

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

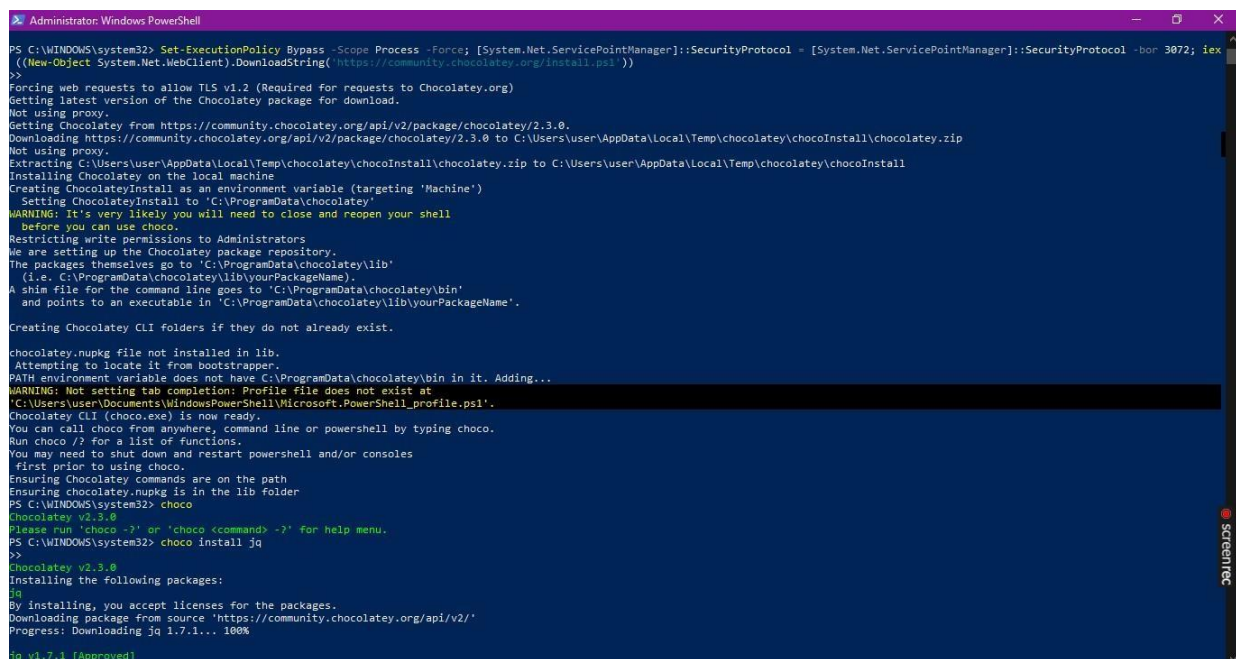
**employees.json:**

```
[
  {
    "id": 1,
    "name": "Alice Johnson",
    "department": "Engineering",
    "age": 29,
    "salary": 70000
  },
  {
    "id": 2,
    "name": "Bob Smith",
    "department": "Marketing",
    "age": 35,
    "salary": 55000
  },
  {
    "id": 3,
    "name": "Charlie Davis",
    "department": "Engineering",
```

```
"age": 25,
"salary": 60000
},
{
  "id": 4,
  "name": "Dana Lee",
  "department": "Human Resources",
  "age": 40,
  "salary": 65000
},
{
  "id": 5,
  "name": "Eve Martinez",
  "department": "Finance",
  "age": 45,
  "salary": 75000
}
]
```

## OUTPUT:

### Installation of jq packages:



```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex
((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
>>
Forcing web requests to allow TLS v1.2 (Required for requests to Chocolatey.org)
Getting latest version of the Chocolatey package for download.
Not using proxy.
Getting Chocolatey from https://community.chocolatey.org/api/v2/package/chocolatey/2.3.0.
Downloading https://community.chocolatey.org/api/v2/package/chocolatey/2.3.0 to C:\Users\user\AppData\Local\Temp\chocolatey\chocoInstall\chocolatey.zip
Not using proxy.
Extracting C:\Users\user\AppData\Local\Temp\chocolatey\chocoInstall\chocolatey.zip to C:\Users\user\AppData\Local\Temp\chocolatey\chocoInstall
Installing Chocolatey on the local machine
Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'
WARNING: It's very likely you will need to close and reopen your shell
... before you can use choco.
Restricting write permissions to Administrators
We are setting up the Chocolatey package repository.
The packages themselves go to 'C:\ProgramData\chocolatey\lib'
(i.e. C:\ProgramData\chocolatey\lib\yourPackageName).
A shim file for the command line goes to 'C:\ProgramData\chocolatey\bin'
and points to an executable in 'C:\ProgramData\chocolatey\lib\yourPackageName'.

Creating Chocolatey CLI folders if they do not already exist.

chocolatey.nupkg file not installed in lib.
Attempting to locate it from bootstrapper.
PATH environment variable does not have C:\ProgramData\chocolatey\bin in it. Adding...
WARNING: Not setting tab completion: Profile file does not exist at
'C:\Users\user\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey CLI (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
... first prior to using choco.
Ensuring Chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder
PS C:\WINDOWS\system32> choco
chocolatey v2.3.0
Please run 'choco -?' or 'choco <command> -?' for help menu.
PS C:\WINDOWS\system32> choco install jq
>>
Chocolatey v2.3.0
Installing the following packages:
jq
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading jq 1.7.1... 100%
jq v1.7.1 [Approved]
```

## Running jq queries:

### I. Projection:

```
jq ".[] | {name: .name, salary: .salary}" Desktop/employees.json
```

```
C:\Users\user>jq ".[] | {name: .name, salary: .salary}" Desktop/employees.json
{
  "name": "Alice Johnson",
  "salary": 70000
},
{
  "name": "Bob Smith",
  "salary": 55000
},
{
  "name": "Charlie Davis",
  "salary": 60000
},
{
  "name": "Dana Lee",
  "salary": 65000
},
{
  "name": "Eve Martinez",
  "salary": 75000
}
```

### II. Aggregation:

```
jq "[.[] | .salary] | add" Desktop/employees.json
```

```
C:\Users\user>jq "[.[] | .salary] | add" Desktop/employees.json
325000
```

### III. Remove:

```
jq "del(.[] | .age)" Desktop/employees.json
```

```
C:\Users\user>jq "del(.[] | .age)" Desktop/employees.json
[
  {
    "id": 1,
    "name": "Alice Johnson",
    "department": "Engineering",
    "salary": 70000
  },
  {
    "id": 2,
    "name": "Bob Smith",
    "department": "Marketing",
    "salary": 55000
  },
  {
    "id": 3,
    "name": "Charlie Davis",
    "department": "Engineering",
    "salary": 60000
  },
  {
    "id": 4,
    "name": "Dana Lee",
    "department": "Human Resources",
    "salary": 65000
  },
  {
    "id": 5,
    "name": "Eve Martinez",
    "department": "Finance",
    "salary": 75000
  }
]
```

#### IV. Count:

```
jq ". | length" Desktop/employees.json
```

```
C:\Users\user>jq ". | length" Desktop/employees.json  
5
```

#### V. Limit:

```
jq ".[0:3]" Desktop/employees.json
```

```
C:\Users\user>jq ".[0:3]" Desktop/employees.json  
[  
  {  
    "id": 1,  
    "name": "Alice Johnson",  
    "department": "Engineering",  
    "age": 29,  
    "salary": 70000  
  },  
  {  
    "id": 2,  
    "name": "Bob Smith",  
    "department": "Marketing",  
    "age": 35,  
    "salary": 55000  
  },  
  {  
    "id": 3,  
    "name": "Charlie Davis",  
    "department": "Engineering",  
    "age": 25,  
    "salary": 60000  
  }  
]
```

#### VI. Skip:

```
jq ".[2:]" Desktop/employees.json
```

```
C:\Users\user>jq ".[2:]" Desktop/employees.json  
[  
  {  
    "id": 3,  
    "name": "Charlie Davis",  
    "department": "Engineering",  
    "age": 25,  
    "salary": 60000  
  },  
  {  
    "id": 4,  
    "name": "Dana Lee",  
    "department": "Human Resources",  
    "age": 40,  
    "salary": 65000  
  },  
  {  
    "id": 5,  
    "name": "Eve Martinez",  
    "department": "Finance",  
    "age": 45,  
    "salary": 75000  
  }  
]
```

## VII. Sort:

```
jq "sort_by(.age)" Desktop/employees.json
```

```
C:\Users\user>jq "sort_by(.age)" Desktop/employees.json
[
  {
    "id": 3,
    "name": "Charlie Davis",
    "department": "Engineering",
    "age": 25,
    "salary": 60000
  },
  {
    "id": 1,
    "name": "Alice Johnson",
    "department": "Engineering",
    "age": 29,
    "salary": 70000
  },
  {
    "id": 2,
    "name": "Bob Smith",
    "department": "Marketing",
    "age": 35,
    "salary": 55000
  },
  {
    "id": 4,
    "name": "Dana Lee",
    "department": "Human Resources",
    "age": 40,
    "salary": 65000
  },
  {
    "id": 5,
    "name": "Eve Martinez",
    "department": "Finance",
    "age": 45,
    "salary": 75000
  }
]
```

## 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.