**Homework\_Lesson\_37**

Задание:

1. Создать конфигурацию для GCP Storage. Добавить модуль, который будет отвечать за доп. параметры бакета (имя, размер, файлы для закачки)

2. Используйте созданный модуль в основной конфигурации для развертывания инфраструктуры.

3. Настройте удаленное хранение tfstate в Terraform, используя GCP Storage в связке с CloudSQL

4. Перенесите свою существующую инфраструктуру на удаленное хранение стейта.

5. Создать в ручном режиме любой ресурс (лучше всего бакет либо ВМ) и импортируйте его в свою конфигурацию.

**Создадим бакет для хранения состояния Terraform (tfstate) и добавим создание Cloud SQL экземпляра**

gcpa4607@tms:~/HW37$ terraform apply

Terraform used the selected providers to generate the following execution plan. Resource actions are

indicated with the following symbols:

+ create

Terraform will perform the following actions:

# google\_sql\_database.appdb will be created

+ resource "google\_sql\_database" "appdb" {

+ charset = (known after apply)

+ collation = (known after apply)

+ deletion\_policy = "DELETE"

+ id = (known after apply)

+ instance = "tf-cloudsql-db"

+ name = "appdb"

+ project = (known after apply)

+ self\_link = (known after apply)

}

# google\_sql\_database\_instance.db will be created

+ resource "google\_sql\_database\_instance" "db" {

+ available\_maintenance\_versions = (known after apply)

+ connection\_name = (known after apply)

+ database\_version = "POSTGRES\_15"

+ deletion\_protection = true

+ dns\_name = (known after apply)

+ encryption\_key\_name = (known after apply)

+ first\_ip\_address = (known after apply)

+ id = (known after apply)

+ instance\_type = (known after apply)

+ ip\_address = (known after apply)

+ maintenance\_version = (known after apply)

+ master\_instance\_name = (known after apply)

+ name = "tf-cloudsql-db"

+ private\_ip\_address = (known after apply)

+ project = (known after apply)

+ psc\_service\_attachment\_link = (known after apply)

+ public\_ip\_address = (known after apply)

+ region = "europe-north1"

+ self\_link = (known after apply)

+ server\_ca\_cert = (known after apply)

+ service\_account\_email\_address = (known after apply)

+ replica\_configuration (known after apply)

+ settings {

+ activation\_policy = "ALWAYS"

+ availability\_type = "ZONAL"

+ connector\_enforcement = (known after apply)

+ disk\_autoresize = true

+ disk\_autoresize\_limit = 0

+ disk\_size = (known after apply)

+ disk\_type = "PD\_SSD"

+ pricing\_plan = "PER\_USE"

+ tier = "db-f1-micro"

+ user\_labels = (known after apply)

+ version = (known after apply)

+ backup\_configuration (known after apply)

+ ip\_configuration (known after apply)

+ location\_preference (known after apply)

}

}

# google\_sql\_user.usersvc will be created

+ resource "google\_sql\_user" "usersvc" {

+ host = (known after apply)

+ id = (known after apply)

+ instance = "tf-cloudsql-db"

+ name = "appuser"

+ password = (sensitive value)

+ project = (known after apply)

+ sql\_server\_user\_details = (known after apply)

}

# module.app\_bucket.google\_storage\_bucket.bucket will be created

+ resource "google\_storage\_bucket" "bucket" {

+ force\_destroy = true

+ id = (known after apply)

+ labels = (known after apply)

+ location = "EUROPE-NORTH1"

+ name = "my-unique-app-bucket-1234"

+ project = (known after apply)

+ public\_access\_prevention = (known after apply)

+ self\_link = (known after apply)

+ storage\_class = "STANDARD"

+ uniform\_bucket\_level\_access = (known after apply)

+ url = (known after apply)

+ soft\_delete\_policy (known after apply)

+ versioning (known after apply)

+ website (known after apply)

}

# module.app\_bucket.google\_storage\_bucket\_object.initial["welcome.txt"] will be created

+ resource "google\_storage\_bucket\_object" "initial" {

+ bucket = "my-unique-app-bucket-1234"

+ content = (sensitive value)

+ content\_type = (known after apply)

+ crc32c = (known after apply)

+ detect\_md5hash = "different hash"

+ id = (known after apply)

+ kms\_key\_name = (known after apply)

+ md5hash = (known after apply)

+ media\_link = (known after apply)

+ name = "welcome.txt"

+ output\_name = (known after apply)

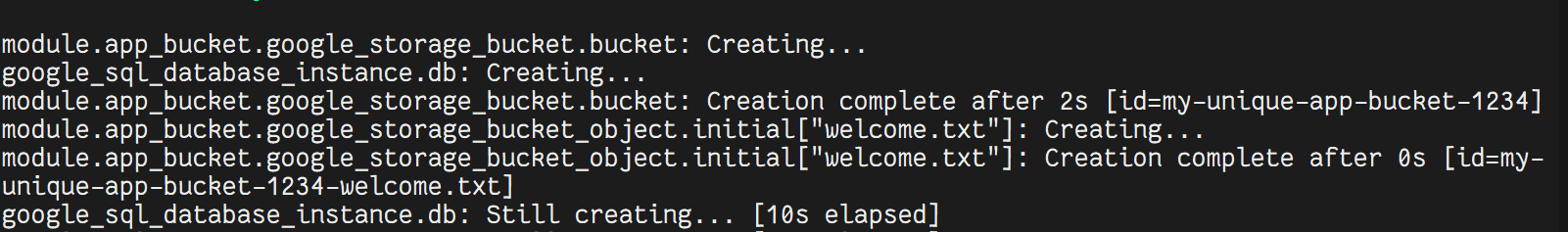
+ self\_link = (known after apply)

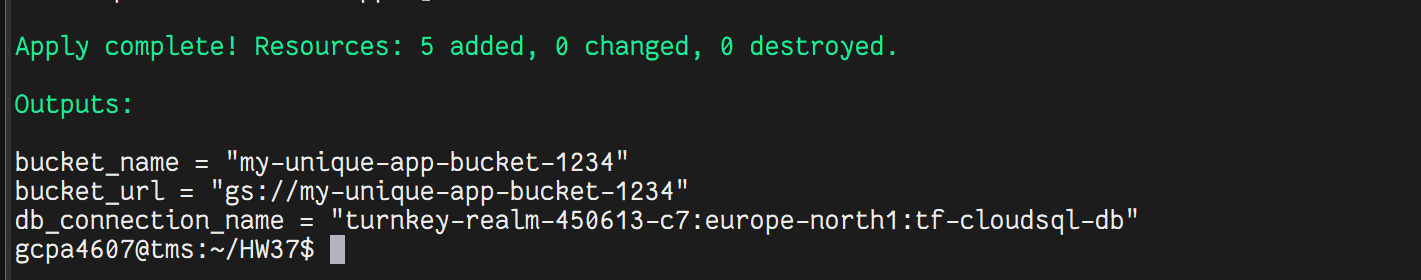
+ source = "welcome.txt"

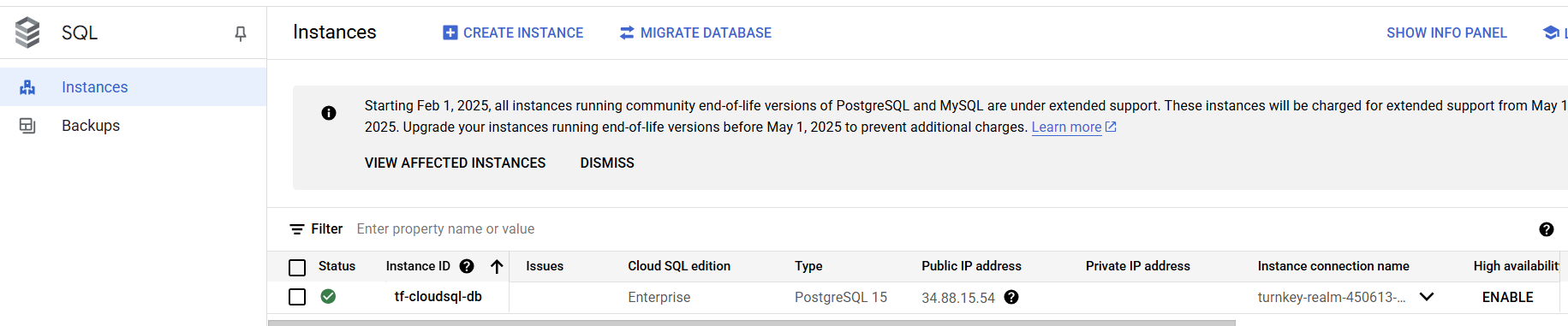
+ storage\_class = (known after apply)

}

Plan: 5 to add, 0 to change, 0 to destroy.



****

****

В процессе Terraform создал бакет и загрузил в него начальный welcome.txt