

---

POSGRADOS UPS

MAESTRÍA EN SOFTWARE

DISEÑO DE ARQUITECTURAS DE SISTEMAS

---

Santiago David Cordero Crespo  
[s.cordero@est.ups.edu.ec](mailto:s.cordero@est.ups.edu.ec)

Introducción

## Integración fuera de linea

La empresa comercial XYZ tiene una cartera de clientes amplia a nivel nacional, más de 60 mil clientes, a sus clientes se les entrega una tarjeta de afiliación por medio de la cual los cliente realizan compras a crédito y tiene que de manera mensual realizar los pagos de sus compras, la cartera no se recupera de manera rápido teniendo deudas pendientes de cobrar, por lo que la empresa decide compartir información del historial de pagos con un empresa ML1 especializada en elaborar modelos de predicción para determinar si un cliente realizará o no el pago que le corresponde en el mes; la empresa ML1 le solicita información demográfica, la cantidad de días en los que realizó el pago, valores que tenia que pagar y el valor real del pago, posterior a establecer el intercambio de información, las empresas llegan a un acuerdo para que de manera mensual la empresa XYZ genere la información en archivos planos y se los coloque en un buzón STP de la empresa ML1, posterior la empresa ML1 lee la información de este archivo para cargar en una data de datos y realizar el procesamiento.

Diseñe e implemente por medio de la utilización un framework de integración el proceso de lectura del archivo en formato CSV para posterior a realizar algunas validaciones básicas se ingrese esta información a la base de datos.

- El valor de la factura a pagar no puede ser menor o igual a cero.
- El valor pagado no puede ser menor o igual a cero.

Al finalizar la carga se solicita que se presente un log o registros en el cual se especifique la cantidad de información que se carga y la que presento errores de validación

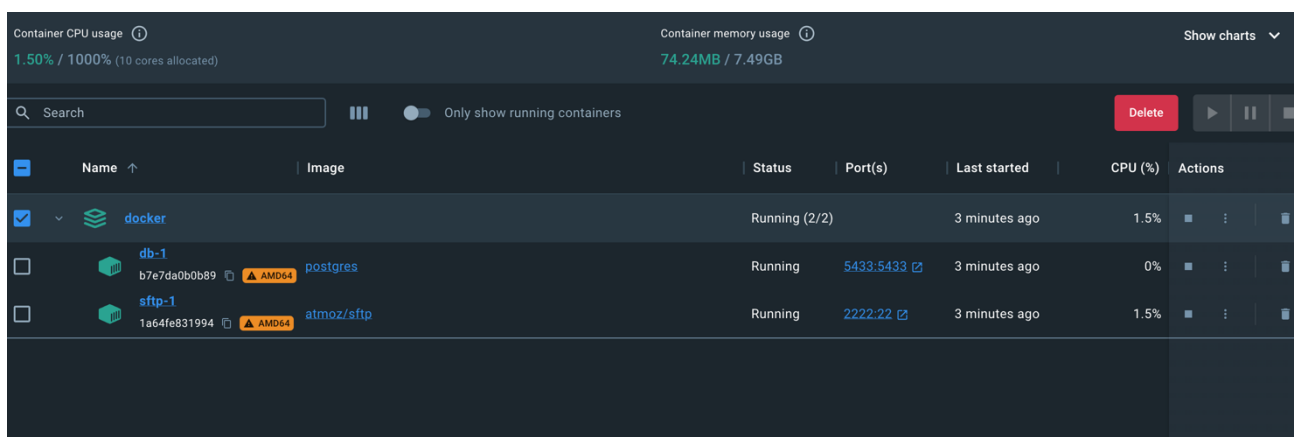
Desarrollo de la practica  
Para la práctica se utilizó:  
Máquina host una Mac M2 RAM 32 GB  
Base de datos Postgres  
SFTP

1. En la raíz del proyecto /first-camel-integration/src/Docker

docker-compose up

```
s.corderoc@desarrollo docker % pwd
/Users/s.corderoc/INTEGRACION_EMPRESARIAL/first-camel-integration/src/docker
s.corderoc@desarrollo docker % docker compose up
[+] Running 6/21
  :: postgres_sdcc 14 layers [#####] 0B/0B Pulling
    ✓ 8a1e25ce7c4f Pull complete
    ✓ 002317ed8722 Pull complete
    ✓ c223965bd9a8 Pull complete
    ✖ 847682431a68 Waiting
    ✖ 8d29ba654727 Waiting
    ✖ fd133663e42b Waiting
    ✖ 13de11c6ecda Waiting
    ✖ 45bb35744214 Waiting
    ✖ d4082e63ce2c Waiting
    ✖ 269f33c511c1 Waiting
    ✖ 7cbaf3c85093 Waiting
    ✖ f1c82efa0dcd Waiting
    ✖ e9d0d3c40657 Waiting
    ✖ 68bf5c580643 Waiting
  :: sftp 5 layers [#####] 35.69MB/71.42MB Pulling
    ✖ ec335f17d0c7 Downloading [=====] 21.52MB/55.08MB
    ✖ 29934b75d894 Downloading [=====] 14.17MB/16.33MB
    ✓ 237020ca00d8 Download complete
    ✓ b5e297aaaa79 Download complete
    ✓ 1db036b6a6df Download complete
```

2. Verificamos la creación de los contenedores



The screenshot shows the Docker Desktop interface. At the top, it displays 'Container CPU usage' at 1.50% / 1000% (10 cores allocated) and 'Container memory usage' at 74.24MB / 7.49GB. Below this is a search bar and a toggle for 'Only show running containers'. A table lists the containers:

Name	Image	Status	Port(s)	Last started	CPU (%)	Actions
docker		Running (2/2)		3 minutes ago	1.5%	[Stop] [Restart] [Delete]
db-1	b7e7da0b0b89 postgres	Running	5433:5433	3 minutes ago	0%	[Stop] [Restart] [Delete]
sftp-1	1a64fe831994 atm0z/sftp	Running	2222:22	3 minutes ago	1.5%	[Stop] [Restart] [Delete]

3. Ejecución del proyecto

Ingresamos al proyecto raíz

```
/first-camel-integration/
```

Empaquetamos el código

```
mvn clean package -DSkipTest
```

Finalmente ejecutamos

```
mvn camel:run
```

## 4. Resultados

```
[INFO] Starting Camel ...
[che.camel.learn.MainApp.main()] MainSupport INFO Apache Camel (Main) 4.0.4 is starting
Standard Commons Logging discovery in action with spring-jcl: please remove commons-logging.jar from classpath in order to avoid potential conflicts
[che.camel.learn.MainApp.main()] FileEndpoint INFO Endpoint is configured with noop=true so forcing endpoint to be idempotent as well
[che.camel.learn.MainApp.main()] FileEndpoint INFO Using default memory based idempotent repository with cache max size: 1000
[che.camel.learn.MainApp.main()] RemoteFileEndpoint INFO Endpoint is configured with noop=true so forcing endpoint to be idempotent as well
[che.camel.learn.MainApp.main()] RemoteFileEndpoint INFO Using default memory based idempotent repository with cache max size: 1000
[che.camel.learn.MainApp.main()] AbstractCamelContext INFO Apache Camel 4.0.4 (camel-1) is starting
[che.camel.learn.MainApp.main()] SftpOperations INFO Known host file not configured, using user known host file: /Users/s.corderoc/.ssh/known_hosts
[che.camel.learn.MainApp.main()] AbstractCamelContext INFO Routes startup (started:2)
[che.camel.learn.MainApp.main()] AbstractCamelContext INFO Started route1 (file://src/datos)
[che.camel.learn.MainApp.main()] AbstractCamelContext INFO Started route2 (sftp://localhost:2222/upload)
[che.camel.learn.MainApp.main()] AbstractCamelContext INFO Apache Camel 4.0.4 (camel-1) started in 662ms (build:0ms init:0ms start:662ms)
[che.camel.learn.MainApp.main()] AbstractCamelContext INFO Known host file not configured, using user known host file: /Users/s.corderoc/.ssh/known_hosts
[ ] thread #1 - file://src/datos SftpOperations
Total de filas a agregar: 15049
Filas Correctas: 15049
Filas Erroneas: 14952
```

Las filas agregadas son 15049

Verificamos que la información se insertó en la base de datos.

Propiedades
Datos
Diagrama ER

postgres
Bases de Datos
postgres
Esquemas
public
Tablas
myta

mytable
Enter a SQL expression to filter results (use Ctrl+Space)

	codigo	abc_id	abc_limite	abc_sexo	abc_educacion	abc_casado	abc_edad	abc_pago_0	abc_pago_2	abc_pago_3	abc_pago_4	abc_pago_5	abc_pago_6
1	3	90000	90000	2	2	34	0	1500	1000	1000	1000	5000	
2	2	4	50000	50000	2	1	37	0	2019	1200	1100	1069	1000
3	5	5	50000	50000	2	1	57	-1	36681	10000	9000	689	679
4	6	4	50000	50000	1	2	37	0	1815	657	1000	1000	800
5	7	5	500000	500000	1	2	29	0	40000	38000	20239	13750	13770
6	11	6	200000	200000	3	2	34	0	12	50	300	3738	66
7	15	7	250000	250000	1	2	29	0	3000	3000	3000	3000	3000
8	18	8	320000	320000	1	1	49	0	10000	75940	20000	195599	50000
9	21	9	130000	130000	3	2	39	0	1537	1000	2000	930	33764
10	26	10	50000	50000	3	2	23	0	1426	1001	1432	1062	997
11	28	11	50000	50000	3	2	30	0	1300	1000	1500	1000	1012
12	32	12	50000	50000	2	2	33	2	1500	1000	1000	1000	716
13	33	13	100000	100000	1	2	32	0	3511	3302	3204	3200	2504
14	34	14	500000	500000	2	1	54	-2	22827	7521	71439	981	51582
15	37	15	280000	280000	2	1	40	0	8060	6300	6400	6400	6737
16	41	16	360000	360000	1	2	33	0	7000	6000	188840	28000	4000
17	42	17	70000	70000	1	2	25	0	4500	4042	2500	2800	2500
18	43	18	10000	10000	2	2	22	0	2927	1000	300	1000	500
19	44	19	140000	140000	2	1	37	0	3000	3000	4000	4000	3000
20	49	20	380000	380000	2	2	32	-1	15138	24677	11851	11875	8251
21	50	21	20000	20000	1	2	24	0	1315	704	928	912	1069
22	52	22	100000	100000	3	3	43	0	1606	1500	2000	1500	1000
23	54	23	180000	180000	1	2	25	1	2010	1762	1762	1790	1622
24	55	24	150000	150000	1	2	29	2	1718	1749	1500	2000	5000

## Conclusión

Se ha llevado una base de datos postgres y un sftp , al finalizar se pudo notar la potencialidad de usar camel para el procesamiento de informacion y gracias a las librerias se simplifica la codificacion.

Para la prueba usamos un Docker que proporciona una forma eficiente, portátil y escalable de gestionar y desplegar estos servidores.

Proporciona aislamiento, consistencia, agilidad y facilidad de gestión, lo que resulta en un proceso de desarrollo y despliegue más fluido, una mayor eficiencia en el uso de recursos y una mejor experiencia para los usuarios finales.

## ADR GITHUB

<https://github.com/scordero1234/first-camel-integration.git>