20 July

Main.provider.tf

main.tf ke andar variable daalaa thaa

Types of variable – Primitive and Advanced

Primitive variable banana ke liye use kre – String, Boolean, Numbers

Now if we have to make many rgs so repetition is not good practice so list came.

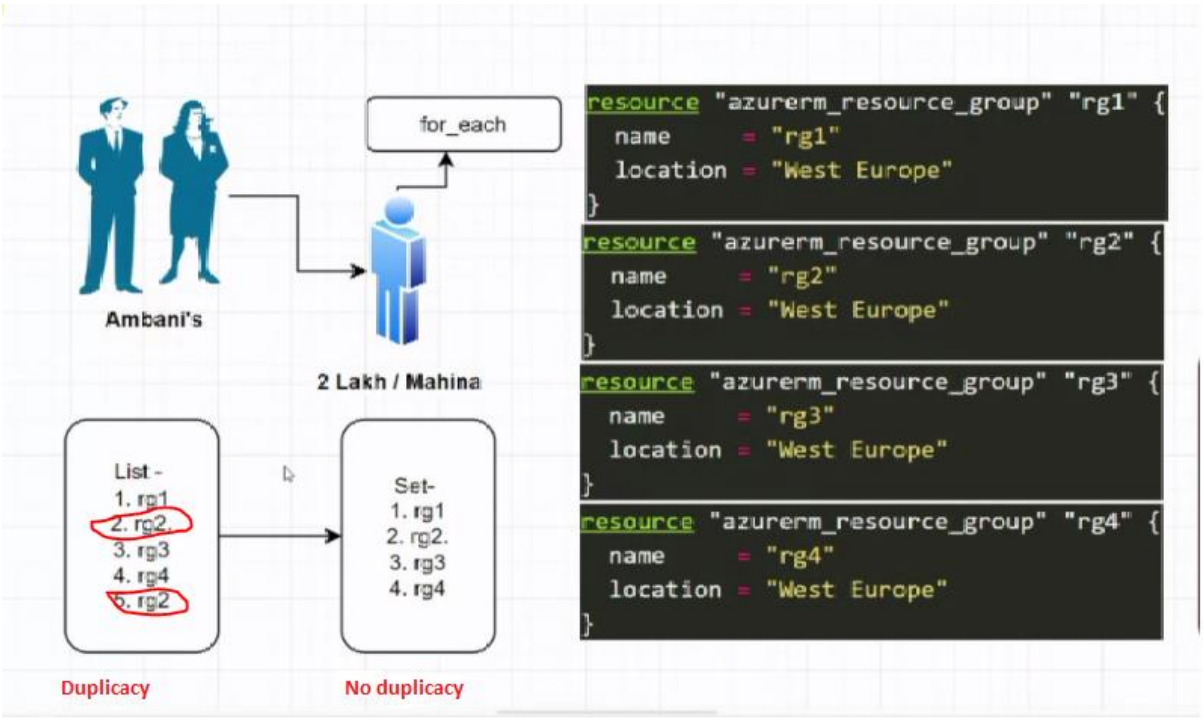
++

1) List – Collection of same kind of attributes (like name and location) is called List

2) List is useful – when we have to create **10 or multiple resource groups** only in one region like WestEurope. So to overcome this problem **map** used to come in market

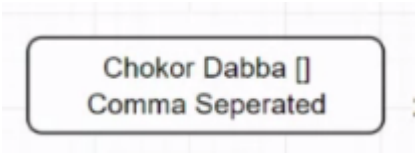
3) What is **For each** – For each creates all the resource groups that we provide in the list means it reads every element in the given list and then creates resource blocks for the given number of elements. For eg – rg1, rg2, rg3, rg4 as shown below

4) What is **set** – Set is used to remove duplicacy of the elements given in the list ie every element is unique. As 2 rgs with same name cannot be created at the same time, otherwise our code will not work.

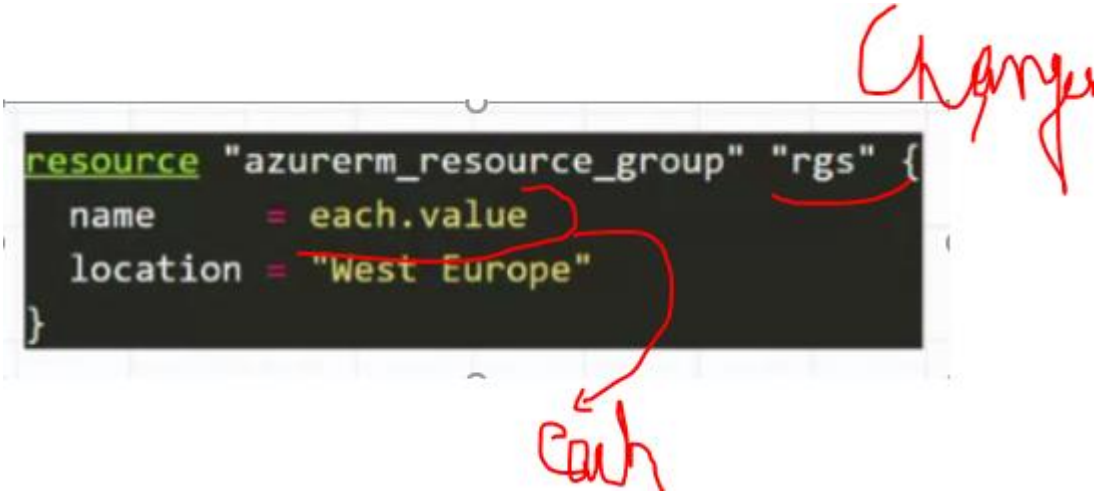


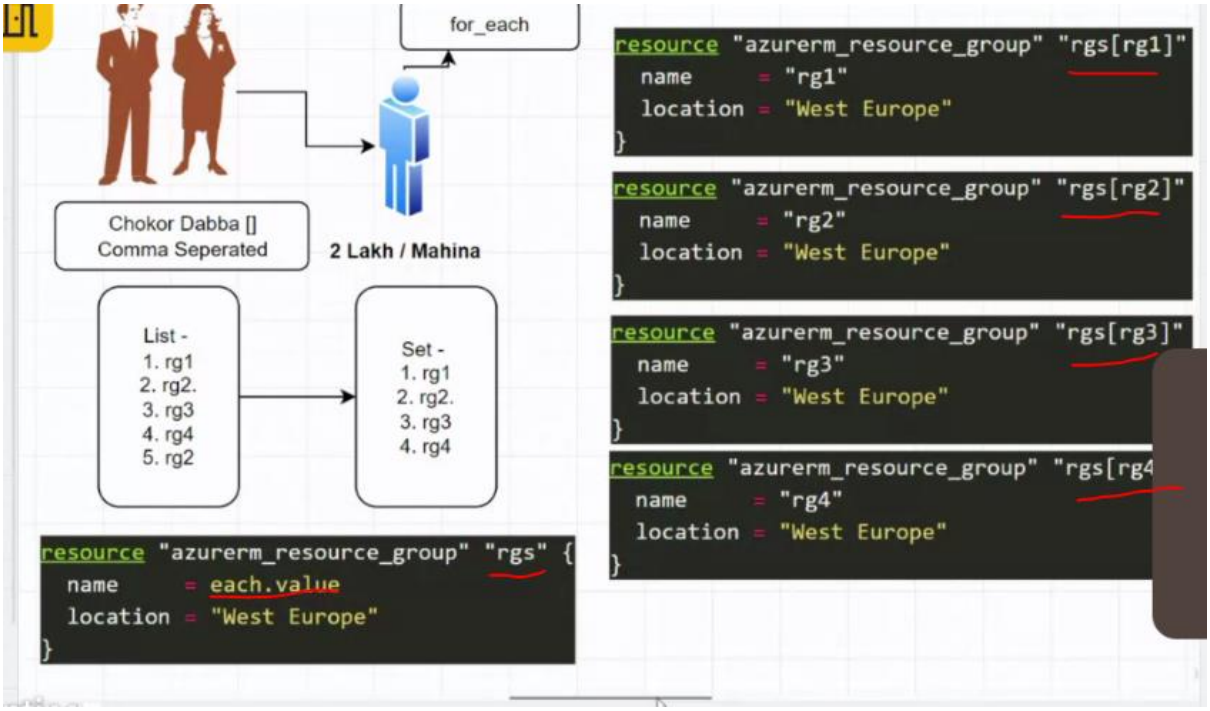
++

AGENDA – Pehle set se krke dekhte hai

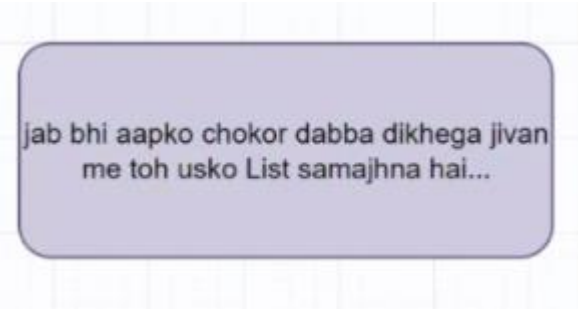


Chokor dibbe ke andar by default list hota hai as per terraform





4) **For each** always goes inside **resource block –> mug up**



As per above statement Whenever we will any array (chokor dabba) then terraform will consider it as list. But for considering it as set we have explicitly define the set().

5)

6) In [] content is considered as list but as per doc, terraform understands set of strings not list then conflict is there

7)

main.tf 1

main.tf > ...

```
variable "many_rgs" {
  type = set()
  default = ["rgkane1","rgkane2","rgkane3","rgkane4"]
}


resource "azurerm_resource_group" "rgblockkane" {
  for_each = var.many_rgs
  name     = each.value
  location = "West Europe"
}

```

8) **terraform init**

```
The set type constructor requires one argument specifying the element type.

Error: Invalid type specification

  on main.tf line 2, in variable "many_rgs":
   2:   type = set()

The set type constructor requires one argument specifying the element type.

PS C:\Batch 16\Devops 16\TerraformListForEach>
```

So above error means we have to define element type inside set ()

main.tf > resource "azurerm_resource_group" "rgblockkane" > location

```
variable "many_rgs" {
  type = set(string)
  default = ["rgkane1","rgkane2","rgkane3","rgkane4"]
}
```

terraform validate

terraform fmt

az login

terraform plan

```
  + resource "azurerm_resource_group" "rgblockkane" {
      + id       = (known after apply)
      + location = "westeurope"
      + name     = "rgkane3"
    }

  # azurerm_resource_group.rgblockkane["rgkane4"] will be created
  + resource "azurerm_resource_group" "rgblockkane" {
      + id       = (known after apply)
      + location = "westeurope"
      + name     = "rgkane4"
    }

Plan: 4 to add, 0 to change, 0 to destroy.
```

++

AGENDA – Agar set ki jagah list me pass kre to kya hogaa

1)

main.tf

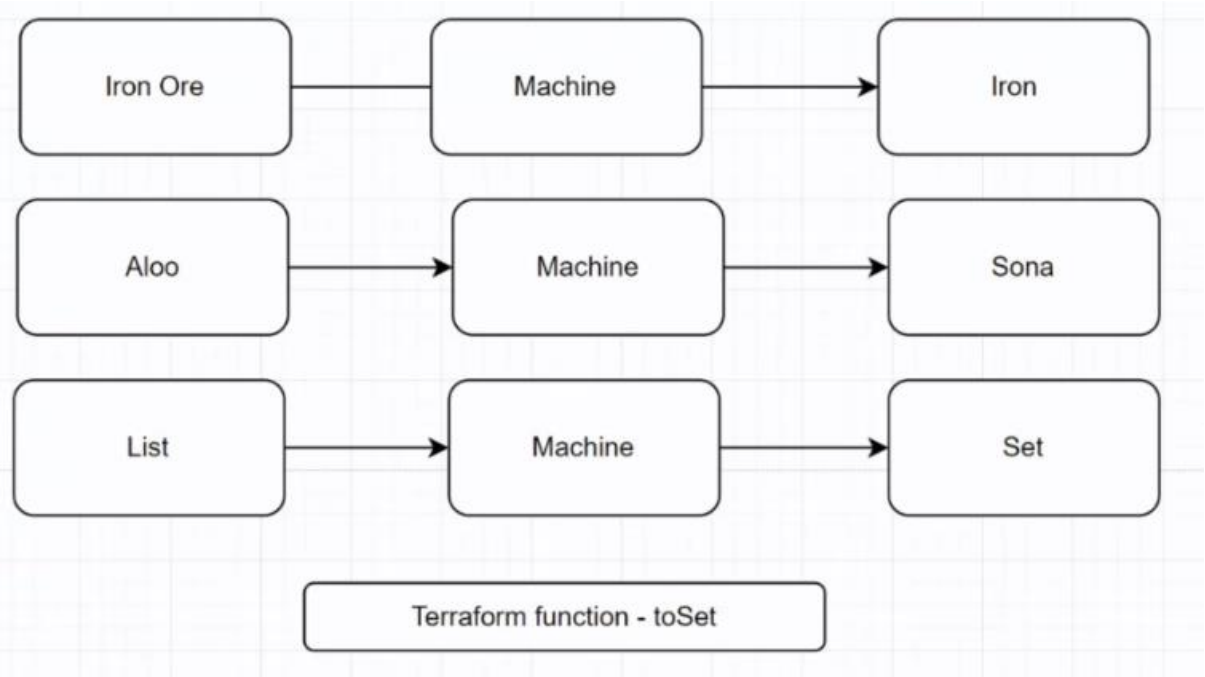
```
variable "many_rgs_banao" {        #declare a variable
  type    = list(string)
  default = ["rgkane1", "rgkane2", "rgkane3", "rgkane4"]  #assign a variable
}


resource "azurerm_resource_group" "rgblockkane" {
  for_each = var.many_rgs_banao          #use a variable
  name     = each.value
  location = "West Europe"
}
```

terraform plan

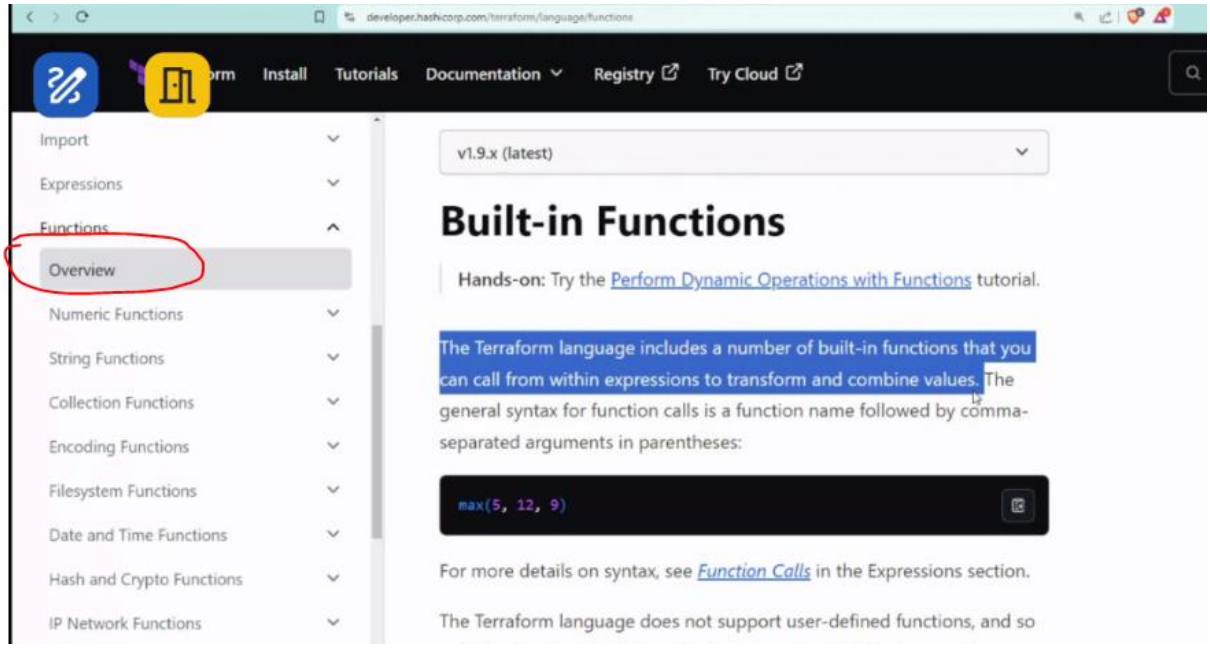
```
  string.

PS C:\Batch 16\Devops 16\TerraformListForEach> terraform plan

  Error: Invalid for_each argument

    on main.tf line 8, in resource "azurerm_resource_group" "rgblockkane":
     8:   for_each = var.many_rgs_banao
        |----------------
        | var.many_rgs_banao is a list of string

  The given "for_each" argument value is unsuitable: the "for_each" argument must be a map, or set of strings, and you have provided a value of type list of
  string.

PS C:\Batch 16\Devops 16\TerraformListForEach> clear
```

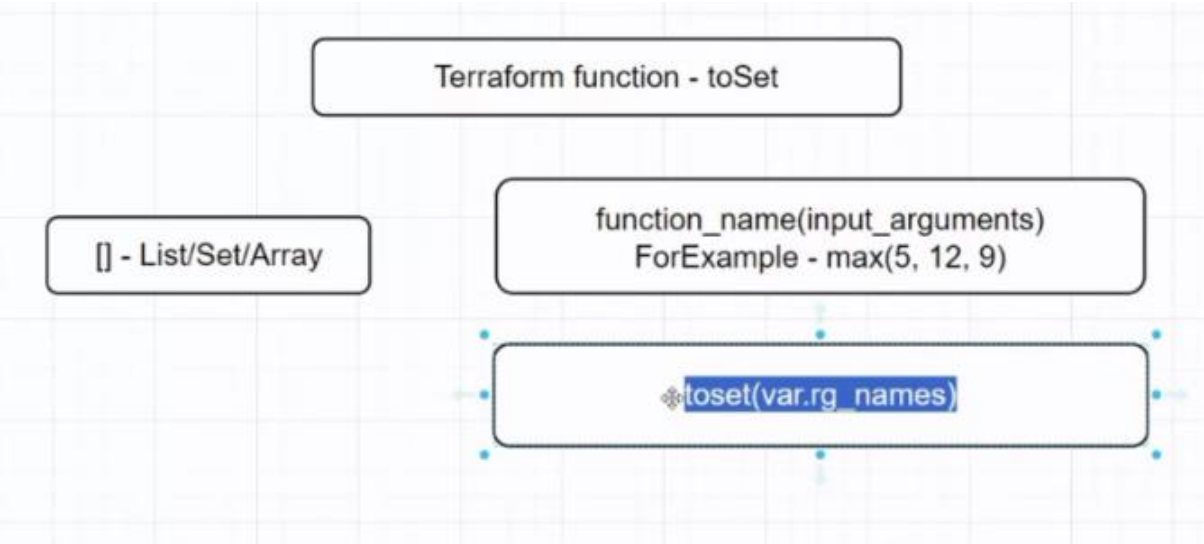
So while doing from list it is giving us the error

Now we will use terraform function



2) **SEARCH – Terraform functions**





3)

```
main.tf > ...
1   variable "many_rgs_banao" {
2     type    = list(string)
3     default = ["rgkane1", "rgkane2", "rgkane3", "rgkane4"]
4   }
5
6
7   resource "azurerm_resource_group" "rgblockkane" {
8     for_each = toset(var.many_rgs_banao)
9     name     = each.value
10    location = "West Europe"
11  }
12
13
14
```

4) For each and count – 2 types of loops are there in terraform and both are known as **Meta arguments**

5) Whatever we write below resource group are known as **meta arguments**

6) For each – Best method of coding

7) What do you understand by meta arguments in terraform?

Used for looping

++

DRAWBACKS OF LISTS

1) In list, terraform does not allow to create rgs like rg1, rg2, rg3, rg4 in different locations. So to overcome this problem we will use map

++