Terraform on Oracle Cloud Infrastructure

A Primer for Database Administrators

www.viscosityna.com

@ViscosityNA

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

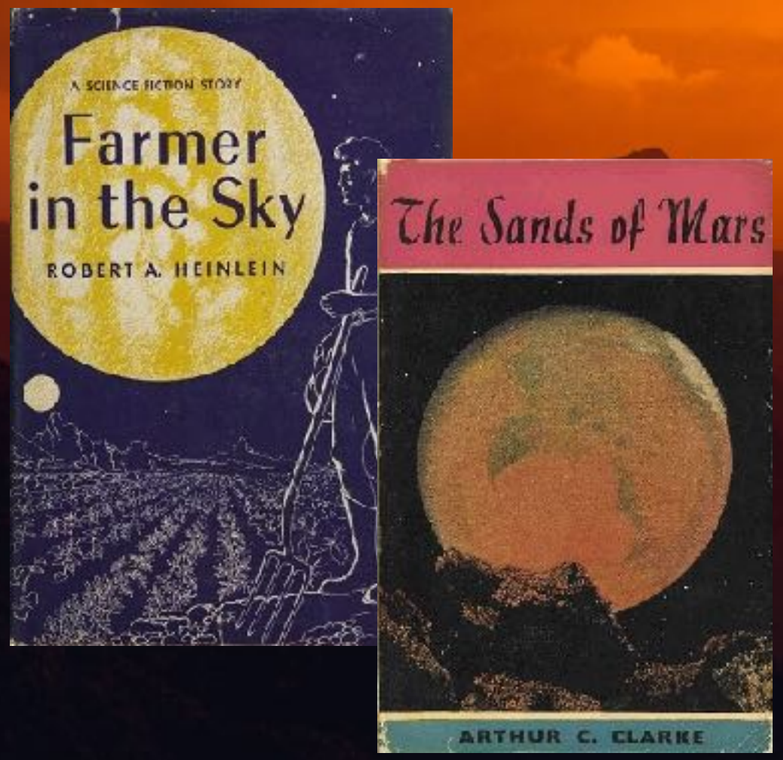
Containers and Docker ·:· Virtualization

ORACLE Certified Professional

ORACLE Certified Specialist

ter·ra·form *verb*

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



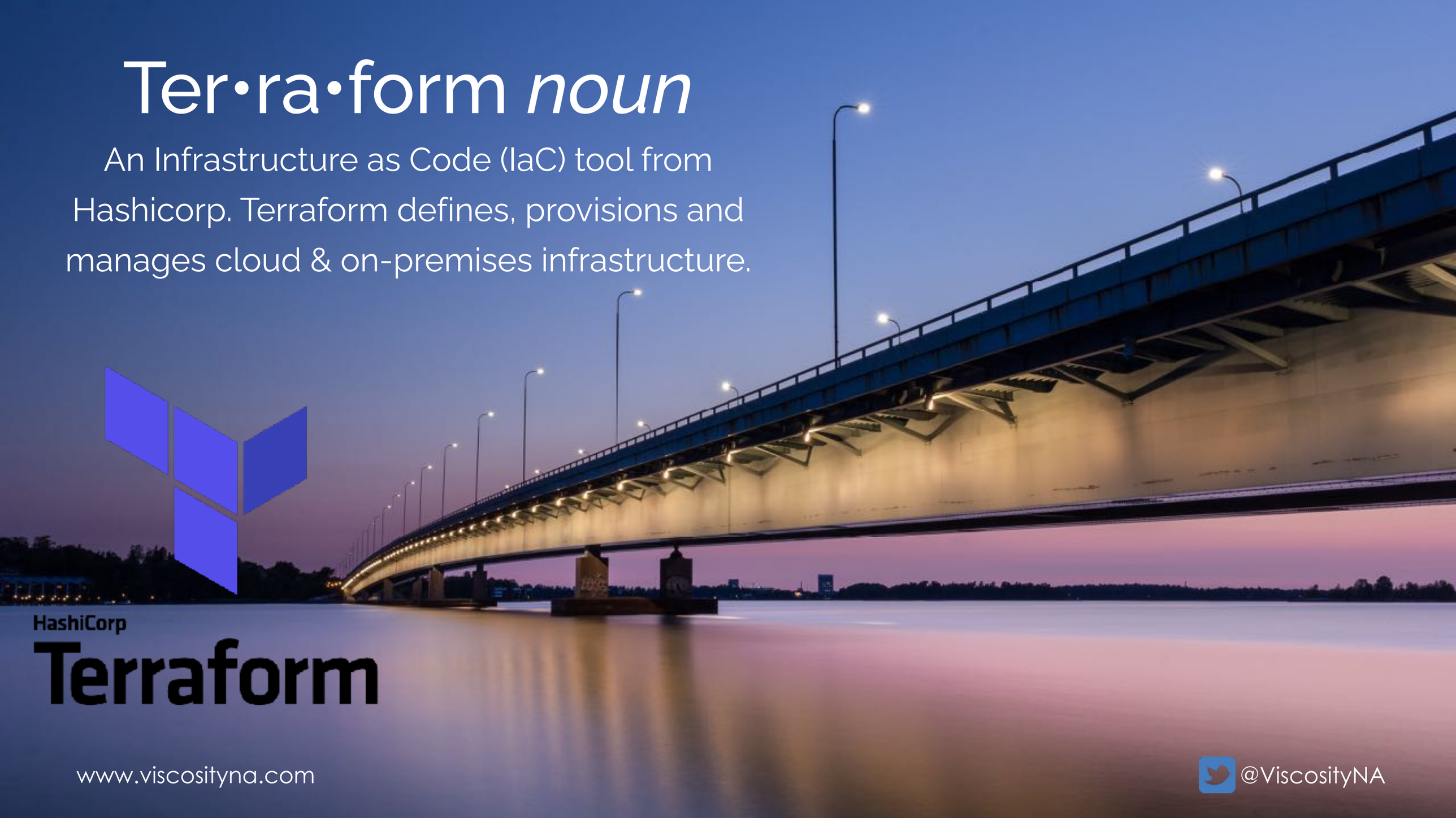
To transform an environment to support life

www.viscosityna.com

@ViscosityNA

Ter·ra·form *noun*

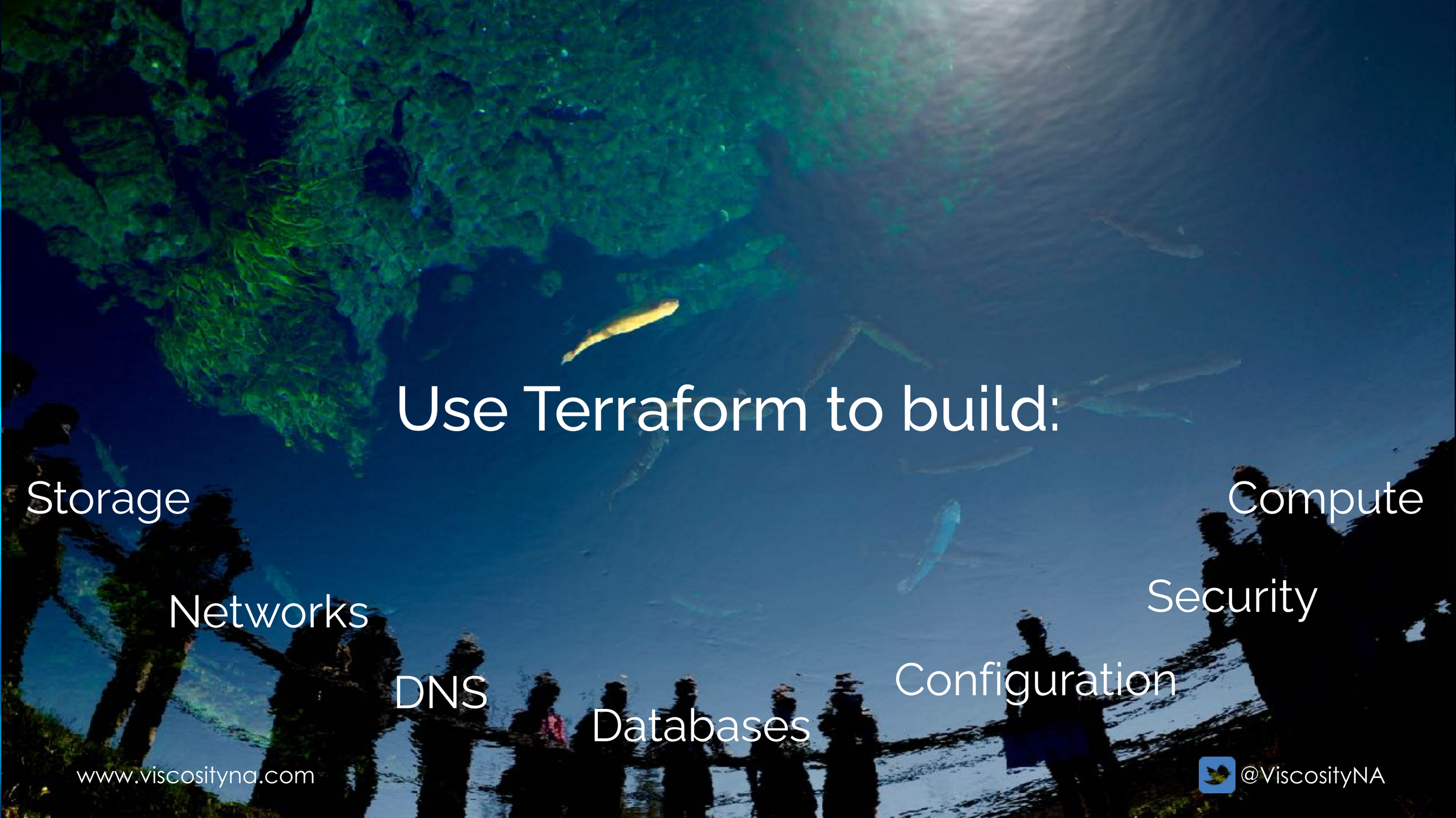
An Infrastructure as Code (IaC) tool from Hashicorp. Terraform defines, provisions and manages cloud & on-premises infrastructure.



HashiCorp
Terraform

www.viscosityna.com

@ViscosityNA


Use Terraform to build:
Storage
Compute
Networks
Security
DNS
Configuration
Databases
www.viscosityna.com
@ViscosityNA

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

- Get fish tank *a*, *p*ump *b*, *h*eater *c*...
- Assemble them per...
- Add *x* liters water...
- Add *y* grams salt...
- Set temperature to *z*˚C...
- Add *n* fish...

Scales poorly!

Imperative languages

Declarative languages

These "experts"are
Terraform providers

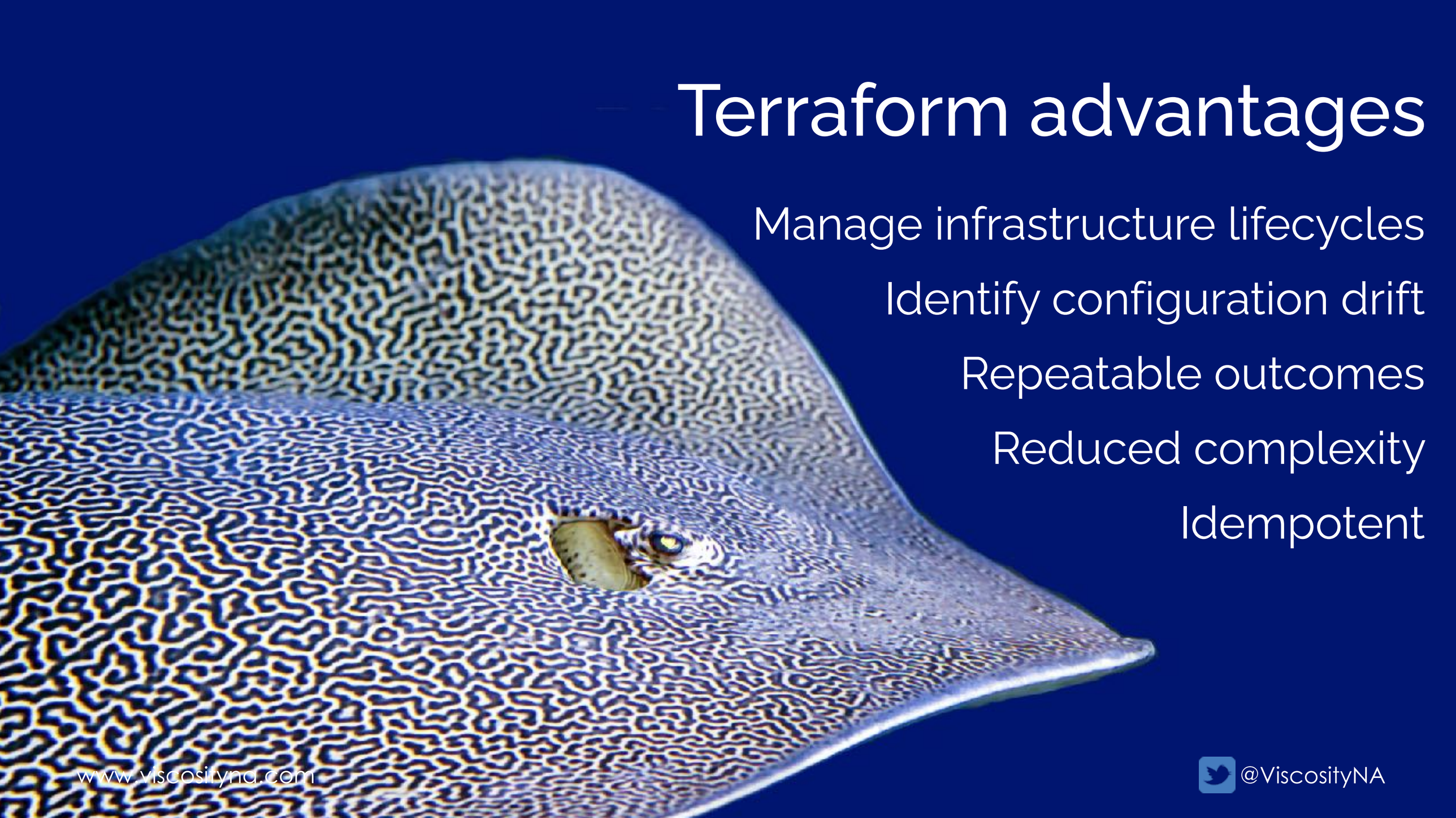
Tell the expert:
Scale and shape
Special needs, requirements
What you already have

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

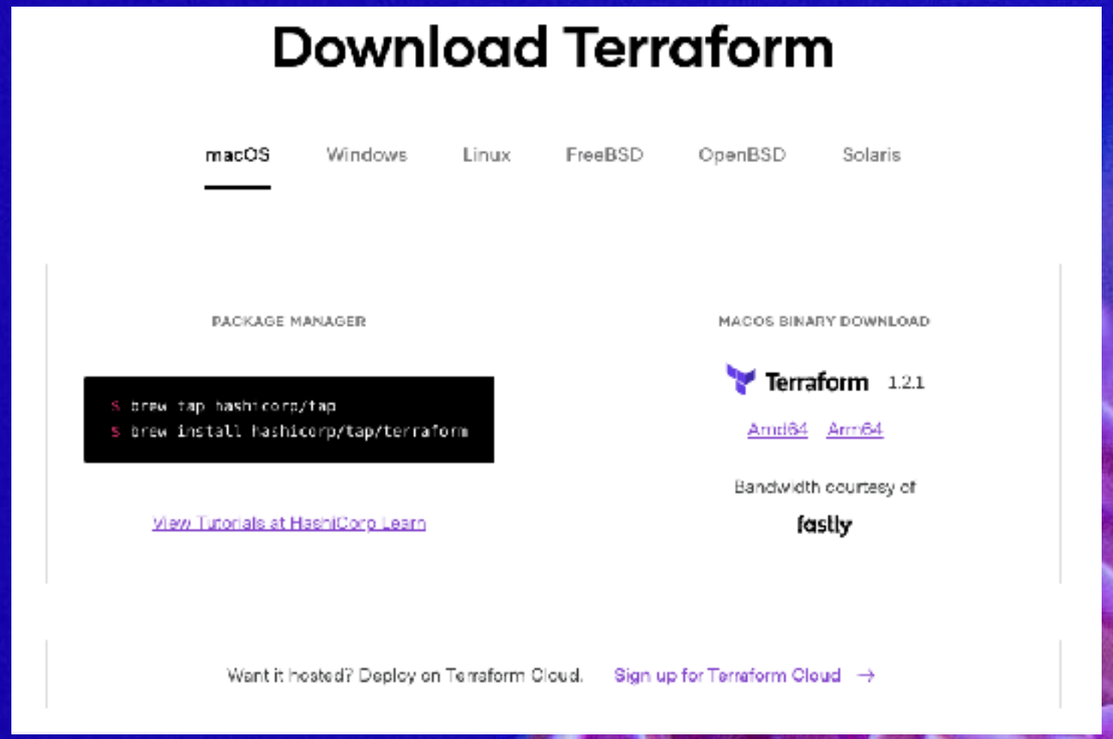
Repeatable outcomes

Reduced complexity

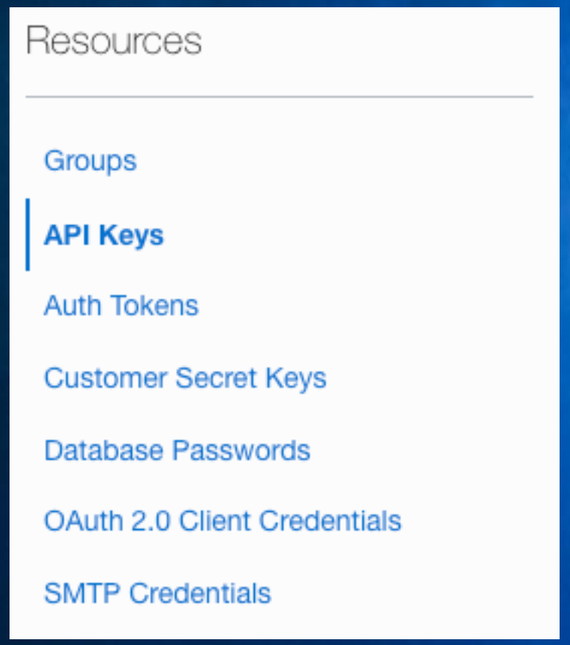
Idempotent

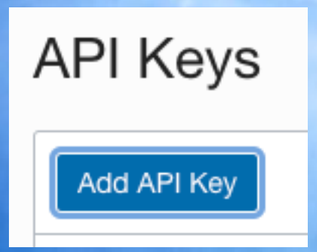
Get Terraform

https://www.terraform.io/downloads



Generate an OCI public key





Identity & Security → Users

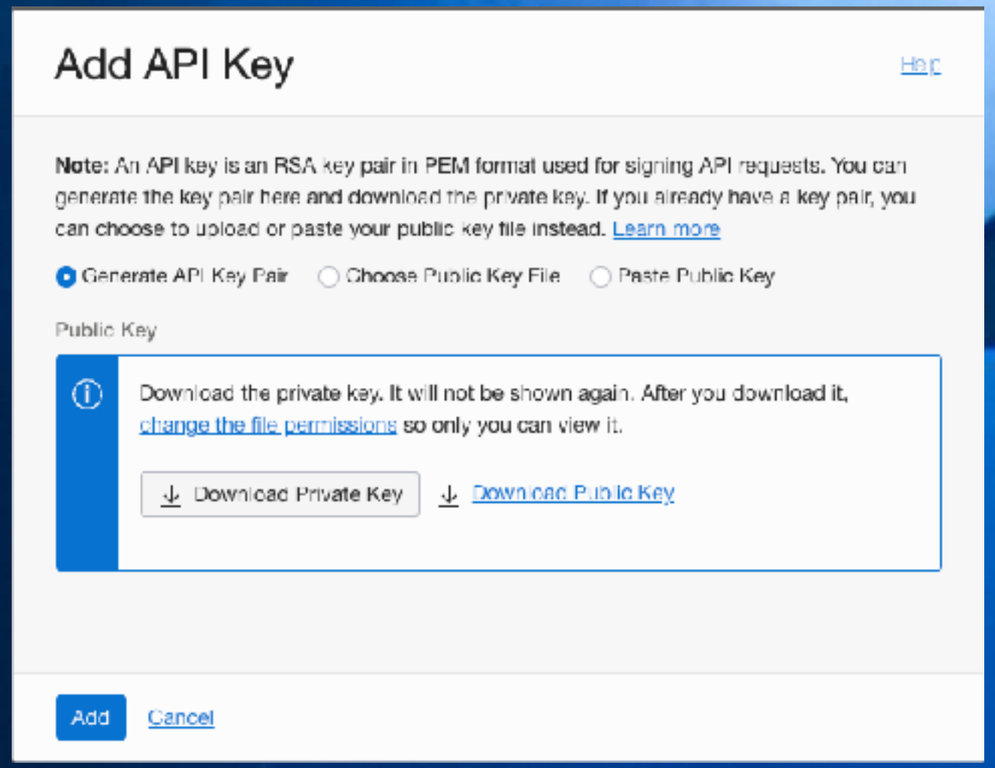
Choose a user to run Terraform

Select "API Keys" in the Resources menu

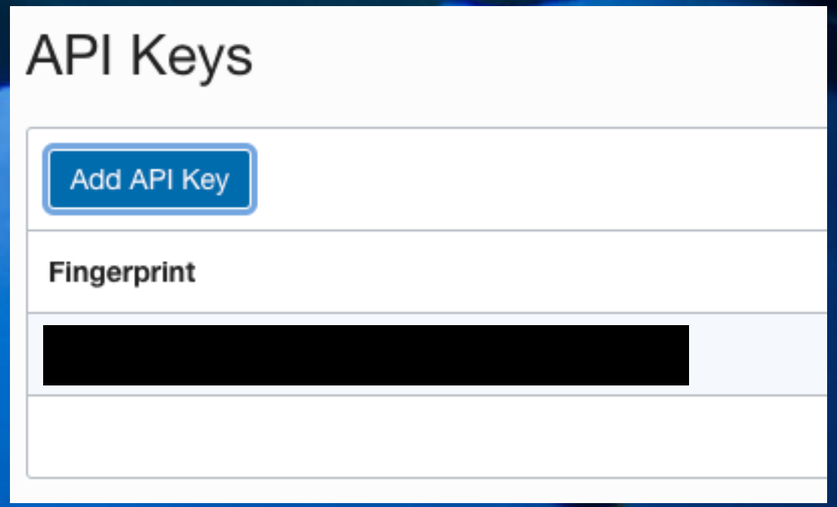
Click the "Add API Key" button

Generate an OCI public key (continued)

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

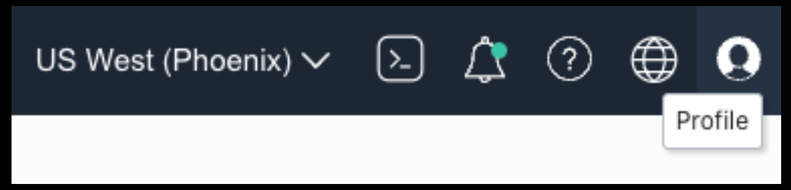


- Copy the key fingerprint

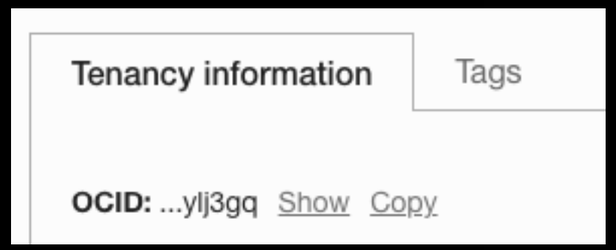


Get the `tenancy_ocid`

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

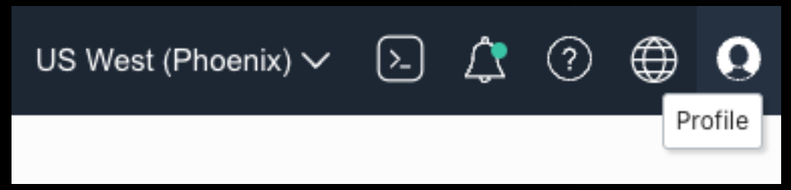


Use the "Copy" link under "Tenancy information"

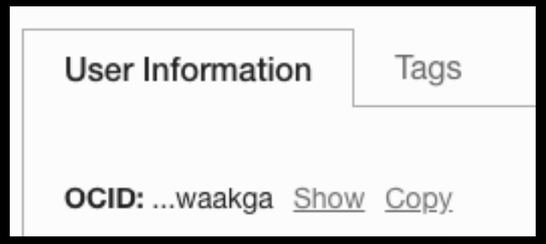


Get the `user_ocid`

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

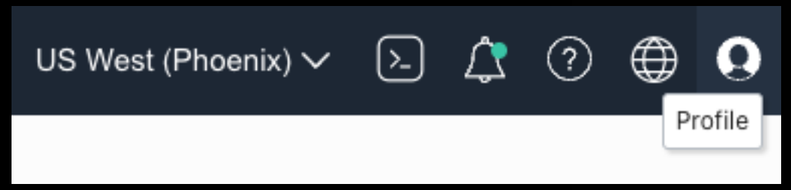


Use the "Copy" link under "User information"

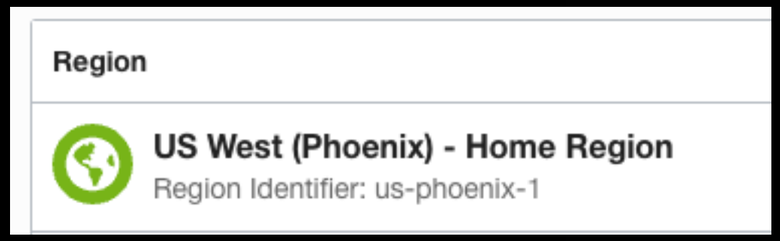


Get the Region Identifier

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

Terraform variables

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

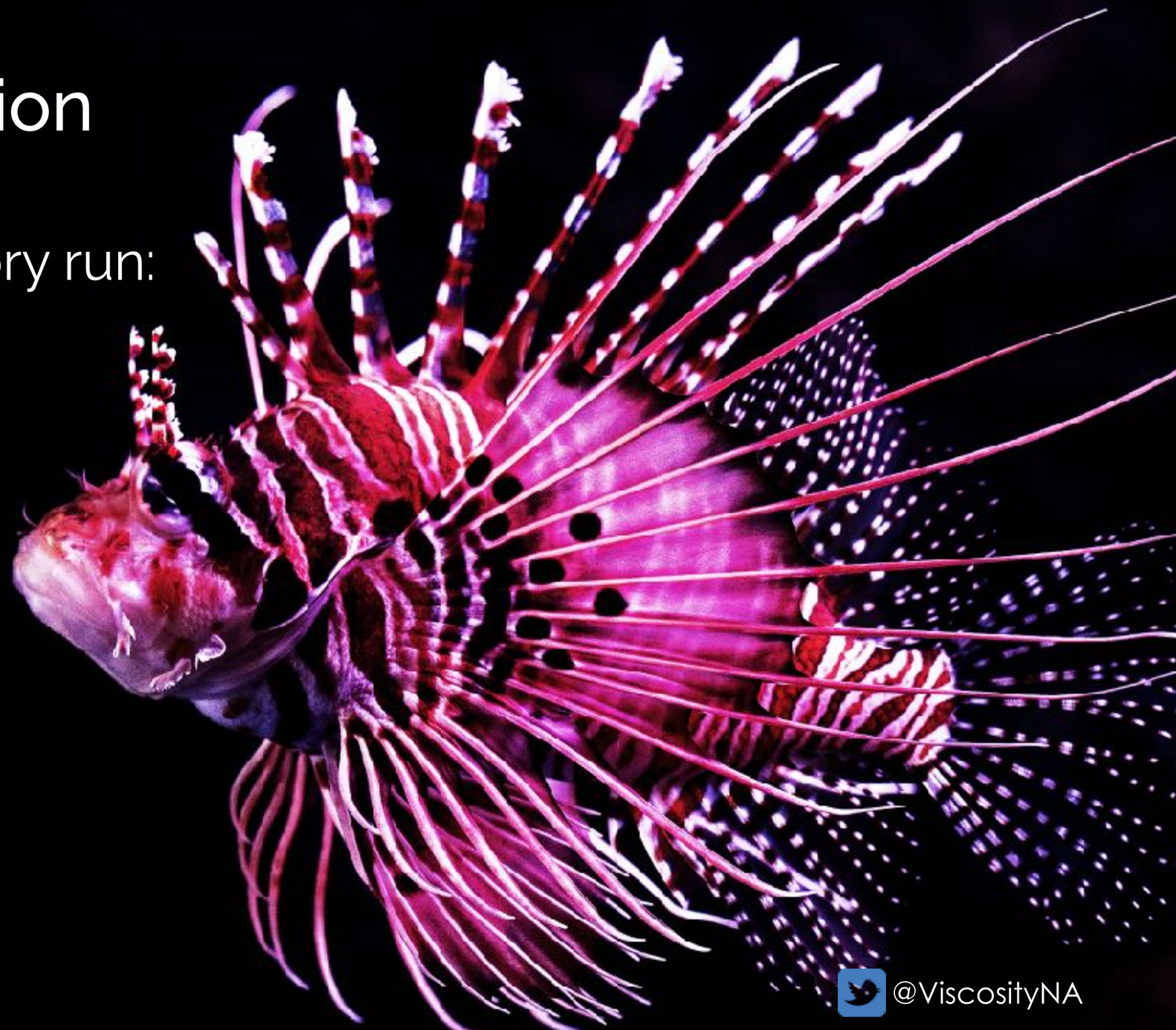
Same variables as defined in `variables.tf`

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.!

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

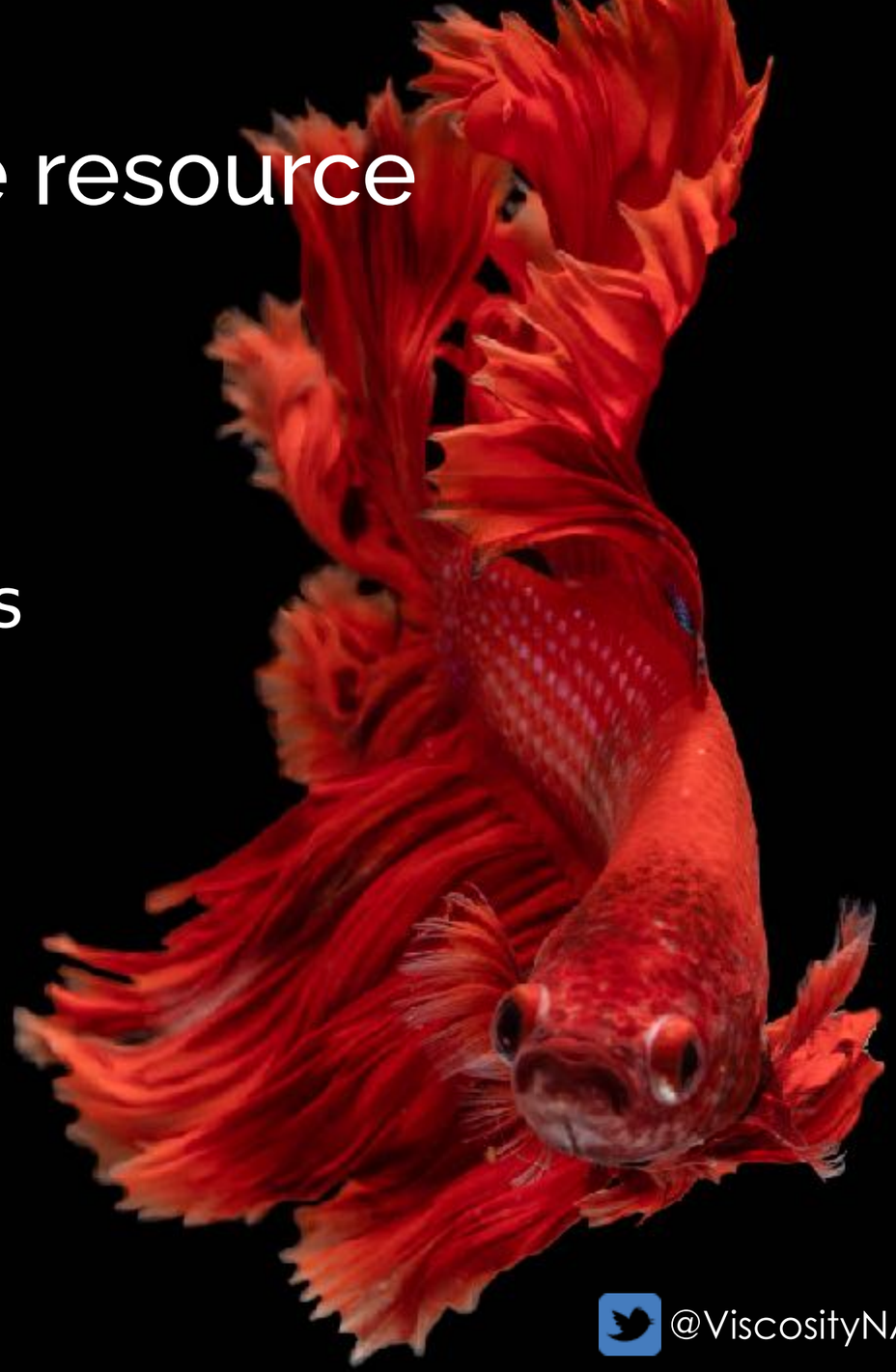
```
> terraform plan

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
  db_version              = var.db_version
  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
  license_model           = var.license_model
  admin_password          = var.admin_password
}
```

Type of resource

Name we're giving to the Terraform resource

Values used to create the ADB

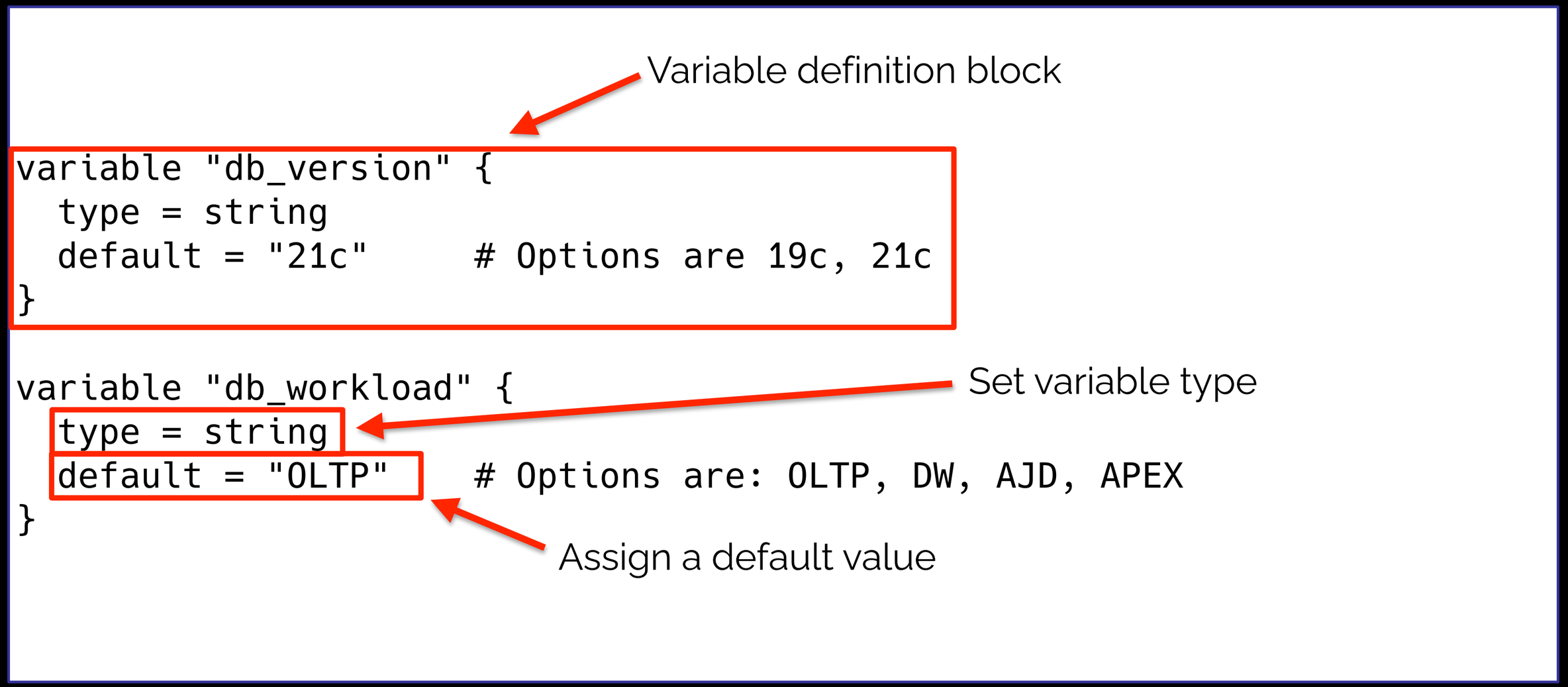
Information the provider needs to create an ADB

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`



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 = "XXXXXXXXXXXXXXXXXXXXXX"

# 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`

Questions?

- oraclesean.com
- @oraclesean
- sean.scott@viscosityna.com
- https://github.com/oraclesean
- https://www.linkedin.com/in/soscott
- Search "OracleSean" on YouTube