Código Documentado

```
import boto3
import datetime

def lambda_handler(event, context):
    rds_client = boto3.client('rds')
    s3_client = boto3.client('s3')

    # Obtener el último snapshot de RDS
    snapshots = rds_client.describe_db_snapshots(
        DBInstanceIdentifier='prod-02',
        SnapshotType='manual'
    )
    latest_snapshot = max(snapshots['DBSnapshots'], key=lambda x:
x['SnapshotCreateTime'])

    # Exportar el snapshot a S3
    export_task_id = f"task-
{datetime.datetime.now().strftime('%Y%m%d%H%M%S')}"
    source_arn = latest_snapshot['DBSnapshotArn']
    s3_bucket_name = 'dbs3export'
    iam_role_arn = 'arn:aws:iam::253021683072:role/iamroles3bck'
    kms_key_id = 'arn:aws:kms:us-east-1:253021683072:key/be6a11f7-16d9-
4ee2-bc0c-eab9e87bf7f1'

    rds_client.start_export_task(
        ExportTaskIdentifier=export_task_id,
        SourceArn=source_arn,
        S3BucketName=s3_bucket_name,
        IamRoleArn=iam_role_arn,
        KmsKeyId=kms_key_id,
        S3Prefix='exported-snapshots/'
    )

    # Eliminar los archivos Parquet anteriores en S3
    bucket_objects = s3_client.list_objects_v2(Bucket=s3_bucket_name,
Prefix='exported-snapshots/')
    if 'Contents' in bucket_objects:
        for obj in bucket_objects['Contents']:
            if obj['Key'].endswith('.parquet'):
                s3_client.delete_object(Bucket=s3_bucket_name,
Key=obj['Key'])

    return {
        'statusCode': 200,
        'body': f'Successfully started export task {export_task_id} and
deleted old Parquet files.'
    }
```

Lanzar Manualmente

Para lanzar la función Lambda manualmente y ver los logs de ejecución, sigue estos pasos:

1. **Guardar el Código y Crear el Archivo ZIP**

   Guarda el código anterior en un archivo llamado `lambda_function.py` y crea un archivo ZIP con el contenido:

   ```
   zip function.zip lambda_function.py
   ```

2. **Crear y Configurar el Rol IAM**

   Crea un archivo `trust-policy.json` con la política de confianza:

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "lambda.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

   Luego, crea el rol IAM y adjunta las políticas de permisos necesarias:

   ```
    # Crear el rol IAM con la política de confianza
    aws iam create-role --role-name lambda-execution-role --assume-role-
   policy-document file://trust-policy.json

    # Adjuntar las políticas necesarias al rol IAM
    aws iam attach-role-policy --role-name lambda-execution-role --
   policy-arn arn:aws:iam::aws:policy/service-
   role/AWSLambdaBasicExecutionRole
    aws iam attach-role-policy --role-name lambda-execution-role --
   policy-arn arn:aws:iam::aws:policy/AmazonRDSFullAccess
    aws iam attach-role-policy --role-name lambda-execution-role --
   policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
    aws iam attach-role-policy --role-name lambda-execution-role --
   policy-arn arn:aws:iam::aws:policy/AWSKeyManagementServicePowerUser
   ```

3. **Desplegar la Función Lambda**

Obtén el ARN del rol IAM creado:

```
ROLE_ARN=$(aws iam get-role --role-name lambda-execution-role --query
'Role.Arn' --output text)
```

Crea la función Lambda:

```
aws lambda create-function --function-name export-rds-snapshot --
runtime python3.8 --role $ROLE_ARN --handler
lambda_function.lambda_handler --zip-file fileb://function.zip
aws lambda update-function-configuration --function-name export-rds-
snapshot --timeout 900
```

4. **Configurar el Disparador de CloudWatch Events**

Crea una regla de CloudWatch Events para ejecutar la función Lambda una vez al día:

```
aws events put-rule --name daily-rds-snapshot-export --schedule-
expression 'rate(1 day)'
```

Agrega permisos para que CloudWatch Events invoque la función Lambda:

```
aws lambda add-permission --function-name export-rds-snapshot --
statement-id daily-rds-snapshot-export --action
'lambda:InvokeFunction' --principal events.amazonaws.com --source-arn
arn:aws:events:REGION:ACCOUNT_ID:rule/daily-rds-snapshot-export
```

Reemplaza `REGION` y `ACCOUNT_ID` con tu región de AWS y ID de cuenta.

Obtén el ARN de la función Lambda:

```
FUNCTION_ARN=$(aws lambda get-function --function-name export-rds-
snapshot --query 'Configuration.FunctionArn' --output text)
```

Asigna la regla de CloudWatch Events a la función Lambda:

```
aws events put-targets --rule daily-rds-snapshot-export --targets
"Id"="1","Arn"="$FUNCTION_ARN"
```

5. **Probar la Configuración**

Puedes probar la función Lambda ejecutándola manualmente:

```
aws lambda invoke --function-name export-rds-snapshot output.txt
```

Revisa el archivo `output.txt` para ver los resultados y verifica en S3 que los archivos Parquet anteriores se eliminen y que el snapshot se exporte correctamente.

Ver Logs de Ejecución

Para ver los logs de ejecución de la función Lambda, puedes usar CloudWatch Logs:

1. **Listar los Grupos de Logs**:

   ```
   aws logs describe-log-groups
   ```

   Busca el grupo de logs correspondiente a tu función Lambda (normalmente tendrá un nombre similar a `/aws/lambda/export-rds-snapshot`).

2. **Ver los Streams de Logs**:

   ```
   aws logs describe-log-streams --log-group-name /aws/lambda/export-rds-snapshot
   ```

3. **Ver los Logs**:

   ```
   aws logs get-log-events --log-group-name /aws/lambda/export-rds-snapshot --log-stream-name <log-stream-name>
   ```

Mantenimiento

Para hacer mantenimiento en la función Lambda, puedes actualizar el código y redeployar:

1. **Actualizar el Código**:

   Edita el archivo `lambda_function.py` según sea necesario.

2. **Crear un Nuevo Archivo ZIP**:

   ```
   zip function.zip lambda_function.py
   ```

3. **Actualizar la Función Lambda**:

```
aws lambda update-function-code --function-name export-rds-snapshot --
zip-file fileb://function.zip
```

4. **Probar y Verificar**:

   Ejecuta manualmente la función y verifica los logs como se describió anteriormente para asegurar que todo funcione correctamente.

Con estos pasos, podrás desplegar, ejecutar, verificar los logs y mantener tu función Lambda utilizando AWS CLI.