

Exp.No: 6**Import A JSON File From The Command Line. Apply The Following Actions With The Data Present In The JSON File Where, Projection, Aggregation, Remove, Count, Limit, Skip And Sort**

To import a JSON file from the command line and apply the following actions with the data present in the JSON file where, projection, aggregation, remove, count, limit, skip and sort using jq tool.

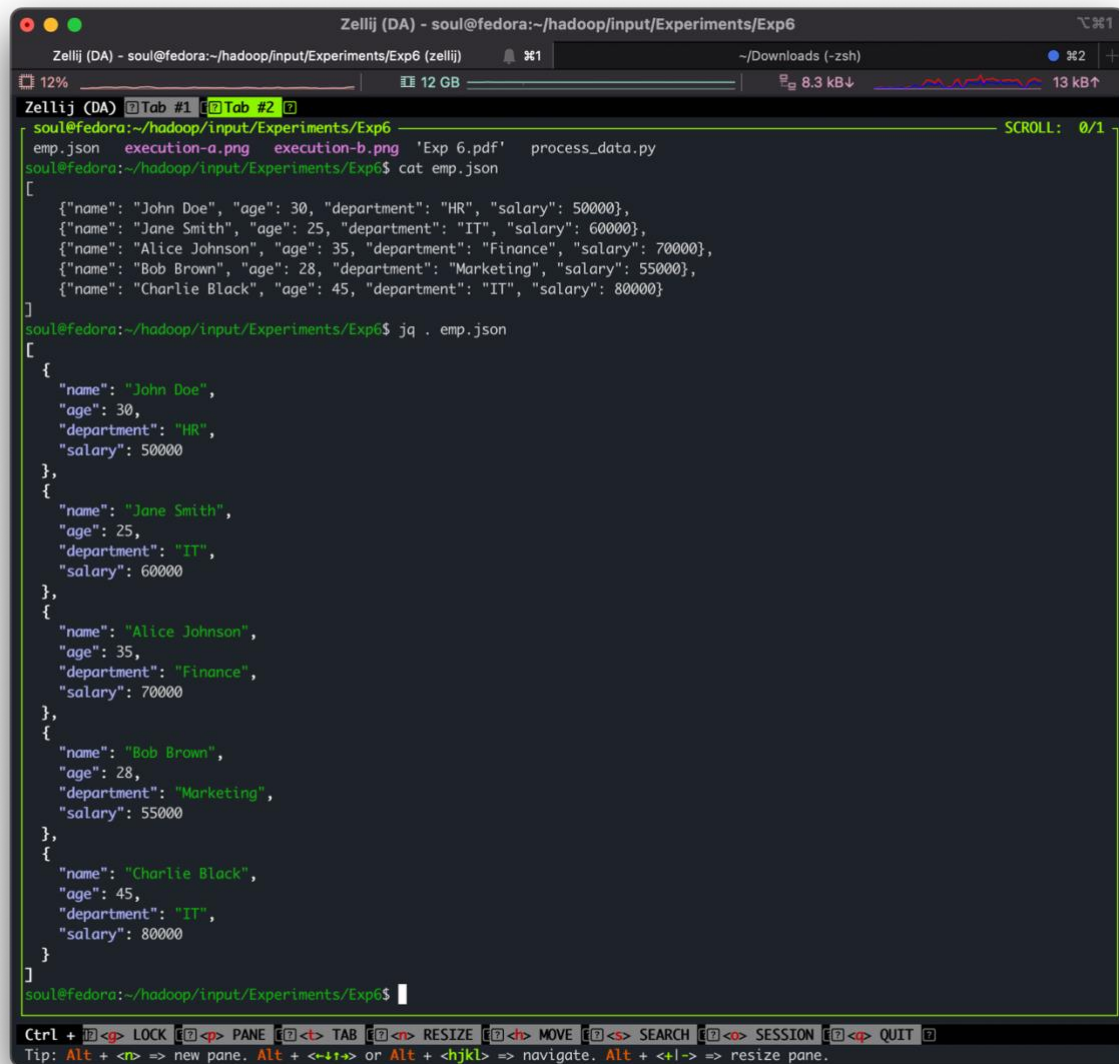
Procedure:

- Create a json file 'emp.json' and provide data in it.

```
[
  {
    "name" : "Anu",
    "age":12,
    "dept": "Computer",
    "salary":10000
  },
  {
    "name" : "Bob",
    "age" :14,
    "dept" : "HR",
    "salary":15000
  },
  {
    "name": "Jane Smith",
    "age": 25,
    "department": "IT",
    "salary": 60000
  },
  {
    "name": "Alice Johnson",
    "age": 35,
    "department": "Finance",
    "salary": 70000
  },
  {
    "name": "Bob Brown",
    "age": 28,
    "department": "Marketing",
    "salary": 55000
  }
]
```

Output:

Projecting the JSON data using jq :



The screenshot shows a terminal window titled "Zellij (DA) - soul@fedora:~/hadoop/input/Experiments/Exp6". The terminal displays the contents of a file named "emp.json" and the output of a jq command. The JSON data is an array of five objects, each representing an employee with fields for name, age, department, and salary. The jq command ". emp.json" is used to project the data, resulting in a formatted JSON output where each object is on a new line and properly indented.

```
Zellij (DA) - soul@fedora:~/hadoop/input/Experiments/Exp6
soul@fedora:~/hadoop/input/Experiments/Exp6$ cat emp.json
[
  {"name": "John Doe", "age": 30, "department": "HR", "salary": 50000},
  {"name": "Jane Smith", "age": 25, "department": "IT", "salary": 60000},
  {"name": "Alice Johnson", "age": 35, "department": "Finance", "salary": 70000},
  {"name": "Bob Brown", "age": 28, "department": "Marketing", "salary": 55000},
  {"name": "Charlie Black", "age": 45, "department": "IT", "salary": 80000}
]
soul@fedora:~/hadoop/input/Experiments/Exp6$ jq . emp.json
[
  {
    "name": "John Doe",
    "age": 30,
    "department": "HR",
    "salary": 50000
  },
  {
    "name": "Jane Smith",
    "age": 25,
    "department": "IT",
    "salary": 60000
  },
  {
    "name": "Alice Johnson",
    "age": 35,
    "department": "Finance",
    "salary": 70000
  },
  {
    "name": "Bob Brown",
    "age": 28,
    "department": "Marketing",
    "salary": 55000
  },
  {
    "name": "Charlie Black",
    "age": 45,
    "department": "IT",
    "salary": 80000
  }
]
soul@fedora:~/hadoop/input/Experiments/Exp6$
```

Ctrl + [?] LOCK [?] PANE [?] TAB [?] RESIZE [?] MOVE [?] SEARCH [?] SESSION [?] QUIT [?]
Tip: Alt + <+> => new pane. Alt + <+> or Alt + <h> => navigate. Alt + <+> => resize pane.

Python Script to process data :

```

Zellij (DA) - soul@fedora:~/hadoop/input/Experiments/Exp6
Zellij (DA) - soul@fedora:~/hadoop/input/Experiments/Exp6 (zellij) 12 GB 1.0 kB↓ 2.0 kB↑
Zellij (DA) Tab #1 Tab #2
soul@fedora:~/hadoop/input/Experiments/Exp6
GNU nano 7.2 process_data.py
from hdfs import InsecureClient
import pandas as pd
import json

# Connect to HDFS
hdfs_client = InsecureClient('http://localhost:9870', user='root')

# Read JSON data from HDFS
try:
    with hdfs_client.read('/emp.json', encoding='utf-8') as reader:
        json_data = reader.read() # Read the raw data as a string
        if not json_data.strip(): # Check if data is empty
            raise ValueError("The JSON file is empty.")
        print(f"Raw JSON Data: {json_data[:1000]}") # Print first 1000 characters for debugging
        data = json.loads(json_data) # Load the JSON data
except json.JSONDecodeError as e:
    print(f"JSON Decode Error: {e}")
    exit(1)
except Exception as e:
    print(f"Error reading or parsing JSON data: {e}")
    exit(1)

# Convert JSON data to DataFrame
try:
    df = pd.DataFrame(data)
except ValueError as e:
    print(f"Error converting JSON data to DataFrame: {e}")
    exit(1)

# Projection: Select only 'name' and 'salary' columns
projected_df = df[['name', 'salary']]

# Aggregation: Calculate total salary
total_salary = df['salary'].sum()

# Count: Number of employees earning more than 50000
high_earners_count = df[df['salary'] > 50000].shape[0]

# Limit: Get the top 5 highest earners
top_5_earners = df.nlargest(5, 'salary')

# Skip: Skip the first 2 employees
skipped_df = df.iloc[2:]

# Remove: Remove employees from a specific department
filtered_df = df[df['department'] != 'IT']

# Save the filtered result back to HDFS

```

Read 77 lines

Help Write Out Where Is Cut Execute Location M-U Undo M-A Set Mark
Exit Read File Replace Paste J Justify Go To Line M-E Redo M-G Copy

Ctrl + [L] LOCK [P] PANE [T] TAB [R] RESIZE [M] MOVE [S] SEARCH [SE] SESSION [Q] QUIT [H]
Tip: Alt + [C] => new pane. Alt + [I] or Alt + [HJKL] => navigate. Alt + [I] => resize pane.

Script Execution :

```
soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6
soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6$ hdfs dfs -cat /exp6/*
[{"name":"John Doe","age":30,"department":"HR","salary":50000}, {"name":"Alice Johnson","age":35,"department":"Finance","salary":70000}, {"name":"Bob Brown","age":28,"department":"Marketing","salary":55000}]soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6$ hdfs dfs -ls /exp6
Found 1 items
-rw-r--r-- 1 root supergroup 205 2024-09-16 20:48 /exp6/filtered_employees.json
soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6$ python process_data.py
Raw JSON Data: [
  {"name": "John Doe", "age": 30, "department": "HR", "salary": 50000},
  {"name": "Jane Smith", "age": 25, "department": "IT", "salary": 60000},
  {"name": "Alice Johnson", "age": 35, "department": "Finance", "salary": 70000},
  {"name": "Bob Brown", "age": 28, "department": "Marketing", "salary": 55000},
  {"name": "Charlie Black", "age": 45, "department": "IT", "salary": 80000}
]

Filtered JSON file saved successfully.
Projection: Select only name and salary columns
  name  salary
0  John Doe  50000
1  Jane Smith 60000
2  Alice Johnson 70000
3    Bob Brown 55000
4  Charlie Black 80000
Aggregation: Calculate total salary
Total Salary: 315000

# Count: Number of employees earning more than 50000
Number of High Earners (>50000): 4
```

```
soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6
Top 5 Earners:
  name  age  department  salary
4  Charlie Black  45      IT  80000
2  Alice Johnson  35    Finance  70000
1    Jane Smith  25      IT  60000
3    Bob Brown  28    Marketing  55000
0    John Doe   30      HR   50000

Skipped DataFrame (First 2 rows skipped):
  name  age  department  salary
2  Alice Johnson  35    Finance  70000
3    Bob Brown  28    Marketing  55000
4  Charlie Black  45      IT  80000

Filtered DataFrame (Sales department removed):
  name  age  department  salary
0  John Doe   30      HR   50000
2  Alice Johnson  35    Finance  70000
3    Bob Brown  28    Marketing  55000
soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6$ cat emp.json
[
  {"name": "John Doe", "age": 30, "department": "HR", "salary": 50000},
  {"name": "Jane Smith", "age": 25, "department": "IT", "salary": 60000},
  {"name": "Alice Johnson", "age": 35, "department": "Finance", "salary": 70000},
  {"name": "Bob Brown", "age": 28, "department": "Marketing", "salary": 55000},
  {"name": "Charlie Black", "age": 45, "department": "IT", "salary": 80000}
]
soul@fedora:~/hadoop-3.4.0/input/Experiments/Exp6$
```

RESULT:

Thus to import a JSON file from the command line and apply the following actions with the data present in the JSON file where, projection, aggregation, remove, count, limit, skip and sort using jq tool is completed successfully