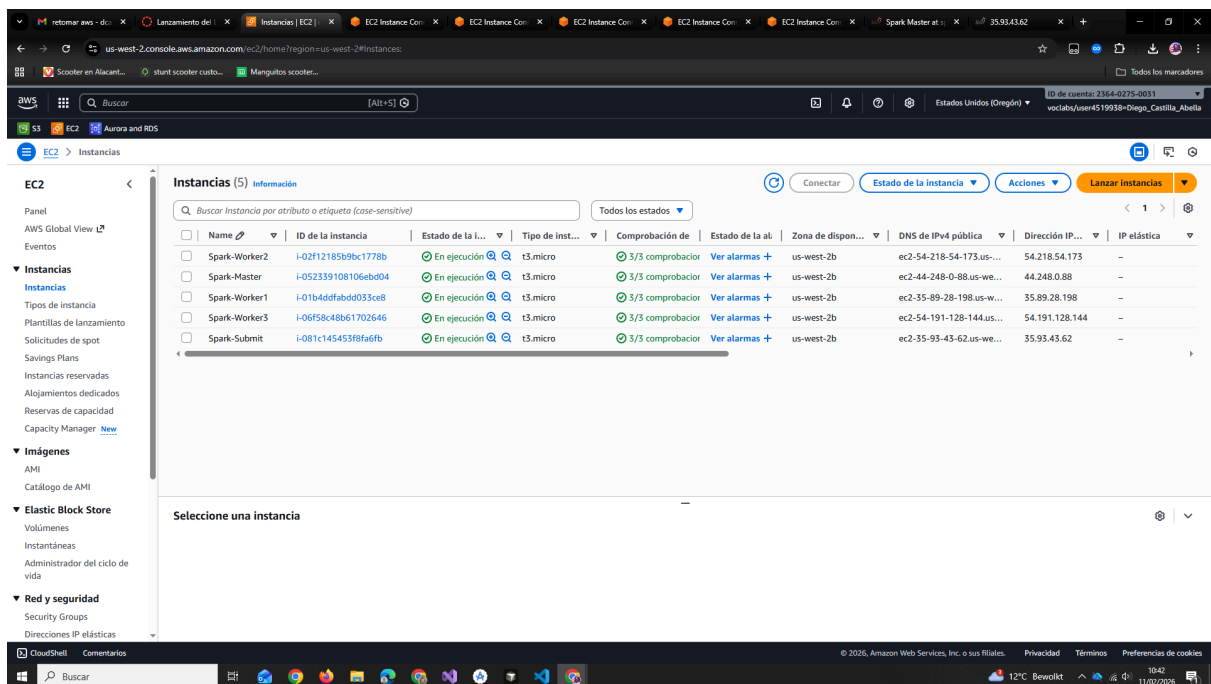


# 1. Preparación de la Infraestructura en AWS

El primer paso consistió en el despliegue de 5 instancias EC2 independientes para garantizar la separación de roles:

- Spark Master: Actúa como el orquestador de recursos del clúster.
- Spark Workers: Son los nodos encargados de realizar el procesamiento pesado de datos de forma paralela.
- Submit: La EC2 que utilizaremos para lanzar las aplicaciones mediante el comando spark-submit.
- Configuración de Red: Se definieron Security Groups para permitir el tráfico en los puertos 7077 (RPC de Spark), 8080 (Web UI), y 3306 (acceso a la base de datos RDS).



## 2. Configuración del Entorno y Resolución de Dependencias

- Una vez operativas las máquinas, se instaló Spark 4.0.1 que al ser una versión tan reciente las librerías de hadoop aun no estaban integradas por lo que tuvimos que hacer el downgrade a la versión 3.5.0 la cual si incluye estos paquetes nativamente, ademas probe primero solo a hacer el downgrade en master y en submit pero entonces los workers no conectaban ya que spark a la hora de comunicarse es muy riguroso con las versiones del mismo. Hacemos downgrade a todo a 3.5.0 y estos ya se conectan y aparecen en el panel web del nodo master 8080.

Una vez con el script python : Spark no podía comunicarse con S3 nativamente. para solucionario se descargaron e integraron manualmente los archivos JAR hadoop-aws-3.3.4.jar y aws-java-sdk-bundle-1.12.262.jar para permitir el uso del protocolo s3a://. Además tambien se incluyó el driver de MySQL para permitir que Spark escribiera los resultados finales en la base de datos RDS.

The screenshot shows the Spark Master web interface at `spark://ip-172-31-30-219.us-west-2.compute.internal:7077`. The interface displays the following information:

- URL:** `spark://ip-172-31-30-219.us-west-2.compute.internal:7077`
- Alive Workers:** 3
- Cores In use:** 6 Total, 6 Used
- Memory In use:** 3.0 GiB Total, 3.0 GiB Used
- Resources In use:**
- Applications:** 1 Running, 4 Completed
- Drivers:** 0 Running, 0 Completed
- Status:** ALIVE

**Workers (5)**

| Worker ID                                 | Address             | State | Cores      | Memory                       | Resources |
|---|---------------------|-------|------------|------------------------------|-----------|
| worker-20260211091034-172.31.18.227-42921 | 172.31.18.227:42921 | DEAD  | 2 (0 Used) | 1024.0 MiB (0.0 B Used)      |           |
| worker-20260211092224-172.31.18.227-44979 | 172.31.18.227:44979 | ALIVE | 2 (2 Used) | 1024.0 MiB (1024.0 MiB Used) |           |
| worker-20260211092434-172.31.28.92-39393  | 172.31.28.92:39393  | ALIVE | 2 (2 Used) | 1024.0 MiB (1024.0 MiB Used) |           |
| worker-20260211092445-172.31.25.64-41763  | 172.31.25.64:41763  | DEAD  | 2 (0 Used) | 1024.0 MiB (0.0 B Used)      |           |
| worker-20260211092732-172.31.25.64-39297  | 172.31.25.64:39297  | ALIVE | 2 (2 Used) | 1024.0 MiB (1024.0 MiB Used) |           |

**Running Applications (1)**

| Application ID          | Name                | Cores | Memory per Executor | Resources Per Executor | Submitted Time      | User     | State   | Duration |
|-------------------------|---------------------|-------|---------------------|------------------------|---------------------|----------|---------|----------|
| app-20260211093215-0004 | (kill) Top Products | 6     | 1024.0 MiB          |                        | 2026/02/11 09:32:15 | ec2-user | RUNNING | 11 s     |

**Completed Applications (4)**

| Application ID          | Name         | Cores | Memory per Executor | Resources Per Executor | Submitted Time      | User     | State    | Duration |
|-------------------------|--------------|-------|---------------------|------------------------|---------------------|----------|----------|----------|
| app-20260211092945-0003 | Top Products | 6     | 1024.0 MiB          |                        | 2026/02/11 09:29:45 | ec2-user | FINISHED | 40 s     |
| app-20260211092749-0002 | Top Products | 6     | 1024.0 MiB          |                        | 2026/02/11 09:27:49 | ec2-user | FINISHED | 41 s     |
| app-20260211092522-0001 | Top Products | 6     | 1024.0 MiB          |                        | 2026/02/11 09:25:22 | ec2-user | FINISHED | 41 s     |
| app-20260211091327-0000 | Top Products | 2     | 1024.0 MiB          |                        | 2026/02/11 09:13:27 | ec2-user | FINISHED | 28 s     |

us-west-2-console.aws.amazon.com/rds/home/region=us-west-2?databaseid=myrds-cluster=fa...

AWS Aurora and RDS

myrds

Resumen

Identificador de base de datos: myrds

Estado: Disponible

Rol: Instancia

Motor: MySQL Community

Recomendaciones: 4 Informativo

CPU: 6.04%

Clase: db.t3.micro

Actividad actual: 0 Conexiones

Región y AZ: us-west-2a

Conectividad y seguridad

Conectarse mediante: Fragmentos de código

Puerta de enlace a Internet: Desactivado

Lenguaje de programación: MySQL (macOS)

Tipo de punto de conexión: Punto de conexión de la instancia

Pasos de conexión

```
1 mysql -h myrds.c0uzhtegpio.us-west-2.rds.amazonaws.com -P 3306 -u admin -p 'Enter_DB_Password' --ssl-verify-server-cert --ssl-ca=/certs/global-bundle.pem mysql
```

Recursos de computación conectados (0)

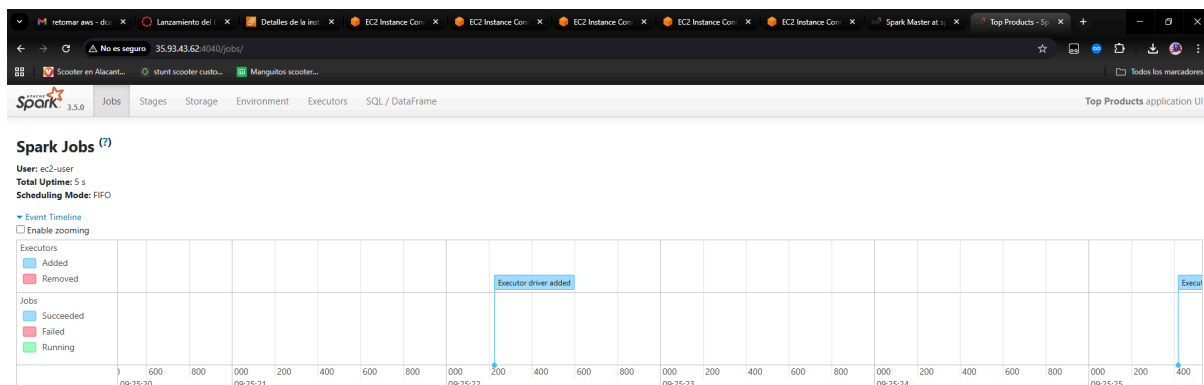
us-west-2-console.aws.amazon.com/ec2-instance-connect/home/region=us-west-2?connType=standard&instanceId=i-081c145453f8fa6fb&osUser=ec2-user&sshPort=22&addressFamily=ipv4

AWS CloudShell

```
+ Window [row_number() window specification (fecha#80, importe_total#108 DESC NULLS LAST, specified window frame (row frame, unbounded preceding (), current row ())) AS ranking#114], [fecha#80], [importe_total#108] DESC NULLS LAST)
+ Sort [fecha#80 ASC NULLS FIRST, importe_total#108 DESC NULLS LAST], false, 0
+ Exchange hashpartitioning([fecha#80, 200], ENSURE_REQUIREMENTS, [plan_id#745])
+ WindowGroupLimit [fecha#80], [importe_total#108 DESC NULLS LAST], row_number(), 10, Partial
+ Sort [fecha#80 ASC NULLS FIRST, importe_total#108 DESC NULLS LAST], false, 0
+ HashAggregate(key=[fecha#80, product_id#46], function=[sum(quantity#47), sum(importe_total#68)], output=[fecha#80, product_id#46, unidades#106L, importe_total#108])
+ Exchange hashpartitioning([fecha#80, product_id#46, 200], ENSURE_REQUIREMENTS, [plan_id#739])
+ HashAggregate(key=[fecha#80, product_id#46], function=[partial_sum(quantity#47), partial_sum(importe_total#68)], output=[fecha#80, product_id#46, sum#133L, sum#134])
+ Project [product_id#46, quantity#47, (cast(quantity#47 as double) * unit_price#48) AS importe_total#6, date_format(cast(order_date#20 as timestamp), yyyy-MM-dd, Some(UTC)) AS fecha#80]
+ BroadcastHashJoin [order_id#17], [order_id#45], Inner, BuildInfo: false
+ BroadcastExchange HashAggregateBroadcastMode(List(Cast(input[0, int, false] as bigint)), false), [plan_id#734]
+ Filter isnotnull(order_id#17)
+ FileScan csv [order_id#17, order_date#20] Batched: false, DataFilters: [isnotnull(order_id#17)], Format: CSV, Location: InMemoryFileIndex(1 paths) [s3a://mybucket-comercio360/comercio360/raw/orders.csv], PartitionFilters: [], PushedFilters: [IsNotNull(order_id)], ReadSchema: struct<order_id:int,order_date:date>
+ FileScan csv [order_id#45,product_id#46,quantity#47,unit_price#48] Batched: false, DataFilters: [isnotnull(order_id#45)], Format: CSV, Location: InMemoryFileIndex(1 paths) [s3a://mybucket-comercio360/comercio360/raw/order_items.csv], PartitionFilters: [], PushedFilters: [IsNotNull(order_id)], ReadSchema: struct<order_id:int,product_id:int,quantity:int,unit_price:double>
```

¡-081c145453f8fa6fb (Spark-Submit)

PublicIPs: 55.93.43.62 PrivateIPs: 172.31.22.250



### 3. Procesamiento de Datos (Flujo ETL)

El proceso de datos siguió el pipeline obligatorio S3 → Spark → RDS:

1. **Ingesta** : Spark leyó orders.csv y order\_items.csv directamente desde el bucket de Amazon S3.
2. **Procesamiento**:
  - Se realizó un Join entre los pedidos y sus líneas de detalle para asociar productos y precios.
  - Se calcularon las ventas totales por producto y fecha.
  - Se aplicó una función de ventana (Window) para generar un ranking y filtrar únicamente el Top 10 diario por facturación.
3. **Persistencia**: Los resultados se guardaron en S3 en formato CSV para auditoría.
  - Simultáneamente, los datos se insertaron en la tabla top\_products de Amazon RDS mediante una conexión JDBC en modo overwrite.

**Details for Query 2**

Submitted Time: 2026/02/11 09:32:39  
Duration: 8 s  
Succeeded Jobs: 4 5 6 7

☒ Show the Stage ID and Task ID that corresponds to the max metric

▼ Details

```
== Physical Plan ==
Execute SaveIntoDataSourceCommand (1)
  +- SaveIntoDataSourceCommand (2)
    +- Filter (14)
      +- Project (13)
        +- Project (12)
          +- Window (11)
            +- Project (10)
              +- Aggregate (9)
                +- Project (8)
                  +- Project (7)
                    +- Project (6)
                      +- Join (5)
                        +- LogicalRelation (3)
                          +- LogicalRelation (4)

(1) Execute SaveIntoDataSourceCommand
Output: []

(2) SaveIntoDataSourceCommand
Arguments: org.apache.spark.sql.execution.datasources.jdbc.JdbcRelationProvider@41a78c29, [url=***** (redacted), driver=com.mysql.cj.jdbc.Driver, database=top_products, user=admin, password=***** (redacted)], Overwrite

(3) LogicalRelation
Arguments: csv, [order_id#17, customer_id#18, store_id#19, order_date#20, payment_method#21], false

(4) LogicalRelation
Arguments: csv, [order_item_id#44, order_id#45, product_id#46, quantity#47, unit_price#48, discount#49], false

(5) Join
Arguments: Inner, (order_id#17 = order_id#45)

(6) Project
Arguments: [order_id#17, customer_id#18, store_id#19, order_date#20, payment_method#21, order_item_id#44, product_id#46, quantity#47, unit_price#48, discount#49]

(7) Project
Arguments: [order_id#17, customer_id#18, store_id#19, order_date#20, payment_method#21, order_item_id#44, product_id#46, quantity#47, unit_price#48, discount#49, (cast(quantity#47 as double) * unit_price#48) AS importe_total#68]
```

## 4. Análisis de Ejecución (DAG)

A través de la Spark Web UI, se analizó el Grafo Acíclico Dirigido (DAG) de la consulta:

- El sistema dividió el trabajo en varios Stages (etapas) para optimizar el movimiento de datos entre los workers.
- Se observó un nodo de tipo Exchange para agrupar los datos distribuidos por todo el clúster.
- Finalmente, el comando `SaveIntoDataSourceCommand` validó que el flujo terminó con éxito enviando los datos a la base de datos externa.

mybucket-comercio360 Bucket

us-west-2.console.aws.amazon.com/s3/buckets/mybucket-comercio360?region=us-west-2&prefix=comercio360/resultado\_top\_productos/&showversions=false

Amazon S3

resultado\_top\_productos/

Objetos

Objetos (2)

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener una lista de todos los objetos de su bucket. Para que otras personas obtengan acceso a sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

Buscar objetos por prefijo

| <input type="checkbox"/> | Nombre   | Tipo | Última modificación         | Tamaño   | Clase de almacenamiento |
|--------------------------|--|------|-----------------------------|----------|-------------------------|
| <input type="checkbox"/> | <a href="#">_SUCCESS</a>   | -    | 11 Feb 2026 10:13:56 AM CET | 0 B      | Estándar                |
| <input type="checkbox"/> | <a href="#">part-00000-58336af3-b212-468b-8af6-1b3f7228a0f5-c000.csv</a> | csv  | 11 Feb 2026 10:13:55 AM CET | 102.3 KB | Estándar                |

mybucket-comercio360 Bucket

us-west-2.console.aws.amazon.com/s3/buckets/mybucket-comercio360?region=us-west-2&prefix=comercio360/&showversions=false

Amazon S3

comercio360/

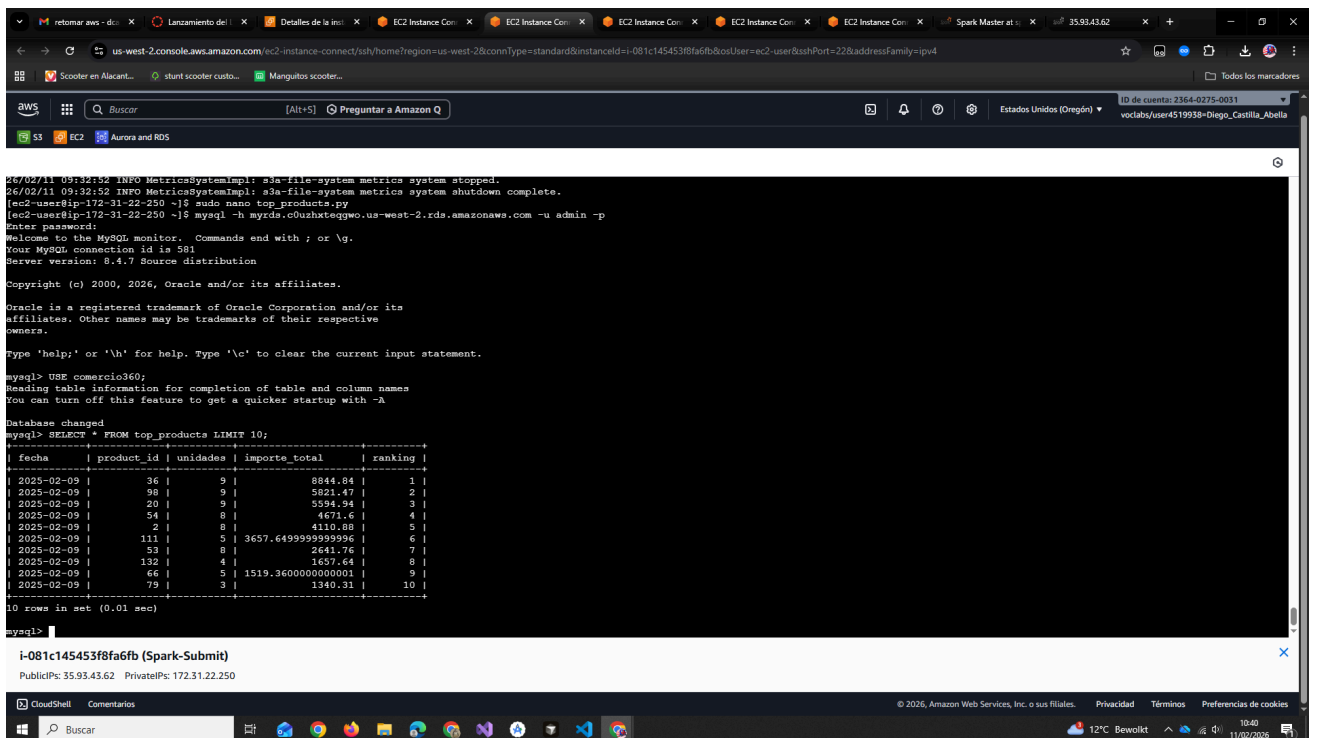
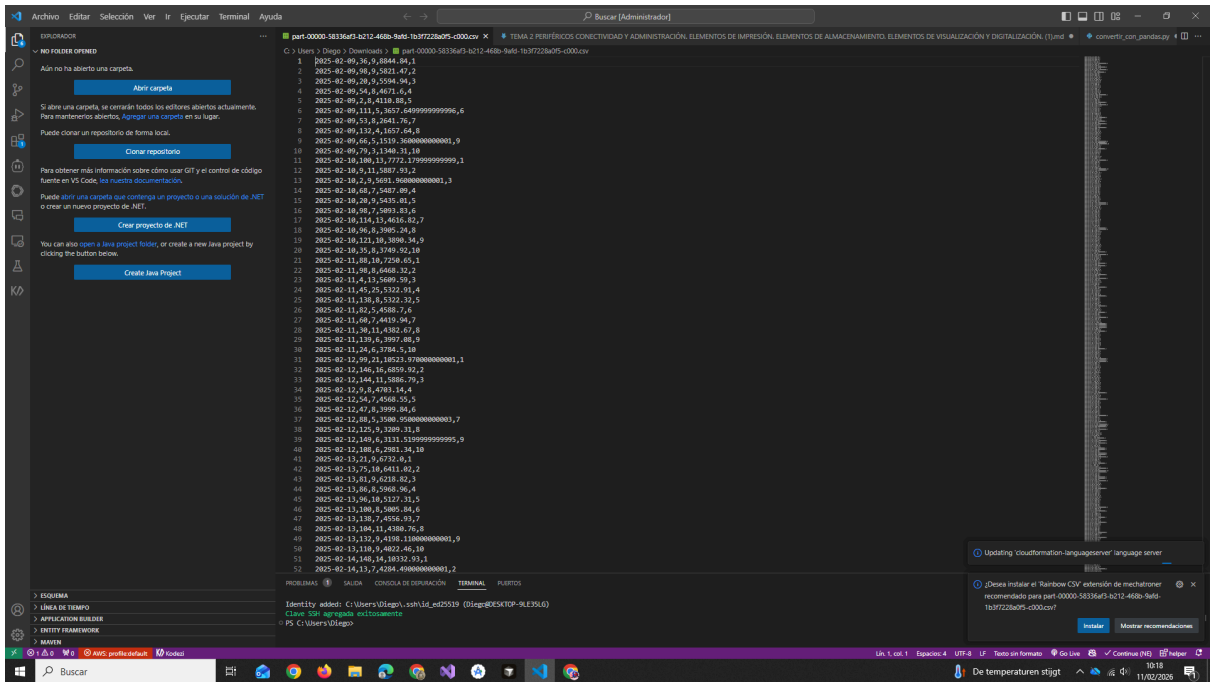
Objetos

Objetos (2)

Los objetos son las entidades fundamentales que se almacenan en Amazon S3. Puede utilizar el [inventario de Amazon S3](#) para obtener una lista de todos los objetos de su bucket. Para que otras personas obtengan acceso a sus objetos, tendrá que concederles permisos de forma explícita. [Más información](#)

Buscar objetos por prefijo

| <input type="checkbox"/> | Nombre                                   | Tipo    | Última modificación | Tamaño | Clase de almacenamiento |
|--------------------------|--|---------|---------------------|--------|-------------------------|
| <input type="checkbox"/> | <a href="#">raw/</a>                     | Carpeta | -                   | -      | -                       |
| <input type="checkbox"/> | <a href="#">resultado_top_productos/</a> | Carpeta | -                   | -      | -                       |



## 5. Verificación de Resultados

El proceso finalizó en ambos servicios:

- **En S3:** Se generaron los archivos de resultados con la estructura de fecha, producto e importe.
- **En RDS:** Se ejecutó una consulta SQL (`SELECT * FROM top_products LIMIT 10;`) que confirmó que los 10 mejores productos diarios están correctamente almacenados y disponibles para aplicaciones de negocio.