

## Homework\_Lesson20\_Ansible\_3

Цель: получить практический опыт работы с Ansible-ролями

Задание 1: просмотрите и повторите все манипуляции, указанные в видеоуроке по ссылке <https://www.youtube.com/watch?v=9pHMZnb3JDQ>

В данном ролике рассматриваются настройки Ansible-ролей.

Задание 2: (опционально, на выбор):

- Роль для настройки базы данных. Напишите роль, которая устанавливает и настраивает базу данных (например, MySQL или PostgreSQL). Роль должна принимать переменные для конфигурирования базы данных, такие как пароль для root-пользователя и порт, на котором база данных будет слушать.

### Выполнение 1-ого задания:

Создание ролей в Ansible:

```
$ mkdir roles                                \\ создаем директорию roles
$ ansible-galaxy init deploy_apache_web      \\ создание роли с названием deploy_apache_web
$ tree                                       \\ показывает структуру директории roles
```



```
makarov@DESKTOP-UG6J7T7:~/ansible/roles$ tree
.
├── deploy_apache_web
│   ├── defaults
│   │   └── main.yml
│   ├── files
│   ├── handlers
│   │   └── main.yml
│   ├── meta
│   │   └── main.yml
│   ├── README.md
│   ├── tasks
│   │   └── main.yml
│   ├── templates
│   ├── tests
│   │   ├── inventory
│   │   └── test.yml
│   └── vars
│       └── main.yml
└── 10 directories, 8 files
```

```
$ nano deploy_apache_web/defaults/main.yml
```

Далее, мы наш playbook для настройки/установки чего либо превращаем в роль, т.е. содержимое playbook переносится в роль, тип tasks playbook переносится в tasks -> main.yml нашей созданной роли.

```
$ ansible-playbook playbook6.yml
```

### Выполнение 2-ого задания:

Создаем роль для настройки базы данных PostgreSQL.

На WSL выполняем следующие действия:

```
$ ssh-keygen -t rsa -b 4096                \\ создание ключа
$ ssh-copy-id user@192.168.100.14          \\ копируем наш ключ на ВМ
```

Создаем отдельную директорию ansible\_postgres.

```
$ mkdir ansible_postgres  
$ cd ./ansible_postgres
```

Создаем файлы hosts.ini и конфигурационный файл ansible.cfg.

```
$ touch hosts.ini  
$ touch ansible.cfg  
$ nano hosts.ini
```

```
GNU nano 7.2                                hosts.ini  
[servers]  
user-VirtualBox      ansible_host=192.168.100.14  ansible_user=user
```

В файле hosts.ini прописываем наши сервера на котором будем устанавливать и настраивать Postgresql

```
$ nano ansible.cfg
```

```
GNU nano 7.2                                ansible.cfg  
[defaults]  
inventory             = ./hosts.ini
```

Указываем файл hosts.ini

```
$ touch vars.yml      \\ создаем файл переменных  
$ nano vars.yml       \\ добавляем пользователя БД, имя БД и пароль
```

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ cat vars.yml  
db_user: admin  
db_password: qwerty_123  
db_name: tms
```

Данные переменные, нам понадобятся для создания пользователя и БД в PostgreSQL

Создаем playbook для установки/настройки Postgres. Сначала впишем несколько строк для установки PostgreSQL и попробуем проверить:

```
$ nano install_pg.yml
```

Добавляем следующие строки:

```
---  
- name: Install PostgreSQL and create DB, db_user  
  hosts: servers  
  become: yes  
  vars_files:  
  - vars.yml  
  
pre_tasks:  
  - name: Install PostgreSQL  
    ansible.builtin.apt:
```

```
name: postgresql
state: latest
update_cache: true
```

Пробуем запустить playbook:

```
$ ansible-playbook install_pg.yml
```

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml
PLAY [Install PostgreSQL and create DB, db_user] *****
TASK [Gathering Facts] *****
ok: [user-VirtualBox]
TASK [Install PostgreSQL] *****
changed: [user-VirtualBox]
PLAY RECAP *****
user-VirtualBox : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ |
```

Установка PostgreSQL успешна

Заходим удаленно по SSH на нашу VM:

```
$ ssh user@192.168.100.14
```

```
$ sudo apt remove postgresql*           \\ удаляем PostgreSQL
```

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ssh user@192.168.100.14
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-51-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

50 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun Jan 26 15:25:08 2025 from 192.168.100.13
user@user-VirtualBox:~$ sudo apt remove postgresql*
```

```
$ exit           \\ возвращаемся к нашей WSL (Ansible-Master)
```

Добавляем в playbook поднятие сервиса postgresql и добавляем в автозагрузку:

tasks:

- name: Start and enable services

service:

```
name: postgresql
```

```
state: started           \\ Запускаем сервис
```

```
enabled: yes             \\ добавляем в автозагрузку (enabled)
```

Переходим к наполнению нашего playbook (install\_pg.yml). Нам нужно, чтоб после установки СУБД PostgreSQL, у нас залогинилось пользователем postgres (который по умолчанию появляется, после установки PostgreSQL) и указать: создание базы данных tms и юзера admin. Также можно будет добавить строки на загрузку каких-нибудь данных в созданную БД tms.

Добавляем создание БД и пользователя, которых мы указали в vals.yml, в файл install\_pg.yml:

```
- name: Create database
  postgresql_db:
    state: present
    name: "{{ db_name }}"
  become: yes
  become_user: postgres

- name: Create db user
  postgresql_user:
    state: present
    name: "{{ db_user }}"
    password: "{{ db_password }}"
  become: yes
  become_user: postgres
```

Пробуем для тестирования запустить сборку playbook (install\_pg.yml):

```
$ ansible-playbook install_pg.yml
```

Выдает ошибку, что нехватает библиотек.

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml
PLAY [Install PostgreSQL and create DB, db_user] *****

TASK [Gathering Facts] *****
ok: [user-VirtualBox]

TASK [Install PostgreSQL] *****
changed: [user-VirtualBox]

TASK [Start and enable services] *****
ok: [user-VirtualBox]

TASK [Create database] *****
fatal: [user-VirtualBox]: FAILED! => {"changed": false, "msg": "Failed to import the required Python library (psycopg2) on user-VirtualBox's Python /usr/bin/python3. Please read the module documentation and install it in the appropriate location. If the required library is installed, but Ansible is using the wrong Python interpreter, please consult the documentation on ansible_python_interpreter"}

PLAY RECAP *****
user-VirtualBox      : ok=3    changed=1    unreachable=0    failed=1    skipped=0    rescued=0    ignored=0
```

Ошибки из-за нехватки библиотеки Python psycopg2 на стороне ВМ

Покопавшись в инете, необходимо в наш playbook добавить задачу на установку пакетов python3-psycopg2. Данная библиотека позволяет использовать модули PostgreSQL, т.е. postgresql\_db, postgresql\_user, postgresql\_pg\_hba и postgresql\_privs. Добавим его в pre\_tasks где указана установка пакетов PostgreSQL:

```
- name: Install Python packages
  apt:
    name: python3-psycopg2
    state: present
```

```

makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ cat install_pg.yml
---
- name: Install PostgreSQL and create DB, db_user
  hosts: servers
  become: yes
  vars_files:
  - vars.yml

  pre_tasks:
    - name: Install PostgreSQL
      ansible.builtin.apt:
        name: postgresql
        state: latest
        update_cache: true

    - name: Install Python packages
      apt:
        name: python3-psycopg2
        state: present

```

Поправленный вариант install\_pg.yml

Пробуем повторный запуск:

\$ ansible-playbook install\_pg.yml

```

makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml
PLAY [Install PostgreSQL and create DB, db_user] *****
TASK [Gathering Facts] *****
ok: [user-VirtualBox]
TASK [Install PostgreSQL] *****
ok: [user-VirtualBox]
TASK [Install Python packages] *****
changed: [user-VirtualBox]
TASK [Start and enable services] *****
ok: [user-VirtualBox]
TASK [Create database] *****
changed: [user-VirtualBox]
TASK [Create db user] *****
changed: [user-VirtualBox]
PLAY RECAP *****
user-VirtualBox      : ok=6   changed=3   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0

```

Успешный запуск, после добавления установки пакетов

На ВМ user-VirtualBox(192.168.100.14) появилась БД tms и пользователь admin:

Name	Owner	Encoding	Collate	Ctype	Access privileges
postgres	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	
template0	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=c/postgres +
template1	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=c/postgres +
tms	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	postgres=CTc/postgres

(4 rows)

```

postgres=# \du
List of roles
Role name | Attributes | Member of
-----+-----+-----
admin    |             | {}
postgres | Superuser, Create role, Create DB, Replication, Bypass RLS | {}

```

Добавляем также привилегии пользователю admin для БД tms и к примеру выполнение SQL-скрипта на добавление таблицы с сообщением:

Добавляем привилегии пользователю (admin):

```
- name: Grant user db_user to db
  postgresql_privs:
    type: database
    database: "{{ db_name }}"
    roles: "{{ db_user }}"
    grant_option: no
    privs: all
    become: yes
    become_user: postgres
```

Перед добавлением задачи в Ansible, создаем файл CREATE\_TABLE\_ANSIBLE.sql и переносим его на VM либо даем доступ к этому файлу:

```
CREATE TABLE IF NOT EXISTS test (
  message varchar(255) NOT NULL
);
INSERT INTO test(message) VALUES('Ansible is fun by tms');
ALTER TABLE test OWNER TO "admin";
```

Добавляем саму задачу в наш playbook:

```
- name: Add data to our db
  become: true
  become_user: postgres
  shell: "psql -d {{ db_name }} -f /ALL_FILES/CREATE_TABLE_ANSIBLE.sql"
```

Можно добавить смену порта, по умолчанию в PostgreSQL порт 5432, попробуем поменять его на 5434. Будем использовать модуль postgresql\_set. Эту задачу нужно добавить сразу после задачи установки самого PostgreSQL. Т.е. мы её добавляем в pre\_tasks:.

```
- name: Change listen port 5434 to db
  become: yes
  become_user: postgres
  postgresql_set:
    name: port
    value: "{{ db_port }}"
    notify:
```

- Restart PostgreSQL

Добавим notify который используется для связи задач с обработчиками handlers, для запуска определенных действий, в нашей ситуации, это перезапуск службы PostgreSQL, после глобальных настроек, типа смены портов:

```
handlers:
- name: Restart PostgreSQL
  systemd:
    name: postgresql
    state: restarted
```

Делаем запуск playbook:

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml

PLAY [Install PostgreSQL and create DB, db_user] *****

TASK [Gathering Facts] *****
ok: [user-VirtualBox]

TASK [Install PostgreSQL] *****
changed: [user-VirtualBox]

TASK [Change listen port 5433 to PG] *****
fatal: [user-VirtualBox]: FAILED! => {"changed": false, "msg": "Unsupported parameters for (postgresql_set) module: notify, state. Supported parameters include: ca_cert, connect_params, db, login_host, login_password, login_unix_socket, login_user, name, port, reset, session_role, ssl_cert, ssl_key, ssl_mode, trust_input, value (host, login, login_db, login_port, ssl_rootcert, unix_socket)."}

PLAY RECAP *****
user-VirtualBox : ok=2 changed=1 unreachable=0 failed=1 skipped=0 rescued=0 ignored=0
```

Ошибки с сменой порта, после добавления become: yes, ошибки пропали, но выяснилось это позже)

В параллель подготовил второй, рабочий вариант, замены порта, позже работал уже с ним:

```
- name: Change port to file /etc/postgresql/14/main/postgresql.conf
  become: yes
  become_user: postgres
  lineinfile:
    path: /etc/postgresql/14/main/postgresql.conf
    regexp: '^#?port ='
    line: 'port = {{ db_port }}'
  notify:
    - Restart PostgreSQL
```

После повторных запусков playbook появились следующие ошибки:

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml

PLAY [Install PostgreSQL and create DB, db_user] *****

TASK [Gathering Facts] *****
ok: [user-VirtualBox]

TASK [Install PostgreSQL] *****
ok: [user-VirtualBox]

TASK [Install Python packages] *****
ok: [user-VirtualBox]

TASK [Change port to file /etc/postgresql/14/main/postgresql.conf] *****
ok: [user-VirtualBox]

TASK [Start and enable services] *****
ok: [user-VirtualBox]

TASK [Create database] *****
fatal: [user-VirtualBox]: FAILED! => {"changed": false, "msg": "unable to connect to database: connection to server on socket \"/var/run/postgresql/.s.PGSQL.5433\" failed: No such file or directory\n\tIs the server running locally and accepting connections on that socket?\n"}

PLAY RECAP *****
user-VirtualBox : ok=5 changed=0 unreachable=0 failed=1 skipped=0 rescued=0 ignored=0
```

Ошибки на создание БД, т.е psql немог залогиниться из-за смены порта, покакой причине, мне небыло понятно

Кучу времени прошло, пытался сменить номер порта, к примеру на 5434, переустанавливал локально на ВМ полностью Postgres. + позже выяснилось, запуская playbook, ошибки начали валиться и на смене порта, и ошибки типа нет такого конфигурационного файла (/etc/postgresql/14/main/postgresql.conf (psql также из под пользователя postgres, не запускался)). После удаления пакетов postgres, а именно через --purge:

```
$ apt-get --purge remove postgresql*
```

При следующей установке, все файлы конфигурации появились!!! До этого удаление, выполнял без параметра --purge.

Но, ошибка с созданием БД, так и осталась. Пытаясь локально на ВМ, для проверки, создать пользователем postgres БД tms, также выдавало ошибку:

```
postgres@user-VirtualBox:~$ psql createdb tms
psql: error: connection to server on socket "/var/run/postgresql/.s.
PGSQL.5434" failed: FATAL: Peer authentication failed for user "tms"
postgres@user-VirtualBox:~$ psql createdb -p 5434 tms
```

Позже выяснилось, что нужно делать изменения внутри файла pg\_hba.conf

```
# "local" is for Unix domain socket connections only
local all all peer
# IPv4 local connections:
host all all 127.0.0.1/32 scram-sha-256
# IPv6 local connections:
host all all ::1/128 scram-sha-256
```

Последние значения нужно поменять на trust.

На ВМ после корректировки pg\_hba.conf, пробуем запустить повторно создание БД:

`$ psql createdb tms2` \\ пытаемся создать tms2, т.к. БД tms уже руками через psql, для проверки также создавал, в psql проблем не было

```
postgres@user-VirtualBox:~$ createdb tms
createdb: error: database creation failed: ERROR: database "tms" already exists
postgres@user-VirtualBox:~$ createdb tms2
```

Значит, по мимо смены порта, нам необходимо добавить корректировку файла pg\_hba.conf:

```
- name: Change file /etc/postgresql/14/main/pg_hba.conf
  become: yes
  become_user: postgres
  shell: sed -i 's/peer/trust/; s/scram-sha-256/trust/; s/scram-sha-256/trust/'
  /etc/postgresql/14/main/pg_hba.conf
```

Также путем тестирования и запусков нашего Playbook, все также не запускалось создание БД, выяснилось, что при смене порта, нужно везде, где делается создание БД, пользователя и назначение грантов, добавить наш измененный port:



```

- name: Create database
  postgresql_db:
    state: present
    port: "{{ db_port }}"
    name: "{{ db_name }}"
    become: yes
    become_user: postgres

- name: Create db user
  postgresql_user:
    state: present
    port: "{{ db_port }}"
    name: "{{ db_user }}"
    password: "{{ db_password }}"
    become: yes
    become_user: postgres

- name: Grant user db_user to db
  postgresql_privs:
    type: database
    port: "{{ db_port }}"
    database: "{{ db_name }}"
    roles: "{{ db_user }}"
    grant_option: no
    privs: all

```

После всех корректировок, удаляем с VM полностью postgresql и запускаем наш полностью исправленный playbook (install\_pg.yml), его приложу в директории с pdf-файлом:

```

makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml

PLAY [Install PostgreSQL and create DB, db_user] *****

TASK [Gathering Facts] *****
ok: [user-VirtualBox]

TASK [Install PostgreSQL] *****
changed: [user-VirtualBox]

TASK [Install Python packages] *****
ok: [user-VirtualBox]

TASK [Change file /etc/postgresql/14/main/pg_hba.conf] *****
changed: [user-VirtualBox]

TASK [Change port to file /etc/postgresql/14/main/postgresql.conf] *****
changed: [user-VirtualBox]

RUNNING HANDLER [Restart PostgreSQL] *****
changed: [user-VirtualBox]

TASK [Start and enable services] *****
ok: [user-VirtualBox]

TASK [Create database] *****
changed: [user-VirtualBox]

TASK [Create db user] *****
changed: [user-VirtualBox]

TASK [Grant user db_user to db] *****
changed: [user-VirtualBox]

TASK [Add data to our db] *****
changed: [user-VirtualBox]

PLAY RECAP *****
user-VirtualBox      : ok=11  changed=8  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0

makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ nano install_pg.yml

```

Успешное выполнение всех задач, которые мы указали в нашем файле install\_pg.yml

Также можем проверить локально на VM, БД, пользователя, порт и созданную таблицу:

```

$ psql
psql=#\list    \\выводит список существующих БД в PG

```

List of databases					
Name	Owner	Encoding	Collate	Ctype	Access privileges
postgres	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	
template0	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=c/postgres +
template1	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=c/postgres +
tms	postgres	UTF8	en_US.UTF-8	en_US.UTF-8	=Tc/postgres +
					postgres=CTc/postgres+ admin=CTc/postgres

(4 rows)

`psql=#\du`      `\\`выводит список существующих пользователей в БД

```
postgres=# \du
```

List of roles		
Role name	Attributes	Member of
admin		{}
postgres	Superuser, Create role, Create DB, Replication, Bypass RLS	{}

Пробуем подключится к БД, обязательно указывая порт и пользователя:

`$ psql tms -h localhost -p 5434 -U admin`

`$ tms=> select * from test;`

```
postgres@user-VirtualBox:~$ psql tms -h localhost -p 5434 -U admin
psql (14.15 (Ubuntu 14.15-0ubuntu0.22.04.1))
SSL connection (protocol: TLSv1.3, cipher: TLS_AES_256_GCM_SHA384, bits: 256, compression: off)
Type "help" for help.

tms=> select * from test;
      message
-----
 Ansible is fun by tms
(1 row)

tms=> █
```

Ansible is fun by tms

Создаем роли в Ansible:

`$ mkdir role`

`\\` создаем директорию roles

`$ ansible-galaxy init deploy_postgresql`

`\\` создание роли с названием deploy\_postgresql

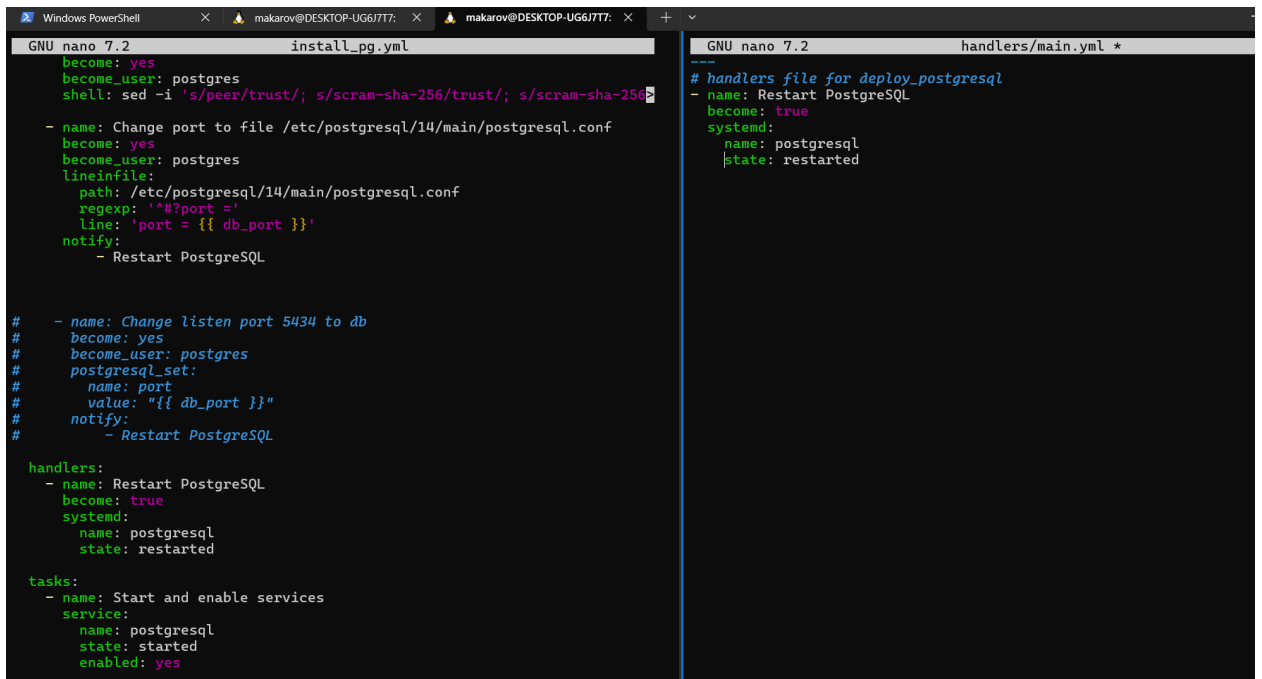
`$ tree`

`\\` показывает структуру директории roles

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ mkdir roles
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-galaxy init deploy_postgresql
- Role deploy_postgresql was created successfully
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ tree
.
├── ansible.cfg
├── deploy_postgresql
│   ├── defaults
│   │   └── main.yml
│   ├── files
│   ├── handlers
│   │   └── main.yml
│   ├── meta
│   │   └── main.yml
│   ├── README.md
│   ├── tasks
│   │   └── main.yml
│   ├── templates
│   ├── tests
│   │   ├── inventory
│   │   └── test.yml
│   └── vars
│       └── main.yml
├── hosts.ini
├── install_pg.yml
├── roles
└── vars.yml

11 directories, 12 files
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ █
```

Переносим наш playbook install\_pg.yml в роль deploy\_postgresql:



```
GNU nano 7.2 install_pg.yml
---
- name: Install PostgreSQL
  become: yes
  become_user: postgres
  shell: sed -i 's/peer/trust/; s/scram-sha-256/trust/; s/scram-sha-256/...'

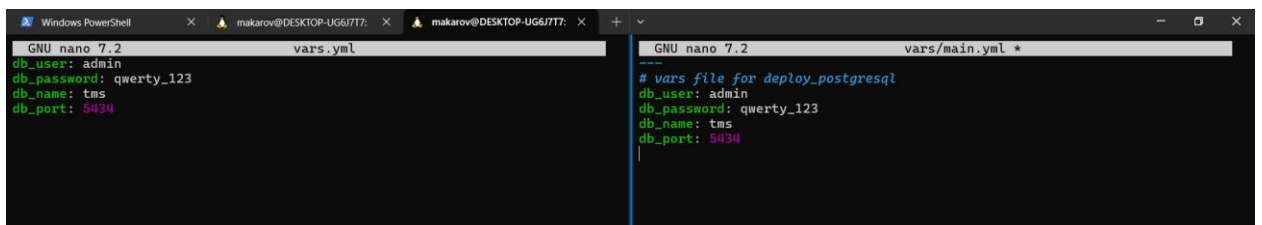
- name: Change port to file /etc/postgresql/14/main/postgresql.conf
  become: yes
  become_user: postgres
  lineinfile:
    path: /etc/postgresql/14/main/postgresql.conf
    regexp: '^#?port ='
    line: 'port = {{ db_port }}'
  notify:
    - Restart PostgreSQL

#
# - name: Change listen port 5434 to db
#   become: yes
#   become_user: postgres
#   postgresql_set:
#     name: port
#     value: "{{ db_port }}"
#   notify:
#     - Restart PostgreSQL

handlers:
  - name: Restart PostgreSQL
    become: true
    systemd:
      name: postgresql
      state: restarted

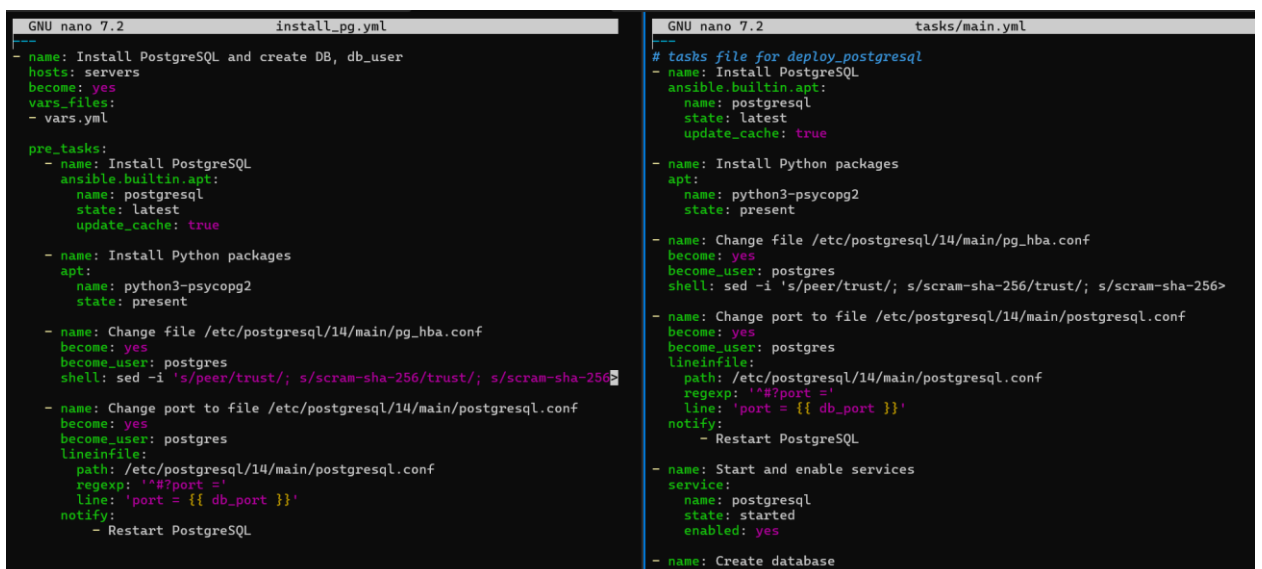
tasks:
  - name: Start and enable services
    service:
      name: postgresql
      state: started
      enabled: yes
```

Переносим handlers



```
GNU nano 7.2 vars.yml
db_user: admin
db_password: qwerty_123
db_name: tms
db_port: 5434
```

Переносим vars



```
GNU nano 7.2 tasks/main.yml
---
# tasks file for deploy_postgresql
- name: Install PostgreSQL
  ansible.builtin.apt:
    name: postgresql
    state: latest
    update_cache: true

- name: Install Python packages
  apt:
    name: python3-psycopg2
    state: present

- name: Change file /etc/postgresql/14/main/pg_hba.conf
  become: yes
  become_user: postgres
  shell: sed -i 's/peer/trust/; s/scram-sha-256/trust/; s/scram-sha-256/...'

- name: Change port to file /etc/postgresql/14/main/postgresql.conf
  become: yes
  become_user: postgres
  lineinfile:
    path: /etc/postgresql/14/main/postgresql.conf
    regexp: '^#?port ='
    line: 'port = {{ db_port }}'
  notify:
    - Restart PostgreSQL

- name: Start and enable services
  service:
    name: postgresql
    state: started
    enabled: yes

- name: Create database
  postgresql_db:
```

Переносим наши tasks

Также нужно поправить наш файл install\_pg.yml:

```
---
- name: Install PostgreSQL, configure and create db
  hosts: servers
  become: yes
```

roles:

- { role: deploy\_postgresql, when: ansible\_system == 'Linux' }

\$ ansible-playbook install\_pg.yml

```
makarov@DESKTOP-UG6J7T7:~/ansible_postgres$ ansible-playbook install_pg.yml

PLAY [Install PostgreSQL, configure and create db] *****

TASK [Gathering Facts] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Install PostgreSQL] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Install Python packages] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Change file /etc/postgresql/14/main/pg_hba.conf] *****
changed: [user-VirtualBox]

TASK [deploy_postgresql : Change port to file /etc/postgresql/14/main/postgresql.conf] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Start and enable services] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Create database] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Create db user] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Grant user db_user to db] *****
ok: [user-VirtualBox]

TASK [deploy_postgresql : Add data to our db] *****
changed: [user-VirtualBox]

PLAY RECAP *****
user-VirtualBox      : ok=10  changed=2  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0
```

Успешное выполнение!

Все конфигурационные файлы Ansible находятся вместе с pdf-отчетом.