

BIG DATA

Pentaho - Ejemplo 5



ALUMNO: José Antonio Díaz Aranda
CURSO: IA Y BIG DATA
MÓDULO: BD

Tenemos de base la configuración del ejercicio 3, por lo que se explicará desde ese.

Hacemos un merge join y lo unimos por ManufacturerID

Merge join

Step name: Merge join 2

First Step: Order por ManufacturerID

Second Step: Ordenar fabricantes ManufacturerID

Join Type: INNER

Keys for 1st step:

#	Key field
1	Manufac...

Get key fields

Keys for 2nd step:

#	Key field
1	Manufac...

Get key fields

[Help](#) [OK](#) [Cancel](#)

Después ordenamos por ManufacturerID

Sort rows

Step name: Sort rows 3

Sort directory: %%java.io.tmpdir%% [Browse...](#)

TMP-file prefix: out

Sort size (rows in memory): 1000000

Free memory threshold (in %):

Compress TMP Files? ☒

Only pass unique rows? (verifies keys only) ☐

Fields:

#	Fieldname	Ascending	Case sensitive compare?	Sort based on current locale?	Collator Strength	Presorted
1	ManufacturerID	Y	N	N	0	N

Cuando lo tengamos ordenado creamos el group by y lo unimos por Manufacturer ID

Group by

Step name

Include all rows? ☐

Temporary files directory [Browse...](#)

TMP-file prefix

Add line number, restart in each ☐

Line number field name

Always give back a result row ☐

The fields that make up the group:

#	Group field
1	ManufacturerID
2	Manufacturer

[Get Fields](#)

Aggregates :

#	Name	Subject	Type	Value
1	Unidades	Units	Sum	
2	Recaudación	Revenue	Sum	

[Get lookup fields](#)

Ahora creamos un calculador para obtener el precio medio

Calculator

Step name

☒ Throw an error on non existing files

Fields:

#	New field	Calculation	Field A	Field B	Field C	Value type	Length	Precision	Remove	Conversion mask	Decimal symbol	Grouping symbol	Currency sy
1	PrecioMedio	A / B	Recaudacion	Unidades		None			N				

Ahora haremos la parte de configuración de S3:

Creamos un bucket

VPC

Bucket type [Info](#)

☒ General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ Directory - New
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

Format: s3://bucket/prefix

Object Ownership [Info](#)

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

☐ ACLs disabled (recommended)
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using

☒ ACLs enabled
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be



Block Public Access settings for this bucket

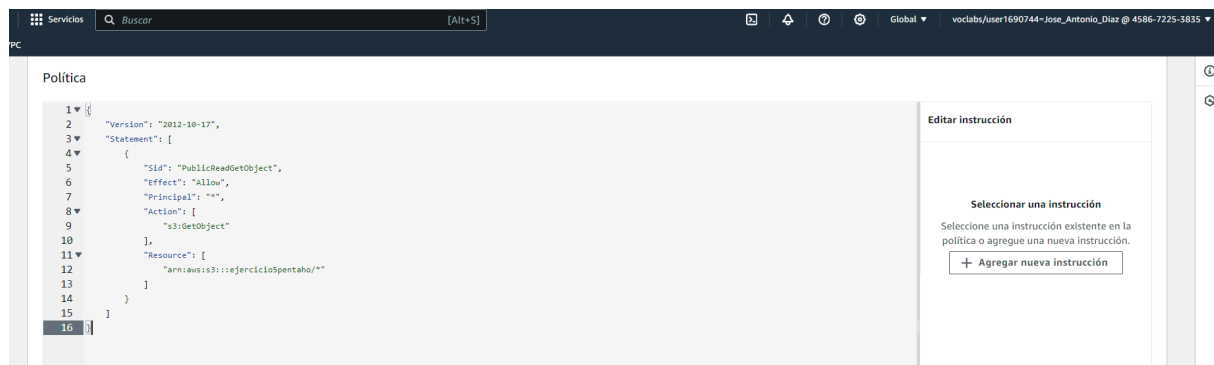
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ Block *all* public access

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Metemos esta configuración en la política



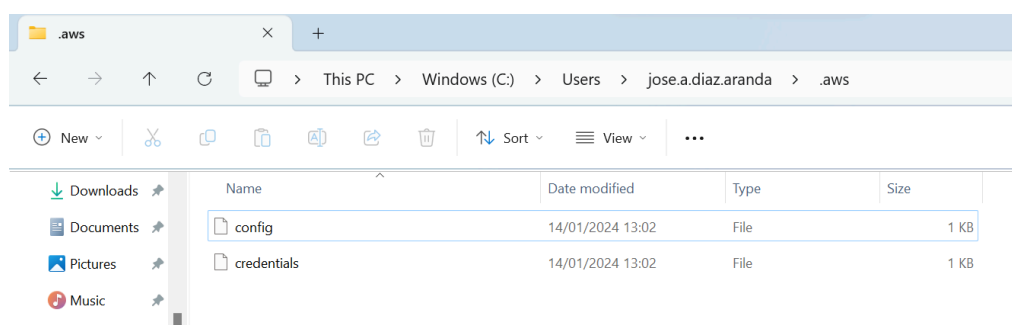
Instalamos awsCLI

<https://awscli.amazonaws.com/AWSCLIV2.msi>

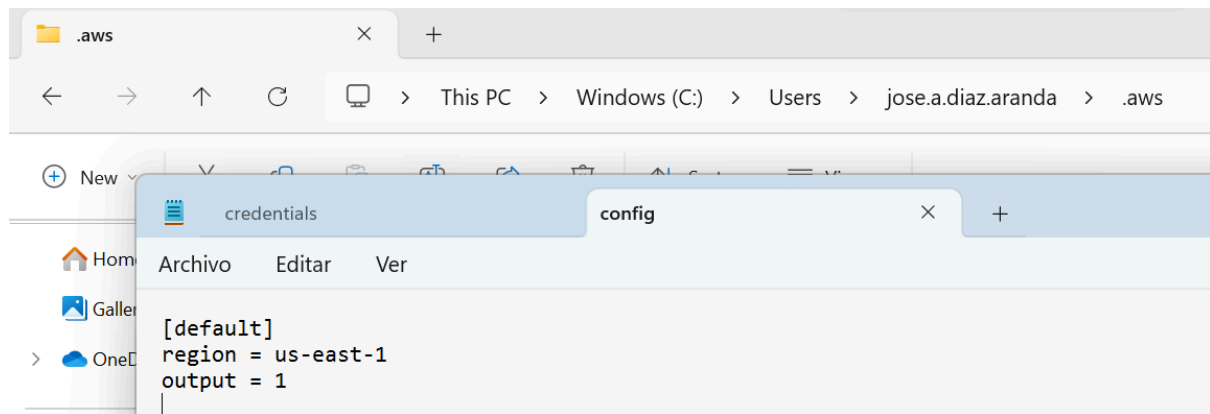
Ejecutamos este comando en la terminal

```
C:\Users\jose.a.diaz.aranda>aws configure
AWS Access Key ID [None]: 1
AWS Secret Access Key [None]: 1
Default region name [None]: 1
Default output format [None]: 1
```

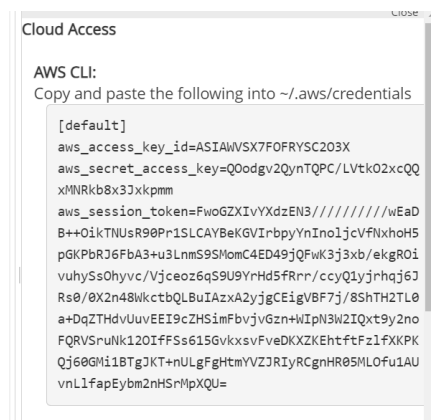
Se nos habrá creado esta carpeta



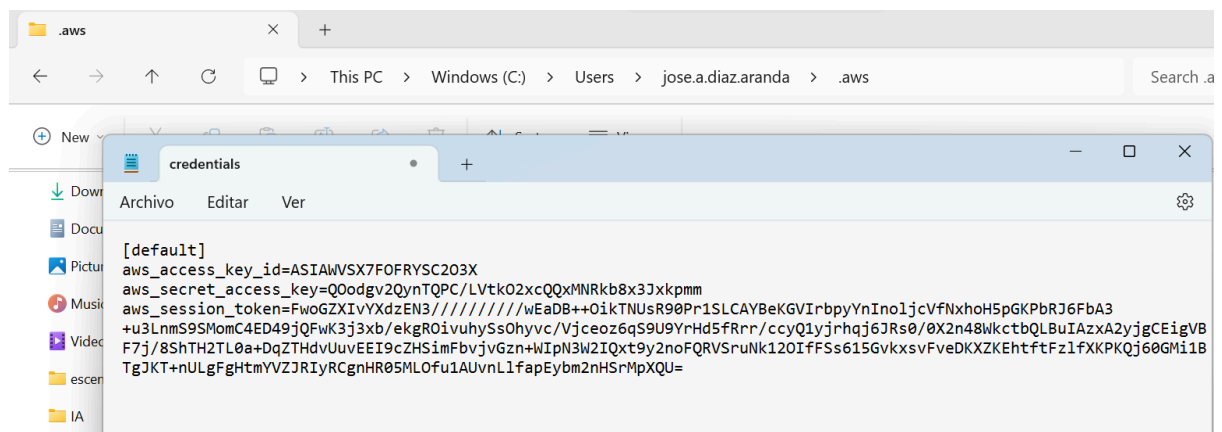
En config cambiamos la región



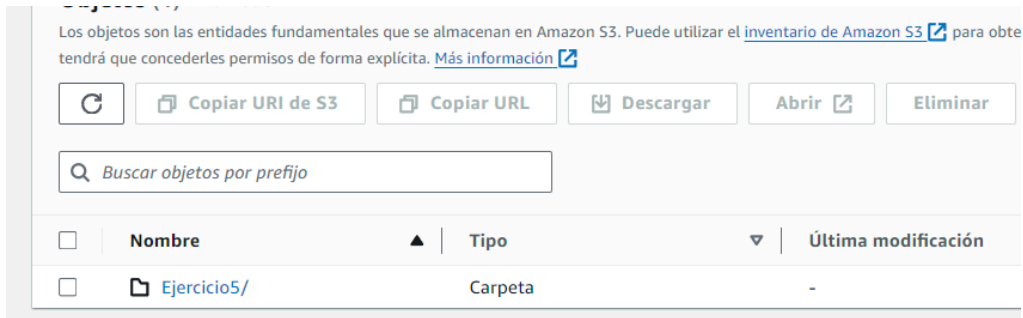
Ahora tenemos que introducir las credenciales que tenemos en la pestaña aws details de nuestro laboratorio



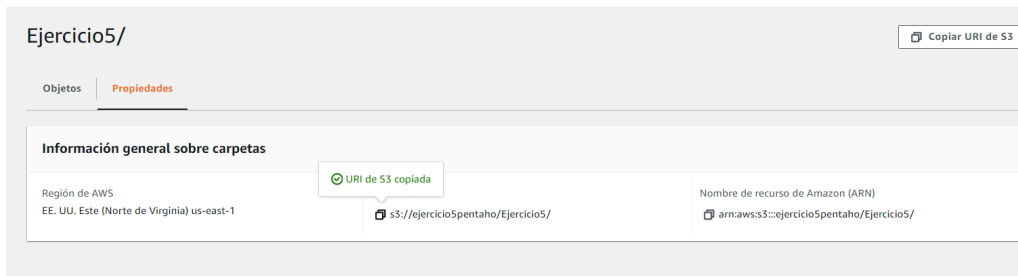
Lo introducimos en el fichero de credentials



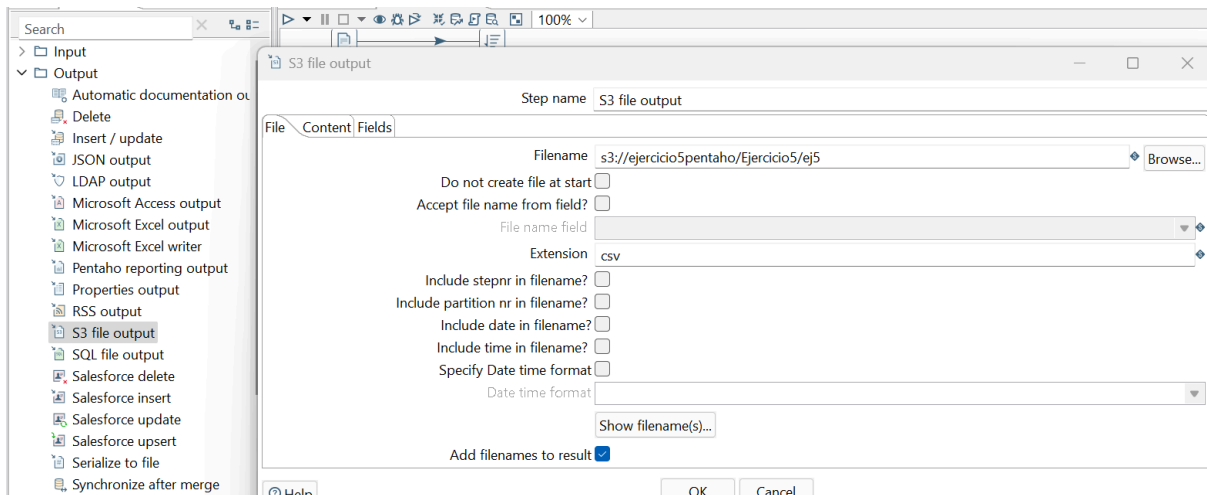
Creamos una carpeta en el bucket



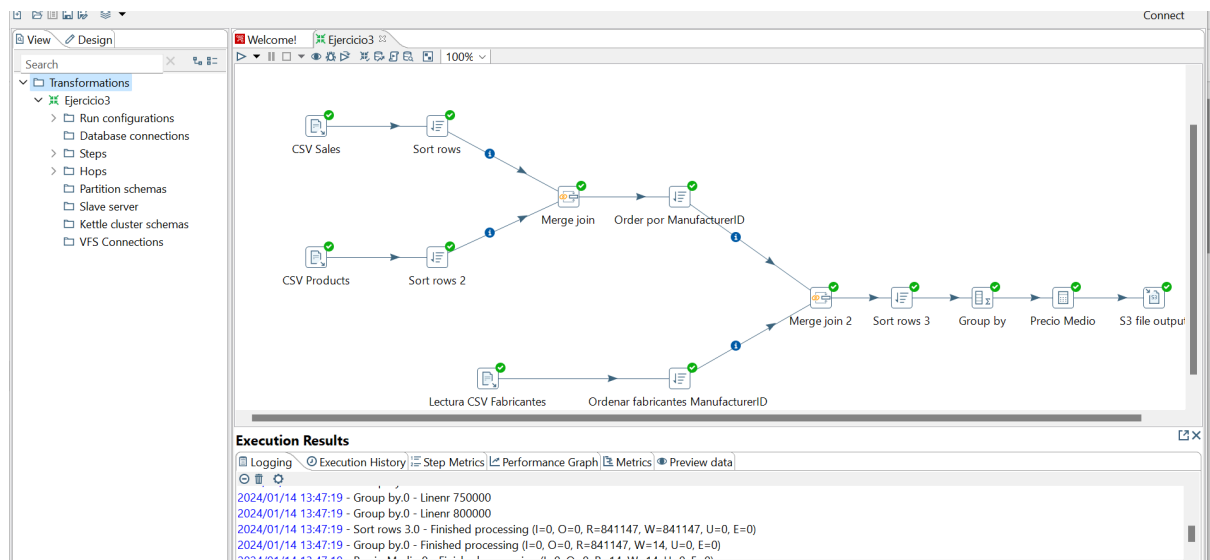
Ahora en propiedades copiamos el enlace para meterlo en el pentaho



Metemos en un fichero S3 output el código y ejecutamos



Lo ejecutamos



Vemos el resultado

