MONGO DB : Estilo JSON, guarda informacion más versatil para el uso del programador

BD Key-Value

Son grandes tablas de hash persistentes

Es difusa, pues muchas de las aplicaciones de otro tipo usa key-values.

BD de Grafos RDF

Especializadas para guardar relaciones:

En general, almacenan en property graphs

EJ: Neo4J, Virtuoso, Jena, Blazegraph

JSON:

Su nombre viene de JavaScript Object Notation:

Estandar de intercambio de datos semiestructurados / datos en la Web.

-JSON se acopla muy bien a los lenguajes de programacion

Ejemplo:

{"statuses":{ "id": 3425141314, "text": "hola"}}

La base son los pares key-value. Los objetos se definen entre {} y contienen una cantidad arbitraria de pares key-value.

{"nombre": "" , "apellido": ""}

Los arreglos se escriben entre [] y contienen valores.

SQL:

- Esquema de datos

- Lenguajes de consulta independientes del codigo

JSON

- Mas felxible, no hay que respetar necesariamente un esquema

- Mas tipos de datos (como arreglos)

- Human - Readable

Hay intentos de lenguajes de consulta para objetos JSON que usen su estructura de arbol, ej: JSONPath

Importante: JSON esta ahi para los programadores que NO buscan deparar datos del codigo.

Por que necesitamos esquemas para JSON

- JSON Schema: propuesta toma fuerza el 2013 - 2014

- Harta investigacion en el DataLab UC

Fernando Suarez - JSON Schema Hablar con Juan

Ejemplo de Schema:

{

{"type": "object",

"properties":{ "first\_name": {"type": "string"},

"last\_name": {"type": "string"},

"age": {"type": int},

"club": {"type": "object",

"properties":{ "name": {"type": "string"},

"league": {"type": "sting"},

"founded": {"type": int}

}

"required": ["name", "league"]

},

"first\_club": {"type": "string"},

"va\_al\_mundial": {"type": "boolean"},

},

"required": ["first\_name", "last\_name", "age", "club"]

}

Notar que esto es altamente jerarquico

BD Key-Value

Independientemente del esque,a

- Arquitectura almacena informacion por medio de pares

- Cada par tiene una llave (identificador) y valor

Especializadas en documentos: almacenan muchos doc json

- Si quiero libros: un documentos por libro

- Si quiero personas: un doc por persona

Que hacen bien:

- Si quiero un libro o persona en particular

- El cruce de informacion simple

Muy util para desplegar informacion en la web.

Si quiero cruzar inforacion

JSON de alumnos

JSON de ramos

Muy facil

Todos los alumnos que toman un ramo ¿Como lo hago?

Los ramos con mas alumnos

No tan facil

Los ramos con mas alumnos de ingenieria

Si el join es complejo.

Plantea que para una base de datos distribuida es imposible mantener simultaneamente las tres caracteristicas:

-Consistency

-Availability

-Partition Tolerance

BD Documentos vs Teorema CAP

Distintas aplicaciones en una misma base de datos acceden a distintos documentos al mismo tiempo.

En general diseñadas para montar varias instancias que (en teoria) tienen la misma informacion

Propagan updates en forma descoordinada

Proveen "Consistencia Eventual"

La consistencia eventual puede generar problemas.

Si dos aplicaciones intentan acceder al mismo documento en Mongo, estas pueden ser versiones diferentes del documento

MapReduce

Algoritmo eficiente para computacion paralela

Paradigma de programacion

Datos tan grandes que no caben en un computador

Repartidos en muchos servidores

Cada servidor no conoce lo que tiene el resto

No nos podemos dar el lujo de..

Ejemplo MapReduce: ver palabra mas utilizado en un texto T

Map: recibe los datos y genera pares key-value

Reduce: compara y reduce

Map:

Recibe un pedazo de texto

Por cada palabra, emite el par (palabra, numero de ocurrencias)

Reduce:

Cada reduce recibe todos los pares asociados a la misma palabra

Junta todos estos pares y suma las ocurrencias

hola como estas | (hola, 1), (como, 1), (estas, 1) | serever1

--------------- |

hola yo bien | (hola, 1), (yo, 1), (bien, 1) | server2

--------------- |

yo no se yo | (yo, 2), (no, 1), (se, 1) | server3

--------------- |

#junto los datos de cada servidor#

(hola): [1,1]

(yo): [1,2]

Genera un arreglo de todos los terminos que se buscan, cosa de tener todos los valores para ese termino en cada servidor.

Nodos encargados de hacer Map

Reciben parte del documento y lo envian a los reducers

Reducers

Nodos encargados de dasfsaf

No es un descubrimiento nuevo, pero recientemente se ha visto calzar perfect con las necesidades de las grande BD

Es la arquitecutra mas importante en sistemas que reciben grande bases de datos:

Hadoop: La implementacion de Google Map- Reduce.

Como hago un join con MapReduce