PROGRAMACION PARA DEVOPS

OBLIGATORIO MARZO 2025

DOCENTE: CRISTIAN HUELMO

ALUMNOS:

MAXIMO BRASSETI

RODRIGO CAMANO

INDICE

[2 - PRIMER SCRIPT DE PYTHON 3](#_Toc201790328)

[2.1 - ¿Qué hace el script? 3](#_Toc201790329)

[2.1.2 - Crea un bucket S3 3](#_Toc201790330)

[2.1.3 - Sube el archivo obli.sql 3](#_Toc201790331)

[2.1.4 - Busca el backup más reciente 3](#_Toc201790332)

[2.1.5 - Subir ese backup al bucket 3](#_Toc201790333)

[2.1.6 - Aclaración 3](#_Toc201790334)

[2.2 - Particularidades del entorno AWS (Academy) 3](#_Toc201790335)

[2.2.1 - Credenciales temporales 3](#_Toc201790336)

[2.2.2 - Nombre de bucket único 4](#_Toc201790337)

[2.3 - Importación de Librerías 4](#_Toc201790338)

[2.3.1 - Librerias 4](#_Toc201790339)

[2.3.2 - Explicación 4](#_Toc201790340)

[2.4 - Consideraciones antes de ejecutar 4](#_Toc201790341)

[2.4.1 - Credenciales AWS actualizadas 4](#_Toc201790342)

[2.4.2 -Archivo obli.sql existente 4](#_Toc201790343)

[2.4.3 - Existencia de backups en ~/Backups/ 5](#_Toc201790344)

[2.4.4 - Conexión a Internet activa 5](#_Toc201790345)

[2.4.5 - Permisos adecuados 5](#_Toc201790346)

[2.5 - Ejecución 5](#_Toc201790347)

[2.5.1 - Escriba en su consola 5](#_Toc201790348)

[2.5.2 - La salida le indicará 5](#_Toc201790349)

[2.6 - README 6](#_Toc201790350)

[3 - SEGUNDO SCRIPT DE PYTHON 9](#_Toc201790351)

[3.1 - ¿Qué hace el script? 9](#_Toc201790352)

[3.1.1 - Crea (o reutiliza) grupos de seguridad 9](#_Toc201790353)

[3.1.2 - Crea (o reutiliza) una instancia RDS MySQL 9](#_Toc201790354)

[3.1.3 - Crea (o reutiliza) una instancia EC2 9](#_Toc201790355)

[3.2 - Importación de librerías 9](#_Toc201790356)

[3.2.1 - Librerias 9](#_Toc201790357)

[3.2.2 - Explicación 9](#_Toc201790358)

[3.3 - Particularidades del entorno (AWS Academy) 10](#_Toc201790359)

[3.3.1 - Credenciales temporales 10](#_Toc201790360)

[3.3.2 - Archivo .env obligatorio 10](#_Toc201790361)

[3.3.3 - Bucket S3 con nombre único 10](#_Toc201790362)

[3.4 - Consideraciones antes de ejecutar 10](#_Toc201790363)

[3.4.1 - Verifique que su archivo obli.sql 10](#_Toc201790364)

[3.4.2 - Asegúrese de tener su archivo .env en el mismo directorio que este script. 10](#_Toc201790365)

[3.4.3 - Las credenciales en su equipo deben estar actualizadas para la sesión del laboratorio actual. 10](#_Toc201790366)

[3.5 - Ejecución 10](#_Toc201790367)

[3.5.1 - Escriba en su consola 10](#_Toc201790368)

[3.5.2 - La salida le indicará 10](#_Toc201790369)

[3.6 - Troubleshooting 11](#_Toc201790370)

[3.6.1 - Acceso Denegado 11](#_Toc201790371)

[3.6.2 - Bucket Existente 11](#_Toc201790372)

[3.6.3 - No se encuentra el archivo .sql 11](#_Toc201790373)

[3.7 - Buenas prácticas 11](#_Toc201790374)

[3.7.1 - No incluya claves o contraseñas directamente en el código. 11](#_Toc201790375)

[3.7.2 - Use nombres únicos para los recursos. 11](#_Toc201790376)

[3.7.3 - Elimine recursos que ya no utilice para evitar costos o conflictos. 11](#_Toc201790377)

[3.7.4 - Agregue logs o un sistema de manejo de errores más robusto si planea escalar este proyecto. 11](#_Toc201790378)

[3.8 - Archivos involucrados 11](#_Toc201790379)

[3.8.1 – PARTE2PYHTON.py: Script principal de automatización 11](#_Toc201790380)

[3.8.2 - .env: Variables sensibles y rutas 11](#_Toc201790381)

[3.8.3 - obli.sql: Base de datos que será restaurado en MySql 11](#_Toc201790382)

[3.8.4 - Directorio Backup: Adentro contiene el Backup 11](#_Toc201790383)

[3.9 - README 12](#_Toc201790384)

# 2 - PRIMER SCRIPT DE PYTHON

## 2.1 - ¿Qué hace el script?

Este script automatiza tareas iniciales de respaldo y almacenamiento en la nube mediante AWS S3. Su función principal es:

### 2.1.2 - Crea un bucket S3

(si no existe) con el nombre especificado.

### 2.1.3 - Sube el archivo obli.sql

Lo sube al bucket, ubicado previamente en el directorio del usuario (~/obli.sql).

### 2.1.4 - Busca el backup más reciente

En formato .tar.gz dentro de la carpeta ~/Backups, que siga el patrón backupSetUID\_\*.tar.gz.

### 2.1.5 - Subir ese backup al bucket

Sube con un nombre dinámico tipo Log\_dd-mm-YYYY, facilitando la trazabilidad por fecha.

### 2.1.6 - Aclaración

Este script es la **primera parte del flujo completo de automatización**, y su correcta ejecución es **requisito previo** para que otro script posterior pueda restaurar la base de datos en una instancia RDS utilizando este archivo.

## 2.2 - Particularidades del entorno AWS (Academy)

Este script fue diseñado para ejecutarse dentro de **laboratorios de AWS Academy**, que presentan ciertas particularidades importantes:

### 2.2.1 - Credenciales temporales

* Las claves de acceso (Access Key, Secret Key) que se usan para interactuar con AWS **expiran automáticamente cada 4 horas** o cada vez que se reinicia el entorno de laboratorio.
* Es **obligatorio actualizar las credenciales locales** antes de ejecutar el script. Si no lo hace, la creación de bucket o la carga de archivos fallará por falta de permisos.

### 2.2.2 - Nombre de bucket único

* AWS exige que los nombres de buckets S3 sean **únicos a nivel global**.
* Si otro alumno o el docente ya utilizó el mismo nombre, debe cambiar el valor de bucket\_name en el script por uno personalizado

## 2.3 - Importación de Librerías

### 2.3.1 - Librerias

* import boto3
* from datetime import datetime
* import os
* import glob

### 2.3.2 - Explicación

* **boto3:** Es la biblioteca oficial de AWS para Python. Se utiliza para interactuar con servicios de AWS, en este caso específicamente **S3**, donde se suben los archivos.
* **datetime:** Permite trabajar con fechas y horas. Aquí se usa para generar un nombre de log dinámico basado en la fecha actual, lo cual facilita el seguimiento de cargas realizadas.
* **os:** Permite interactuar con el sistema operativo, accediendo a rutas como el directorio del usuario (~) o verificando si existen archivos antes de subirlos.
* **glob:** Se utiliza para buscar archivos que coincidan con un patrón, en este caso backupSetUID\_\*.tar.gz. Sirve para encontrar automáticamente el backup más reciente que será subido a S3.

## 2.4 - Consideraciones antes de ejecutar

### 2.4.1 - Credenciales AWS actualizadas

* Asegúrese de tener actualizadas las credenciales AWS de su sesión activa del laboratorio (verifique usando aws configure list si lo desea).

### 2.4.2 -Archivo obli.sql existente

* El archivo obli.sql debe estar ubicado en el directorio raíz del usuario (~/obli.sql).
* Este archivo es fundamental para la restauración de la base de datos más adelante.

### 2.4.3 - Existencia de backups en ~/Backups/

* El script buscará automáticamente el backup más reciente que coincida con el patrón backupSetUID\_\*.tar.gz.
* Si no existen archivos con ese nombre, el script no podrá continuar.

### 2.4.4 - Conexión a Internet activa

* Para interactuar con AWS y subir archivos a S3, se requiere una conexión a Internet estable.

### 2.4.5 - Permisos adecuados

* El usuario que ejecuta el script debe tener permisos suficientes sobre AWS S3 en el laboratorio.
* Si obtiene errores tipo “AccessDenied” al subir archivos o crear buckets, revise sus credenciales y rol asignado.

## 2.5 - Ejecución

### 2.5.1 - Escriba en su consola

**python3 parte1python.py**

### 2.5.2 - La salida le indicará

* Confirmación de creación del bucket o su existencia
* Subida exitosa del archivo obli.sql
* Identificación y carga del backup más reciente
* Logs con nombres y errores si los hubiera

# 2.6 - README

# Script de Backup y Carga a Amazon S3

Este script en Python automatiza la creación de un bucket en Amazon S3 y la carga de dos archivos importantes:

1. Un archivo de respaldo SQL (`obli.sql`)

2. Un archivo `.tar.gz` de backups creado previamente, con nombre que comienza con `backupSetUID\_`

Es la primera parte del flujo de trabajo completo, que luego será utilizado por otro script que crea infraestructura en AWS y restaura este backup en una instancia de base de datos MySQL.

---

# Requisitos previos

Antes de ejecutar este script, se debe contar con lo siguiente:

1- AWS CLI configurado con credenciales vigentes del laboratorio actual de AWS Academy.

2- Haber generado previamente un backup `.tar.gz` dentro del directorio:

`~/Backups/`

3- Tener disponible el archivo `obli.sql` en el directorio del usuario: `~/obli.sql`

---

# ¿Qué hace el script?

1. Crea un bucket S3 en la región `us-east-1`.

- Si el bucket ya existe y pertenece al usuario, continúa sin error.

2. Carga el archivo `obli.sql` ubicado en el home del usuario al bucket.

3. Busca el backup más reciente en `~/Backups/` cuyo nombre comience con `backupSetUID\_` y termine en `.tar.gz`.

4. Sube dicho backup al bucket con un nombre dinámico tipo `Log\_dd-mm-YYYY`.

---

# Consideraciones importantes

Credenciales temporales:

- Las credenciales provistas por AWS Academy expiran cada 4 horas o al reiniciar el entorno.

- Asegúrese de que estén actualizadas antes de ejecutar el script.

Archivo `obli.sql` obligatorio:

- Este archivo será restaurado posteriormente en una instancia de base de datos.

- Debe existir en `~/obli.sql` o ajustar la ruta en el script.

Backups en `~/Backups/`:

- El script tomará el archivo más reciente con nombre `backupSetUID\_\*.tar.gz`.

- Si no hay archivos con ese patrón, el script finalizará con error.

Nombre del bucket debe ser único:

- Aunque el script contempla la existencia previa del bucket, AWS no permite dos buckets con el mismo nombre globalmente.

- Si el bucket fue creado por otro compañero o el docente, edite el valor de `bucket\_name` con un nombre único.

---

# Ejecución

Escriba en su consola:

**python3 parte1python.py**

La salida le indicará:

- Confirmación de creación del bucket o su existencia

- Subida exitosa del archivo obli.sql

- Identificación y carga del backup más reciente

- Logs con nombres y errores si los hubiera

# 3 - SEGUNDO SCRIPT DE PYTHON

## 3.1 - ¿Qué hace el script?

### 3.1.1 - Crea (o reutiliza) grupos de seguridad

Para permitir:

* Conexión MySQL desde EC2 hacia RDS
* Acceso SSH (puerto 22) y HTTPS (puerto 443) a la EC2

### 3.1.2 - Crea (o reutiliza) una instancia RDS MySQL

* con los valores provistos desde .env.

### 3.1.3 - Crea (o reutiliza) una instancia EC2

* AWS CLI instalado y configurado automáticamente
* Acceso a las credenciales temporales definidas en variables de entorno
* Descarga automática del archivo obli.sql desde el bucket
* Restauración del archivo SQL en la base de datos

## 3.2 - Importación de librerías

### 3.2.1 - Librerias

* import boto3
* from dotenv import load\_dotenv
* import os

### 3.2.2 - Explicación

* **boto3:** Es la biblioteca base para interactuar con **servicios AWS como EC2, RDS y S3**. Se utiliza para crear instancias, grupos de seguridad, y bases de datos.
* **dotenv.load\_dotenv:** Permite cargar variables de entorno desde un archivo .env. Es una **buena práctica de seguridad** para mantener claves, contraseñas o configuraciones fuera del código fuente.
* **os:** Permite acceder a variables de entorno y manipular rutas o archivos del sistema. En este script se usa para extraer variables definidas en el .env.

## 3.3 - Particularidades del entorno (AWS Academy)

### 3.3.1 - Credenciales temporales

Las credenciales de AWS en laboratorios de AWS Academy **expiran cada 4 horas** o al reiniciar el laboratorio. Por tanto:

* Antes de ejecutar el script, **actualice sus credenciales locales**.
* El script incluye mecanismos para pasar esas credenciales a la EC2 mediante variables de entorno y archivos de configuración .aws.

### 3.3.2 - Archivo .env obligatorio

Por seguridad, **no se incluyen claves ni contraseñas en el código**. En su lugar, el script utiliza un archivo .env externo para proteger esta información sensible.

### 3.3.3 - Bucket S3 con nombre único

En AWS, los nombres de buckets S3 deben ser **únicos a nivel global**.  
Si un compañero de equipo o docente ya ha utilizado el mismo nombre, **el script fallará al intentar crear el bucket** o acceder a él. Cambie el nombre en .env si es necesario.

## 3.4 - Consideraciones antes de ejecutar

### 3.4.1 - Verifique que su archivo obli.sql

* Esté ubicado en: **~/<su\_usuario>/obli.sql**
* Modifique el script si cambia esta ruta.

### 3.4.2 - Asegúrese de tener su archivo .env en el mismo directorio que este script.

### 3.4.3 - Las credenciales en su equipo deben estar actualizadas para la sesión del laboratorio actual.

## 3.5 - Ejecución

### 3.5.1 - Escriba en su consola

**python3 PARTE2PYTHON.py**

### 3.5.2 - La salida le indicará

* Qué recursos fueron creados o reutilizados
* Si el volcado del archivo SQL fue exitoso
* Posibles errores en caso de conflictos de nombres o credenciales

## 3.6 - Troubleshooting

### 3.6.1 - Acceso Denegado

* **Al acceder al bucket desde EC2**  
  Asegúrese de que el archivo .aws/credentials fue correctamente creado en la instancia y las variables de entorno fueron exportadas.

### 3.6.2 - Bucket Existente

* Cambie el nombre del bucket en su archivo .env por uno único.

### 3.6.3 - No se encuentra el archivo .sql

* Verifique la ruta y nombre del archivo local (obli.sql), o revise si el script de la primera parte que lo crea ya fue ejecutado.

## 3.7 - Buenas prácticas

### 3.7.1 - No incluya claves o contraseñas directamente en el código.

### 3.7.2 - Use nombres únicos para los recursos.

### 3.7.3 - Elimine recursos que ya no utilice para evitar costos o conflictos.

### 3.7.4 - Agregue logs o un sistema de manejo de errores más robusto si planea escalar este proyecto.

## 3.8 - Archivos involucrados

### 3.8.1 – PARTE2PYHTON.py: Script principal de automatización

### 3.8.2 - .env: Variables sensibles y rutas

### 3.8.3 - obli.sql: Base de datos que será restaurado en MySql

### 3.8.4 - Directorio Backup: Adentro contiene el Backup

# 3.9 - README

# Automatización de Infraestructura AWS con Python (EC2 + RDS + S3)

Este script en Python automatiza el aprovisionamiento de infraestructura en AWS, incluyendo:

- Creación de Grupos de Seguridad (Security Groups)

- Despliegue de una instancia de base de datos RDS (MySQL)

- Despliegue de una instancia EC2

- Configuración automática de AWS CLI y credenciales en la EC2

- Descarga y restauración de un archivo `.sql` desde un bucket S3 a la base de datos

---

# Requisitos previos

Antes de ejecutar el script, asegúrese de cumplir con los siguientes requisitos:

- Tener configurado AWS CLI en su máquina local con credenciales activas del laboratorio actual (AWS Academy).

- Haber creado previamente el archivo `.sql` que desea restaurar (`obli.sql`).

- Contar con el archivo `.env` en el mismo directorio que el script.

---

# Estructura del `.env`

El archivo `.env` debe contener lo siguiente:

```dotenv

DB\_INSTANCE\_CLASS=db.t3.micro

ENGINE=mysql

USER\_NAME=[Nombre de usuario]

DB\_PASSWORD= [Contrasena de mysql]

DB\_ENDPOINT= [Punto de acceso de la base de datos]

bucket=nombre-unico-del-bucket

DATA\_AWS\_CONFIG=[contenido codificado con \n]

DATA\_AWS\_CREDENTIALS=[contenido codificado con \n]

DATA\_AWS\_PROFILE=default

El contenido de DATA\_AWS\_CONFIG y DATA\_AWS\_CREDENTIALS se genera copiando los archivos reales y convirtiendo los saltos de línea a \\n para que puedan ser utilizados dentro de variables de entorno.

Ejecución

Escriba en su consola:

**python3 PARTE2PYTHON.py**

La salida le indicará:

- Qué recursos fueron creados o reutilizados

- Si el volcado del archivo SQL fue exitoso

- Posibles errores en caso de conflictos de nombres o credenciales