

Terraform on Oracle Cloud Infrastructure

A Primer for Database Administrators





Sean Scott

25 years working with Oracle technology

Oracle OpenWorld :: Collaborate/IOUG ::

Upgrades & Migration :: High Availability :: Disaster Recovery :: Scalability

RAC :: Data Guard :: Engineered Systems :: Cloud :: Performance

Sharding :: Observability Platforms (TFA, AHF, GIMR, CHA)

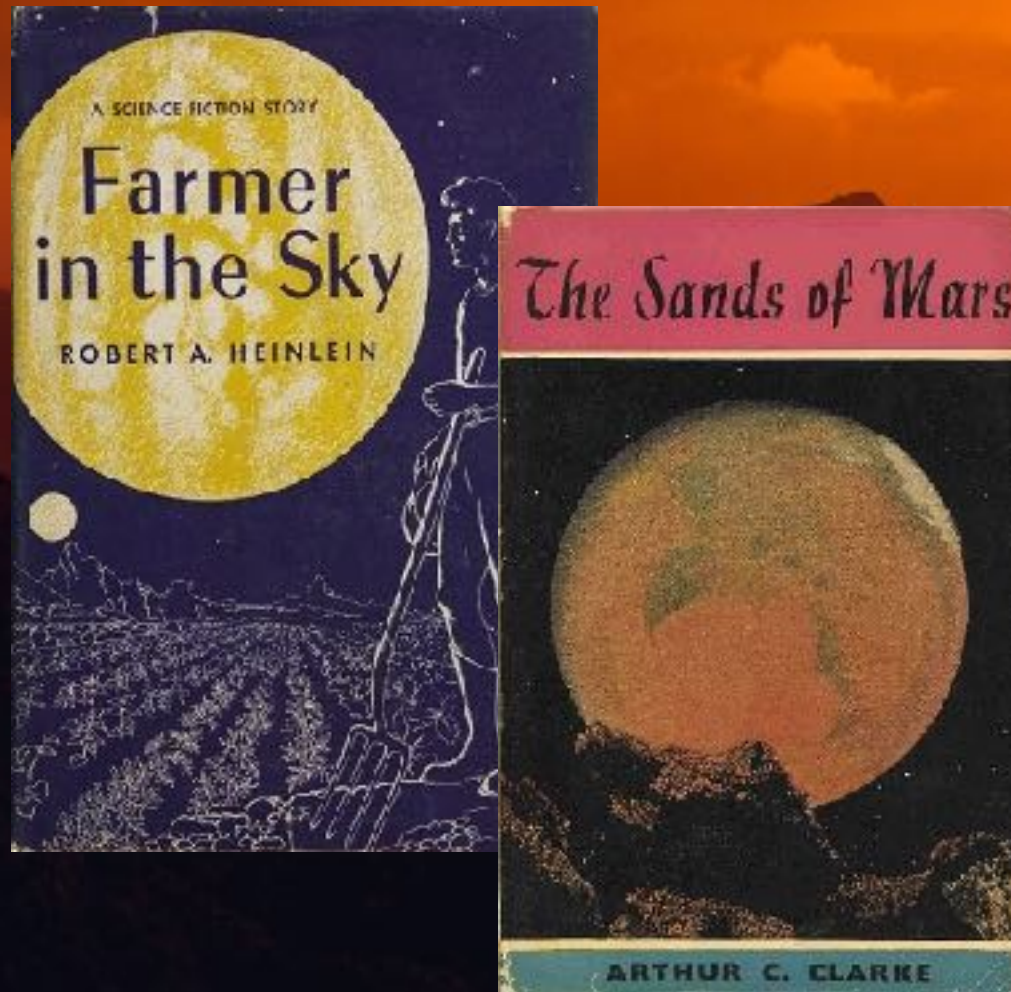
DevOps :: Automation :: Infrastructure as Code

Containers and Docker :: Virtualization



ter·ra·form *verb*

Latin "*terra*" (earth), English "*form*"



*To transform an environment to
support life*













Ter•ra•form *noun*

An Infrastructure as Code (IaC) tool from Hashicorp. Terraform defines, provisions and manages cloud & on-premises infrastructure.



HashiCorp

Terraform

An underwater photograph showing a clear blue water environment. In the upper left, there is a large, dense cluster of green seaweed or coral. Several fish of various sizes are swimming in the water. In the lower right, a diver is visible, partially obscured by the water's surface. The overall scene is serene and natural.

Use Terraform to build:

Storage

Compute

Networks

Security

DNS

Configuration

Databases

"The steps for building a 10 liter fish tank are:"

- Get fish tank a , pump b , heater c ...
- Assemble them per...
- Add x liters water...
- Add y grams salt...
- Set temperature to $z^{\circ}\text{C}$...
- Add n fish...

Scales poorly!

Imperative languages

Declarative languages

Tell the expert:

Scale and shape

Special needs, requirements

What you already have

These "experts" are
Terraform providers

Terraform *provider*

Providers are *implementation experts*

- Understand dependencies
- Interpret configurations
- Build the *declared* infrastructure

Available for OCI, Azure, AWS, GCP, on-premises, etc.



Terraform advantages

Manage infrastructure lifecycles

Identify configuration drift

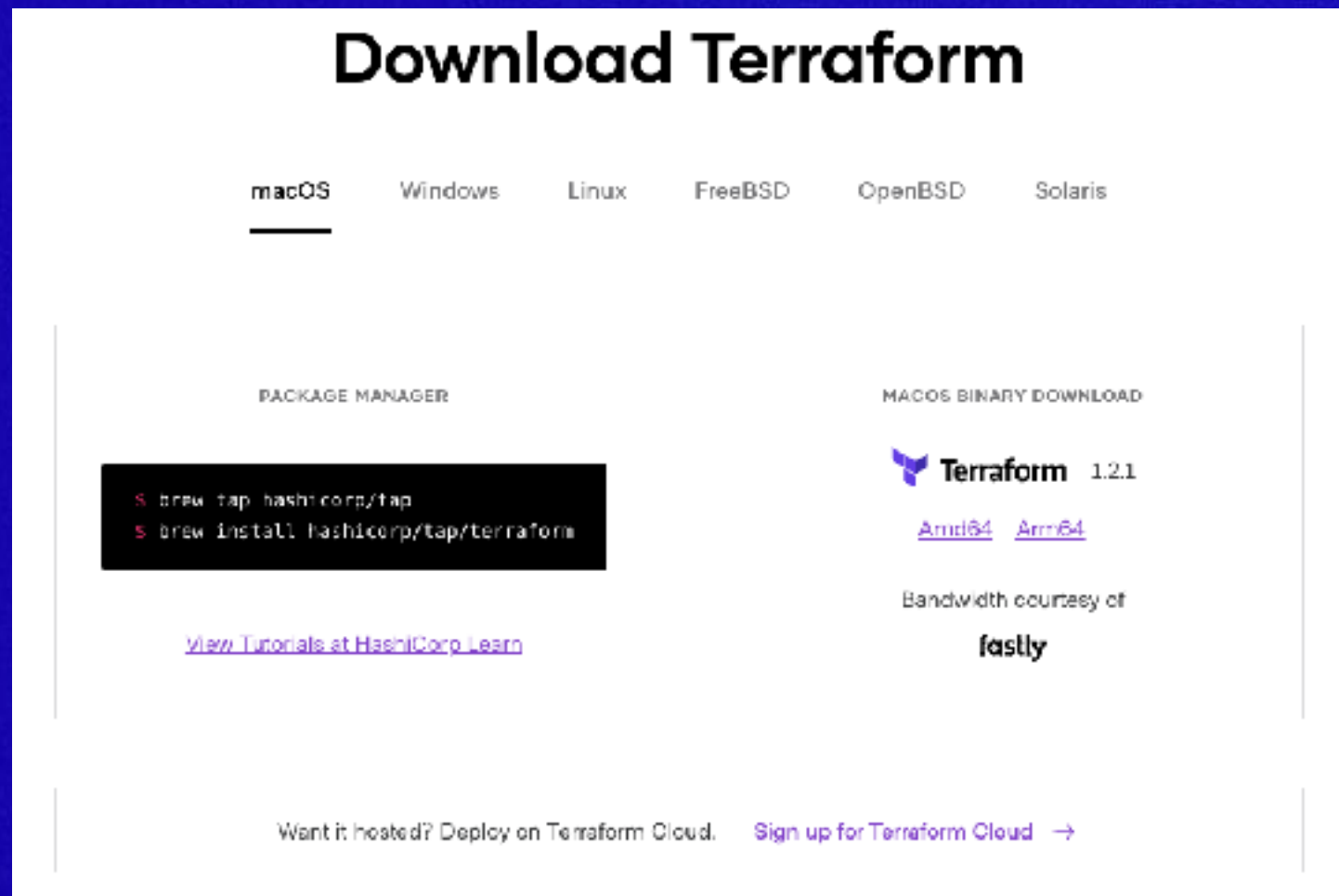
Repeatable outcomes

Reduced complexity

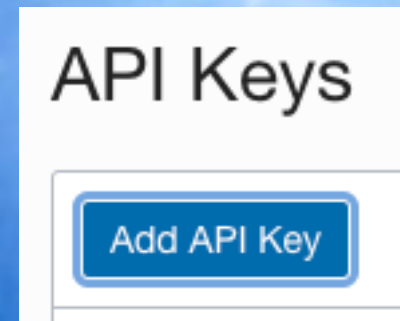
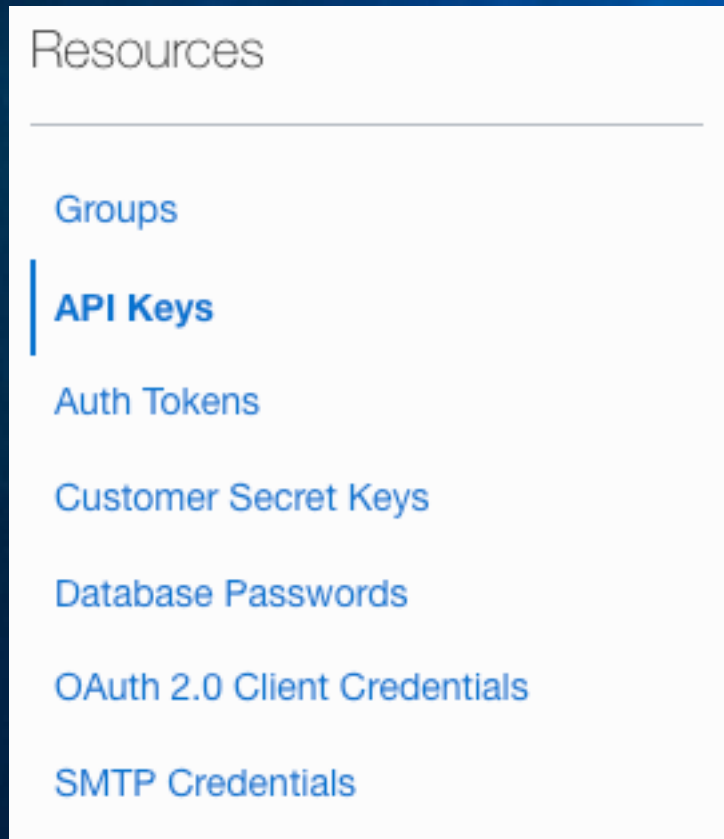
Idempotent

Get Terraform

<https://www.terraform.io/downloads>

A screenshot of the Terraform download page. The page has a white background with a black header bar. The title "Download Terraform" is in large, bold, black font. Below the title is a navigation bar with links for "macOS", "Windows", "Linux", "FreeBSD", "OpenBSD", and "Solaris". The "macOS" link is underlined. The main content area is divided into two columns. The left column is titled "PACKAGE MANAGER" and contains a black box with white text showing two terminal commands: "brew tap hashicorp/tap" and "brew install hashicorp/tap/terraform". Below this is a link "View Tutorials at HashiCorp Learn". The right column is titled "MACOS BINARY DOWNLOAD" and features the Terraform logo, the version "1.2.1", and links for "Am64" and "Arm64". Below this is a note "Bandwidth courtesy of fastly". At the bottom of the page, there is a footer with the text "Want it hosted? Deploy on Terraform Cloud." and a link "Sign up for Terraform Cloud" with a right arrow.

Generate an OCI public key



Identity & Security → Users

Choose a user to run Terraform

Select "API Keys" in the Resources menu

Click the "Add API Key" button

Generate an OCI public key (continued)

- Follow the dialog instructions
- Click the "Add" button

Add API Key [Help](#)

Note: An API key is an RSA key pair in PEM format used for signing API requests. You can generate the key pair here and download the private key. If you already have a key pair, you can choose to upload or paste your public key file instead. [Learn more](#)

☒ Generate API Key Pair ☐ Choose Public Key File ☐ Paste Public Key

Public Key

i Download the private key. It will not be shown again. After you download it, [change the file permissions](#) so only you can view it.

[Download Private Key](#) [Download Public Key](#)

Add [Cancel](#)

- Copy the key fingerprint

API Keys

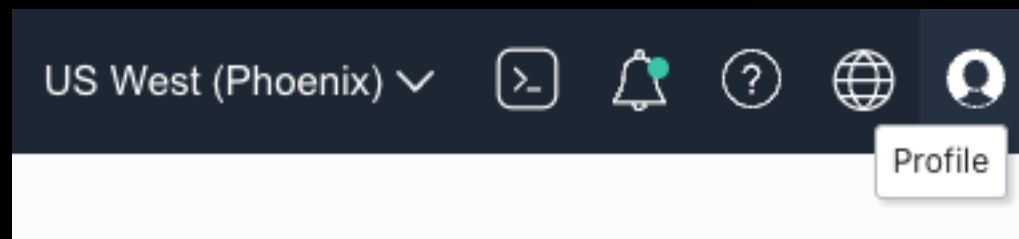
Add API Key

Fingerprint

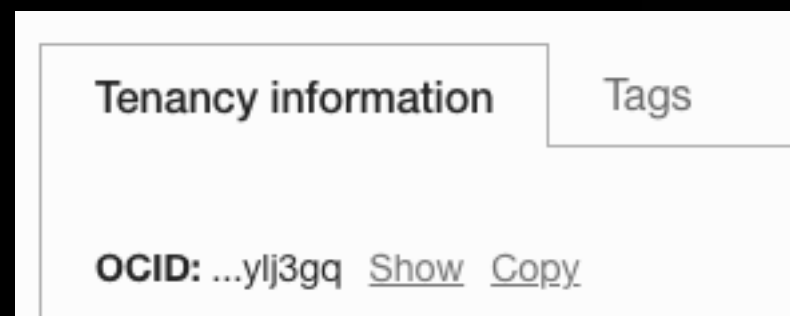
[Redacted Fingerprint]

Get the `tenancy_ocid`

Click the profile at top right and select "Tenancy" from the menu

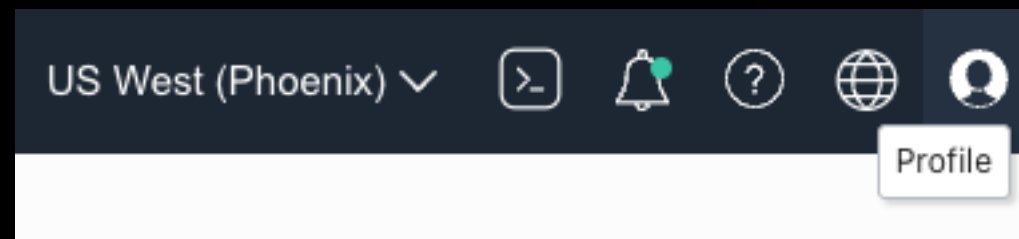


Use the "Copy" link under "Tenancy information"

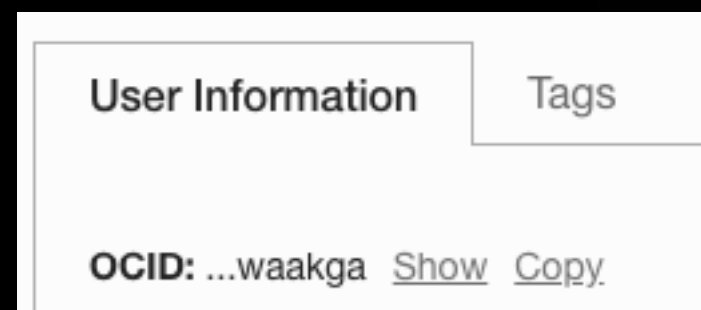


Get the user_ocid

Click the profile at top right and select "User Settings" from the menu

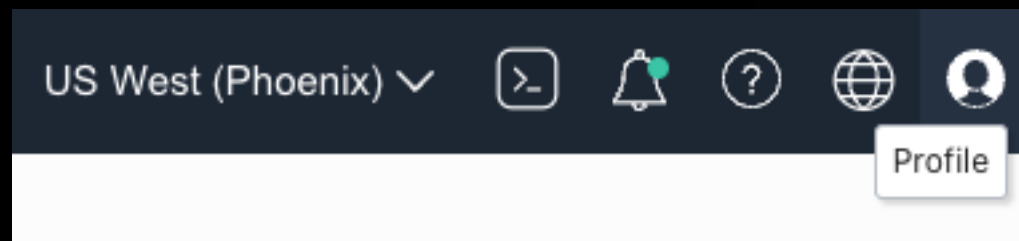


Use the "Copy" link under "User information"

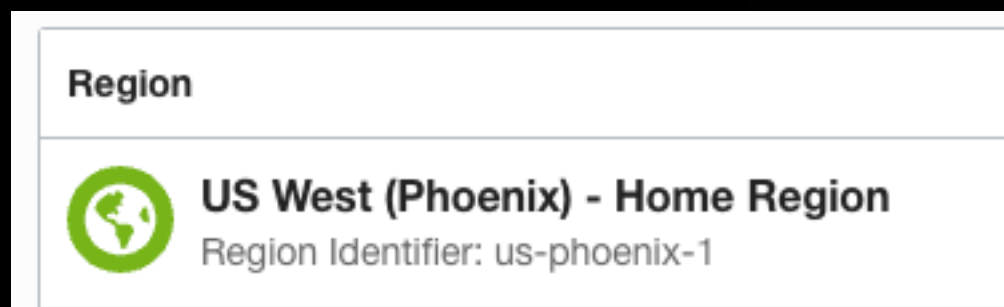


Get the Region Identifier

Click the profile at top right and select the region dropdown from the menu



Copy the
Region Identifier



Get the private key path and fingerprint

`private_key_path`

Path used to create the keys

`fingerprint`

The fingerprint generated
during API key creation

Start a new Terraform project

Create a project directory & add files:

- providers.tf
- variables.tf
- terraform.tfvars
- main.tf
- outputs.tf

Project files:

<https://github.com/oraclesean/terraform-for-oracle-dbas>



providers.tf

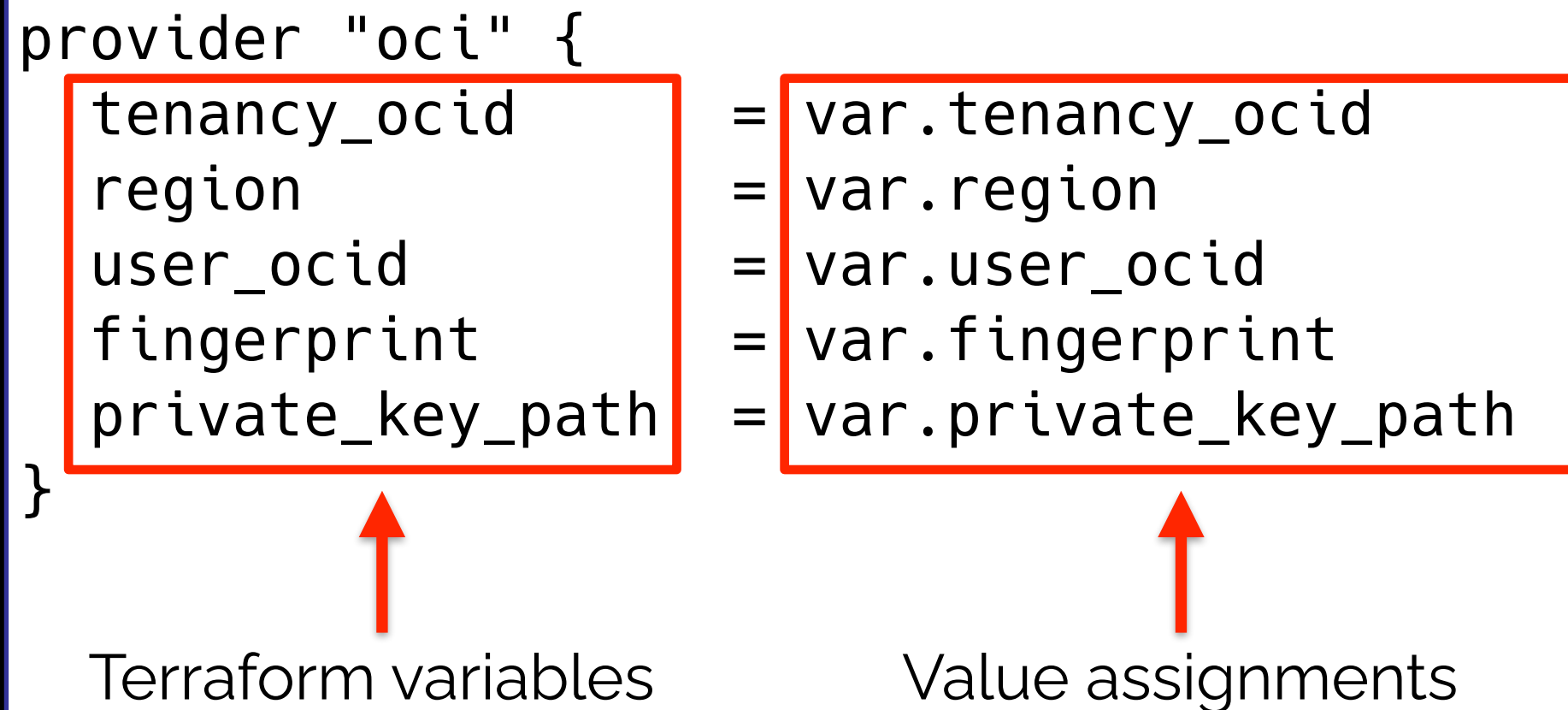
```
provider "oci" {  
  tenancy_ocid  
  region  
  user_ocid  
  fingerprint  
  private_key_path  
}
```

Terraform variables

=

```
var.tenancy_ocid  
var.region  
var.user_ocid  
var.fingerprint  
var.private_key_path
```

Value assignments



variables.tf

```
# Terraform tenancy variables
```

```
variable "tenancy_ocid" {}  
variable "region" {}  
variable "user_ocid" {}  
variable "fingerprint" {}  
variable "private_key_path" {}
```

Value assignment *could* go here

Variable definitions

terraform.tfvars

```
# Terraform tenancy variable values
```

```
tenancy_ocid = Your tenancy_ocid  
region       = Your region identifier  
user_ocid    = Your user_ocid  
fingerprint  = Your fingerprint  
private_key_path = Your private_key_path
```

Hard-coded variable
assignments



Limiting hard-coded value
assignments to `terraform.tfvars`
means no changes are needed
elsewhere to run this *same*
configuration on different tenancies,
to scale the configuration, etc.!



Same variables as defined in `variables.tf`

Test the configuration

- From the project directory run:
terraform init
terraform plan
terraform apply



terraform init

```
> terraform init
```

```
Initializing the backend...
```

```
Initializing provider plugins...
```

- Finding latest version of hashicorp/oci...
- Installing hashicorp/oci v4.76.0...
- Installed hashicorp/oci v4.76.0 (signed by HashiCorp)

```
Terraform has been successfully initialized!
```

You may now begin working with Terraform. Try running "terraform plan" to see any changes that are required for your infrastructure. All Terraform commands should now work.

If you ever set or change modules or backend configuration for Terraform, rerun this command to reinitialize your working directory. If you forget, other commands will detect it and remind you to do so if necessary.

terraform plan

```
> terraform plan
```

```
No changes. Your infrastructure matches the configuration.
```

```
Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
```


terraform apply

```
> terraform apply
```

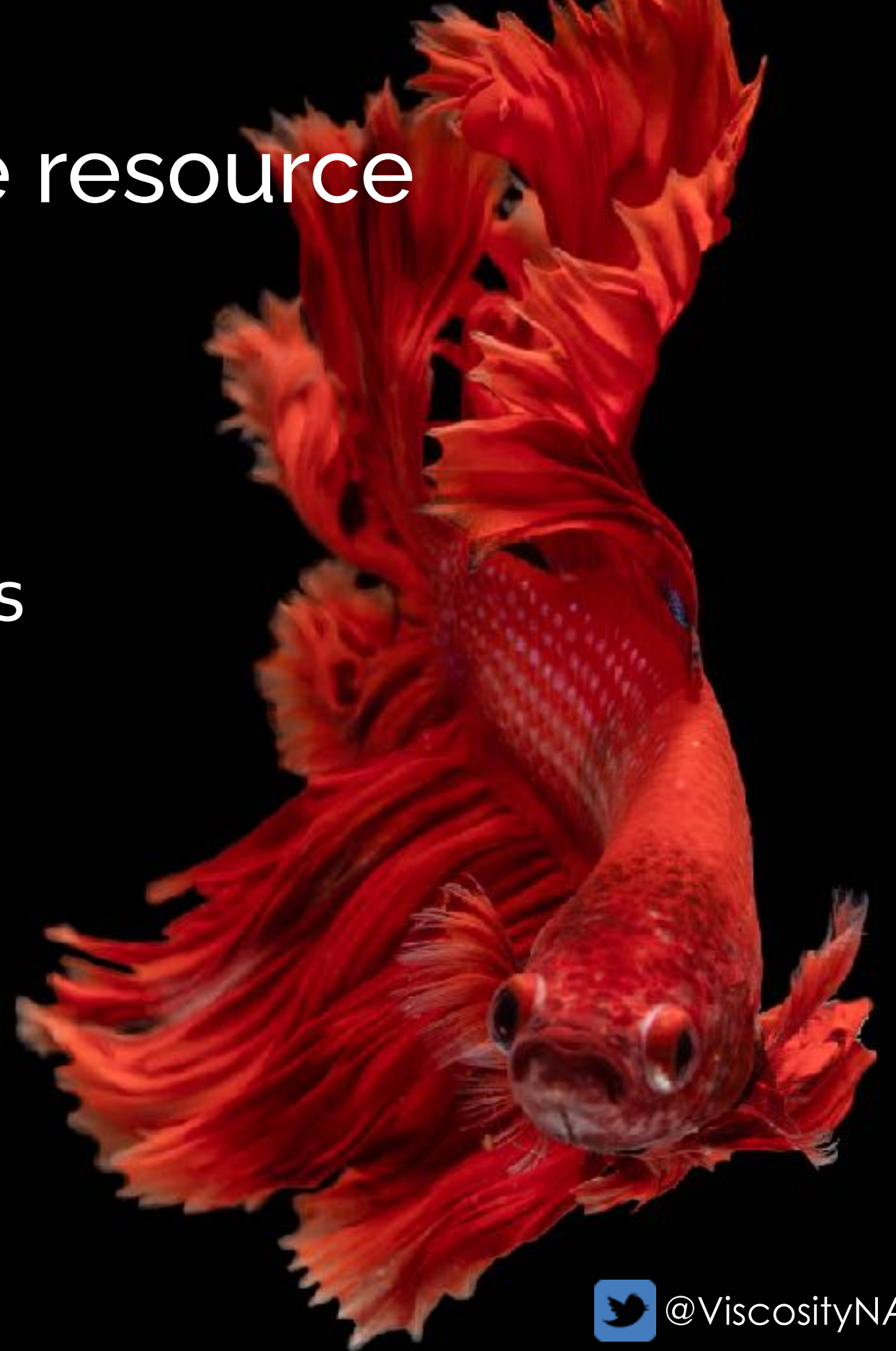
```
No changes. Your infrastructure matches the configuration.
```

```
Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
```

```
Apply complete! Resources: 0 added, 0 changed, 0 destroyed.
```


Add an Autonomous Database resource

- Add the ADB configuration in `main.tf`
- Add ADB variables to `variables.tf`
- Add ADB values to `terraform.tfvars`
- Add output variables to `output.tf`



main.tf

```
# Autonomous database resource
resource "oci_database_autonomous_database" "autonomous_db" {
  compartment_id = var.tenancy_ocid # (Creates in root)
  db_name        = var.db_name
  display_name   = var.display_name Name we're giving to
  db_version     = var.db_version the Terraform resource
  db_workload    = var.db_workload
  cpu_core_count = var.cpu_core_count
  data_storage_size_in_tbs = var.data_storage_size_in_tbs
  is_free_tier   = var.is_free_tier Values used to
  license_model  = var.license_model create the ADB
  admin_password = var.admin_password Information the provider
} needs to create an ADB
```


Add database variables to `variables.tf`

```
# Autonomous DB variables
```

```
variable "db_name"           { type = string }  
variable "display_name"     { type = string }  
variable "admin_password"   { type = string }
```



Variable definitions

Add database variables to `variables.tf`

Variable definition block

```
variable "db_version" {  
  type = string  
  default = "21c"      # Options are 19c, 21c  
}
```

```
variable "db_workload" {  
  type = string  
  default = "OLTP"     # Options are: OLTP, DW, AJD, APEX  
}
```

Set variable type

Assign a default value

Add database variables to `variables.tf`

```
variable "cpu_core_count" {  
  type = number  
  default = 1  
}  
  
variable "data_storage_size_in_tbs" {  
  type = number  
  default = 1  
}
```


Add database variables to `variables.tf`

```
variable "is_free_tier" {  
  type = string  
  default = "true"      # Must be false for AJD, APEX  
}  
  
variable "license_model" {  
  type = string  
  default = "LICENSE_INCLUDED"  
}
```


Add database values to terraform.tfvars

```
# Autonomous database variable values
```

```
db_name          = "ADB21C"  
display_name     = "ADB21C"  
admin_password   = "XXXXXXXXXXXXXXXXXXXXXXXXXXXX"
```

```
# Default overrides
```

```
#db_version      =  
#db_workload     =  
#cpu_core_count  =  
#data_storage_size_in_tbs =  
#is_free_tier    =  
#license_model   =
```

ADB values likely to
change for each DB

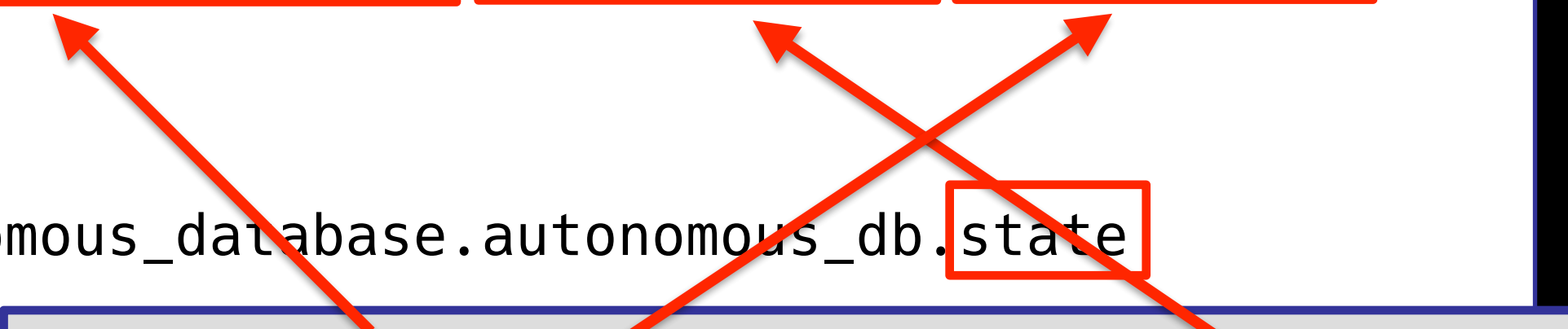
To override defaults,
un-comment the line
and set a value

outputs.tf

```
output "db_name" {  
  value = oci_database_autonomous_database.autonomous_db.display_name  
}
```

```
output "db_state" {  
  value = oci_database_autonomous_database.autonomous_db.state  
}
```

```
resource "oci_database_autonomous_database" "autonomous_db" {  
  compartment_id = var.tenancy_ocid  
  db_name        = var.db_name  
  display_name   = var.display_name  
  ...  
}
```



Create the database!

- Run:
terraform plan
terraform apply

terraform plan

```
> terraform plan
```

Terraform used the selected providers to generate the following execution plan.
Resource actions are indicated with the following symbols:

- + create

Terraform will perform the following actions:

```
# oci_database_autonomous_database.autonomous_db will be created
+ resource "oci_database_autonomous_database" "autonomous_db" {
  + actual_used_data_storage_size_in_tbs      = (known after apply)
  + admin_password                            = (sensitive value)
```

```
...
```


terraform plan (Continued)

...

Plan: 1 to add, 0 to change, 0 to destroy.

Changes to Outputs:

+ db_name = "ADB21C"

+ db_state = (known after apply)

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.

terraform apply

```
> terraform apply
```

```
...
```

```
Plan: 1 to add, 0 to change, 0 to destroy.
```

```
Changes to Outputs:
```

```
+ db_name = "ADB21C"
```

```
+ db_state = (known after apply)
```

```
Do you want to perform these actions?
```

```
Terraform will perform the actions described above.
```

```
Only 'yes' will be accepted to approve.
```

```
Enter a value: yes
```


terraform apply (Continued)

```
...
Enter a value: yes

oci_database_autonomous_database.autonomous_db: Creating...
oci_database_autonomous_database.autonomous_db: Still creating... [10s elapsed]
...
oci_database_autonomous_database.autonomous_db: Creation complete after 1m31s
Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Outputs:

```
db_name = "ADB21C"
```

```
db_state = "AVAILABLE"
```


Writing Terraform configurations with style

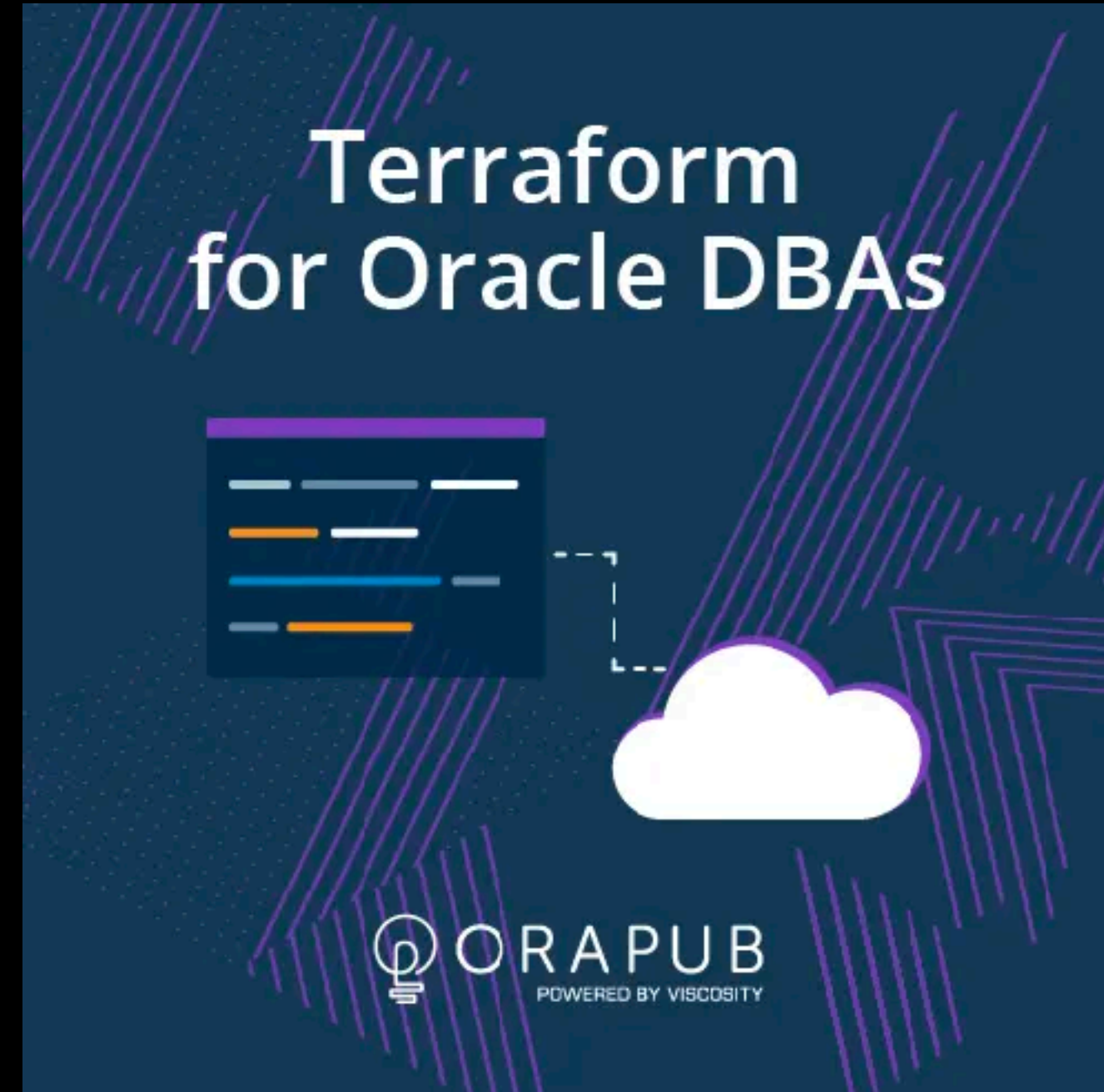
- Everything in a single file: `main.tf`
- Separate files for each resource:
 - More portable/reusable code
 - `compute.tf`
 - `vcn.tf`
 - `storage.tf`
 - `variables.tf`



Want more?

- *Terraform for Oracle DBAs* online class
- Three, two-hour sessions: Sept 1, 8, 15
- Use Terraform to build:
 - Automate Oracle DB web, object storage
 - Create VMs and compute resources
 - Provision interfaces with Storage Manager
 - Define and deploy configurations from a web page

Register: <https://www.orapub.com/lvc-terraform>



Questions?



oraclesean.com



[@oraclesean](https://twitter.com/oraclesean)



sean.scott@viscosityna.com



<https://github.com/oraclesean>



<https://www.linkedin.com/in/soscott>



Search "OracleSean" on YouTube



