Ansible

What is Ansible?

Ans: Ansible is an open-source software provisioning, configuration management and application deployment tool enabling infrastructure as code. Simply ansible is an configuration management tool. The only dependency of ansible is Python.

How to install Ansible on Ubuntu:

```
sudo –i

sudo apt-add-repository ppa:ansible/ansible

sudo apt-get update

sudo apt-get install ansible

ansible –version
```

```
# output:

# ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/root/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.5 (default, Jul 28 2020, 12:59:40) [GCC 9.3.0]
```

```
cd /etc/ansible/

ls

vi hosts

i

# enter private IP addresses of remote servers with group names at the end of the hosts file like:

[web]
172.31.6.128

[db]
172.31.1.46
```

Press esc

```
:wq! #to save the file.

Ssh-keygen #generate ssh key to connect with remote servers.
```

```
cd

ls –a

cd .ssh/

ls

# Output:

  authorized_keys  id_rsa  id_rsa.pub  known_hosts


cat id_rsa.pub

# copy the pub key

# now open remote server and go to root

Sudo –i

cd .ssh

ls

# output:

  authorized_keys

vi authorized_keys

# paste the copied id_rsa.pub from ansible server.

Press esc

:wq!

Systemctl restart sshd

# move to ansible server and type the following command to check whether the remote servers connected properly or not.

Ansible –m ping web/db

# here web/db are the server groups created in inventory(hosts) file.

# will get a ping pong message which is a indication of task succeeded.

# ensure that ssh port 22 is opened in all the servers security group. Add private IP of ansible server and also open port 80 in all the remote servers.
```

Ansible playbook to install nginx webserver:

cd /etc/ansible/

vi nginx.yaml

# here nginx is the playbook name which we will write by using yaml code. File must end with .yml or .yaml

# Write playbook now

```
---
- hosts: web
  tasks:
    - name: ensure nginx is at the latest version
      apt: name=nginx state=latest
    - name: start nginx
      service:
          name: nginx
          state: start
```

press esc

:wq!

Ansible-playbook nginx.yaml

# nginx.yaml is the yaml file name.

Ansible playbook to stop nginx webserver:

```
---
- hosts: web
  tasks:
    - name: stop nginx
      service:
          name: nginx
          state: stopped
    - name: ensure nginx is not installed
      apt: name=nginx state=absent
```

Ansible playbook to fetch a file from remote to local:

```
---
- hosts: web
  tasks:
   - name: "fetch jofetch txt file from remote to local"
     fetch:
      src: /var/www/jofetch.txt
      dest: /etc/ansible
```

```
# specify the hosts, src and dest as per the requirement
```

Ansible playbook to copy a file local to remote server:

```
---
- hosts: web
  tasks:
   - name: "copy index file from local to remote"
     copy:
      src: /opt/index.html
      dest: /opt/

# specify the hosts, src and dest as per the requirement.
```

Ansible playbook to install Mysql without roles:

```
---
 - name: install my sql database and create a user
   hosts: db
   become: yes
   tasks:
    - name: install mysql and dependent packages
      package:
       name: "{{item}}"
       state: present
       update_cache: yes
      loop:
        - mysql-server
        - python3-mysqldb
    - name: start mysql
      service:
         name: mysql
         state: started
    - name: create a demo database
      mysql_db:
       name: mysql
       state: present
    - name: create demo user with all precilages and grant
      mysql_user:
       name: jyothsna
       password: jo123
       priv: '*.*:ALL,GRANT'
       state: present
```

Ansible Roles:

# To overcome the disadvantages with playbooks we will use ansible roles. With the help of ansible roles we can reuse the playbooks which is difficult with playbooks.
# to create ansible roles use the following command.

Ansible-galaxy init devops

# devops is the role name.

cd devops/
ls

# output:
README.md defaults files handlers meta tasks templates tests vars (directories under the role created by default).

Mysql_server :

```
---
- hosts: db
  become: true
  roles:
     - dev
```

cd /dev/tasks/main.yml :

```
---
# tasks file for dev
- name: install mysql-server
  apt: name=mysql-server state=present update_cache=yes
```

MY_SQL SERVER INSTALLATION USING ANSIBLE ROLES:

```
name: install my sql database and create a user
hosts: web
become: yes
tasks:
 - name: install mysql and dependent packages
   Package:
    name: "{{item}}"
    state: present
    update_cache: yes
   loop:
     - mysql-server
     - mysql-client
     - python-mysqldb

- name: start mysql
   services:
      name: mysql
      state: started

 - name: create mysql user and password
   mysql_user:
    name: jomysql  "{{dbuser}}"
    password: jo1234 "{{dbpass}}"
    priv: '*.*:ALL,GRANT'
    state: present
```

PATH:- $ansible# cd dev/

Tasks vars handlers tests

tasks:

(1)Main.yml

```
- include: install.yml
- include: start.yml
- include: cred.yml
- include: dbname.yml
```

(2)install.yml

```
 - name: install mysql and dependent packages
   package:
    name: "{{item}}"
    state: present
    update_cache: yes
   loop:
     - mysql-server
     - mysql-client
     - python3-mysqldb
```

(2)start.yml

```
- name: start mysql
   services:
      name: mysql
      state: started
```

(4)cred.yml

```
- name: create a username and password
   mysql_user:
    name: abhi
    pasword: abhi1234
    priv: "*.*:ALL,GRANT"
    State: present
```

(5)dbname.yml

```
- name: create a demo database
   mysql_db:
    name: abhidb
    state: present
```

Vars:

(1)main.yml

```
dbuser: abhi
dbpass: jo1234
dbname: jomysql
```