Integración DynamoDB Streams + AWS Lambda

🔷 Paso 1: Crear la tabla en DynamoDB

```
aws dynamodb create-table \
--table-name Usuarios \
--attribute-definitions AttributeName=UsuarioID,AttributeType=S \
--key-schema AttributeName=UsuarioID,KeyType=HASH \
--provisioned-throughput ReadCapacityUnits=5,WriteCapacityUnits=5
```

🔷 Paso 2: Habilitar DynamoDB Streams

```
aws dynamodb update-table \
--table-name Usuarios \
--stream-specification StreamEnabled=true,StreamViewType=NEW_AND_OLD_IMAGES
```

🧠 ¿Qué significa?

- StreamEnabled=true: activa el stream
- StreamViewType=NEW_AND_OLD_IMAGES: guarda versiones **antes y después** de cada cambio

---

🔷 Paso 3: Crear la función Lambda en la consola

Desde la consola de AWS Lambda:

- Elige **“Crear función”**
- Tipo: **Desde cero**
- Nombre: procesarCambiosUsuarios
- Tiempo de ejecución: Python 3.13
- Rol: **Usar un rol existente > LabRole**

✅ Este rol ya tiene permisos para interactuar con DynamoDB y escribir logs.

🔷 Paso 4: Agregar código a Lambda

Reemplaza el contenido por:

```
import json
def lambda_handler(event, context):
    print("=== EVENTO RECIBIDO ===")
    print(json.dumps(event, indent=2))
    return { 'statusCode': 200,'body': json.dumps('OK') }
```

✅ Este código imprimirá en CloudWatch el contenido de los eventos generados en DynamoDB.

🔷 **Paso 5: Agregar el desencadenador de DynamoDB**

1. En la consola de Lambda, haz clic en **Agregar desencadenador**
2. Elige **DynamoDB**
3. Tabla: Usuarios
4. Activar desencadenador: ✅
5. Tipo de vista: NEW_AND_OLD_IMAGES

¡Listo! Tu función Lambda ahora escuchará los cambios que ocurran en la tabla.

🔷 **Paso 6: Insertar ítems para probar**

```
aws dynamodb put-item \
--table-name Usuarios \
--item '{"UsuarioID": {"S": "001"}, "Nombre": {"S": "Felipe"}, "Edad": {"N": "35"}}'
```

🧠 **¿Qué significa?**

- --item: envía los datos en formato JSON
- {"S": "texto"}: tipo String
- {"N": "número"}: tipo Number

Puedes agregar más ítems para probar distintos eventos:

```
aws dynamodb put-item \
--table-name Usuarios \
--item '{"UsuarioID": {"S": "002"}, "Nombre": {"S": "Gabriela"}, "Edad": {"N": "28"}}'


aws dynamodb update-item \
--table-name Usuarios \
--key '{"UsuarioID": {"S": "001"}}' \
--update-expression "SET Edad = :nuevaEdad" \
--expression-attribute-values '{":nuevaEdad": {"N": "36"}}'


aws dynamodb delete-item \
--table-name Usuarios \
--key '{"UsuarioID": {"S": "002"}}'
```

🔷 **Paso 7: Ver los resultados en CloudWatch**

1. Ve a **CloudWatch > Grupos de registros**
2. Busca /aws/lambda/procesarCambiosUsuarios
3. Entra al log más reciente

Verás algo como:

```
== EVENTO RECIBIDO ===
{
  "Records": [
    {
      "eventName": "INSERT",
      "dynamodb": {
        "NewImage": {
          "UsuarioID": {"S": "001"},
          "Nombre": {"S": "Felipe"},
          "Edad": {"N": "35"}
        }
      }
    }
  ]
}
```

RESUMEN: ¿Qué aprendimos?

🟪 **Amazon DynamoDB Streams**

Permite **capturar eventos en tiempo real** cuando se insertan, actualizan o eliminan registros en una tabla. Estos eventos pueden ser consumidos por otros servicios (como Lambda).

🟦 **AWS Lambda**

Servicio de cómputo sin servidor. Ejecuta funciones (código) automáticamente ante eventos, sin necesidad de administrar servidores. Ideal para automatizar tareas.

🟩 **Amazon CloudWatch**

Servicio de monitoreo. Guarda los logs generados por servicios como Lambda. Nos permite **ver la salida** del código ejecutado, detectar errores o auditar eventos.