



D02

**Modern Db2 job automation
- maintenance with Ansible -
Markus Fraune, ITGAIN**

IDUG

2021 EMEA Db2 Tech Conference

ITGAIN ✦

Edinburgh, Scotland | October 17 – 21, 2021

Agenda

- Ansible and AWX basics
- Getting Db2 into the game
- Backup Sample
- Pros and Cons / where to use

Infrastructure as Code (IaC) (1|2)

- Manage and provision systems through machine-readable definition files with automatic pipelines, rather than physical hardware configuration or interactive tools
- For virtual AND bare-metal servers including configuration resources
- Mostly stored in a version control system (svn, gitlab, github, git...)
- declarative definitions and scripts, rather than manual processes
- Two approaches: declarative (functional) vs. imperative (procedural)

Infrastructure as Code (IaC) (2|2)

Tool	Released by	Method	Approach	Written in	Comments
Chef	Chef (2009)	Pull	Declarative and imperative	Ruby	
Otter	Inedo	Push	Declarative and imperative	-	Windows oriented
Puppet	Puppet (2005)	Pull	Declarative and imperative	C++, Clojure, Ruby	
SaltStack	SaltStack	Push and Pull	Declarative and imperative	Python	
CFEngine	Northern.tech	Pull	Declarative	C	
Terraform	HashiCorp (2014)	Push	Declarative	Go	
Ansible / Ansible Tower	Rad Hat (2012)	Push	Declarative and imperative	Python	

Configuration as Code (CaC)

- Sometimes used as a synonym for IaC or as a part of it (they overlap at some point)
- “only” for application configuration data
- Separate from IaC to be able to deploy config without new software (has its own pipelines)
- Uses its own repositories and branches
- Goal is to keep it simple and make configuration resistant going from development to production
- Tools (example): Ansible, Consul, Terraform, Vagrant, Packer, most IaC Tools

History of Ansible

- 2012: Michael DeHaan started developing Ansible tool
- 2013: Company Ansible Inc. (originally AnsibleWorks, Inc.) was founded by Michael DeHaan, Timothy Gerla and Saïd Ziouani
- 2015: Red Hat acquired Ansible
- 2016: First AnsibleFest (annual conference)
- 2017: AWX (Upstream Project for Ansible Tower) got Open Source

Ansible Nodes

- **Control Node**
 - machine where Ansible is installed and command like `ansible` or `ansible-playbook` can be run on (laptop, shared desktop, server)
 - Python installation needed
 - Windows is not supported
 - Can have more than one
- **Managed Node**
 - Systems that will be managed by ansible
 - Mostly called hosts
 - Ansible is not installed on a managed host

Ansible Inventory (1 | 2)



- One (or more) yaml files that contain lists of hosts
- Sometimes called the “hostfile”
- Contains information like IP addresses of managed hosts
- Nodes can be organized in (nested) groups
- Default Groups all and ungrouped (hosts with no other group than all)
- Hosts can and will be in more than one group
- Typically hosts grouped by what (database, webservice), where (local region, datacenter) and when (stages like dev, prod, test)

Ansible Inventory (2|2)

```
all:
  hosts:
    nogroup.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        dbone.example.com:
        dbtwo.example.com:
        dbthree.example.com:
  east:
    hosts:
      foo.example.com:
      dbone.example.com:
      dbtwo.example.com:
  west:
    hosts:
      bar.example.com:
      dbthree.example.com:
```

```
prod:
  children:
    east:
test:
  children:
    west:
```

```
1  all:
2  .. hosts:
3  .... nogroup.example.com:
4  .. children:
5  .... webservers:
6  ..... hosts:
7  ..... foo.example.com:
8  ..... bar.example.com:
9  .... dbservers:
10  ..... hosts:
11  ..... dbone.example.com:
```

Ansible Variables

- Can be placed in almost every stage of ansible
 - variables will be overridden from next “deeper” part (the last listed variable will override all others)
 - Part of that order:
 - Role Defaults
 - Inventory
 - Group Vars
 - Host Vars
 - Playbook Vars
 - Role Vars
 - Extra vars (when running ansible or ansible-playbook)
- (full list on next slide (hidden))

Ansible Inventory Vars

atlanta:

hosts:

host1:

db2_port: 50001

db2_max_connections: 100

host2:

db2_port: 50002

db2_max_connections: 250

Ansible Group Vars (inventory file)

atlanta:

hosts:

host1:

host2:

vars:

db2_port: 50003

db2_max_connections: 500

Ansible Group Vars (<group>.yml)

All.yml

db2_port: 50000

db2_max_connections: 100

Atlanta.yml

db2_port: 50003

db2_max_connections: 500

Ansible Host Vars (dbone.example.com.yml)

db2_svcename: 50000

db2_max_connections: 100

db2_database_name: SAMPLE

db2_admin_user: db2adm1

db2_day_of_backup: 1

db2_hour_of_backup: 18

Ansible Playbook

- Repeatable and re-usable "plays" to execute specific tasks to a set of hosts
- Can be used for simple configuration management or complex environment orchestration
- Rule of thumb: if a task is going to be executed more than once it will fit perfectly in a playbook
- Can include other playbooks and/or roles
- No difference in result if executed for the first or 10time: result must be the same (Idempotency)

Ansible Modules

- Discrete unit of code that can be executed from cli or within a playbook/role task
- Gets executed on the managed node, return values will be collected
- Sometimes referred as „task plugins“ or „library plugins“
- Commonly used modules:
 - command
 - shell
 - service
 - copy
 - Template

https://docs.ansible.com/ansible/2.9/modules/modules_by_category.html

Ansible Modules Examples

- name: restart webserver

service:

name: httpd

state: restarted

- name: reboot the servers

command: /sbin/reboot -t now

Ansible Plugins

- Discrete pieces of code that extend Ansible's core functionality
- Are usually executed on the control node
- Used for
 - Connecting to an inventory
 - Logging output
 - Switch to another user (become)

Ansible Roles

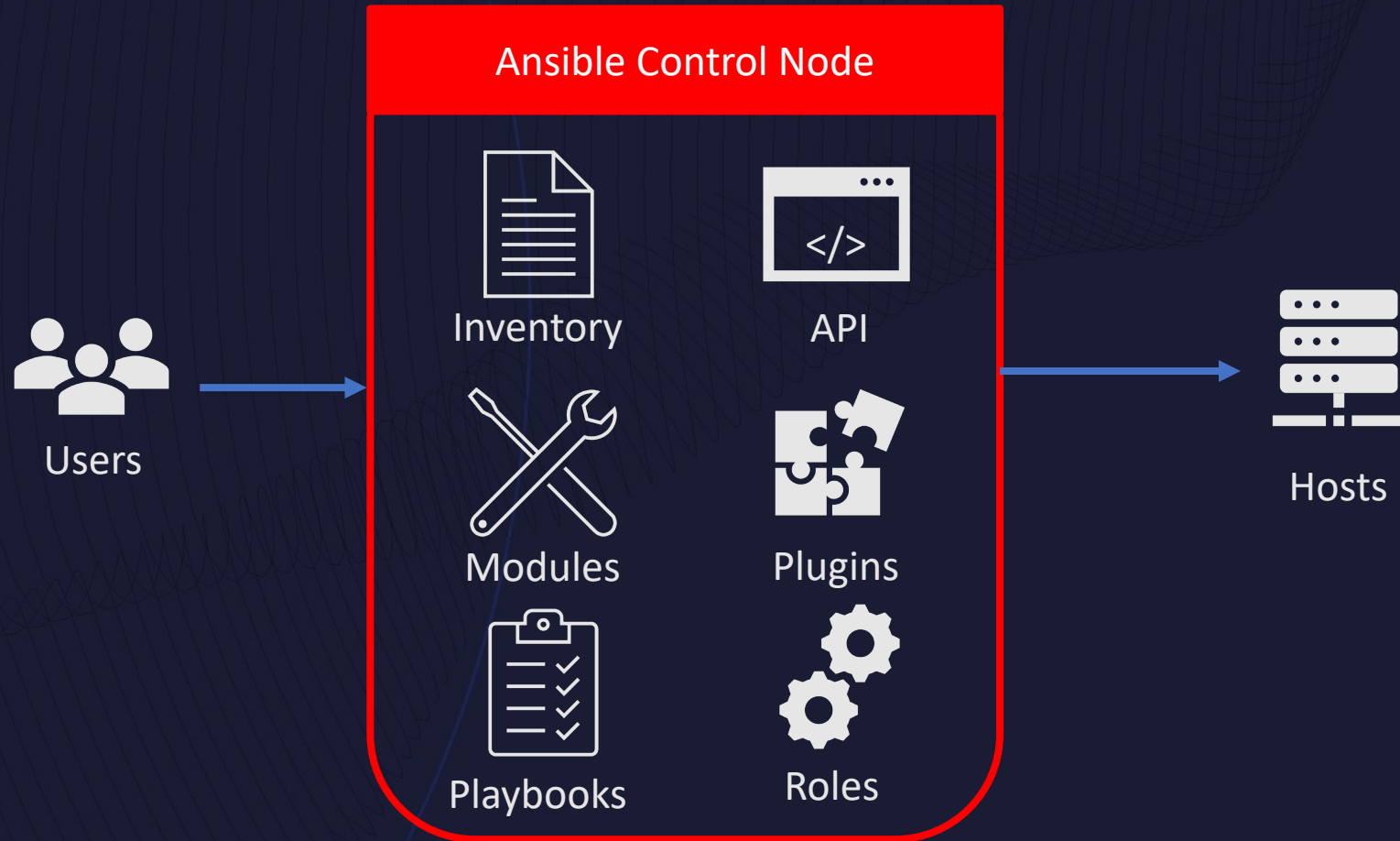
- Group a set of tasks, vars, template, files etc. to a role
- Has a known file structure
- Easily re-usable and share-able
- Can't be executed without a play
- Examples for roles:
 - webservers
 - db2server
 - fileservers
 - db2_backup_scripts
- A play for a given set of hosts can include several roles

Ansible Roles Directory Structure

```
# playbooks
site.yml
webservers.yml
fooservers.yml
roles/
  common/
    tasks/
    handlers/
    library/
    files/
    templates/
    vars/
    defaults/
    meta/
  webservers/
    tasks/
    defaults/
    meta/
```

- Minimum is one directory per role
- Not used directories can be omitted
- Within every directory a main.yml will be called per default
- Tasks/main.yml is the default task file to be executed first (if not explicitly specified)

Ansible Architecture (Basic)



AWX Basics

- Open Source Community Project (The AWX Project)
<https://github.com/ansible/awx/>
- Upstream for Ansible Tower (Enterprise Offering by Red Hat)
- Projects
- Job Templates
- Job Scheduler
- Workflows

AWX Dashboard

The screenshot displays the AWX Dashboard interface. At the top left is the AWX logo. The top right corner contains user information (awx), a notification bell with 0 alerts, and other utility icons. A left-hand navigation menu lists various sections: VIEWS (Dashboard, Jobs, Schedules, My View), RESOURCES (Templates, Credentials, Projects, Inventories, Inventory Scripts), ACCESS (Organizations, Users, Teams), and ADMINISTRATION (Credential Types, Notifications, Management Jobs, Instance Groups, Applications, Settings).

The main dashboard area features a 'DASHBOARD' header and a row of six summary cards:

- 4 HOSTS
- 2 FAILED HOSTS
- 2 INVENTORIES
- 0 INVENTORY SYNC FAILURES
- 2 PROJECTS
- 0 PROJECT SYNC FAILURES

Below these cards is a 'JOB STATUS' section with a line chart. The chart shows the number of jobs over time from July 24 to August 24. The y-axis is labeled 'JOBS' and ranges from 0 to 3. The x-axis is labeled 'TIME' and shows dates. A red line represents the total number of jobs, which remains at 0 until August 22, then rises to 3 on August 24. A green line represents a subset of jobs, which also remains at 0 until August 22, then rises to 1 on August 24. The chart includes filters for PERIOD (PAST MONTH), JOB TYPE (ALL), and VIEW (ALL).

Below the chart are two sections: 'RECENTLY USED TEMPLATES' and 'RECENT JOB RUNS'. Both sections have a 'VIEW ALL' link.

RECENTLY USED TEMPLATES

NAME	ACTIVITY	ACTIONS
Role Db2 Sample SQL		
Sample Playbook		
Role Db2 Backup		

RECENT JOB RUNS

NAME	TIME
Role Db2 Sample SQL	24.8.2021 15:34:58
Role Db2 Sample SQL	24.8.2021 15:31:40
Sample Playbook	24.8.2021 15:30:37
Role Db2 Backup	18.3.2021 02:17:04
Role Db2 Backup	18.3.2021 02:14:00

AWX Job Templates

Role Db2 Backup

DETAILS | PERMISSIONS | NOTIFICATIONS | COMPLETED JOBS | SCHEDULES | ADD SURVEY

* NAME: Role Db2 Backup

DESCRIPTION:

* JOB TYPE: Run PROMPT ON LAUNCH

* INVENTORY: AWX Db2 Testing Inventory PROMPT ON LAUNCH

* PROJECT: AWX Db2 Testing

* PLAYBOOK: role_db2_backup.yml

CREDENTIALS: ansible user PROMPT ON LAUNCH

FORKS: 0

LIMIT: db2_servers PROMPT ON LAUNCH

* VERBOSITY: 0 (Normal) PROMPT ON LAUNCH

JOB TAGS: PROMPT ON LAUNCH

SKIP TAGS: PROMPT ON LAUNCH

LABELS:

INSTANCE GROUPS:

JOB SLICING: 1

TIMEOUT: 0

SHOW CHANGES: PROMPT ON LAUNCH

OPTIONS:
 ENABLE PRIVILEGE ESCALATION
 ENABLE PROVISIONING CALLBACKS
 ENABLE WEBHOOK
 ENABLE CONCURRENT JOBS
 ENABLE FACT CACHE

EXTRA VARIABLES: YAML JSON PROMPT ON LAUNCH

```
1 ---
```

LAUNCH | CANCEL | SAVE


AWX Job Templates - Survey


Role Db2 Backup | SURVEY

EDIT SURVEY PROMPT

* PROMPT

DESCRIPTION

* ANSWER VARIABLE NAME 

* ANSWER TYPE 



MINIMUM MAXIMUM

DEFAULT ANSWER

REQUIRED


PREVIEW

* AFTER HOW MANY HOURS A NEW BACKUP SHOULD BE STARTED?
If last backup is older than x a new one would be started, else not

1  

ROLE DB2 BACKUP

* AFTER HOW MANY HOURS A NEW BACKUP SHOULD BE STARTED?
If last backup is older than x a new one would be started, else not

1 

AWX Job Scheduler - Add Schedule

Backup Schedule - Every 2 Hours

* NAME: Backup Schedule - Every 2 Hours

* START DATE: 25.8.2021

* START TIME (HH24:MM:SS): 16:00:00

* LOCAL TIME ZONE: Europe/Berlin

* REPEAT FREQUENCY: Hour

FREQUENCY DETAILS

* EVERY: 2 HOURS

* END: Never

SCHEDULE DESCRIPTION

every 2 hours

OCCURRENCES (Limited to first 10) DATE FORMAT LOCAL TIME ZONE UTC

- 08-25-2021 16:00:00
- 08-25-2021 18:00:00
- 08-25-2021 20:00:00
- 08-25-2021 22:00:00
- 08-26-2021 00:00:00
- 08-26-2021 02:00:00
- 08-26-2021 04:00:00
- 08-26-2021 06:00:00
- 08-26-2021 08:00:00
- 08-26-2021 10:00:00

PROMPT CANCEL SAVE

AWX Job Scheduler - things to know

- „No“ limit in number of schedules for template
- Schedule sticks to a template, can't start more than one template
- A schedule starts the template with its predefined settings, no prompt or extra variables can be specified
- Mostly you create a template “for” a schedule, like a template to backup database on Monday with a schedule that starts this template on Mondays
- Jobs may start later than given times of the schedule because of load of AWX System

AWX Workflows

WORKFLOW VISUALIZER | Demo Workflow

TOTAL NODES 4

```
graph LR; START[START] --> RoleDb2SampleSQL[Role Db2 Sample SQL]; RoleDb2SampleSQL --> RoleDb2Backup[Role Db2 Backup]; RoleDb2SampleSQL --> SamplePlaybook[Sample Playbook];
```

The diagram illustrates a workflow with four nodes. It begins with a green 'START' node. A blue arrow points to a 'Role Db2 Sample SQL' node, which has a small 'IT' icon. From this node, two arrows branch out: a green one to 'Role Db2 Backup' and a red one to 'Sample Playbook', both also featuring 'IT' icons. A dashed box is positioned below the 'Role Db2 Sample SQL' node.

ADD A NODE

Template

SEARCH [] Q KEY

- Demo Job Template ?
- Demo Workflow Workflow ?
- Role Db2 Backup ?
- Role Db2 Sample SQL ?
- Sample Playbook ?

ITEMS 1 - 5

* RUN
Always

* CONVERGENCE
Any

The inventory of this node will not be overridden by the parent workflow inventory.

CANCEL SELECT

CLOSE SAVE

Ansible and Db2 - What do I need?

- Ansible Development Environment
 - System / Container running Ansible (Control Node)
 - System / Container to test your playbooks on (like db2 container, “empty” Alpine Linux container) -> Can be role / playbooks as well
- Repository to store and version your files (e.g. gitlab, github...)
- Running AWX K8S Environment (e.g. with minikube in a VM)
- SSH Access from four Control Node and AWX system to Managed Nodes
- Code Editor with a linter (e.g. Visual Studio Code)

Ansible and Db2 - basic host file (db2srv1.mydomain.org)

```
db2_app_user: db2_tech_u
db2_inst_user: db2inst1
db2_inst_con: server_encrypt
db2_dbname: proddb
db2_db_path: "/db2/{{ db2_inst_user }}/{{ db2_db_name }}/dbpath"
db2_data_path: "/db2/{{ db2_inst_user }}/{{ db2_db_name }}/data"
db2_db_pagesize: 32
```

Ansible and Db2 - Server Provisioning

Add db2 app users to System and add software dependencies

- name: Add user '{{ db2_app_user }}' with a bash, groups 'admins' and 'developers'
ansible.builtin.user:
 - name: "{{ db2_app_user }}"
 - home: "/db2/homes/{{ db2_app_user }}"
 - shell: /bin/bash
 - groups: admins,developers
 - append: yes
- name: Upgrade all packages on the Server
apt: upgrade=yes
- name: Install the latest versions of each component
apt:
 - name:
 - gcc
 - python
 - state: latest

Ansible and Db2 - Db2 Provisioning (1|3)

- Bring db2 software package onto the system

- name: Download db2_install_package

```
get_url:
```

```
url: http://software_repo.mydomain.org/db2/luw/11.5.6/db2_install_package.tar.gz
```

```
dest: /tmp/db2_install_package.tar.gz
```

- Unarchive db2 software package

- name: Unarchive a file that is already on the remote machine

```
ansible.builtin.unarchive:
```

```
src: /tmp/db2_install_package.tar.gz
```

```
dest: /tmp/db2_install_package
```

```
remote_src: yes
```


Ansible and Db2 - Db2 Provisioning (2|3)

- Install Db2

- name: Copy Response File

```
ansible.builtin.template:
```

```
  src: response.file.template
```

```
  dest: /tmp/response.file
```

- name: Install Db2 Server

```
command: /tmp/db2_install_package/install/db2setup -r /tmp/response.file
```

- Create Db2 Instance

- name: Create Db2 instance {{db2_inst_user }}

```
command: /opt/.../instance/db2icrt -a {{ db2_inst_con }} -u db2fenc1 {{db2_inst_user }}
```

Ansible and Db2 - Db2 Provisioning (3|3)

- Create Database

```
- name: create database {{ db2_dbname }} in instance {{ db2_inst_user }}
  become: true
  become_method: su
  become_user: "{{ db2_inst_user }}"
  shell: bash -lci 'db2 create database {{ db2_dbname }} on {{ db2_data_path }} dbpath on
{{ db2_db_path }} page size {{ db2_db_pagesize }}k '
```

- Grant App User

```
- name: grant App user
  become: true
  become_method: su
  become_user: "{{ db2_inst_user }}"
  shell: bash -lci 'db2 +o connect to {{ db2_dbname }} && db2 "grant connect, createtab,
dataaccess on database to user {{ db2_app_user }}"'
```

Ansible and Db2 - Role Sample SQL (1 | 4)

```
# main.yml - get all local installations
- name: get local db2 installations
  shell: db2ls |awk '/V/ {print $1}'
  register: db2_install_path_list
  changed_when: "db2_install_path_list.rc < 0"
- name: debug output
  debug:
    msg: "myscript has error: {{ db2_install_path_list.stderr }}"
  when: db2_install_path_list.stderr | length > 0
  failed_when: db2_install_path_list.stderr | length > 0
- name: include db2_get_instances
  include_tasks: db2_get_instances.yml
  with_items: "{{ db2_install_path_list.stdout_lines }}"
```

▼ ROLE_DB2_SAMPLE_SQL

▼ tasks

- ! db2_get_databases.yml
- ! db2_get_instances.yml
- ! db2_sample_select.yml
- ! main.yml

Ansible and Db2 - Role Sample SQL (2 | 4)

```
# db2_get_instances.yml
- name: get local instances for path {{ item }}
  shell: "{{ item }}/instance/db2ilist"
  register: db2_instance_list
  changed_when: "db2_instance_list.rc < 0"
- name: debug output
  debug:
    msg: "myscript has error: {{ db2_instance_list.stderr }}"
  when: db2_instance_list.stderr | length > 0
  failed_when: db2_instance_list.stderr | length > 0
- name: include db2_get_databases
  include_tasks: db2_get_databases.yml
  with_items: "{{ db2_instance_list.stdout_lines }}"
```

```
▼ ROLE_DB2_SAMPLE_SQL
  ▼ tasks
    ! db2_get_databases.yml
    ! db2_get_instances.yml
    ! db2_sample_select.yml
    ! main.yml
```


Ansible and Db2 - Role Sample SQL (3 | 4)

```
# db2_get_databases.yml
- name: set fact db2 instance vara
  ansible.builtin.set_fact:
    db2_instance: "{{ item }}"
- name: get local databases for instance {{ item }}
  become: true
  become_method: su
  become_user: "{{ db2_instance }}"
  shell: bash -lci 'db2 list db directory|awk -v RS= '/Indirect/' grep "Database name"|cut -d "=" -f 2|sort|uniq'
  register: db2_database_list
  changed_when: "db2_database_list.rc < 0"
- name: debug output
  debug:
    msg: "myscript has error: {{ db2_database_list.stderr }}"
    when: db2_database_list.rc > 0
    failed_when: db2_database_list.rc > 0
- name: include db2_sample_select
  include_tasks: db2_sample_select.yml
  with_items: "{{ db2_database_list.stdout_lines }}"
```

▼ ROLE_DB2_SAMPLE_SQL

▼ tasks

- ! db2_get_databases.yml
- ! db2_get_instances.yml
- ! db2_sample_select.yml
- ! main.yml

Ansible and Db2 - Role Sample SQL (4 | 4)

```
# db2_sample_select.yml
- name: sample select for database {{ item }} at instance {{ db2_instance }}
  become: true
  become_method: su
  become_user: "{{ db2_instance }}"
  shell: bash -lci 'db2 +o connect to {{ item }} && db2 -v "select * from
syscat.tables"'
  register: sql_output
  changed_when: "sql_output.rc < 0"
  failed_when: sql_output.rc > 4
- name: debug output
  debug:
    msg: "mysql has a non zero rc ({{ sql_output.rc }}): {{ sql_output.stdout }}"
  when: sql_output.rc > 0
  failed_when: sql_output.rc > 0
```

▼ ROLE_DB2_SAMPLE_SQL

▼ tasks

- ! db2_get_databases.yml
- ! db2_get_instances.yml
- ! db2_sample_select.yml
- ! main.yml

Ansible and Db2 - Role Backup (1 | 5)

- In comparison to sample sql role:
 - Added defaults/main.yml for default variables
 - Include in “db2_get_databases.yml” is now db2_backup and not “db2_sample_select”

```
# defaults/main.yml  
backup_every_hours: 1
```

▼ ROLE_DB2_BACKUP

▼ defaults

! main.yml

▼ tasks

! db2_backup.yml

! db2_get_databases.yml

! db2_get_instances.yml

! main.yml

Ansible and Db2 - Role Backup (2|5)

```
- name: check if database is active
  become: true
  become_method: su
  become_user: "{{ db2_instance }}"
  shell: bash -lci 'db2 list active databases | grep -i {{
item }}'
  register: list_active_db_output
  changed_when: "list_active_db_output.rc < 0"
  failed_when: "list_active_db_output.rc > 3"
- name: debug list active db output
  debug:
    msg: "List Active datatabases showed {{ item }} as active
database, no offline backup possible"
  when: "list_active_db_output.rc == 0"
```

▼ ROLE_DB2_BACKUP

▼ defaults

! main.yml

▼ tasks

! db2_backup.yml

! db2_get_databases.yml

! db2_get_instances.yml

! main.yml

Ansible and Db2 - Role Backup (3 | 5)

```
- name: check if last db backup timestamp is older than {{ backup_every_hours }}
  hours
  become: true
  become_method: su
  become_user: "{{ db2_instance }}"
  shell: bash -lci 'db2 +o connect to {{ item }} && db2 -x "select
timestamp(LAST_BACKUP) as last_backup from table(mon_get_database(-1)) as t where
LAST_BACKUP >= current timestamp - {{ backup_every_hours }} hours"'
  register: backup_old_enough
  changed_when: "backup_old_enough.rc < 0"
  failed_when: "backup_old_enough.rc > 3"
  when: "list_active_db_output.rc == 1"
```

```
- name: debug check last backup ts
  debug:
    msg: "Last Backup TS of {{ item }} is not old enough - {{
backup_old_enough.stdout }}"
  when: backup_old_enough is not skipped and "backup_old_enough.rc == 0"
```

▼ ROLE_DB2_BACKUP

▼ defaults

! main.yml

▼ tasks

! db2_backup.yml

! db2_get_databases.yml

! db2_get_instances.yml

! main.yml

Ansible and Db2 - Role Backup (4 | 5)

```
- name: block backup database {{ item }} at instance {{ db2_instance }}
  become: true
  become_method: su
  become_user: "{{ db2_instance }}"
  block:
    - name: try to backup {{ item }} at instance {{ db2_instance }}
      shell: bash -lci 'db2 -v backup db {{ item }} to /dev/null'
      register: backup_output
      changed_when: "backup_output.rc == 0"
      failed_when: "backup_output.rc > 3"
      when: "list_active_db_output.rc == 1 and backup_old_enough.rc == 1"
      ignore_errors: yes
  rescue:
    - name: print out backup stdout
      debug:
        msg: "backup has a non zero rc ({{ backup_output.rc }}): {{ backup_output.stdout }}"
```

▼ ROLE_DB2_BACKUP

▼ defaults

! main.yml

▼ tasks

! db2_backup.yml

! db2_get_databases.yml

! db2_get_instances.yml

! main.yml

Ansible and Db2 - Role Backup (5 | 5)

```
- name: print backup output
  debug:
    msg: "backup was successfull. output: {{ backup_output.stdout }}"
  when: backup_output is succeeded and backup_output is not skipped
```

▼ ROLE_DB2_BACKUP

▼ defaults

! main.yml

▼ tasks

! db2_backup.yml

! db2_get_databases.yml

! db2_get_instances.yml

! main.yml

Ansible and Db2 - Backup Template (1 | 4)

JOBS / 273 - Role Db2 Backup

DETAILS

STATUS ● Running

STARTED 24.8.2021 17:50:38

FINISHED Not Finished

JOB TEMPLATE [Role Db2 Backup](#)

JOB TYPE Run

LAUNCHED BY [awx](#)

INVENTORY [AWX Db2 Testing Inventory](#)

PROJECT [AWX Db2 Testing](#)

REVISION 861f858

PLAYBOOK [role_db2_backup.yml](#)

CREDENTIAL [ansible user](#)

LIMIT db2_servers

ENVIRONMENT /var/lib/awx/venv/ansible

EXECUTION NODE awx-5ffbfd489c-lskm8

INSTANCE GROUP [tower](#)

EXTRA VARIABLES [YAML](#) [JSON](#) [EXPAND](#)

```
1 backup_every_hours: 1
2
```

Role Db2 Backup

PLAYS TASKS HOSTS ELAPSED 00:00:35

JOB IS STILL RUNNING

```
1 Enter passphrase for /tmp/awx_273__15zqbay/artifacts/273/ssh_key_data:
2 Identity added: /tmp/awx_273__15zqbay/artifacts/273/ssh_key_data (ansible@localhost.localdomain)
3
4 PLAY [all] ***** 17:50:42
5
6 TASK [Gathering Facts] ***** 17:50:42
7 ok: [docker_db2_test1]
8 ok: [docker_db2_test2]
9
10 TASK [db2 backup role] ***** 17:50:52
11
12 TASK [role_db2_backup : get local db2 installations] ***** 17:50:52
13 ok: [docker_db2_test1]
14 ok: [docker_db2_test2]
15
16 TASK [role_db2_backup : debug output] ***** 17:50:55
17 skipping: [docker_db2_test1]
18 skipping: [docker_db2_test2]
19
20 TASK [role_db2_backup : include db2_get_instances] ***** 17:50:56
21 included: /tmp/awx_273__15zqbay/project/roles/role_db2_backup/tasks/db2_get_instances.yml for docker_db2_test1
22 , docker_db2_test2
23
24 TASK [role_db2_backup : get local instances for path /opt/ibm/db2/V11.5] ***** 17:50:57
25 ok: [docker_db2_test2]
26 ok: [docker_db2_test1]
27
28 TASK [role_db2_backup : debug output] ***** 17:51:00
29 skipping: [docker_db2_test1]
30 skipping: [docker_db2_test2]
31
31 TASK [role_db2_backup : include db2_get_databases] ***** 17:51:00
```


Ansible and Db2 - Backup Template (2|4)

```
SEARCH Q KEY
```

```
- ^ v ^ v
```

```
68 TASK [role_db2_backup : debug list active db output] ***** 17:51:08
```

```
69 skipping: [docker_db2_test1]
```

```
70
```

```
71 TASK [role_db2_backup : check if last db backup timestamp is older than 1 hours] *** 17:51:08
```

```
72 ok: [docker_db2_test1]
```

```
73
```

```
74 TASK [role_db2_backup : debug check last backup ts] ***** 17:51:13
```

```
75 ok: [docker_db2_test1] => {
```

```
76   "msg": "Last Backup TS of DB1 is not old enough - "
```

```
77 }
```

```
78
```

```
79 TASK [role_db2_backup : sleep 5 sec for the database to deactivate] ***** 17:51:13
```

```
80 ok: [docker_db2_test1]
```

```
81
```

```
82 TASK [role_db2_backup : try to backup DB1 at instance db2inst1] ***** 17:51:20
```

```
83 changed: [docker_db2_test1]
```

```
84
```

```
85 TASK [role_db2_backup : print backup output] ***** 17:51:28
```

```
86 ok: [docker_db2_test1] => {
```

```
87   "msg": "backup was successfull. output: backup db DB1 to /dev/null\n\nBackup successful. The timestamp for
```

```
88   this backup image is : 20210824155124 "
```

```
89 }
```

```
90
```

```
90 TASK [role_db2_backup : check if database is active] ***** 17:51:28
```

```
91 ok: [docker_db2_test1]
```

```
92
```

```
93 TASK [role_db2_backup : debug list active db output] ***** 17:51:31
```

```
94 skipping: [docker_db2_test1]
```

```
95
```

```
96 TASK [role_db2_backup : check if last db backup timestamp is older than 1 hours] *** 17:51:31
```

```
97 ok: [docker_db2_test1]
```

```
98
```

Ansible and Db2 - Backup Template (3 | 4)

```
SEARCH Q KEY
```

```
141 ok: [docker_db2_test2]
```

```
142
```

```
143 TASK [role_db2_backup : debug list active db output] ***** 17:52:16
```

```
144 skipping: [docker_db2_test2]
```

```
145
```

```
146 TASK [role_db2_backup : check if last db backup timestamp is older than 1 hours] *** 17:52:16
```

```
147 ok: [docker_db2_test2]
```

```
148
```

```
149 TASK [role_db2_backup : debug check last backup ts] ***** 17:52:21
```

```
150 ok: [docker_db2_test2] => {
```

```
151   "msg": "Last Backup TS of TEST2 is not old enough - "
```

```
152 }
```

```
153
```

```
154 TASK [role_db2_backup : sleep 5 sec for the database to deactivate] ***** 17:52:21
```

```
155 ok: [docker_db2_test2]
```

```
156
```

```
157 TASK [role_db2_backup : try to backup TEST2 at instance db2inst1] ***** 17:52:28
```

```
158 changed: [docker_db2_test2]
```

```
159
```

```
160 TASK [role_db2_backup : print backup output] ***** 17:52:36
```

```
161 ok: [docker_db2_test2] => {
```

```
162   "msg": "backup was successfull. output: backup db TEST2 to /dev/null\n\nBackup successful. The timestamp fo
```

```
163   r this backup image is : 20210824155233 "
```

```
164 }
```

```
165 TASK [role_db2_backup : set fact db2 instance vara] ***** 17:52:36
```

```
166 ok: [docker_db2_test1]
```

```
167
```

```
168 TASK [role_db2_backup : get local databases for instance db2inst2] ***** 17:52:36
```

```
169 ok: [docker_db2_test1]
```

```
170
```

```
171 TASK [role_db2_backup : debug output] ***** 17:52:39
```

Ansible and Db2 - Backup Template (4 | 4)

```
Role Db2 Backup
PLAYS 1 TASKS 47 HOSTS 2 ELAPSED 00:02:08
SEARCH Q KEY
-
167
168 TASK [role_db2_backup : get local databases for instance db2inst2] ***** 17:52:36
169 ok: [docker_db2_test1]
170
171 TASK [role_db2_backup : debug output] ***** 17:52:39
172 skipping: [docker_db2_test1]
173
174 TASK [role_db2_backup : include db2_backup.yml] ***** 17:52:39
175 [WARNING]: The loop variable 'item' is already in use. You should set the
176 `loop_var` value in the `loop_control` option for the task to something else to
177 avoid variable collisions and unexpected behavior.
178 included: /tmp/awx_273_15zqbay/project/roles/role_db2_backup/tasks/db2_backup.yml for docker_db2_test1
179
180 TASK [role_db2_backup : check if database is active] ***** 17:52:39
181 ok: [docker_db2_test1]
182
183 TASK [role_db2_backup : debug list active db output] ***** 17:52:43
184 skipping: [docker_db2_test1]
185
186 TASK [role_db2_backup : check if last db backup timestamp is older than 1 hours] *** 17:52:43
187 fatal: [docker_db2_test1]: FAILED! => {"changed": false, "cmd": "bash -lci 'db2 -o connect to INST2DB && db2 -x \
"select timestamp(LAST_BACKUP) as last_backup from table(mon_get_database(-1)) as t where LAST_BACKUP >= current t
imestamp - 1 hours\"'", "delta": "0:00:01.999825", "end": "2021-08-24 15:52:46.672205", "failed_when_result": true
, "msg": "non-zero return code", "rc": 4, "start": "2021-08-24 15:52:44.672380", "stderr": "bash: no job control i
n this shell", "stderr_lines": ["bash: no job control in this shell"], "stdout": "", "stdout_lines": []}
188
189 PLAY RECAP ***** 17:52:46
190 docker_db2_test1 : ok=33 changed=3 unreachable=0 failed=1 skipped=8 rescued=0 igno
red=0
191 docker_db2_test2 : ok=14 changed=1 unreachable=0 failed=0 skipped=4 rescued=0 igno
red=0
192
```


Ansible and Db2 - What to use it for?

- Repeating jobs
- Repeating activities
- If you have more than 1 system you want to run jobs on
- Existing Scripts
 - Deployment / Copy to managed node
 - Execute via schedule on managed node
 - add crontab via role to execute scripts locally
- New Tasks
 - Create plays and roles for new requirements
 - Enhance and reuse existing plays/tasks/roles

Ansible and Db2 - Pros / Cons

- Cons
 - New / Something to learn
 - Slightly higher complexity
- Pros
 - Fast learning curve
 - Existing scripts can be reused
 - Repeatable and checked-in jobs and configuration lowers the risk of a failure
 - Easy to execute on a high number of hosts
 - Shareable scripts
 - Tons of pre-existing roles and plays available online

Session: D02

Speaker: Markus Fraune

Company: ITGAIN

Email Address: markus.fraune@itgain.de

Twitter: [@maggusf](https://twitter.com/maggusf)

Please fill out your session evaluation!