Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

Experiment - 13

1. **Create a CSV file named employees.csv with columns Name, Age, Department, Salary. Load the data, filter employees older than 30, group by Department, and calculate the average, minimum, and maximum salary for each department.**

main.py

```
# Exp-13: Q1 - Employees DataFrame Analysis (No CSV file needed)
# Create DataFrame, filter, group, and perform aggregation in memory

import pandas as pd

# Step 1: Create sample employee data
employees = pd.DataFrame({
    "Name": ["Alice", "Bob", "Charlie", "David", "Eva", "Frank", "Grace",
        "Henry"],
    "Age": [25, 32, 45, 28, 36, 41, 29, 39],
    "Department": ["HR", "IT", "Finance", "HR", "IT", "Finance", "IT", "HR"],
    "Salary": [40000, 55000, 75000, 42000, 65000, 80000, 60000, 47000]
})

print("=== Employees Data ===")
print(employees)

# Step 2: Filter employees older than 30
filtered = employees[employees["Age"] > 30]
print("\n=== Employees older than 30 ===")
print(filtered)

# Step 3: Group by Department and calculate average, min, and max salary
salary_stats = filtered.groupby("Department")["Salary"].agg(["mean", "min",
    "max"]).reset_index()
print("\n=== Salary Stats by Department ===")
print(salary_stats)
```

Output

```
=== Employees Data ===
      Name  Age Department  Salary
0    Alice   25         HR   40000
1      Bob   32         IT   55000
2  Charlie   45    Finance   75000
3    David   28         HR   42000
4      Eva   36         IT   65000
5    Frank   41    Finance   80000
6    Grace   29         IT   60000
7    Henry   39         HR   47000

=== Employees older than 30 ===
      Name  Age Department  Salary
1      Bob   32         IT   55000
2  Charlie   45    Finance   75000
4      Eva   36         IT   65000
5    Frank   41    Finance   80000
7    Henry   39         HR   47000

=== Salary Stats by Department ===
  Department     mean    min    max
0    Finance  77500.0  75000  80000
1         HR  47000.0  47000  47000
2         IT  60000.0  55000  65000

=== Code Execution Successful ===
```

2. **Create a CSV file named sales.csv with columns SaleID, Product, Amount, Customer. Load the data, filter sales with amount greater than 50,000, group by Product, and calculate total and average sales per product.**

main.py

```
# Exp-13 Q2: Sales Analysis (No pandas)
from collections import defaultdict
sales = [
    {"SaleID":101, "Product":"Laptop", "Amount":75000, "Customer":"A"},
    {"SaleID":102, "Product":"Mouse", "Amount":2000, "Customer":"B"},
    {"SaleID":103, "Product":"Keyboard", "Amount":50000, "Customer":"C"},
    {"SaleID":104, "Product":"Laptop", "Amount":120000, "Customer":"D"},
    {"SaleID":105, "Product":"Mouse", "Amount":60000, "Customer":"E"}
]
# Filter sales > 50000
filtered = [s for s in sales if s["Amount"] > 50000]
# Group by Product and calculate total and average
grouped = defaultdict(list)
for s in filtered:
    grouped[s["Product"]].append(s["Amount"])
stats = {p: {"total": sum(v), "average": sum(v)/len(v)} for p,v in grouped.items
    ()}
print("Sales > 50,000:")
for s in filtered:
    print(s)
print("\nTotal & Average Sales per Product:")
for prod, stat in stats.items():
    print(prod, stat)
```

Output

```
Sales > 50,000:
{'SaleID': 101, 'Product': 'Laptop', 'Amount': 75000, 'Customer': 'A'}
{'SaleID': 104, 'Product': 'Laptop', 'Amount': 120000, 'Customer': 'D'}
{'SaleID': 105, 'Product': 'Mouse', 'Amount': 60000, 'Customer': 'E'}

Total & Average Sales per Product:
Laptop {'total': 195000, 'average': 97500.0}
Mouse {'total': 60000, 'average': 60000.0}

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

3. Create a CSV file named student marks.csv with columns StudentID, Name, Maths, Science, English. Load the data, filter students with Maths marks above 80, group by student name, and calculate total and average marks per student.

main.py

```
# Exp-13 Q3: Student Marks Analysis (No pandas)
students = [
    {"StudentID":1,"Name":"Alice","Maths":85,"Science":80,"English":78},
    {"StudentID":2,"Name":"Bob","Maths":72,"Science":68,"English":74},
    {"StudentID":3,"Name":"Charlie","Maths":90,"Science":95,"English":88},
    {"StudentID":4,"Name":"David","Maths":60,"Science":55,"English":65}
]

# Filter students with Maths > 80
filtered = [s for s in students if s["Maths"] > 80]

# Calculate total and average marks
stats = {}
for s in filtered:
    total = s["Maths"] + s["Science"] + s["English"]
    average = total / 3
    stats[s["Name"]] = {"Total": total, "Average": average}

print("Students with Maths > 80:")
for s in filtered:
    print(s)

print("\nTotal & Average Marks per Student:")
for name, stat in stats.items():
    print(name, stat)
```

Output

```
Sales > 50,000:
{'SaleID': 101, 'Product': 'Laptop', 'Amount': 75000, 'Customer': 'A'}
{'SaleID': 104, 'Product': 'Laptop', 'Amount': 120000, 'Customer': 'D'}
{'SaleID': 105, 'Product': 'Mouse', 'Amount': 60000, 'Customer': 'E'}

Total & Average Sales per Product:
Laptop {'total': 195000, 'average': 97500.0}
Mouse {'total': 60000, 'average': 60000.0}

=== Code Execution Successful ===
```

4. Create a CSV file named products.csv with columns ProductID, ProductName, Category, Stock. Load the data, filter products with stock less than 50, group by Category, and calculate the total and average stock per category.

main.py

```
# Exp-13 Q4: Products Stock Analysis (No pandas)
from collections import defaultdict
products = [
    {"ProductID":1,"ProductName":"Pen","Category":"Stationery","Stock":100},
    {"ProductID":2,"ProductName":"Notebook","Category":"Stationery","Stock":30},
    {"ProductID":3,"ProductName":"Eraser","Category":"Stationery","Stock":25},
    {"ProductID":4,"ProductName":"Marker","Category":"Stationery","Stock":80},
    {"ProductID":5,"ProductName":"Book","Category":"Book","Stock":40}
]
# Filter products with Stock < 50
filtered = [p for p in products if p["Stock"] < 50]
# Group by Category and calculate total & average stock
grouped = defaultdict(list)
for p in filtered:
    grouped[p["Category"]].append(p["Stock"])
stats = {cat: {"total": sum(v), "average": sum(v)/len(v)} for cat,v in grouped
    .items()}
print("Products with Stock < 50:")
for p in filtered:
    print(p)
print("\nTotal & Average Stock per Category:")
for cat, stat in stats.items():
    print(cat, stat)
```

Output

```
Products with Stock < 50:
{'ProductID': 2, 'ProductName': 'Notebook', 'Category': 'Stationery', 'Stock': 30}
{'ProductID': 3, 'ProductName': 'Eraser', 'Category': 'Stationery', 'Stock': 25}
{'ProductID': 5, 'ProductName': 'Book', 'Category': 'Book', 'Stock': 40}

Total & Average Stock per Category:
Stationery {'total': 55, 'average': 27.5}
Book {'total': 40, 'average': 40.0}

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

5. Create a CSV file named transactions.csv with columns TransactionID, Customer, Amount, Date. Load the data, filter transactions with amount greater than 1000, group by Customer, and calculate total and average spending per customer.

main.py

```
# Exp-13 Q5: Transactions Analysis
from collections import defaultdict
transactions = [
    {"TransactionID":1,"Customer":"A","Amount":1200,"Date":"2025-10-01"},
    {"TransactionID":2,"Customer":"B","Amount":800,"Date":"2025-10-02"},
    {"TransactionID":3,"Customer":"A","Amount":1500,"Date":"2025-10-03"},
    {"TransactionID":4,"Customer":"C","Amount":2000,"Date":"2025-10-04"},
    {"TransactionID":5,"Customer":"B","Amount":500,"Date":"2025-10-05"}
]
# Filter transactions > 1000
filtered = [t for t in transactions if t["Amount"] > 1000]
# Group by Customer and calculate total & average
grouped = defaultdict(list)
for t in filtered:
    grouped[t["Customer"]].append(t["Amount"])

stats = {c: {"total": sum(v), "average": sum(v)/len(v)} for c,v in grouped.items()}
print("Transactions > 1000:")
for t in filtered:
    print(t)
print("\nTotal & Average Spending per Customer:")
for customer, stat in stats.items():
    print(customer, stat)
```

Output

```
Transactions > 1000:
{'TransactionID': 1, 'Customer': 'A', 'Amount': 1200, 'Date': '2025-10-01'}
{'TransactionID': 3, 'Customer': 'A', 'Amount': 1500, 'Date': '2025-10-03'}
{'TransactionID': 4, 'Customer': 'C', 'Amount': 2000, 'Date': '2025-10-04'}

Total & Average Spending per Customer:
A {'total': 2700, 'average': 1350.0}
C {'total': 2000, 'average': 2000.0}

=== Code Execution Successful ===
```

6. Create a CSV file named employee_salary.csv with columns EmpID, Name, Department, Salary. Load the data, filter employees with salary above 60,000, group by Department, and calculate mean, max, and min salary per department.

main.py

```
# Exp-13 Q6: Employee Salary Analysis
from collections import defaultdict
employees = [
    {"EmpID":1,"Name":"Alice","Department":"HR","Salary":50000},
    {"EmpID":2,"Name":"Bob","Department":"IT","Salary":70000},
    {"EmpID":3,"Name":"Charlie","Department":"Finance","Salary":60000},
    {"EmpID":4,"Name":"David","Department":"HR","Salary":45000},
    {"EmpID":5,"Name":"Eva","Department":"IT","Salary":80000}
]
# Filter salary > 60000
filtered = [e for e in employees if e["Salary"] > 60000]
# Group by Department and calculate mean, max, min
grouped = defaultdict(list)
for e in filtered:
    grouped[e["Department"]].append(e["Salary"])

stats = {d: {"mean": sum(v)/len(v), "max": max(v), "min": min(v)} for d,v in grouped.items()}
print("Employees with Salary > 60000:")
for e in filtered:
    print(e)
print("\nSalary Stats per Department:")
for dept, stat in stats.items():
    print(dept, stat)
```

Output

```
Employees with Salary > 60000:
{'EmpID': 2, 'Name': 'Bob', 'Department': 'IT', 'Salary': 70000}
{'EmpID': 5, 'Name': 'Eva', 'Department': 'IT', 'Salary': 80000}

Salary Stats per Department:
IT {'mean': 75000.0, 'max': 80000, 'min': 70000}

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

7. Create a CSV file named orders.csv with columns OrderID, Customer, Product, Quantity, OrderDate. Load the data, filter orders with quantity greater than 5, group by Product, and calculate total quantity sold per product.

main.py

```
# Exp-13 Q7: Orders Analysis
from collections import defaultdict
orders = [
    {"OrderID":1,"Customer":"A","Product":"Pen","Quantity":10},
    {"OrderID":2,"Customer":"B","Product":"Notebook","Quantity":3},
    {"OrderID":3,"Customer":"A","Product":"Pen","Quantity":7},
    {"OrderID":4,"Customer":"C","Product":"Marker","Quantity":6}
]
# Filter quantity > 5
filtered = [o for o in orders if o["Quantity"] > 5]
# Group by Product and calculate total quantity
grouped = defaultdict(int)
for o in filtered:
    grouped[o["Product"]] += o["Quantity"]
print("Orders with Quantity > 5:")
for o in filtered:
    print(o)
print("\nTotal Quantity Sold per Product:")
for product, qty in grouped.items():
    print(product, qty)
```

Output

```
Orders with Quantity > 5:
{'OrderID': 1, 'Customer': 'A', 'Product': 'Pen', 'Quantity': 10}
{'OrderID': 3, 'Customer': 'A', 'Product': 'Pen', 'Quantity': 7}
{'OrderID': 4, 'Customer': 'C', 'Product': 'Marker', 'Quantity': 6}

Total Quantity Sold per Product:
Pen 17
Marker 6

=== Code Execution Successful ===
```

8. Create a CSV file named movies.csv with columns MovieID, Title, Genre, Rating. Load the data, filter movies with rating above 8, group by Genre, and calculate average, maximum, and minimum rating per genre.

main.py

```
# Exp-13 Q8: Movies Analysis
from collections import defaultdict
movies = [
    {"MovieID":1,"Title":"Movie1","Genre":"Action","Rating":8.5},
    {"MovieID":2,"Title":"Movie2","Genre":"Drama","Rating":7.9},
    {"MovieID":3,"Title":"Movie3","Genre":"Action","Rating":9.0},
    {"MovieID":4,"Title":"Movie4","Genre":"Comedy","Rating":8.2}
]
# Filter rating > 8
filtered = [m for m in movies if m["Rating"] > 8]
# Group by Genre and calculate mean, max, min
grouped = defaultdict(list)
for m in filtered:
    grouped[m["Genre"]].append(m["Rating"])
stats = {g: {"mean": sum(v)/len(v), "max": max(v), "min": min(v)} for g
    ,v in grouped.items()}
print("Movies with Rating > 8:")
for m in filtered:
    print(m)
print("\nRating Stats by Genre:")
for genre, stat in stats.items():
    print(genre, stat)
```

Output

```
Movies with Rating > 8:
{'MovieID': 1, 'Title': 'Movie1', 'Genre': 'Action', 'Rating': 8.5}
{'MovieID': 3, 'Title': 'Movie3', 'Genre': 'Action', 'Rating': 9.0}
{'MovieID': 4, 'Title': 'Movie4', 'Genre': 'Comedy', 'Rating': 8.2}

Rating Stats by Genre:
Action {'mean': 8.75, 'max': 9.0, 'min': 8.5}
Comedy {'mean': 8.2, 'max': 8.2, 'min': 8.2}

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

9. Create a CSV file named weather.csv with columns Date, City, Temperature, Humidity. Load the data, filter days with temperature above 35°C, group by City, and calculate maximum, minimum, and average temperature for each city.

main.py | Share | Run | Output | Clear

```
from collections import defaultdict
weather = [
    {"Date":"2025-10-01","City":"CityA","Temperature":36,"Humidity":50},
    {"Date":"2025-10-02","City":"CityB","Temperature":34,"Humidity":60},
    {"Date":"2025-10-03","City":"CityA","Temperature":38,"Humidity":55}
]
# Filter temperature > 35
filtered = [w for w in weather if w["Temperature"] > 35]
# Group by City and calculate max, min, average
grouped = defaultdict(list)
for w in filtered:
    grouped[w["City"]].append(w["Temperature"])
stats = {c: {"max": max(v), "min": min(v), "average": sum(v)/len(v)} for c,v in grouped.items()}
print("Days with Temperature > 35:")
for w in filtered:
    print(w)
print("\nTemperature Stats per City:")
for city, stat in stats.items():
    print(city, stat)
```

```
Days with Temperature > 35:
{'Date': '2025-10-01', 'City': 'CityA', 'Temperature': 36, 'Humidity': 50}
{'Date': '2025-10-03', 'City': 'CityA', 'Temperature': 38, 'Humidity': 55}

Temperature Stats per City:
CityA {'max': 38, 'min': 36, 'average': 37.0}

=== Code Execution Successful ===
```

10. Create a CSV file named bank transactions.csv with columns TransactionID, Customer, Type, Amount. Load the data, filter deposits, group by Customer, and calculate total, maximum, and average deposit amount per customer.

```
# Exp-13 Q10: Bank Transactions
from collections import defaultdict
bank = [
    {"TransactionID":1,"Customer":"A","Type":"Deposit","Amount":1500},
    {"TransactionID":2,"Customer":"B","Type":"Withdrawal","Amount":500},
    {"TransactionID":3,"Customer":"A","Type":"Deposit","Amount":2000},
    {"TransactionID":4,"Customer":"C","Type":"Deposit","Amount":1200}
]
# Filter deposits only
filtered = [b for b in bank if b["Type"]=="Deposit"]
# Group by Customer and calculate total, max, average
grouped = defaultdict(list)
for b in filtered:
    grouped[b["Customer"]].append(b["Amount"])
stats = {c: {"total": sum(v), "max": max(v), "average": sum(v)/len(v)} for c,v in grouped.items()}
print("Deposits Only:")
for b in filtered:
    print(b)
print("\nDeposit Stats per Customer:")
for customer, stat in stats.items():
    print(customer, stat)
```

```
Deposits Only:
{'TransactionID': 1, 'Customer': 'A', 'Type': 'Deposit', 'Amount': 1500}
{'TransactionID': 3, 'Customer': 'A', 'Type': 'Deposit', 'Amount': 2000}
{'TransactionID': 4, 'Customer': 'C', 'Type': 'Deposit', 'Amount': 1200}

Deposit Stats per Customer:
A {'total': 3500, 'max': 2000, 'average': 1750.0}
C {'total': 1200, 'max': 1200, 'average': 1200.0}

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

11. Create a CSV file named hospital.csv with columns PatientID, Name, Department, Admission Date. Load the data, filter patients admitted in the last month, group by Department, and calculate the total number of patients per department.

main.py

```
from collections import defaultdict
patients=[{"PatientID":1,"Name":"Alice","Department"
    :"Cardiology","AdmissionDate":"2025-09-15"},
        {"PatientID":2,"Name":"Bob","Department":"Neurology"
            ,"AdmissionDate":"2025-10-10"},
        {"PatientID":3,"Name":"Charlie","Department"
            :"Cardiology","AdmissionDate":"2025-10-05"},
        {"PatientID":4,"Name":"David","Department"
            :"Orthopedics","AdmissionDate":"2025-09-25"}]
filtered=[p for p in patients if p["AdmissionDate"]>="2025-09
    -30"]
grouped=defaultdict(int)
for p in filtered: grouped[p["Department"]]+=1
print("Filtered Patients:")
for p in filtered: print(p)
print("\nTotal Patients per Department:")
for dept, count in grouped.items(): print(dept, count)
```

Output

```
Filtered Patients:
{'PatientID': 2, 'Name': 'Bob', 'Department': 'Neurology',
    'AdmissionDate': '2025-10-10'}
{'PatientID': 3, 'Name': 'Charlie', 'Department': 'Cardiology',
    'AdmissionDate': '2025-10-05'}

Total Patients per Department:
Neurology 1
Cardiology 1

=== Code Execution Successful ===
```

12. Create a CSV file named online store.csv with columns OrderID, Customer, Product, Amount. Load the data, filter orders with amount greater than 200, group by Customer, and calculate average, total, and maximum order amount per customer.

main.py

```
from collections import defaultdict
orders=[{"OrderID":1,"Customer":"A","Product":"Pen","Amount"
    :150},
        {"OrderID":2,"Customer":"B","Product":"Notebook"
            ,"Amount":250},
        {"OrderID":3,"Customer":"A","Product":"Marker"
            ,"Amount":300},
        {"OrderID":4,"Customer":"C","Product":"Book","Amount"
            :100}]
filtered=[o for o in orders if o["Amount"]>200]
grouped=defaultdict(list)
for o in filtered: grouped[o["Customer"]].append(o["Amount"])
stats={c:{"total":sum(v),"average":sum(v)/len(v),"max":max(v)}
    for c,v in grouped.items()}
print("Orders with Amount > 200:")
for o in filtered: print(o)
print("\nStats per Customer:")
for c, stat in stats.items(): print(c, stat)
```

Output

```
Orders with Amount > 200:
{'OrderID': 2, 'Customer': 'B', 'Product': 'Notebook', 'Amount':
    250}
{'OrderID': 3, 'Customer': 'A', 'Product': 'Marker', 'Amount': 300}

Stats per Customer:
B {'total': 250, 'average': 250.0, 'max': 250}
A {'total': 300, 'average': 300.0, 'max': 300}

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

13. Create a CSV file named flights.csv with columns FlightID, Airline, Source, Destination, Delay. Load the data, filter flights with delay above 30 minutes, group by Airline, and calculate total flights, average, and maximum delay per airline.

main.py | Share | Run | Output | Clear

```
from collections import defaultdict
flights=[{"FlightID":1,"Airline":"AirA","Source":"CityX","Destination":"CityY","Delay":45},
    {"FlightID":2,"Airline":"AirB","Source":"CityY","Destination":"CityZ","Delay":20},
    {"FlightID":3,"Airline":"AirA","Source":"CityX","Destination":"CityZ","Delay":35},
    {"FlightID":4,"Airline":"AirC","Source":"CityZ","Destination":"CityX","Delay":50}]
filtered=[f for f in flights if f["Delay"]>30]
grouped=defaultdict(list)
for f in filtered: grouped[f["Airline"]].append(f["Delay"])
stats={a:{"total_flights":len(v),"average":sum(v)/len(v),"max":max(v)} for a,v in grouped.items()}
print("Flights with Delay > 30:")
for f in filtered: print(f)
print("\nDelay Stats per Airline:")
for a, stat in stats.items(): print(a, stat)
```

```
Flights with Delay > 30:
{'FlightID': 1, 'Airline': 'AirA', 'Source': 'CityX', 'Destination': 'CityY', 'Delay': 45}
{'FlightID': 3, 'Airline': 'AirA', 'Source': 'CityX', 'Destination': 'CityZ', 'Delay': 35}
{'FlightID': 4, 'Airline': 'AirC', 'Source': 'CityZ', 'Destination': 'CityX', 'Delay': 50}

Delay Stats per Airline:
AirA {'total_flights': 2, 'average': 40.0, 'max': 45}
AirC {'total_flights': 1, 'average': 50.0, 'max': 50}

=== Code Execution Successful ===
```

14. Create a CSV file named inventory.csv with columns ProductID, ProductName, Supplier, Stock. Load the data, filter out-of-stock products, group by Supplier, and calculate total out-of-stock products per supplier.

main.py | Share | Run | Output | Clear

```
from collections import defaultdict
inventory=[{"ProductID":1,"ProductName":"Pen","Supplier":"S1","Stock":0},
    {"ProductID":2,"ProductName":"Notebook","Supplier":"S2","Stock":50},
    {"ProductID":3,"ProductName":"Eraser","Supplier":"S1","Stock":0},
    {"ProductID":4,"ProductName":"Marker","Supplier":"S3","Stock":30}]
filtered=[i for i in inventory if i["Stock"]==0]
grouped=defaultdict(int)
for i in filtered: grouped[i["Supplier"]]+=1
print("Out-of-Stock Products:")
for i in filtered: print(i)
print("\nOut-of-Stock Count per Supplier:")
for s, count in grouped.items(): print(s, count)
```

```
Out-of-Stock Products:
{'ProductID': 1, 'ProductName': 'Pen', 'Supplier': 'S1', 'Stock': 0}
{'ProductID': 3, 'ProductName': 'Eraser', 'Supplier': 'S1', 'Stock': 0}

Out-of-Stock Count per Supplier:
S1 2

=== Code Execution Successful ===
```

Name : Rajani
Roll No.: C-19
Software Proficiency Program II (Python Programming)

15. Create a CSV file named ecommerce.csv with columns OrderID, Category, SubCategory, Sales. Load the data, filter sales above 500, group by Category and SubCategory, and calculate total, average, and maximum sales per combination.

main.py | Share | Run | Output | Clear

```
from collections import defaultdict
sales=[{"OrderID":1,"Category":"Electronics","SubCategory":"Mobile","Sales":600},
       {"OrderID":2,"Category":"Electronics","SubCategory":"Laptop","Sales":1200},
       {"OrderID":3,"Category":"Furniture","SubCategory":"Chair","Sales":300},
       {"OrderID":4,"Category":"Electronics","SubCategory":"Mobile","Sales":700}]
filtered=[s for s in sales if s["Sales"]>500]
grouped=defaultdict(list)
for s in filtered: grouped[(s["Category"],s["SubCategory"])].append(s["Sales"])
stats={k:{"total":sum(v),"average":sum(v)/len(v),"max":max(v)} for k,v in grouped.items()}
print("Sales > 500:")
for s in filtered: print(s)
print("\nSales Stats per Category & SubCategory:")for k, stat in stats.items(): print(k, stat)
```

```
Sales > 500:
{'OrderID': 1, 'Category': 'Electronics', 'SubCategory': 'Mobile', 'Sales': 600}
{'OrderID': 2, 'Category': 'Electronics', 'SubCategory': 'Laptop', 'Sales': 1200}
{'OrderID': 4, 'Category': 'Electronics', 'SubCategory': 'Mobile', 'Sales': 700}

Sales Stats per Category & SubCategory:
('Electronics', 'Mobile') {'total': 1300, 'average': 650.0, 'max': 700}
('Electronics', 'Laptop') {'total': 1200, 'average': 1200.0, 'max': 1200}

=== Code Execution Successful ===
```

16. Create a CSV file named employee_performance.csv with columns EmpID, Name, Department, Rating, Bonus. Load the data, filter employees with Rating above 4.0, group by Department, and calculate average rating and total bonus per department.

main.py | Share | Run | Output

```
from collections import defaultdict
data=[{"ID":1,"Type":"A","Value":100},
      {"ID":2,"Type":"B","Value":200},
      {"ID":3,"Type":"A","Value":150},
      {"ID":4,"Type":"B","Value":50}]
filtered=[d for d in data if d["Value"]>100]
grouped=defaultdict(list)
for d in filtered: grouped[d["Type"]].append(d["Value"])
stats={t:{"total":sum(v),"average":sum(v)/len(v)} for t,v in grouped.items()}
print("Filtered Data:")
for d in filtered: print(d)
print("\nStats per Type:")
for t, stat in stats.items(): print(t, stat)
```

```
Filtered Data:
{'ID': 2, 'Type': 'B', 'Value': 200}
{'ID': 3, 'Type': 'A', 'Value': 150}

Stats per Type:
B {'total': 200, 'average': 200.0}
A {'total': 150, 'average': 150.0}

=== Code Execution Successful ===
```