



#### ACTIVIDADES POR DESARROLLAR:

1. Introducción a las bases de datos no relacionales (NOSQL)
2. instalación de un sistema manejador de base de datos no relacional (MONGODB)
3. Comandos utilizados para MONGODB
4. Migrar base de datos relacionales a no relacional
5. Implementar desde PYTHON la conectividad hacia MONGODB

#### EVIDENCIA(S) A ENTREGAR:

1. Escribe un ensayo sobre que son las bases de datos NOSQL y aplicaciones existentes.
2. Elabora informe de la ejecución del plan de instalación de MONGODB.
3. Realiza los ejercicios propuestos en el presente taller.
4. Realiza la conversión de la base de datos SALUD hacia MONGODB y entrega el script de creación y el respectivo informe.
5. Desarrolla en Python un programa que se conecte a una base de datos con una colección llamada EPS y muestra por pantalla los datos guardados en MONGODB

**DURACIÓN: 24 HORAS 8 horas presenciales 16 horas trabajo autónomo.**

#### CONTROL DEL DOCUMENTO

	Nombre	Cargo	Dependencia	Fecha
Autor (es)	JOSE FERNANDO GALINDO SUAREZ	INSTRUCTOR	CGMLTI	14/09/2022

**CONTROL DE CAMBIOS** (diligenciar únicamente si realizan ajustes al taller)

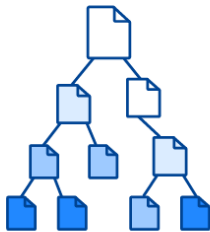
	Nombre	Cargo	Dependencia	Fecha	Razón del Cambio
Autor (es)					

## PRESENTACIÓN

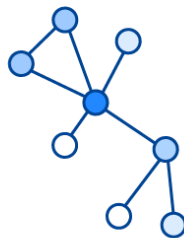
El auge de internet ha propiciado la generación de grandes volúmenes de datos de fuentes de información como sistemas de información, correos electrónicos, redes sociales, sensores de internet de las cosas; en diferentes formatos o sin formato alguno, a esto lo llamamos información estructurada, semiestructura y no estructuradas. Las bases de datos no relacionales llamadas NOSQL de la sigla “**NOT ONLY SQL**”, es un sistema manejador de base de datos para trabajos de BIGDATA orientada a colecciones y documentos, utilizando para los documentos el formato JSON (Java Script Object Notation).

## TIPO DE BASE DE DATOS NOSQL.

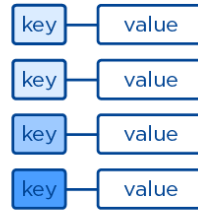
Document



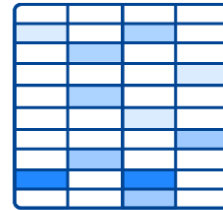
Graph



Key-Value



Wide-column



## CARACTERISTICAS DE LAS BASES DE DATOS NOSQL



- Normalmente son distribuidas
- Escalables
- Flexibles
- No soportan subconsulta y uniones de tablas (JOIN)
- No soportan los métodos ACID (Atomicity, Consistency, Isolation, Durability)
- Garantiza el paradigma BASE (Base Ability, Soft State, Eventual Consistency)



## PRACTICA MONGODB

Para comenzar la práctica, se debe descargar los archivos con GIT (ver “VIDEOS/Instalación/ clonarrepositoriogit.cmd” o sin GIT (ver “VIDEOS/Instalación/ descargar archivos mongodb.cmd” ). Los archivos se encuentran en: <http://github.com/fegasu/EDTMONGODB> o descargar [Aquí](#)

Descargar los siguientes archivos de instalación:

- Servidor MongoDB (SW/mongoserver.cmd)
- Consola MongoDB (SW/mongosh.cmd)
- Mongo Compas (SW/mongocompass.cmd)

## INSTALACION SERVIDOR MONGODB

Para la instalación del servidor MONGODB se recomienda ver el video: “VIDEOS/Instalación/mongoinstalacion.cmd”

## INSTALACION CONSOLA MONGODB

Para la instalación de la consola de MONGODB se recomienda ver el video: “VIDEOS/Instalación/instalarmongosh.cmd”

## MOSTRAR LAS BASE DE DATOS CREADAS

Ingresar a MONGOSH y digitar el comando:

```
show databases;
```

## CREAR UNA BASE DATOS

```
use prueba;
```

Tener en cuenta cuando se escriba db se refiere a la base de datos que se está usando. El comando use se debe utilizar para seleccionar una base de datos.

## CREANDO UNA COLECCIÓN

Una colección guarda un documento o muchos documentos, utilice el siguiente comando:

```
db.createCollection("emp")
```





## INSERTAR UN DOCUMENTO DENTRO DE UNA COLECCIÓN

Se utiliza la clausula “insertOne” de la siguiente manera para insertarlo dentro de la colección “emp”, para esto se debe escribir el documento en el formato JSON así:

```
emp1={  
  "idemp": 1,  
  "nom": "Pedro Palitos",  
  "genero": "M",  
  "edad": 23,  
  "dni": 19407970  
}
```

Antes de insertarlo se debe verificar que el JSON fue escrito de forma correcta, para esto se debe verificar con la pagina [Validador y formateador JSON Online - JSON Lint](#).





```
1 {  
2   "idemp": 1,  
3   "nom": "Pedro Palitos",  
4   "genero": "M",  
5   "edad": 23,  
6   "dni": 19407970  
7 }
```

**Resultados**

Valid JSON

Para insertarlo dentro de la colección “emp” de la base de datos prueba

```
db.emp.insertOne(emp1);
```

Inserte los siguientes datos:

idemp	Nom	Genero	Edad	dni
1	Pedro Palitos	M	23	19456970
2	Maria Casquito	F	32	
3	Celia Gacha	F	43	



## INSERTANDO MAS DE UN DOCUMENTO A LA VEZ

Construimos los documentos emp4 y emp5 así:

```
emp4={
  "idemp":4,
  "nom":"Maria la bandida",
  "genero":"F",
  "edad":32
}
emp5={
  "idemp": 5,
  "nom": "ROSA MELO",
  "genero": "F",
  "edad":25
}
```

Utilizamos la cláusula “insertMany” así:

```
db.emp.insertMany([emp4,emp5])
```

También podemos hacerlo así:

```
db.emp.insertMany([
  {
    "idemp":6,
    "nom":"Juan Tolomeo",
    "genero":"M",
    "edad":37
  },
  {
    "idemp": 7,
    "nom": "Carlos Pichamata",
    "genero": "M",
    "edad":52
  }
])
```

Ingresar los siguientes empleados:

Idemp	Nom	Genero	Edad	dni
8	Manuela Beltran	F	15	123456
9	Monica Galindo	F	23	6789123
10	Elver Galindo	M	21	



11	Simón Tolomeo	M	51	1357901
----	---------------	---	----	---------

## LISTAR TODOS LOS DOCUMENTOS DE UNA COLECCIÓN

Se utiliza la cláusula “find()” así:

```
db.emp.find();
```

Para buscar con una condición simple, por ejemplo, buscar el empleado 6:

```
db.emp.find({  
  idemp:"6"  
})
```

Para encontrar los empleados mayores a 30

```
db.emp.find(  
  {edad:{$gt:30}}  
)
```

Para encontrar los empleados mayores o iguales a 30

```
db.emp.find(  
  {edad:{$gte:30}}  
)
```

Para encontrar los empleados menores a 30

```
db.emp.find(  
  {edad:{$lt:30}}  
)
```

Para encontrar los empleados menores o iguales a 30

```
db.emp.find(  
  {edad:{$lte:30}}  
)
```

Para encontrar los empleados de 23, 42 y 21

```
db.emp.find(  
  {  
    edad:{$in:[23,42,21]}  
  }  
)
```

## UTILIZANDO BUSQUEDAS CON OPERADORES LOGICOS

Utilizando el operador lógico AND

```
db.emp.find({  
  $and:[  
    {edad:{$lte:35}},  
    {genero:"F"}  
  ]  
})
```





Utilizando el operador lógico OR

```
db.emp.find({
  $or:[
    {edad:{$lte:35}},
    {genero:"F"}]
})
```

Encuentre los empleados que tienen "dni":

```
db.emp.find(
  {
    dni:{$exists:true}
  }
)
```

## ORDENANDO LA SALIDA DE LOS DOCUMENTOS

De forma descendente:

```
db.emp.find().sort(
  {
    edad:-1
  }
)
```

Encontrar el empleado que tiene la máxima edad

```
db.emp.find().sort(
  {
    edad:-1
  }
).limit(1)
```

## UTILIZAR EXPRESIONES REGULARES

Encontrar empleados que contenga la palabra "Simón"

```
db.emp.find(
  {
    nom:/^ Simón /
  }
)
```

Encontrar empleados que contenga la palabra "ver"

```
db.emp.find(
  {
    nom:/ver/
  }
)
```

Cuantos documentos hay en la colección "emp"







```
db.emp.find().count()
```

7

Cuantos empleados hay con una edad menor o igual a 25

```
db.emp.find({edad:{$lte:25}}).count()
```

4

Mostrar dos documentos a partir del tercer documento

```
db.emp.find().skip(2).limit(2)
```

## MOSTRAR EL NOMBRE DE LOS DOCUMENTOS DE LA COLECCION

```
db.emp.find().forEach(df =>print(df.nom))
```

## ASIGNAR A UNA VARIABLE UN DOCUMENTO

```
var user=db.emp.findOne()  
user.nom='PEDRO PALOS GORDOS'
```

## ACTUALIZAR UN DOCUMENTO

```
db.emp.updateOne(  
  {idemp: 1} ,  
  {$set:{"nom":" Juan Charrasqueado"}}  
)
```

## ADICIONAR UN NUEVO ATRIBUTO A UN DOCUMENTO

```
db.emp.updateOne(  
  {idemp: 1} ,  
  {$set:{"Correo":"fegasu@misena.edu.co"}}  
)
```

## INCREMENTAR UN ATRIBUTO EN TODOS LOS DOCUMENTOS

```
db.emp.updateMany(  
  {} ,  
  {$inc:{edad:1}}  
)
```

## DECREMENTAR UN ATRIBUTO EN TODOS LOS DOCUMENTOS

```
db.emp.updateMany(  
  {} ,  
  {$inc:{edad:-1}}  
)
```

## BORRAR UN DOCUMENTO

redis



couchDB



neo4j



hBASE

riak



Cassandra



MongoDB



Sleek





```
db.emp.deleteOne(  
  {idemp:7}  
)
```

## ADICIONAR ATRIBUTOS A UN DOCUMENTOS

```
Emp12={  
  "idemp":12,  
  "nom":"Carlos Villagran",  
  "edad":62,  
  "dni":{  
    "tipo":"CC",  
    "numero":19407906,  
    "fecha":"14/07/1979"  
  },  
  "cursos":["python","java","php","mongodb"],  
  "mensajes":[  
    {  
      "subject":"Reunión de planeación",  
      "body":"agenda de desarrollo web",  
      "from":"sotico@gmail.com",  
      "fecha":"2022-02-27"  
    },  
    {  
      "body":"Entrega de stickholder del proyecto",  
      "from":"cpinilla@gmail.com",  
      "fecha":"2022-03-12"  
    }  
  ]  
}  
db.emp.insertOne(emp12)
```

## AGREGA LA PROPIEDAD A LOS DOCUMENTOS QUE NO LO TENGA.

Agrega la propiedad dni a los documentos que no lo tienen

```
db.emp.updateMany(  
  {  
    'dni':{$exists:false}  
  },  
  {  
    $set:{'dni':{  
      'tipo':'CC',  
      'numero':0,  

```





```
'fecha': ''  
}  
}  
}  
)
```

## ADICIONA DOCUMENTOS AL FINAL.

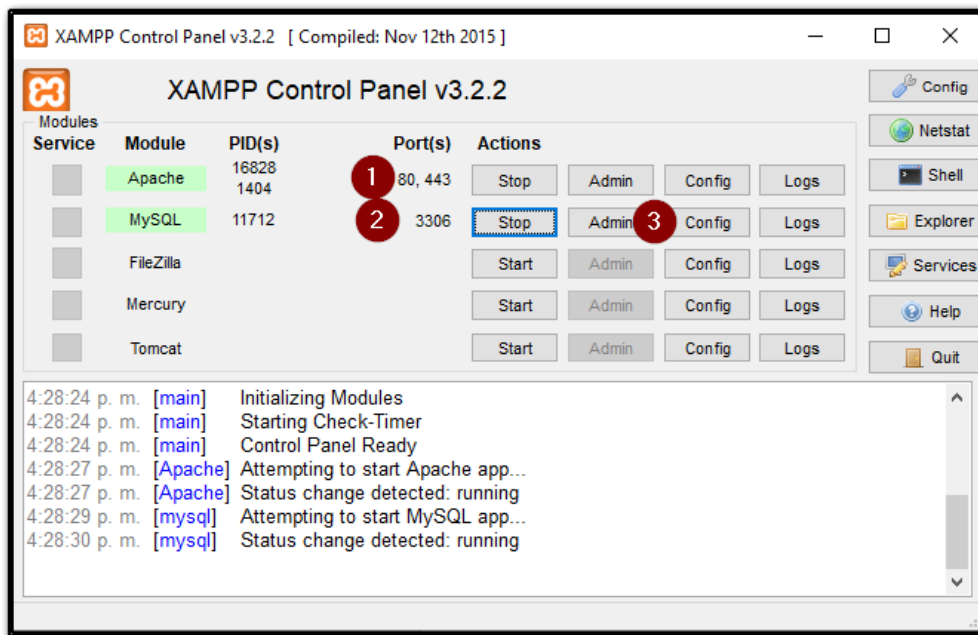
```
db.emp.updateOne(  
  {  
    idemp:9  
  },  
  {$push:{  
    mensajes:{  
      subject:"Pruebas TYT",  
      body:"Pruebas TyT Tecnologos Sena",  
      from:"secretaria@sena.edu.co",  
      fecha:"2022-08-27"  
    }  
  }  
})
```

## Migrar una base de datos MySQL a una base de datos MongoDB



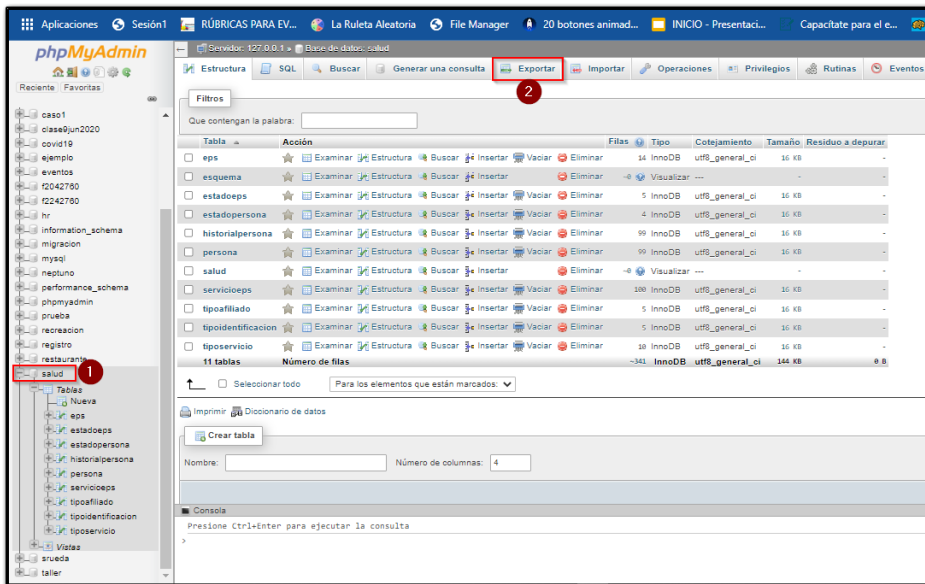
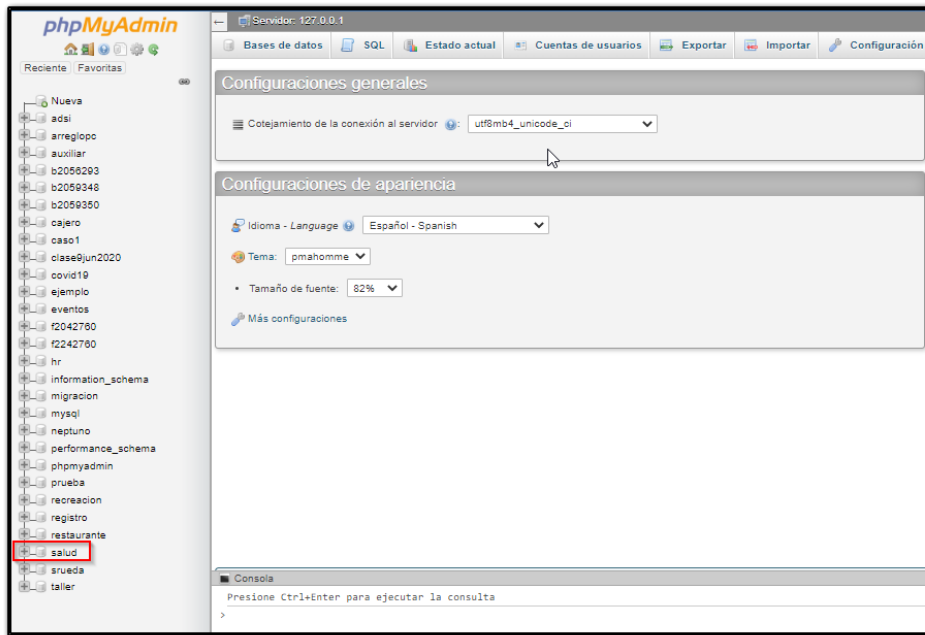


Servicio Nacional de Aprendizaje  
Formato Taller  
Centro de Gestión de Mercados, Logística y Tecnologías de la Información.



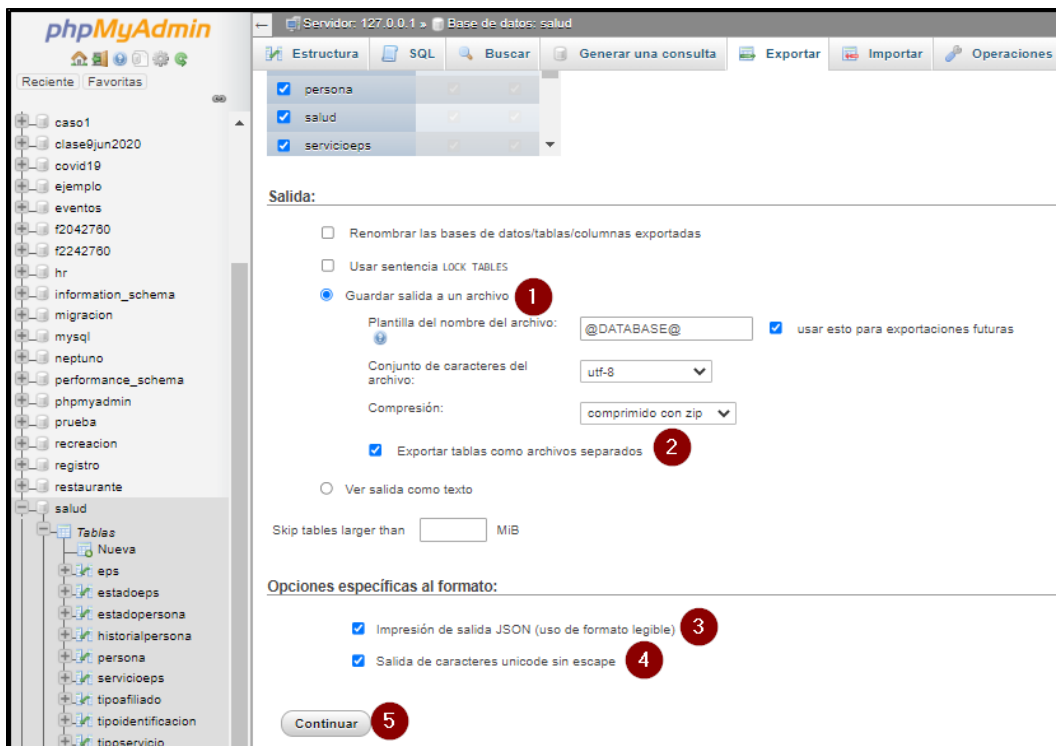
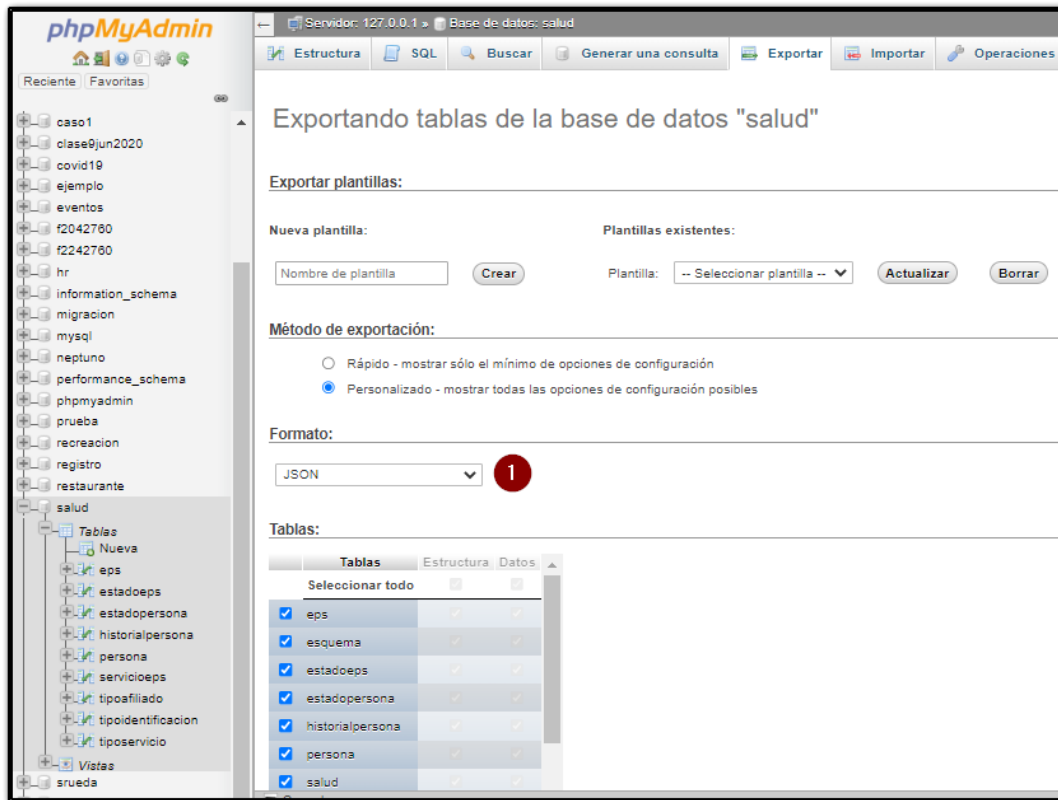


Servicio Nacional de Aprendizaje  
Formato Taller  
Centro de Gestión de Mercados, Logística y Tecnologías de la Información.





Servicio Nacional de Aprendizaje  
Formato Taller  
Centro de Gestión de Mercados, Logística y Tecnologías de la Información.





## CONVIRTIENDO CSV A JSON

Desde phpmyadmin convertimos la base de datos Neptuno a formato JSON

Utilizar la herramienta online [Convertidor de CSV a JSON \(convertcsv.com\)](https://convertcsv.com)

The screenshot shows the convertcsv.com website. On the left, there are navigation links for various file formats (CSV, Word, XML, YML, etc.) and data tools (CSV/Excel, Herramientas de datos). The main area has a text input field for the CSV file. Below it, there are buttons for 'Borrar entrada', 'Ejemplo 1', and 'Ejemplo 2'. The steps for conversion are listed: Paso 2: Elige las opciones de entrada (opcional), Paso 3: Elija las opciones de salida (opcional), Paso 4: Crear resultados personalizados a través de la plantilla (opcional), and Paso 5: Generar salida. Under 'Elija el tipo de conversión', there are four buttons: 'CSV a JSON', 'CSV a JSON con clave', 'Matriz CSV a JSON', and 'Matriz de columnas CSV a JSON'. The 'Datos de resultados' section shows a download icon and the text 'Resultados de la producción'.

Convertimos la tabla categoría

```
[
{
  "IdCategoria": 1,
  "NombreCategoria": "Bebidas",
  "Descripcion": "Gaseosas, caf?, t?, cervezas y maltas"
},
{
  "IdCategoria": 2,
  "NombreCategoria": "Condimentos",
  "Descripcion": "Salsas dulces y picantes, delicias, comida para untar y aderezos"
},
{
  "IdCategoria": 3,
  "NombreCategoria": "Reposter?a",
  "Descripcion": "Postres, dulces y pan dulce"
},
{
  "IdCategoria": 4,
  "NombreCategoria": "L?cteos",
  "Descripcion": "Quesos"
}
```



```
},  
{  
  "IdCategoria": 5,  
  "NombreCategoria": "Granos/Cereales",  
  "Descripcion": "Pan, galletas, pasta y cereales"  
},  
{  
  "IdCategoria": 6,  
  "NombreCategoria": "Carnes",  
  "Descripcion": "Carnes preparadas"  
},  
{  
  "IdCategoria": 7,  
  "NombreCategoria": "Frutas/Verduras",  
  "Descripcion": "Frutas secas y queso de soja"  
},  
{  
  "IdCategoria": 8,  
  "NombreCategoria": "Pescado/Marisco",  
  "Descripcion": "Pescados, mariscos y algas"  
}  
]
```

## CONECTANDO DESDE PYTHON A MONGODB

Se necesita instalar la biblioteca pymongo así:

```
Pip install pymongo
```

Realizar el siguiente código en un archivo llamado: "MongoCliente.py"

```
from pymongo import MongoClient  
MONGO_URI='mongodb://localhost'  
cliente=MongoClient(MONGO_URI)  
db=cliente['prueba']  
coleccion=db['emp']  
eps1=coleccion.find()  
for r in eps1:  
    print(r['idemp']+"-"+r['nom'])
```

Ejecute el programa

```
Python MongoCliente.py
```





## RETO POR DESARROLLAR

- Cargar los datos que se encuentran en “DATOS/COVID19/COVID19-JULIO2020.csv” en una base de datos MONGODB llamada “COVID” en una colección llamada “CASOS”.
- Realizar las siguientes consultas:
  - Cuantos casos se registraron en Bogotá
  - Cuantos casos se recuperaron en Bogotá
  - Cual fue la ciudad que más se recuperaron
  - Cual fue la ciudad que más se recuperaron siendo mujeres.
  - Cuantos fallecidos que fueron mujeres y mayores de 50 años.
  - Cuantos fallecidos que fueron hombres y menores de 50 años.
  - Cuantos casos fueron por personas del extranjero
  - Cuantos casos se registraron en Antioquía, menores de 30 años y por persona extranjeras
  - Cuantos casos se registraron por personas de España
  - Cuantos casos se registraron por personas de edad entre 20 y 50 años
- Realizar un programa que muestre cada una de las consultadas solicitadas en el punto anterior.

## PRODUCTOS SOLICITADOS.

- Documento de tipo texto con las consultas solicitadas
- Documento fuente del programa Python
- Informe grafico de desarrollo del taller.

## MODO DE ENTREGA.

Se debe empaquetar los productos solicitados en un archivo empaquetado llamado “TallerNoSQL.zip”

Realizado por el instructor José Fernando Galindo Suarez  
Centro de Gestión de Mercados, Logística y Tecnologías de la información  
Jgalindos @sena.edu.co ©2022





Servicio Nacional de Aprendizaje

Formato Taller

Centro de Gestión de Mercados, Logística y Tecnologías de la Información.



redis



couchDB



neo4j



hbase



riak



Cassandra



MongoDB



Sleek

