli	Home Theater	50	500.00	25,000.00
2-9-18	Central	Hermann	Luis	Television	36	1,198.00	43,128.00
2-26-18	Central	Timothy	David	Cell Phone	27	225.00	6,075.00
3-15-18	West	Timothy	Stephen	Television	56	1,198.00	67,088.00
4-1-18	East	Martha	Alexander	Home Theater	60	500.00	30,000.00
4-18-18	Central	Martha	Steven	Television	75	1,198.00	89,850.00
5-5-18	Central	Hermann	Luis	Television	90	1,198.00	1,07,820.00
5-22-18	West	Douglas	Michael	Television	32	1,198.00	38,336.00
6-8-18	East	Martha	Alexander	Home Theater	60	500.00	30,000.00
6-25-18	Central	Hermann	Sigal	Television	90	1,198.00	1,07,820.00
7-12-18	East	Martha	Diana	Home Theater	29	500.00	14,500.00

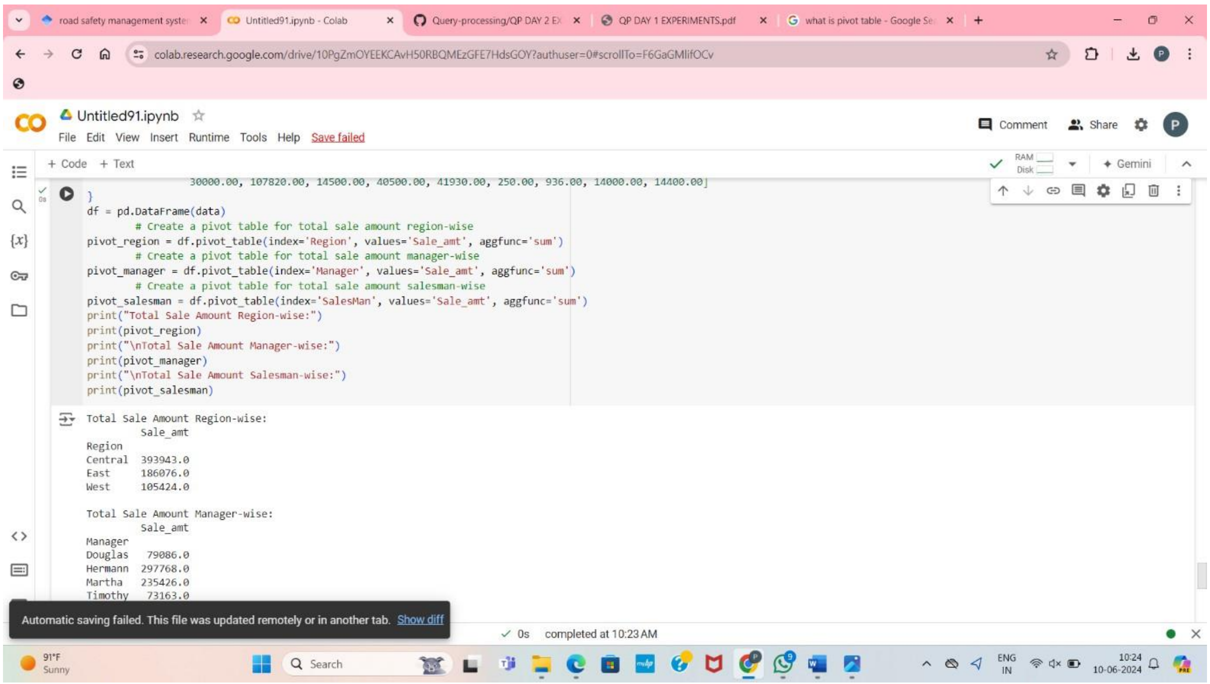
7-29-18	East	Douglas	Karen	Home Theater	81	500.00	40,500.00
8-15-18	East	Martha	Alexander	Television	35	1,198.00	41,930.00
9-1-18	Central	Douglas	John	Desk	2	125.00	250.00
9-18-18	East	Martha	Alexander	Video Games	16	58.50	936.00
10-5-18	Central	Hermann	Sigal	Home Theater	28	500.00	14,000.00
10-22-18	East	Martha	Alexander	Cell Phone	64	225.00	14,400.00



```
import pandas as pd
# Create a DataFrame with the provided sales data
data = {
    'OrderDate': ['1-6-18', '1-23-18', '2-9-18', '2-26-18', '3-15-18', '4-1-18', '4-18-18', '5-5-18', '5-22-18', '6-8-18',
                  '6-25-18', '7-12-18', '7-29-18', '8-15-18', '9-1-18', '9-18-18', '10-5-18', '10-22-18'],
    'Region': ['East', 'Central', 'Central', 'Central', 'West', 'East', 'Central', 'Central', 'West', 'East', 'Central', 'East',
               'East', 'East', 'Central', 'East', 'Central', 'East'],
    'Manager': ['Martha', 'Hermann', 'Hermann', 'Timothy', 'Timothy', 'Martha', 'Martha', 'Hermann', 'Douglas',
                'Martha', 'Hermann', 'Martha', 'Douglas', 'Martha', 'Douglas', 'Martha', 'Hermann', 'Martha'],
    'SalesMan': ['Alexander', 'Shelli', 'Luis', 'David', 'Stephen', 'Alexander', 'Steven', 'Luis', 'Michael', 'Alexander',
                 'Sigal', 'Diana', 'Karen', 'Alexander', 'John', 'Alexander', 'Sigal', 'Alexander'],
    'Item': ['Television', 'Home Theater', 'Television', 'Cell Phone', 'Television', 'Home Theater', 'Television',
             'Television', 'Television', 'Home Theater', 'Television', 'Home Theater', 'Home Theater', 'Television', 'Desk',
             'Video Games', 'Home Theater', 'Cell Phone'],
    'Units': [95, 50, 36, 27, 56, 60, 75, 90, 32, 60, 90, 29, 81, 35, 2, 16, 28, 64],
    'Unit_price': [1198.00, 500.00, 1198.00, 225.00, 1198.00, 500.00, 1198.00, 1198.00, 1198.00, 500.00,
                   1198.00, 500.00, 500.00, 1198.00, 125.00, 58.50, 500.00, 225.00],
    'Sale_amt': [13810.00, 25000.00, 43128.00, 6075.00, 67088.00, 30000.00, 89850.00, 107820.00, 38336.00,
                 30000.00, 107820.00, 14500.00, 40500.00, 41930.00, 250.00, 936.00, 14000.00, 14400.00]
}
df = pd.DataFrame(data)
        # Create a pivot table for total sale amount region-wise
pivot_region = df.pivot_table(index='Region', values='Sale_amt', aggfunc='sum')
        # Create a pivot table for total sale amount manager-wise
pivot_manager = df.pivot_table(index='Manager', values='Sale_amt', aggfunc='sum')
        # Create a pivot table for total sale amount salesman-wise
pivot_salesman = df.pivot_table(index='SalesMan', values='Sale_amt', aggfunc='sum')
print("Total Sale Amount Region-wise:")
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 10:23 AM


road safety management syste
Untitled91.ipynb - Colab
Query-processing/QP DAY 2 EX
QP DAY 1 EXPERIMENTS.pdf
what is pivot table - Google Se
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=F6GaGMIifOCv
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
Comment Share
+ Code + Text
RAM Disk
Gemini
91°F Sunny
Search
ENG IN
10:24 10-06-2024

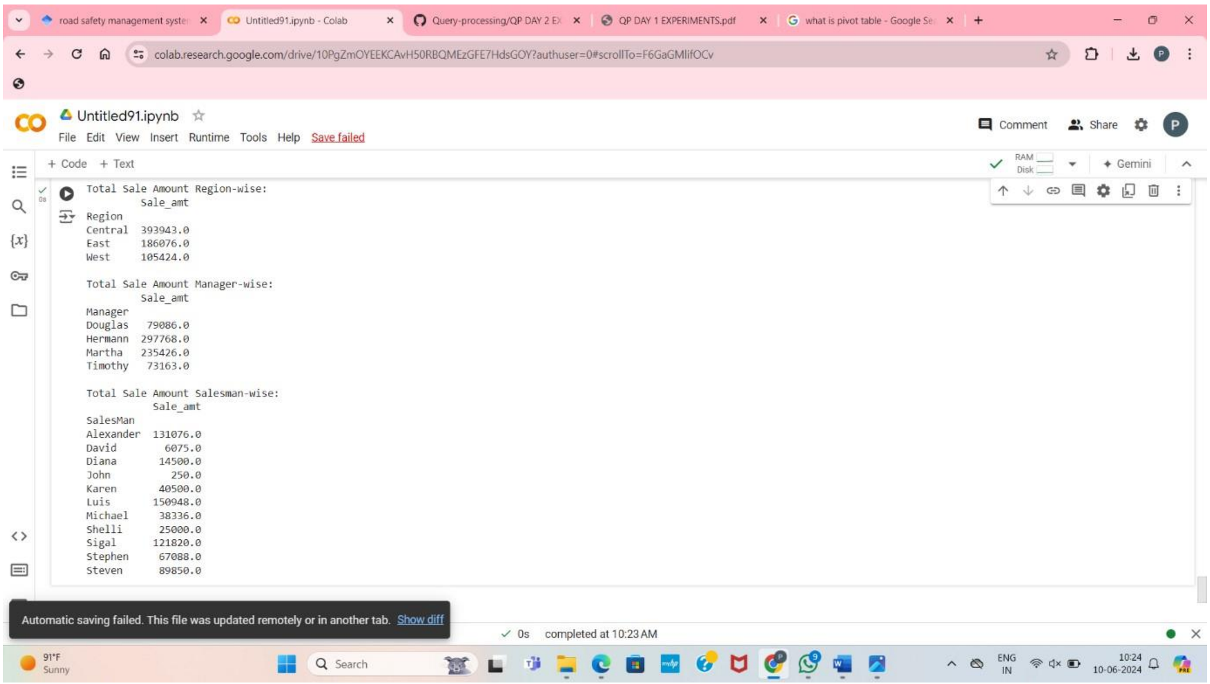
```
               30000.00, 107820.00, 14500.00, 40500.00, 41930.00, 250.00, 936.00, 14000.00, 14400.00]
}
df = pd.DataFrame(data)
        # Create a pivot table for total sale amount region-wise
pivot_region = df.pivot_table(index='Region', values='Sale_amt', aggfunc='sum')
        # Create a pivot table for total sale amount manager-wise
pivot_manager = df.pivot_table(index='Manager', values='Sale_amt', aggfunc='sum')
        # Create a pivot table for total sale amount salesman-wise
pivot_salesman = df.pivot_table(index='SalesMan', values='Sale_amt', aggfunc='sum')
print("Total Sale Amount Region-wise:")
print(pivot_region)
print("\nTotal Sale Amount Manager-wise:")
print(pivot_manager)
print("\nTotal Sale Amount Salesman-wise:")
print(pivot_salesman)
```

```
Total Sale Amount Region-wise:
         Sale_amt
Region
Central  393943.0
East     186076.0
West     105424.0

Total Sale Amount Manager-wise:
         Sale_amt
Manager
Douglas   79086.0
Hermann  297768.0
Martha   235426.0
Timothy   73163.0
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 10:23 AM

OUTPUT:


road safety management syste
Untitled91.ipynb - Colab
Query-processing/QP DAY 2 EX
QP DAY 1 EXPERIMENTS.pdf
what is pivot table - Google Se
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=F6GaGMIifOCv
Untitled91.ipynb
File Edit View Insert Runtime Tools Help Save failed
Comment Share
+ Code + Text
RAM Disk
Gemini
91°F Sunny
Search
ENG IN
10:24 10-06-2024

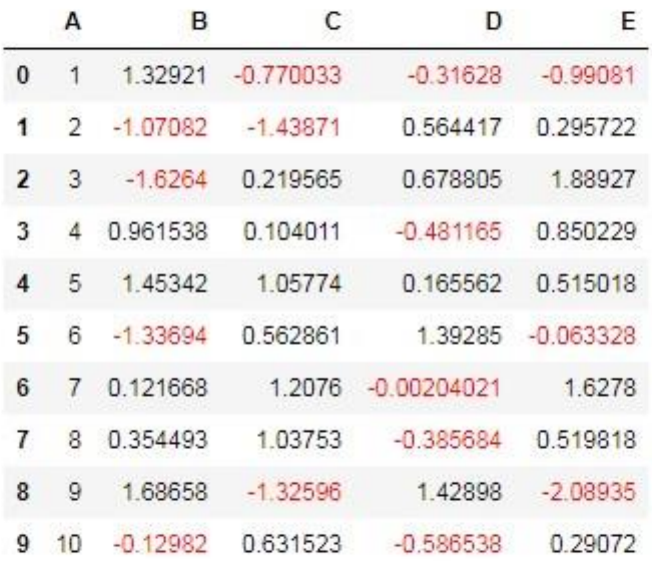
```
Total Sale Amount Region-wise:
         Sale_amt
Region
Central  393943.0
East     186076.0
West     105424.0

Total Sale Amount Manager-wise:
         Sale_amt
Manager
Douglas   79086.0
Hermann  297768.0
Martha   235426.0
Timothy   73163.0

Total Sale Amount Salesman-wise:
           Sale_amt
SalesMan
Alexander  131076.0
David        6075.0
Diana       14500.0
John          250.0
Karen       40500.0
Luis       150948.0
Michael     38336.0
Shelli      25000.0
Sigal      121820.0
Stephen     67088.0
Steven      89850.0
```

Automatic saving failed. This file was updated remotely or in another tab. Show diff

✓ 0s completed at 10:23 AM

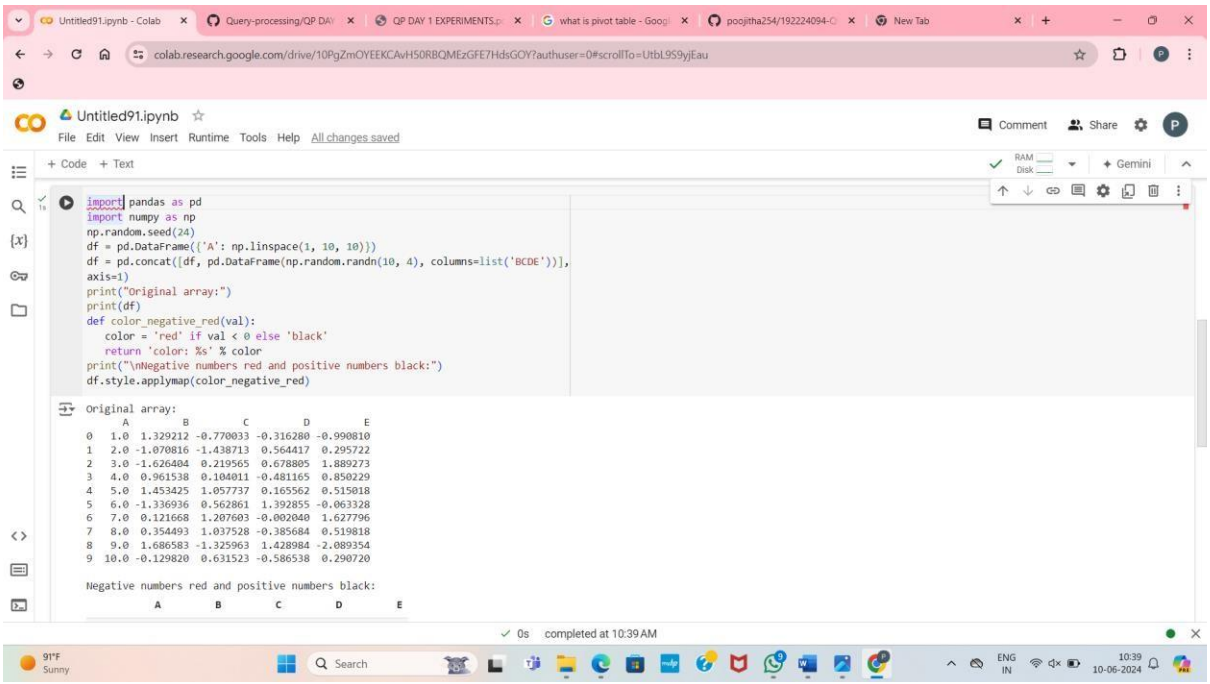
10.Create a dataframe of ten rows, four columns with random values. Write a Pandas program to highlight the negative numbers red and positive numbers black.

Expected Output:



	A	B	C	D	E
0	1	1.32921	-0.770033	-0.31628	-0.99081
1	2	-1.07082	-1.43871	0.564417	0.295722
2	3	-1.6264	0.219565	0.678805	1.88927
3	4	0.961538	0.104011	-0.481165	0.850229
4	5	1.45342	1.05774	0.165562	0.515018
5	6	-1.33694	0.562861	1.39285	-0.063328
6	7	0.121668	1.2076	-0.00204021	1.6278
7	8	0.354493	1.03753	-0.385684	0.519818
8	9	1.68658	-1.32596	1.42898	-2.08935
9	10	-0.12982	0.631523	-0.586538	0.29072

CODE:

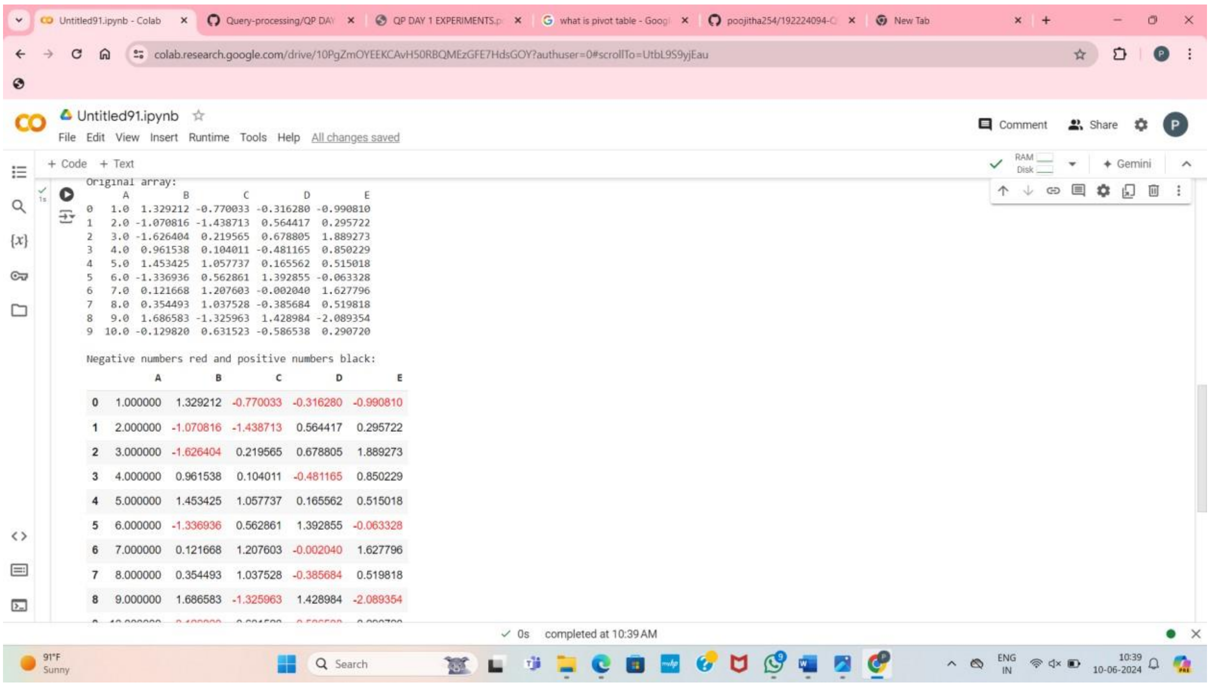


Untitled91.ipynb - Colab
colab.research.google.com/drive/10PgZmOYEEKCAvH50RBQMEzGFE7HdsGOY?authuser=0#scrollTo=UtbL9S9yjEau
Untitled91.ipynb
File Edit View Insert Runtime Tools Help All changes saved
+ Code + Text
Comment
Share
Gemini
0s completed at 10:39 AM

```
import pandas as pd
import numpy as np
np.random.seed(24)
df = pd.DataFrame({'A': np.linspace(1, 10, 10)})
df = pd.concat([df, pd.DataFrame(np.random.randn(10, 4), columns=list('BCDE'))],
axis=1)
print("Original array:")
print(df)
def color_negative_red(val):
   color = 'red' if val < 0 else 'black'
   return 'color: %s' % color
print("\nNegative numbers red and positive numbers black:")
df.style.applymap(color_negative_red)
```

```
Original array:
      A         B         C         D         E
0   1.0  1.329212 -0.770033 -0.316280 -0.990810
1   2.0 -1.070816 -1.438713  0.564417  0.295722
2   3.0 -1.626404  0.219565  0.678805  1.889273
3   4.0  0.961538  0.104011 -0.481165  0.850229
4   5.0  1.453425  1.057737  0.165562  0.515018
5   6.0 -1.336936  0.562861  1.392855 -0.063328
6   7.0  0.121668  1.207603 -0.002040  1.627796
7   8.0  0.354493  1.037528 -0.385684  0.519818
8   9.0  1.686583 -1.325963  1.428984 -2.089354
9  10.0 -0.129820  0.631523 -0.586538  0.290720

Negative numbers red and positive numbers black:
          A         B         C         D         E
```

OUTPUT:



```
Original array:
      A         B         C         D         E
0   1.0  1.329212 -0.770033 -0.316280 -0.990810
1   2.0 -1.070816 -1.438713  0.564417  0.295722
2   3.0 -1.626404  0.219565  0.678805  1.889273
3   4.0  0.961538  0.104011 -0.481165  0.850229
4   5.0  1.453425  1.057737  0.165562  0.515018
5   6.0 -1.336936  0.562861  1.392855 -0.063328
6   7.0  0.121668  1.207603 -0.002040  1.627796
7   8.0  0.354493  1.037528 -0.385684  0.519818
8   9.0  1.686583 -1.325963  1.428984 -2.089354
9  10.0 -0.129820  0.631523 -0.586538  0.290720

Negative numbers red and positive numbers black:
```

	A	B	C	D	E
0	1.000000	1.329212	-0.770033	-0.316280	-0.990810
1	2.000000	-1.070816	-1.438713	0.564417	0.295722
2	3.000000	-1.626404	0.219565	0.678805	1.889273
3	4.000000	0.961538	0.104011	-0.481165	0.850229
4	5.000000	1.453425	1.057737	0.165562	0.515018
5	6.000000	-1.336936	0.562861	1.392855	-0.063328
6	7.000000	0.121668	1.207603	-0.002040	1.627796
7	8.000000	0.354493	1.037528	-0.385684	0.519818
8	9.000000	1.686583	-1.325963	1.428984	-2.089354