

# Documentación Proyecto Hadoop + Hive + Spark

---

## Estructura del Proyecto

```
Proyecto-hive/
|
├── docker-compose.yml      # Orquestación de servicios Hadoop + Hive + Spark
├── Dockerfile              # Imagen extendida de Hive con driver PostgreSQL
├── hadoop.env               # Variables de entorno de Hadoop
├── hive.env                 # Variables de entorno de Hive
├── hive-site.xml            # Configuración de conexión de Hive al Metastore (PostgreSQL)
├── postgresql-42.7.5.jar    # Driver JDBC de PostgreSQL
├── docker                   # (archivo vacío que no usamos, se puede borrar)
└── app/                      # Carpeta para notebooks o scripts Spark (Python/Scala)
```

### ❖ Función de cada archivo

- docker-compose.yml: Define servicios Hadoop core (namenode, datanode, etc.), Hive (metastore + server), PostgreSQL y Spark. Configura puertos, volúmenes y dependencias.
- Dockerfile: Extiende la imagen de Hive (bde2020/hive) para incluir el driver PostgreSQL en /opt/hive/lib/.
- hadoop.env: Variables de entorno de Hadoop: CLUSTER\_NAME, configuración HDFS, YARN, etc.
- hive.env: Variables de Hive, conecta Hive con PostgreSQL y HDFS.
- hive-site.xml: Configuración crítica para el metastore. Define driver, URL de conexión, usuario y contraseña.
- postgresql-42.7.5.jar: Driver JDBC para conexión de Hive con PostgreSQL.
- app/: Scripts Spark o notebooks para procesar datos y construir el pipeline Bronze → Silver → Gold.

## ◊ Flujo de trabajo

### 1. Levantar servicios en orden:

- Namenode y Datanode
- ResourceManager y NodeManager
- HistoryServer
- PostgreSQL (para metastore)
- Hive Metastore (con schema inicializado en PostgreSQL)
- Hive Server2
- Spark Master y Workers

### 2. HDFS:

- Crear carpetas /bronze, /silver, /gold
- Cargar datasets: docker cp + hdfs dfs -put

### 3. Hive:

- Crear tablas externas apuntando a carpetas en HDFS
- Ejemplo:  
CREATE EXTERNAL TABLE movies (...)  
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','  
STORED AS TEXTFILE  
LOCATION '/data/input';

### 4. Spark (carpeta app/):

- Scripts que limpian datos en Bronze y escriben en Silver
- Transformaciones adicionales hacia Gold
- Consultas desde Hive para validación

## ◊ Comandos útiles

### - Ver contenedores:

```
docker ps
```

### - Subir archivo local al contenedor namenode:

```
docker cp "C:/ruta/movie.csv" namenode:/tmp/movie.csv
```

### - Mover archivo a HDFS:

```
hdfs dfs -mkdir -p /data/input  
hdfs dfs -put /tmp/movie.csv /data/input
```

### - Ver contenido en HDFS:

```
hdfs dfs -ls /data/input
```

- Conexión Hive:

```
beeline -u jdbc:hive2://localhost:10000
```

- Consulta rápida:

```
show databases;  
show tables;  
select count(*) from movies;
```

## Documento Técnico del Pipeline

### 1. Arquitectura

- **Fuente de datos (Source):** SQL Server on-premise/local
- **Ingestión:** Spark (conector JDBC + driver mssql-jdbc)
- **Almacenamiento inicial (Bronze):** HDFS → datos crudos en formato **Parquet (Snappy)**
- **Procesamiento (Silver):** Spark limpia, estandariza y guarda datos listos para consumo analítico
- **Almacenamiento intermedio (Silver):** HDFS → datos refinados en Parquet
- **Capa de consulta:** Hive External Tables para análisis SQL

### 2. Contenedores (Docker Compose)

- spark-master → ejecuta jobs PySpark
- namenode, datanode → HDFS para almacenamiento
- hive-metastore + hive-server2 → consultas SQL externas
- resourcemanager y nodemanager → YARN para scheduling

### Scripts creados

#### ◆ **test\_connection.py**

- Objetivo: validar conexión entre Spark ↔ SQL Server
- Acciones:
  - Se conecta con jdbc:sqlserver://host.docker.internal:1433
  - Lista las tablas de la BD olva

◆ **extract\_clientes.py**

- Objetivo: extraer tabla Clientes desde SQL Server → HDFS Bronze
- Acciones:
  - Lee datos con .read.jdbc()
  - Guarda en /bronze/clientes en formato **Parquet Snappy**
  - Imprime registros de prueba

◆ **process\_clientes.py**

- Objetivo: transformar datos de Bronze → Silver
- Acciones:
  - Lee datos de /bronze/clientes
  - Normaliza nombres (quita tildes, espacios, etc.)
  - Elimina duplicados
  - Guarda en /silver/clientes

◆ **pipeline\_clientes.py**

- Objetivo: orquestar los pasos anteriores
- Acciones:
  1. Ejecuta extract\_clientes.py
  2. Espera 10s (para asegurar escritura)
  3. Ejecuta process\_clientes.py
  4. Confirma datos guardados en Silver

## 5. Problemas encontrados y soluciones

### Montaje de contenedores (Docker)

- **Error al levantar servicios con docker compose up (contenedores que se caían o quedaban en unhealthy)**  
Solución: revisar logs de cada contenedor (docker logs <container\_id>), corregir rutas de volúmenes, esperar a que el metastore de Hive inicialice antes de levantar hive-server2.

- **Problemas de red entre contenedores (Spark ↔ SQL Server local)**  
Solución: usar host.docker.internal como hostname en lugar de la IP local o nombre de máquina (DESKTOP-XXXX), habilitar el puerto 1433 en el firewall con New-NetFirewallRule.
- **No se encontraba el driver JDBC en Spark**  
Solución: montar el .jar (mssql-jdbc-13.2.0.jre8.jar) en la carpeta /opt/spark/jars del contenedor y añadirlo en la configuración de Spark (--jars y spark.jars).

## Ejecución de scripts PySpark

- **Error con f-strings (SyntaxError: invalid syntax)**  
Solución: contenedor trae Python 2.7 por defecto → reemplazar f-strings (f"..."") por .format() para compatibilidad.
- **Error inicial de conexión JDBC**  
Solución: confirmar que SQL Server estaba escuchando en 0.0.0.0:1433 y habilitar protocolo TCP/IP en SQL Server Configuration Manager.

## HDFS y Spark

- **No se visualizaban los datos desde el contenedor**  
Solución: usar hdfs dfs -ls /bronze/clientes desde namenode para listar y validar \_SUCCESS y archivos .parquet.
- **Procesamiento Silver no se ejecutaba después de extracción**  
Solución: añadir un sleep(10) en el pipeline para dar tiempo a Spark/HDFS de finalizar la escritura antes de leer en el siguiente paso.

## Captura de las Soluciones

Carpetas creadas en HDFS para el almacenamiento y procesamiento de los datos.

```
cat. unable to write to output stream.
root@6776e7a934dd:/# hdfs dfs -mkdir -p /data/bronze
root@6776e7a934dd:/# hdfs dfs -mkdir -p /data/silver
root@6776e7a934dd:/# hdfs dfs -mkdir -p /data/gold
root@6776e7a934dd:/# hdfs dfs -ls /data
Found 4 items
drwxr-xr-x  - root supergroup          0 2025-08-28 16:50 /data/bronze
drwxr-xr-x  - root supergroup          0 2025-08-28 16:50 /data/gold
drwxr-xr-x  - root supergroup          0 2025-08-28 16:41 /data/input
drwxr-xr-x  - root supergroup          0 2025-08-28 16:50 /data/silver
root@6776e7a934dd:/# hdfs dfs -ls /data/bronze
```

Creación de la base de datos en HIVE para el proyecto y creación de la tabla para probar funcionamiento de HIVE

## Ejecución del Pipeline de datos

```

Iniciando pipeline de datos de clientes...
Paso 1: Extrayendo datos de SQL Server a Bronze...
Ejecutando extract_clientes.py...
extract_clientes.py completado con exito
Datos extraidos de SQL Server:
+-----+-----+-----+-----+-----+
|ClienteID|Nombre |Email   |Telefono|CreateTime|UpdateTime
+-----+-----+-----+-----+-----+
|1    |Juan Pérez |juan.perez@email.com |089000100 |2025-08-24 21:27:20.383|2025-08-24 21:31:48.536
|2    |María López |maria22.nueva@email.com |0987654321 |2025-08-24 21:27:20.383|2025-08-24 21:31:59.697
|3    |Carlos Ruiz |carlos.ruiz@email.com |0971122334 |2025-08-24 21:27:20.383|null
|4    |Ana Torres |ana.torres@email.com |0965544332 |2025-08-24 21:27:20.383|null
|5    |Pedro Gómez |pedro.gomez@email.com |0956677889 |2025-08-24 21:27:20.383|null
+-----+-----+-----+-----+-----+

Total de registros: 5
Datos guardados en HDFS Bronze en: hdfs://namenode:8020/bronze/clientes

Esperando 10 segundos antes de procesar Bronze -> Silver...
Paso 2: Procesando datos de Bronze a Silver...
Ejecutando process_clientes.py...
process_clientes.py completado con exito
Datos leidos desde Bronze
Registros encontrados: 5
Procesamiento completado
Registros finales: 5
Datos guardados en: hdfs://namenode:8020/silver/clientes

Pipeline completado correctamente. Datos disponibles en Silver.
log4j:WARN No appenders could be found for logger (org.apache.spark.utilShutdownHookManager).
log4j:WARN Please initialize the log4j system properly.
log4j:WARN See http://logging.apache.org/log4j/1.2/faq.html#noconfig for more info.

C:\Users\DARY\Desktop\Darv\Prueba-BigData\Proyecto-hive>

```

Consulta a directorio de HDFS Bronce sobre los datos extraídos de SQL server de mi maquina local

```
C:\Users\DARY\Desktop\Dary\Prueba-BigData\Proyecto-hive>docker exec -it namenode hdfs dfs -ls /bronze/clientes
Found 2 items
-rw-r--r--  3 root supergroup      0 2025-08-28 20:58 /bronze/clientes/_SUCCESS
-rw-r--r--  3 root supergroup  1890 2025-08-28 20:58 /bronze/clientes/part-00000-5124d3fa-c05b-4834-b6af-db71a7b07caa-c00
0.snappy.parquet

C:\Users\DARY\Desktop\Dary\Prueba-BigData\Proyecto-hive>
```

Consulta al directorio Silver de HDFS a los datos Procesados desde Bronce

```
C:\Users\DAKY\Desktop\DAKY\Prueba-BigData\Proyecto-hive>docker exec -it namenode hdfs dfs -ls /silver/clientes
Found 7 items
-rw-r--r--  3 root supergroup          0 2025-08-28 20:45 /silver/clientes/_SUCCESS
-rw-r--r--  3 root supergroup    718 2025-08-28 20:45 /silver/clientes/part-00000-914bdf98-1085-4411-95b1-493debb1301e-c000.snappy.parquet
-rw-r--r--  3 root supergroup   1954 2025-08-28 20:45 /silver/clientes/part-00043-914bdf98-1085-4411-95b1-493debb1301e-c000.snappy.parquet
-rw-r--r--  3 root supergroup   1972 2025-08-28 20:45 /silver/clientes/part-00051-914bdf98-1085-4411-95b1-493debb1301e-c000.snappy.parquet
-rw-r--r--  3 root supergroup   1981 2025-08-28 20:45 /silver/clientes/part-00066-914bdf98-1085-4411-95b1-493debb1301e-c000.snappy.parquet
-rw-r--r--  3 root supergroup   1954 2025-08-28 20:45 /silver/clientes/part-00182-914bdf98-1085-4411-95b1-493debb1301e-c000.snappy.parquet
-rw-r--r--  3 root supergroup   2008 2025-08-28 20:45 /silver/clientes/part-00174-914bdf98-1085-4411-95b1-493debb1301e-c000.snappy.parquet

C:\Users\DAKY\Desktop\DAKY\Prueba-BigData\Proyecto-hive|
```

## Resultados en Hive y su tabla externa apuntado al directorio silver/clientes

```

: jdbc:hive2://localhost:10000> CREATE EXTERNAL TABLE IF NOT EXISTS silver.clientes (
    .->     ClienteID INT,
    .->     Nombre STRING,
    .->     Email STRING,
    .->     Telefono STRING,
    .->     CreateTime TIMESTAMP,
    .->     UpdateTime TIMESTAMP
    .-> )
    .-> STORED AS PARQUET
    .-> LOCATION 'hdfs://namenode:8020/silver/clientes';
o rows affected (0.588 seconds)
: jdbc:hive2://localhost:10000> USE silver;
o rows affected (0.333 seconds)
: jdbc:hive2://localhost:10000> SELECT * FROM clientes LIMIT 10;
-----+-----+-----+-----+-----+-----+
| clientes.clienteid | clientes.nombre | clientes.email | clientes.telefono | clientes.createetime | clientes.updatetime |
|-----+-----+-----+-----+-----+-----+
| 1 | Juan P?rez | juan.perez@email.com | 089000100 | 2025-08-24 21:27:20.383 | 2025-08-24 21:31:48.536
| 3 | Carlos Ruiz | carlos.ruiz@email.com | 0971122334 | 2025-08-24 21:27:20.383 | 2025-08-24 21:27:20.383
| 5 | Pedro G?mez | pedro.gomez@email.com | 0956677889 | 2025-08-24 21:27:20.383 | 2025-08-24 21:27:20.383
| 4 | Ana Torres | ana.torres@email.com | 0965544332 | 2025-08-24 21:27:20.383 | 2025-08-24 21:27:20.383
| 2 | Mar?a L?pez | maria22.nueva@email.com | 0987654321 | 2025-08-24 21:27:20.383 | 2025-08-24 21:31:59.697
|-----+-----+-----+-----+-----+-----+

```

## SQL Server Local con sus datos de pruebas

	ClienteID	Nombre	Email	Telefono	CreateTime	UpdateTime
1	1	Juan Pérez	juan.perez@email.com	089000100	2025-08-24 21:27:20.383	2025-08-24 21:31:48.536
2	2	Maria López	maria22.nueva@email.com	0987654321	2025-08-24 21:27:20.383	2025-08-24 21:31:59.697
3	3	Carlos Ruiz	carlos.ruiz@email.com	0971122334	2025-08-24 21:27:20.383	NULL
4	4	Ana Torres	ana.torres@email.com	0965544332	2025-08-24 21:27:20.383	NULL
5	5	Pedro Gómez	pedro.gomez@email.com	0956677889	2025-08-24 21:27:20.383	NULL

## Servicios levantados en Docker-compose

```

C:\Windows\System32\cmd.e x + x
C:\Users\DARY\Desktop\Dary\Prueba-BigData\Proyecto-hive>docker ps
CONTAINER ID IMAGE COMMAND NAMES CREATED STATUS PORTS
2ecdac3936f bde2020/spark-worker:2.4.5-hadoop2.7 "/bin/bash /worker.sh" 2 hours ago Up 2 hours 0.0.0.0:8
081->8081/tcp, [::]:8081->8081/tcp spark-worker-1
85239de85c57 bde2020/spark-master:2.4.5-hadoop2.7 "/bin/bash /master.sh" 2 hours ago Up 2 hours 0.0.0.0:7
7977f7077/tcp, [::]:7077->7077/tcp, 0.0.0.0:8080->8080/tcp, [::]:8080->8080/tcp spark-master
79cdcf342f56 hive-with-postgres "entrypoint.sh /bin/..." 6 hours ago Up 6 hours (healthy) 0.0.0.0:1
0000->10000/tcp, [::]:10000->10000/tcp
ede9b673641b hive-with-postgres "entrypoint.sh /opt/..." 6 hours ago Up 6 hours (healthy) 0.0.0.0:9
083->9083/tcp, [::]:9083->9083/tcp
cd70dbb8cbde postgres:13 "docker-entrypoint.s..." 6 hours ago Up 6 hours 0.0.0.0:5
432->5432/tcp, [::]:5432->5432/tcp
hive-metastore
hive-metastore-postgresql
f2864838cf65 bde2020/hadoop-resourcemanager:2.0.0-hadoop2.7.4-java8 "/entrypoint.sh /run..." 6 hours ago Up 6 hours (healthy) 0.0.0.0:8
088->8088/tcp, [::]:8088->8088/tcp resourcemanager
4feacc10f05c bde2020/hadoop-historyserver:2.0.0-hadoop2.7.4-java8 "/entrypoint.sh /run..." 6 hours ago Up 6 hours (healthy) 0.0.0.0:8
188->8188/tcp, [::]:8188->8188/tcp historyserver
f3dc6a2d95bf bde2020/hadoop-datanode:2.0.0-hadoop2.7.4-java8 "/entrypoint.sh /run..." 6 hours ago Up 6 hours (healthy) 0.0.0.0:9
876->50075/tcp, [::]:9876->50075/tcp datanode
6776e7a934dd bde2020/hadoop-namenode:2.0.0-hadoop2.7.4-java8 "/entrypoint.sh /run..." 6 hours ago Up 6 hours (healthy) 0.0.0.0:8
020->8020/tcp, [::]:8020->8020/tcp, 0.0.0.0:9871->50070/tcp namenode
C:\Users\DARY\Desktop\Dary\Prueba-BigData\Proyecto-hive>

```

## **Conclusiones**

Ha sido una experiencia enriquecedora que me permitió reafirmar la importancia de un Ingeniero de Datos o Desarrollador Big Data no radica únicamente en conocer las herramientas, sino en comprender cómo orquestar ecosistemas complejos para crear soluciones robustas y escalables.

Quiero expresar mi agradecimiento por esta oportunidad, ya que me permitió poner en práctica mis conocimientos y al mismo tiempo enfrentar nuevos retos dentro del ecosistema Big Data.

Más allá del resultado, valoro enormemente lo aprendido, pues cada error superado me permitió fortalecer mis capacidades como Ingeniero/Desarrollador Big Data. Así, incluso si no llego a quedar seleccionado, esta experiencia ya representa un paso importante en mi crecimiento profesional y una motivación extra para seguir avanzando en este camino.