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

AIM:

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,
```

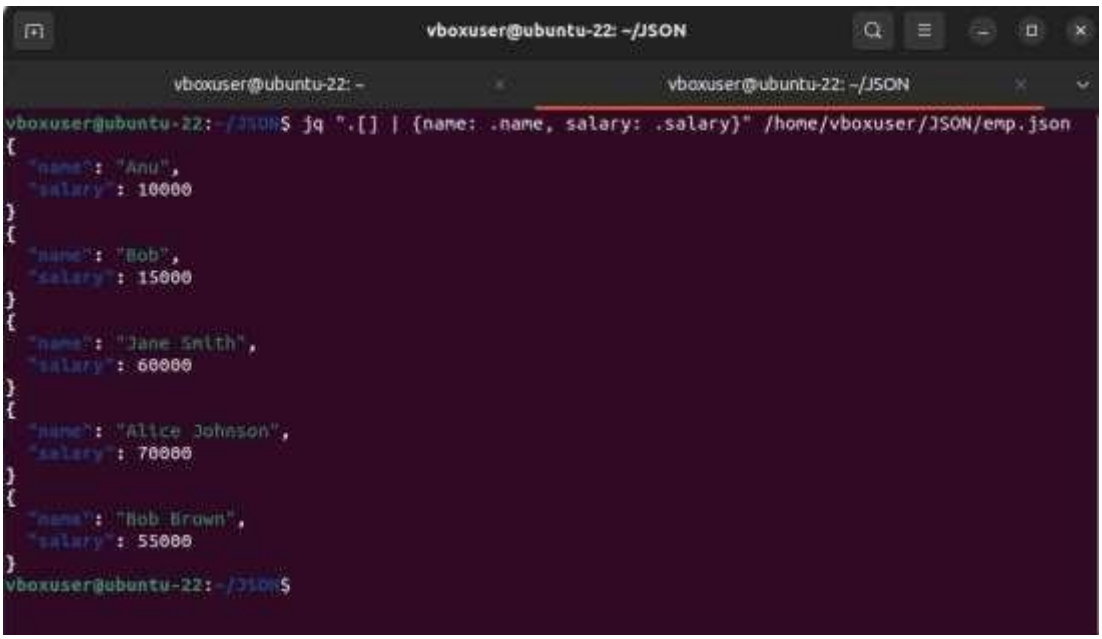
```
"department": "Finance",
"salary": 70000
},
{
"name": "Bob Brown",
"age": 28,
"department": "Marketing",
"salary": 55000
}
]
```


- Open the command prompt.
- Navigate to the folder where emp.json is stored.
- Load and view the JSON data with jq.
- Use the jq commands for projection, aggregation, removal, counting, limiting, and sorting operations.

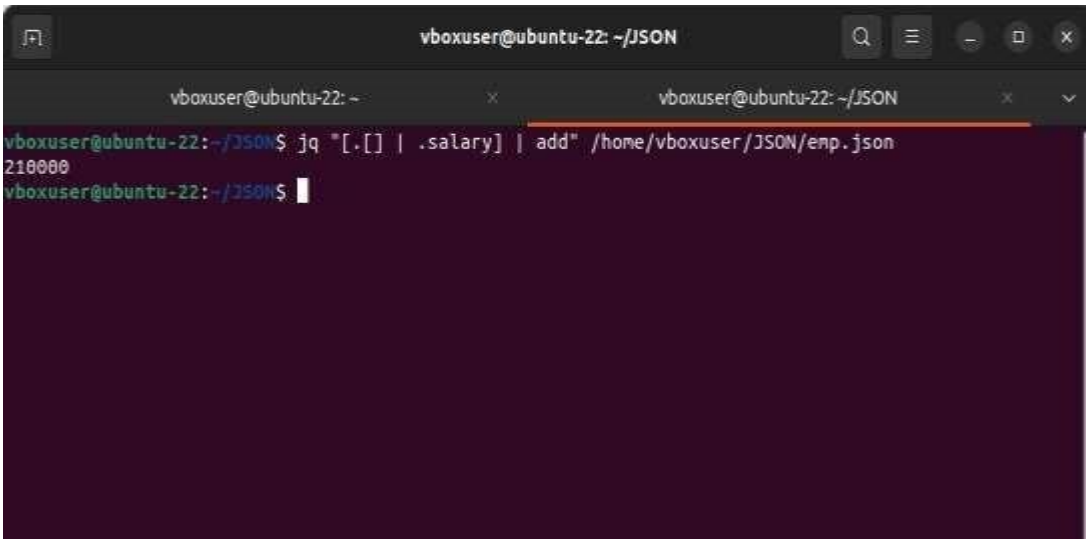
OUTPUT:

Running jq queries:

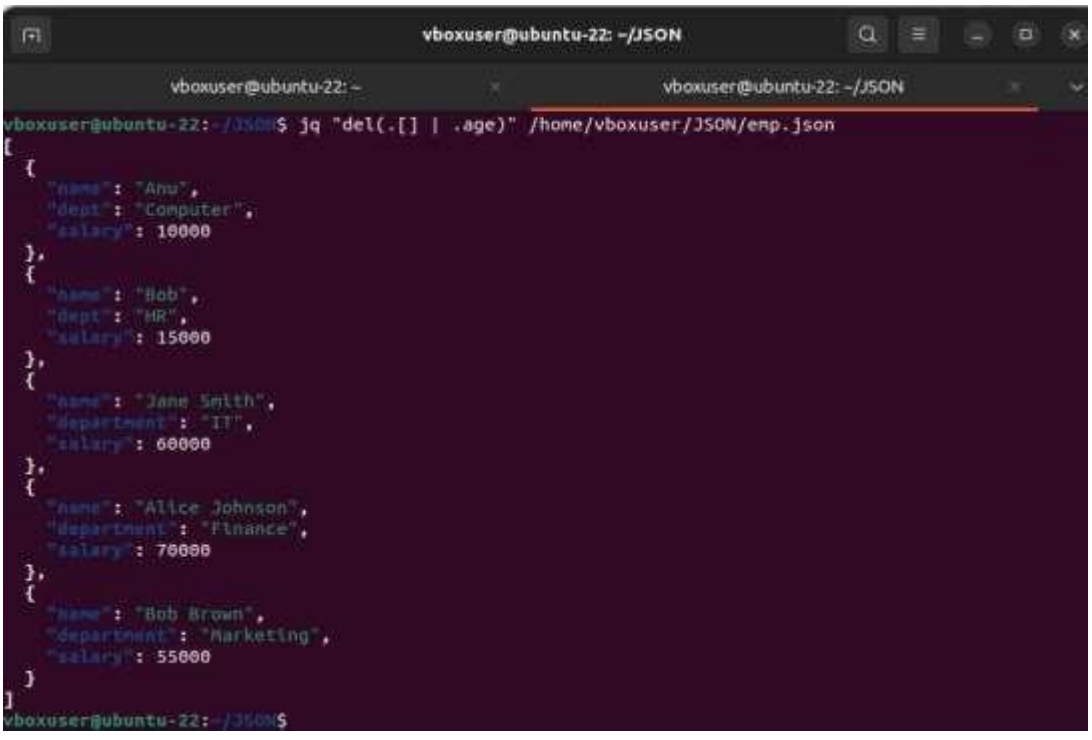
1. ***Projection:***



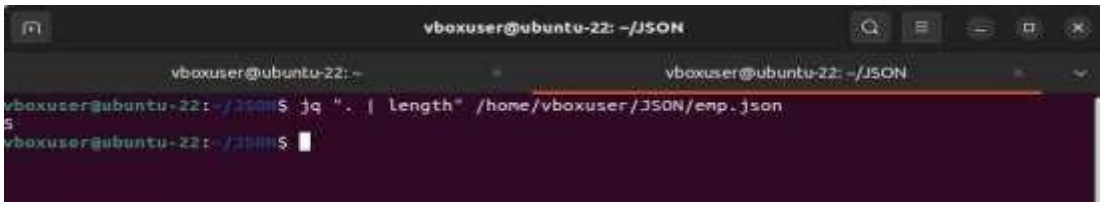
2. *Aggregation:*



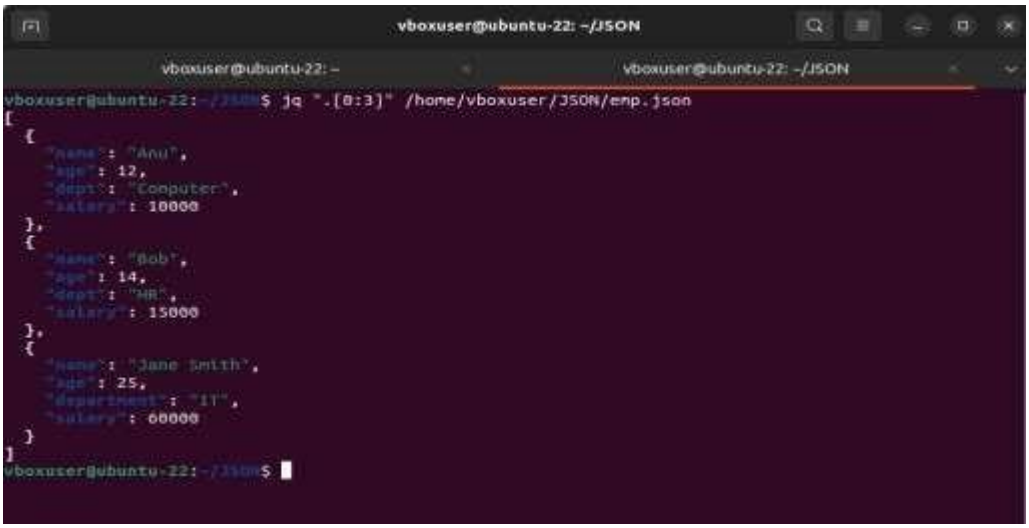
3. *Remove:*



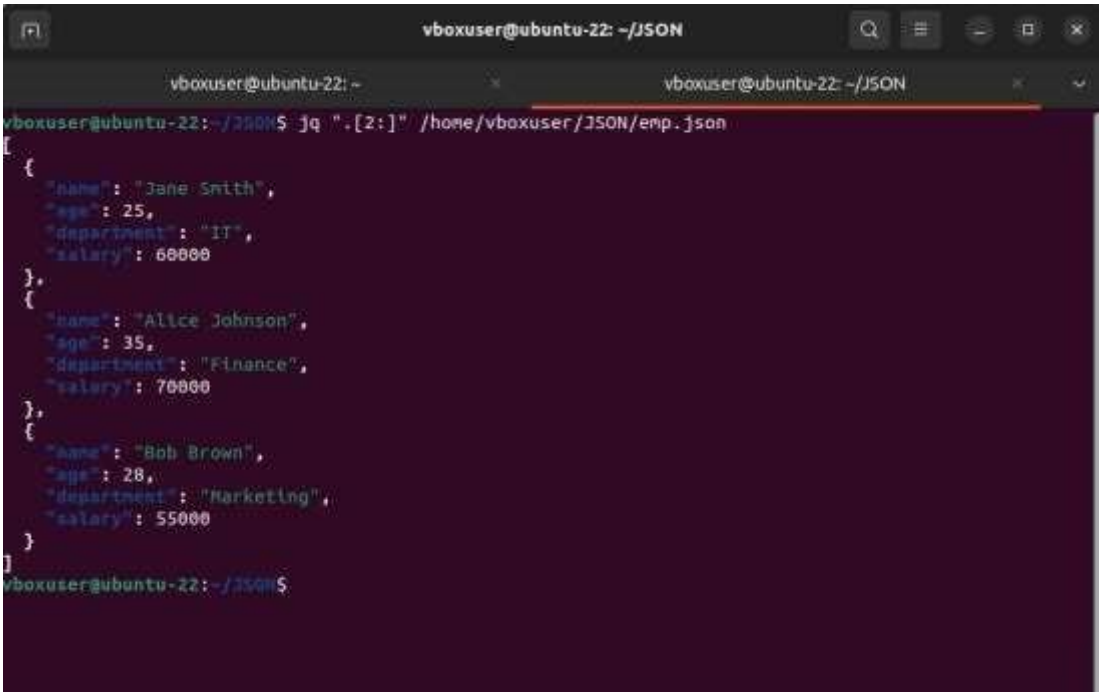
4. *Count:*



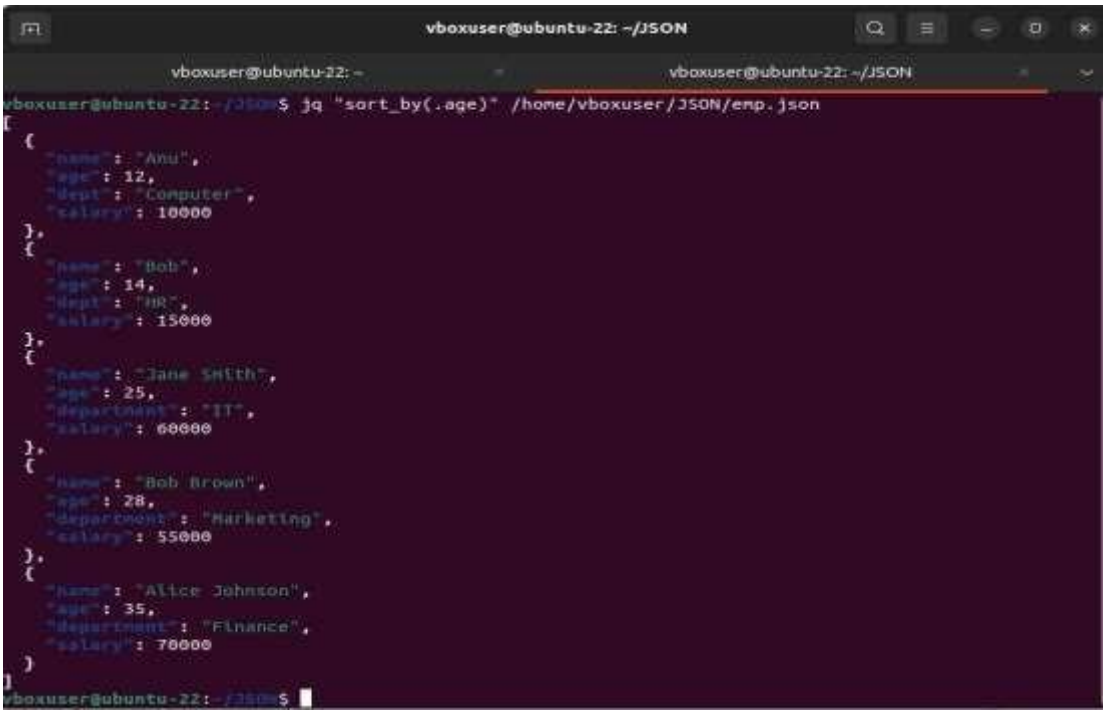
5. *Limit:*



6. *Skip:*



7. *Sort:*



8. Analyzing json data with python

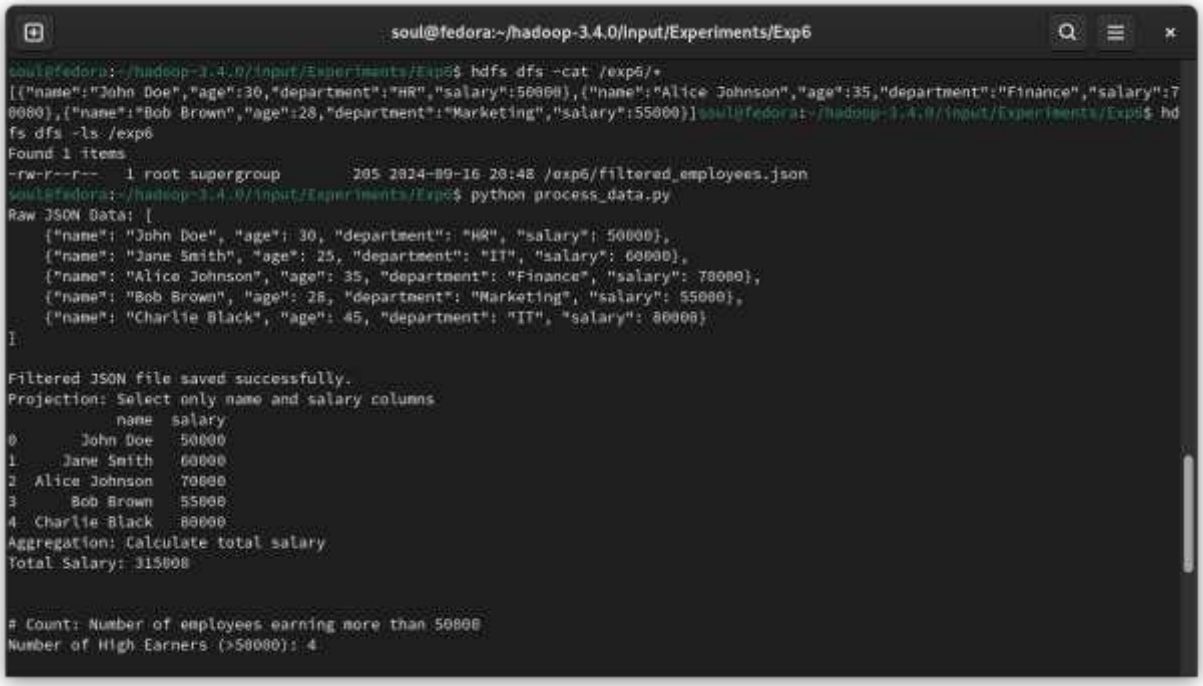
1. Load emp.json into hdfs root folder
2. Run the below python code with root privileges

```
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)
```

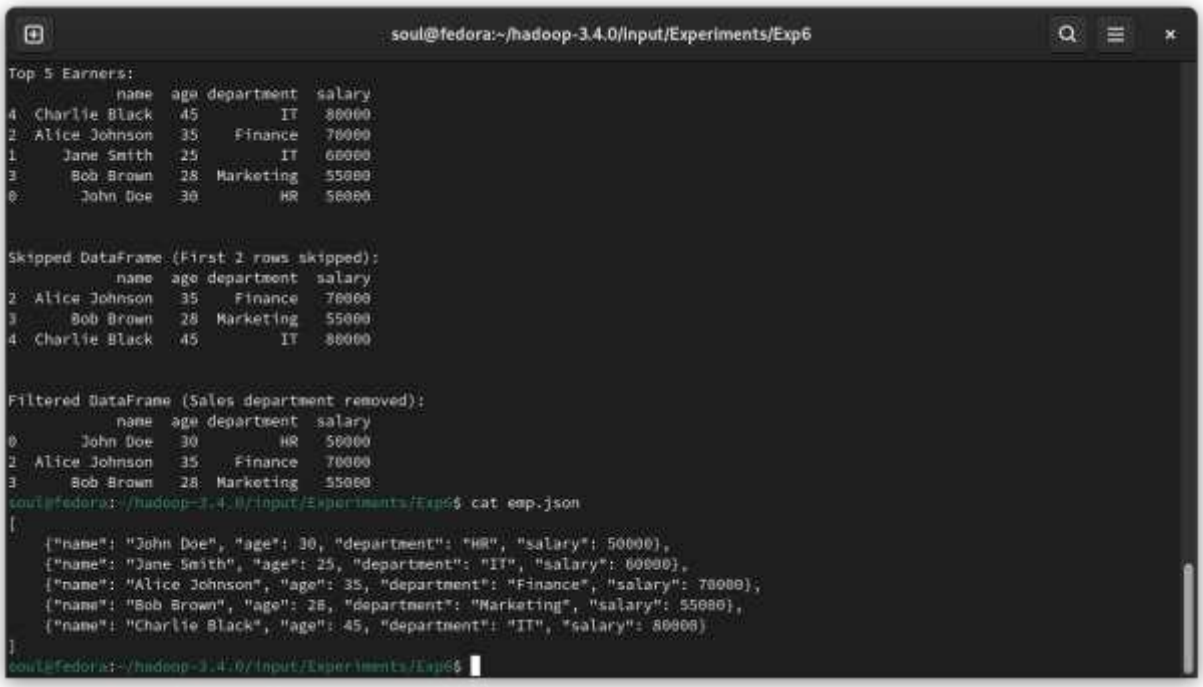
```
# Convert JSON data to DataFrame try:
   df = pd.DataFrame(data) except ValueError as e:
print(f"Error converting JSON data to DataFrame: {e}")
exit(1)

# 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 filtered_json
= filtered_df.to_json(orient='records') try:
   with hdfs_client.write('/exp6/filtered_employees.json', encoding='utf-8',
overwrite=True) as writer:        writer.write(filtered_json)    print("Filtered
JSON file saved successfully.") except Exception as e:    print(f"Error saving
filtered JSON data: {e}")    exit(1)

# Print results
print(f"Projection: Select only name and salary columns")
print(f"{projected_df}") print(f"Aggregation: Calculate
total salary") print(f"Total Salary: {total_salary}")
print(f"\n")



print(f"# Count: Number of employees earning more than 50000")

print(f"Number of High Earners (>50000): {high_earners_count}")
print(f"\n")
```

```
print(f"limit Top 5 highest salary")

print(f"Top 5 Earners: \n{top_5_earners}") print(f"\n")
print(f"Skipped DataFrame (First 2 rows skipped): \n{skipped_df}")
print(f"\n")
print(f"Filtered DataFrame (Sales department removed): \n{filtered_df}")
```

Execution - a



Execution – b



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