Recuerda marcar tu asistencia

Asistencia en LMS

Semana 3 Sesión 01

Bases de datos no relacionales

Ing Edwin Garcia

El futuro digital es de todos

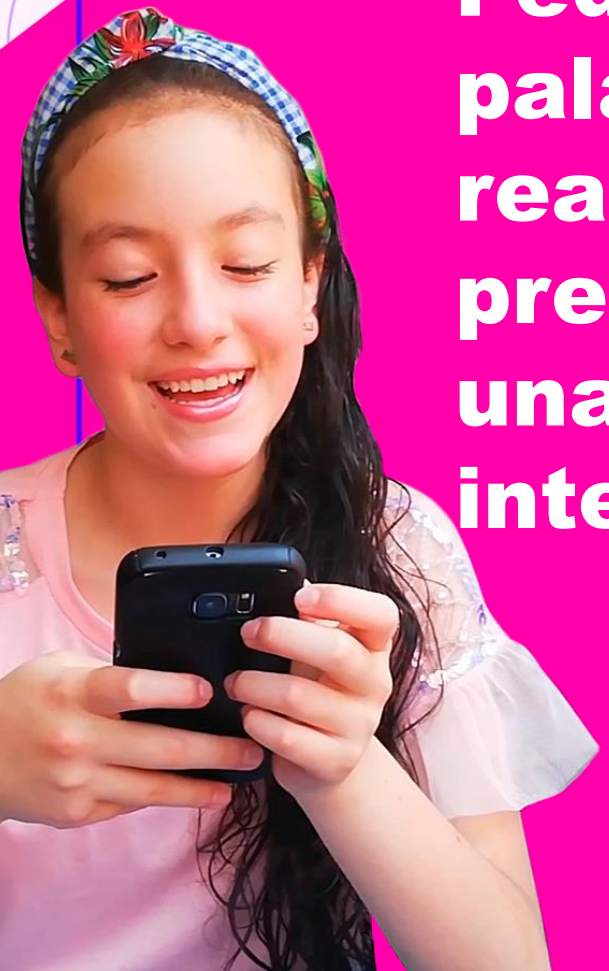
MinTIC

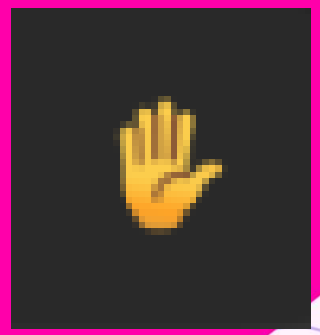
Hechos QUE CONECTAN

Recuerden marcar su asistencia

Asistencia en LMS

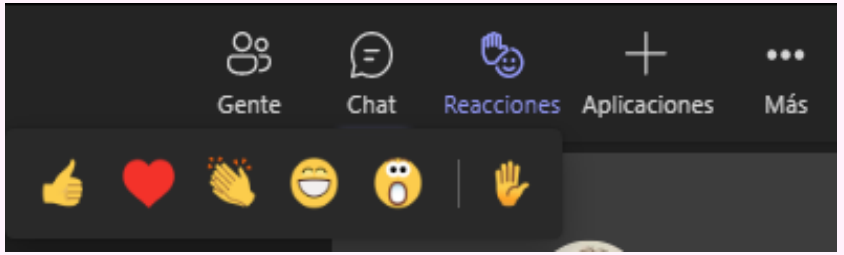
Pedir la palabra para realizar una pregunta o una intervención

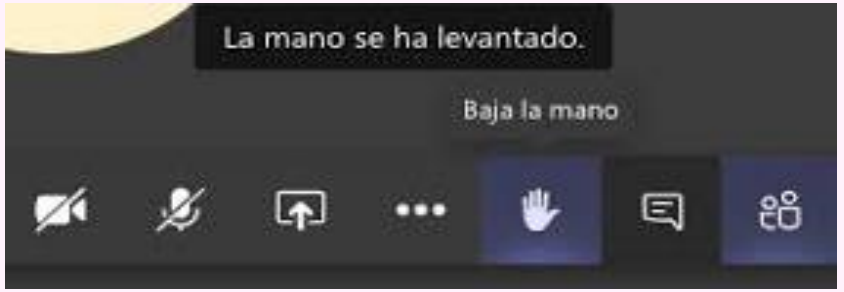




El futuro digital es de todos

MinTIC





Misión TIC 2022

unab
VIGILADA MINEDUCACIÓN

Hechos QUE CONECTAN

¿Qué son las bases de datos NoSQL?

Las bases de datos NoSQL están diseñadas específicamente para modelos de datos específicos y tienen esquemas flexibles para crear aplicaciones modernas

Las bases de datos NoSQL son ampliamente reconocidas porque son fáciles de desarrollar, por su funcionalidad y el rendimiento a escala.

¿Cómo funciona una base de datos NoSQL (no relacionales)?

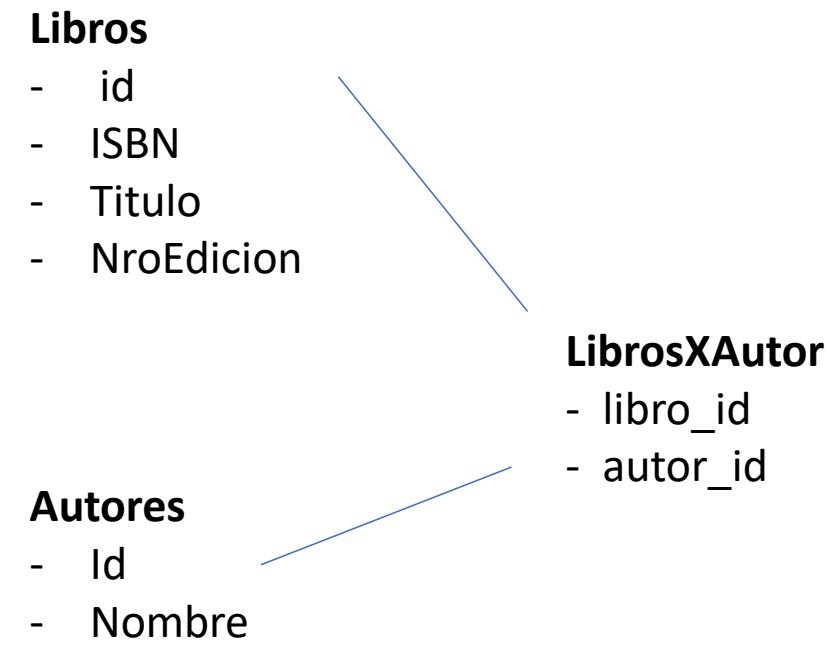
Las bases de datos NoSQL utilizan una variedad de modelos de datos para acceder y administrar datos.

Estos tipos de bases de datos están optimizados específicamente para aplicaciones que requieren grandes volúmenes de datos, baja latencia y modelos de datos flexibles, lo que se logra mediante la flexibilización de algunas de las restricciones de coherencia de datos en otras bases de datos.

Considere el ejemplo de modelado del esquema para una base de datos simple de libros:

En una base de datos relacional, un registro de libros a menudo se enmascara (o "normaliza") y se almacena en tablas separadas, y las relaciones se definen mediante restricciones de claves primarias y externas.

En este ejemplo, la tabla **Libros** tiene las columnas **ISBN**, **Título del libro** y **Número de edición**, la tabla **Autores** tiene las columnas **IDAutor** y **Nombre de autor** y, finalmente, la tabla **Autor-ISBN** tiene las columnas **IDAutor** e **ISBN**.



El modelo relacional está diseñado para permitir que la base de datos aplique la integridad referencial entre tablas en la base de datos, normalizada para reducir la redundancia y, generalmente, está optimizada para el almacenamiento.

id	isbn	Titulo	edicion	autor
1	001	libro1	01	Edwin, eddy
2	002	libro2	01	Edwin,eddy

id	Libro_id	Autor_id
1	1	1
2	1	2

id	nombre
1	edwin
2	Eddy

En una base de datos NoSQL, el registro de un libro generalmente se almacena como un documento JSON. Para cada libro, el elemento, **ISBN**, **Título del libro**, **Número de edición**, **Nombre autor** y **IDAutor** se almacenan como atributos en un solo documento. En este modelo, los datos están optimizados para un desarrollo intuitivo y escalabilidad horizontal.

- {“ISBN”,
- “Nombre”,
- “Edicion”,
- “Autor”,
- “Id_autor”

- }

¿Por qué debería usar una base de datos NoSQL?

Las bases de datos NoSQL se adaptan perfectamente a muchas aplicaciones modernas, como dispositivos móviles, web y juegos, que requieren bases de datos flexibles, escalables, de alto rendimiento y altamente funcionales para proporcionar excelentes experiencias de usuario.

Flexibilidad: las bases de datos NoSQL generalmente ofrecen esquemas flexibles que permiten un desarrollo más rápido y más iterativo. El modelo de datos flexible hace que las bases de datos NoSQL sean ideales para datos semiestructurados y no estructurados.

Escalabilidad: las bases de datos NoSQL generalmente están diseñadas para escalar usando clústeres distribuidos de hardware en lugar de escalar añadiendo servidores caros y sólidos. Algunos proveedores de la nube manejan estas operaciones en segundo plano, como un servicio completamente administrado.

Alto rendimiento: la base de datos NoSQL está optimizada para modelos de datos específicos y patrones de acceso que permiten un mayor rendimiento que el intento de lograr una funcionalidad similar con bases de datos relacionales.

Altamente funcional: las bases de datos NoSQL proporcionan API altamente funcionales y tipos de datos que están diseñados específicamente para cada uno de sus respectivos modelos de datos.

Tipos de bases de datos NoSQL

Clave-valor: las bases de datos clave-valor son altamente divisibles y permiten escalado horizontal a escalas que otros tipos de bases de datos no pueden alcanzar. Los casos de uso como juegos, tecnología publicitaria e IoT se prestan particularmente bien con el modelo de datos clave-valor.

```
{
  “clave”, “Valor”,
 “clave1, “valor”
},
{
  “clave”, “Valor”
},
{
  “clave”, “Valor”,
  “clave”, “Valor”,
  “clave”, “Valor”

}
```

Documentos: en el código de aplicación, los datos se representan a menudo como un objeto o un documento de tipo JSON porque es un modelo de datos eficiente e intuitivo para los desarrolladores. Las bases de datos de documentos facilitan a los desarrolladores el almacenamiento y la consulta de datos en una base de datos mediante el uso del mismo formato de modelo de documento que emplean en el código de aplicación.

```
?{

}
```

```
Documento{
   “clave”:
  “clave1”:
   “clave2”:
}
```

```
documento,   {
              “clave”, “Valor”,
             “clave1, “valor”,
              “clave2”, nulo
             },
             {
              “clave”, “Valor”
documento,   “clave1, nulo
              “clave2”,nulo

             },
             {
documento,    “clave”, “Valor”,
              “clave1”, “Valor”,
              “clave2”, “Valor”

             }
```

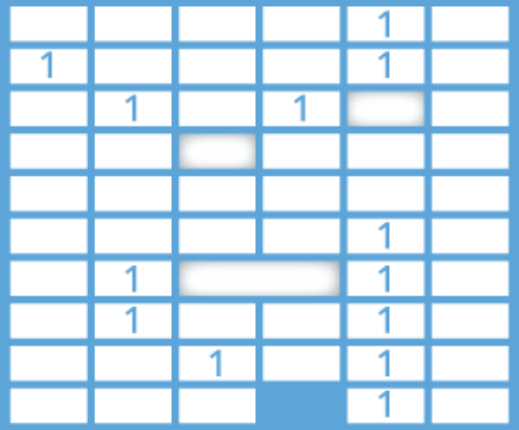
La naturaleza flexible, semiestructurada y jerárquica de los documentos y las bases de datos de documentos permite que evolucionen según las necesidades de las aplicaciones. El modelo de documentos funciona bien con catálogos, perfiles de usuario y sistemas de administración de contenido en los que cada documento es único y evoluciona con el tiempo.

Gráficos: el propósito de una base de datos de gráficos es facilitar la creación y la ejecución de aplicaciones que funcionan con conjuntos de datos altamente conectados. Los casos de uso típicos para una base de datos de gráficos incluyen redes sociales, motores de recomendaciones, detección de fraude y gráficos de conocimiento.

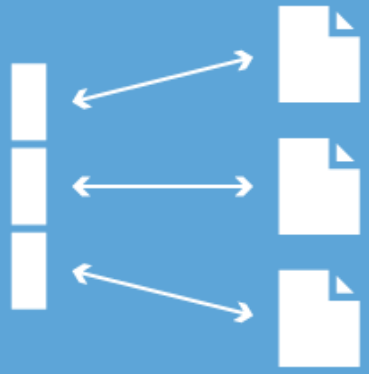
Key-Value

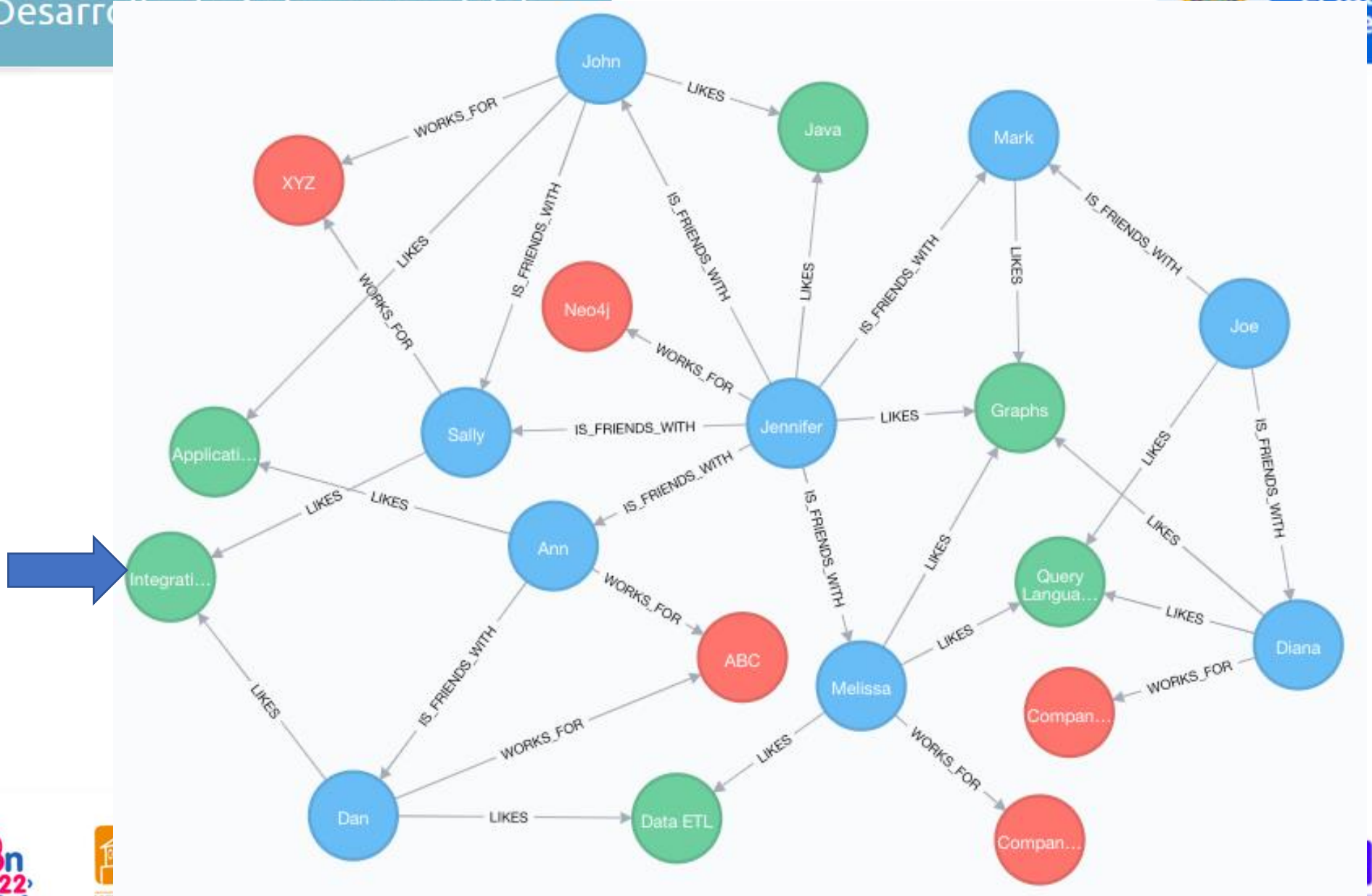
Graph DB

Column Family



Document




John
Java
XYZ
Mark
Neo4j
Joe
Graphs
Applicati...
Sally
Jennifer
Integrati...
Ann
Query Langua...
Diana
ABC
Melissa
Compan...
Dan
Data ETL
Compan...
WORKS_FOR
LIKES
IS_FRIENDS_WITH

En memoria: las aplicaciones de juegos y tecnología publicitaria tienen casos de uso como tablas de clasificación, tiendas de sesión y análisis en tiempo real que requieren tiempos de respuesta de microsegundos y pueden tener grandes picos de tráfico en cualquier momento.

Buscar: muchas aplicaciones generan registros para ayudar a los desarrolladores a solucionar problemas.

SQL (relacional) en comparación con NoSQL (no relacional)

Durante décadas, el modelo de datos predominante utilizado para el desarrollo de aplicaciones era el modelo de datos relacional empleado por bases de datos relacionales como Oracle, DB2, SQL Server, MySQL y PostgreSQL.

No fue sino hasta mediados y finales de la década del 2000 que otros modelos de datos comenzaron a adoptarse y aumentó su uso significativamente. Para diferenciar y categorizar estas nuevas clases de bases de datos y modelos de datos, se acuñó el término "NoSQL". Con frecuencia, los términos "NoSQL" y "no relacional" se usan indistintamente.

Aunque hay muchos tipos de bases de datos NoSQL con distintas características, en la tabla siguiente se muestran algunas de las diferencias entre las bases de datos SQL y NoSQL.

	SQL	NoSQL
Cargas de trabajo óptimas	Las bases de datos relacionales están diseñadas para aplicaciones de procesamiento de transacciones online (OLTP) altamente coherentes y transaccionales, y son buenas para el procesamiento analítico online (OLAP).	Las bases de datos NoSQL están diseñadas para varios patrones de acceso a datos que incluyen aplicaciones de baja latencia. Las bases de datos de búsqueda NoSQL están diseñadas para hacer análisis sobre datos semiestructurados.

Modelo de datos	El modelo relacional normaliza los datos en tablas conformadas por filas y columnas. Un esquema define estrictamente las tablas, las filas, las columnas, los índices, las relaciones entre las tablas y otros elementos de las bases de datos. La base de datos impone la integridad referencial en las relaciones entre tablas.	Las bases de datos NoSQL proporcionan una variedad de modelos de datos, como clave-valor, documentos y gráficos, que están optimizados para el rendimiento y la escala.

Propiedades ACID

Las bases de datos relacionales ofrecen propiedades de atomicidad, coherencia, aislamiento y durabilidad (ACID):

- La atomicidad requiere que una transacción se ejecute por completo o no se ejecute en absoluto.
- La coherencia requiere que una vez confirmada una transacción, los datos deban acoplarse al esquema de la base de datos.
- El aislamiento requiere que las transacciones simultáneas se ejecuten por separado.
- La durabilidad requiere la capacidad de recuperarse de un error inesperado del sistema o de un corte de energía y volver al último estado conocido.

Las bases de datos NoSQL a menudo hacen concesiones al flexibilizar algunas de las propiedades ACID de las bases de datos relacionales para un modelo de datos más flexible que puede escalar horizontalmente. Esto hace que las bases de datos NoSQL sean una excelente opción para casos de uso de baja latencia y alto rendimiento que necesitan escalar horizontalmente más allá de las limitaciones de una sola instancia.

Rendimiento	Normalmente, el rendimiento depende del subsistema de disco. Se necesita la optimización de consultas, índices y estructura de tabla para lograr el máximo rendimiento.	El rendimiento es, por lo general, depende del tamaño del clúster de hardware subyacente, la latencia de red y la aplicación que efectúa la llamada.

Escalado	Las bases de datos relacionales generalmente escalan en forma ascendente las capacidades de computación del hardware o la ampliación mediante la adición de réplicas para cargas de trabajo de solo lectura.	Las bases de datos NoSQL normalmente se pueden particionar porque los patrones de acceso son escalables mediante el uso de arquitectura distribuida para aumentar el rendimiento que proporciona un rendimiento constante a una escala casi ilimitada.

API

Solicita almacenar y recuperar datos que están comunicados mediante consultas que se ajustan a un lenguaje de consulta estructurado (SQL). Estas consultas son analizadas y ejecutadas por la base de datos relacional.

Las API basadas en objetos permiten a los desarrolladores almacenar y recuperar fácilmente estructuras de datos. Las claves de partición permiten que las aplicaciones busquen pares de clave-valor, conjuntos de columnas o documentos semiestructurados que contengan atributos y objetos de aplicación serializados.

Bases de Datos no relacionales

- Cassandra:

Base de datos creada por Apache del tipo clave-valor. Dispone de un lenguaje propio para realizar consultas CQL (Cassandra Query Language). Cassandra es una aplicación Java por lo que puede correr en cualquier plataforma que cuente con la JVM. Es multiplataforma.

- Redis

Se trata de una base de datos del tipo clave-valor.

Tiene la ventaja de que sus operaciones son atómicas y persistentes. Sin embargo, Redis no permite realizar consultas, solo se puede insertar y obtener datos, además de las operaciones comunes sobre conjuntos (diferencia, unión e inserción).

- CouchDB

Sistema creado por Apache y escrito en el lenguaje Erlang que funciona en la mayoría de sistemas POSIX (multiplataforma).

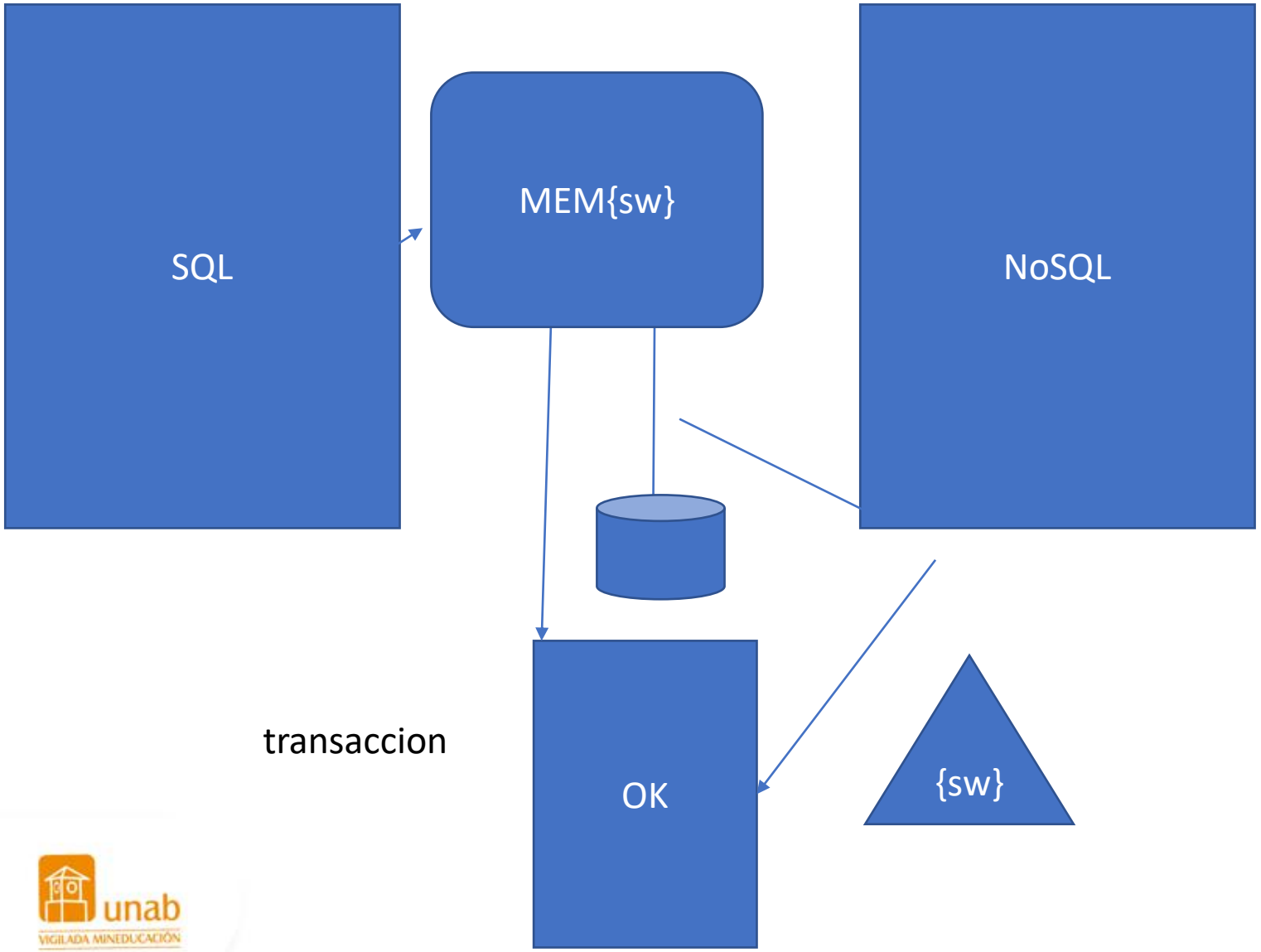
Como características más importantes cabe destacar el uso de RESTful HTTP API como interfaz y JavaScript como principal lenguaje de interacción. Para el almacenamiento de los datos se utilizan archivos JSON.

- MongoDB

Base de datos creada por MongoDB Inc. (anteriormente 10gen) del tipo orientada a documentos, de esquema libre, es decir, que cada entrada puede tener un esquema de datos diferente que nada tenga que ver con el resto de registros almacenados.

Para el almacenamiento de la información, utiliza un sistema propio de documento conocido con el nombre BSON, que es una evolución del formato JSON pero con la peculiaridad de que puede almacenar datos representados de forma binaria.

SQL	MongoDB
Tabla	Conjunto - colleciones
Fila	Documento
Columna	Campo
Clave principal	ObjectId
Índice	Índice



Select c1 activa?
Select saldo cuenta where 1
If(saldo > mover)
Upd c1 – 500
Select c2 activa?
Udp c2 – saldo - 500

Ok!!!

cuenta1	cuenta2
1500	700

Arquitectura de Mongo DB

Partes

MongoDB

Base de datos

Colecciones

Documentos

SQL

Base de datos

Tablas

Registros

Arquitectura de Mongo DB

Base de datos: cada una de ellas tiene un conjunto propio de archivos en el sistema con varias bases de datos existentes en un solo servidor.

Colección: comprende un conjunto de documentos de base de datos. La colección existe dentro de una única base de datos. Dentro de la colección, los documentos tienen campos variados aunque la mayoría de los documentos que se establecen en una misma colección persiguen el mismo objetivo

Documento: el conjunto de pares clave-valor puede ser designado como documento. Los documentos están relacionados con esquemas dinámicos, lo que permite no tener la misma estructura en una sola colección. Del mismo modo, los campos comunes en el documento de una colección, puede tener diferentes tipos de datos

Tipo de datos

Las aplicaciones reales suelen manejar diferentes tipos de datos, todo depende del tipo de aplicación y la lógica del proyecto, por esto MongoDB nos trae otros tipos de datos, veámoslos a continuación.

Object (Objeto)

Este tipo de datos almacena documentos incrustados, a los cuales también se les conoce como documentos anidados. Los documentos incrustados o anidados son aquellos tipos de documentos que contienen un documento dentro de otro documento.

```
Posts{
      User
}
```

```
User{
}
```

Object Id

Cada vez que creamos un nuevo documento en una colección, MongoDB crea de manera automática un id de objeto único para ese documento. Este se crea en el campo **_id** en MongoDB para cada documento.

Los datos que se almacenan en **_id** son de formato hexadecimal y la longitud del id es de 12 bytes que consisten en:

4 bytes para el valor timestamp,

5 bytes para valores aleatorios

3 bytes para contador

También puedes crear tu propio _id, solo debes asegurarte que ese identificador sea único.

```
{
  "_id": "5s5454545f5j5984werwewrf", // Identificador
  "nombre": "Torta de Chocolate",
  "Ingredientes": {
    "azucar": "600 grs",
    "harina": "420 grs",
    "cacao": "1/2 taza",
    "huevos": 4,
    "cucharadasvainilla": 2
  }
}
```

El identificador de un documento al ser único nos permitirá gestionar sus valores de manera independiente sin afectar a los demás documentos.

- Posts{
- _id: 001
- }

- Posts{
- _id: 002
- }

- Posts{
- _id: 20220919735435O1
- }

Undefined (Indefinido)

Este tipo de datos, tal como su nombre lo indica, almacena valores indefinidos. Veamos el siguiente ejemplo en donde un campo tiene un valor indefinido:

```
{
  "_id": "331sgf453265fghfgh5fg56s",
  "nombre": "Gelatina de Naranja",
  "precio": undefined, // Undefined (Indefinido)
  "stock": 39
}
```

Binary Data (Datos Binarios)

Este tipo de datos se utiliza para almacenar datos de 0 y 1, es decir binarios, veamos un ejemplo a continuación:

```
{
  "_id": "5ddf55hj54g55sa454fs49er",
  "nombre": "Jugo de Naranja",
  "precio": 2.5,
  "caducidad": "10100100011" // Dato Binario
  "stock": 27
}
```

Es importante recordar que en el área de la informática los unos y ceros representan muchos valores, es como una manera de guardar datos a bajo nivel o en un formato puro del computador.

Date (Fecha)

Este tipo de datos es también uno de los más usados, pues se considera que la mayoría de datos debe tener una fecha para poder gestionarlos mejor. El tipo de dato fecha es un entero (integer) de 64 bits que representa el número de milisegundos. El tipo de datos BSON generalmente admite la fecha y hora UTC y está firmado. Si el valor de este tipo de datos es negativos, entonces representa fechas anteriores al año 1970.

Existen diferentes métodos para devolver fechas, se puede devolver como un string o como un objeto de fecha usando *Date()*, *new Date()* y *new ISODate()*.

```
{
  "_id": "7jsh8d657d8e12f5s46g5d4s",
  "nombre": "Arroz con Leche",
  "precio": 3.5,
  "stock": 30,
  "fecha_creacion": "Wed Aug 25 2021 09:16:11 GMT-0500 (hora estándar de Perú)" // Fecha, usando Date()
}
```

El tipo de datos Fecha se suele usar para almacenar la fecha de creación , actualización, eliminación y en otro tipo de situaciones en donde se requiera almacenar la Fecha.

Llave Min y Max

Este tipo de dato nos permite comparar valores, por ejemplo con *Min* podemos comparar el elemento más bajo y con *Max* podemos comparar el valor más alto, ambos son tipos de datos internos de un documento.

```
{
  "_id": "7jsh8d657d8e12f5s46g5d4s",
  "nombre": "Jugo de Fresa",
  "precio": 2.8,
  "stock": 25,
  "valorminimo": { "$minKey : 5" }, // Valor Minimo
  "valormaximo": { "$maxKey : 5" }  // Valor Máximo
}
```

- Symbol
- Expresión Regular
- Javascript
- Timestamp

Modelar una base de datos en MONGODB

Blog

Posts{
id,
title,
text,
image,
created_at
fecha
}

Categories{
id,
nombre,
des
}

Comments{
id,
text,
??
}

Image{
id,
??,
url,
des
}

Users{
id,
name,
surname,
password,
email,
status
}

Blog

Categories{
id,
nombre,
des
}

Ciencia software

Posts{
id,
title,
text,
image,
created_at
fecha,
categories{
nombre,
},
,
}
Comments{
text,
Image{
url,
des
}
Users{
name,
surname,
email,
}

Comments{
id,
text,
??
}

Image{
id,
??,
url,
des
}

Users{
id,
name,
surname,
password,
email,
status
}

Blog

Categories{
id,
nombre,
des
}

Ciencia software

Posts{
id,
title,
text,
image,
created_at
fecha,
cat_id,
u_id

}

Comments{
id,
text,
post_id
}

Image{
id,
post_id,
url,
des
}

Users{
id,
name,
surname,
password,
email,
status
}

Bibliografía

https://blog.nubecolectiva.com/tipos-de-datos-en-mongodb-parte-2/

https://www.syntonize.com/la-mejor-base-de-datos-nosql-desde-cero/

https://es.wikipedia.org/wiki/NoSQL

https://ayudaleyprotecciondatos.es/bases-de-datos/no-relacional/

https://blog.nubecolectiva.com/tipos-de-datos-en-mongodb-parte-3-final/

https://blog.nubecolectiva.com/5-guis-para-trabajar-con-mongo-db/

https://www.youtube.com/watch?v=zsdJAuxOV3U

https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=video&cd=&cad=rja&uact=8&ved=2ahUKEwj-4tyt1KD6AhXrSTABHWM0DYMQtwJ6BAgSEAI&url=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3DZdlude8l8w4&usg=AOvVaw2zWTR4s8FQNEbeHC_dyFm8

El futuro digital es de todos

MinTIC

Practicar los temas tratados

MisionTIC2022

unab VIGILADA MINEDUCACIÓN

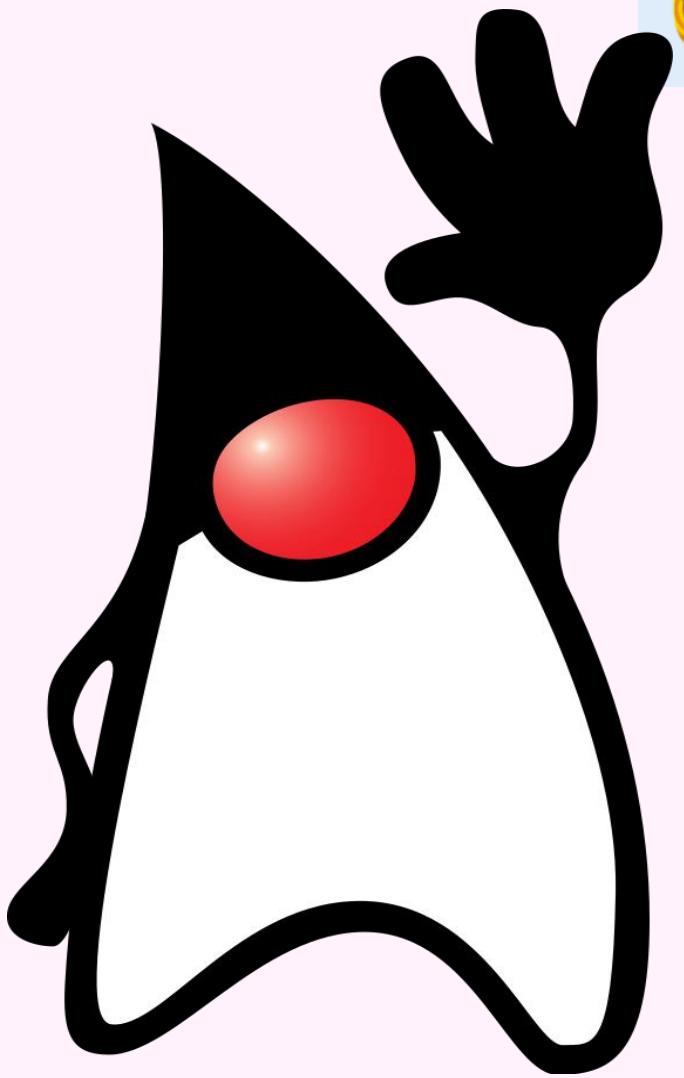
Hechos QUE CONECTAN

Nos vemos en la siguiente sesión

El futuro digital es de todos

MinTIC

Muchas Gracias



www.mintic.gov.co

MisionTIC2022

unab
VIGILADA MINEDUCACIÓN