



UNIVERSIDAD
GERARDO BARRIOS

Líderes en Gestión del Conocimiento



CouchDb Manual
de instalación,
configuración y
uso.

Tabla de contenido

INTRODUCCION :	4
¿Qué es CouchDB?	4
Características principales:	5
Instalación de DBMS NOSQL.	6
Pasos para la Instalación de Apache CouchDB:	6
Requerimientos de Hardware	6
Descarga del archivo	6
Configuración en el asistente	6
Finalizando Instalación	9
Primeros pasos CouchDB.....	9
CRUD CouchDB	11
Creando, exportando e insertando vistas de CouchDB (NoSQL) a MySQL.....	14
Creando vistas.....	14
Exportando vistas:	16
Trabajando con PostMan	16
Convirtiendo JSON a CSV	18
Insertando vistas.....	20
Trabajando con PENTAH0	20
Conectando PENTAH0 con MySQL.....	21
Creando tabla MySQL desde PENTAH0	22
Verificando datos en MySQL.....	25
Replicación local en CouchDB	26
Referencias	33



Apache CouchDB

INTRODUCCION :

¿Qué es CouchDB?

CouchDB es un sistema de gestión de bases de datos el cual aplica lo mejor de las bases de datos documentales las cuales son NoSQL. Apache Software Foundation es la organización responsable del desarrollo de Couch, que desde 2005 comenzó a trabajar en el software libre Apache CouchDB y lo lleva mejorando desde entonces. (Sevilla, 2018)

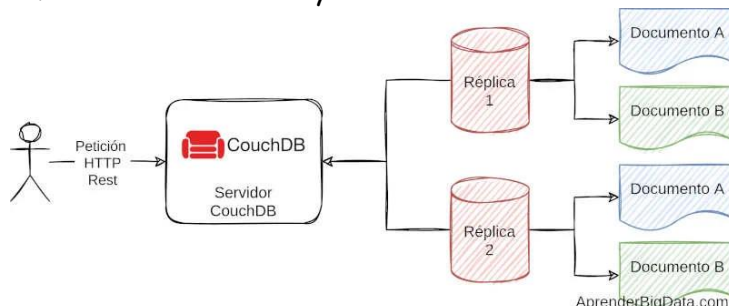
Los principales sistemas operativos con los que CouchDB es compatible son los siguientes:

- ✓ Linux,
- ✓ Unix,
- ✓ macOS y
- ✓ Windows (IONOS, 2020)

Base de datos NoSQL que usa JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos basado en un subconjunto del lenguaje de **JavaScript**. CouchDB no guarda sus datos en filas y columnas si no que los guarda como documentos, en pocas palabras este gestor permite almacenar la información de manera no estructurada, cada base de datos es una colección de documentos independientes, y cada documento contiene sus propios datos y su esquema.

CouchDB es un sistema gestor de bases de datos no relacional, basado completamente en tecnologías sobre las que está implementada la web, como HTTP, JSON o Javascript. Es una base de datos distribuida, capaz de adaptarse tanto a ambientes de servidor como a clientes de diversos tipos. Pensado especialmente para usarse como infraestructura en aplicaciones web. Su principal característica es la definición del Couch Replication Protocol, un sistema que permite que una variedad de productos puedan comunicarse entre sí, compartiendo datos y realizando una sincronización de estos de manera automática. (Zhunio, 2011)

Muchas tecnologías utilizan el protocolo de replicación de CouchDB desde servidores en clúster, teléfonos o navegadores de todo tipo. La ventaja es que los sistemas remotos pueden gestionar los datos y tenerlos disponibles incluso si está offline, CouchDB se encarga de gestionar las inconsistencias haciendo la sincronización de la información cuando haya conexión a internet.



Características principales:

- * Almacena los datos como documentos.
- * Provee una semántica de atomicidad, consistencia, aislamiento y durabilidad (ACID).
- * Los datos almacenados se estructuran por medio de vistas. Cada vista es creada por medio de una función de JavaScript, puede indexar vistas y mantener actualizados estos índices a medida que se agregan, actualizan o eliminan estos documentos.
- * 6 señor teniendo en mente la replicación bidireccional (o sincronización) y la operación offline.
- * Todos los ítems tienen una URI única que queda expuesta vía HTTP. REST usa los métodos HTTP POST, GET, PUT y DELETE para las cuatro operaciones básicas CRUD (Create, Read, Update, Delete) con todos los recursos.
- * CouchDB garantiza consistencia eventual para poder ofrecer tanto disponibilidad como tolerancia a las particiones.
- * puede replicar datos a dispositivos (como teléfonos inteligentes) que pueden quedar offline y manejar automáticamente la sincronización de los datos cuando el dispositivo vuelve a estar en línea.

Las replications en la sincronización de CouchDB hacen que sea un recurso ideal para dispositivos móviles donde la conectividad de red no está asegurada pero la aplicación debe seguir operando fuera de línea. CouchDB se usa en determinadas aplicaciones para Android, tales como "SpreadLyrics", así como aplicaciones para Facebook como "Will you Kissme" o "Birthday Greeting Cards", o bien sitios web como "Friendpaste".

Instalación de DBMS NOSQL.

Pasos para la Instalación de Apache CouchDB:

Requerimientos de Hardware

Windows

1. Microsoft® Windows® 7/8/10 (32-bit o 64-bit), Microsoft® Windows® Server® 2008 R2, 2012 R2, 2016, 2019 (32-bit o 64-bit). Fuente: página oficial de Apache
2. 6 GB RAM mínimo - 8 GB RAM recomendado
3. 10 GB de espacio disponible - 15 GB recomendado.
4. 1280 x 800 resolución mínima de pantalla

Descarga del archivo

1. Primero descargamos el instalador desde la página oficial de apache CouchDB

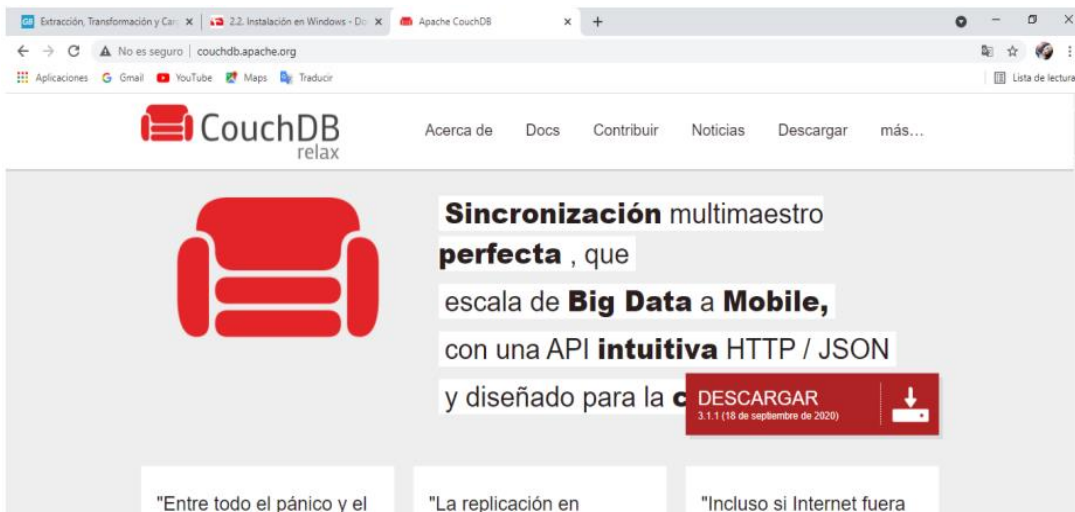


Imagen: Fuente página Apache CouchDB

Configuración en el asistente

2. Ahora procederemos a abrir el instalador. Siga los pasos del asistente de instalación; A continuación, en la pantalla "Bienvenida"

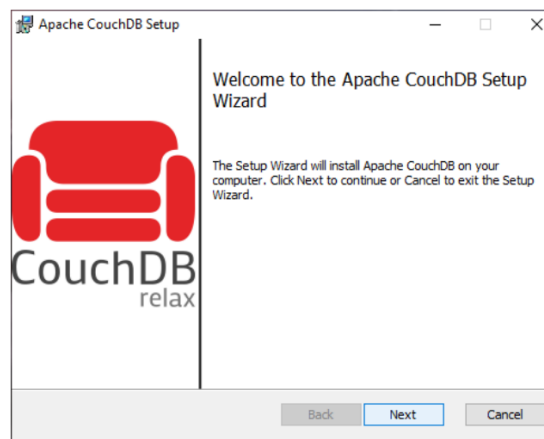


Imagen: Fuente propia (Instalación CouchDB)

3. Acepta el acuerdo de licencia



Imagen: Fuente propia (Instalación CouchDB)

4. Seleccione el directorio de instalación.

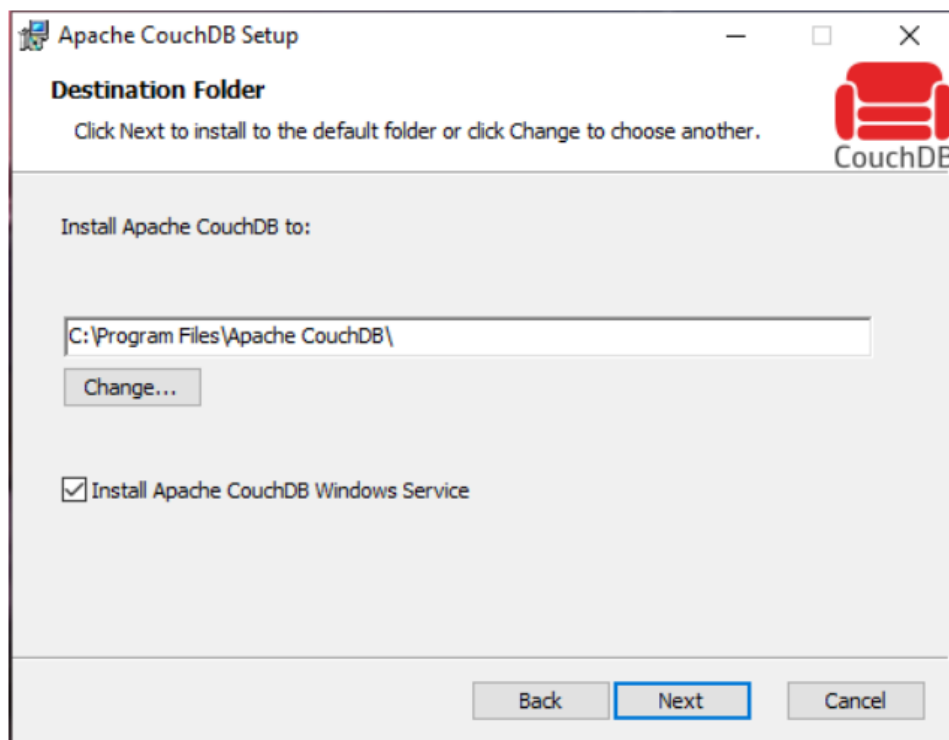
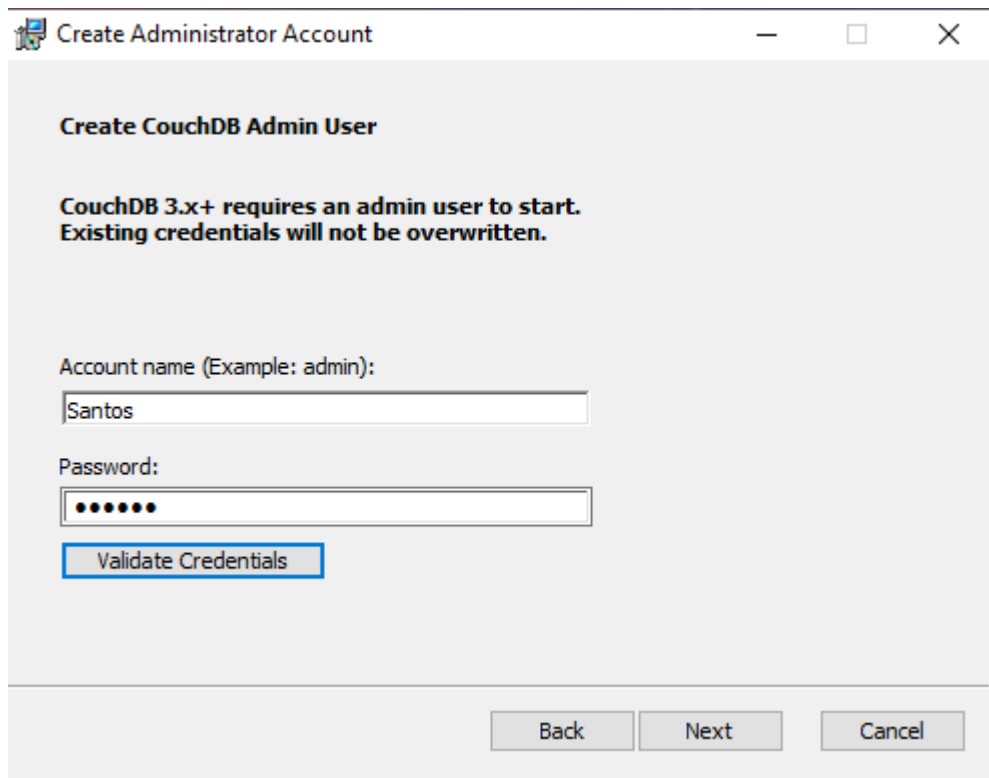


Imagen: Fuente propia (Instalación CouchDB)

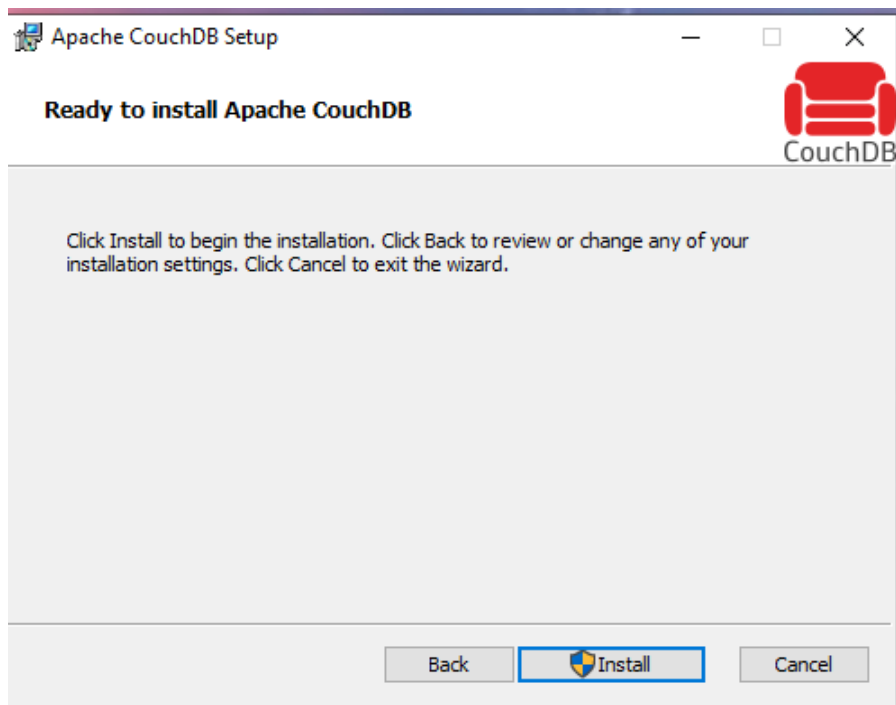
5. Especifique el nombre del usuario y contraseña



The screenshot shows a window titled "Create Administrator Account". Inside, it says "Create CouchDB Admin User" and "CouchDB 3.x+ requires an admin user to start. Existing credentials will not be overwritten." There are two input fields: "Account name (Example: admin):" with the text "Santos" entered, and "Password:" with masked characters. Below the password field is a "Validate Credentials" button. At the bottom of the window are "Back", "Next", and "Cancel" buttons.

Imagen: Fuente propia (Instalación CouchDB)

6. Ahora click en install



The screenshot shows a window titled "Apache CouchDB Setup". It says "Ready to install Apache CouchDB" and includes the CouchDB logo. Below, it instructs: "Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard." At the bottom are "Back", "Install", and "Cancel" buttons.

Imagen: Fuente propia (Instalación CouchDB)

Finalizando Instalación

7. Y hemos finalizado la instalación

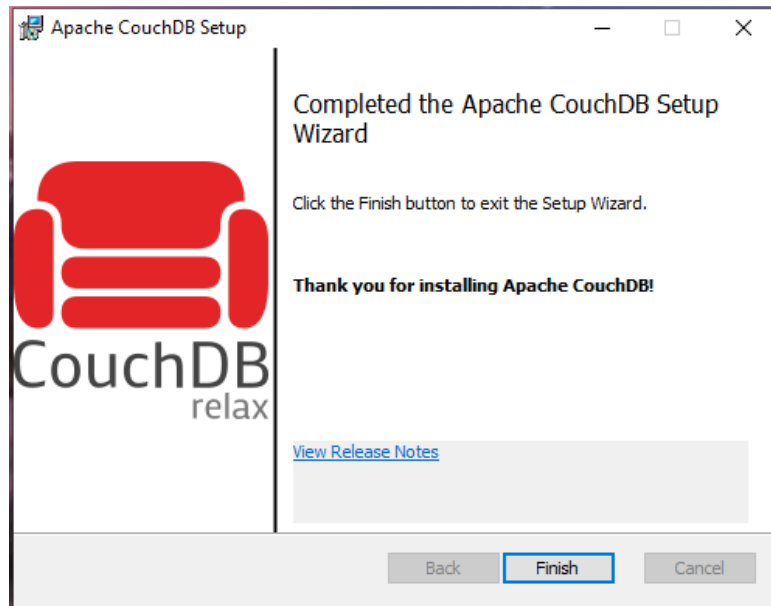
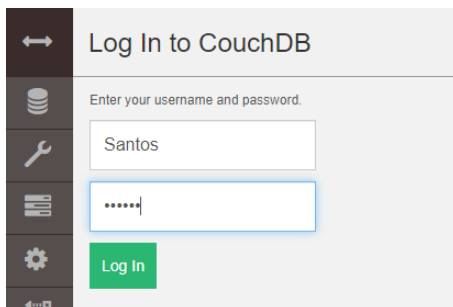


Imagen: Fuente propia (Instalación CouchDB)

Primeros pasos CouchDB

- Abrir couchDb, este se abre automáticamente al momento de finalizar la instalación de no ser así se puede hacer manualmente.
- Ingrese con el usuario y contraseña que creo al momento de instalar.



- Una vez ingresado el usuario verá esta la pantalla principal

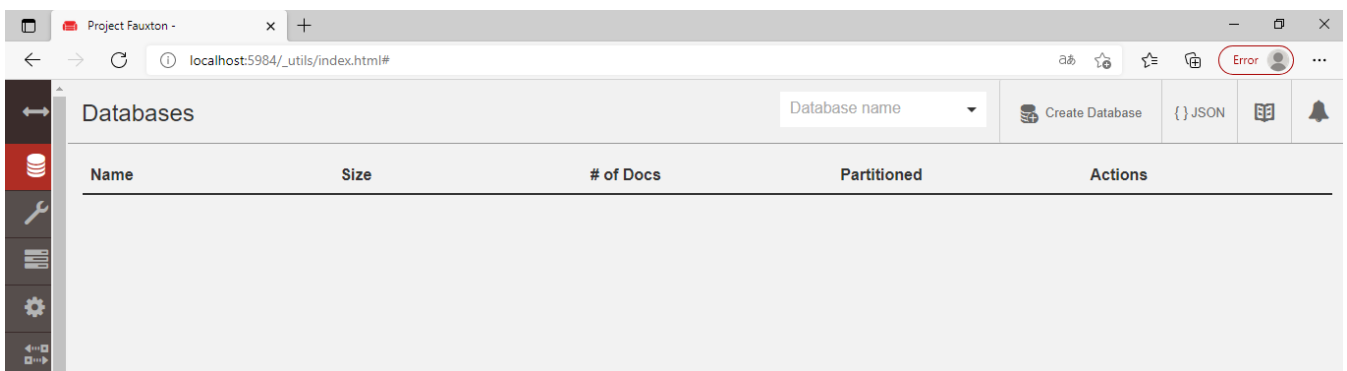


Imagen: Fuente propia (primeros pasos CouchDB)

- Ahora proceda a verificar la instalación.

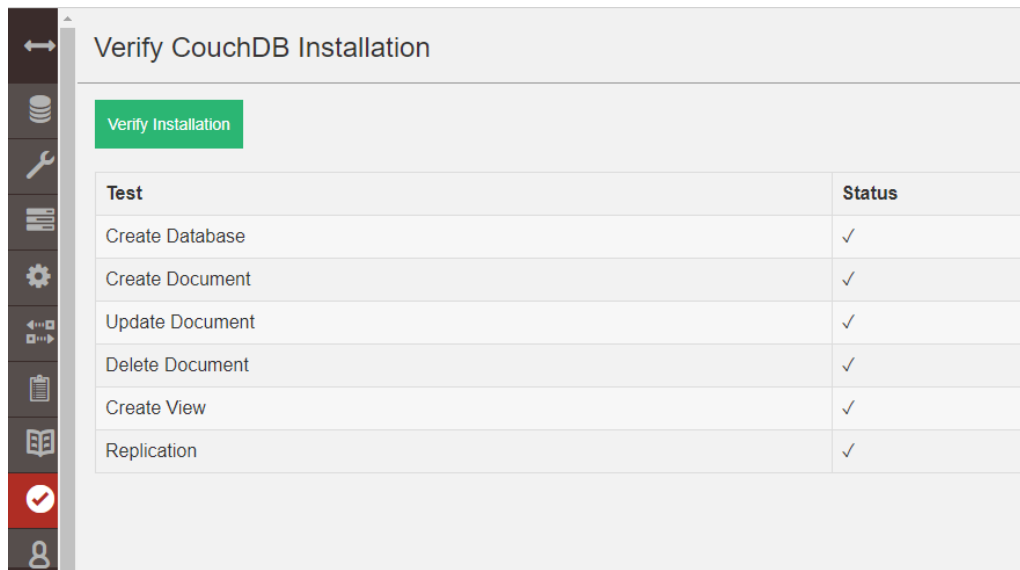


Imagen: Fuente propia (verificación de la instalación CouchDB)

- Ahora crearemos una base de datos solo con fines de prueba

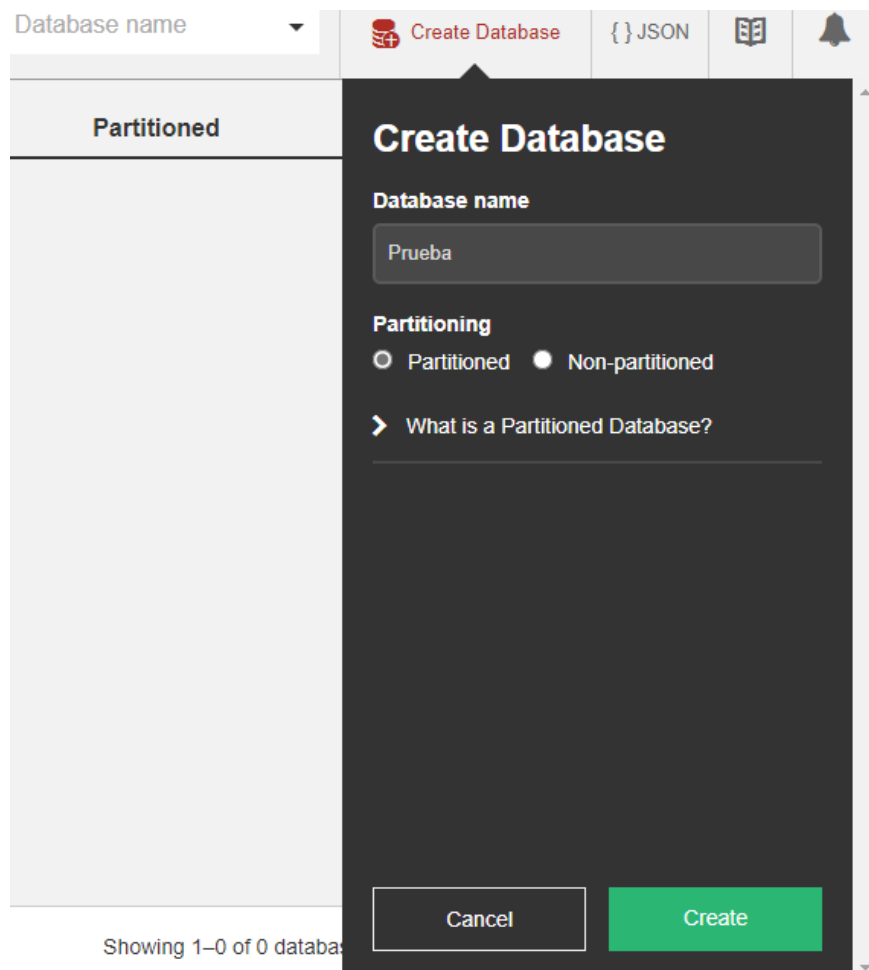


Imagen: Fuente propia (prueba CouchDB)

- Y fue creada con éxito

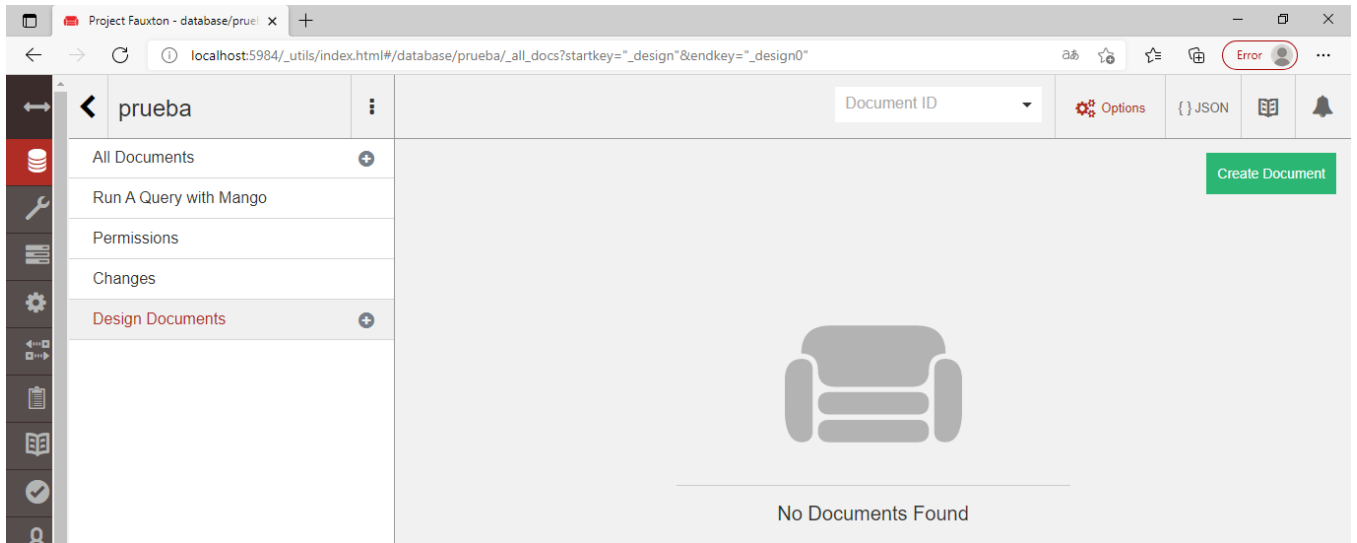


Imagen: Fuente propia (prueba CouchDB)

CRUD CouchDB

En la cmd utilizando cURL para el servidor de CouchDB, con el siguiente comando donde el operador "PUT" permite la creación de nuevos elementos también debemos tener en cuenta que en la línea de código es necesario agregar las credenciales, vamos a crear una base de datos con el nombre de crud

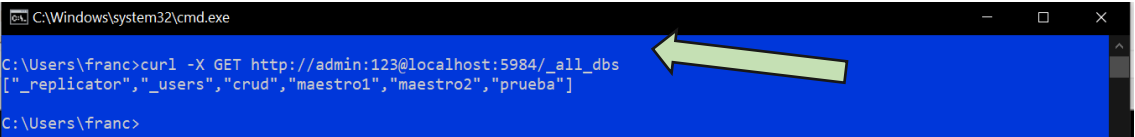


Imagen: Fuente propia (cmdCouchDB)

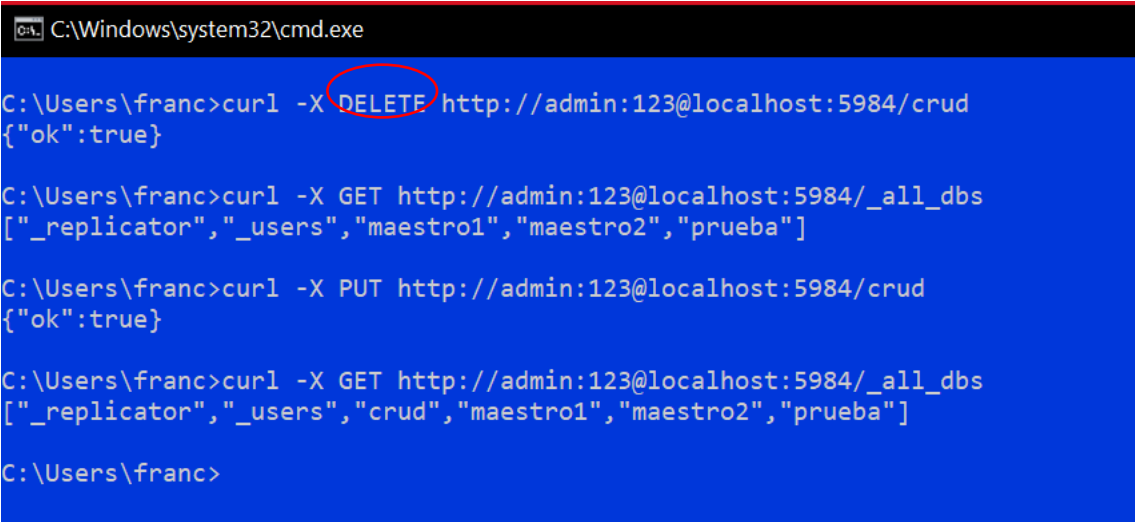
Ahora vamos a revisar que efectivamente se creó la base de datos:

Databases				Database name	Create Database	{ } JSON	
Name	Size	# of Docs	Partitioned	Actions			
<u>_replicator</u>	3.2 KB	3	No				
<u>_users</u>	3.1 KB	2	No				
crud	0 bytes	0	No				
maestro1	1.8 KB	2	No				
maestro2	1.8 KB	2	No				
prueba	38.8 KB	6	No				

Con el operador "GET" vamos a poder visualizar desde la consola cuales son las bases de datos con la ayuda de "all_dbs", mostrando como resultado el nombre de las bases existentes:



Para poder borrar una base de datos utilizaremos la sentencia "DELETE", luego consultamos las bases de datos existentes y vemos que ya se ha borrado, la volvemos a crear para continuar



Para crear un documento nos vamos a la interfaz del navegador que se llama fauxton, vamos a crear el documento en formato json para agregarlo a la coleccion solo digitamos los campos con la informacion que queremos:

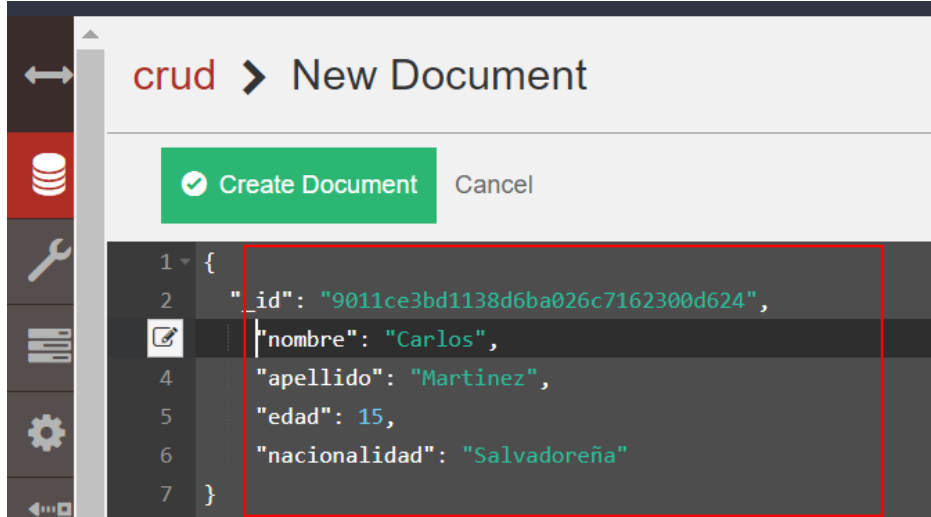


Imagen: Fuente propia (Interfas CouchDB)

Para obtener datos de un documento podemos auxiliarnos del operador "GET" acompañado de la base de datos que contiene dicho documento junto con el "id" de este último, y obtenemos el resultado como se muestra en la imagen:

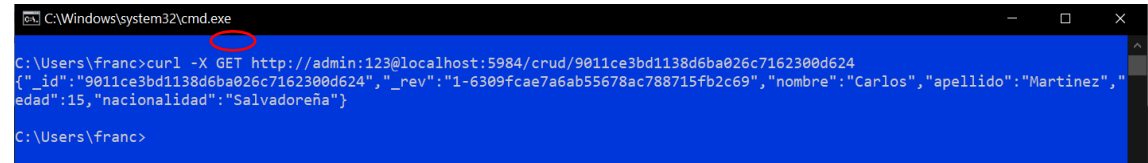


Imagen: Fuente propia (Operador GET CouchDB)

También podemos ir a la interfaz gráfica del navegador y modificar algún campo que quisiéramos de un documento modificando el código de este como se muestra a continuación:

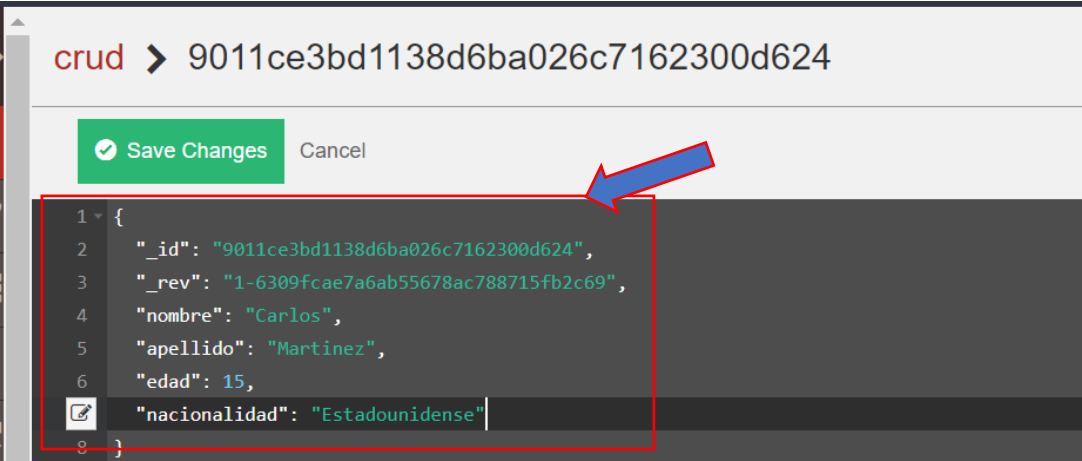
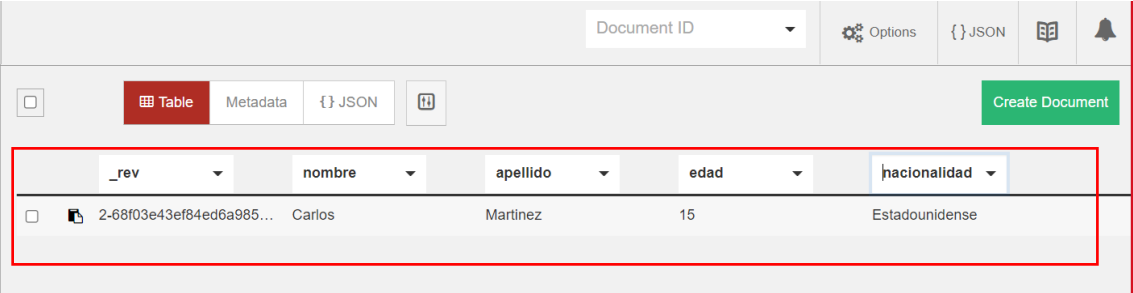


Imagen: Fuente propia (Interfaz CouchDB)

Vemos que el documento se ha modificado:



Para eliminar un documento solo debemos utilizar el comando "DELETE" especificar el nombre de la base de datos seguido del id y el número de revisión:

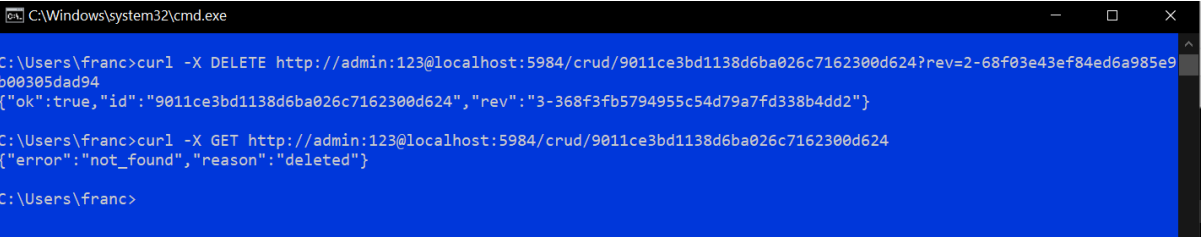
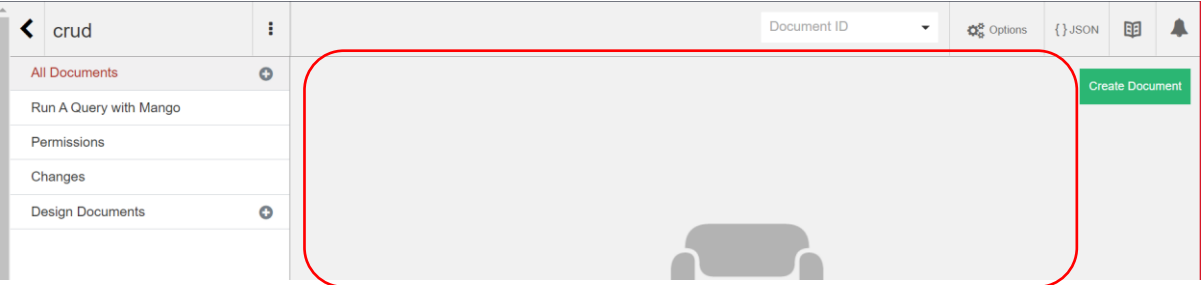


Imagen: Fuente propia (Operador Delete CouchDB)

Vamos a la interfaz gráfica y vemos que no tenemos ningún documento en nuestra base de datos porque como se lo indicamos se ha borrado efectivamente:



Creando, exportando e insertando vistas de CouchDB (NoSQL) a MySQL

Creando vistas

Primero, creamos una base de datos la cual llamamos **“prueba”**, dentro de ella vamos a crear cinco documentos, clienteid,nombre, apellido, ciudad, serán los datos a registrar.

TableMetadataJSON

Create Document







	_id	clienteid	nombre	apellido	ciudad
<input type="checkbox"/>	 0595b2646dd15d2b35be497e7a0000b9	1	Junior	Franco	San Miguel
<input type="checkbox"/>	 5e3d0aed77fb1d1ddd42eacbf6000e86	2	Jose	Julian	Usulután
<input type="checkbox"/>	 5e3d0aed77fb1d1ddd42eacbf600258e	3	Maria	carmelina	Morazan
<input type="checkbox"/>	 5e3d0aed77fb1d1ddd42eacbf6003baf	4	Lolita	Sanchez	San Miguel
<input type="checkbox"/>	 74fd0290492cd0cd99d520826f003202	5	Carlos	Romero	San Jacinto
<input type="checkbox"/>	 _design/desing1				

Imagen: Fuente propia (Interfaz CouchDB)

Vamos a crear una vista, para esto damos clic derecho en el apartado **“Desing Documents”** de las tres opciones que se despliegan, vamos a seleccionar **“New View”**, el objetivo de este es apartado es obtener una vista que personalice los valores de cada documento y obtenerlos en formato **json**.

<div>prueba</div>	<div>All Documents</div>	<div>Run A Query with Mango</div>	<div>Permissions</div>	<div>Changes</div>	<div>Design Documents</div>
	<div>Add New</div>	<div>New Doc</div>	<div>New View</div>	<div>Mango Indexes</div>	
	<div>Table</div>	<div>Metadata</div>	<div>{ } JSC</div>		
	<div>id</div>				
	<div>0595b2646dd15d2b35be497e7a0000b9</div>				
	<div>5e3d0aed77fb1d1ddd42eacbf6000e86</div>				
	<div>5e3d0aed77fb1d1ddd42eacbf600258e</div>				
	<div>5e3d0aed77fb1d1ddd42eacbf6003baf</div>				
	<div>74fd0290492cd0cd99d520826f003202</div>				

Imagen: Fuente propia (Interfaz CouchDB)

Designamos el nombre del documento, porque en CouchDB las vistas tambien son consideradas documentos, si en Lenguaje SQL hacemos un SELECT, para CouchDB necesitamos programar isinstrucciones en javascript.

Por lo tanto, la funcion doc, nos va permitir mediante la palabra reservada **emit**, especificar la clave y el valor de la consulta, en nuestro caso vamos a dejar el clienteid, especificado como key, y los valores a obtener los especificamos dentro de llaves. Asignando un campo nombre para cada valor por ejemplo para el valor **doc.clienteid**. le asignamos como nombre de campo **"Clienteid"**

New View

Design Document ?

New document

_design/ desing1

Index name ?

desing1

Map function ?

```
1 function (doc) {  
2   emit(doc.clienteid,{Clienteid:doc.clienteid, Nombre: doc.nombre,Apellido: doc.apellido, Ciudad: doc.ciudad})  
3 }
```

Reduce (optional) ?

NONE

✔ Create Document and then Build Index

Cancel

Imagen: Fuente propia (Interfaz CouchDB)

Guardamos la vista:

Document ID

Options

{ } JSON

Table

Metadata

{ } JSON

View Saved.






id	key	value
 0595b2646dd15d2b35be497e7a0000b9	1	{ "Clienteid": "1", "Nombre": "Junior", "Apellido": "Franco", "Ciudad": "San Miguel" }
 5e3d0aed77fb1d1ddd42eacbf6000e86	2	{ "Clienteid": "2", "Nombre": "Jose", "Apellido": "Julian", "Ciudad": "Usulután" }
 5e3d0aed77fb1d1ddd42eacbf600258e	3	{ "Clienteid": "3", "Nombre": "Maria", "Apellido": "carmelina", "Ciudad": "Morazan" }
 5e3d0aed77fb1d1ddd42eacbf6003baf	4	{ "Clienteid": "4", "Nombre": "Lolita", "Apellido": "Sanchez", "Ciudad": "San Miguel" }
 74fd0290492cd0cd99d520826f003202	5	{ "Clienteid": "5", "Nombre": "Carlos", "Apellido": "Romero", "Ciudad": "San Jacinto" }

Imagen: Fuente propia (Interfaz CouchDB)

Exportando vistas:

Trabajando con PostMan

Fauxton, nos genera una URL para poder consultar las vistas que nosotros creamos, en este siguiente paso nos dirigimos a la version web de "POSTMAN" pegamos la URL de la consulta, no sin antes especificarlo las credenciales de acceso en el apartado "Autorization" damos clic en "Send" y observamos que nos obtiene los datos de nuestra consulta, ahora damos clic al lado derecho en el que dice "Save Response".

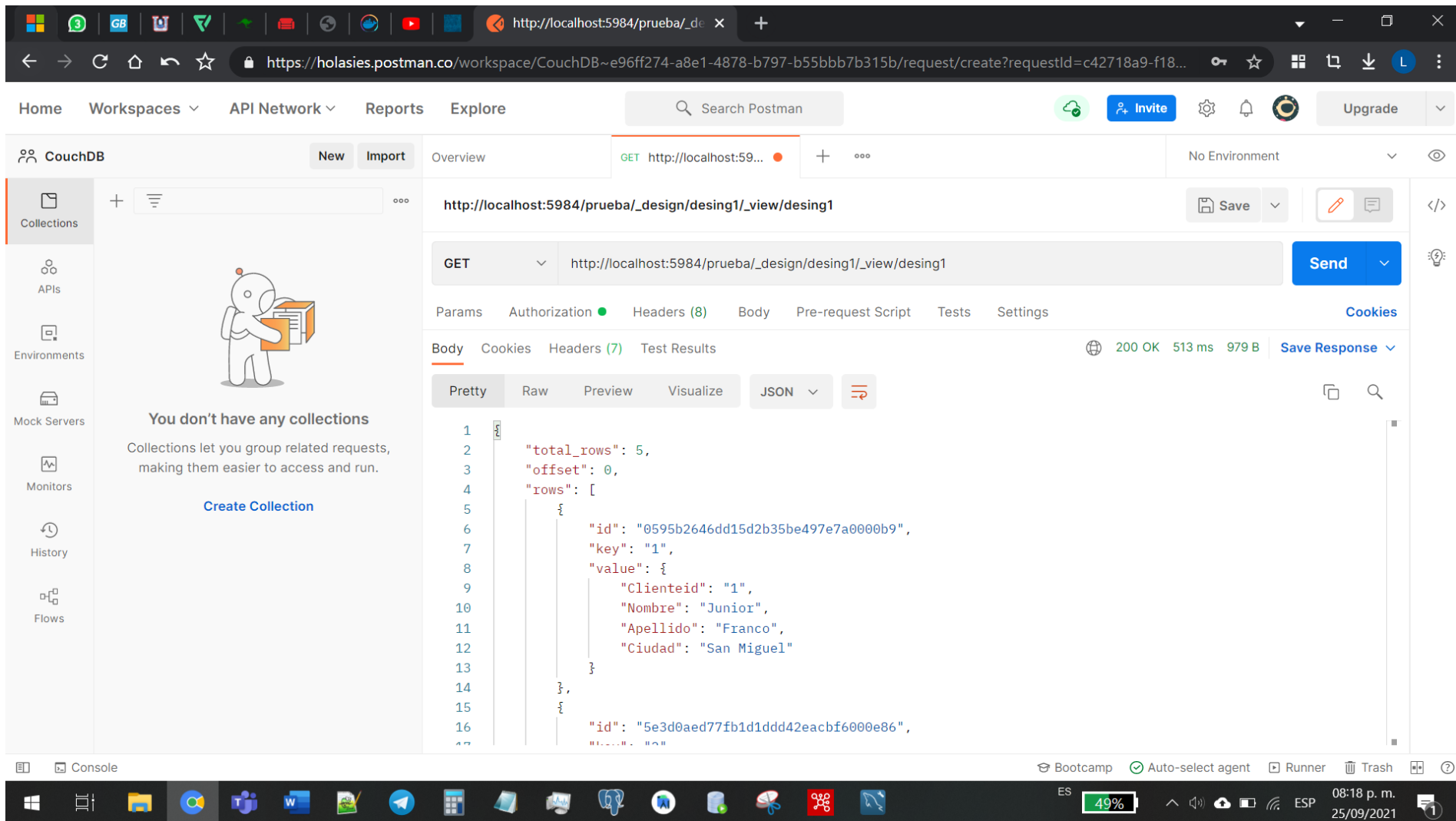


Imagen: Fuente propia Trabajando con POSTMAN

Lo vamos a guardar con el nombre **“vistadecouch.json”**

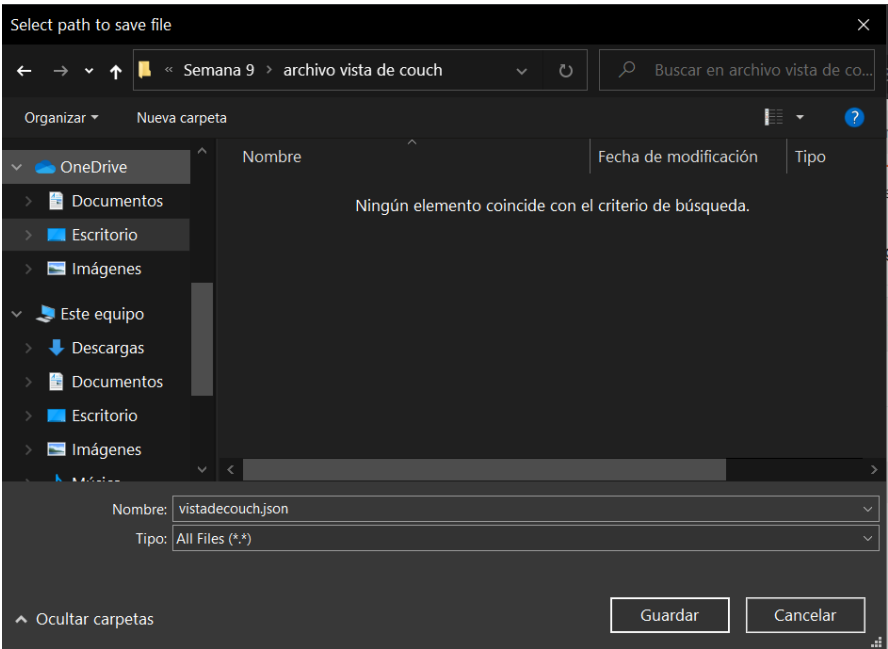


Imagen: Fuente propia Trabajando con POSTMAN

Observamos la estructura del archivo desde **Notepad++**

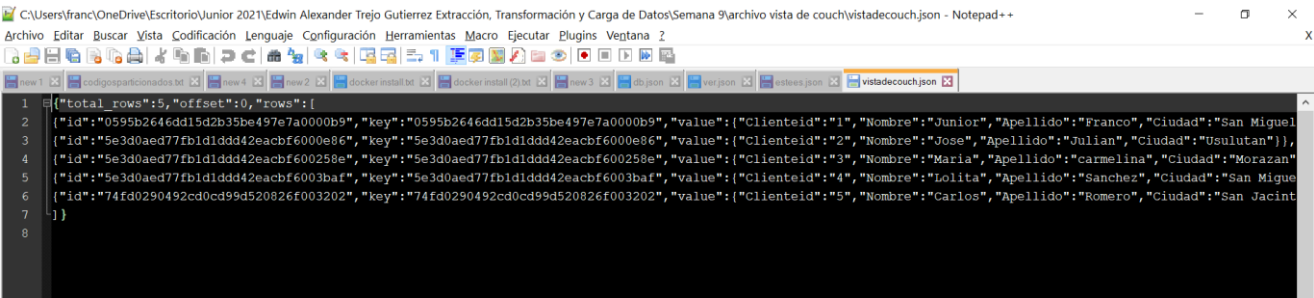
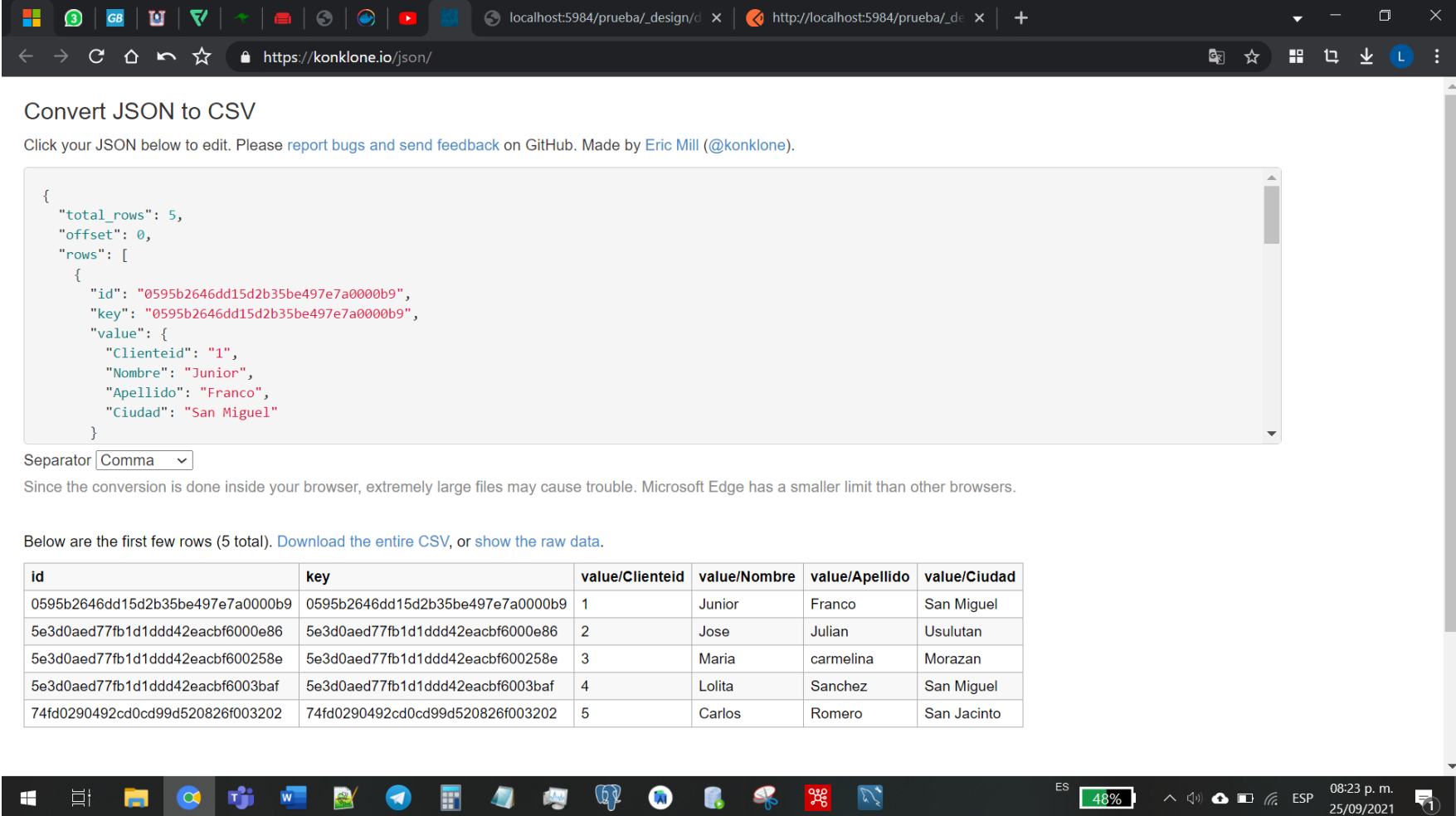


Imagen: Fuente propia Trabajando con POSTMAN

Convirtiendo JSON a CSV

Con la ayuda de la siguiente herramienta en línea vamos a convertir nuestro JSON a CSV, observamos como nos genera las columnas y ordena los datos, automaticamente nos detecta los valores de nuestro archivo y nos permite poder descargar la información.



Convert JSON to CSV

Click your JSON below to edit. Please [report bugs and send feedback](#) on GitHub. Made by [Eric Mill \(@konklone\)](#).

```
{
  "total_rows": 5,
  "offset": 0,
  "rows": [
    {
      "id": "0595b2646dd15d2b35be497e7a0000b9",
      "key": "0595b2646dd15d2b35be497e7a0000b9",
      "value": {
        "Clienteid": "1",
        "Nombre": "Junior",
        "Apellido": "Franco",
        "Ciudad": "San Miguel"
      }
    }
  ]
}
```

Separator

Since the conversion is done inside your browser, extremely large files may cause trouble. Microsoft Edge has a smaller limit than other browsers.

Below are the first few rows (5 total). [Download the entire CSV](#), or [show the raw data](#).


id	key	value/Clienteid	value/Nombre	value/Apellido	value/Ciudad
0595b2646dd15d2b35be497e7a0000b9	0595b2646dd15d2b35be497e7a0000b9	1	Junior	Franco	San Miguel
5e3d0aed77fb1d1ddd42eacbf6000e86	5e3d0aed77fb1d1ddd42eacbf6000e86	2	Jose	Julian	Usulután
5e3d0aed77fb1d1ddd42eacbf600258e	5e3d0aed77fb1d1ddd42eacbf600258e	3	Maria	carmelina	Morazan
5e3d0aed77fb1d1ddd42eacbf6003baf	5e3d0aed77fb1d1ddd42eacbf6003baf	4	Lolita	Sanchez	San Miguel
74fd0290492cd0cd99d520826f003202	74fd0290492cd0cd99d520826f003202	5	Carlos	Romero	San Jacinto

Imagen: Fuente propia convirtiendo JSON a CSV



Imagen: Fuente propia convirtiendo JSON a CSV

Visualizamos la información descargada desde el bloc de notas.

 result.csv: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
id,key,value/Clienteid,value/Nombre,value/Apellido,value/Ciudad
0595b2646dd15d2b35be497e7a0000b9,0595b2646dd15d2b35be497e7a0000b9,1,Junior,Franco,San Miguel
5e3d0aed77fb1d1ddd42eacbf6000e86,5e3d0aed77fb1d1ddd42eacbf6000e86,2,Jose,Julian,Usulután
5e3d0aed77fb1d1ddd42eacbf600258e,5e3d0aed77fb1d1ddd42eacbf600258e,3,Maria,carmelina,Benavidez
5e3d0aed77fb1d1ddd42eacbf6003baf,5e3d0aed77fb1d1ddd42eacbf6003baf,4,Lolita,Sanchez,San Miguel
74fd0290492cd0cd99d520826f003202,74fd0290492cd0cd99d520826f003202,5,Carlos,Romero,San Jacinto
```

Imagen: Fuente propia Descarga de archivo

Insertando vistas

Trabajando con PENTAHO

Desde la herramienta **PENTAHO**, seleccionamos un "CSV File Input" y buscamos la información descargada.

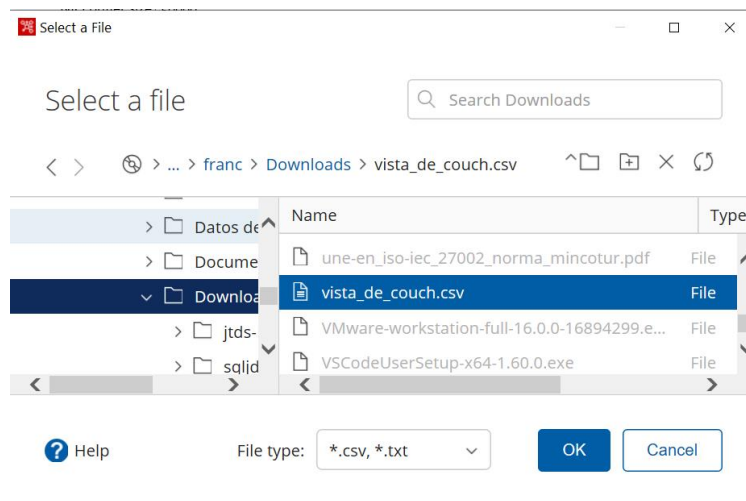


Imagen: Fuente propia Trabajando con PENTAHO

Las configuraciones son las siguientes: delimitado por comas, hacemos clic en **"Get Fields"** y automáticamente nos reconoce los campos de la tabla, el tipo de datos y por consecuencia la longitud de los mismos.

CSV file input

Step name: CSV file input

Filename: C:\Users\franc\Downloads\vista_de_couch.csv Browse...

Delimiter: , Insert TAB

Enclosure: "

NIO buffer size: 50000

Lazy conversion? ☒

Header row present? ☒

Add filename to result ☐

The row number field name (optional):

Running in parallel? ☐

New line possible in fields? ☐

Format: mixed


File encoding:

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim type
1	id	String		32		\$.	,	none
2	key	String		32		\$.	,	none
3	value/Clienteid	Integer	#	15	0	\$.	,	none
4	value/Nombre	String		6		\$.	,	none
5	value/Apellido	String		9		\$.	,	none
6	value/Ciudad	String		11		\$.	,	none

Help OK Get Fields Preview Cancel

Imagen: Fuente propia Trabajando con PENTAHO

Realizamos un preview de la información y verificamos el orden de los valores.

 Examine preview data

Rows of step: CSV file input (5 rows)


#	id	key	value/Clienteid	value/Nombre	value/Apellido	value/Ciudad
1	0595b2646dd15d2b35be497e7a0000b9	0595b2646dd15d2b35be497e7a0000b9	1	Junior	Franco	San Miguel
2	5e3d0aed77fb1d1ddd42eacbf6000e86	5e3d0aed77fb1d1ddd42eacbf6000e86	2	Jose	Julian	Usulután
3	5e3d0aed77fb1d1ddd42eacbf600258e	5e3d0aed77fb1d1ddd42eacbf600258e	3	Maria	carmelina	Morazan
4	5e3d0aed77fb1d1ddd42eacbf6003baf	5e3d0aed77fb1d1ddd42eacbf6003baf	4	Lolita	Sanchez	San Miguel
5	74fd0290492cd0cd99d520826f003202	74fd0290492cd0cd99d520826f003202	5	Carlos	Romero	San Jacinto

Imagen: Fuente propia Trabajando con PENTAHO

Conectando PENTAHO con MySQL

Seguidamente...

En un archivo SQL File Output, realizamos una conexión con MySQL, y seleccionamos una tabla en Este ejemplo vamos a seleccionar la tabla importado.

 SQL file output

Step name

General Content

Connection

Connection

Target schema

Target table

Output File

Add create table statement ☒

Add truncate table statement ☐

Start new line for each ☒

Filename

Create Parent folder ☐

Do not create at start ☐

Extension

Include stepnr in filename? ☐

Include date in filename? ☐

Include time in filename? ☐

Append ☐

Split every ... rows

Add File to result filenames ☐

Imagen: Fuente propia Conectando PENTAHO con MySQL

Creando tabla MySQL desde PENTAHO

Hacemos clic en el botón **"SQL"** de la imagen anterior y ejecutamos la sentencia que nos generará automáticamente.

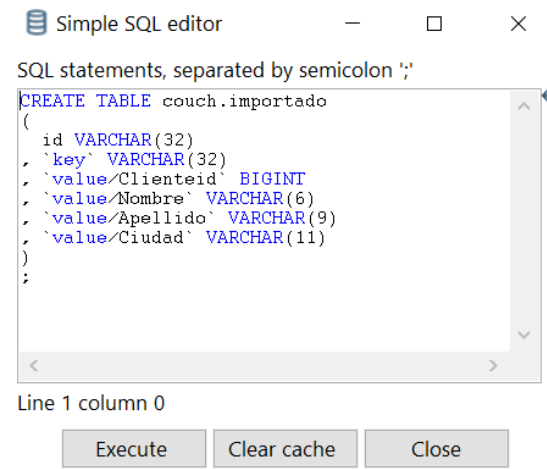


Imagen: Fuente propia creando tabla MySQL

Seguidamente...

Seleccionamos un **Table output** especificamos la conexión, obtenemos los valores del archivo SQL que se generó anteriormente y damos clic en "OK".

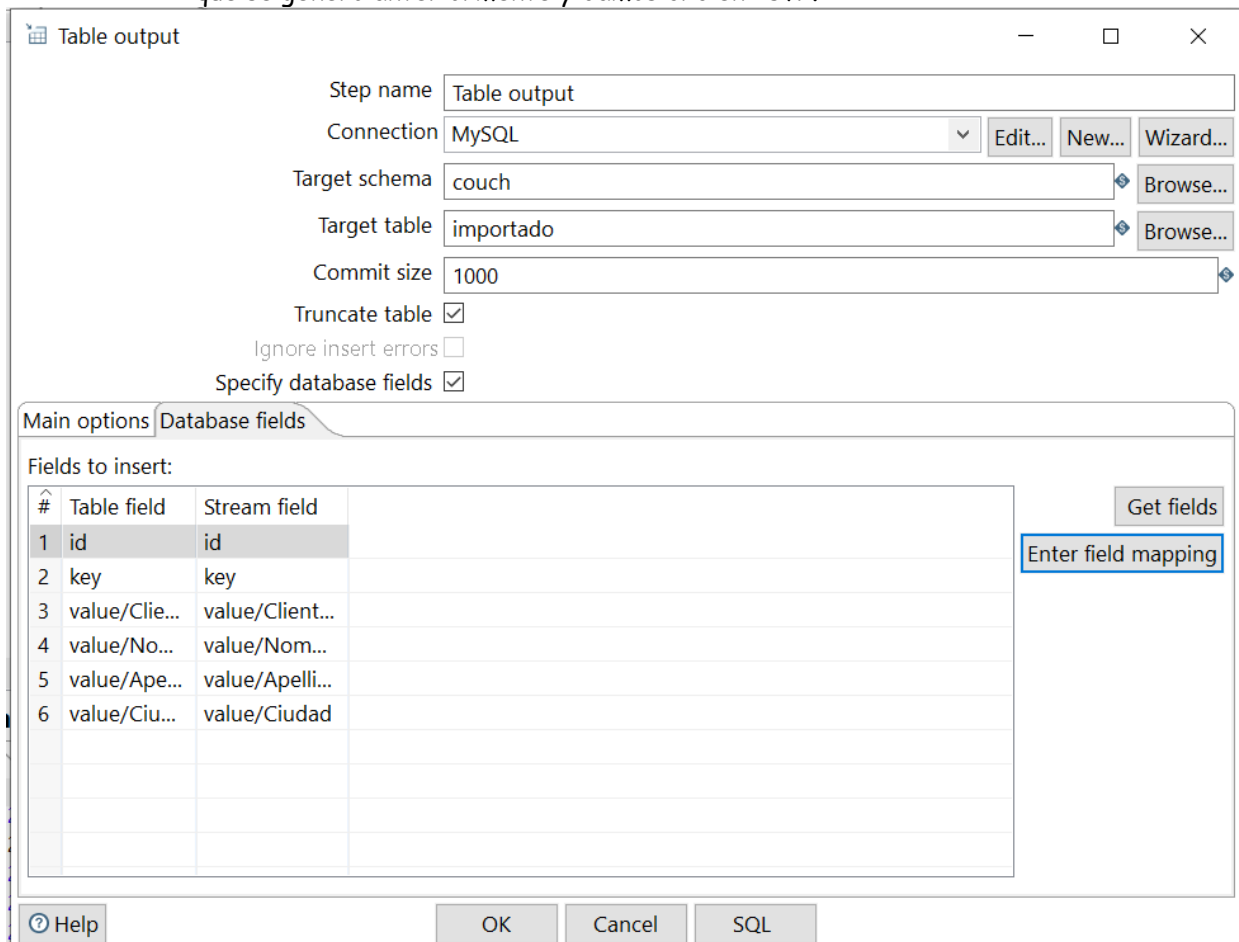


Imagen: Fuente propia creando tabla en MySQL

Ejecutamos las operaciones que hemos configurado anteriormente, y efectivamente no devuelven ningún error.

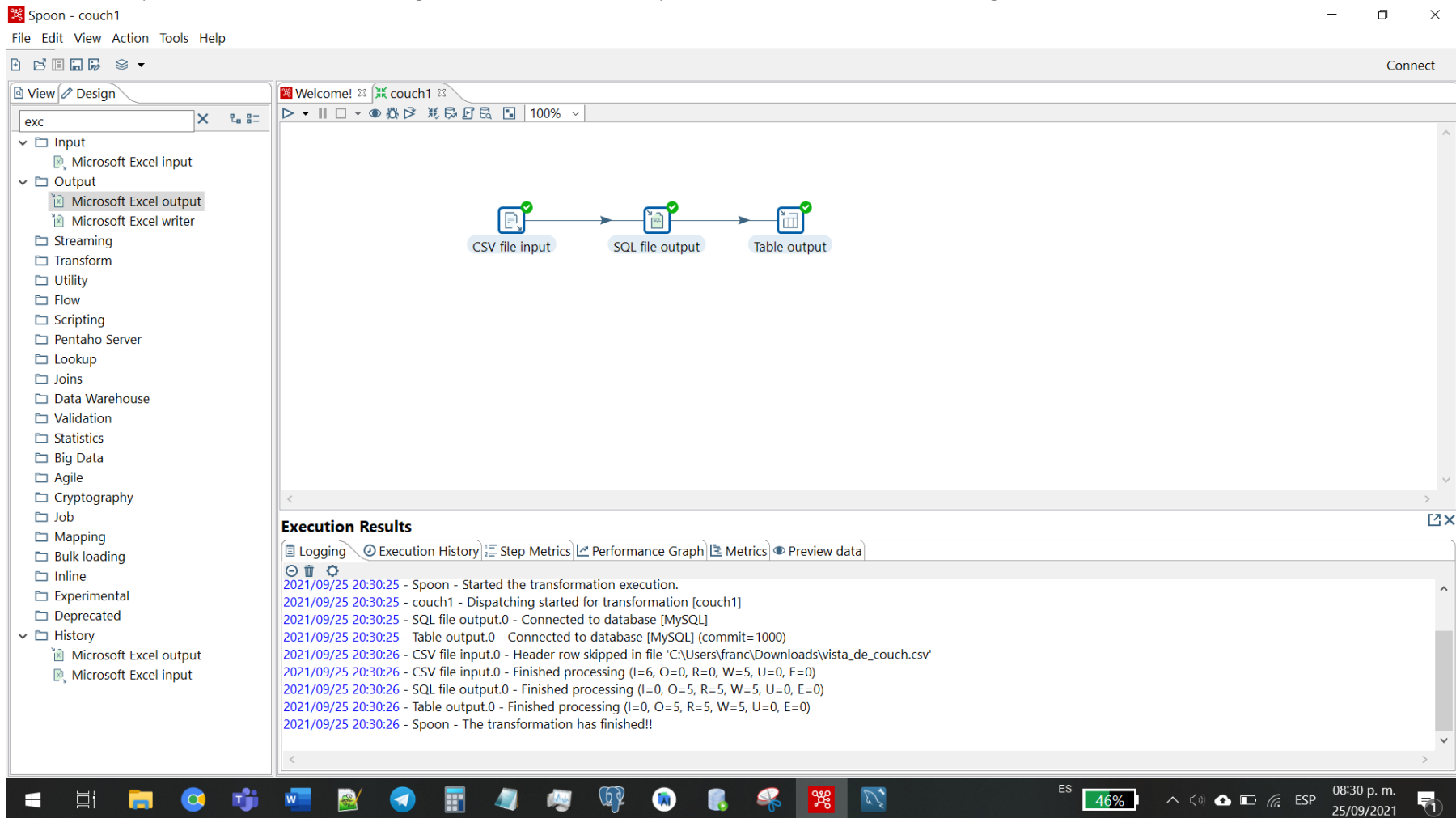
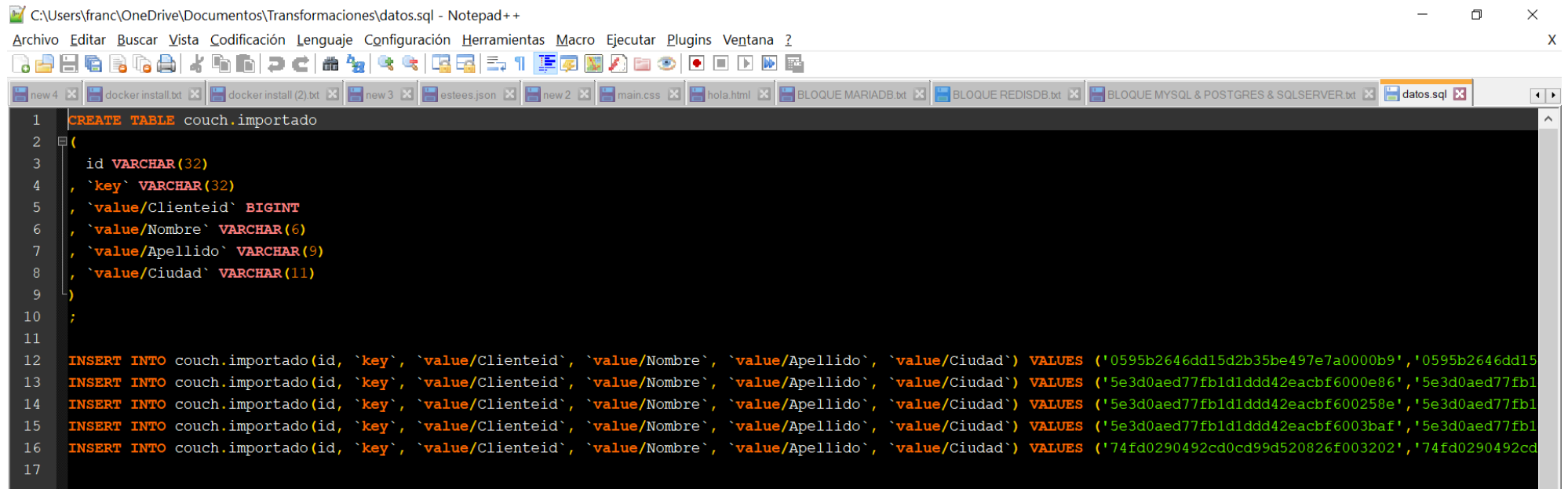


Imagen: Fuente propia ejecutando procesos de ETL

Observamos la estructura que se genera con el **SQL OUTPUT**.



```

1 CREATE TABLE couch.importado
2 (
3   id VARCHAR(32)
4   , `key` VARCHAR(32)
5   , `value/Clienteid` BIGINT
6   , `value/Nombre` VARCHAR(6)
7   , `value/Apellido` VARCHAR(9)
8   , `value/Ciudad` VARCHAR(11)
9 )
10 ;
11
12 INSERT INTO couch.importado(id, `key`, `value/Clienteid`, `value/Nombre`, `value/Apellido`, `value/Ciudad`) VALUES ('0595b2646dd15d2b35be497e7a0000b9', '0595b2646dd15
13 INSERT INTO couch.importado(id, `key`, `value/Clienteid`, `value/Nombre`, `value/Apellido`, `value/Ciudad`) VALUES ('5e3d0aed77fb1d1ddd42eacbf6000e86', '5e3d0aed77fb1
14 INSERT INTO couch.importado(id, `key`, `value/Clienteid`, `value/Nombre`, `value/Apellido`, `value/Ciudad`) VALUES ('5e3d0aed77fb1d1ddd42eacbf600258e', '5e3d0aed77fb1
15 INSERT INTO couch.importado(id, `key`, `value/Clienteid`, `value/Nombre`, `value/Apellido`, `value/Ciudad`) VALUES ('5e3d0aed77fb1d1ddd42eacbf6003baf', '5e3d0aed77fb1
16 INSERT INTO couch.importado(id, `key`, `value/Clienteid`, `value/Nombre`, `value/Apellido`, `value/Ciudad`) VALUES ('74fd0290492cd0cd99d520826f003202', '74fd0290492cd
17

```

Imagen: Fuente propia Archivo generado en formato SQL

Verificando datos en MySQL

Nos dirigimos a MySQL desde la interfaz Worbenck, y realizamos una consulta de datos, observamos que se han procesado correctamente. Y con esto nos damos cuenta que la información se ha transferido desde un gestor NoSQL de clave-valor, hacia un gestor relacional.

MySQL Workbench

Local instance MySQL80 x

File Edit View Query Database Server Tools Scripting Help

Navigator para SQL scripts de MySQL

SCHEMAS

Filter objects

- archivo_plano
- couch**
 - Tables
 - importado
 - Columns
 - id
 - key
 - value/Cienteid
 - value/Nombre
 - value/Apellido
 - value/Ciudad
 - Indexes
 - Foreign Keys

Administration Schemas

Information

Schema: couch

1 • `SELECT * FROM couch.importado;`

2

Result Grid

id	key	value/Cienteid	value/Nombre	value/Apellido	value/Ciudad
0595b2646dd15d2b35be497e7a0000b9	0595b2646dd15d2b35be497e7a0000b9	1	Junior	Franco	San Miguel
5e3d0aed77fb1d1ddd42eacb6000e86	5e3d0aed77fb1d1ddd42eacb6000e86	2	Jose	Julian	Usulután
5e3d0aed77fb1d1ddd42eacb600258e	5e3d0aed77fb1d1ddd42eacb600258e	3	Maria	carmelina	Morazan
5e3d0aed77fb1d1ddd42eacb6003baf	5e3d0aed77fb1d1ddd42eacb6003baf	4	Lolita	Sanchez	San Miguel
74fd0290492cd0cd99d520826f003202	74fd0290492cd0cd99d520826f003202	5	Carlos	Romero	San Jacinto

importado 6 x

Output

Action Output

#	Time	Action	Message
✓ 1	20:31:12	SELECT * FROM couch.importado LIMIT 0, 1000	5 row(s) returned
✓ 2	20:38:24	SELECT * FROM couch.importado LIMIT 0, 1000	5 row(s) returned

Imagen: Fuente propia Verificando datos en MySQL

Replicación local en CouchDB

Para iniciar con el proceso de replicación local en **CouchDB**, vamos a crear dos bases de datos, las cuales serán y funcionarán en modalidad maestro, lo que significa que si en la primer base de datos, agregamos un documento se va replicar en la segunda base de datos, y si en esta segunda base de datos, modificamos este mismo documento, se va replicar esa modificación en la primer base de datos, será una replicación Bidireccional.

✓ Crear una base de datos denominada maestro1:

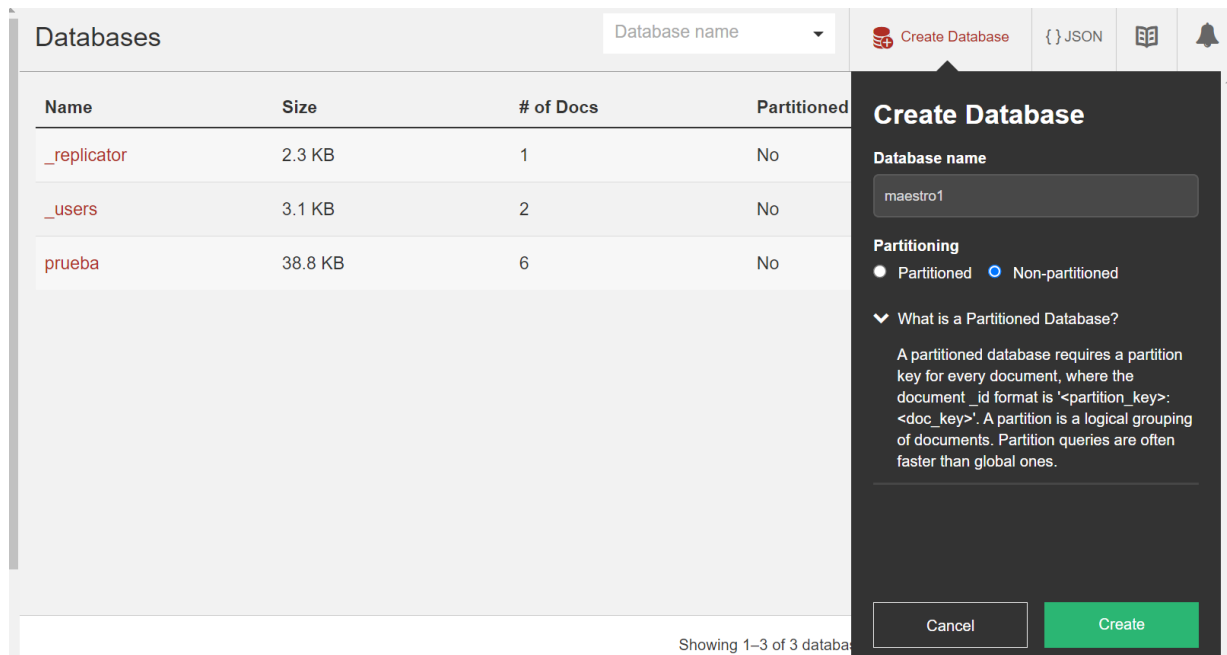


Imagen: Fuente propia Replicacion CouchDB

✓ Crear una segunda base de datos denominada maestro2:

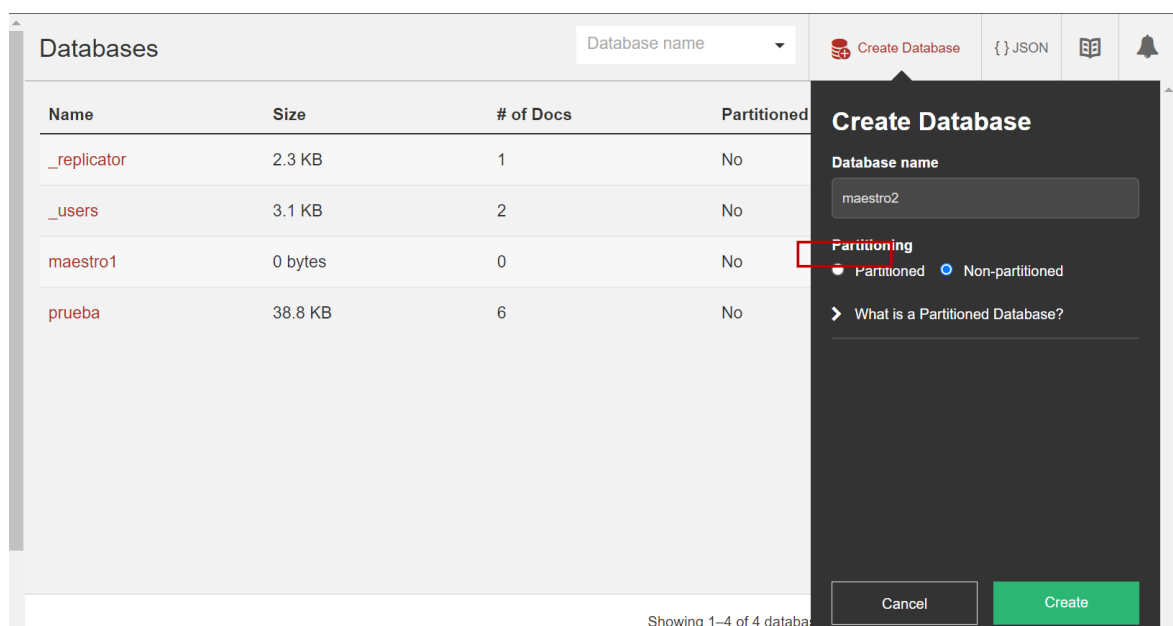


Imagen: Fuente propia Replicacion CouchDB

En el panel nos dirigimos al apartado llamado **"Replication"** y damos clic en **"New Replication"**:

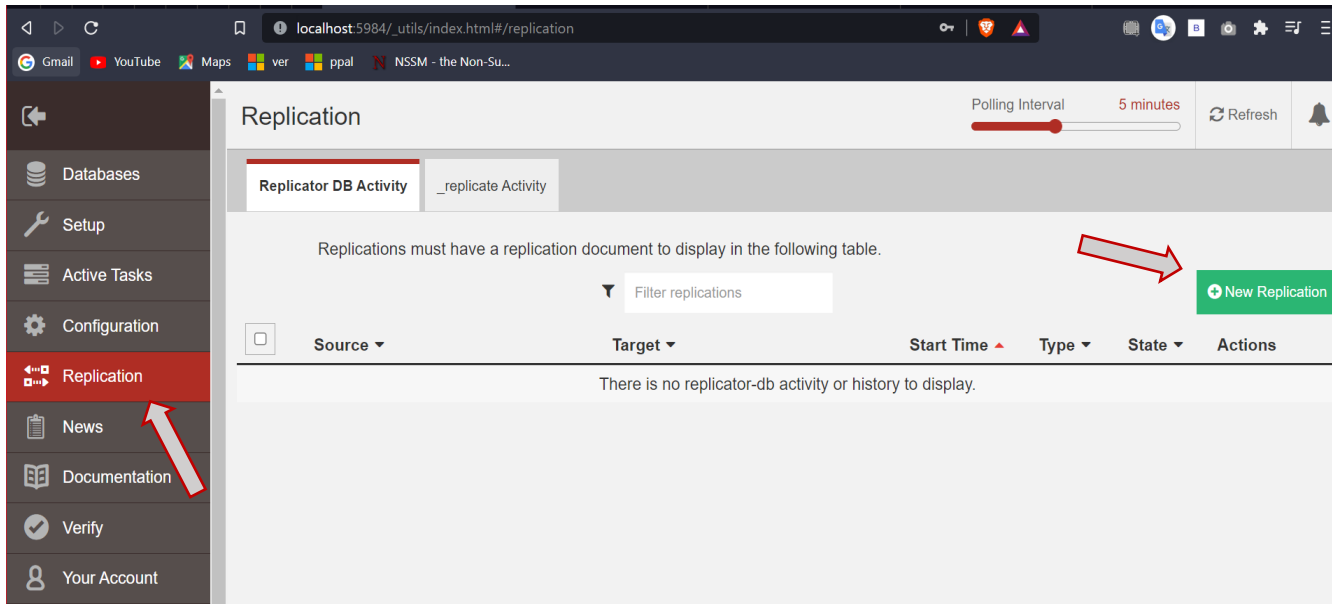


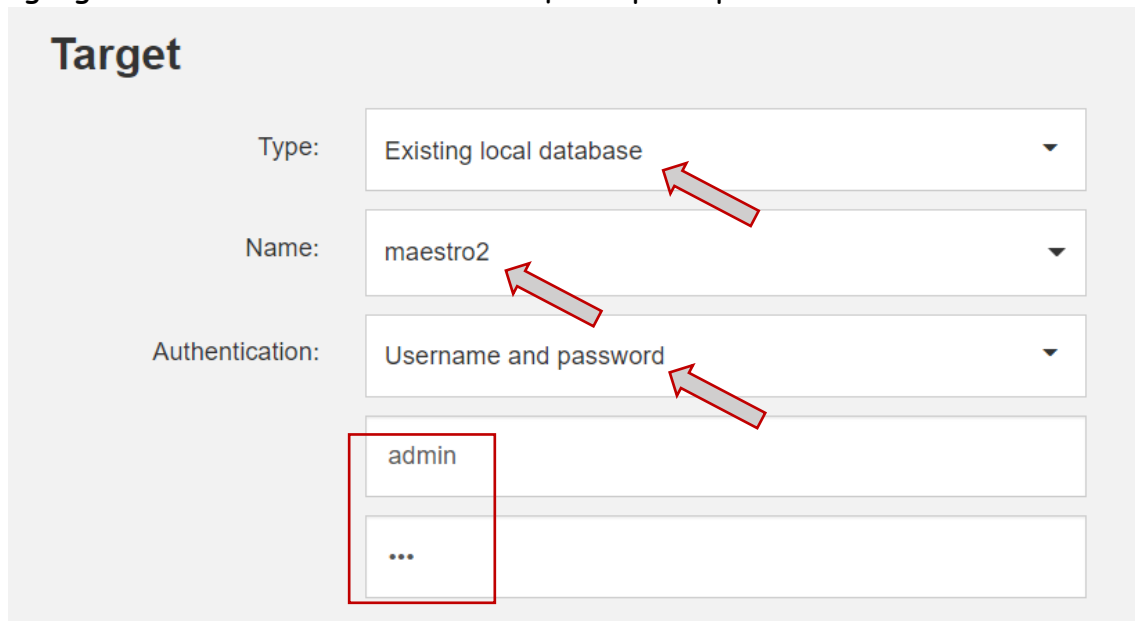
Imagen: Fuente propia Replicacion CouchDB

En el menú que se despliega vamos a configurar los siguientes apartados:

1. El origen, en este caso vamos a seleccionar la primera base de datos creada, "maestro1", con las credenciales que nos permiten ingresar al gestor, mismas que pusimos al momento de realizar la instalación en nuestro computador.

Imagen: Fuente propia Replicacion CouchDB

2. El destino, seleccionamos en el tipo una base de datos local y será "maestro2" agregamos las mismas credenciales que al principio.



Target

Type: Existing local database

Name: maestro2

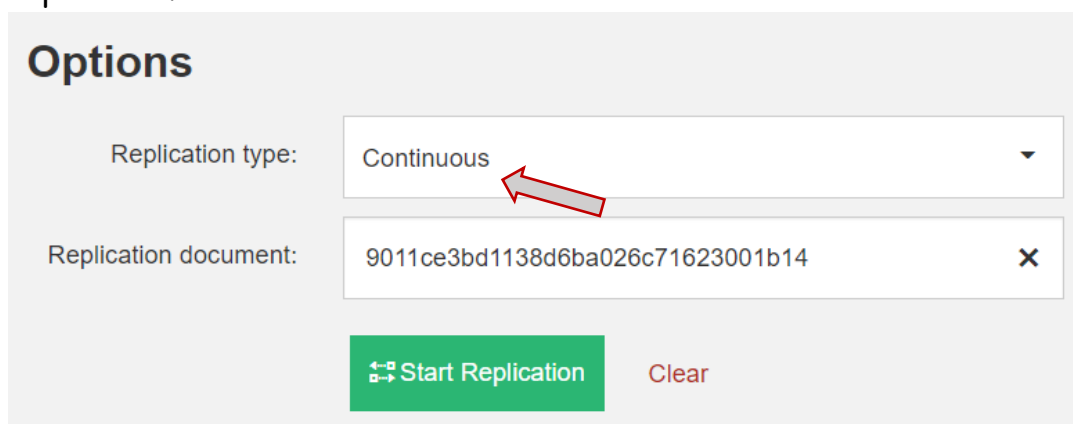
Authentication: Username and password

admin

...

Imagen: Fuente propia Replicacion CouchDB

3. Finalmente en el apartado de opciones, nos pregunta que tipo de replicación queremos realizar, y seleccionamos "continuamente". Ahora damos clic en iniciar la replicación.



Options

Replication type: Continuous

Replication document: 9011ce3bd1138d6ba026c71623001b14

Start Replication Clear

Imagen: Fuente propia Replicacion CouchDB

Nos volverá aparecer esta ventana y observamos que nos aparece la primera replicación configurada, hasta este punto ya tenemos una replicación unidireccional, es decir que nuestra base de datos maestro1, ya está replicando datos en el maestro2, para que sea bidireccional, vamos a crear una nueva replicación para que nuestro maestro1 también pueda replicar datos del maestro2.

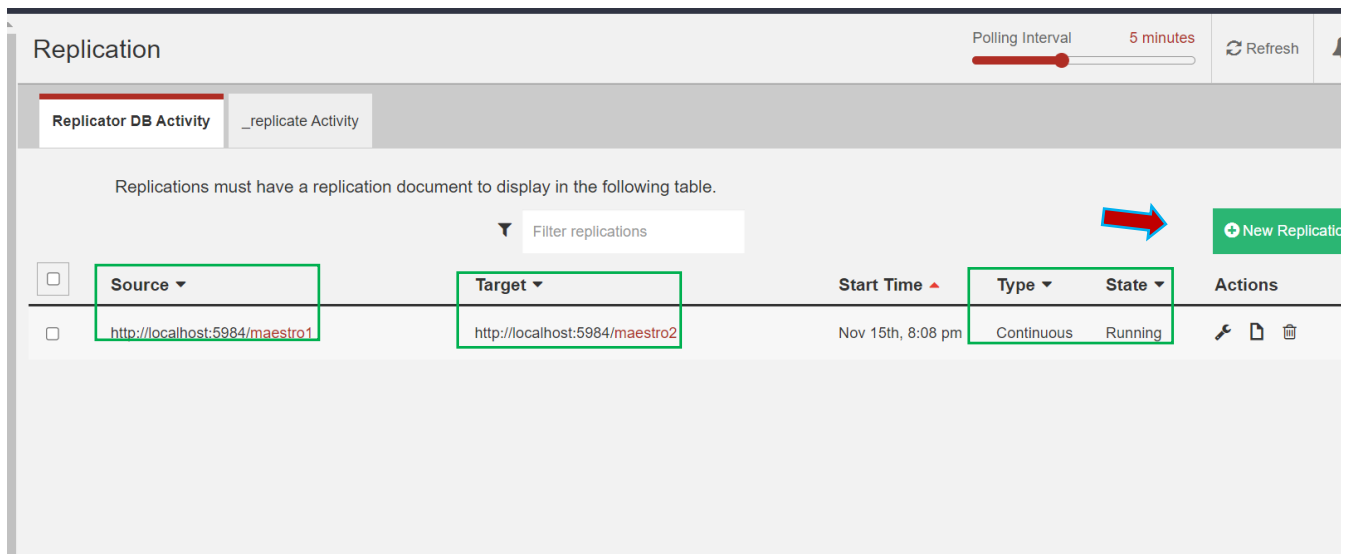


Imagen: Fuente propia Replicacion CouchDB

Ahora haremos lo inverso que realizamos al principio, de **maestro2**, tomaremos nuestro origen de datos, siempre con las credenciales que nos permiten ejecutar esta acción y seleccionar esta BD:

The screenshot shows the 'Source' configuration form. It has three dropdown menus: 'Type' (set to 'Local database'), 'Name' (set to 'maestro2'), and 'Authentication' (set to 'Username and password'). Below these are two input fields: the first contains 'admin' and the second contains '...'. A green box highlights the 'admin' field and the '...' field. Three red arrows point to the 'Type', 'Name', and 'Authentication' dropdowns.

Imagen: Fuente propia Replicacion CouchDB

En el destino seleccionamos la base de datos donde se replicará la data del maestro2, y de igual manera nos autenticamos.

Target

Type: Existing local database

Name: maestro1

Authentication: Username and password

admin

...

Imagen: Fuente propia Replicacion CouchDB

Finalmente seleccionamos la opción de replicación continua y damos clic en iniciar la replicación:

Options

Replication type: Continuous

Replication document: Custom ID (optional)

Start Replication

Clear

Imagen: Fuente propia Replicacion CouchDB

Y obtenemos el siguiente resultado, dos configuraciones de replicación:

Replication

Polling Interval: 5 minutes

Refresh

Replicator DB Activity

_replicate Activity

Replication from maestro2 to maestro1 has been scheduled.

Replications must have a replication document to display in the following table.

Filter replications

New Replication

Source	Target	Start Time	Type	State	Actions
http://localhost:5984/maestro1	http://localhost:5984/maestro2	Nov 15th, 8:08 pm	Continuous	Running	
http://localhost:5984/maestro2	http://localhost:5984/maestro1	Nov 15th, 8:10 pm	Continuous	Running	

Imagen: Fuente propia Replicacion CouchDB

Comprobando funcionalidad:

Ingresamos al maestro1 y creamos un nuevo documento:

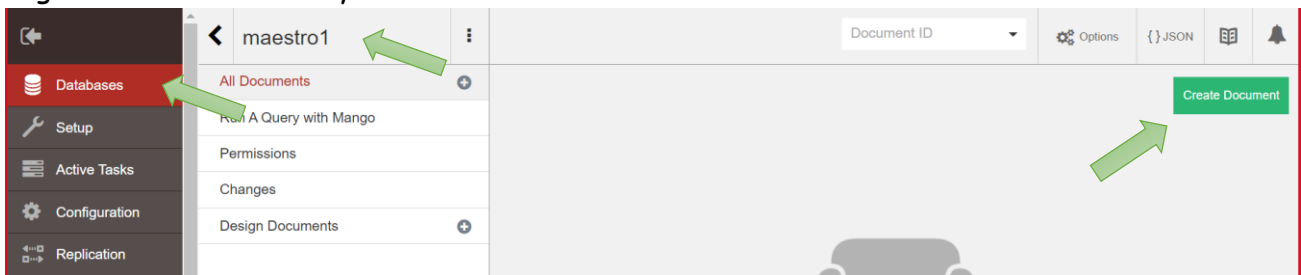


Imagen: Fuente propia Comprobando la Replicación CouchDB

En formato Json desde el documento, vamos a agregar una coma luego de id que nos genera por defecto y vamos a digitar dos campos, un nombre y un origen, con el valor replica y el campo "origen" como **maestro1**.

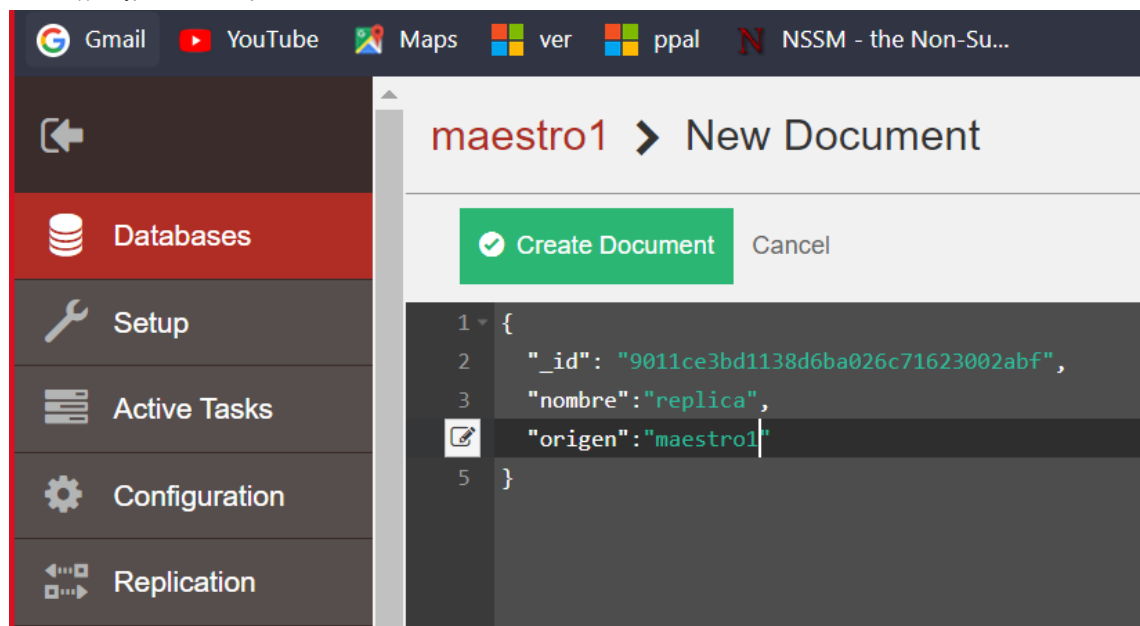


Imagen: Fuente propia Comprobando la Replicación CouchDB

Seguidamente nos dirigimos al maestro2, y observamos que nos ha replicado exitosamente el documento

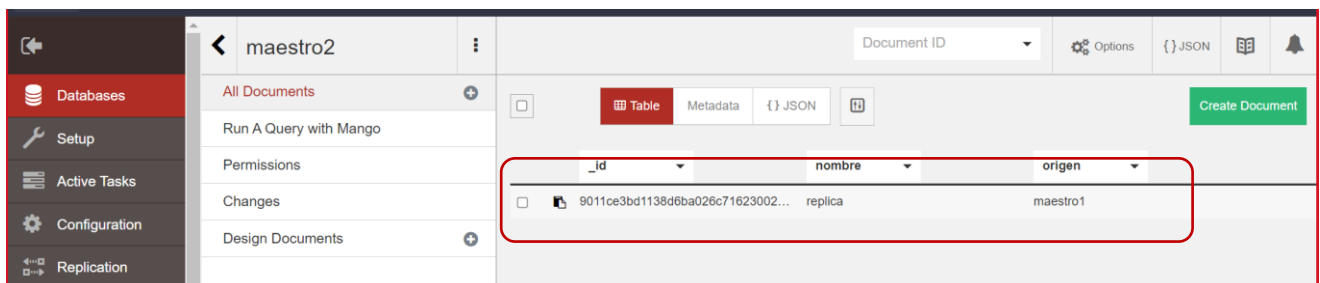


Imagen: Fuente propia Comprobando la Replicación CouchDB

Desde el maestro dos, abrimos el documento, y ahora le agregamos dos campos más, le agregaremos el campo nombre2, con el valor replica2, y seguidamente un origen2, con el valor maestro2.

Esto con el objetivo de validar, que los datos se repliquen de manera uniforme en ambas bases de datos locales:

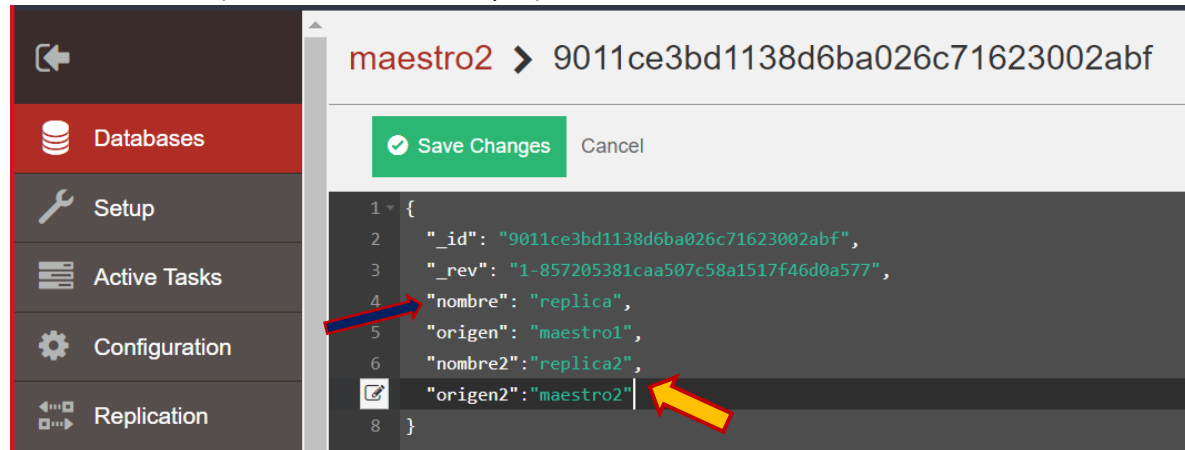


Imagen: Fuente propia Comprobando la Replicación CouchDB

Hemos regresado al maestro1, y ahora visualizamos que se ha modificado nuestro documento y que ahora tenemos los dos campos anezados en el documento; del paso anterior.

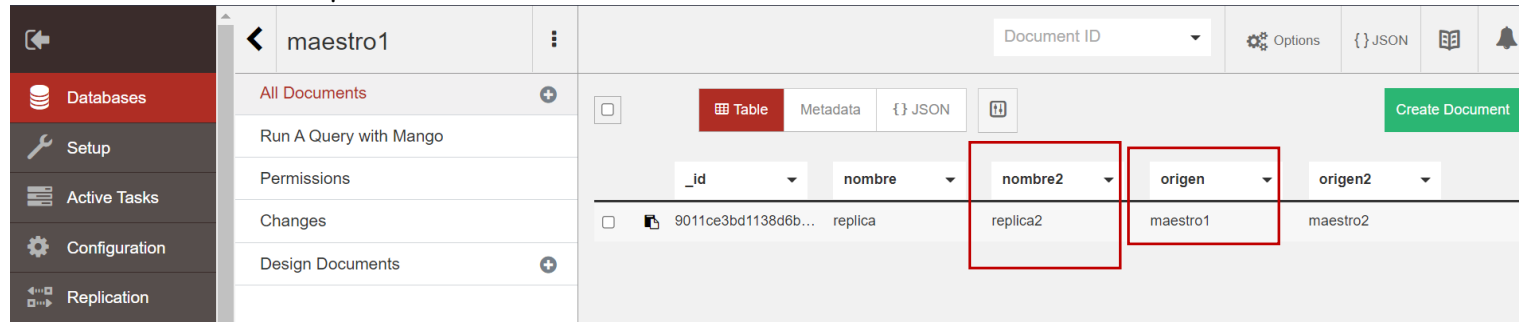


Imagen: Fuente propia Comprobando la Replicación CouchDB

De esta manera concluimos que la replicación se está realizando sin ningún inconveniente, que ahora tenemos comunicación entre dos bases de datos dentro del mismo servidor y que es una excelente práctica en entornos de producción puesto que aseguramos al doble una alta disponibilidad de los datos.

Además de que, sencillamente podemos agregar y configurar más entornos creando una estructura de beneficio según sea el caso en que se requiera implementar.

Referencias

Foundation, A. S. (2021). *CouchDB Homepage*. Obtenido de CouchDB Homepage:
<https://docs.couchdb.org/en/stable/whatsnew/index.html>

IONOS, D. G. (27 de Mayo de 2020). *presentación de CouchDB*. Obtenido de presentación de CouchDB:
<https://www.ionos.es/digitalguide/hosting/cuestiones-tecnicas/presentacion-de-couchdb/>

Sevilla, N. E. (2018). *ESTUDIO DE LA INTEGRACIÓN DEL LENGUAJE DE PROGRAMACIÓN PHP*. Obtenido de ESTUDIO DE LA INTEGRACIÓN DEL LENGUAJE DE PROGRAMACIÓN PHP: <https://core.ac.uk/download/pdf/200329976.pdf>

Zhunio, D. M. (Diciembre de 2011). *Estudio del uso de MongoDB como alternativa a las bases de datos relacionales*. Obtenido de Estudio del uso de MongoDB como alternativa a las bases de datos relacionales:
<http://repositorio.uisrael.edu.ec/bitstream/47000/140/1/UISRAEL-EC-SIS-378.242-349.pdf>