Terraform Data Sources – Detailed Teaching Module

A **data source in Terraform** is a way to **fetch and use information about existing resources** or infrastructure that Terraform itself may not manage.

- **It is read-only:** It does not create or modify infrastructure, only retrieves data.
- **Use case:** When you need details (like IDs, names, or configurations) of existing cloud resources to use in your Terraform configuration.

When Do We Use Data Sources?

- When you need to reference an **existing VPC, subnet, or resource group** that Terraform didn't create.
- When you want **dynamic information** (e.g., the latest OS AMI in AWS).
- When you want to avoid **hardcoding values** like IDs or names.

Example 1: AWS AMI Data Source

Scenario:
You want to create an EC2 instance, but you don't want to manually find the latest Amazon Linux 2 image (AMI) from your AWS account.

```
provider "aws" {
 region = "us-east-1"
}


# STEP 1: Data source to fetch latest Amazon Linux 2 AMI
data "aws_ami" "amazon_linux" {
 most_recent = true    # always get the latest image
 owners      = ["amazon"] # owned by AWS

 filter {
  name   = "name"
  values = ["amzn2-ami-hvm-*-x86_64-gp2"]
 }
```

```
}

# STEP 2: Create an EC2 instance using the data source
resource "aws_instance" "my_ec2" {
  ami           = data.aws_ami.amazon_linux.id
  instance_type = "t2.micro"
}

# STEP 3: Output the fetched AMI ID
output "latest_ami" {
  value = data.aws_ami.amazon_linux.id
}
```

Save the file as main.tf.

Run the commands:

```
terraform init
terraform plan
```

Observe how Terraform **fetches the AMI ID dynamically** in the plan output.

Example 2: AWS VPC and Subnets

Scenario:
You want to deploy an EC2 instance into the **default VPC**, but you need to find the **subnet IDs** dynamically.

```
provider "aws" {
  region = "us-east-1"
}

# STEP 1: Get the default VPC
data "aws_vpc" "default" {
  default = true
}
```

```
# STEP 2: Get all subnets in the default VPC
data "aws_subnet_ids" "default_subnets" {
  vpc_id = data.aws_vpc.default.id
}

# STEP 3: Launch EC2 in the first subnet
resource "aws_instance" "my_ec2" {
  ami           = "ami-0c02fb55956c7d316"  # static AMI for test
  instance_type = "t2.micro"
  subnet_id     = tolist(data.aws_subnet_ids.default_subnets.ids)[0]
}

output "first_subnet" {
  value = tolist(data.aws_subnet_ids.default_subnets.ids)[0]
}
```

Review Questions

1. What is the key difference between resource and data in Terraform?
2. Why should you use a data source instead of hardcoding IDs?
3. Can you create a resource with only a data source? Why or why not?
4. How do you reference a data source in another resource?