PODBA using Ansible Automation Platform 2 (GUI)

Overview: Ansible Automation Platform Version 2 offers new features and components. This readme outlines the process of executing the PODBA playbooks using AAP2 GUI.

Reference to AAP2: https://www.ansible.com/blog/introducing-ansible-automation-platform-2

AAP2 installation instructions can be found here:

Chapter 3. Installing Red Hat Ansible Automation Platform Red Hat Ansible Automation Platform 2.3 | Red Hat Customer Portal

Prepare the Execution Environment: This is a one-time setup before executing playbooks from AAP2 GUI.

- Login to the Ansible controller with “awx” user.
- Install python version 3.8 or later [# dnf install python3]
- Install podman using dnf [# dnf install podman]
- Install ansible-builder [$ pip install ansible-builder]
- Install ansible-navigator [$ pip install ansible-navigator], this is not mandatory for GUI.
- In any present working directory, create a directory named "context".
- Download and extract Oracle Instant client software from Oracle site: https://www.oracle.com/database/technologies/instant-client/downloads.html
- Note: For Linux on Power, click on “other platforms” in the above URL.
- Inside the "context" directory place the extracted oracle client software directory with the name "oracle_client".
- Create a file (example: create_podba.yml) with the following content.

```
version: 3

images:
  base_image:
    name: registry.redhat.io/ansible-automation-platform-24/ee-minimal-rhel8:latest
options:
  package_manager_path: /usr/bin/microdnf
additional_build_steps:
  append_base:
    - RUN microdnf install gcc python39-devel libnsl* libaio* find* which* sudo dnf
    - RUN pip3 install wheel
    - RUN python3.9 -m pip install cx_Oracle --upgrade
    - RUN ln -s /usr/lib64/libnsl.so.2 /usr/lib64/libnsl.so.1
    - COPY oracle_client /oracle_client_sw
    - COPY ansible-automation-platform-managed-ca-cert.crt /etc/pki/ca-trust/source/anchors
dependencies:
  galaxy: requirements.yml
```

- Make the requirements.yml file

```
collections:
  - ibm.power_aix
```

- Run the following command to build the execution environment image.

```
$ ansible-builder build -t powerodba -f create_podba.yml Running
command:
  podman build -f context/Containerfile -t powerodba context
Complete! The build context can be found at: /var/lib/awx/aap2/context
```

- List the images:

```
$ podman images
REPOSITORY                                                  TAG         IMAGE ID      CREATED             SIZE localhost/powerodba
latest      e04948d6013a  About a minute ago  908 MB
registry.redhat.io/ansible-automation-platform-24/ee-supported-rhel8  latest      b2d26de2d8de  4 months ago
1.79 GB registry.redhat.io/ansible-automation-platform-24/ee-minimal-rhel8   latest      c239714e9480  4 months ago      380
MB
```

Execute playbooks from CLI using ansible-navigator:

1. Create a file called ansible-navigator.yml inside the {{ collection_name }}/playbooks directory with the following content.

```
$ cat ansible-navigator.yml
---
ansible-navigator:      execution-
environment:        enabled: True
  image: powerodba:latest        # Name of the REPOSITORY:TAG
```

2. Follow the readme files under "docs" to understand how to update the required variables for each task.
3. Run the following command to execute the playbooks. The following example shows execution of manage-db-directories.yml playbook.

```
ansible-navigator run <playbook name> --pp=missing --m stdout -i <name of inventory file>
Example:
$ ansible-navigator run manage-db-directories.yml --pp=missing --m stdout -i hosts.yml
```

To use the escalated privileges, please use "--playbook-artifact-enable false" at the end of the command.

Example:

ansible-navigator run db-opatch.yml --pp=missing --m stdout -i hosts.yml --ask-become-pass --playbookartifact-enable false

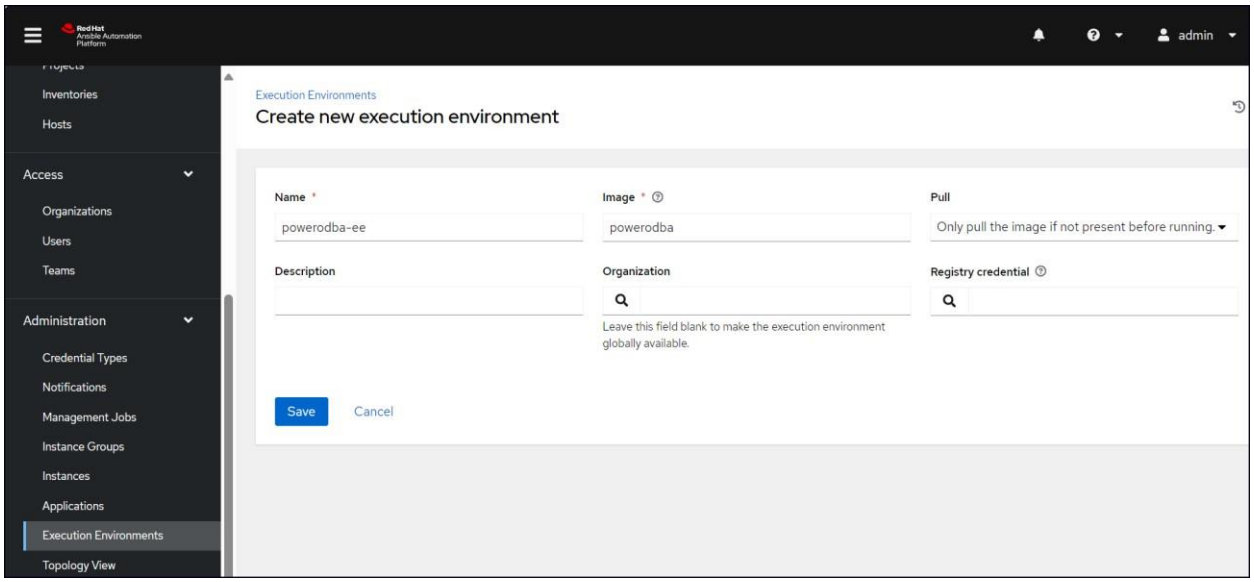
Execute the playbooks from GUI:

PODBA has two types of modules – few uses “cx_Oracle” module which is a python connector to access Oracle databases and the others won’t use cx_Oracle, they require “ssh” connectivity.

The playbook template setup will be a little different between the two types of modules. Hence, we’re going to setup a project in AAP2 and demonstrate one example playbook which uses cx_Oracle and another one which doesn’t require cx_Oracle.

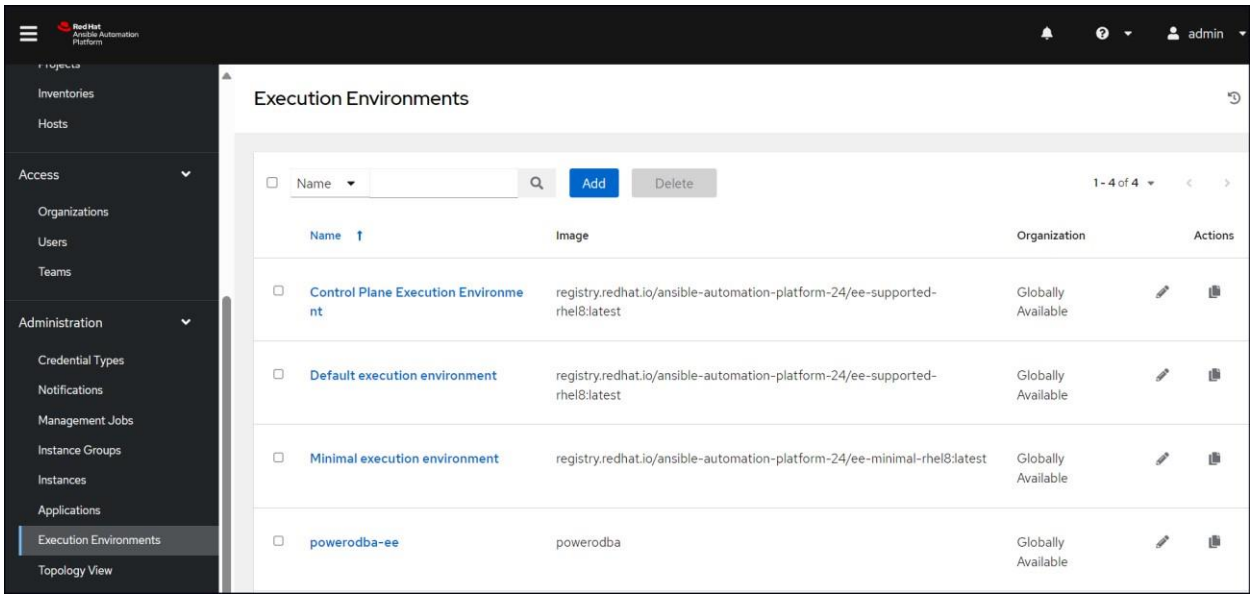
Create Project:

Step 1: Login to AAP2 and “Create new execution environment”.

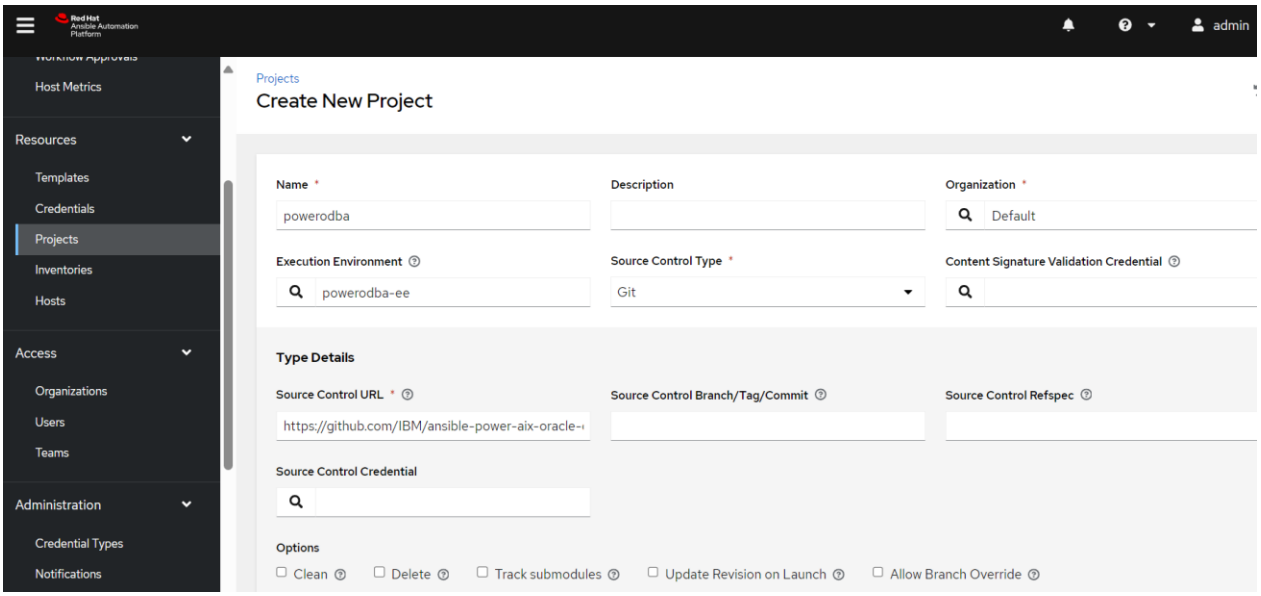


Name: Any desired name for identification.

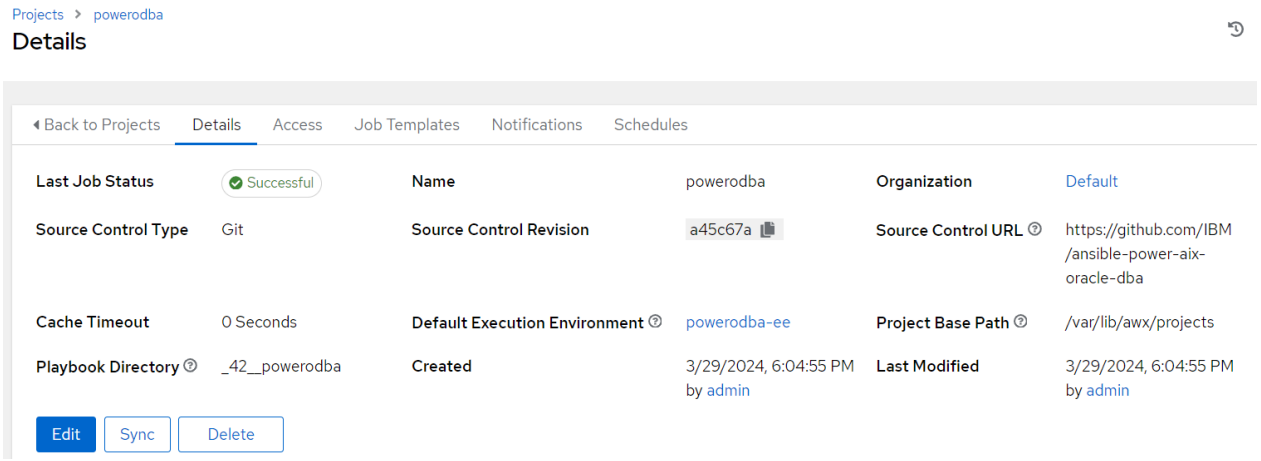
Image: Provide the name of the image which was created earlier in this document using “ansible-builder”.



Step 2: Create a new project.

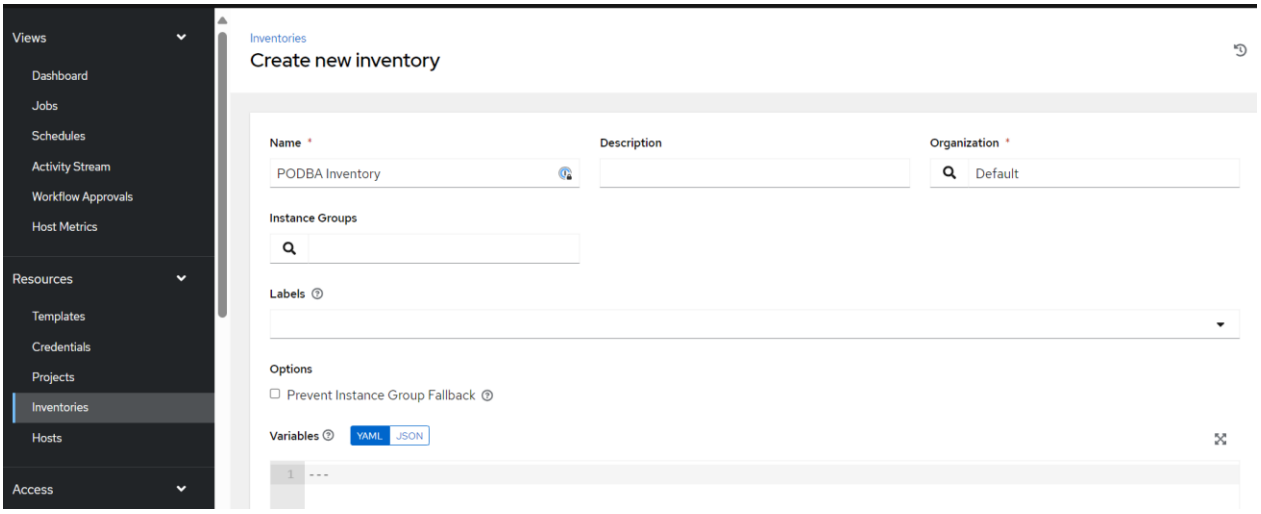


Name: Any desired name for identification.
Organization: Your existing Organization name or leave it “Default”
Execution Environment: Name of the Execution Environment created in Step 1 of **Create Project** section.
Source Control Type: Git
Source Control URL: https://github.com/IBM/ansible-power-aix-oracle-dba
Click “save”
The “Last Job Status” must show Successful as shown below.



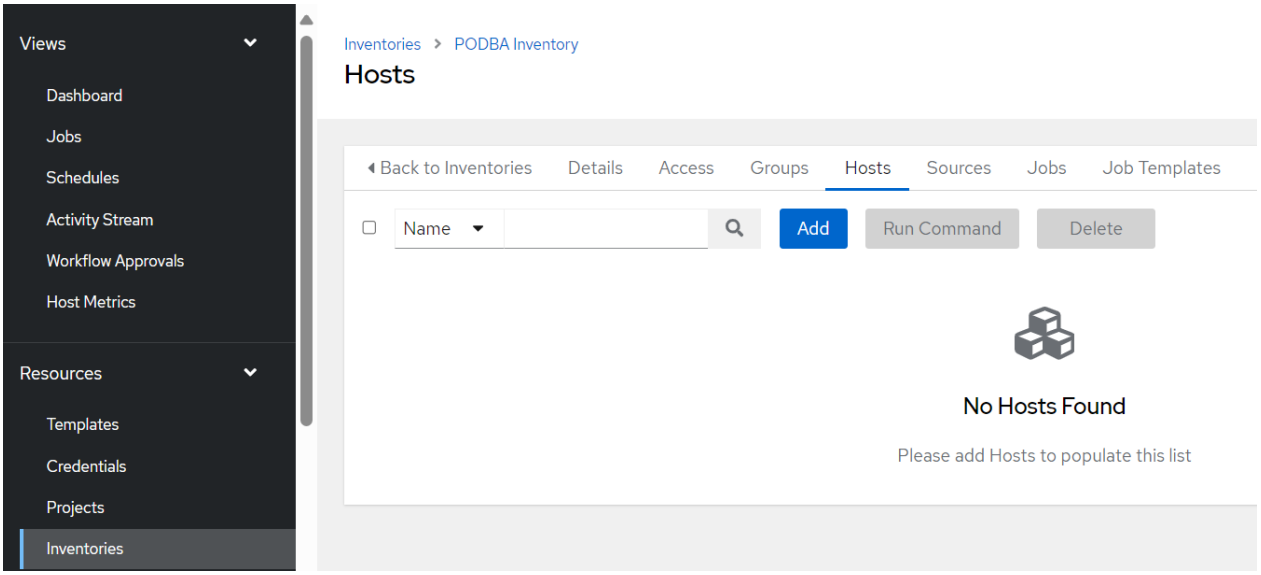
Example 1 – Create Database : We are going to demonstrate how to create a Single Instance container database (CDB) named “testdb”, this database will reside on Automatic Storage Management (ASM).

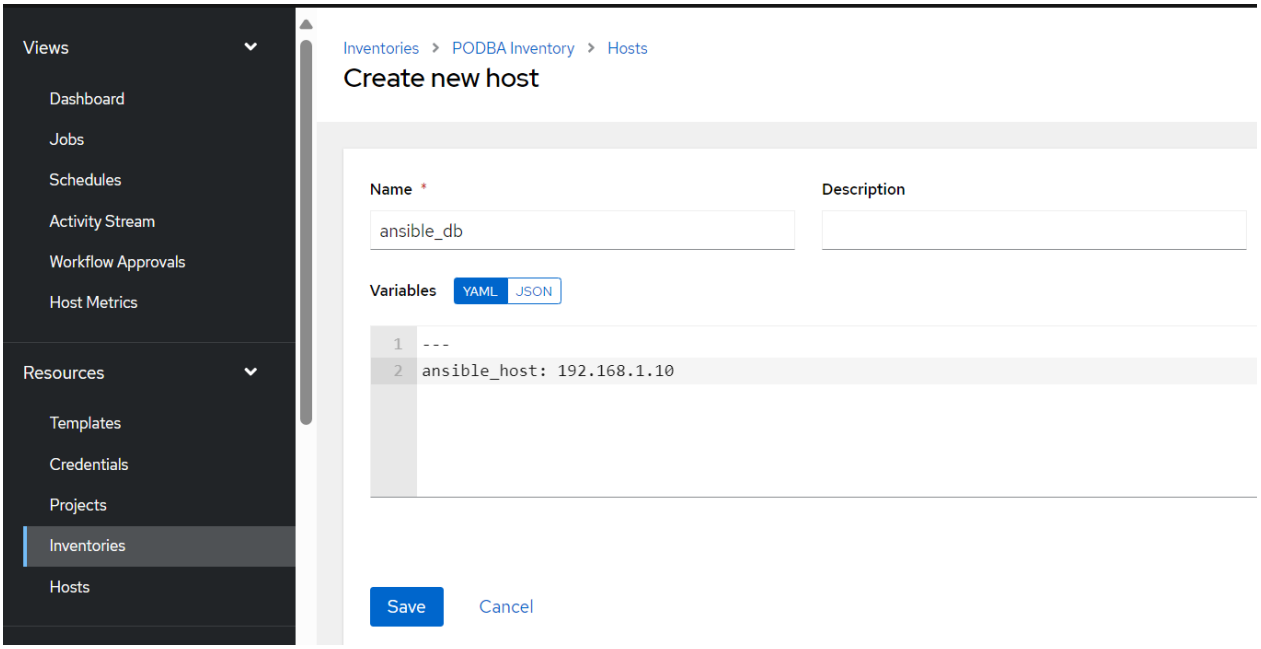
Step 1: Use an existing inventory or create a new one. In this case, I’m creating a new one.



Name: PODBA Inventory (provide any name for identification) and Click on Save.

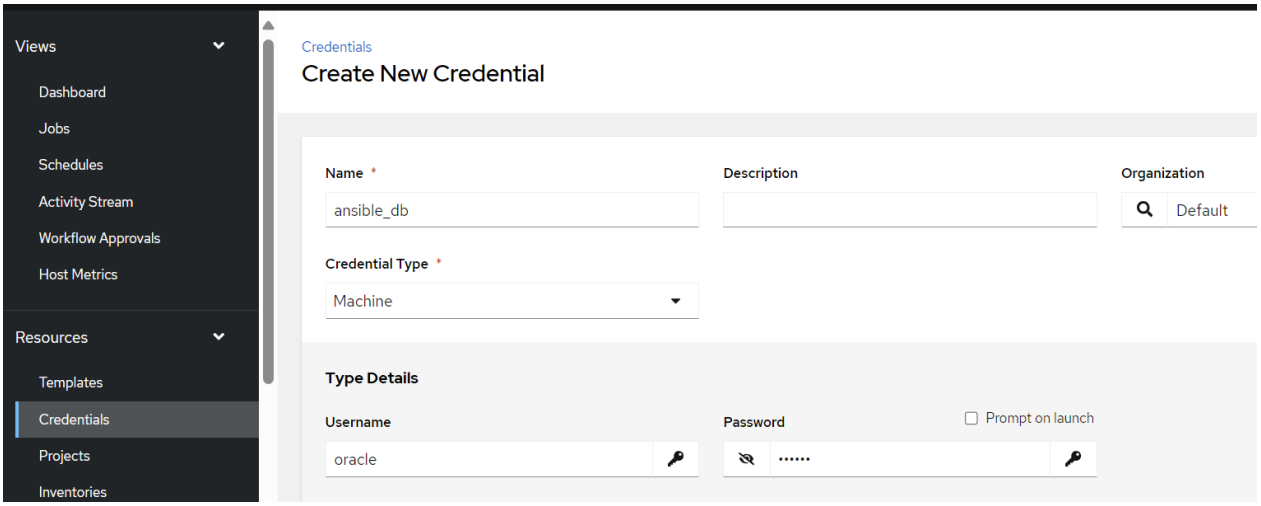
Step 2: Add the target lpar hostname where you need to create the database to the inventory.





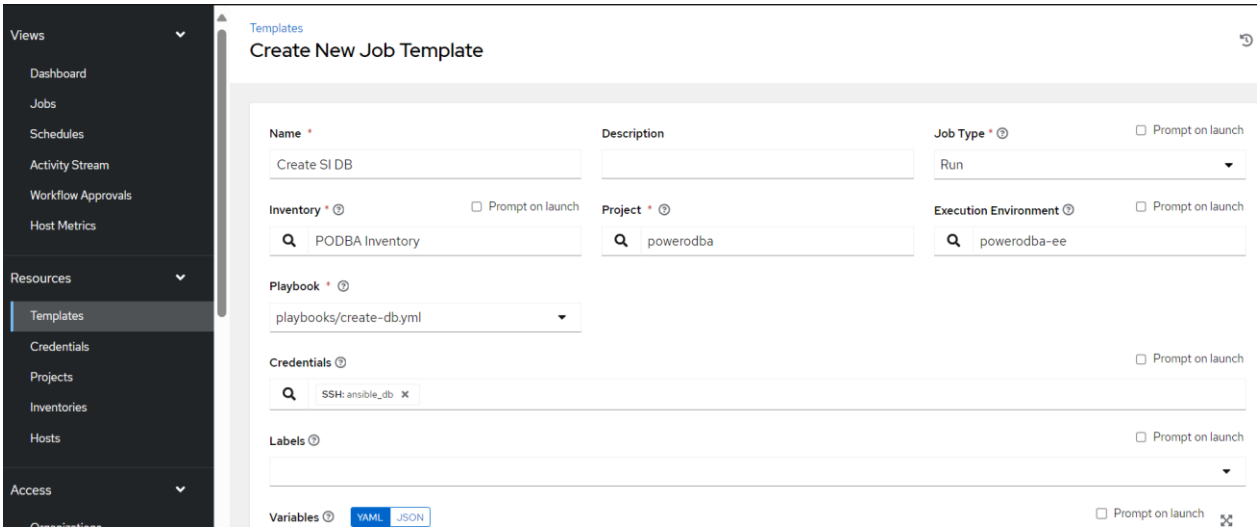
Name: provide the hostname of the lpar along with the host IP address under the variables section and Click on Save.

Step 3: Setup the credentials of the lpar which we have added in the PODBA inventory in the previous step.



Name: provide any name for identification and update the rest of the details as shown in the above fig. and Click on Save.

Step 4: Create a new job template for “Create DB” playbook.



Name: Create SI DB (any name for identification). Update all the other details as shown in the above fig.

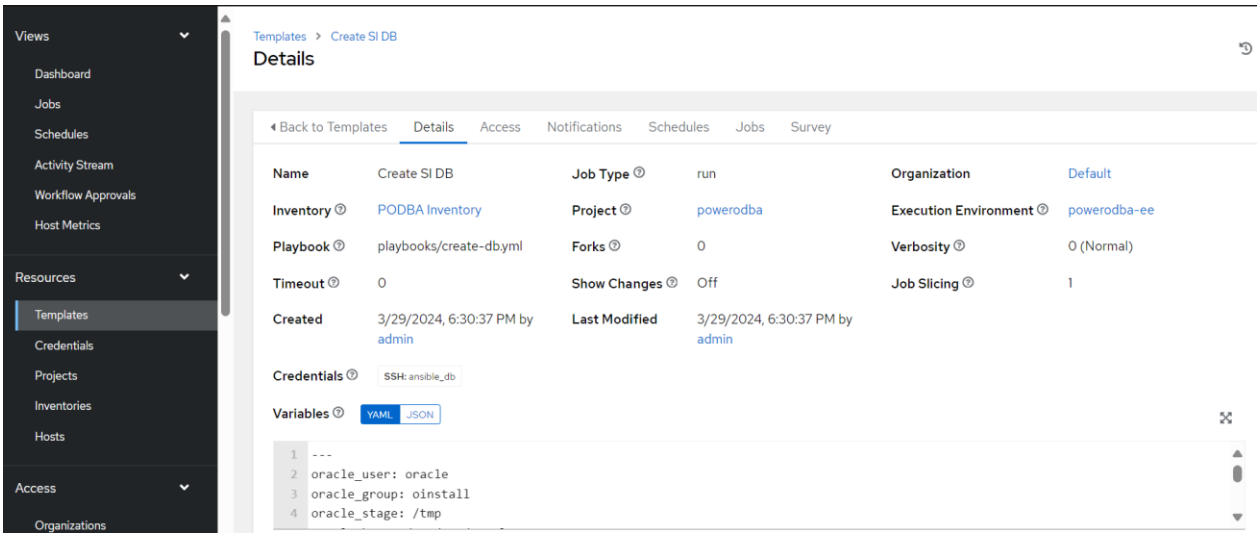
Variables: This section is an important one. Please refer the readme “Readme-Create-Database.txt” in the “docs” folder of this collection to understand more about each variable. I have used the following variables.

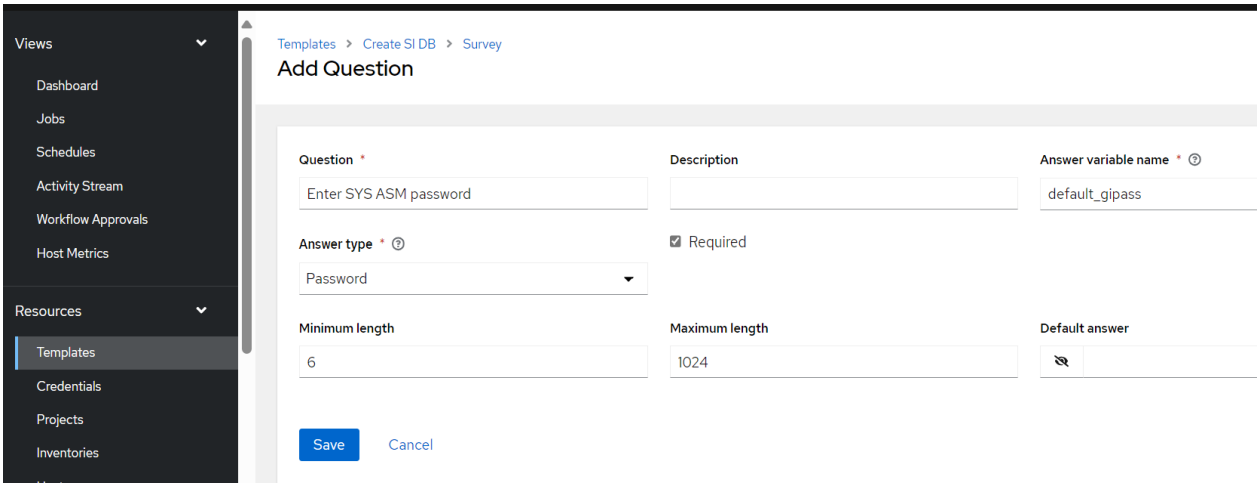
```
oracle_user: oracle
oracle_group: oinstall
oracle_stage: /tmp
oracle_base: /u01/app/oracle
oracle_dbf_dir_fs:
oracle_reco_dir_fs:
oracle_dbf_dir_asm: '+DATA1'
oracle_reco_dir_asm: '+DATA1'

oracle_databases:
  - home: db1
    oracle_version_db: 19.3.0.0
    oracle_home: /u01/db19c_2
    oracle_edition: EE
    oracle_db_name: testdb
    oracle_db_type: SI
    is_container: True
    pdb_prefix: testpdb
    num_pdbs: 1
    storage_type: ASM
    service_name: testdb
    oracle_init_params: "sga_max_size=10g,sga_target=10g"
    oracle_db_mem_totalmb: 15000
    oracle_database_type: MULTIPURPOSE
    redolog_size_in_mb: 50
    state: present
```

Click on Save.

Step 5: After saving the template, create two surveys for prompting the “SYS password of ASM” and a new “SYS user password” during runtime.

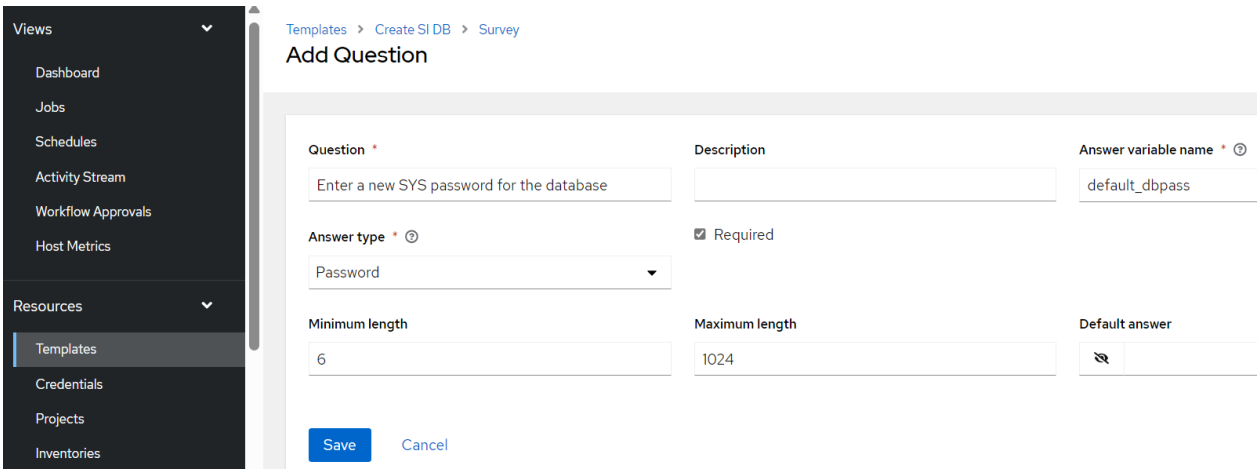




Question: Enter something to type the password, this will be prompted during runtime.
Answer variable name: Update “default_gipass”, this is explicitly set inside the roles.
Answer type: Password.

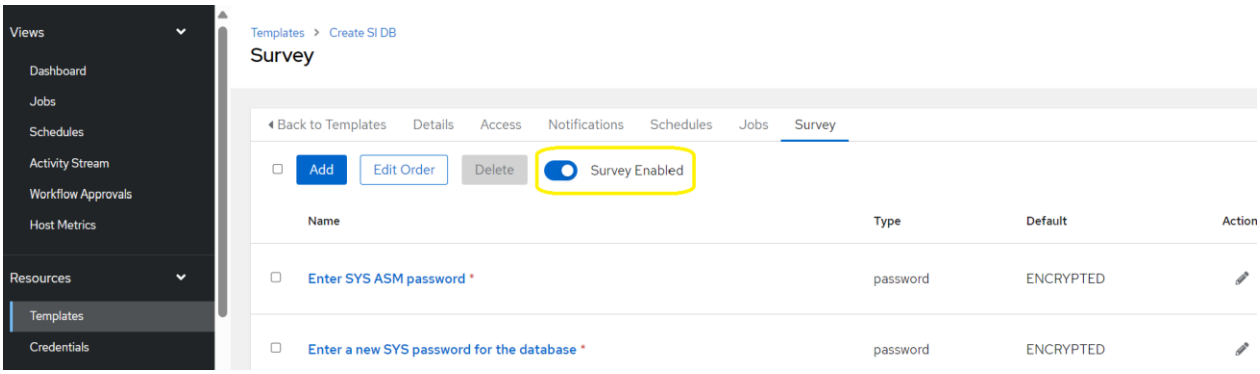
Click on Save.

Step 6: Create another survey for the new database's SYS user.

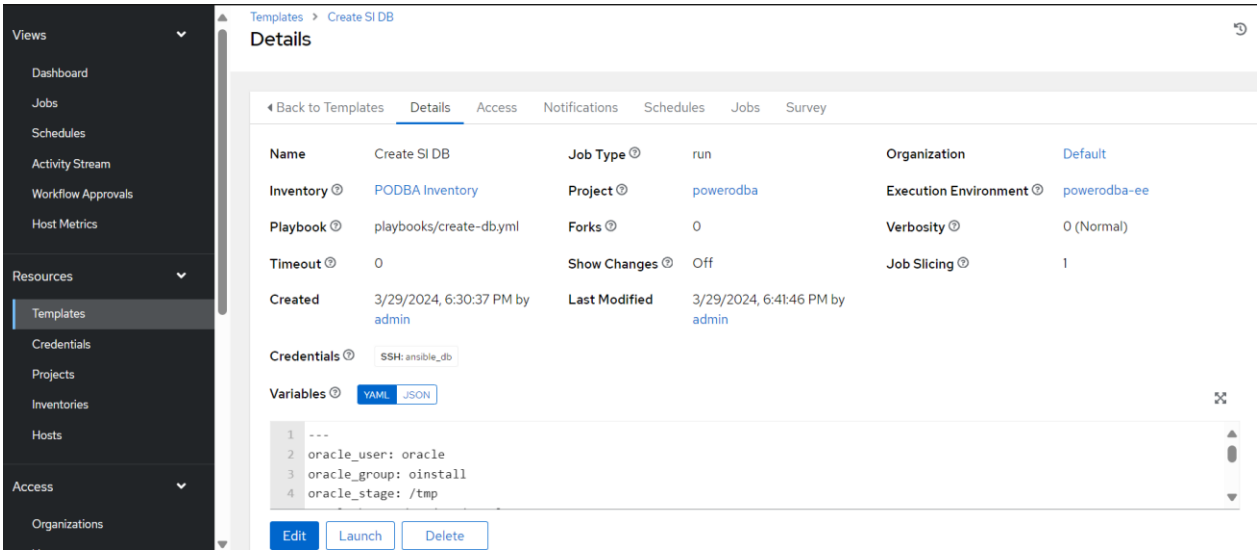


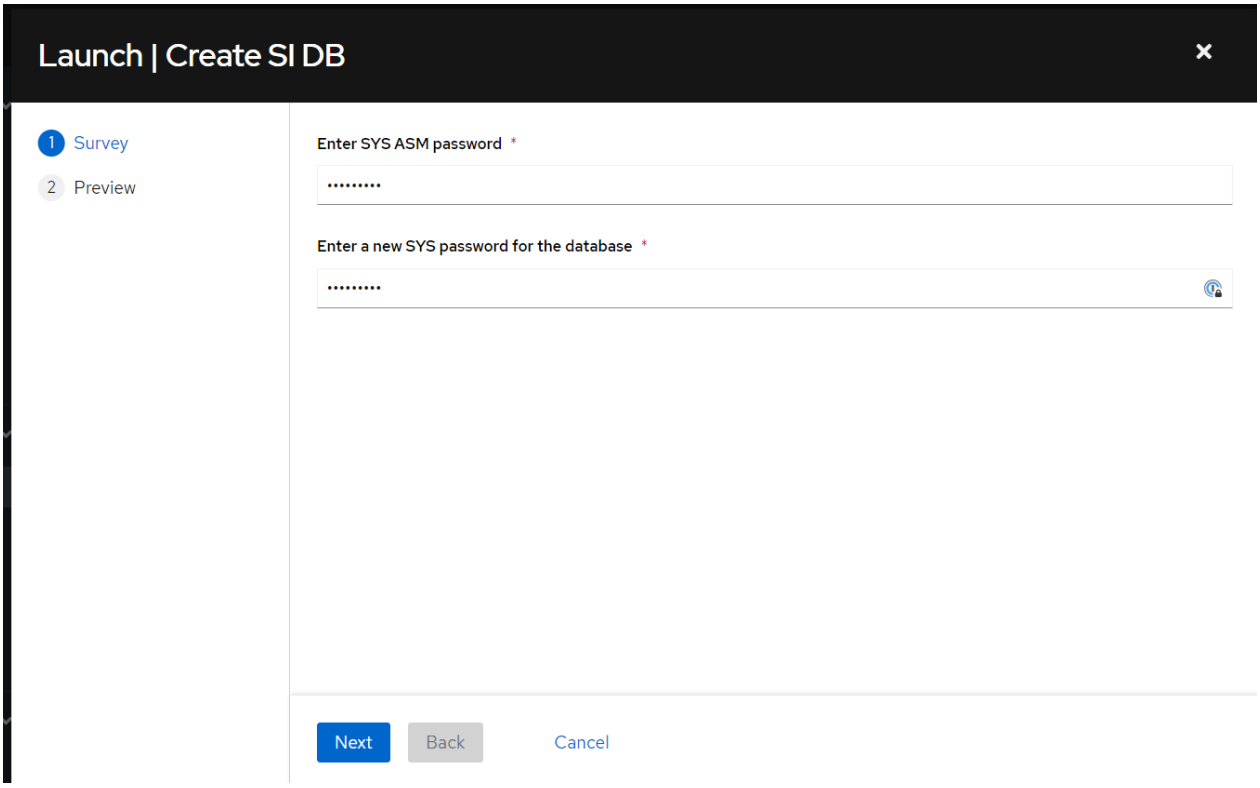
Click on Save.

Step 6: Make sure to toggle "Survey Enabled".

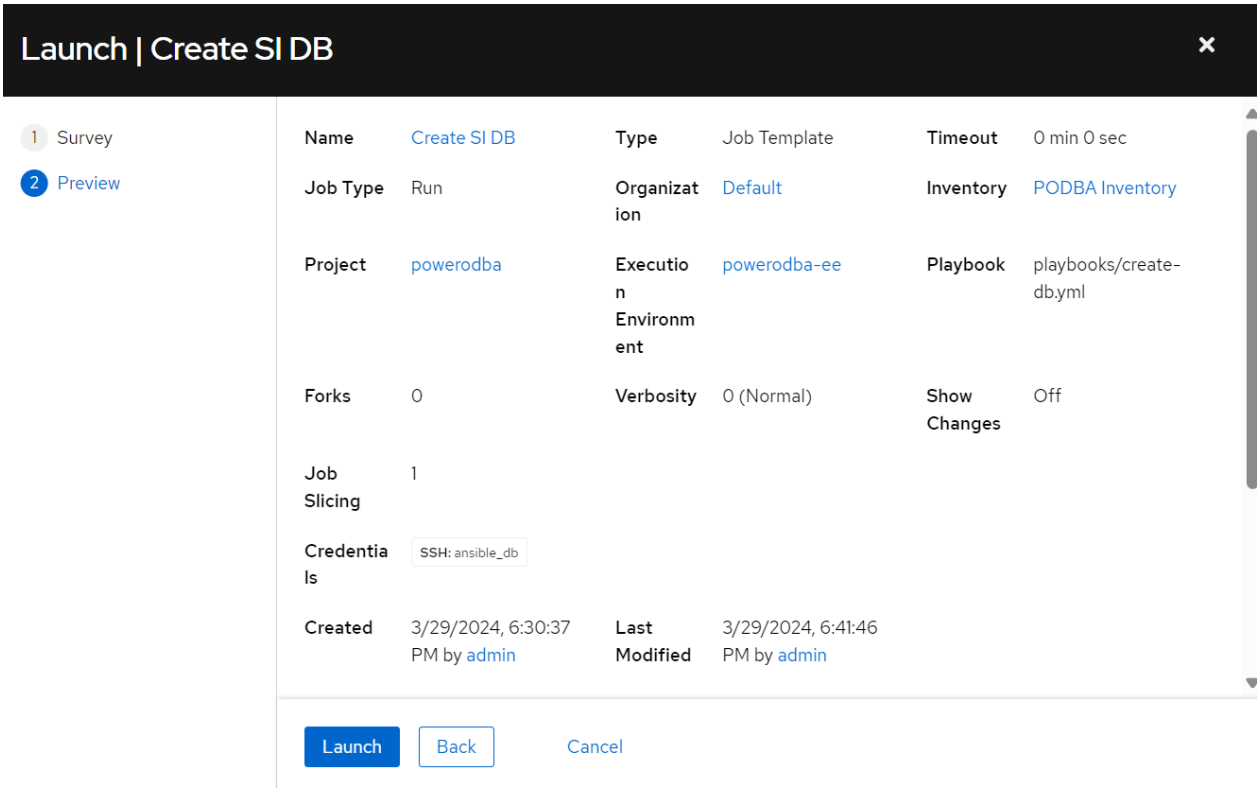


Step 7: Launch the “Create SI DB” template



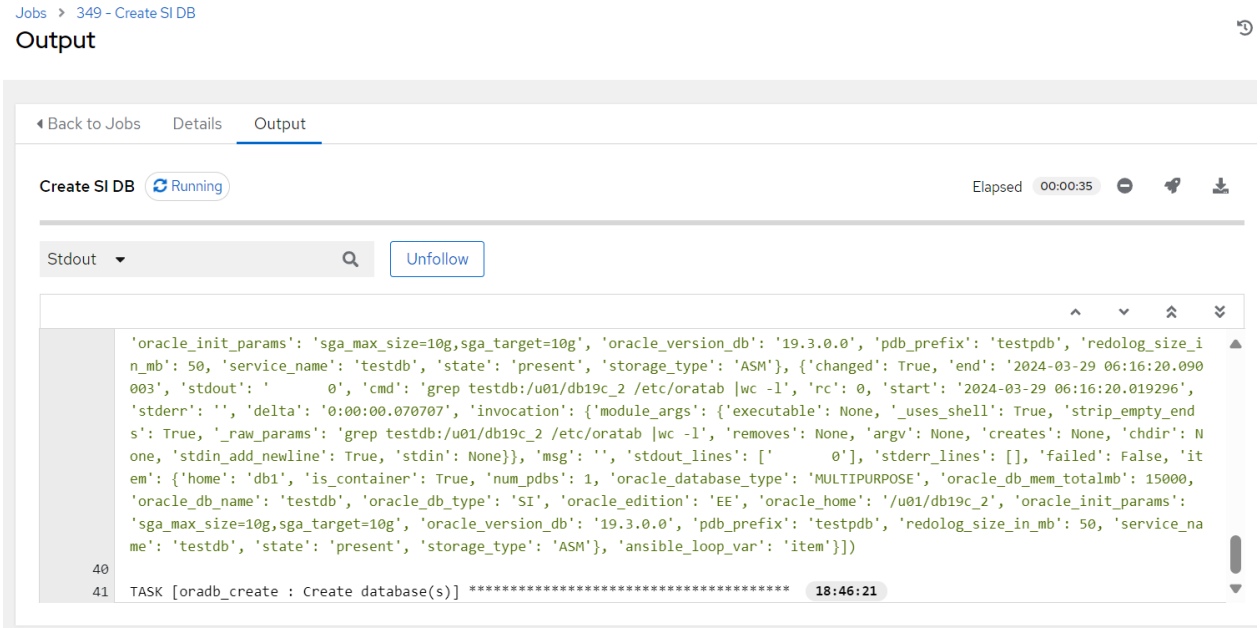


Input the passwords and click Next.



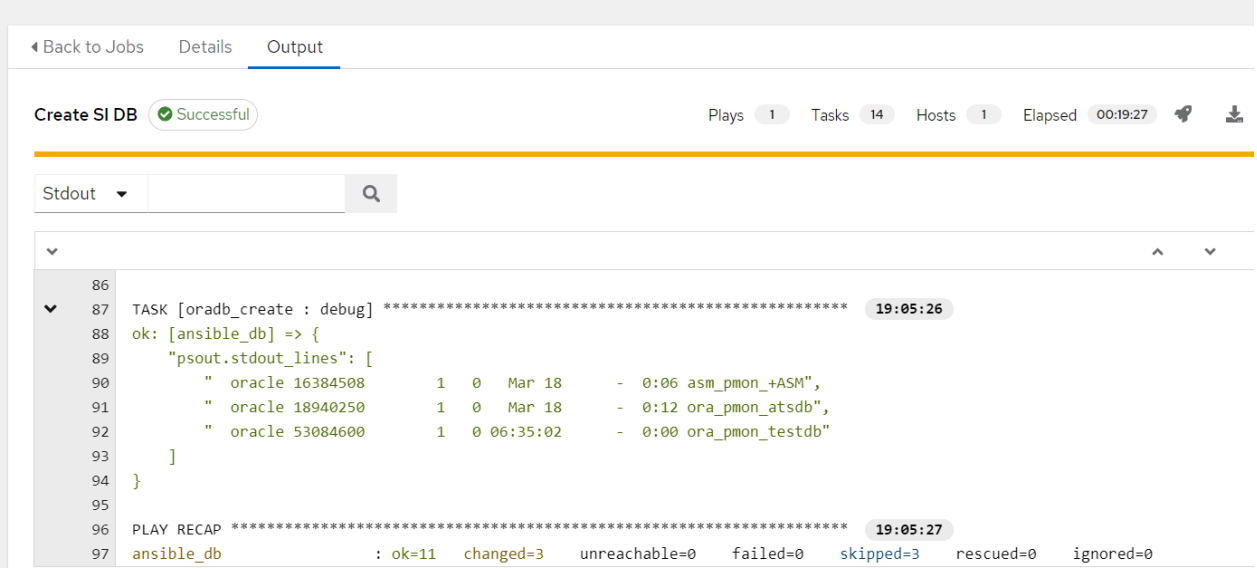
Click on Launch.

The status of the job shows running along with the output.



Jobs > 349 - Create SI DB

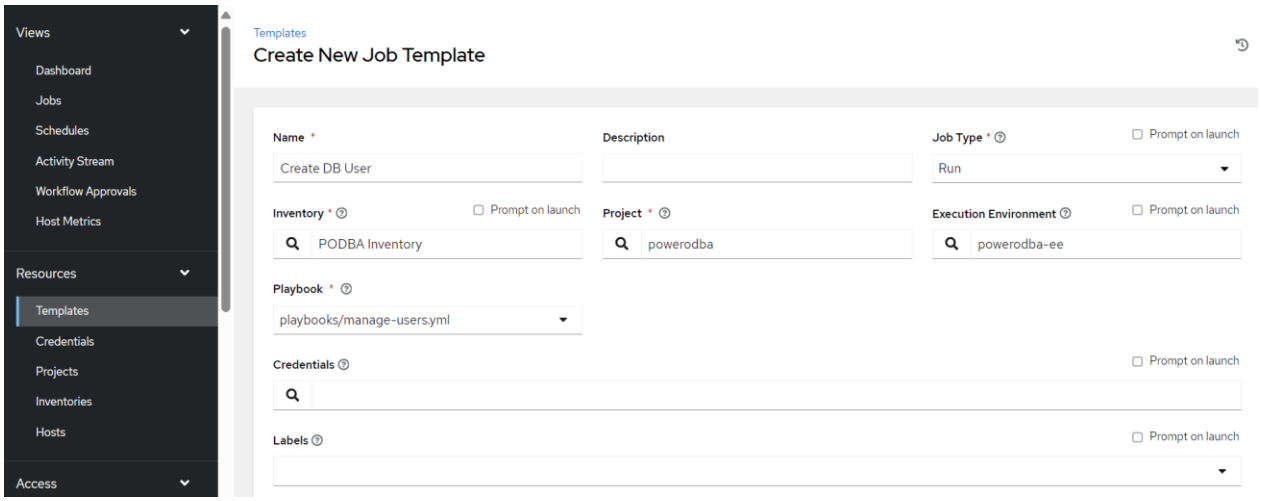
Output



DB creation successful.

Example 2 – Create DB users: We are going to demonstrate how to create two DB users namely testuser1 and testuser2 in a PDB called testpdb.

Step 1: Create a new job template for “Create User” playbook.



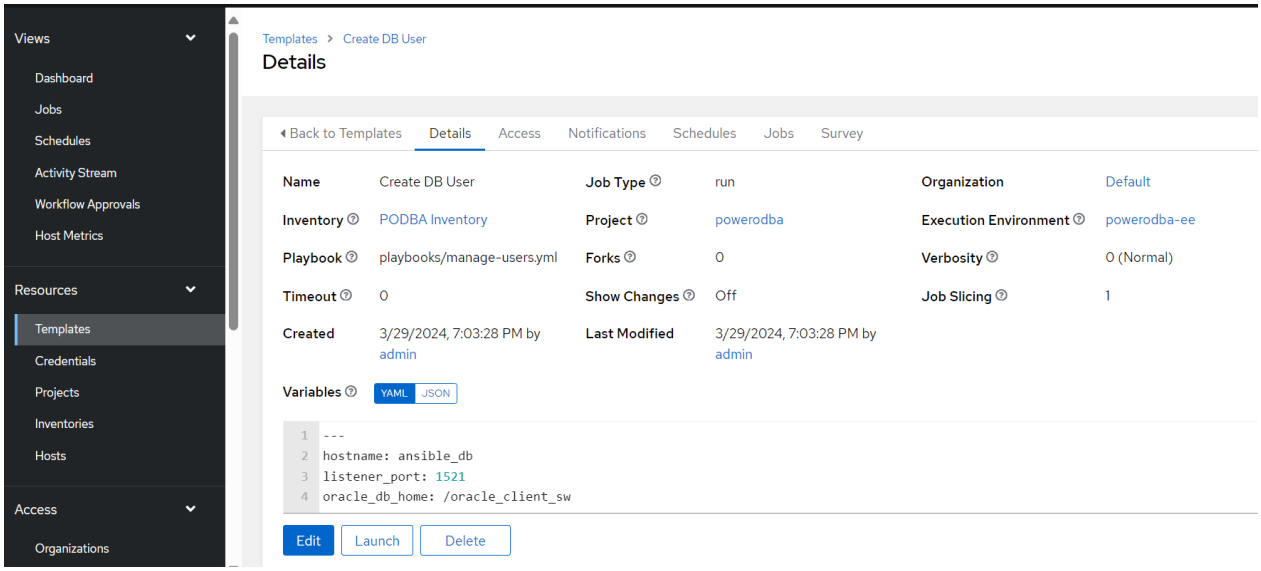
Step 2: Please refer the readme “Readme-Manage-Users.txt” in the “docs” folder of this collection to understand more about each variable. I have used the following variables.

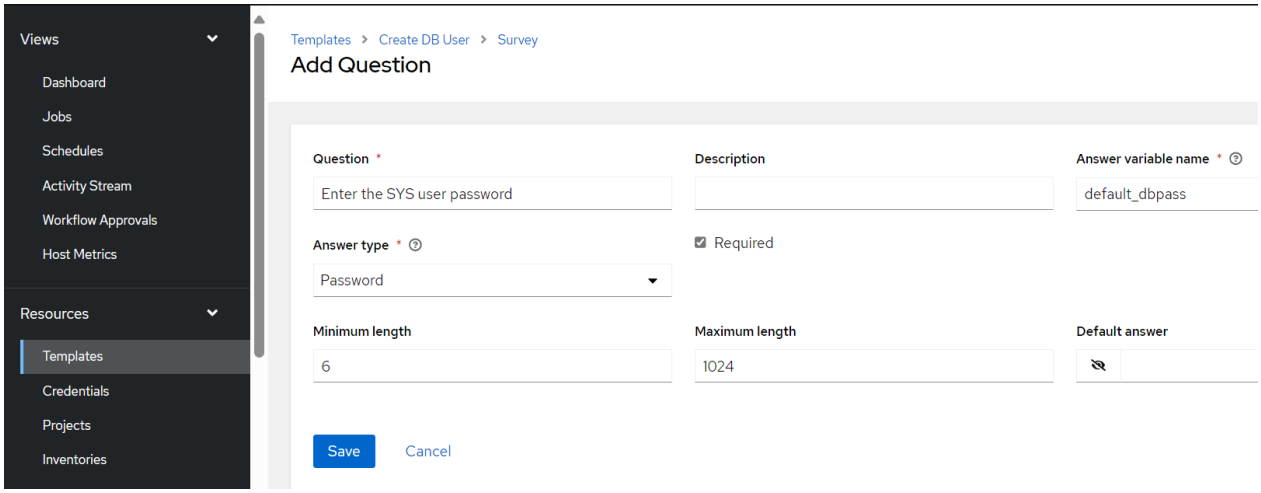
```
hostname: ansible_db
listener_port: 1521
oracle_db_home: /oracle_client_sw

oracle_databases:
      - users:
         - schema: testuser1
        default_tablespace: users
        service_name: testpdb
        schema_password: oracle4u
        grants_mode: enforce
        grants:
         - connect
         - resource
        state: present
      - users:
         - schema: testuser2
        default_tablespace: users
        service_name: testpdb
        grants_mode: enforce
        grants:
         - connect
        schema_password: oracle4u
        state: present
```

Click on Save.

Step 3: After saving the template, create a survey for prompting the “SYS user password” during runtime.

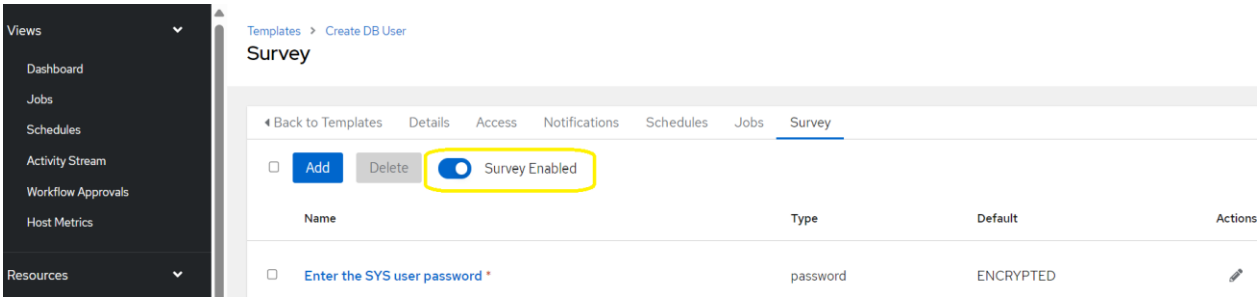




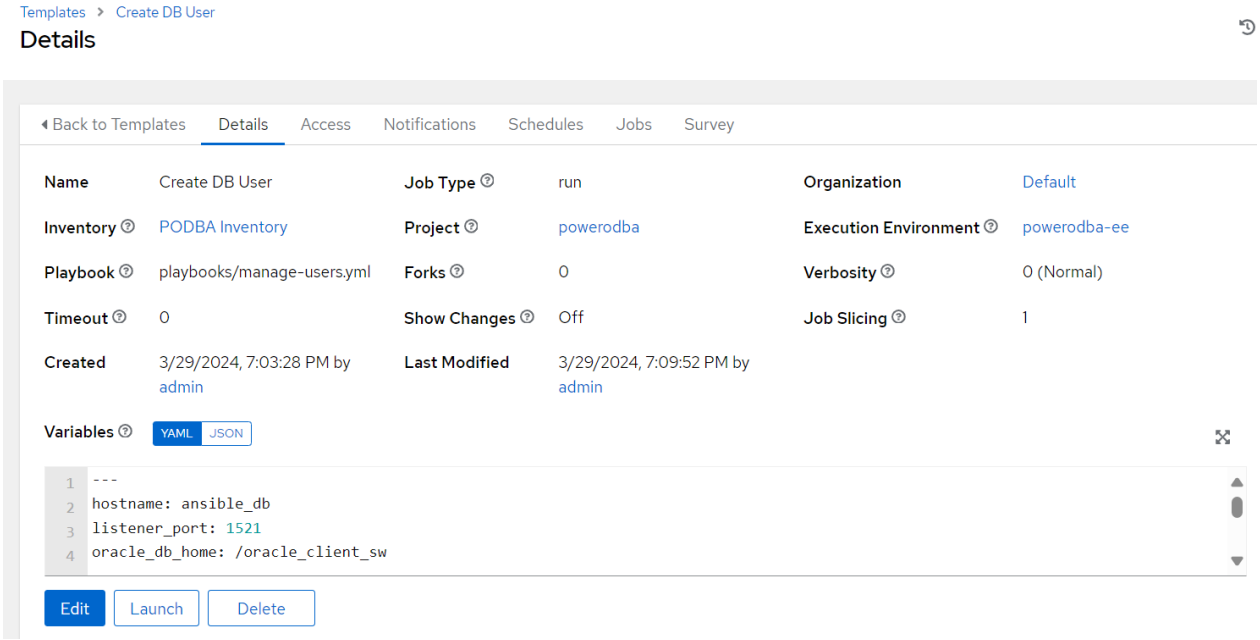
Question: Enter something to type the password, this will be prompted during runtime.
Answer variable name: Update “default_dbpass”, this is explicitly set inside the roles.
Answer type: Password.

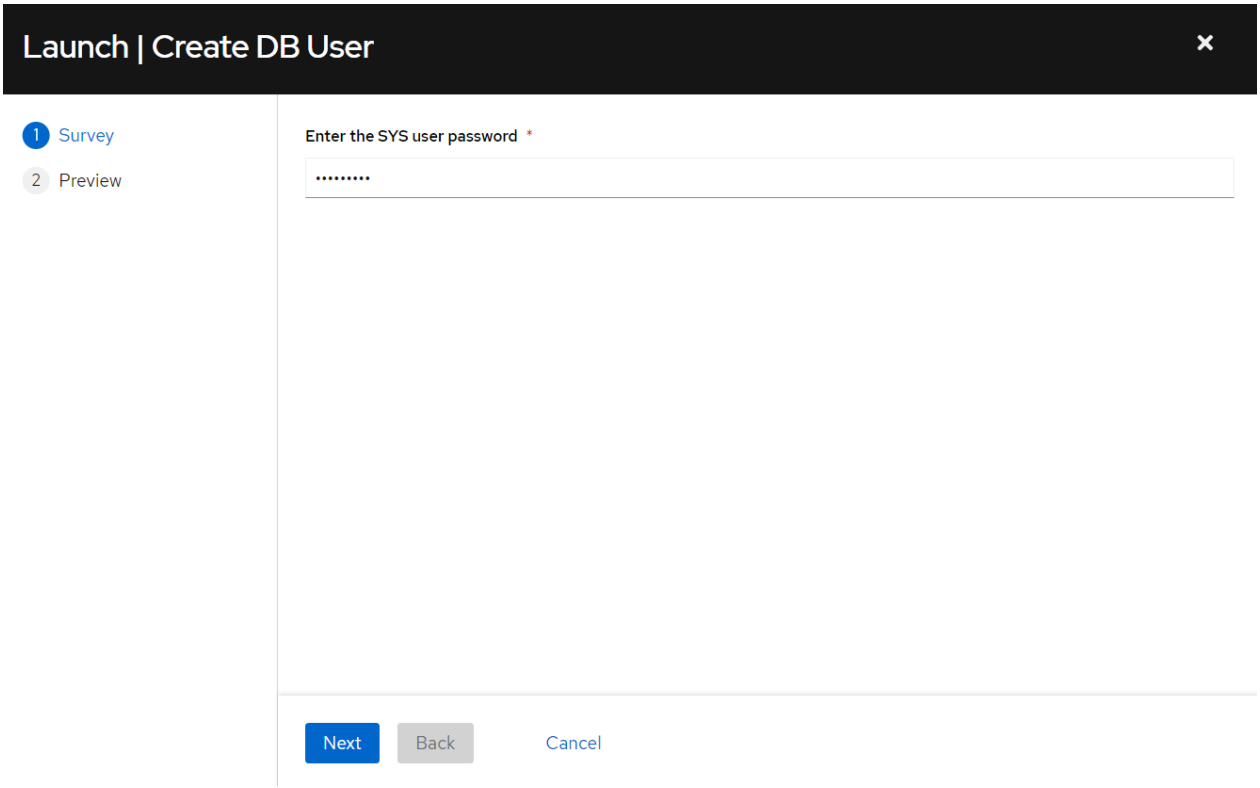
Click on Save.

Make sure to toggle “Survey Enabled”.

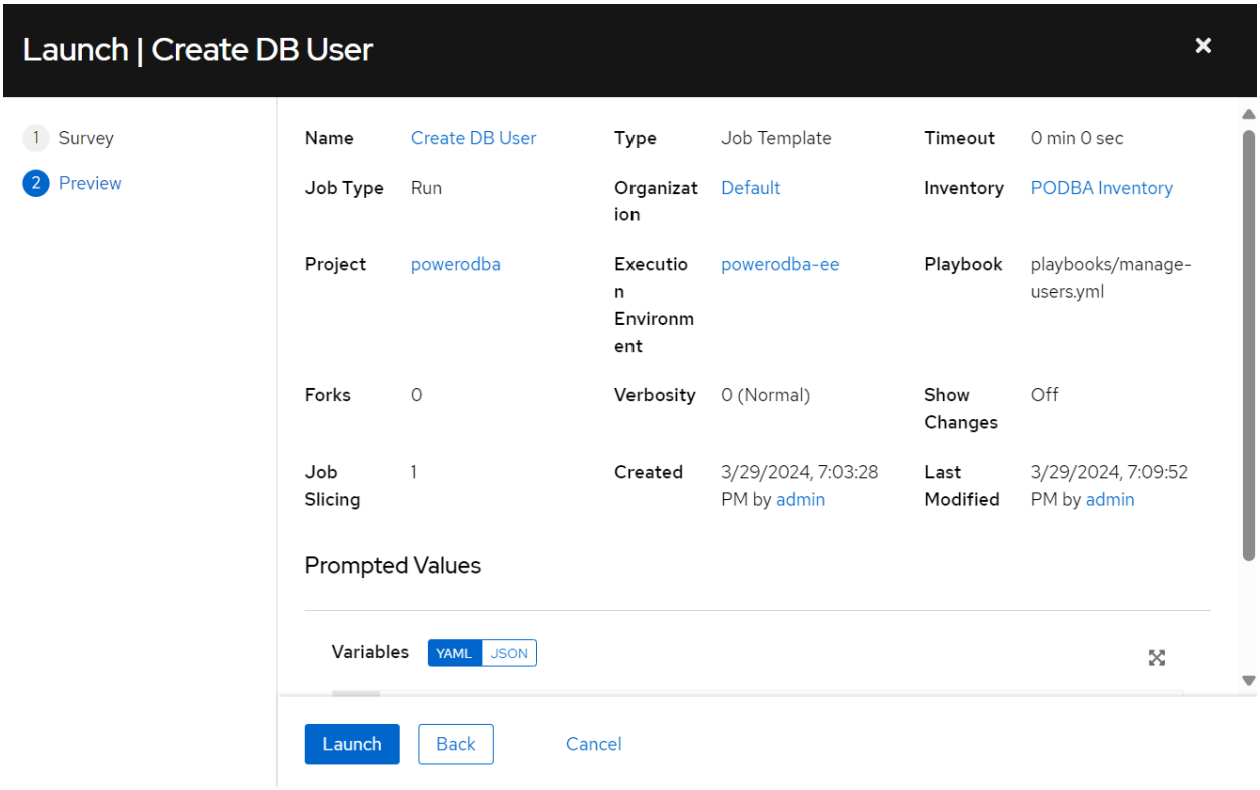


Launch the “Create Users” template.

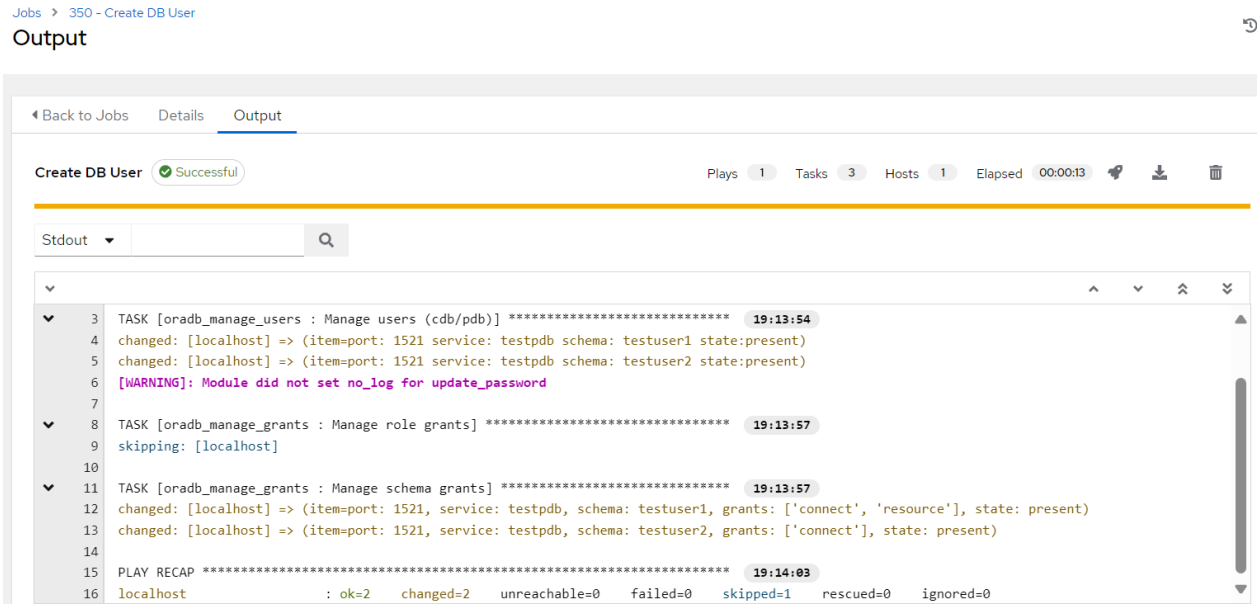




Input the SYS user password and click next.



Click on Launch



User creation and grants successful.