HashiCorp Terraform

Infrastructure as Code para quem entende de SQL e Python

APRENDA EM 8 MINUTOS
(SE VOCÊ SOUBER SQL E PYTHON)

Passe pro lado e veja o ebook completo

Jornada de Dados

www.suajornadadedados.com.br

Introdução Terraform

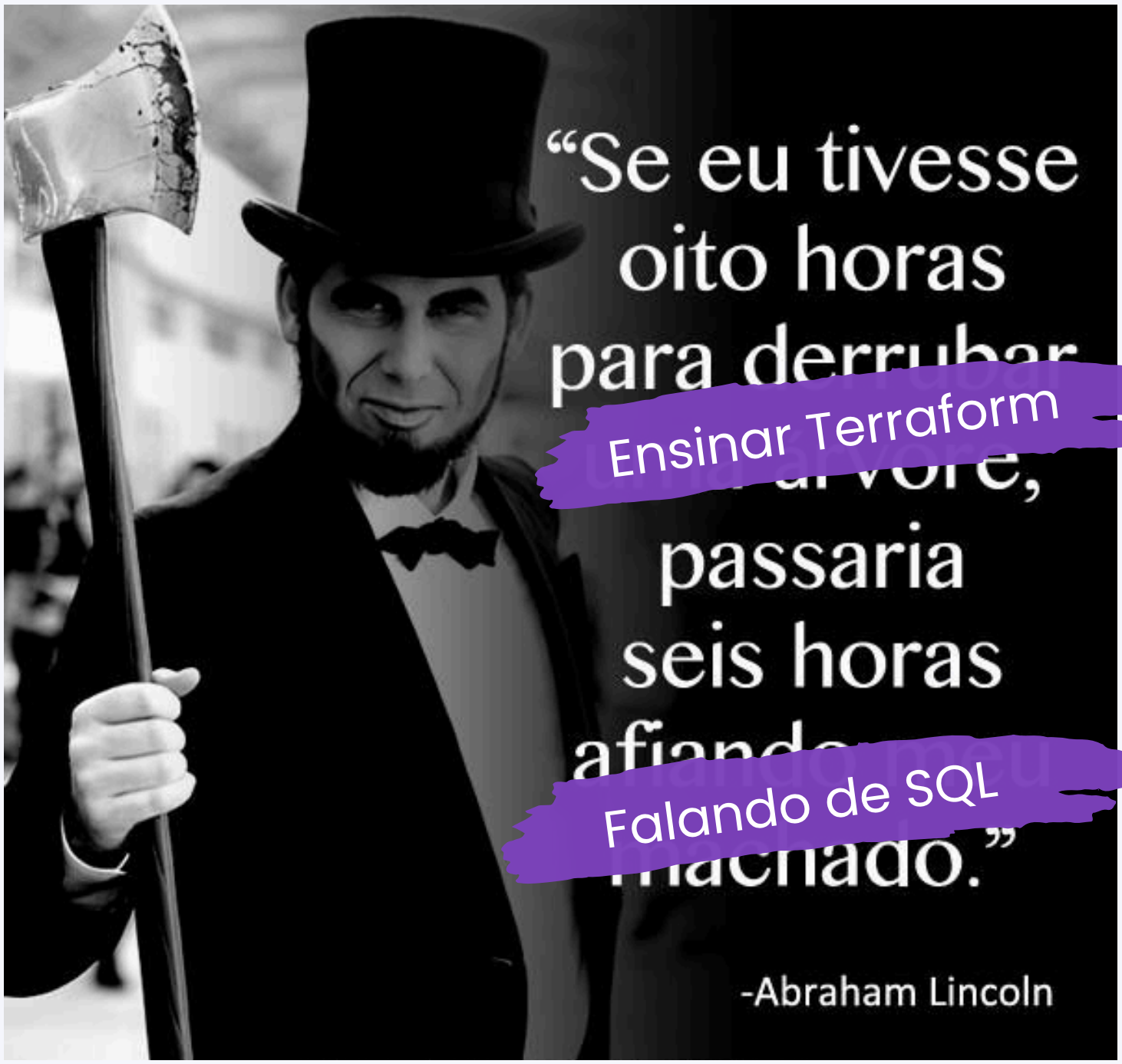
Terraform é uma ferramenta de IaC (Infrastructure as Code) que permite a criação, atualização e gestão de infraestrutura de forma declarativa.

E eu vou te **ensinar em 8 minutos.**

Mas para eu te ensinar em 8 minutos. **Precisamos falar 6 minutos sobre SQL e Python.**

Vamos la?

É a mesma coisa do machado



Sabe o Python?

Python é uma linguagem de programação amplamente utilizada em diversas áreas, incluindo desenvolvimento web, ciência e engenharia de dados, automação e muito mais.

Python é imperativo, o que significa que você **escreve explicitamente o passo a passo das operações que deseja realizar.** Se liga no exemplo.

Crie ai um algoritmo que **ordene elementos** em Python

Ficou nervoso né?

Mas fica tranquilo, aqui não é entrevista de live code.

Eu só quero que você saiba que existem **várias soluções** para isso, destacando a natureza imperativa de Python, que permite especificar **passo a passo como a tarefa deve ser realizada.**

Olha resolvendo por **Bubble Sort**

Exemplo 1 – Bubble Sort:

É um algoritmo de **ordenação simples**. Ele funciona percorrendo a lista, comparando e trocando elementos de lugar várias vezes.

```python
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]
    return arr

import random
numbers = [random.randint(1, 1000) for _ in range(100)]
sorted_numbers = bubble_sort(numbers)
print(sorted_numbers)
```

Quer ver outra solução?

Exemplo 2 – Quick Sort:

É um algoritmo de **ordenação eficiente e amplamente utilizado**. Ele segue o paradigma "dividir para conquistar".

```python
import random

def quick_sort(arr):
    if len(arr) <= 1:
        return arr
    pivot = arr[len(arr) // 2]
    left = [x for x in arr if x < pivot]
    middle = [x for x in arr if x == pivot]
    right = [x for x in arr if x > pivot]
    return quick_sort(left) + middle + quick_sort(right)

numbers = [random.randint(1, 1000) for _ in range(100)]
sorted_numbers = quick_sort(numbers)
print(sorted_numbers)
```

Legal não é?

e SQL?

SQL é uma **linguagem declarativa** usada para gerenciar e manipular bancos de dados relacionais.

Em vez de especificar como realizar uma tarefa, **você descreve o que deseja que seja feito**, e o sistema determina a melhor forma de executar a operação.

Vamos ver outro exemplo?

Crie ai um algoritmo que **ordene elementos** em SQL

Ficou tranquilo né?

Dessa vez foi fácil, queria que toda live code fosse assim. Em SQL, **ordenar dados é simples**.

Podemos ordenar os registros de uma tabela simplesmente escrevendo **ORDER BY** e pronto.

```sql
-- Ordenar os registros pela data de venda de forma ascendente
SELECT * FROM vendas
ORDER BY data_venda ASC;

-- Ordenar os registros pelo preço de venda de forma descendente
SELECT * FROM vendas
ORDER BY preco DESC
```

Declarativo vs Imperativo

Abordagens **declarativa e imperativa** são dois paradigmas de programação distintos.

Declarativo = você descreve o que deseja que seja feito (SQL)

Imperativo = você especifica como fazer (Python)

Já passaram uns 8 minutos né,
Sabe o que já temos?

O machado afiado

Duvida?

Vou botar **o primeiro slide** de novo e você vai entender de Terraform

Introdução Terraform

Terraform é uma ferramenta de IaC (Infrastructure as Code) que permite a criação, atualização e gestão de infraestrutura de **forma declarativa**.

Viu só?

Olha **quem é declarativo** também!

O que é bom, se **fosse imperativo** seria muito mais difícil, rs.

Declarativo como SQL

Assim como o SQL, onde você **declara o que deseja criar**, em dados.

```sql
CREATE TABLE users (
  id INT PRIMARY KEY,
  name VARCHAR(100)
);
```

o Terraform permite **declarar a infraestrutura que você** deseja.

```hcl
resource "aws_s3_bucket" "meu_bucket" {
  bucket = "meu-bucket-de-exemplo"
  acl    = "private"
}
```

Plano de Execução

Antes de aplicar mudanças, o Terraform gera um **plano de execução que mostra** o que será alterado.

```
terraform plan
```

Da mesma forma que o **SQL pode exibir um plano de execução para uma query**.

```
-- Explica o plano de ordenação anterior
EXPLAIN SELECT * FROM vendas
ORDER BY data_venda ASC;
```

Aplicando Mudanças

O comando **terraform apply** é usado para aplicar as mudanças descritas no código.

```
terraform apply
```

Assim como executar **uma query SQL** para atualizar um banco de dados, o Terraform aplica as configurações na infraestrutura.

```
CREATE TABLE users (
  id INT PRIMARY KEY,
  name VARCHAR(100)
);
```

Destruindo Recursos

O comando **terraform destroy** remove todos os recursos gerenciados pelo Terraform.

```
terraform destroy
```

Isso é similar ao **comando DROP no SQL**, que remove tabelas e dados, garantindo que nada seja deixado para trás.

```
DROP TABLE users;
```

Ou seja

Tanto o Terraform quanto o SQL seguem uma abordagem declarativa, **onde o foco é descrever o resultado desejado**.

Essa semelhança facilita a adoção do Terraform por analistas de dados, **aproveitando o conhecimento existente em SQL para gerenciar infraestrutura.**

Resumo

SQL ->

Linguagem declarativa **para gerenciar** Banco de dados

Terraform ->

Linguagem declarativa **para gerenciar** Infraestrutura

Criamos um bucket (recurso)

```
resource "aws_s3_bucket" "meu_bucket" {
  bucket = "meu-bucket-de-exemplo"
  acl    = "private"
}
```

```
terraform plan
```

```
terraform apply
```

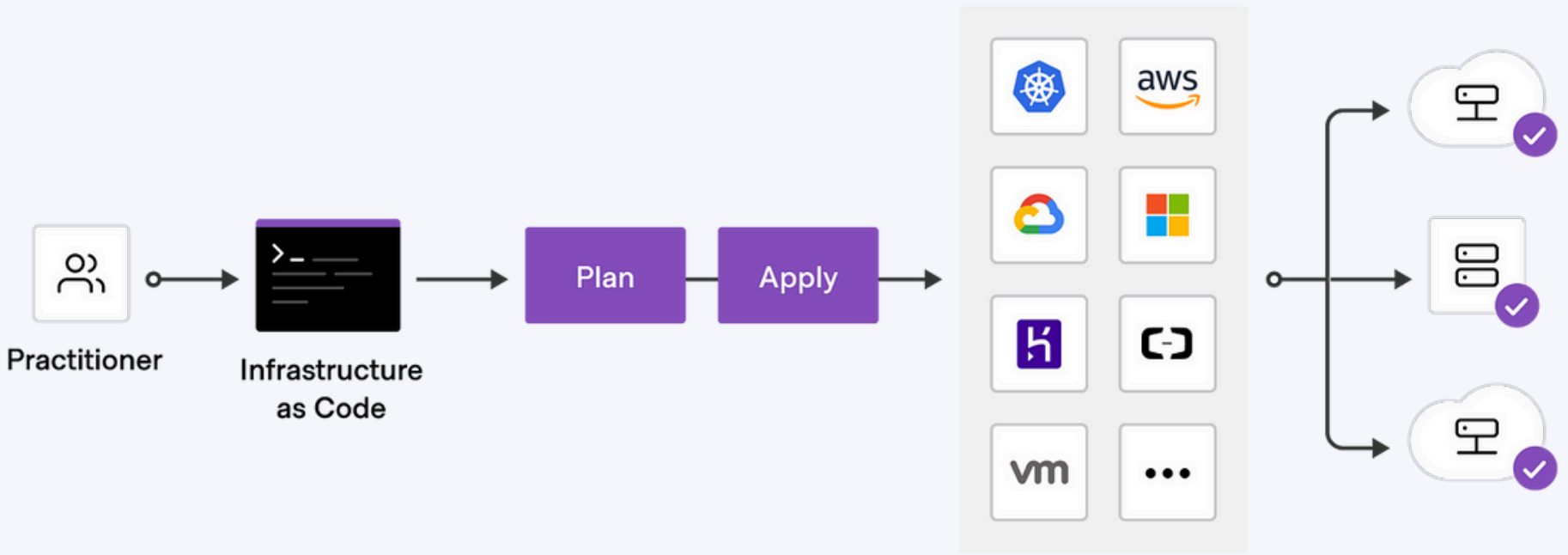
```
terraform destroy
```

De uma maneira simplificada, esses são os comandos para criar recursos.

Jornada de Dados

www.suajornadadedados.com.br

Você fala o que quer...
... E o terraform se vira



Resumo: Terraform é uma ferramenta de IaC (Infrastructure as Code) que permite definir, provisionar e gerenciar infraestrutura através **de um código declarativo.** Ele é amplamente utilizado devido **à sua capacidade de automatizar a criação de ambientes.**

Difícil?

Te falei que **não é era.**

Não deixe ninguém te falar que programação **não é para você**, ok?

Apesar de ter muitos termos **específicos**, no final é uma espécie de **"SQL"** para gerenciar recursos de infra

Obrigado!

Se alguma postagem minha já te ajudou, curte e comente!

Sua interação pode ajudar outras pessoas a se beneficiarem desse conhecimento.

(E ajuda muito nosso projeto!)

Qualquer dúvida me manda uma mensagem!

Jornada de Dados

www.suajornadadedados.com.br