

# **PR\_07.1 Dani Gayol Rodríguez**

PR_07.1 Dani Gayol Rodríguez.....	1
Apartado A.....	1
1.) Crea un clúster Hadoop EMR con 1 master y dos nodos. Selecciona la opción Core Hadoop (versión 7.0.0). .....	2
Apartado B.....	2
1.) Viene en formato Excel. Desde el propio Excel puedes convertirlo a formato 'csv' .....	3
2.) Crea una carpeta con tu nombre en el directorio user del HDFS de EMR .....	3
3.) Crea dentro de él una carpeta llamada ventas y sube a ella el 'csv' que obtuviste anteriormente.....	3
Apartado C .....	3
1.) ¿Cuántos registros tiene la tabla? Big Data.....	4
2.) Mostrar registros con cantidades mayores o iguales a cero. .....	4
3.) A partir de la consulta anterior, mostrar registros precio mayor a cero .....	4
4.) A partir de la consulta anterior, mostrar solamente los registros con algún valor en el campo CustomerID. .....	4
5.) ¿Cuántos registros tiene la última consulta? .....	4
6.) Almacena la consulta final del punto 4 en un fichero llamado ventas.csv dentro de la carpeta de apartado B.....	4
Apartado D .....	4
1.) Crear base de datos.....	5
2.) Crear tabla externa sobre los datos RAW (CSV).....	5
3.) Hive no maneja muy bien el formato de fecha original, conviértelo a d/M/yyyy H:mm.....	5
4.) Crea la misma estructura de tabla, pero particionada por año y mes. ....	5
5.) Inserta los registros del punto 1.2 en la tabla particionada.....	5
Apartado E.....	5
1.) 10 clientes con mayor gasto total .....	6
2.) Clientes con más compras (cantidad de facturas) .....	6

3.) 10 productos más vendidos .....	6
4.) 10 Productos más rentables (suma de precio unitario por cantidad).....	6
5.) Países con mayor volumen de ventas Análisis temporal .....	6
6. Ventas totales por mes (suma de precio unitario por cantidad) .....	6
7. Hora del día con más actividad.....	6
Apartado F.....	6
1.) En tu servidor MySql en la máquina EC2 crea una base de datos y una tabla para almacenar los datos del fichero ventas.csv.....	7
2.) Exporta con SQOOP los datos de ventas.csv a la tabla que creaste en el punto anterior.....	7

# Apartado A

**1.) Crea un clúster Hadoop EMR con 1 master y dos nodos.**  
**Selecciona la opción Core Hadoop (versión 7.0.0).**

Primero de todo, tenemos que arrancar el laboratorio de AWS, una vez iniciado, buscamos “Amazon EMR” en el buscador



The screenshot shows the AWS search interface with the query "Amazon EMR". On the left, there's a sidebar with links like "Servicios (165)", "Características (429)", etc. The main area lists services: "EMR" (highlighted with a red box), "AWS Private Certificate Authority", and "Amazon Redshift".

A continuación, le damos al botón de “Crear Cluster” y lo configuramos como dice el enunciado del ejercicio

**Crear clúster** Información

▼ **Nombre y aplicaciones - obligatorio** Información  
Asigne un nombre a su clúster y elija las aplicaciones que desea instalar en él.

**Nombre**  
Hadoop EMR

**Versión de Amazon EMR** Información  
Una versión contiene un conjunto de aplicaciones que se puede instalar en el clúster.  
emr-7.0.0 ▾

### Paquete de aplicaciones



- |                                                           |                                                                    |                                                |
|-----------------------------------------------------------|--------------------------------------------------------------------|------------------------------------------------|
| <input type="checkbox"/> AmazonCloudWatchAgent 1.300031.1 | <input type="checkbox"/> Flink 1.18.0                              | <input type="checkbox"/> HBase 2.4.17          |
| <input type="checkbox"/> HCatalog 3.1.3                   | <input checked="" type="checkbox"/> Hadoop 3.3.6                   | <input checked="" type="checkbox"/> Hive 3.1.3 |
| <input checked="" type="checkbox"/> Hue 4.11.0            | <input checked="" type="checkbox"/> JupyterEnterpriseGateway 2.6.0 | <input type="checkbox"/> JupyterHub 1.5.0      |
| <input checked="" type="checkbox"/> Livy 0.7.1            | <input type="checkbox"/> MXNet 1.9.1                               | <input type="checkbox"/> Oozie 5.2.1           |
| <input type="checkbox"/> Phoenix 5.1.3                    | <input checked="" type="checkbox"/> Pig 0.17.0                     | <input type="checkbox"/> Presto 0.283          |
| <input checked="" type="checkbox"/> Spark 3.5.0           | <input checked="" type="checkbox"/> Sqoop 1.4.7                    | <input type="checkbox"/> TensorFlow 2.11.0     |
| <input checked="" type="checkbox"/> Tez 0.10.2            | <input type="checkbox"/> Trino 426                                 | <input type="checkbox"/> Zeppelin 0.10.1       |
| <input type="checkbox"/> ZooKeeper 3.5.10                 |                                                                    |                                                |

Ahora creamos los dos nodos que nos piden

Nombre	Tipo de instancia	Tamaño de instancia(s)
Central	m4.large	1
Nodo 1	m4.large	1
Nodo 2	m4.large	1

También le tuve que crear un par de claves de acceso

#### ▼ Configuración de seguridad y par de claves de EC2 Información

Elija una configuración de seguridad o cree una nueva que pueda reutilizar con otros clústeres.

##### Configuración de seguridad

Seleccione la configuración del servicio de cifrado, autenticación, autorización y metadatos de instancia del clúster.

<input type="button" value="Elegir una configuración de segi"/>	<input type="button" value="Examinar"/>	<input type="button" value="Crear configuración de seguridad"/>
-----------------------------------------------------------------	-----------------------------------------	-----------------------------------------------------------------

##### Par de claves de Amazon EC2 para el protocolo SSH al clúster Información

<input type="button" value="emr"/>	<input type="button" value="Examinar"/>	<input type="button" value="Crear par de claves"/>
------------------------------------	-----------------------------------------	----------------------------------------------------

Ahora le vamos a desbloquear el acceso publico

## ▼ EMR en EC2

- Clústeres
  - Blocs de notas y repositorios
  - de Git
  - Eventos
- Bloquear el acceso público**
- Configuraciones de seguridad

se especifique explícitamente como una excepción.

### Configuración del bloqueo del acceso público

#### Bloquear el acceso público

Activado

#### Excepciones de rango de puertos

Un clúster se puede lanzar con reglas del grupo de seguridad que permiten el tráfico de puerto.

22

## Configuración del bloqueo del acceso público

### Bloquear el acceso público

Este cambio solo afecta a los nuevos clústeres de EMR. Los clústeres de EMR existentes no se ven afectados.

Activar (*recomendado*)

Bloquear el acceso público a todos los puertos, excepto los que se agregan como excepciones.

Desactivar

Permitir el acceso público en función de las reglas del grupo de seguridad.

Ahora vamos a modificar el grupo de seguridad para poder conectarnos por “ssh”

## Red y seguridad Información

### Red

#### Virtual Private Cloud (VPC)

vpc-0da527f677e65a9bf ↗

#### Subredes y zonas de disponibilidad (AZ)

subnet-07c8d2c469c627a2d ↗ | us-east-1e

## ▼ Grupos de seguridad de EC2 (firewall)

#### Nodo principal

#### Grupos de seguridad administrados de EMR

sg-0a91caaee8b3aa783f ↗

#### Grupos de seguridad adicionales

#### Nodos principales y de tareas

#### Grupos de seguridad administrados de EMR

sg-02c2c569d408cc571 ↗

#### Grupos de seguridad adicionales



Finalmente, ya nos podemos conectar mediante la terminal de Windows

```
C:\Users\Mañana>ssh -i "C:\Users\Mañana\Downloads\emr.pem" hadoop@ec2-54-172-132-91.compute-1.amazonaws.com
```

```
'_#_
~\_\_#####_          Amazon Linux 2023
~~ \_\#####\
~~  \###|
~~   \#/ ___  https://aws.amazon.com/linux/amazon-linux-2023
~~    \~' '->
~~     /
~~..-.-/_/
~~/_/_/
~~/_m/_'

Last login: Tue Dec 16 08:34:34 2025

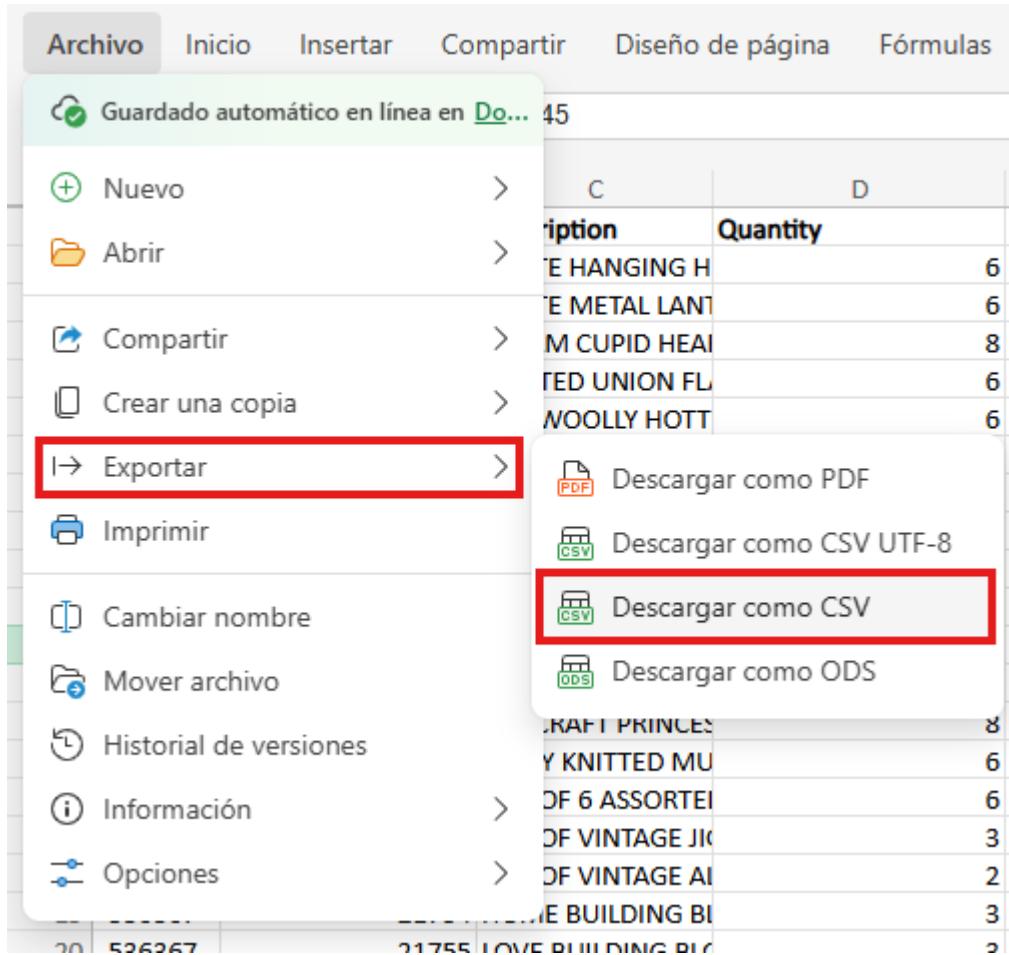
EEEEEEEEEEEEEEEEEE MMMMMMMM      MMMMMMM RRRRRRRRRRRRRR
E:::::::E:::::E M:::::M      M:::::M R:::::R
EE:::::E:::::E M:::::M      M:::::M R:::::RRRRR:::::R
E:::::E     EEEEE E:::::M      M:::::M RR:::R     R:::R
E:::::E     M:::::M::::M      M::::M::::M R:::R     R:::R
E:::::E:::::E EEEEEE M::::M M::::M M::::M R:::::RRRRR:::::R
E:::::E:::::E EEEEEE M::::M M::::M::::M M::::M R:::::R:::::RR
E:::::E:::::E EEEEEE M::::M M::::M M::::M R:::::RRRRR:::::R
E:::::E     M:::::M M::::M M::::M R:::::R     R:::R
E:::::E     EEEEE E:::::M      MMM M:::::M R:::::R     R:::R
EE:::::E:::::E E:::::E M:::::M      M:::::M R:::::R     R:::R
E:::::::E:::::E M:::::M      M:::::M RR:::::R     R:::::R
EEEEEEEEEEEEEEEEEE MMMMMMM      MMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-51-157 ~]$ |
```

## Apartado B

**1.) Viene en formato Excel. Desde el propio Excel puedes convertirlo a formato 'csv'**

Desde Excel cargamos el archivo y una vez cargado lo podemos exportar en formato “.csv”



**2.) Crea una carpeta con tu nombre en el directorio user del HDFS de EMR**

Ahora vamos a crear una carpeta con mi nombre en el directorio “/user”

```
[hadoop@ip-172-31-51-157 ~]$ hdfs dfs -mkdir /user/dani-gayol  
[hadoop@ip-172-31-51-157 ~]$ hdfs dfs -ls /user  
Found 9 items  


|            |   |        |                |   |                  |                  |
|------------|---|--------|----------------|---|------------------|------------------|
| drwxr-xr-x | - | hadoop | hdfsadmingroup | 0 | 2025-12-16 08:56 | /user/dani-gayol |
| drwxrwxrwx | - | hadoop | hdfsadmingroup | 0 | 2025-12-16 08:29 | /user/hadoop     |
| drwxr-xr-x | - | mapred | mapred         | 0 | 2025-12-16 08:29 | /user/history    |
| drwxrwxrwx | - | hdfs   | hdfsadmingroup | 0 | 2025-12-16 08:29 | /user/hive       |
| drwxrwxrwx | - | hue    | hue            | 0 | 2025-12-16 08:29 | /user/hue        |
| drwxrwxrwx | - | livy   | livy           | 0 | 2025-12-16 08:29 | /user/livy       |
| drwxrwxrwx | - | oozie  | oozie          | 0 | 2025-12-16 08:30 | /user/oozie      |
| drwxrwxrwx | - | root   | hdfsadmingroup | 0 | 2025-12-16 08:29 | /user/root       |
| drwxrwxrwx | - | spark  | spark          | 0 | 2025-12-16 08:29 | /user/spark      |


```

### 3.) Crea dentro de él una carpeta llamada ventas y sube a ella el 'csv' que obtuviste anteriormente.

Primero tenemos que pasar el “.csv” desde Windows hasta la máquina virtual y después al HDFS

```
C:\Users\Mañana>scp -i "C:\Users\Mañana\Downloads\emr.pem" "C:\Users\Mañana\Downloads\Online-Retail.csv" hadoop@ec2-54-1-72-132-91.compute-1.amazonaws.com:/home/hadoop/  
Online-Retail.csv  
100% 44MB 9.3MB/s 00:04
```

Ahora ya podemos crear la carpeta y pegarle el archivo

```
[hadoop@ip-172-31-51-157 ~]$ hdfs dfs -mkdir /user/dani-gayol/ventas  
[hadoop@ip-172-31-51-157 ~]$ hdfs dfs -put /home/hadoop/Online-Retail.csv /user/dani-gayol/ventas/  
[hadoop@ip-172-31-51-157 ~]$ hdfs dfs -ls /user/dani-gayol/ventas/  
Found 1 items  
-rw-r--r-- 1 hadoop hdfsadmingroup 46079956 2025-12-16 09:09 /user/dani-gayol/ventas/Online-Retail.csv
```

# Apartado C

Ahora vamos a usar “PIG”, para ello primero de todo, cargamos los datos

```
grunt> ventas = LOAD '/user/dani-gayol/ventas/Online-Retail.csv' USING PigStorage(',') AS  
(InvoiceNo:chararray, StockCode:chararray, Description:chararray, Quantity:int, InvoiceDate:chararray, UnitPrice:double, CustomerID:int, Country:chararray);
```

## 1.) ¿Cuántos registros tiene la tabla? Big Data

```
grunt> DESCRIBE ventas; COUNT = FOREACH (GROUP ventas ALL) GENERATE COUNT(ventas); DUMP COUNT;
```

(541910)

## 2.) Mostrar registros con cantidades mayores o iguales a cero.

```
grunt> ventas_cantidad = FILTER ventas BY Quantity >= 0; DUMP ventas_cantidad;
```

```
(581587,22367,CHILDRENS APRON SPACEBOY DESIGN,8,09/12/2011 12:50,,12680,France)  
(581587,22629,SPACEBOY LUNCH BOX ,12,09/12/2011 12:50,,12680,France)  
(581587,23256,CHILDRENS CUTLERY SPACEBOY ,4,09/12/2011 12:50,,12680,France)  
(581587,22613,PACK OF 20 SPACEBOY NAPKINS,12,09/12/2011 12:50,,12680,France)  
(581587,22899,CHILDREN'S APRON DOLLY GIRL ,6,09/12/2011 12:50,,12680,France)  
(581587,23254,CHILDRENS CUTLERY DOLLY GIRL ,4,09/12/2011 12:50,,12680,France)  
(581587,23255,CHILDRENS CUTLERY CIRCUS PARADE,4,09/12/2011 12:50,,12680,France)  
(581587,22138,BAKING SET 9 PIECE RETROSPOT ,3,09/12/2011 12:50,,12680,France)
```

## 3.) A partir de la consulta anterior, mostrar registros precio mayor a cero

```
grunt> ventas_precio = FILTER ventas_cantidad BY UnitPrice > 0; DUMP ventas_precio;
```

```
(581493,POST,POSTAGE,1,09/12/2011 10:10,15.0,12423,Belgium)  
(581494,POST,POSTAGE,2,09/12/2011 10:13,18.0,12518,Germany)  
(581570,POST,POSTAGE,1,09/12/2011 11:59,18.0,12662,Germany)  
(581574,POST,POSTAGE,2,09/12/2011 12:09,18.0,12526,Germany)  
(581578,POST,POSTAGE,3,09/12/2011 12:16,18.0,12713,Germany)
```

## 4.) A partir de la consulta anterior, mostrar solamente los registros con algún valor en el campo CustomerID.

```
grunt> ventas_final = FILTER ventas_precio BY CustomerID IS NOT NULL; DUMP ventas_final;
```

```
(581232,POST,POSTAGE,4,08/12/2011 10:26,40.0,12358,Austria)
(581266,POST,POSTAGE,5,08/12/2011 11:25,18.0,12621,Germany)
(581279,POST,POSTAGE,3,08/12/2011 11:35,18.0,12437,France)
```

## 5.) ¿Cuántos registros tiene la última consulta?

```
grunt> COUNT_FINAL = FOREACH (GROUP ventas_final ALL) GENERATE COUNT(ventas_final); DUMP COUNT_FINAL;
```

```
(1712)
```

## 6.) Almacena la consulta final del punto 4 en un fichero llamado ventas.csv dentro de la carpeta de apartado B.

```
grunt> STORE ventas_final INTO '/user/dani-gayol/ventas/ventas_limpias.csv' USING PigStorage(',');
```

```
Successfully stored 1712 records (110820 bytes) in: "/user/dani-gayol/ventas/ventas_limpias.csv"
```

# Apartado D

## 1.) Crear base de datos

```
[hadoop@ip-172-31-53-111 ~]$ hive
Hive Session ID = 529764af-ffb2-45da-908a-15e9c1925356

Logging initialized using configuration in file:/etc/hive,
hive> CREATE DATABASE IF NOT EXISTS retail;
OK
Time taken: 0.575 seconds
hive> USE retail;
OK
Time taken: 0.031 seconds
```

## 2.) Crear tabla externa sobre los datos RAW (CSV)

```
hive> CREATE EXTERNAL TABLE ventas_raw ( InvoiceNo STRING, StockCode STRING, Description STRING, Quantity INT, InvoiceDa
te STRING, UnitPrice DOUBLE, CustomerID INT, Country STRING ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ';' STORED AS TE
XTFILE LOCATION '/user/dani-gayol/ventas/ventas_limpias.csv/part-v000-o000-r-00000';|
```

```
hive> CREATE EXTERNAL TABLE ventas_raw ( InvoiceNo STRING, StockCode STRING, Description STRING, Quantity INT, InvoiceDa
te STRING, UnitPrice DOUBLE, CustomerID INT, Country STRING ) ROW FORMAT DELIMITED FIELDS TERMINATED BY ';' STORED AS TE
XTFILE LOCATION '/user/dani-gayol/ventas/ventas_limpias.csv';
OK
Time taken: 0.13 seconds
```

## 3.) Hive no maneja muy bien el formato de fecha original, conviértelo a d/M/yyyy H:mm

```
hive> SELECT InvoiceNo, StockCode, Description, Quantity, from_unixtime(unix_timestamp(InvoiceDate, 'd/M/yyyy H:mm')) AS
InvoiceDate, UnitPrice, CustomerID, Country FROM ventas_raw;|
```

InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
581493	POST	POSTAGE 1	1	2011-12-09 10:10:00	15.0	12423	Belgium
581494	POST	POSTAGE 2	2	2011-12-09 10:13:00	18.0	12518	Germany
581570	POST	POSTAGE 1	1	2011-12-09 11:59:00	18.0	12662	Germany
581574	POST	POSTAGE 2	2	2011-12-09 12:09:00	18.0	12526	Germany
581578	POST	POSTAGE 3	3	2011-12-09 12:16:00	18.0	12713	Germany

```
Time taken: 2.241 seconds, Fetched: 1712 row(s)
```

## 4.) Crea la misma estructura de tabla, pero particionada por año y mes.

```
hive> CREATE TABLE ventas_particionada ( InvoiceNo STRING, StockCode STRING, Description STRING, Quantity INT, InvoiceDa
te TIMESTAMP, UnitPrice DOUBLE, CustomerID INT, Country STRING ) PARTITIONED BY (year INT, month INT) STORED AS PARQUET;
OK
Time taken: 0.071 seconds
```

## 5.) Inserta los registros del punto 1.2 en la tabla particionada.

Al intentar hacerlo, me salía un error por lo tanto tuve que activar modo dinámico no estricto

```
hive> SET hive.exec.dynamic.partition=true;
hive> SET hive.exec.dynamic.partition.mode=nonstrict;
hive> INSERT INTO TABLE ventas_particionada PARTITION (year, month) SELECT InvoiceNo, StockCode, Description, Quantity,
from_unixtime(unix_timestamp(InvoiceDate,'d/M/yyyy H:mm')) AS InvoiceDate, UnitPrice, CustomerID, Country, year(from_unixtime(unix_timestamp(InvoiceDate,'d/M/yyyy H:mm'))) AS year, month(from_unixtime(unix_timestamp(InvoiceDate,'d/M/yyyy H:mm'))) AS month FROM ventas_raw;
```

```
Query ID = hadoop_2025121710412_32090819-ded8-44bd-bee2-6082b2adde0b
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  

 VERTICES      MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED    1        1        0        0        0        0  

Reducer 2 .... container  SUCCEEDED    2        2        0        0        0        0  

Reducer 3 .... container  SUCCEEDED    2        2        0        0        0        0  

-----  

VERTICES: 03/03  [=====>>>] 100%  ELAPSED TIME: 19.05 s  

-----  

Loading data to table retail.ventas_particionada partition (year=null, month=null)  

Loaded : 13/13 partitions.  

Time taken to load dynamic partitions: 2.199 seconds  

Time taken for adding to write entity : 0.003 seconds  

OK  

Time taken: 31.392 seconds
```

Finalmente, vamos a comprobar que se hizo correctamente

```
hive> SHOW PARTITIONS ventas_particionada;
OK
year=2010/month=12
year=2011/month=1
year=2011/month=10
year=2011/month=11
year=2011/month=12
year=2011/month=2
year=2011/month=3
year=2011/month=4
year=2011/month=5
year=2011/month=6
year=2011/month=7
year=2011/month=8
year=2011/month=9
Time taken: 0.136 seconds, Fetched: 13 row(s)
```

# Apartado E

## 1.) 10 clientes con mayor gasto total

```
hive> SELECT CustomerID, SUM(Quantity*UnitPrice) AS total_gasto FROM ventas_particionada GROUP BY CustomerID ORDER BY total_gasto DESC LIMIT 10;
Query ID = hadoop_20251217110928_30810056-7d6d-4d32-a65a-0158e39a869e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  
 VERTICES    MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  
-----  
Map 1 ..... container  SUCCEEDED   1       1       0       0       0       0  
Reducer 2 ..... container  SUCCEEDED   2       2       0       0       0       0  
Reducer 3 ..... container  SUCCEEDED   1       1       0       0       0       0  
-----  
VERTICES: 03/03  [=====>>] 100%  ELAPSED TIME: 7.88 s
```

17450	10496.0
14911	4364.0
15581	2750.0
12471	2400.0
14607	2120.0
12540	1820.0
12748	1788.0
15482	1646.0
14646	1458.0
12681	1422.0

## 2.) Clientes con más compras (cantidad de facturas)

```
hive> SELECT CustomerID, COUNT(DISTINCT InvoiceNo) AS facturas FROM ventas_particionada GROUP BY CustomerID ORDER BY facturas DESC LIMIT 10;
Query ID = hadoop_20251217111045_5d2d51dc-f990-4fc8-8922-55d51c030d17
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  
 VERTICES    MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  
-----  
Map 1 ..... container  SUCCEEDED   1       1       0       0       0       0  
Reducer 2 ..... container  SUCCEEDED   2       2       0       0       0       0  
Reducer 3 ..... container  SUCCEEDED   1       1       0       0       0       0  
-----  
VERTICES: 03/03  [=====>>] 100%  ELAPSED TIME: 7.97 s
```

14911	85
12569	31
12682	29
12681	22
12471	21
12621	18
14646	18
12540	17
12474	16
12437	15

### 3.) 10 productos más vendidos

```
hive> SELECT Description, SUM(Quantity) AS total_vendido FROM ventas_particionada GROUP BY Description ORDER BY total_vendido DESC LIMIT 10;
Query ID = hadoop_20251217111139_e82b481f-cd6e-4874-b7c7-9445d7eb3523
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  

 VERTICES      MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED    1        1        0        0        0        0        0  

Reducer 2 ..... container  SUCCEEDED    2        2        0        0        0        0        0  

Reducer 3 ..... container  SUCCEEDED    1        1        0        0        0        0        0  

-----  

VERTICES: 03/03  [=====>>] 100%  ELAPSED TIME: 7.57 s
```

POSTAGE 3110	
PINK HANGING HEART T-LIGHT HOLDER	480
METAL TUBE CHIME ON BAMBOO	392
VICTORIAN SEWING BOX SMALL	240
FLOWERS CHANDELIER T-LIGHT HOLDER	240
BEADED CHANDELIER T-LIGHT HOLDER	240
IVORY CHANDELIER T-LIGHT HOLDER	240
Manual	221
VANILLA SCENT CANDLE JEWELLED BOX	216
RED RETROSPOT CAKE STAND	204

### 4.) 10 Productos más rentables (suma de precio unitario por cantidad)

```

hive> SELECT Description, SUM(Quantity*UnitPrice) AS total_rentable FROM ventas_particionada GROUP BY Description ORDER
BY total_rentable DESC LIMIT 10;
Query ID = hadoop_20251217111251_8d58f262-380d-4965-9072-37d3e8c7b428
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  

  VERTICES    MODE     STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

Map 1 ..... container    SUCCEEDED    1      1      0      0      0      0  

Reducer 2 ..... container    SUCCEEDED    2      2      0      0      0      0  

Reducer 3 ..... container    SUCCEEDED    1      1      0      0      0      0  

-----  

VERTICES: 03/03  [=====>>>] 100% ELAPSED TIME: 7.88 s

```

POSTAGE	69012.0
VINTAGE RED KITCHEN CABINET	8125.0
CARRIAGE	6686.0
LOVE SEAT ANTIQUE WHITE METAL	6210.0
Manual	5447.0
RUSTIC SEVENTEEN DRAWER SIDEBOARD	5415.0
VINTAGE BLUE KITCHEN CABINET	3685.0
CHEST NATURAL WOOD 20 DRAWERS	2745.0
BOTANICAL GARDENS WALL CLOCK	1975.0
CINDERELLA CHANDELIER	1800.0

## 5.) Países con mayor volumen de ventas Análisis temporal

```

hive> SELECT Country, SUM(Quantity*UnitPrice) AS ventas_totales FROM ventas_particionada GROUP BY Country ORDER BY ventas_totales DESC;
Query ID = hadoop_20251217111348_cf46a110-1dd2-4e79-9c2e-e3ec94e3542f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  

  VERTICES    MODE     STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

Map 1 ..... container    SUCCEEDED    1      1      0      0      0      0  

Reducer 2 ..... container    SUCCEEDED    2      2      0      0      0      0  

Reducer 3 ..... container    SUCCEEDED    1      1      0      0      0      0  

-----  

VERTICES: 03/03  [=====>>>] 100% ELAPSED TIME: 7.19 s

```

United Kingdom	53159.0
Germany	21155.0
France	15713.0
Spain	5917.0
EIRE	5154.0
Belgium	4269.0
Switzerland	4027.0
Finland	3673.0
Norway	2996.0
Portugal	2508.0
Netherlands	1947.0
Italy	1663.0
Sweden	1539.0
Austria	1456.0
Cyprus	1265.0
Denmark	744.0
Malta	655.0
Channel Islands	443.0
Poland	360.0
Australia	350.0
Greece	335.0
Singapore	315.0
Israel	255.0
European Community	141.0
Czech Republic	40.0

## 6. Ventas totales por mes (suma de precio unitario por cantidad)

```

hive> SELECT year(InvoiceDate) AS year, month(InvoiceDate) AS month, SUM(Quantity*UnitPrice) AS total FROM ventas_particionada GROUP BY year(InvoiceDate), month(InvoiceDate) ORDER BY year, month;
Query ID = hadoop_20251217111455_a31a6180-8b28-4ae5-aa67-9e7f4ba8c85a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----  

      VERTICES    MODE     STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED  

-----  

Map 1 ..... container  SUCCEEDED   1       1       0       0       0       0  

Reducer 2 ..... container SUCCEEDED   2       2       0       0       0       0  

Reducer 3 ..... container SUCCEEDED   1       1       0       0       0       0  

-----  

VERTICES: 03/03  [=====>>] 100%  ELAPSED TIME: 8.63 s
-----
```

2010	12	9101.0
2011	1	8943.0
2011	2	5794.0
2011	3	11241.0
2011	4	7338.0
2011	5	8336.0
2011	6	8766.0
2011	7	6415.0
2011	8	12021.0
2011	9	13196.0
2011	10	12845.0
2011	11	22760.0
2011	12	3323.0

## 7. Hora del día con más actividad

```
hive> SELECT hour(InvoiceDate) AS hora, COUNT(*) AS total FROM ventas_particionada GROUP BY hour(InvoiceDate) ORDER BY t
otal DESC LIMIT 1;
Query ID = hadoop_20251217111555_2f4f6cd3-267c-47e7-944c-1a889f2bb768
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1765966144432_0004)

-----
```

VERTICES	MODE	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1 .....	container	SUCCEEDED	1	1	0	0	0	0
Reducer 2 .....	container	SUCCEEDED	2	2	0	0	0	0
Reducer 3 .....	container	SUCCEEDED	1	1	0	0	0	0

```
VERTICES: 03/03 [=====>>] 100% ELAPSED TIME: 8.58 s
```

10            251

# Apartado F

**1.) En tu servidor MySql en la máquina EC2 crea una base de datos y una tabla para almacenar los datos del fichero ventas.csv.**

Ahora para almacenar los datos en una tabla de mi base de datos que tengo en “EC2”, voy a usar la siguiente

The screenshot shows the AWS Management Console with the EC2 service selected. On the left, there's a navigation pane with options like Panel, Vista global de EC2, Eventos, Instancias (selected), and Tipos de instancia. The main area is titled "Instancias (1/5) Información". It includes a search bar and a dropdown for "Todos los estados". A table lists three instances:

	Name	ID de la instancia	Estado de la i...	Tipo de inst...	Comprobación de
<input checked="" type="checkbox"/>	Ubuntu_MySql	i-05629e587ad36a7c3	En ejecución	t2.micro	2/2 comprobacion
<input type="checkbox"/>		i-0ba836abe166f4241	En ejecución	m4.large	Inicializando
<input type="checkbox"/>		i-027a93c3e84150692	Pendiente	m4.large	-

Primero de todo, me conecto a la instancia que tengo creada

```
C:\Users\Mañana>ssh -i "C:\Users\Mañana\Downloads\ubuntu.pem" ubuntu@54.86.196.253
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 6.8.0-1043-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Thu Dec 18 07:45:55 UTC 2025

System load:  0.0          Processes:           103
Usage of /:   43.7% of 7.57GB  Users logged in:     0
Memory usage: 57%          IPv4 address for eth0: 172.31.21.121
Swap usage:   0%          

Expanded Security Maintenance for Applications is not enabled.

6 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Dec  2 07:44:52 2025 from 158.99.18.30
ubuntu@ip-172-31-21-121:~$ |
```

Una vez conectado, voy a crear la base de datos, para ello entramos en “mysql”

```
ubuntu@ip-172-31-21-121:~$ sudo mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 12
Server version: 8.0.44-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> |
```

Una vez dentro, creamos la base de datos y las tablas

```
mysql> CREATE DATABASE retaildb;
Query OK, 1 row affected (0.01 sec)

mysql> USE retaildb;
Database changed
```

```
mysql> CREATE TABLE ventas ( InvoiceNo VARCHAR(20), StockCode VARCHAR(20), Description VARCHAR(255), Quantity INT, InvoiceDate DATETIME, UnitPrice DOUBLE, CustomerID INT, Country VARCHAR(50) );
Query OK, 0 rows affected (0.06 sec)
```

Finalmente, permitimos accesos remotos

```
mysql> GRANT ALL PRIVILEGES ON retaildb.* TO 'admin'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)
```

## 2.) Exporta con SQOOP los datos de ventas.csv a la tabla que creaste en el punto anterior.

```
[hadoop@ip-172-31-58-155 ~]$ sqoop export --connect jdbc:mysql://54.86.196.253:3306/retaildb --username admin --password admin --table ventas --export-dir /user/dani-gayol/ventas/ventas_limpias.csv --fields-terminated-by ';' --lines-terminated-by '\n'
```