UNIVERSIDAD TÉCNICA DEL NORTE

FACULTAD DE INGENIERÍA EN CIENCIAS APLICADAS

CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES

TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS COMPUTACIONALES

TEMA:

"ESTUDIO DE NUEVAS TECNOLOGÍAS DE GESTIÓN DE BASES DE DATOS NOSQL PARA EL DESARROLLO DE APLICACIONES WEB 2.0."

APLICATIVO:

"SISTEMA PROTOTIPO PARA EL CONTROL DE CAMPEONATOS Y JUGADORES DESARROLLADO PARA LA LIGA DEPORTIVA PARROQUIAL SAN PABLO DEL LAGO"

AUTOR: INUCA MORALES MARCO GEOVANY

DIRECTOR: ING. JOSÉ LUIS RODRÍGUEZ

IBARRA – ECUADOR

2015

"ESTUDIO DE NUEVAS TECNOLOGÍAS DE GESTIÓN DE BASES DE DATOS NOSQL PARA EL DESARROLLO DE APLICACIONES WEB 2.0."

AUTOR- Marco Geovany INUCA MORALES

Facultad de Ingeniería en Ciencias Aplicadas, Universidad Técnica del Norte, Av. 17 de Julio 5-21, Ibarra, Imbabura

Carrera de Ingeniería en Sistemas, Universidad Técnica del Norte, Av. 17 de Julio 5-21, Ibarra, Imbabura

geomarkito@gmail.com

Resumen.- El objetivo de la presente investigación se centra dar a conocer los beneficios de las bases de datos NOSQL, con el fin de conocer sus ventajas, desventajas así como también sus funcionalidades, para así poder definir la mejor de las bases de acuerdo a sus necesidades y requerimientos, para el desarrollo de aplicaciones web 2.0.

Es interesante destacar que las NOSQL existen desde hace muchos años (1970), con experimentos en universidades europeas y norteamericanas, sobre sistemas de almacenamiento abiertos. Los almacenamientos de par Clave/Valor, vieron la luz en 1979 con el proyecto DBM, y más tarde BerkeleyDB (1986).

Lo que se quiere demostrar con esta investigación es la capacidad, funcionamiento, escalabilidad, velocidad de consultas en lectura de datos y compatibilidad que tiene cada una de las clasificaciones antes mencionadas con el fin de dar a conocer a estudiantes de la FICA la existencia de esta herramienta con una alternativa de desarrollo para las Aplicaciones Web 2.0 que hoy en día han tomado fuerza.

Palabras Claves

NOSQL,Clave/Valor, DBM, BerkeleyDB, capacidad, funcionamiento, escalabilidad, web 2.0.

Abstract.- The aim of this research is to publicize the benefits of NoSQL databases, in order to know its advantages, disadvantages as well as its functionality, in order to define the best of the foundation according to their needs and requirements for the development of Web 2.0 applications.

Interestingly NoSQL there for many years (1970), with experiments in European and American universities, on open storage systems. Storages pair key / value, saw the light in 1979 with the DBM project and later BerkeleyDB (1986).

What we want to demonstrate with this research is the ability, performance, scalability, speed reading of data queries and support that each one of the classifications listed above in order to inform students of the existence of this FICA tool with an alternative development for Web 2.0 applications that have gained strength today.

Keywords

NoSQL,key / value, DBM, BerkeleyDB, capacity, performance, scalability, web 2.0.

1. Introducción

En la actualidad se vive en un mundo en el que la tecnología se encuentra muy desarrollada, esto ha cambiado la forma de pensamiento y de vida de muchas personas, gran cantidad de Empresas e Instituciones se han visto en la necesidad de introducirse en el mundo de la informática para brindar un mejor servicio a la comunidad, en vista a la creciente necesidad de representar información, manejarla, explotarla y compartirla.

Ante esto el inconveniente que se presenta es que la información que se manipula en Instituciones, Empresas, Organizaciones que crecen a pasos agigantados y las bases de datos Normales Relacionales, no cumplen con los requerimientos para satisfacer las necesidades de estos, esto se debe a la gran capacidad de acceso a la información.

Es por esta razón, que se han visto en la necesidad de investigar y analizar la tecnología de Gestión de Bases de Datos NoSQL en sus clasificaciones más importantes las cuales permiten a los usuarios trabajar con grandes volúmenes de datos con mayor rapidez.

La taxonomía de las bases de datos NoSQL entre las principales por su implementación que son:

- ✓ Bases de datos key-value.
- ✓ Bases de datos orientadas a columnas.
- ✓ Bases de datos orientadas a documentos.
- ✓ Bases de datos orientadas a grafos.

Las características principales de una base de datos NoSQL responden a:

a) Modelos de datos variables y flexibles: Siempre hay un modelo, una estructura, pero la rigidez del modelo relacional desaparece
b) Escalabilidad sencilla, pueden responder a necesidades pequeñas y a altos volúmenes de trabajo
c) Alta velocidad de respuesta a peticiones. Se tiene un menor tiempo de respuestas al realizar las consultas a las bases NOSQL
d) Diferentes lenguajes de consulta (no SQL). Las diferentes bases de datos NOSQL tienen su propio lenguaje de consulta.

2. Materiales y Métodos

En los últimos años, principalmente por los incrementos de las redes sociales como Facebook y los grandes sistemas distribuidos como Google App Engine, existen varios problemas con los RDBMS actuales que pueden suponer una seria limitante para la construcción de aplicaciones

Si bien la mayoría de las aplicaciones se centran mucho en las características del modelo relacional, para otras aplicaciones (las cuales se alejen un poco del esquema típico) se empiecen a notar distintos problemas inherentes al modelo. Estos problemas son en gran medida el motivo por el que surgió el NOSQL.

Los problemas que se presenta en la actualidad son:

Leer datos es costoso.- En el modelo relacional los datos se representan mediante conjuntos (tablas) relacionados entre sí. Realizar una consulta al modelo relacional (como todos sabemos), implica juntar grandes conjuntos de datos con operaciones algebraicas (como el producto cartesiano), y luego filtrar todo el conjunto resultante lo cual lleva una gran complejidad computacional.

Transaccionalidad innecesaria.- El objetivo de las transacciones es asegurar la integridad de los datos, aunque con este esquema se dejen de lado otros aspectos como la performance.

Escalabilidad.- El gran problema de las bases relacionales es la escalabilidad. Estas fueron pensadas para correr en un solo servidor con mucha potencia, como mucho tener replicaciones y balanceo de carga.

Representación del modelo en una RDBMS.- Si bien es posible representar la mayoría de modelos usando el modelo relacional, no siempre resulta la mejor opción.

BASES DE DATOS RELACIONALES

Sistema de gestión de bases de datos Relacional o RDBMS, el cual es capaz de producir, manipular y gestionar bases de datos de tipo relacional, los sistemas de bases de datos relacionales son aquellos que almacenan y administran de manera lógica los datos en forma de tablas. Una tabla es, a su vez un método por el cual podemos representar los datos en filas y columnas.

Un RDBMS proporcionar a los usuarios la capacidad de almacenar los datos en la bases de datos, acceder a ellos y actualizarlos. Esta es la función fundamental de un RDBMS y por supuesto, el RDBMS debe ocultar al usuario la estructura física interna (la organización de los archivos y las estructuras de almacenamiento)

Figura 1.- Esquema – Representación Modelo Relacional

¿QUÉ ES NOSQL?

Las bases de datos NoSQL son sistemas de almacenamiento de información que no cumplen con el esquema entidad-relación al que todos nos acostumbramos, estas bases de datos son no relacionales y no proporcionan garantías ACID, el

término hace referencia a un conjunto de características requeridas para que una serie de instrucciones puedan ser consideradas como una transacción, tampoco poseen un esquema fijo de tablas ni sentencias JOIN, con la cual permite la combinación de registros de dos o más tablas en una base de datos relacional,
En realidad, un sistema de este tipo no es siquiera una base de datos entendida como tal, sino un sistema de almacenamiento distribuido para gestionar datos dotados de una cierta estructura, estructura que además puede ser enormemente flexible. (Dans, 2011).

CARACTERÍSTICAS NOSQL

Las bases de datos tienen características únicas y una reputación sólida para la integridad de datos, pero todo esto puede resultar demasiado para nuestras necesidades.

Como menciona (Wiesse, 2011) en un artículo publicado sobre las características principales de una base de datos NOSQL las cuales responden a:

- ✓ Modelos de datos variables y flexibles: siempre hay un modelo, una estructura, pero la rigidez del modelo relacional desaparece…
- ✓ Escalabilidad sencilla, pueden responder a necesidades pequeñas y a altos volúmenes de trabajo.
- ✓ Alta velocidad de respuesta a peticiones. El tiempo a un es un muy alto costo
- ✓ Diferentes lenguajes de consulta (no SQL). Lenguajes muy fáciles de entender y comprender.

DIFERENCIAS ENTRE BASES DE DATOS NOSQL

En los últimos años una gran variedad de bases de datos NoSQL se ha desarrollado principalmente por los profesionales y empresas de la web para adaptarse a sus requisitos específicos con respecto al rendimiento escalabilidad, mantenimiento y conjunto de características.

Ausencia de esquema en los registros de datos

Esta primera característica significa que los datos no tienen una definición de atributos fija, es decir: Cada registro puede contener una información con diferente forma cada vez, pudiendo así almacenar sólo los atributos que interesen en cada uno de ellos, facilitando el polimorfismo de datos bajo una misma colección de información.

Escalabilidad Horizontal

La escalabilidad es la propiedad deseable de un sistema, una red o un proceso, que indica su habilidad para reaccionar y adaptarse sin perder calidad, o bien manejar el crecimiento continuo de trabajo de manera fluida, o bien para estar preparado para hacerse más grande sin perder calidad en los servicios ofrecidos.

(López, 2011), en uno de sus comentarios dice que.

Estas bases de datos tienen la ventaja de su rapidez y escalabilidad horizontal y los inconvenientes de que no garantizan las transacciones. Dependiendo del tipo de proyecto, puede que se adapte mejor una base de datos de este tipo o una relacional, todo depende de las necesidades del proyecto.

ARQUITECTURA DE LOS SISTEMAS DE GESTIÓN DE BASES DE DATOS NOSQL

Bastantes sistemas NoSQL emplean una arquitectura distribuida, manteniendo los datos de forma redundante en varios servidores, usando frecuentemente una tabla hash distribuida. De esta forma, el sistema puede realmente escalar añadiendo más servidores, y el fallo en un servidor puede ser tolerado.
Hoy en día se pueden encontrar diferentes sistemas de bases de datos distribuidos que potencian diferentes atributos. De hecho, existe un teorema sobre sistemas distribuidos, el teorema CAP, que los agrupa de forma clara. Según este teorema, un sistema distribuido no puede satisfacer de manera completa los siguientes atributos al mismo tiempo: Consistencia Fuerte, Alta Disponibilidad y Tolerancia a Particionamiento. Teniendo en cuenta que una base de datos distribuida puede cumplir con dos de estos atributos de manera satisfactoria,

se pueden establecer grupos de estos sistemas de acuerdo a estas características. En la siguiente agrupación se pueden observar algunos ejemplos:

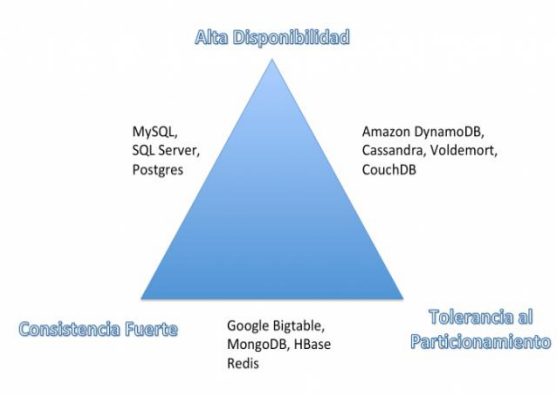

FIGURA 2: Arquitectura- Teorema CAP (Browne, 2011)

La consistencia es decir, si y cómo un sistema está en un estado coherente después de la ejecución de una operación. Un sistema distribuido para ser considerado generalmente coherente después de una operación de actualización de algún escritor donde todos los lectores puedan ver sus actualizaciones de algún origen de datos compartido.

La alta disponibilidad que significa que un sistema ha sido diseñado y aplicado de una manera que le permite continuar la operación (es decir, lo que permite leer y escribir operaciones) si nodos en un accidente de clúster o algunas piezas de hardware o de software se han reducido debido a mejoras.

Tolerancia partición se entiende como la capacidad del sistema para continuar la operación en presencia de particiones de red. Esto ocurre si dos o más "islas" de nodos de la red que surgen (temporalmente o permanentemente) no pueden conectarse entre sí. Algunas personas también entienden la tolerancia partición comola capacidad de un sistema para hacer frente a la adición dinámica y la eliminación de nodos.

En su el articulo Brewer señala rasgos y ejemplos de las tres opciones diferentes que se puede hacer de acuerdo a su teorema CAP.

TABLA 1: Teorema CAP- Opciones Rasgos y Ejemplos

Opción	Rasgos	Ejemplos
Consistencia + Disponibilidad **(Pierde Particiones)**	2 fase de confirmación de protocolos de validación	Bases de datos de un solo sitio las bases de datos del clúster LDAP Sistema de archivos XFS
Consistencia + Partición tolerancia **(Pierde Disponibilidad)**	bloqueo pesimista Hacer particiones minoritarios disponible	bases de datos distribuidas bloqueo Distribuido protocolos Mayoría
Disponibilidad tolerancia + Partición **(Pierde consistencia)**	Vencimientos / arrendamientos resolución de conflictos optimista	Coda Web Cachinge [sic!] DNS

Las bases de datos NoSQL parten de la base en la que las tablas no existen como tal, sino que la información se almacena de forma distinta, generalmente como llave-valor, como una tabla en la que las columnas son dinámicas, pueden cambiar sin perder la agrupación de la información, así es que puedo tener 'personas' con más atributos que otras, puedo cambiar la estructura de mi información dinámicamente sin tener que rediseñar todo de nuevo.

TAXONOMÍA DE LOS MODELOS DE DATOS

Ha habido varios enfoques para clasificar las bases de datos NoSQL, cada uno con diferentes categorías y subcategorías. Debido a la variedad de enfoques y superposiciones es difícil de conseguir y mantener una visión general de las bases de datos no relacionales. Sin embargo, la clasificación básica de que la mayoría estaría de acuerdo en se basa en

el modelo de datos. A algunos de estos y sus prototipos son:

- ✓ Columna: HBase, Accumulo, Cassandra
- ✓ Documento: MarkLogic, MongoDB, Couchbase
- ✓ Valor-clave: Dynamo, Riak, Redis, MemcacheDB, Project Voldemort
- ✓ Gráfico: Neo4J, Allegro, Virtuoso

(Yen, 2009) En un artículo publicado, escribe sobre las diferentes categorías de las bases de datos NoSQL como se muestra en la siguiente tabla.

TABLA 2: Clasificación.- Taxonomía de las bases de datos según (Yen, 2009)

Categorías	Matching Database
KV Cache	Memcached, Repcached, Coherence, Infinispan, eXtreme Scale, Cache, Velocity, Terracotta, Gigaspaces XAP
KV Store	Keyspace, Flare, SchemaFree, RAMCloud
KV Store - Eventually consistent	Dynamo, Voldemort, Dynomite, SubRecord, MotionDb, DovetailDB
Data-structures server	Redis
KV Store - Ordered	TokyoTyrant, Lightcloud, NMDB, Luxio, MemcacheDB, Actord
Tuple Store	Gigaspaces, Coord, Apache River
Object Database	ZopeDB, DB4O, Shoal, Perst
Document Store	MongoDB, Jackrabbit, XML-Databases, ThruDB, CloudKit, Persevere, Riak Basho, Scalaris
Wide Columnar Store	BigTable, HBase, Cassandra, Hypertable, KAI, OpenNeptune, Qbase, KDI

Una similar taxonomía sobre las bases de datos NoSQL hace (Strauch, 2012) quien en su documentos menciona que Ken North en uno de sus artículos de "databases in the cloud" que existen distribuciones disponibles en informática en entornos en Cloud computing, en un pequeño resumen en la tabla 2

TABLA 3: Clasificación- La categorización por (North, 2013)

Categorías	Bases de datos
Distributed Hash Table, Key-Value Data Stores	memcached MemcacheDB Project Voldemort Scalaris Tokyo Cabinet
Entity-Attribute-Value Datastores	Amazon SimpleDB Google AppEngine datastore Microsoft SQL Data Services Google Bigtable Hadoop HyperTable HBase
Amazon Platform	Amazon SimpleDB

Document Stores, Column Stores	Sybase IQ Vertica Analytic Database Apache CouchDB

Una clasificación similar nos hace mención (Cattell, 2013) donde las diferentes bases de datos NoSQL se caracterizan primariamente por su modelo de datos

TABLA 4: Clasificación- Categorización según (Cattell, 2013)

Categorías	Bases de Datos Por Modelo
Key-value Stores Redis	Scalaris, Tokyo Tyrant, Voldemort, Riak
Document Stores SimpleDB	CouchDB, MongoDB, Terrastore
Extensible Record Stores Bigtable	HBase, HyperTable, Cassandra

CLASIFICACIÓN BASADA EN LAS NECESIDADES DEL CLIENTE

Se presenta un enfoque diferente para clasificar las bases de datos NoSQL, subsumir las bases de datos según las necesidades del cliente:

Primera Característica.- Es clase de bases de datos proporciona un gran número de características de alto nivel que hacen que el trabajo de el programador sea fácil. En el lado negativo son difíciles de escalar.

Ejemplos: Oracle, Microsoft SQL Server, IBM DB2, MySQL, PostgreSQL, Amazon RDS.

Primera Escala.- Este tipo de bases de datos tiene que escalar desde el principio. En el lado negativo, carecen de características particulares y ponen de nuevo la responsabilidad al programado.

Ejemplos: Project Voldemort, Ringo, Amazon SimpleDB, Kai, Dynomite, Yahoo PNUTS, ThruDB, Hypertable, CouchDB, Cassandra, MemcacheDB.

Simple estructura de almacenamiento.- estas clases de bases de datos subsumen a clave /valor (Key/value) en un énfasis de almacenar y recupera conjuntos de estructura arbitraria. La desventaja es que no tienen las características o escalabilidad de otros sistemas.

Ejemplos: file systems, Cassandra, BerkelyDB, Amazon SimpleDB.

Propósito de almacenamiento optimizado.- Estas son las bases de datos que han sido diseñados y construidos para ser buenos en cosas como almacenamiento de datos o procesamiento de flujo.

Ejemplo: StreamBase, Vertica, VoltDB, Aster Data, Netezza, Greenplum.

CLASIFICACIÓN EN LA PRESENTE TESIS

Bases de Datos Clave/Valor (Key / value): Llave-valor es la forma más típica, como un hashmap donde cada elemento está identificado por una llave única, lo que permite la recuperación de la información de manera muy rápida. muchas de ellas están basadas en la publicación de google acerca de su bigtable y de Amazon. Dentro de estas bases de datos podemos encontrar a bigtable de google, simpledb de Amazon, Cassandra, HBase (Hadoop), Riak, Voldemort, tokio-cabbinet y MemcacheDB entre otras.

Bases de Datos Documentales (document based): Estas almacenan la información como un documento (generalmente con una estructura simple como JSON o bson) y con una llave única. podemos encontrar a MongoDB y CouchDB entre las más importantes de este tipo.

Bases de Datos en Grafos (graph db): Hay otras bases de datos que almacenan la información como grafos donde las relaciones entre los nodos son lo más importante. Son muy útiles para representar información de redes sociales. Encontramos a Neo4J entre otras.

Bases de datos orientadas a columnas: Esta es la cuarta clase de bases de datos que se realizara la investigación estos sistemas de bases de datos que tienen la característica de almacenar los datos en

forma de columna. Tienen una ventaja principal de este tipo de sistema, es que permite el acceso a grandes volúmenes de datos de forma rápida porque se puede acceder como una unidad a los datos de un atributo particular en una tabla. Un SMBD orientada a columnas es un sistema de gestión de bases de datos que almacena su contenido por columnas (atributos) y no por filas (registros) como lo hacen los SMBD relacionales.

Base De Datos Mongodb

Es una más de las bases de datos NOSQL que también ofrece almacenamiento de documentos. No brinda soporte REST, pero se basa en un almacenamiento JSON. También tiene Map/Reduce, y ofrece un poderoso conjunto de APIs de consulta muy parecido a SQL.

Este es un punto clave de MongoDB, que resulta muy cómodo a personas que tienen conocimientos de SQL. MongoDB también ofrece replicación maestro-esclavo, y están trabajando en escalabilidad basada en autosharding y en tolerancia a fallos.

MongoDB es una base de datos documental sin esquema escrito en C + + y se desarrolló en un proyecto de código abierto que es impulsado principalmente por la empresa 10gen Inc, que también ofrece servicios profesionales en torno a MongoDB. (Strauch, 2012)

De acuerdo a sus desarrolladores la meta principal de MongoDB es cerrar la brecha entre los almacenamientos key/value rápidos y altamente escalables y sistemas de gestión de múltiples funciones tradicionales RDBMS bases de datos relacionales.

Según el artículo publicado por la organización de MongoDB (MongoDB Inc, 2014) Se trata de una base de datos ágil que permite a los esquemas cambiar más rápidamente para que así las aplicaciones evolucionen, sin dejar de ofrecer a los desarrolladores la funcionalidad que esperan de las bases de datos tradicionales, como los índices secundarios, con un lenguaje de consulta de plena y estricta consistencia.

Estructura De Mongodb

Conjunto de colecciones dinámicas, solo existen si hay almacenadas en ellas, colecciones o documentos.

Colecciones: lo que sería una tabla para las bases de datos relacionales. Están compuestas de documentos, estos no se someten a esquemas fijos.

Documentos: son un conjunto de líneas en formato JSON.

Cada documento se almacena en formato BSON. BSON es una representación codificada en binario de un formato de documento de tipo JSON donde la estructura está cerca de un conjunto anidado de clave /valor pares. BSON es un súper-conjunto de JSON y soporta tipos adicionales como expresiones regulares, datos binarios, y la fecha. Cada documento tiene un identificador único, que MongoDB puede generar, si no se especifica explícitamente al insertar los datos en una colección, este genera un id de objeto automáticamente como se muestra en la figura

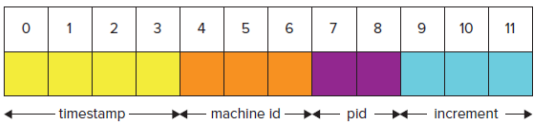

FIGURA 3: Demostración – Generar Id en Colecciones.

Se pueden crear índices para algunos atributos de los documentos, de modo que MongoDB mantendrá una estructura interna eficiente para el acceso a la información por los contenidos de estos atributos.

BSON es un formato de intercambio de datos usado principalmente para su almacenamiento y transferencia en la base de datos MongoDB. Es una representación binaria de estructuras de datos y mapas. El nombre BSON está basado en el término JSON y significa Binary JSON (JSON Binario).

Comparado a JSON, BSON está diseñado para tener un almacenamiento y velocidad más eficiente. Los elementos largos contienen el campo “tamaño” para facilitar su escaneo, lo que provoca que en

algunos casos BSON use más espacio en memoria que JSON. (WIKIPEDIA, 2013)

Esta base de datos se construye para la escalabilidad, el rendimiento y la alta disponibilidad, la ampliación de las implementaciones de servidores individuales hasta grandes y complejas arquitecturas multi-sitio. Mediante el aprovechamiento in-memory computing, MongoDB proporciona un alto rendimiento tanto para lecturas y escrituras. Replicación nativa de MongoDB y failover automatizado permiten la confiabilidad de nivel empresarial y flexibilidad operativa. MongoDB es de código binario está disponible para los sistemas operativos Windows, Linux, OS X y Solaris, esta base de datos es adecuada para la el desarrollo de aplicaciones en los siguientes ámbitos:

Almacenamiento y registro de eventos

Para sistemas de manejo de documentos y contenido

Comercio Electrónico

Juegos

Problemas de alto volumen de lecturas

Aplicaciones móviles

Almacén de datos operacional de una página Web

Manejo de contenido

Almacenamiento de comentarios

Votaciones

Registro de usuarios

Perfiles de usuarios

Sesiones de datos

Proyectos que utilizan metodologías de desarrollo iterativo o ágiles

Manejo de estadísticas en tiempo real

MongoDB es utilizado para uno o varios de estos casos por varias empresas

Características Principales

Lo siguiente es una breve descripción de las características principales de MongoDB:

Consultas Ad hoc

MongoDB soporta la búsqueda por campos, consultas de rangos y expresiones regulares. Las consultas pueden devolver un campo específico del documento pero también puede ser una función JavaScript definida por el usuario.

Indexación

Cualquier campo en un documento de MongoDB puede ser indexado, al igual que es posible hacer índices secundarios. El concepto de índices en MongoDB es similar a los encontrados en base de datos relacionales.

Replicación

MongoDB soporta el tipo de replicación maestro-esclavo. El maestro puede ejecutar comandos de lectura y escritura. El esclavo puede copiar los datos del maestro y sólo se puede usar para lectura o para copia de seguridad, pero no se pueden realizar escrituras. El esclavo tiene la habilidad de poder elegir un nuevo maestro en caso de que se caiga el servicio con el maestro actual.

Balanceo de carga

MongoDB se puede escalar de forma horizontal usando el concepto de "shard". El desarrollador elige una llave shard, la cual determina cómo serán distribuidos los datos en una colección.

Almacenamiento de archivos

MongoDB puede ser utilizado con un sistema de archivos, tomando la ventaja de la capacidad que tiene MongoDB para el balanceo de carga y la replicación de datos utilizando múltiples servidores para el almacenamiento de archivos múltiple

MongoDB tiene la capacidad de realizar consultas utilizando JavaScript, haciendo que estas sean enviadas directamente a la base de datos para ser ejecutadas.

Ventajas Y Desventajas

Ventajas	**Desventajas**

No requiere operaciones JOIN Escala horizontalmente Dispone de MapReduce	Se le considera que es no es producto maduro. No escala con herramientas de Business Intelligence.
Uso de memoria en vez de disco Alta frecuencia de escritura y lectura Cambios de esquema de datos frecuente	Requiere conocimientos de programación
Utilizan estructura de datos sencilla, tipo clave/valor AutoSharding	La migración de datos es casi imposible de un proveedor a otro. Restringe el tamaño de datos a un máximo de 2GB en sistemasde 32 bytes

2.1. Arquitectura del Sistema

Una vez obtenida las necesidades de la aplicación sus funciones y requerimientos, se procedea diseñar la arquitectura del sistema y la arquitecturafuncional. Se especificará cada uno de los módulos con losque contará el sistema y su respectiva funcionalidad.

El Modelo, Vista y Controlador se basa en laarquitectura de MVC, que es el sistema base en el que se aplica para el desarrollo de la aplicacion, por lo tanto todas las características de la arquitectura serán similares a las de SugarCRM.

Modelo: El modelo es aquello que soporta los datos (o capa de negocio). La representación específica de la información con la cual el sistema opera.

En este caso el modelo está formado por la base de datos MongoDB.

Vista: La vista presenta un modelo en el cual se pueda interactuar.

Dentro de nuestra aplicación nuestra vista está conformada por los archivos de CSS, JavaScript, PHP y HTML.

Controlador: El controlador responde a eventos, usualmente acciones del usuario, e invoca peticiones al modelo y, probablemente, a la vista. En el caso corresponde al archivo liga-campeonato.

Este es el archivo principal, donde comienza la ejecución de la aplicación. Levanta el servidor de xampp y php, captura los GET y POST que llegan del cliente redirigiendo esas acciones a liga-campeonato. Configura el puerto y el host al que se conectará el cliente, pudiendo estar esta configuración implícita en el archivo o indicándolo como opciones del comando al ser ejecutado.

3. Resultados

Una vez finalizada la investigación se logra obtenerun producto de software de alta calidad, que cumple contodas las necesidades expuestas al inicio del desarrollo.

Basándose en la ventajas, desventajas y características de la base de datos NoSQL.

La implementación de la aplicación para la liga parroquial de san pablo del lago se muestran en la siguientes ventanas.

FIGURA 4: Pantalla principal o portada del Aplicación

SECCIÓN LIGA

En esta sección de liga se despliega un menú con dos opciones las cuales permite el ingreso de los datos a los documentos Campeona y Equipos de la liga.

Para poder crear campeonatos debemos dar clic en el enlace Liga\campeonato, de la misma forma para crear los equipos hacemos clic en Liga\Equipos

FIGURA 5: Pantalla del Sistema para Ingreso de datos Campeonato Fuente: [Propia]

SECCIÓN JUGADORES

En esta sección podemos ingresar a realizar el registro de información de jugadores, tarjetas y pases para lo cual se deberá clic en el acceso correspondiente. En cada formulario nos permitirá realizar la creación de un nuevo documento, modificar, eliminar y guardar.

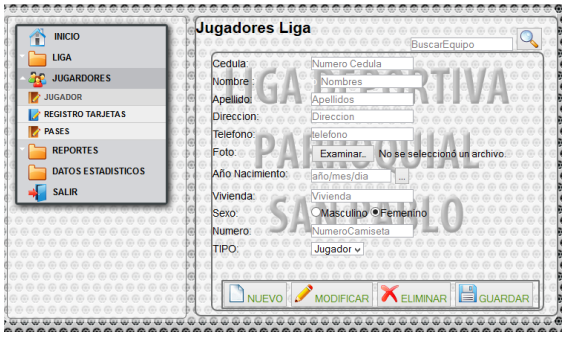

FIGURA 6: Pantalla del Sistema para Ingreso de datos jugadoresFuente: [Propia]

SECCIÓN REPORTES

En esta sección podremos visualizar e imprimir los reportes de los jugadores, carnets, nominas, reportes de tarjetas, Documentos de Pases

FIGURA 7: Pantalla del Sistema para visualizar los reportes Fuente: [Propia]

FIGURA 8: Pantalla del Sistema para imprimir los reportes. Fuente: [Propia.]

4. Conclusiones

Al final del documento de tesis sobre el "Estudio De Nuevas Tecnologías De Gestión De Bases De Datos NOSQL, Para El Desarrollo De Aplicaciones Web 2.0" puedo decir que se ha cumplido los objetivos planteados al inicio del proyecto de la mejor manera.

En la actualidad los desarrolladores de aplicaciones web 2.0 se basan en la confiabilidad, costo, desempeño, rapidez y flexibilidad que las bases de datos NoSQL, que brindan a las grandes empresas como son Facebook, Amazon, twitter, google, por sus beneficios, funciones y alta velocidad.

Las bases de datos NoSQL resulta ser una alternativa potente para el desarrollo de las aplicaciones web que requieren el manejo de gran cantidad de datos, así como también aplicaciones que requieren menor coste en el tiempo de respuesta, estas se adaptan según las necesidades de los programadores ya que son de fácil aprendizaje.

Teniendo en cuenta las fortalezas y debilidades de las bases de datos tanto relacionales como NoSQL

En este documento se da a conocer 4 de las taxonomías principales de bases de datos NoSQL, que al momento están siendo desarrolladas y

utilizadas, estas son las bases de datos Clave/valor, las bases de datos Orientadas a documentos, Bases de datos Orientadas a Grafos, bases de datos Orientadas a Columnas, las cuales presentan diferentes distribuciones como BerkeleyDB y Amazon dynamo bases de datos Clave/valor.

Las bases de datos MongoDB y CouchDB que son bases de datos Documentales, Neo4j y FlockDB que son bases de tipo grafo, las bases de datos de tipo columna como google,s Bigtable, Cassandra, todas estas bases de datos mantienen relaciones en común las cuales les permite ser únicas para cada aplicación.

Por ejemplo BerkeleyDB que tiene un método de table Hash que permite almacenar la información en esquemas organizados por Valores, o MongoDB que almacena la información en esquemas pues su almacenamiento utiliza colecciones , sin una estructura a seguir el cual permite la manipulación de información de una manera rápida. Neo4j una base de datos tipo Grafo la cual permite guardar información en nodos y aristas, estas bases de datos son muy utilizadas por las redes sociales como Facebook, twitter, entre otras.

Las bases de datos tipo columnas que son bases de datos que permite tener una escalabilidad horizontal lo cual permite agregar clúster en cualquier momento.

Teniendo en cuenta la clasificación de las bases de datos que existen hasta el momento hemos tomado la decisión de utilizar una de ellas la cual es relevante para el desarrollo de nuestra aplicación, por sus características e identificación con la aplicación desarrollada para la liga deportiva parroquial de san pablo del lago se ha tomado en cuenta MongoDB una base de datos de tipo documento.

La cual permitirá a la Liga guardar información de sus jugadores y documentos esenciales para la participación en los campeonatos inaugurados en los siguientes años.

Indudablemente el estudio de las nuevas herramientas en este caso las bases de datos NoSQL, significan la exploración y análisis en beneficios una gran alternativa para el desarrollo de aplicaciones web 2.0 en las siguientes generaciones.

Agradecimiento

A todos mis docentes a lo largo de la carrera quienes inculcaron valores y enseñanzas, a mis amigos y compañeros en la vida.

5. Recomendaciones

Las bases de datos NoSQL fueron diseñadas para realizar tareas específicas, en cuanto al almacenamiento de información, como MongoDB que es un motor de búsqueda en las nubes de gran almacenamiento, Casandra que es un motor de búsqueda de Facebook, que son gestores de bases de datos para un alto coste de información.

En el diseño de un modelo de datos, tenga en cuenta cómo las aplicaciones utilizarán la base de datos. Por ejemplo, si la aplicación sólo utiliza documentos recientemente insertados, considere el uso capsulado de colecciones. O si sus necesidades de aplicación se leen principalmente operaciones a una colección, agregar índices de apoyo a consultas comunes puede mejorar el rendimiento.

NoSQL no es apropiado para el desarrollo de todas las aplicaciones y proyectos. Los casos típicos de uso son con aplicaciones de Internet con un gran tráfico y muchos datos (hablamos de gigabytes o petabytes). Pocas aplicaciones tienen este tipo de requisitos, por otra parte, las aplicaciones más pequeñas se benefician de los menores requisitos de las base de datos NoSQL (menor infraestructura, fácil replicación, creación de bases de datos automáticas, etc.) pero debe evaluar si se es capaz de sostener la elección NoSQL.

CouchDB Y MongoDb son bases de datos NoSQL que presentan una flexibilidad al momento de modelar los datos, permiten un desarrollo rápido de las aplicaciones y son súper amigables en el código para la creación de bases de datos y en su lenguaje de consulta que es JSON.

Que la Universidad Técnica del Norte fomente acciones que ayuden en la identificación y

desarrollo de capacidades – personas, procesos y tecnologías – que puedan apoyar el desarrollo de nuevos productos y servicios utilizando soluciones de bases de datos NoSQL

Se recomienda el estudio de las diferentes clasificaciones de bases de datos NoSQL como Bases de datos Clave/Valor, Bases de datos orientadas a grafos y orientadas a columnas como temas de análisis e investigación para el desarrollo de aplicaciones Web 2.0.

Referencias Bibliográficas

10gen, Inc. (2010). *mongoDB. 2010*. Obtenido de http://www.mongodb.org

Apache Software Foundation. (22 de 10 de 2013). *Apache HBASE*. Obtenido de Aticulo : http://hbase.apache.org/

Avinash, L. (07 de 01 de 2014). *Cassandra A structured storage system on a P2P Network.*Obtenido de Blog post: https://www.facebook.com/note.php?note_id=24413138919

Browne, J. (11 de 01 de 2011). *Brewer's CAP Theorem.* Obtenido de :julianbrowne: http://www.julianbrowne.com/article/viewer/brewers-cap-theorem

Cattell, R. (12 de 11 de 2013). *Scalable SQL and NoSQL Data Stores.*Obtenido de http://cattell.net: http://cattell.net/datastores/Datastores.pdf

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., . . . Gruber, R. E. (05 de 01 de 2014). *Bigtable:*. Obtenido de articulo: http://static.googleusercontent.com/media/research.google.com/es//archive/bigtable-osdi06.pdf

Czura, M. (28 de 09 de 2010). *CouchDB In The Wild. 2008–2010.*Obtenido de Wiki article, version 97: http://wiki.apache.org/couchdb/CouchDB_in_the_wild

Dans, E. (28 de 11 de 2011). *Entender el futuro: la evolución de las bases de datos*. Obtenido de www.enriquedans.com: http://www.enriquedans.com/2011/11/entender-el-futuro-la-evolucion-de-las-bases-de-datos.html

David , K., Lehman, E., Leighton, T., Panigrahy, R., Levine, M., & Daniel . (27 de 12 de 1997). *Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web.*Obtenido de Articulo: http://thor.cs.ucsb.edu/~ravenben/papers/coreos/kll+97.pdf

DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., . . . Vogels, W. (1 de 09 de 2007). *Dynamo: Amazon's Highly Available Key-value Store.* Obtenido de articulo: http://www.read.seas.harvard.edu/~kohler/class/cs239-w08/decandia07dynamo.pdf

Edgar F. Codd. (24 de 12 de 2013). *Reglas de Codd.* Obtenido de Documento: http://www.galeon.com/nevifi/Archivos/Codd.pdf

Ellis, J. (14 de 09 de 2013). *NoSQL Ecosytem.* Obtenido de Blog: http://www.rackspacecloud.com/blog/2009/11/09/nosql-ecosystem/

Fundación Wikimedia, Inc. (5 de 11 de 2013). *BERKELEYDB*. Obtenido de Wikipedia.org: http://es.wikipedia.org/wiki/Berkeley_db

Fundación Wikimedia, Inc. (22 de 10 de 2013). *ORM Mapeo Objeto Relacional*. Obtenido de Wikipedia.org: http://es.wikipedia.org/wiki/Mapeo_objeto-relacional

Fundación Wikimedia, Inc. (8 de Diciembre de 2013). *SENTENCIA JOIN*. Obtenido de Wikipedia.org: http://es.wikipedia.org/wiki/Join

Garcete, A. (05 de 01 de 2014). *Base de Datos Orientado a Columnas*. Obtenido de Tesis: http://www.jeuazarru.com/docs/dbco.pdf

Gilbert, S., & Lynch, N. (07 de 12 de 2013). *Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services*. Obtenido de http://lpd.epfl.ch/sgilbert/pubs/BrewersConjecture-SigAct.pdf

Gonzales, M. P. (11 de 12 de 2013). *¿Qué son las Bases de Datos NoSQL?* Obtenido de BLOGSPOT: http://manuelpereiragonzalez.blogspot.com/2011/05/que-son-las-bases-de-datos-nosql.html

Hoff, T. (28 de 10 de 2009). *MySQL or Memcached or Tokyo Tyrant?*Obtenido de Blog post: http://highscalability.com/blog/2009/10/28/and-the-winner-is-mysql-or-memcached-ortokyo-

Ippolito, B. (28 de 03 de 2009). *Drop ACID and think about Data*. Obtenido de Talk at Pycon: http://blip.tv/file/1949416/

Kallen, N., Pointer, R., Kalucki, J., & Ceaser, E. (08 de 01 de 2014). *FlockDB*. Obtenido de artiuculo: https://github.com/twitter/flockdb

Ken, N. (2009). *Databases in the cloud*. Obtenido de Article in Dr. Drobb's Magazine: http://www.drdobbs.com/database/218900502

Labs, F. (2010). *Tokyo Cabinet: a modern implementation of DBM.*Obtenido de http://1978th.net/tokyocabinet/

Lakshman, A., & Malik, P. (12 de 01 de 2014). *Cassandra - A Decentralized Structured Storage System.*Obtenido de documento: http://www.cs.cornell.edu/projects/ladis2009/papers/lakshman-ladis2009.pdf

Leslie, L. (14 de 25 de 2013). *The part-time parliament. In: ACM Transactions on Computer Systems.*Obtenido de Articulos Online: http://research.microsoft.com/en-us/um/people/lamport/pubs/lamport-paxos.pdf

Lin, L. (20 de 04 de 2009). *Some Notes on distributed key value Stores*. Obtenido de Blog post: http://randomfoo.net/2009/04/20/some-notes-on-distributed-key-stores

LITH, A., & MATTSSON, J. (2010). *Investigating storage solutions for large data.* (D. o. Engineering, Ed., & M. G. Inuca , Trad.) Sweden, Sweden.

López, F. A. (09 de 09 de 2011). *Articulo: Bases de Datos NOSQL*. Obtenido de BLOGSPOT: http://federico-wiesse.blogspot.com/2011/09/bases-de-datos-nosql_09.html

Mancuello, J. P. (11 de 12 de 2013). *Bases de datos NOSQ*. Obtenido de http://mistock.lcompras.biz/index.php?option=com_content&view=article&id=2395:httpwwwgenbetadevcombases-de-datosel-concepto-nosql-o-como-almacenar-tus-datos-en-una-base-de&catid=60:seginfo2011&Itemid=120: http://mistock.lcompras.biz/

Martin, L. (27 de 06 de 2013). *Principales tecnologías Big Data: NoSQL*. Obtenido de BLOG: http://www.brainsins.com/es/blog/principales-tecnologias-big-data-nosql/107943

MegaSonik. (1 de 04 de 2010). *NOSQL.es.* Obtenido de blog: http://www.nosql.es/blog/nosql/amazon-dynamo.html

MongoDB Inc. (02 de 01 de 2014). *MongoDB Información general*. Obtenido de Articulos: https://www.mongodb.com/products/mongodb

MongoDB.org. (20 de 01 de 2014). *Introduction to MongoDB.*Obtenido de Blogg: http://docs.mongodb.org/manual/core/introduction/

NAWROTH, A. (08 de 01 de 2014). *Las 10 mejores maneras de conocer Neo4j.* Obtenido de blog: http://blog.neo4j.org/2010/02/top-10-ways-to-get-to-know-neo4j.html

Neo4j. (05 de 01 de 2014). *Capítulo 19. API REST.* Obtenido de Documento: http://docs.neo4j.org/chunked/stable/rest-api.html

North, K. (01 de 09 de 2013). *Databases in the cloud.*Obtenido de dR. DROBB´S mAGAZINE: http://www.drdobbs.com/database/218900502

Paramio, C. (26 de 04 de 2011). *El concepto NoSQL, o cómo almacenar tus datos en una base de datos no relacional*. Obtenido de www.genbetadev.com: http://www.genbetadev.com/bases-de-datos/

Perez, M. (25 de 01 de 2014). *Cinco ventajas competitivas para las empresas que usen Big Data*. Obtenido de blogg: http://www.dirigentesdigital.com/tecnologia/38-tecnologia/214227-cinco-ventajas-competitivas-para-las-empresas-que-usen-big-data

Pritlove, T., Lehnardt, J., & Lang, A. (10 de 06 de 2009). *ouchDB – Die moderne Key/Value-Datenbank lädt Entwickler zum Entspannen ein.*Obtenido de Chaosradio Express Episode 125, Podcast published: http://chaosradio.ccc.de/cre125.html

Richard, J. (19 de 01 de 2009). *Anti-RDBMS: A list of distributed key-value stores*. Obtenido de Blog: http://www.metabrew.com/article/anti-rdbms-a-list-of-distributed-key-value-stores/

Sánchez, J. (01 de 01 de 2004). *Principios sobre Bases de Datos Relacionales.* Obtenido de Articulo: http://www.jorgesanchez.net/bd/bdrelacional.pdf

Scofield, B. (12 de 11 de 2013). *NoSQL Death to relational Databases.*Obtenido de Blog: http://www.slideshare.net/bscofield/nosql-codemash-2010

Silvescu, A., Caragea, D., & Atramentov, A. (25 de 12 de 2013). *Graph Databases.* Obtenido de Articulo: http://www.cs.iastate.edu/~silvescu/papers/gdb/report.pdf

Solis, S. M. (04 de 01 de 2014). *Herramientas para Big Data Neo4J* . Obtenido de Blog: http://santiagomarquezsolis.com/herramientas-para-big-data-neo4j-parte-1/

Steve, C. (01 de 2009). *MemcacheDB.* Obtenido de articulo: http://memcachedb.org/

Strauch, C. (2012). *NoSQL Databases.* Hochschule der Medien, Stuttgart.

TechPluto. (12 de 12 de 2013). *Characteristics of Web 2.0 Technology*. Obtenido de blog: http://www.techpluto.com/web-20-services/

Tiwari, S. (2011). *PROFESSIONAL NoSQL.* Indianapolis, Indiana: John Wiley & Sons, Inc.

What is a REST API [duplicate].(09 de 01 de 2014). Obtenido de Articulo: http://stackoverflow.com/questions/2217758/what-is-a-rest-api

White, T. (27 de 11 de 2007). *Consistent Hashing.* Obtenido de Blogger: https://weblogs.java.net/blog/tomwhite/archive/2007/11/consistent_hash.html

Wiesse, F. (09 de 09 de 2011). *Mi universo es una base de datos.* Recuperado el 18 de 12 de 2013, de Blog: http://federico-wiesse.blogspot.com/2011/09/bases-de-datos-nosql_09.html

Sobre los Autores

Inuca Morales Marco.- Nació en la parroquia de Antonio Ante el 25 de diciembre de 1986 termino la secundaria en el colegio Nacional San Pablo, estudio Ingeniería en Sistemas de Computación en la UTN en la FICA