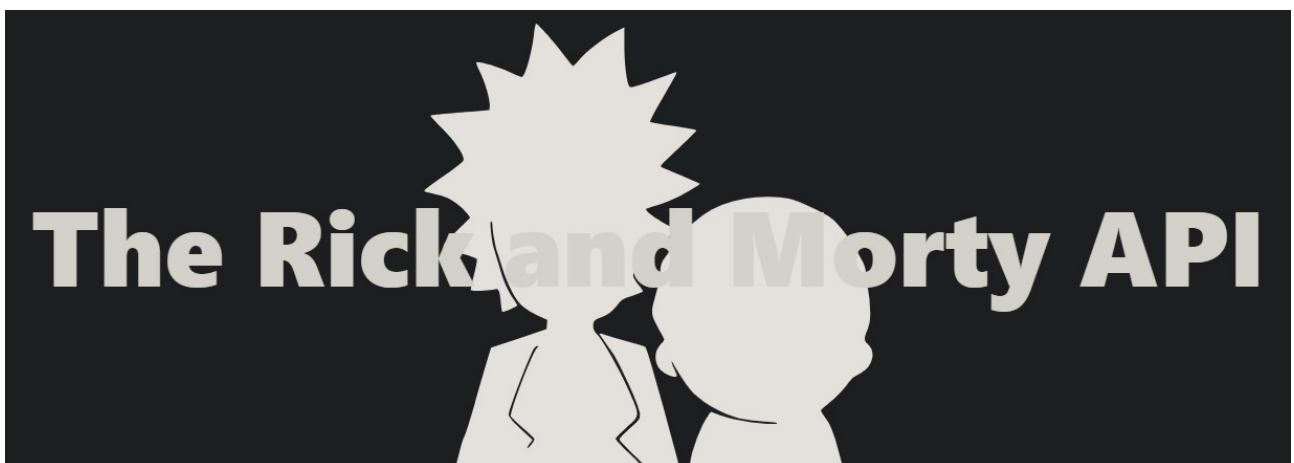


## **Construcción de un Flujo de Datos en Apache NiFi**



**Autor: Samuel Arteaga López**  
**Fecha de entrega: 26/01/2025**  
**Módulo: Big Data Aplicado**

## Índice

1.- Fuente de datos pública seleccionada.....	3
2.- Una explicación detallada de las diferentes transformaciones realizadas.....	3
3.- Descripción del destino donde se almacenan los datos procesados.....	8

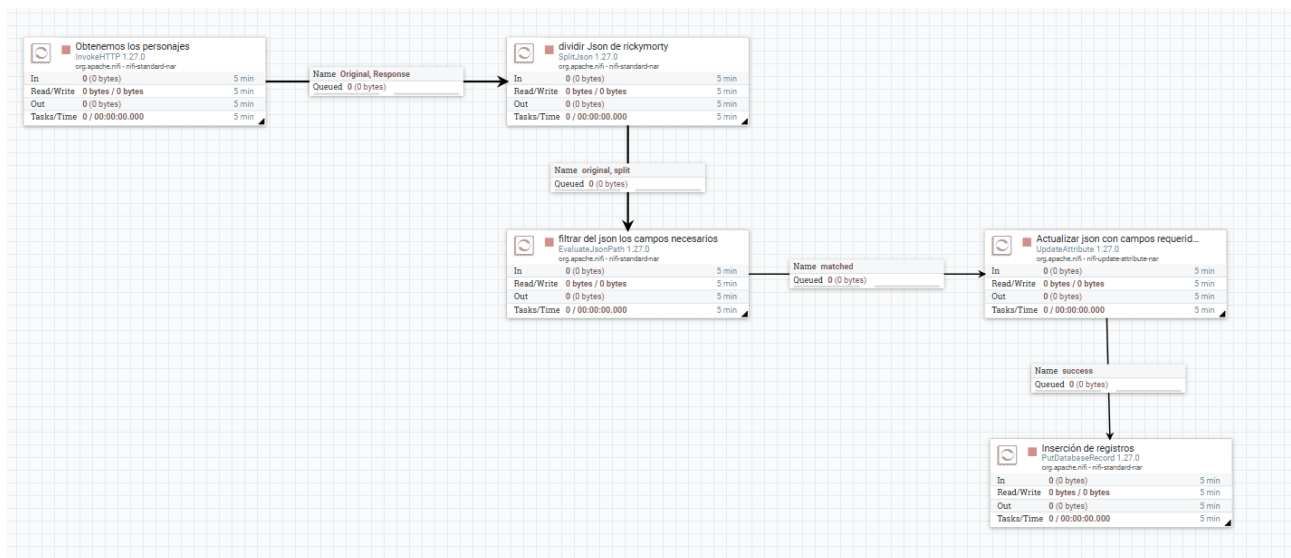
## 1.- Fuente de datos pública seleccionada

Para este proyecto de Apache Nifi se ha seleccionado la API de la serie de animación Rick y Morty, la cual cuenta con más de 800 personajes de los 50 capítulos aproximadamente que tiene la serie.

La elección de porqué se ha escogido la API de Rick and Morty se basa en su accesibilidad, estructura de datos bien definida y el potencial para realizar transformaciones interesantes, además de ser una fuente gratuita y fácil de utilizar, por no hablar del afán por la serie (la recomiendo, si te gusta las series sin sentido).

## 2.- Una explicación detallada de las diferentes transformaciones realizadas.

A continuación detallamos el proceso de transformaciones en este proyecto:



Primero de todo utilizamos InvokeHttp para invocar la llamada a la api de personajes de Rick y Morty

<http://rickandmortyapi.com/api/character>

El json que obtenemos tiene este aspecto

```
"info": {  
  "count": 826,  
  "pages": 42,  
  "next": "https://rickandmortyapi.com/api/character?page=2",  
  "prev": null  
},  
"results": [  
  {  
    "id": 1,  
    "name": "Rick Sanchez",  
    "status": "Alive",  
    "species": "Human",  
    "type": "",  
    "gender": "Male",
```

Nos quedaremos con “results” con el procesador splitJson

**Configure Processor** | SplitJson 1.27.0

Stopped

SETTINGS SCHEDULING **PROPERTIES** RELATIONSHIPS COMMENTS

Required field ⓘ +

Property	Value
JsonPath Expression	<span>?</span> \$.results[*] <span>i</span>
Null Value Representation	<span>?</span> empty string
Max String Length	<span>?</span> 20 MB

CANCEL APPLY

Seguidamente nos quedaremos con los valores que nos interesa para la tabla de la base de datos

Configure Processor | EvaluateJsonPath 1.27.0

Stopped

SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS

Required field

Property	Value
Destination	flowfile-attribute
Return Type	auto-detect
Path Not Found Behavior	ignore
Null Value Representation	empty string
Max String Length	20 MB
gender	\$.gender
name	\$.name
species	\$.species
status	\$.status
type	\$.type

CANCEL
APPLY

donde el dólar (\$) indica la cabecera del json, nos quedamos con todos los atributos-

Configure Processor | UpdateAttribute 1.27.0

Stopped

SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS

Required field

Property	Value
Delete Attributes Expression	No value set
Store State	Do not store state
Stateful Variables Initial Value	No value set
Cache Value Lookup Cache Size	100
filename	\${name}.json
gender	\${gender}
name	\${name}
species	\${species}
status	\${status}
type	\${type}

ADVANCED
CANCEL
APPLY

En updateAttribute lo que realizamos es obtener el valor de los atributos a ser insertados y darle el nombre del personaje al json.

Finalmente insertamos los valores de los atributos seleccionados en nuestra tabla de la base de datos

**Configure Processor** | PutDatabaseRecord 1.27.0

Stopped

SETTINGS SCHEDULING **PROPERTIES** RELATIONSHIPS COMMENTS

Required field

Property	Value
Record Reader	JsonTreeReader
Database Type	MySQL
Statement Type	INSERT
Data Record Path	No value set
Database Connection Pooling Service	DBCPCConnectionPool
Catalog Name	No value set
Schema Name	RickAndMorty
Table Name	personajes
Binary String Format	UTF-8
Translate Field Names	true
Unmatched Field Behavior	Ignore Unmatched Fields
Unmatched Column Behavior	Fail on Unmatched Columns

CANCEL APPLY

En el JsonTreeReader, se ha configurado el procesador para tomar de ejemplo una plantilla de json para evaluar lo que se debe de esperar como entrada, en este caso se ha configurado de la siguiente forma:

**Controller Service Details** | JsonTreeReader 1.27.0

ENABLED DISABLE & CONFIGURE

SETTINGS **PROPERTIES** COMMENTS

Required field

Property	Value
Schema Access Strategy	Use 'Schema Text' Property
Schema Text	{...}

Starting Field Strategy

Max String Length

Allow Comments

Date Format

Time Format

Timestamp Format

```

1 {
2   "type": "record",
3   "name": "Character",
4   "fields": [
5     {"name": "id", "type": "int"},
6     {"name": "name", "type": "string"},
7     {"name": "status", "type": ["null", "string"], "default": null},
8     {"name": "species", "type": ["null", "string"], "default": null},
9     {"name": "type", "type": ["null", "string"], "default": null},
10    {"name": "gender", "type": ["null", "string"], "default": null}
11  ]
12 }
  
```

OK

Con este añadimos una regla de validación frente a los múltiples jsons que recibimos después de ser manipulados.

**Controller Service Details**
DBCPConnectionPool 1.27.0

▶ **ENABLED**
 **DISABLE & CONFIGURE**

SETTINGS

PROPERTIES

COMMENTS

**Required field**

Property		Value
Database Connection URL		jdbc:mysql://mysql:3306/RickAndMorty
Database Driver Class Name		com.mysql.cj.jdbc.Driver
Database Driver Location(s)		No value set
Kerberos User Service		No value set
Kerberos Credentials Service		No value set
Kerberos Principal		No value set
Kerberos Password		No value set
Database User		root
Password		Sensitive value set
Max Wait Time		500 millis
Max Total Connections		8
Validation query		SELECT 1;

OK

### 3.- Descripción del destino donde se almacenan los datos procesados.

El destino de los datos es una base de datos mysql en un contenedor docker con la imagen de mysql 8.0 utilizando dbeaver creamos la tabla donde se insertan nuestros datos

	id_personaje	A-Z name	A-Z status	A-Z species	A-Z type	A-Z gender
1	1	Rick Sanchez	Alive	Human		Male
2	2	Morty Smith	Alive	Human		Male
3	3	Summer Smith	Alive	Human		Female
4	4	Beth Smith	Alive	Human		Female
5	5	Jerry Smith	Alive	Human		Male
6	6	Abadango Cluster Princess	Alive	Alien		Female
7	7	Abradolf Lincler	unknown	Human	Genetic experiment	Male
8	8	Adjudicator Rick	Dead	Human		Male
9	9	Agency Director	Dead	Human		Male
10	10	Alan Rails	Dead	Human	Superhuman (Ghost trains summoner)	Male
11	11	Albert Einstein	Dead	Human		Male
12	12	Alexander	Dead	Human		Male
13	13	Alien Googah	unknown	Alien		unknown
14	14	Alien Morty	unknown	Alien		Male
15	15	Alien Rick	unknown	Alien		Male
16	16	Amish Cyborg	Dead	Alien	Parasite	Male
17	17	Annie	Alive	Human		Female
18	18	Antenna Morty	Alive	Human	Human with antennae	Male
19	19	Antenna Rick	unknown	Human	Human with antennae	Male
20	20	Ants in my Eyes Johnson	unknown	Human	Human with ants in his eyes	Male
21	21	[NULL]	[NULL]	[NULL]	[NULL]	[NULL]
22	22	Rick Sanchez	Alive	Human		Male
23	23	Morty Smith	Alive	Human		Male
24	24	Summer Smith	Alive	Human		Female
25	25	Beth Smith	Alive	Human		Female

Propiedades						
Table Name: <input type="text" value="personajes"/> <input type="checkbox"/> Partitioned Engine: <input type="text" value="InnoDB"/> Auto Increment: <input type="text" value="0"/> Charset: <input type="text" value="utf8mb4"/> Collation: <input type="text" value="utf8mb4_0900_ai_ci"/> Description: <input type="text"/>						
	Column Name	#	Data Type	Not Null	Auto Increment	Key
<b>Columns</b>	id_personaje	1	int	[v]	[v]	PR
<b>Constraints</b>	A-Z name	2	varchar(255)	[ ]	[ ]	
	A-Z status	3	varchar(50)	[ ]	[ ]	
	A-Z species	4	varchar(100)	[ ]	[ ]	
	A-Z type	5	varchar(100)	[ ]	[ ]	
	A-Z gender	6	varchar(50)	[ ]	[ ]	
<b>Foreign Keys</b>						
<b>References</b>						
<b>Triggers</b>						
<b>Indexes</b>						
<b>Partitions</b>						
<b>Statistics</b>						
<b>DDL</b>						
<b>Virtual</b>						



