02

Configuração do banco de dados

Transcrição

Vimos como poderíamos facilitar a linha de comando usando a configuração adequada de inventário. Falamos também sobre o que fazer se você estiver em um **ambiente heterogêneo e complexo** - o que deve acontecer quando você estiver trabalhando em produção -. O arquivo de inventário será utilizado para configurar diferentes usuários e chaves de autenticação, de acordo com o servidor.

A seguir, veremos como é feita a configuração do **MySQL**, criando um **novo banco**, que será usado pelo **Wordpress**.

Abaixo do trecho referente ao módulo `Instala pacotes de dependência do sistema operacional` , vamos gerar um módulo que preencheremos com `name` seguido de `Cria o banco do MySQL` . O módulo terá o nome `mysql_db` .

```
- name: 'Cria o banco do MySQL'
  mysql_db:
    name: wordpress_db
    login_user: root
    state: present
```

Bancos diferentes e mais famosos - como Oracle, postgreSQL, cross server -, vão ter uma **coleção de módulos** criados dentro do Ansible, que **vai nos ajudar** a administrar e operar o banco. Nós poderemos ver esses módulos na documentação, no caso, analisaremos a seção referente ao `mysql_db` . Encontraremos uma **série de parâmetros** nesta parte. Nem todos os parâmetros serão utilizados no exemplo, mas será útil para **entender como eles funcionam**.

A documentação do Ansible (http://docs.ansible.com/ansible/latest/modules/mysql_db_module.html) é **muito completa**. Nela, você poderá consultar os módulos de outros bancos quando for necessário. Mais adiante, conversaremos sobre como isso pode ser utilizado.

Trabalhando especificamente com `mysql_db` do nosso projeto, teremos que definir qual será o nome do banco que criado: `wordpress_db` . Em seguida, informaremos ao módulo qual usuário do MySQL nós estamos usando para logar. Executaremos um comando dentro do MySQL, depois, usaremos o usuário padrão da instalação do Ubuntu, o `root` , para fazer a autenticação.

No pacote do MySQL Server 5.6, usando o sistema operacional Ubuntu 14.04, por padrão, será criado um usuário sem senha. Caso você precise adicionar esse passo no seu projeto, basta **incluir** o parâmetro `login_password` , passando a senha em seguida. Este **não é o nosso cenário**, por isso, será desnecessário informar a senha.

O próximo passo será informar o `state` , que é incluído em quase todos os módulos do Ansible. Usaremos o estado `present` - com isso, indicamos que desejamos criar um banco de dados. Os estados disponíveis para manipulação de cada módulo mudam de acordo com aquele que é trabalhado. Veremos um exemplo na documentação de como isso funciona para o `mysql_db` .

Como é explicado na documentação, podemos informar no parâmetro ***state*** o seguinte:

- Present: com o qual criamos um banco de dados;
- Absent: com o qual destruímos um banco de dados;

- Dump: Traduzido para o português, significa **despejo**. Com o database dump, temos um registro da estrutura de tabelas, trilhas, o elemento que você tenha criado dentro do banco de dados, juntamente com os *inserts* (entradas).
- Import: é o contrário do Dump. Com ele, pegamos um script SQL pronto e o aplicamos em um banco vazio.

Encontramos **bons exemplos na documentação do Ansible**, no caso, estamos criando uma entrada similar ao modelo apresentado:

```
- name: Create a new database with name 'bobdata'
  mysql_db:
    name: bobdata
    state: present
```

Vemos que o nome dado neste caso foi `bobdata`, mas no nosso projeto, chamaremos de `wordpress_db`. Na documentação, vemos como podemos aplicar ou remover um `dump` do estado.

Faremos uma comparação com o módulo `apt`, que foi o mais utilizado até o momento, consultaremos a documentação para ver quais são os estados disponíveis. No caso, podemos passar as seguintes informações para o parâmetro `state`:

- Latest: com ele definimos que usaremos a última versão disponível dos pacotes utilizados;
- Absent: com ele desisntalamos o pacote, equivalente ao `apt-get-remover`;
- Present: se o pacote já estiver instalado na máquina, será usada a verão disponível, mesmo que ela seja mais antiga;
- Build-dep: sua utilidade é nos garantir que todas as dependências de um determinado pacote serão construídas antes do comando ser executado.

A seguir, vamos rodar o playbook de novo. No Terminal, vamos rodar o seguinte comando:

```
$ ansible-playbook -i hosts provisioning.yml
```

No comando, especificamos que o arquivo de inventário (`hosts`) e o playbook - chamado `provisioning.yml`. Lembre-se que agora **não passaremos** mais o nome do usuário e a chave-privada, porque eles já foram inclusos no arquivo de hosts. Também é importante salvarmos as alterações feitas no arquivo. Quando executado, esperamos que nenhum pacote seja instalado.

De acordo com o retorno, não foi necessário instalar nenhum pacote, e somos informados de que o banco foi criado nessa máquina.

```
$ ansible-playbook -i hosts provisioning.yml

PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.17.177.40]

TASK [Instala pacotes de dependencia do sistema operacional] *******************
ok: [172.17.177.40] => (item=[u'php5', u'pache2', u'libapache2-mod-php5', u'php5-gd', u'libssh2

TASK [Cria o banco do MySQL] ***************************************************
changed: [172.17.177.40]
```

```
PLAY RECAP *********************************************************************
172.17.177.40              : ok=3      changed=1      unreachable=0      failed=0
```

Em seguida, iremos logar para mostrar com seria se tentássemos fazer isso manualmente.

```
$ vagrant ssh
```

Passaremos o usuário `root`, **sem a necessidade** de incluir a senha.

```
vagrant@vagrant-ubuntu-trusty-64:~$ mysql -u root
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 37
Server version: 5.6.33-0ubuntu0.14.04.1 (Ubuntu)
```

Se executarmos `show databases`, conseguiremos ver os bancos que foram criados.

```
mysql> show databases;
+----------------------+
| Database             |
+----------------------+
| information_schema   |
| mysql                |
| performance_schema   |
| wordpress_db         |
+----------------------+
4 rows in set (0.00 sec)
```

Vemos que `wordpress_db` recém criado consta na lista. Se usarmos o comando `use wordpress_db`, **confirmaremos sua utilização**.

```
mysql> use wordpress_db;
Database changed
mysql> show tables;
Empty set (0.00 sec)
```

Não encontraremos tabelas nele, porque elas ainda não foram criadas. Por último, vamos sair da máquina virtual.

```
Mysql> quit;
Bye
vagrant@vagrant-ubuntu-trusty-64:~$ logout
Connection to 127.0.0.1 closed.
```

Se quiséssemos deletar o banco, bastaria alterar o estado no arquivo `provisioning.yml` para `absent`.

```
- name: 'Cria o banco do MySQL'
  mysql_db:
    name: wordpress_db
```

```
    login_user: root
    state: absent
```

Ao rodarmos o playbook novamente, o banco deverá ter sido deletado. No retorno, veremos que houve uma **modificação**.

```
$ ansible-playbook -i hosts provisioning.yml

PLAY [all] *********************************************************************

TASK [Gathering Facts] *********************************************************
ok: [172.17.177.40]

TASK [Instala pacotes de dependencia do sistema operacional] *******************
ok: [172.17.177.40] => (item=[u'php5', u'pache2', u'libapache2-mod-php5', u'php5-gd', u'libssh2

TASK [Cria o banco do MySQL] ***************************************************
changed: [172.17.177.40]

PLAY RECAP *********************************************************************
172.17.177.40              : ok=3    changed=1    unreachable=0    failed=0
```

É sempre importante **conferir os comandos** que ele conseguiu gerar uma modificação no estado. Depois, vamos logar como root e verificar "cadê o banco de dados?".

```
mysql> show databases;
+----------------------+
| Database             |
+----------------------+
| information_schema   |
| mysql                |
| performance_schema   |
+----------------------+
3 rows in set (0.00 sec)
```

Vemos que `wordpress_db` sumiu da tabela, isto sinaliza que o Ansible está funcionando conforme o esperado e se comunica perfeitamente com o MySQL.

Se acessarmos de novo a máquina virtual, veremos qual seria o equivalente a tentar criar o banco sem o Ansible. Veremos o que seria preciso fazer logado como usuário com permissão de administrador (o root utilizado).

Nós teríamos que rodar o comando `create database wordpress_db` dentro do MySQL. Para isto, teríamos que criar um script MySQL, passando na linha de comando ou rodando manualmente - lembrando que esta última nunca é a melhor opção, porque a reprodução é trabalhosa.

```
mysql> create database wordpress_db;
Query OK, 1 row affected (0.00 sec)
```

A tendencia é que ao ser executado manualmente, o processo esteja mais propício a resultar em erros. Vamos retornar o estado para `present` .

```
- name: 'Cria o banco do MySQL'
  mysql_db:
    name: wordpress_db
    login_user: root
    state: present
```

Agora quando executarmos uma última vez, ficaremos no estado desejado que informa a criação do banco. Como o banco foi criado manualmente antes, seremos informados que ele já existe.

```
$ ansible-playbook -i hosts provisionng.yml

PLAY [all] ********************************************************************************

TASK [Gathering Facts] ********************************************************************
ok: [172.17.177.40]

TASK [Instala pacotes de dependencia do sistema operacional] ******************************
ok: [172.17.177.40] => (item=[u'php5', u'pache2', u'libapache2-mod-php5', u'php5-gd', u'libssh2

TASK [Cria o banco do MySQL] **************************************************************
ok: [172.17.177.40]

PLAY RECAP ********************************************************************************
172.17.177.40              : ok=3      changed=1      unreachable=0      failed=0
```

Com isso, vimos como é possível **manipular a criação de banco de dados** dentro do MySQL, usando o Ansible, nos aprofundamos sobre **estado de módulos**. Se você tiver alguma dúvida sobre o assunto, seja sobre estados disponíveis ou parâmetros de qualquer configuração, você pode **consultar a documentação** do Ansible.

Esta é a melhor forma para se aprender sozinho e **dominar os módulos** mais necessários, eu avancei bastante lendo a documentação.

A seguir, conversaremos sobre como gerar o usuário que vai administrar o banco de dados do Wordpress.