Terraform Code Bundle

Generated: 2025-09-26 16:36

Files Included:

- 0-storage.tf
- 1-main.tf
- 2-providers.tf
- 5-bigquery.tf
- 9-variables.tf
- 10-outputs.tf
- 15-Deployer-SA.tf
- 20-enable-APIs.tf
- 30-queries.tf
- terraform.tfvars

0-storage.tf

```
# Bucket (force_destroy so 'terraform destroy' removes objects too)
resource "google_storage_bucket" "csv_bucket" {
  name          = "bq-csv-${random_id.suffix.hex}"
  location      = var.gcs_location
  force_destroy = true

  depends_on = [google_project_service.services]
}

# Pick up all CSVs in ./data
locals {
  csv_files = fileset("${path.module}/data", "*.csv")
}

# Upload every CSV found in ./data
resource "google_storage_bucket_object" "csvs" {
  for_each = toset(local.csv_files)
  name     = each.value
  bucket   = google_storage_bucket.csv_bucket.name
  source   = "${path.module}/data/${each.value}"
}
```

1-main.tf

```
# terraform {
#   required_version = ">= 1.5.0"
#   required_providers {
#     google = {
#       source  = "hashicorp/google"
#       version = "~> 6.0"
#     }
#   }
# }

# # variable "project_id" { type = string }
# variable "region"     {
#      type = string
#      default = "us-central1"
# }
# variable "create_vm"  {
#      type = bool
#      default = true
# }
# variable "vm_count"   {
#      type = number
#      default = 1
# }

# provider "google" {
#   project = var.project_id
#   region  = var.region
# }




resource "google_compute_instance" "ubuntu_vm" {
  count        = var.create_vm ? var.vm_count : 0
  name         = "ubuntu-${count.index + 1}"
  machine_type = "e2-micro"
  zone         = "${var.region}-b"
  tags         = ["ssh"]

  boot_disk {
    initialize_params {
      # Ubuntu LTS image
      image = "projects/ubuntu-os-cloud/global/images/family/ubuntu-2404-lts"
      size  = 10
      type  = "pd-balanced"
    }
  }

  network_interface {
    network       = "default"
    access_config {} # ephemeral public IP (remove if not needed)
  }

  metadata_startup_script = <<-EOT
    #!/bin/bash
    set -euo pipefail
    apt-get update -y
    apt-get install -y curl jq
```

1-main.tf

```
    echo "Ubuntu VM ready."
  EOT
}
```

2-providers.tf

```
terraform {
  required_version = ">= 1.5.0"
  required_providers {
    google = {
      source  = "hashicorp/google"
      version = "~> 6.0"
    }
    random = {
      source  = "hashicorp/random"
      version = "~> 3.6"
    }
  }
}

provider "google" {
  project = var.project_id
  region  = var.region
}

# Enable required APIs
resource "google_project_service" "services" {
  for_each = toset([
    "bigquery.googleapis.com",
    "storage.googleapis.com"
  ])
  project             = var.project_id
  service             = each.key
  disable_on_destroy  = false
}

# Suffix to keep names unique
resource "random_id" "suffix" {
  byte_length = 3
}
```

5-bigquery.tf

```
# Dataset (delete contents on destroy)
resource "google_bigquery_dataset" "ds" {
  dataset_id                  = "demo_ds"
  location                    = var.bq_location
  delete_contents_on_destroy = true

  depends_on = [google_project_service.services]
}

# Load job: all CSVs uploaded above -> raw_all table
resource "google_bigquery_job" "load_raw_all" {
  job_id   = "load-raw-all-${random_id.suffix.hex}"
  location = var.bq_location

  # Wait until uploads are done
  depends_on = [google_storage_bucket_object.csvs]

  load {
    # Use wildcard so you can add/remove CSVs without changing Terraform
    source_uris = ["gs://${google_storage_bucket.csv_bucket.name}/*.csv"]

    destination_table {
      project_id = var.project_id
      dataset_id = google_bigquery_dataset.ds.dataset_id
      table_id   = "raw_all"
    }

    source_format         = "CSV"
    autodetect            = true
    skip_leading_rows     = 1      # set to 0 if your CSVs have no header
    write_disposition     = "WRITE_TRUNCATE"
    allow_quoted_newlines = true
  }
}

# Handy view to peek at data
resource "google_bigquery_table" "v_raw_preview" {
  dataset_id = google_bigquery_dataset.ds.dataset_id
  table_id   = "v_raw_preview"

  view {
    query          = "SELECT * FROM `${var.project_id}.${google_bigquery_dataset.ds.
dataset_id}.raw_all` LIMIT 100"
    use_legacy_sql = false
  }

  deletion_protection = false
}
```

9-variables.tf

```
variable "project_id"        { type = string }
variable "region"            {
     type = string
     default = "us-central1"
}
variable "gcs_location"      {
     type = string
     default = "US"
}
variable "bq_location"       {
     type = string
     default = "US"
}

# Optional: create an Ubuntu VM for testing (default: false)
variable "create_vm"         {
     type = bool
     default = false
}

# Optional: number of VMs if create_vm = true
variable "vm_count"          {
     type = number
     default = 1
}
```

10-outputs.tf

```
# output "vpc-id"        { value = module.hq-vpc.id }
# output "vpc-self-link" { value = module.hq-vpc.self-link }

# output "customer-service-vm-ip" { value = module.vm-customer-service.nat-ip }
# output "production-vm-ip"       { value = module.vm-production.nat-ip }
# output "finance-vm-ip"          { value = module.vm-finance.nat-ip }

output "bucket_name"   { value = google_storage_bucket.csv_bucket.name }
output "dataset_id"    { value = google_bigquery_dataset.ds.dataset_id }
output "tables"        { value = ["raw_all", "query1_result", "query2_result", "v_ra
w_preview (view)"] }
```

15-Deployer-SA.tf

```
resource "google_service_account" "tf" {
  account_id   = "assignment-12-472020"
  display_name = "Terraform Deployer"
}
```

20-enable-APIs.tf

```
locals {
  apis = [
    "serviceusage.googleapis.com",
    "cloudresourcemanager.googleapis.com",
    "iam.googleapis.com",
    "compute.googleapis.com",
    "sqladmin.googleapis.com",
    "servicenetworking.googleapis.com",
    "networkconnectivity.googleapis.com",
    "dns.googleapis.com",
    "secretmanager.googleapis.com",
    "logging.googleapis.com",
    "monitoring.googleapis.com",
    "iap.googleapis.com",
    # optional:
    # "datamigration.googleapis.com",
    # "pubsub.googleapis.com",
    # "run.googleapis.com",
    # "vpcaccess.googleapis.com",
    # "cloudkms.googleapis.com",
    # "bigquery.googleapis.com",
    # "bigqueryconnection.googleapis.com",
  ]
}

resource "google_project_service" "enable" {
  for_each = toset(local.apis)
  project  = var.project_id
  service  = each.key
}
```

30-queries.tf

```
# Query #1 -> writes to table query1_result
resource "google_bigquery_job" "query1" {
  job_id   = "query1-${random_id.suffix.hex}"
  location = var.bq_location

  depends_on = [google_bigquery_job.load_raw_all]

  query {
    query = <<-SQL
      -- Example Query 1 — edit this SQL
      SELECT *
      FROM `${var.project_id}.${google_bigquery_dataset.ds.dataset_id}.raw_all`
      WHERE TRUE
    SQL

    use_legacy_sql = false

    destination_table {
      project_id = var.project_id
      dataset_id = google_bigquery_dataset.ds.dataset_id
      table_id   = "query1_result"
    }
    write_disposition = "WRITE_TRUNCATE"
  }
}

# Query #2 -> writes to table query2_result
resource "google_bigquery_job" "query2" {
  job_id   = "query2-${random_id.suffix.hex}"
  location = var.bq_location

  depends_on = [google_bigquery_job.load_raw_all]

  query {
    query = <<-SQL
      -- Example Query 2 — edit this SQL
      SELECT COUNT(*) AS row_count
      FROM `${var.project_id}.${google_bigquery_dataset.ds.dataset_id}.raw_all`
    SQL

    use_legacy_sql = false

    destination_table {
      project_id = var.project_id
      dataset_id = google_bigquery_dataset.ds.dataset_id
      table_id   = "query2_result"
    }
    write_disposition = "WRITE_TRUNCATE"
  }
}

################################################################
########  test QUERY   ##########################################
#  SELECT a1, a2 * 10 WHERE a1 == "Buy" && a4.indexOf('oil') != -1 ORDER BY parseInt
(a2), a4 LIMIT 100
#
################################################################
```

terraform.tfvars

```
project_id = "assignment-12-472020"
region     = "us-east1"
# zone    = "us-east1-b"   # optional override
vpc_name   = "hq-vpc"
```