



# Automatically deploy Schema Changes with Ansible and Liquibase

Robert Marz

18. September 2018



swiss oracle  
user group

# Robert Marz

Client

Senior Technical Architect  
with database centric view of the world

its-people

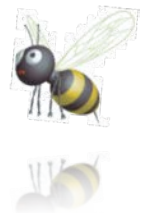
Portfolio Manager Database Technologies  
Blog Editor

DOAG

Active Member Database Community  
in charge of Cloud topics



ORACLE  
ACE



@RobbieDatabee



blog.its-people.de

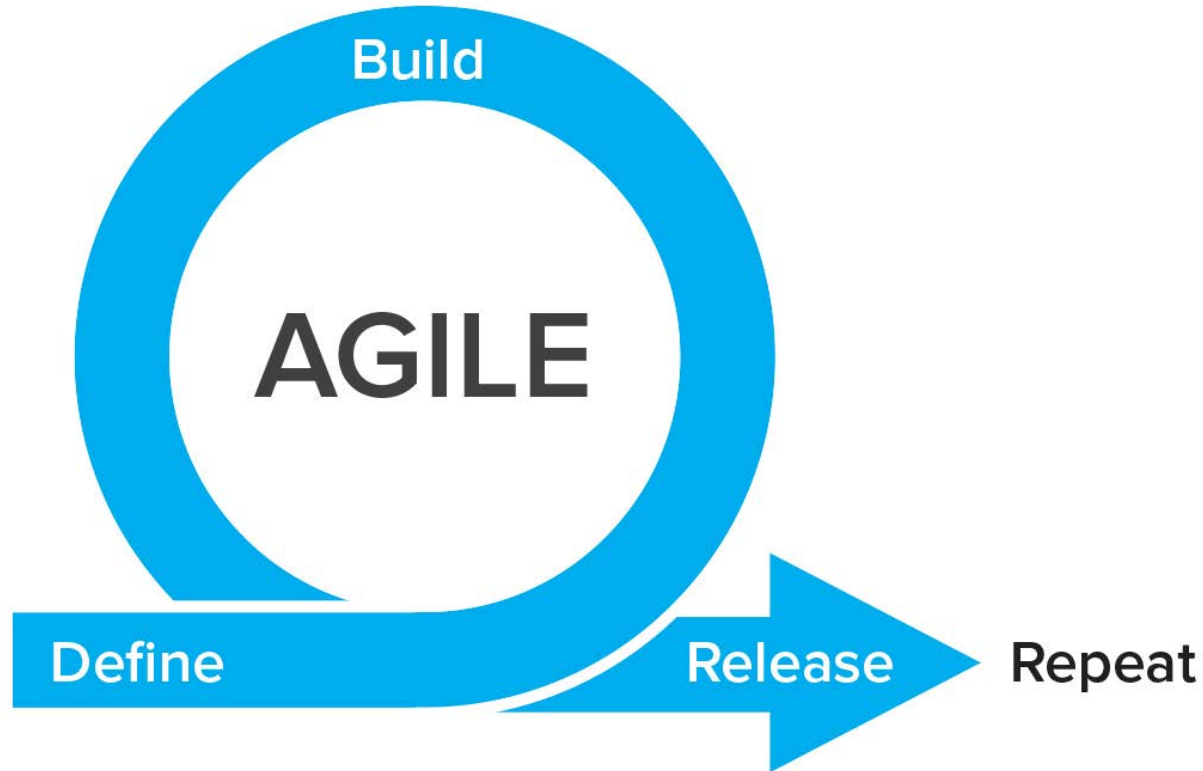


Robert.Marz  
@its-people.de

# Agile and DevOps – Database Challenges



# Agile meets Databases



---

Short Development cycles

**Agile**

---

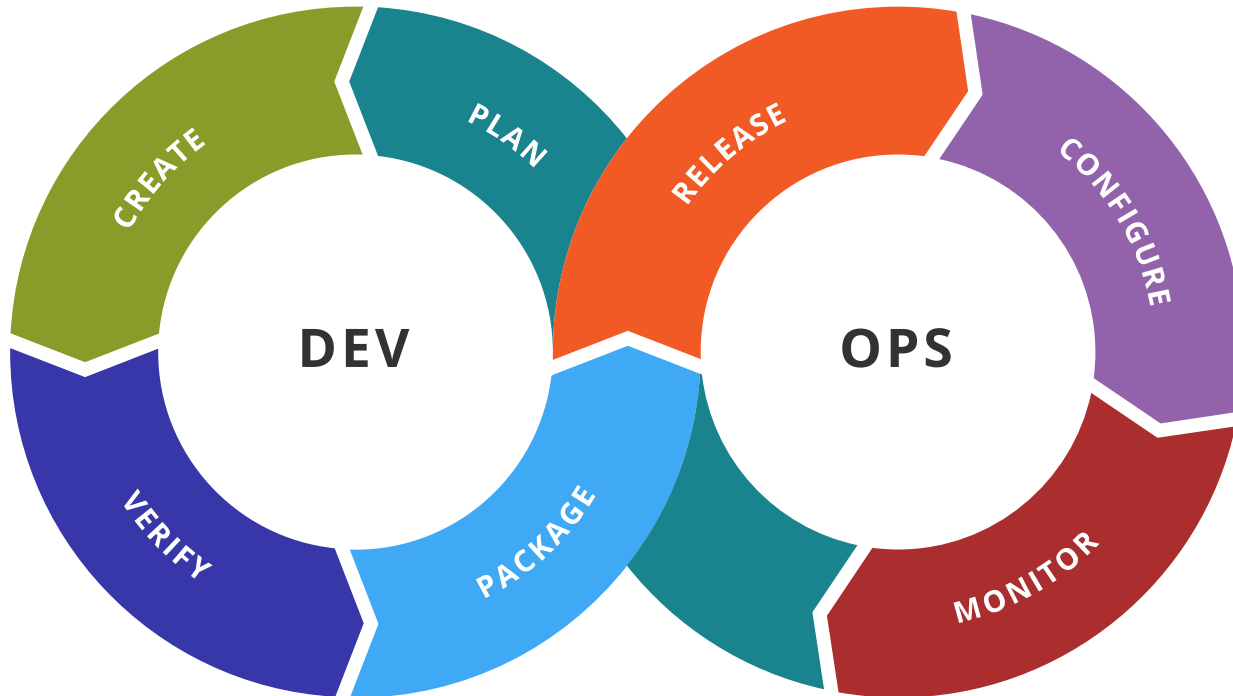
Many Iterations

---

Frequent Schema Changes

---

# DevOps Requirements for Databases



---

Continuous  
Integration

---

Rolling Releases  
Back and Forward

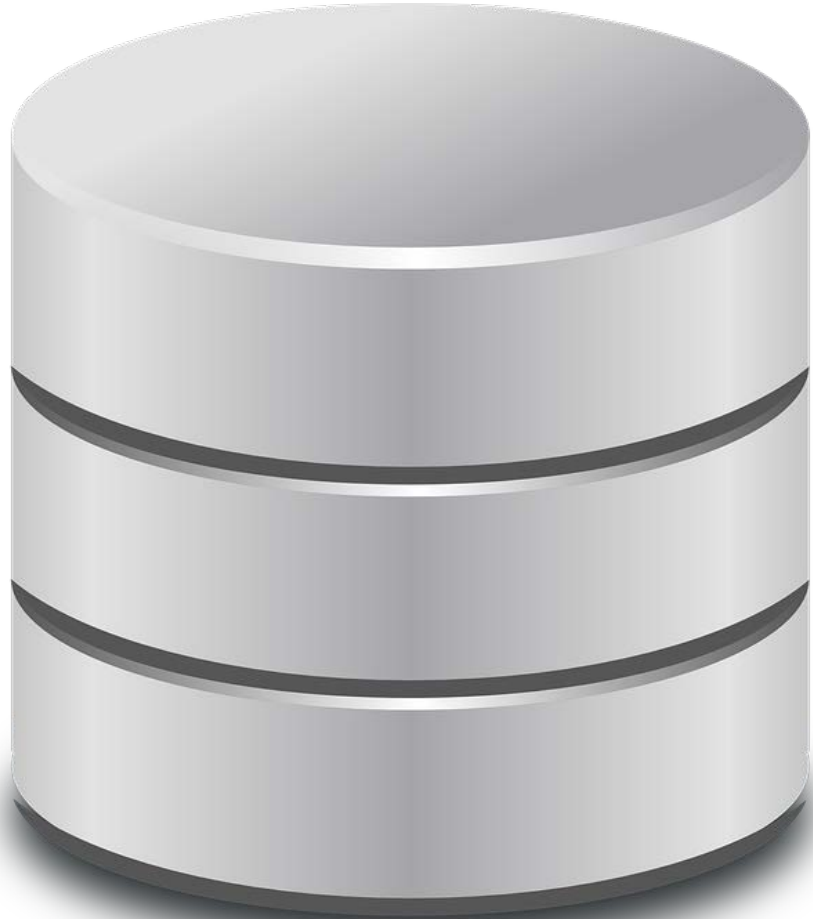
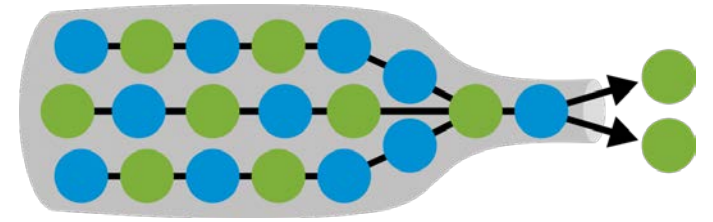
---

Multiple (DB-)  
Servers

---

**DevOps**

# Database Challenges



## Database Challenges

DB  
Schemas

DevOps  
Bottleneck

Scripts

create  
modify  
revert

Existing  
Data

# Liquibase – Overview



## DB Changes

Deploy, Rollback  
Track, Manage, Document  
DDL, Sources, Data

## Supported Databases

Oracle  
MySQL, PostgreSQL, SQL Server, DB2 & more

## Incremental Changes

Change sets / Change logs  
File formats  
YAML, JSON, XML, SQL  
Changelog Tables inside DBs

## Java based Open Source

Commercial Product: Datical

# Ansible in a Nutshell



ANSIBLE

Agentless IT Automation

Open Source

Based on python

by RedHat

Parallel  
execution

Linux / OS X  
via ssh

Windows  
PowerShell Remoting

Ansible  
Playbooks

describe the desired  
state

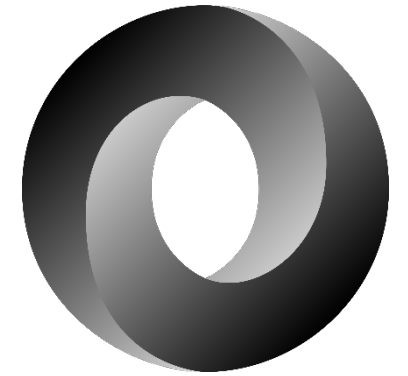
YAML – Human and  
Machine readable

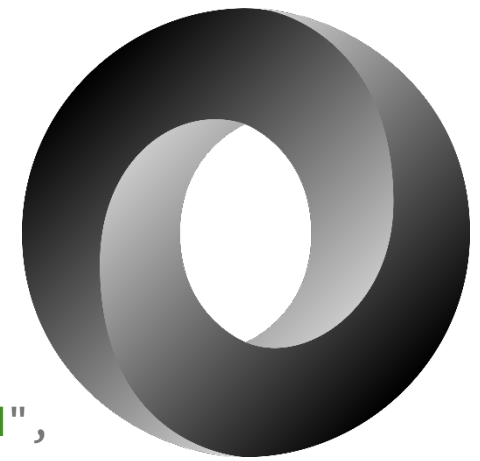
# YAML & JSON: two related file formats

```
databaseChangeLog:
- changeSet:
  id: 2
  author: nvoxland
  changes:
  - addColumn:
    tableName: person
    columns:
    - column:
      name: username
      type: varchar(8)
```

# YAML

```
{ "databaseChangeLog": [
  { "changeSet": {
    "id": "2",
    "author": "nvoxland",
    "changes": [
      { "addColumn": {
        "tableName": "person",
        "columns": [
          { "column": {
            "name": "username",
            "type": "varchar(8)"
          }
        ]
      }
    ]
  }
}
]
}
```





## JSON: Java Script Object Notation

Data: "Key": "Value"

Brackets: {} - Groupings

[] - Arrays

Schemaless

```
{ "databaseChangeLog": [  
  { "changeSet": {  
    "id": "2",  
    "author": "nvoxland",  
    "changes": [  
      { "addColumn": {  
        "tableName": "person",  
        "columns": [  
          { "column": {  
            "name": "username",  
            "type": "varchar(8)"  
          }  
        ]  
      }  
    ]  
  }  
}]  
}
```



YAML:  
YAML Ain't Markup Language

Human readable	structured data format
Data Types	Scalar (Key: Value) Sequences (Arrays) Mappings
Grouping	Whitespace is important Brackets as in JSON
JSON	can be valid YAML

```
databaseChangeLog:
  - changeSet:
      id: 2
      author: nvoxland
      changes:
        - addColumn:
            tableName: person
            columns:
              - column:
                  name: username
                  type: varchar(8)
```

# Ansible – The Basics

Only on Control Machine

Python 2 or 3

Windows not supported

Packages are available

yum install ansible

apt-get install ansible

...

## Installation



ansible

ansible-playbook

ansible-console

REPL for debugging

ansible-vault

Encrypts sensitive Files

...

## Command Line Interface



# Ansible Inventory

Inventory lists the Hosts Ansible can Change

Default location	/etc/ansible/hosts
File formats	ini
	YAML
Grouping	
Static or dynamic	
Variables	per Host
	Per Group
connection types	ssh (default)
	local, docker

```
mail.example.com
jumper ansible_port=5555 ansible_host=192.0.2.50

[webservers]
foo.example.com http_port=80 maxRequestsPerChild=808
bar.example.com http_port=303 maxRequestsPerChild=909
www[01:50].example.com

[dbservers]
one.example.com
two.example.com
three.example.com
db-[a:f].example.com

[dbservers:vars]
ntp_server=ntp.atlanta.example.com
proxy=proxy.atlanta.example.com
halon_system_timeout=30
self_destruct_countdown=60
escape_pods=2

[CoolApp:children]
dbservers
webservers
```



# Ansible Playbooks (1/3)



## Playbook

### Task Sequence

single steps

describe **desired state**

YAML (default)

### Modules

Abstract System tasks

Over 1300 predefined

Write your own

### Variable values

Defined in Playbook

Facts gathered from Hosts

Passed from Inventory

```
---
- hosts: stredax.dev
  remote_user: root
  tasks:
```

```
- name: Determine old APEX dirs
  shell: ls -1d /app/apex/{5*,4*,o*} 2>/dev/null
  ignore_errors: True
  register: apex_dirs
- name: Remove old APEX dirs
  file:
    path: "{{ item }}"
    state: absent
    with_items: "{{ apex_dirs.stdout_lines }}"
- name: unzip apex and ords
  unarchive:
    src: ./apex_ords.tar.gz
    dest: /app/apex
- name: Link ords
  file:
    path: /app/apex/ords
    src: /app/apex/ords.{{ ords_version }}
    state: link
- name: Change Oracle PWDs
  script: change_ords_pwds.sh
  remote_user: orastred
```



## Playbook

### Variables

Jinja2 templating

Filters available

### Loops

Iterate: {{ item }}

Define: with\_items

Task Results (register)

```
---
- hosts: stredax.e
  remote_user: root
  tasks:
  - name: Determine old APEX dirs
    shell: ls -ld /app/apex/{5*,4*,o*} 2>/dev/null
    ignore_errors: True
    register: apex_dirs
  - name: Remove old APEX dirs
    file:
      path: "{{ item }}"
      state: absent
    with_items: "{{ apex_dirs.stdout_lines }}"
  - name: unzip apex and ords
    unarchive:
      src: ./apex_ords.tar.gz
      dest: /app/apex
  - name: Link ords
    file:
      path: /app/apex/ords
      src: /app/apex/ords.3.0.10
      state: link
  - name: Change Oracle PWDs
    script: change_ords_pwds.sh
    remote_user: orastred
```

# Ansible Playbooks (3/3)

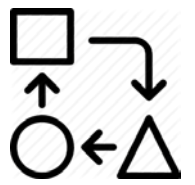
```

---
- hosts: webservers
vars:
- docroot: /var/www/stredax/public
tasks:
- name: Install Nginx
  apt:
    pkg: nginx
    state: installed
    update_cache: true
  register: nginxinstalled
  notify:
    - Start Nginx
- name: Create Web Root
  when: nginxinstalled|success
  file:
    dest: {{ docroot }}
    mode: 775
    state: directory
    owner: www-data
    group: www-data
  notify:
    - Reload Nginx

```

## handlers:

- name: Start Nginx  
service: name=nginx state=started
- name: Reload Nginx  
service: name=nginx state=reloaded



## Playbook

### Conditions

When result |  
success/failed/skipped

"foo" == "bar"

### Notify

### Handlers

Tasks

Executed when notified

# Ansible Roles

**Roles** organize multiple related tasks and data

predefined folder structure:  
`roles/rolename/...`

ansible executes file  
`main.yml`  
automatically, if present

ansible galaxy :  
community roles repository

**defaults**

Vars with default Values

**files**

Files to be copied unchanged

**handlers**

Playbook Handlers

**meta**

Dependencies

**tasks**

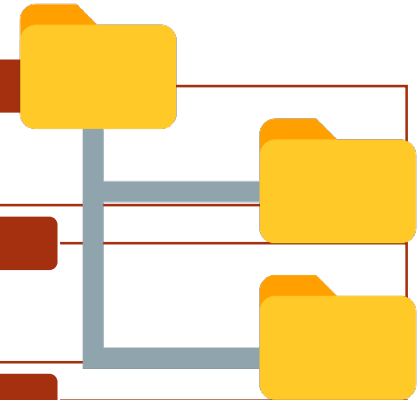
Playbook Tasks

**templates**

Content with Variables, e.g. config files.  
Will be processed by Jinja2 template engine

**vars**

Variables



# Liquibase – a closer Look

## Liquibase

### Installation

[Download Liquibase](#)

unpack

### Requirements:

JDK 1.7 or higher

JDBC driver

### Command Line Interface

liquibase

update[SQL]

rollback[SQL]

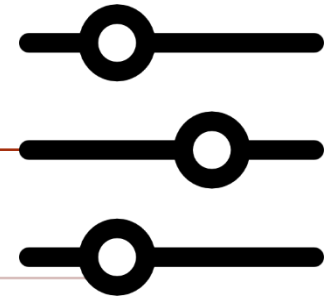
diff / generate

dbDoc



Java





# Parameters

## Required

`--changeLogFile=<path>`

`--username=<value>`

`--password=<value>`

`--url=<value>`

`--driver=<jdbc.driver.ClassName>`

liquibase  
.properties

working dir

Parameter Default Values

# Defining Changes in Liquibase: Changelog

## Changelogs

contain

attributes

preConditions

changesets

nestable

include

keeping  
Track

Table  
databasechangelog

```
databaseChangeLog:
```

```
- preConditions:
```

```
  dbms type: oracle
```

```
  runningAs:
```

```
    username: ROBBIE
```

```
- include:
```

```
  file: robbie-changeset-1.0.yaml
```

```
  relativeToChangelogFile: true
```

```
- include:
```

```
  file: robbie-changeset-1.1.yaml
```

```
  relativeToChangelogFile: true
```

```
- include:
```

```
  file: robbie-changeset-2.0.yaml
```

```
  relativeToChangelogFile: true
```

# Defining Changes in Liquibase: Changeset

## Changesets

contain	Attributes
	Actual changes
	Rollback
identified	ID
	Author
	Changelog Filename
runOnChange	Excutes on change
	Checksum in Table
	PL/SQL code

```

databaseChangeLog:
- changeSet:
  id: TabStockTicker-1
  author: robbie
  changes:
    - createTable:
      tableName: StockTicker
      columns:
        - column:
            name: id
            type: number
            autoIncrement: true
            constraints:
              primaryKey: true
              nullable: false
        - column:
            name: symbol
            type: varchar2(50)
  
```

# Liquibase Changes

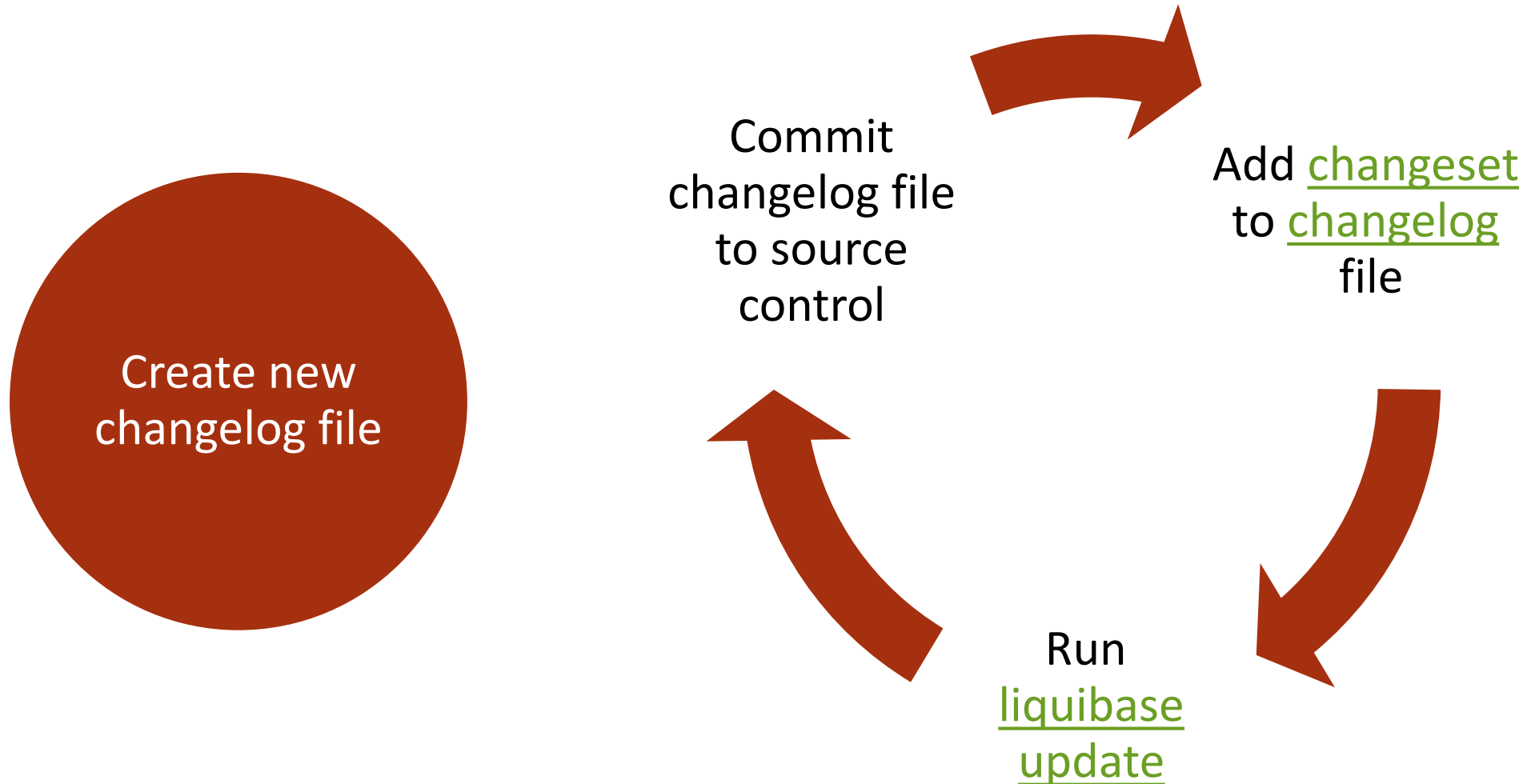
## Changes

Basic Operations	Bundled With rollback
Example	createTable SQL / SQLFile
<u>Extensions</u>	<u>Oracle</u> Build your own

```

databaseChangeLog:
- changeSet:
  id: TabTab1-1
  author: robbie
  changes:
    - createTable:
      tableName: Tab1
      columns:
        - column:
          name: id
          type: number
      rollback:
        - dropTable:
            tableName: Tab1
- changeSet:
  id: TabTab1-2
  author: robbie
  changes:
    - dropTable:
      tableName: Tab1
  rollback:
    changeSetId: TabTab1-1
    changeSetAuthor: robbie
  
```

# Liquibase Workflow



# Liquibase Procedure for the developer

Edit

create new local changeSet

Test

Test the changeSet SQL: Run liquibase update

Change

Change your Application: Edit the AppCode

Test

Test application code & database change together

Commit

Commit changeSet and application code.

# Liquibase Best Practices

Use a master changelog



One change per changeset



PL/SQL Code with `runOnChange="true"`



IDs are literals – use them



Document changesets with comments



Always think about rollback



Leverage Liquibase to manage your Reference Data



# Using Liquibase with Ansible

## Deploy Liquibase to Targets

see example on the right

## Execute liquibase update on target

File Module: Transport changelogs  
Shell Module: run liquibase

## Run Liquibase from Control Host

Local\_action with jdbc URLs  
See Blog Post

## ssh Tunnels

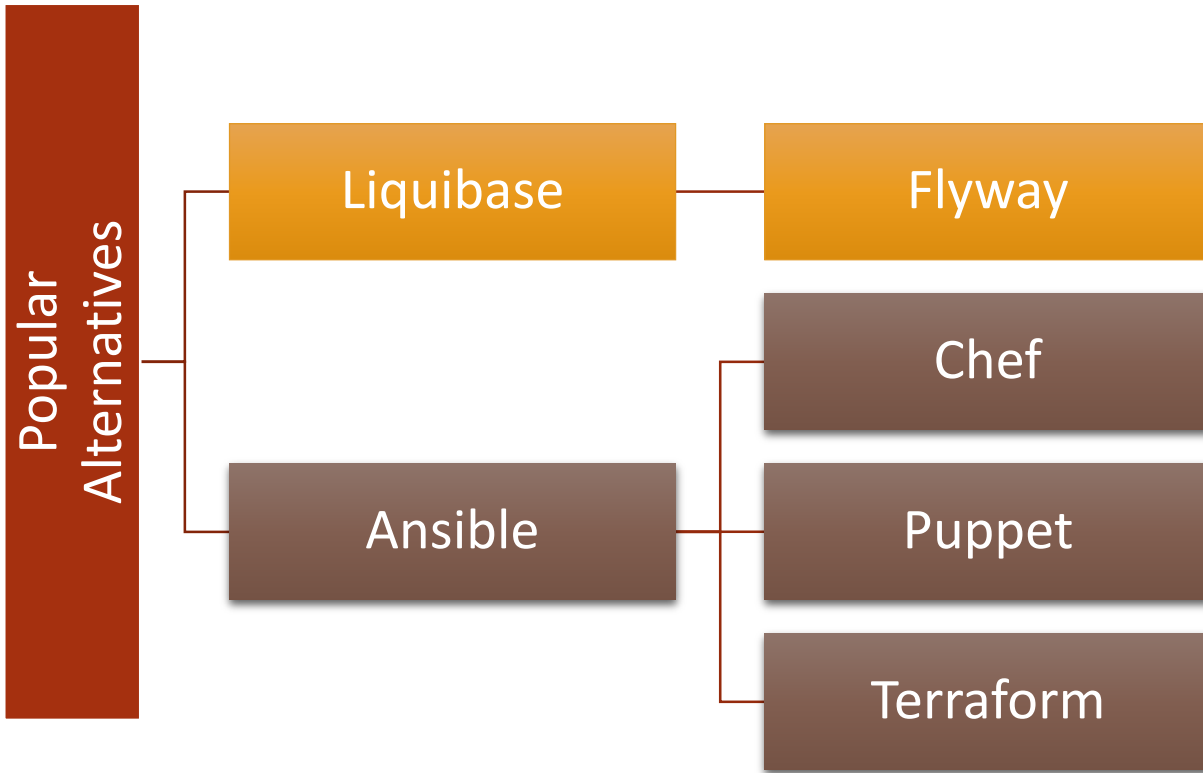
let ansible create ssh tunnels  
Run Liquibase as local action  
connecting to localhost  
See Blog Post

```
---  
- name: Create Liquibase directory  
  file:  
    name: /opt/liquibase  
    state: directory  
    mode: 0755  
  
- name: Install Liquibase  
  unarchive:  
    src:  
      https://github.com/liquibase/liquibase/releases  
      /download/liquibase-parent-{{ liquibase_version  
      }}/liquibase-{{ liquibase_version }}-bin.tar.gz  
    dest: /opt/liquibase  
    copy: no  
    mode: 0755  
    creates: /opt/liquibase/liquibase
```

# Demo



# Alternatives



# Want to know more?

## Liquibase

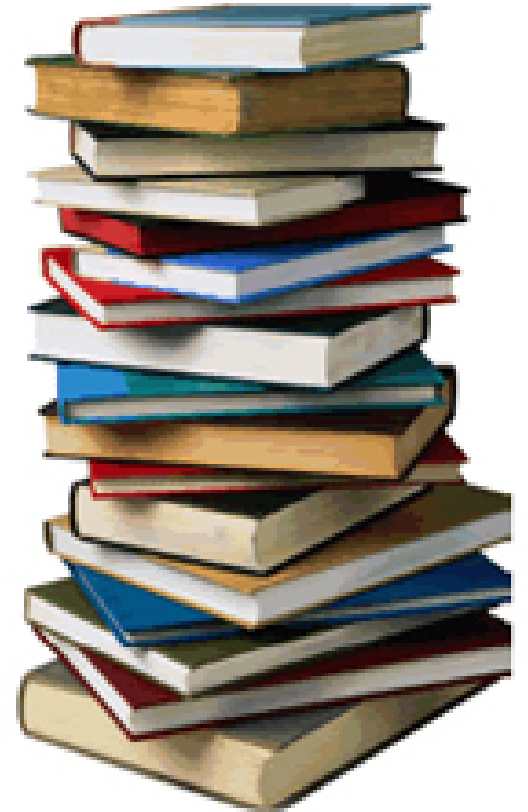
- [Homepage](#)
- [Official Documentation](#)
- [Tutorial: Change Happens by Blaine Carter](#)

## Ansible

- [Homepage](#)
- [Official Documentation](#)
- [Free Webinar](#)

## YAML

- [Tutorial](#)
- [Tutorial Learn X in Y Minutes](#)
- [Reference Card](#)
- [YAML 1.2 Spec](#)



# Conclusion

