Lab: Working with Data Blocks

Cloud infrastructure, applications, and services emit data, which Terraform can query and act on using data sources. Terraform uses data sources to fetch information from cloud provider APIs, such as disk image IDs, or information about the rest of your infrastructure through the outputs of other Terraform configurations.

- Task 1: Query existing resources using a data block
- Task 2: Export attributes from a data lookup

Task 1: Query existing resources using a data block

As you develop Terraform code, you want to make sure the code is developed with reusability in mind. This often means that your code needs to query data in order to get specific attributes or values to deploy resources where needed. For example, if you manually created an S3 bucket in AWS, you might to query information about that bucket so you can use it throughout your configuration. In this case, you would require a data block in Terraform to grab information to be used. Note that in this case, we're going to query data that already exists in AWS, and not a resource that was created by Terraform itself.

In the AWS console, create a new S3 bucket to use for this lab. Just create a bucket with all of the defaults. Don't worry, empty S3 buckets do not incur any costs.

In your `main.tf` file, let's create a data block that retrieves information about our new S3 bucket:

```
data "aws_s3_bucket" "data_bucket" {
  bucket = "my-data-lookup-bucket-btk"
}
```

Now, let's use information from that data lookup to create a new IAM policy to permit access to our new S3 bucket. In your `main.tf` file, add the following code:

```
resource "aws_iam_policy" "policy" {
  name        = "data_bucket_policy"
  description = "Deny access to my bucket"
  policy = jsonencode({
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
```


```
                "s3:List*"
            ],
            "Resource": "${data.aws_s3_bucket.data_bucket.arn}"
        }
    ]
  })
}
```

Run a `terraform plan` to see the proposed changes. Notice how the new policy will be created and the resource in the policy is the ARN of our new S3 bucket. Go ahead and apply the configuration using a `terraform apply -auto-approve`.

Task 2: Export attributes from a data lookup

Now that we have a successful data lookup against our S3 bucket, let's take a look at the attributes that we can export. Browse to the Terraform AWS provider, click on S3 in the left navigation page, and click on `aws_s3_bucket` under Data Sources. (https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/s3_bucket)

Note all of the different attributes that are exported. These are attributes that we can use throughout our Terraform configuration. We've already used `arn`, but we could use other attributes as well.

In the `main.tf` file, add the following outputs so we can see some of the additional information:

```
output "data-bucket-arn" {
  value = data.aws_s3_bucket.data_bucket.arn
}

output "data-bucket-domain-name" {
  value = data.aws_s3_bucket.data_bucket.bucket_domain_name
}

output "data-bucket-region" {
  value = "The ${data.aws_s3_bucket.data_bucket.id} bucket is located
        in ${data.aws_s3_bucket.data_bucket.region}"
}
```

Run a `terraform apply -auto-approve` to see the new outputs. Remember that the data in these outputs originated from our data lookup that we started with in this lab.

Once you're done, feel free to delete the bucket, the data block, and the output blocks if you would like.