Lab: Variable Validation and Suppression

We may want to validate and possibly suppress and sensitive information defined within our variables.

- Task 1: Validate variables in a configuration block
- Task 2: More Validation Options
- Task 3: Suppress sensitive information
- Task 4: View the Terraform State File

Task 1: Validate variables in a configuration block

Create a new folder called `variable_validation` with a `variables.tf` configuration file:

```
variable "cloud" {
  type = string

  validation {
    condition     = contains(["aws", "azure", "gcp", "vmware"], lower(var.
       cloud))
    error_message = "You must use an approved cloud."
  }

  validation {
    condition     = lower(var.cloud) == var.cloud
    error_message = "The cloud name must not have capital letters."
  }
}
```

Perform a `terraform init` and `terraform plan`. Provide inputs that both meet and do not meet the validation conditions to see the behavior.

```
terraform plan -var cloud=aws
terraform plan -var cloud=alibabba
```

Task 2: More Validation Options

Add the following items to the `variables.tf`

```
variable "no_caps" {
    type = string
```


```
    validation {
        condition = lower(var.no_caps) == var.no_caps
        error_message = "Value must be in all lower case."
    }

}

variable "character_limit" {
    type = string

    validation {
        condition = length(var.character_limit) == 3
        error_message = "This variable must contain only 3 characters."
    }
}

variable "ip_address" {
    type = string

    validation {
        condition = can(regex("^(?:[0-9]{1,3}\\.){3}[0-9]{1,3}$", var.
           ip_address))
        error_message = "Must be an IP address of the form X.X.X.X."
    }
}
```

```
terraform plan -var cloud=aws -var no_caps=training
    -var ip_address=1.1.1.1  -var character_limit=rpt

terraform plan -var cloud=all -var no_caps=Training
   -var ip_address=1223.22.342.22  -var character_limit=ga
```

Task 3: Suppress sensitive information

Terraform allows us to mark variables as sensitive and suppress that information. Add the following configuration into your `main.tf`:

```
variable "phone_number" {
  type = string
  sensitive = true
  default = "867-5309"
}

locals {
  contact_info = {
      cloud = var.cloud
```


```
      department = var.no_caps
      cost_code = var.character_limit
      phone_number = var.phone_number
  }

  my_number = nonsensitive(var.phone_number)
}

output "cloud" {
  value = local.contact_info.cloud
}

output "department" {
  value = local.contact_info.department
}

output "cost_code" {
  value = local.contact_info.cost_code
}

output "phone_number" {
  value = local.contact_info.phone_number
}

output "my_number" {
  value = local.my_number
}
```

Execute a `terraform apply` with inline variables.

```
terraform apply -var cloud=aws -var no_caps=training
    -var ip_address=1.1.1.1 -var character_limit=rpt
```

You will notice that the output block errors as it needs to have the `sensitive = true` value set.

```
 Error: Output refers to sensitive values

   on variables.tf line 73:
   73: output "phone_number" {

 To reduce the risk of accidentally exporting sensitive data that was
    intended to be only internal, Terraform requires that any root module
    output containing
 sensitive data be explicitly marked as sensitive to confirm your intent.

 If you do intend to export this data, annotate the output value as
    sensitive by adding the following argument:
     sensitive = true
```


Update the output to set the `sensitive = true` attribute and rerun the apply.

```
output "phone_number" {
  sensitive = true
  value = local.contact_info.phone_number
}
```

```
terraform apply -var cloud=aws -var no_caps=training
    -var ip_address=1.1.1.1  -var character_limit=rpt
```

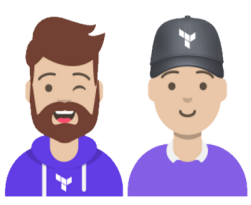
```
Outputs:

cloud = "aws"
cost_code = "rpt"
department = "training"
my_number = "867-5309"
phone_number = <sensitive>
```

Task 4: View the Terraform State File

Even though items are marked as sensitive within the Terraform configuration, they are stored within the Terraform state file. It is therefore critical to limit the access to the Terraform state file.

View the `terraform.tfstate` within your `variable_validation` diredtory.

```
{
  "version": 4,
  "terraform_version": "1.0.4",
  "serial": 3,
  "lineage": "5cfbccdd-b915-ee22-ea3c-17db83258332",
  "outputs": {
    "cloud": {
      "value": "aws",
      "type": "string"
    },
    "cost_code": {
      "value": "rpt",
      "type": "string"
    },
    "department": {
      "value": "training",
      "type": "string"
    },
    "my_number": {
      "value": "867-5309",
      "type": "string"
```

```
    },
    "phone_number": {
      "value": "867-5309",
      "type": "string",
      "sensitive": true
    }
  },
  "resources": []
}
```

TFC Integration

If you would like to see how variables are handled within Terraform Cloud, you can add the following files to your `variable_validation` directory.

`remote.tf`

```
terraform {
  backend "remote" {
    organization = "<<ORGANIZATION NAME>>"

    workspaces {
      name = "variable_validation"
    }
  }
}
```

`terraform.auto.tfvars`

```
cloud           = "aws"
no_caps         = "training"
ip_address      = "1.1.1.1"
character_limit = "rpt"
```

Run a `terraform init` to migrate state to the TFC workspace, followed by a `terraform apply` to show sensitive values with TFC.