2. Panorámico de las Bases de Datos

En el capítulo 2 se ofrece una introducción panorámica a las bases de datos y sus diferentes tecnologías, mostrando detalles particulares de cada una de ellas, así como cuotas de mercado que sirven para comprobar en qué situación se encuentra actualmente el importante mercado de las bases de datos.

Se presenta una cita que reflexiona sobre la importancia de controlar la información, y cómo el uso de herramientas de gestión de información eficientes supone un avance importante.

"*La cantidad de información crece de manera explosiva, y el valor de los datos como activo de las organizaciones está ampliamente reconocido. Para que los usuarios obtengan el máximo rendimiento de sus enormes y complejos conjuntos de datos son necesarias herramientas que simplifiquen las tareas de administrar los datos y de extraer información útil en el momento preciso. En caso contrario los datos pueden convertirse en una carga cuyo coste de adquisición y de gestión supere ampliamente el valor obtenido de ellos."* [1]

Quedando clara la importancia de las bases de datos, podemos definir éstas como un conjunto de datos pertenecientes al mismo contexto y almacenados para su posterior uso, dentro de un entorno digital gestionado mediante software y hardware. El software y hardware están en un proceso de cambio continuo, por lo que resulta importante saber en qué punto de desarrollo se encuentra cada tecnología para conocer qué nivel de beneficio se puede obtener de su uso.

Si bien la tecnología dominadora en las últimas décadas ha sido la formada por las bases de datos relacionales, podemos intuir nuevas corrientes debido a las distintas necesidades de las organizaciones, y sobre todo, debido a las magnitudes espectaculares de datos con las que algunas de ellas deben lidiar a diario. Asimismo, existe un tipo de datos muy útiles para el estructurado de documentos que tiene una sintaxis propia y una potencialidad y facilidad de uso destacable, el lenguaje XML; dada su importancia, suele implementarse con una capa sobre las bases de datos relacionales, sin embargo, dada la relevancia de estos documentos en algunas organizaciones, puede ser más interesante hacer uso de bases de datos que utilicen XML como sistema nativo, y no como una mera capa por encima de la base común, lo cual será tratado en este proyecto.

Al software que interactúa con los programas de aplicación del usuario y con la base de datos le llamamos Sistema Gestor de Bases de Datos, (SGDB). Permiten definir bases de datos haciendo uso, normalmente, de un lenguaje de definición de datos; pueden insertar, actualizar, borrar y extraer datos de la base de datos mediante un lenguaje de manipulación de datos, y a la misma vez proporcionan mecanismos de control de acceso a los usuarios. Cada solución utilizará un gestor específico que proporcionará un uso eficiente de la arquitectura bajo la que opera.

Si atendemos al desarrollo de nuevos negocios en Internet, el aumento de datos en los sistemas es imparable, con una audiencia de millones de usuarios potenciales ya no siempre es suficiente con las soluciones tradicionales. Empresas e instituciones que soportan una carga de tráfico de millones de usuarios deben hacer frente a una idea básica en el mundo de internet: El usuario no está dispuesto a esperar por recibir un servicio; por ello, se debe procurar ofrecer siempre un servicio estable, rápido, y fácilmente escalable a medida que vaya aumentando el número de usuarios. La solución posible para por el conglomerado de tecnologías NoSQL, que huyen de los esquemas estructurados estrictos tradicionales, y apuestan por una relajación de las exigencias que tan populares han hecho a las bases de datos relacionales.

En la siguiente figura se presenta un informe del estado del mercado de las Bases de Datos y su evolución futura, elaborado por la consultora Gartner Group en 2008.

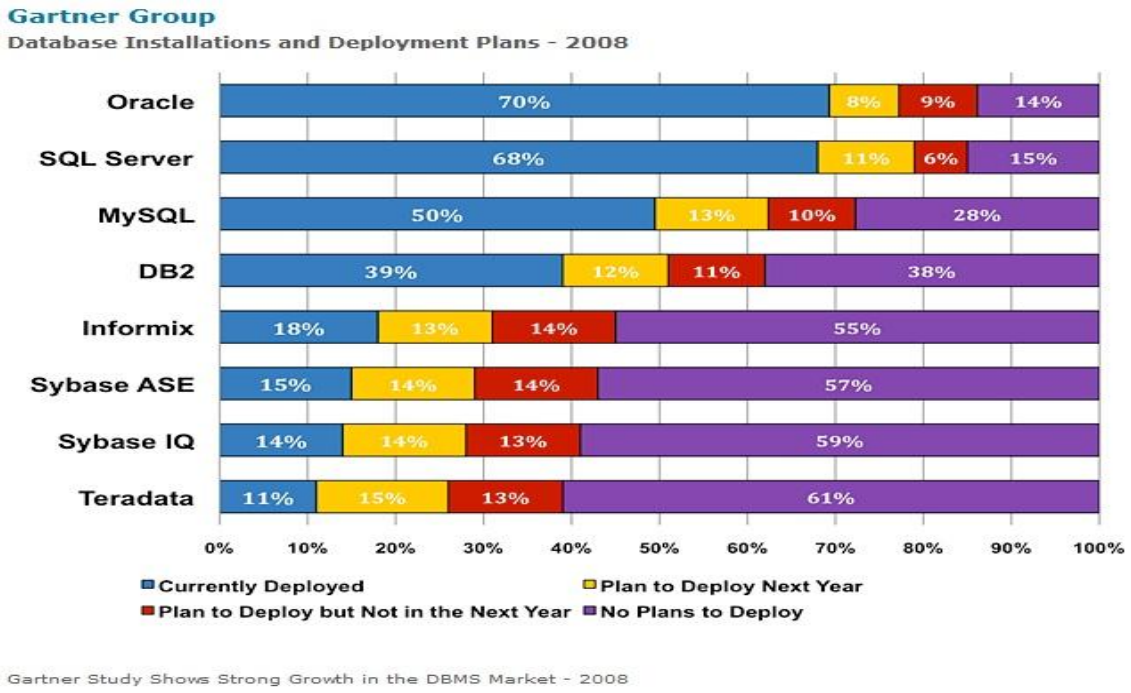

Figura 1: Estudio sobre la implantación de bases de datos. Gartner Group [2]

La cantidad de información que se maneja a día de hoy es tan alta, que en los próximos años veremos cómo puede variar la cuota de mercado en favor de aquellas bases de datos que mejor se adapten a lo que actualmente se viene a llamar Big Data, que no es más que el manejo de cantidades extremas de información. Según el famoso portal web Mashable.com[2], en una infografía presentada en junio de 2011, la cantidad de datos que se mueve anualmente se duplica cada dos años, siendo en 2011 una cantidad de 1,8 zettabytes, ($1{,}8 \times 10^{21}$ Bytes).

Para destacar la importancia que ahora mismo tiene poder manejar la información de manera eficiente, según un estudio de IBM de enero de 2012, el 90% de los datos disponibles en el mundo han sido creados en los 2 últimos años.[3]

Una vez hemos visto en este capítulo la importancia de las bases de datos, así como la idoneidad de cada una en función del tipo y tamaño de audiencia, el siguiente paso en este bloque será estudiar individualmente las tres tecnologías que se han mencionado:

2.1 Bases de datos relacionales, (basadas en lenguaje SQL)

2.2 Bases de datos no relacionales, NoSQL (Not Only SQL)

2.3 Bases de datos XML nativas

2.1 Bases de dato relacionales

En el capítulo introductorio se ha mostrado la relevancia que las bases de datos de tipo relacional tienen actualmente en las organizaciones, copando el mercado de manera muy notoria. Este apartado va a incidir en esa idea, mostrando por qué ha llegado a ser la tecnología preferida para la mayoría.

En el mercado aparecen muchas soluciones relacionales para resolver los problemas presentes en las organizaciones. Existen bases de datos comerciales y libres que se encuentran en continua batalla para conseguir nuevos usuarios, ofreciendo unos soportes muy profesionales y soluciones cada vez más versátiles y robustas que abarcan más tecnologías, en un intento por llegar a todos los usuarios posibles.

Estas bases de datos utilizan como lenguaje de consulta SQL, (Lenguaje Estructurado de Consulta). Es un lenguaje muy desarrollado y con un gran soporte tanto comercial como por parte de la comunidad de internet, lo que sin duda ha propiciado que sea el sistema preferido para la gran mayoría de organizaciones. El lenguaje de consulta permite trabajar con los datos de una base de datos, insertando, seleccionando o actualizando la información.

Los SGBDR, (Sistemas Gestores de Bases de Datos Relacionales) están basados en el modelo de datos relacional propuesto por E.F. Codd en 1970 en su estudio *"A relational model for large shared data banks"*; en este modelo se pretende:

- Permitir independencia en los datos, separando la capa de datos de la capa de aplicación
- Dotar de una base teórica sólida para trabajar los problemas de coherencia y redundancia

Si nos centramos en su arquitectura, una base de datos relacional consiste en un conjunto de tablas, a las que les corresponde un nombre exclusivo; cada fila de la tabla, o tupla, simboliza una relación entre un conjunto de valores. A las columnas de dichas tablas se les llama atributo. Estas bases de datos presentan tablas con grado y cardinalidad; la primera hace referencia al número de atributos que tiene una tabla, mientras que la segunda indica el número de tuplas que están presentes.

Es muy importante destacar las propiedades que deben presentar las relaciones:

- Cada tabla, o relación, tendrá un nombre distinto

- Cada atributo tendrá un nombre distintivo
- Cada tupla es diferente, no habrá filas duplicadas
- Cada celda de la relación contiene un valor único
- Todos los valores de un atributo pertenecen al mismo dominio, (conjunto de valores permitido)

Son estas mismas propiedades las que harán que en ocasiones la rapidez de respuesta del sistema no sea adecuada, debido a la complejidad de cumplir con estas normas, que lleva implícito un alto índice de operaciones de comprobación que ralentizan el comportamiento normal de una base de datos. Es por ello que en ocasiones es necesario optar por otro tipo de bases de datos o, en su defecto, alcanzar un compromiso entre estas propiedades y las necesidades de la organización.

Si atendemos a la consultora IDC, en su estudio de 2011, las bases de datos con mayor cuota de mercado son las que siguen:

	2008		2009		
	Revenue ($M)	Share (%)	Revenue ($M)	Share (%)	2008–2009 Growth (%)
Oracle Corp.	3,641.0	32.4	3,542.5	31.8	-2.7
Microsoft	2,792.0	24.9	2,921.2	26.3	4.6
IBM	3,053.7	27.2	2,882.0	25.9	-5.6
Teradata	430.2	3.8	412.5	3.7	-4.1
Sybase	253.9	2.3	295.8	2.7	16.5
Fujitsu	89.6	0.8	111.1	1.0	24.0
Netezza	88.4	0.8	94.8	0.9	7.2
Progress Software	80.1	0.7	79.0	0.7	-1.4
SAS Institute	72.1	0.6	59.1	0.5	-18.0
Hitachi	39.1	0.3	39.3	0.4	0.5
Other	687.0	6.1	689.7	6.2	0.4
Total	11,227.1	100.0	11,127.0	100.0	-0.9

Source: IDC, 2010

Figura 2: Cuota de mercado bases de datos relacionales

En este apartado se han visto algunos de los aspectos más destacados de las bases de datos relaciones, así como sus sistemas gestores, dando una visión global que permita entender por qué son las que claramente dominan el mercado del que hablamos. A continuación se muestra en qué consisten las bases de datos XML nativas.

2.2 Bases de datos XML nativas

El apartado 2.2 se centra en las bases de datos que trabajan de manera nativa con lenguaje XML, con sus propias peculiaridades y su propio lenguaje de consulta, distinto del que se ha visto anteriormente para bases de datos relacionales, SQL.

Como se ha mencionado, estas bases de datos trabajan con lenguaje XML, (lenguaje de marcas extensible), que destaca por su facilidad de uso y por ser una herramienta muy potente, capaz de trabajar con documentos tan complejos como exija la situación. El hecho de ser un lenguaje de marcado implica que describe documentos incluyendo etiquetas en el mismo, y como característica destacable está que no presenta funciones, ni bucles.

Hay que tener en cuenta que todos los documentos XML crean sus propios lenguajes de marcado personal, es decir, XML es el sustento para crear nuestro propio etiquetado. Es un estándar abierto creado por la World Wide Web Consortium (WC3), y ha sido creado para trabajar con documentos altamente estructurados en internet. Permite que distintas aplicaciones puedan comunicarse, compartiendo toda clase de información, si éstas entienden XML.

La clave que permite que distintas aplicaciones que comprenden XML puedan entender el lenguaje de marcado personal de cada documento reside en que cuando se envía un documento XML, también se está enviando el diccionario con su lenguaje personalizado. De esta forma, cualquier aplicación que pueda analizar la sintaxis de XML, podrá entender cualquier documento XML.

El mismo WC3 [8] nos define la tecnología XML como un conjunto de módulos que ofrecen servicios útiles a las necesidades más frecuentes de los usuarios, sirviendo todo el conjunto para almacenar, estructurar e intercambiar información. Las tecnologías XML más destacadas son:

- XSL: Lenguaje extensible de hojas de estilo, que muestra cómo debería estar estructurado el contenido, cómo debería ser diseñado, e incluso cómo debería ser presentado.
- XPath: Lenguaje de rutas XML, para acceder a partes de un documento XML
- XLink: Lenguaje de enlace XML, que permite insertar elementos en documentos XML para crear enlaces entre recursos XML
- XPointer: Lenguaje de direccionamiento XML, para acceder a la estructura interna de un documento XML, (elementos, atributos y contenido).

- XQL: Lenguaje de consulta XML, que facilita la extracción de datos desde documentos XML

Dos elementos importantes son el DTD, (definición del tipo de documento), que define tanto los elementos de una página como sus atributos, y el XML Esquema, que define qué elementos puede tener un documento XML, cómo están organizados, y qué atributos y de qué tipo pueden tener sus elementos.

Si el documento de interés está centrado en el contenido, (con una estructura irregular), no podremos controlar todas las posibles estructuras del documento, por lo que resulta interesante en este caso el uso de bases de datos XML nativas, que implementan algoritmos de búsqueda e índices específicos que aceleran el acceso a los documentos.

Son bases de datos, y por tanto soportan transacciones, acceso multi-usuario, lenguajes de consulta y demás características propias, pero diseñadas especialmente para el almacenamiento de documentos XML. Algunas características dignas de mencionar son:

- La unidad lógica fundamental de almacenamiento es el documento XML
- Soporta lenguajes de consulta XML
- Preserva el orden del documento, (como mínimo el modelo debe incluir elementos, atributos, manejo de PCDATA y orden dentro del documento)
- Puede usar cualquier estrategia de almacenamiento
- Los documentos se suelen agrupar en colecciones en función de la información contenida
- Validación de los documentos
- Permiten la creación de índices XML que aceleren las consultas realizadas con más frecuencia
- Cada documento XML tendrá un identificador único

Para comprobar cómo trabaja el lenguaje XML se ha escrito una pequeña defunción que sirve de ejemplo. En el ejemplo se observa como la etiqueta autobuses, que puede referirse a aquellos que se encuentren en una estación, tienen como subcampo la etiqueta línea, que especifica las características propias de cada una de las líneas que operan, mostrando un identificador, su matrícula, y las direcciones de origen y destino del trayecto.

```
<Autobuses>
    <Línea>
        <Compañía ID= ab32-di45>Sevilla bus</Compañía>
        <Matrícula>5328</Matrícula>
        <Origen>Sevilla</Origen>
        <Destino>Madrid</Destino>
    </Línea>
    <Línea>
        <Compañía ID= ax36-co45>Molina bus</Compañía>
        <Matrícula>6452</Matrícula>
        <Origen>Sevilla</Origen>
        <Destino>Córdoba</Destino>
    </Línea>

</Autobuses>
```

Figura 3: Ejemplo de código XML

En el mercado, destacan distintas soluciones de bases de datos XML nativas, desde las comerciales, hasta las de libre distribución. También es interesante distinguirlas en función de su organización interna.

NOMBRE	LICENCIA
Tamino	COMERCIAL
eXist	OPEN SOURCE
Xindice	OPEN SOURCE
Marklogic	COMERCIAL

Figura 4: Soluciones de Bases de Datos XML Nativas

Tamino es una solución comercial, proporcionada por SoftwareAG que permite generar, procesar e intercambiar documentos XML. Presenta soporte para los principales Sistemas Operativos, y para las plataformas .NET y J2EE. Permite lenguaje de consulta Xquery.

EXist continúa dando soporte actualizado a su solución de libre distribución, soportando Xquery 1.0, XPath 2.0 y XSLT 2.0; presenta interfaces http: Rest, WebDAVy SOAP entre otras.

Xindice es un proyecto de la fundación Apache, que con el tiempo fue perdiendo fuerza y que a día de hoy ha perdido el soporte. Soporta XPath para las consultas y presenta API para Java.

Marklogic es otra de las soluciones de plena actualidad, que será tratada con mayor profundidad en el capítulo 4, y de la que cabe destacar la gran cantidad de archivos soportados, su soporte para la gran mayoría de Sistemas Operativas, así como los drivers que proporcionan para trabajar con los

lenguajes C# y Java, y la muy cuidada interfaz gráfica basada en http que permite trabajar de forma intuitiva con los datos. Permite lenguaje de consulta Xquery, XPath y XLST, así como la compresión de datos para su almacenamiento interno.

En el apartado que termina se han mostrado algunas de las características que hacen de las bases de datos XML nativas una solución muy válida para algunos propósitos específicos. En ocasiones se opina que éstas forman parte del conjunto de tecnologías NoSQL que detallaremos en el apartado 2.3

2.3 Bases de dato NoSQL

En los anteriores apartados se han presentado las tecnologías relacionales y XML como bases de la arquitectura de unas determinadas bases de datos. En este punto se va a estudiar el porqué de la aparición del conjunto de tecnologías NoSQL, (Not Only SQL).

Con la llegada de la web y los servicios en la "nube" el crecimiento del número de usuarios en las compañías que operan en internet ha sido espectacular, encontrando fácilmente empresas de internet que superan ampliamente el millón de usuarios. Ante el problema del volumen de datos masivos y simultaneidad de accesos de grandes cantidades de usuarios, aparecen soluciones que relajan en cierta forma las propiedades de las bases de datos relacionales, y esto es así porque la escalabilidad de las soluciones supone un punto básico en el desarrollo de las mismas.

La arquitectura intrínseca de las bases de datos relacionales provoca que no se puedan distribuir de forma sencilla sobre varios servidores, y esto es muy importante debido al descenso del rendimiento que experimentan los sistemas de bases de datos relacionales no distribuidos a medida que crece el número de usuarios.

Se presenta a continuación los resultados del estudio que el proveedor de la base de datos CouchDB, (NoSQL), ha realizado sobre el rendimiento de las bases de datos frente al número de usuarios.

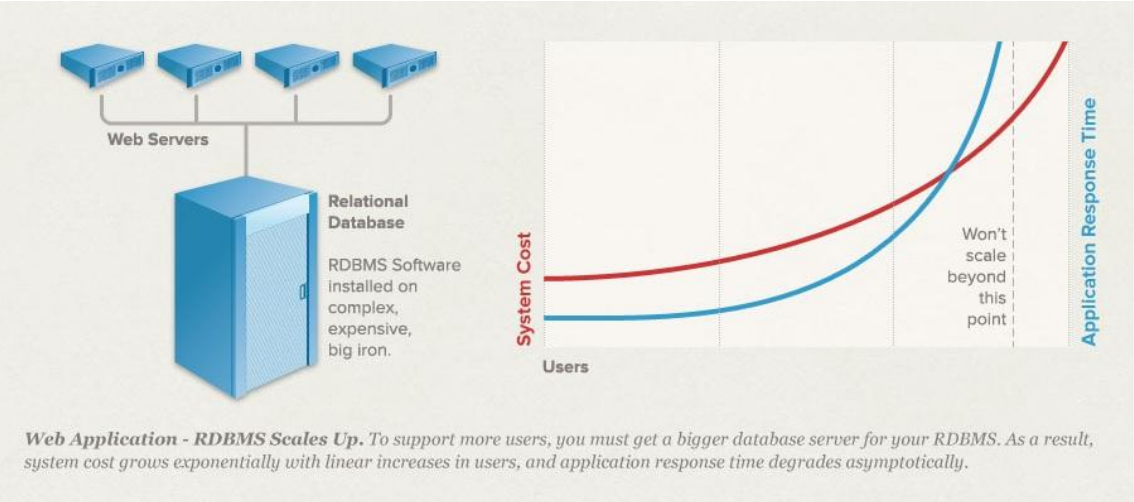

Figura 5: Comportamiento de los SGBDR frente a usuarios [3]

Esta gráfica se ha obtenido directamente de la documentación de su página web[3], y muestra cómo para el caso de bases de datos relacionales, a medida que aumenta el número de usuarios, (escala horizontal), el número de

servidores ha de ser mayor por los problemas de escalabilidad que presentan, y esto lleva a un mayor tiempo de respuesta de las aplicaciones, un peor servicio al usuario, y además, un aumento de costes debido a la compra de nuevos equipos.

En contraposición con lo anterior, y según los estudios de CouchDB [3], las bases de datos NoSQL son particularmente buenas para la escalabilidad, ofreciendo una respuesta lineal en cuanto al coste del sistema a medida que aumenta el número de usuarios, y como dato más destacable, un tiempo de respuesta de las aplicaciones prácticamente iguales a pesar de aumentar la carga de usuarios en el sistema. Esto puede verse en la gráfica que se presenta:

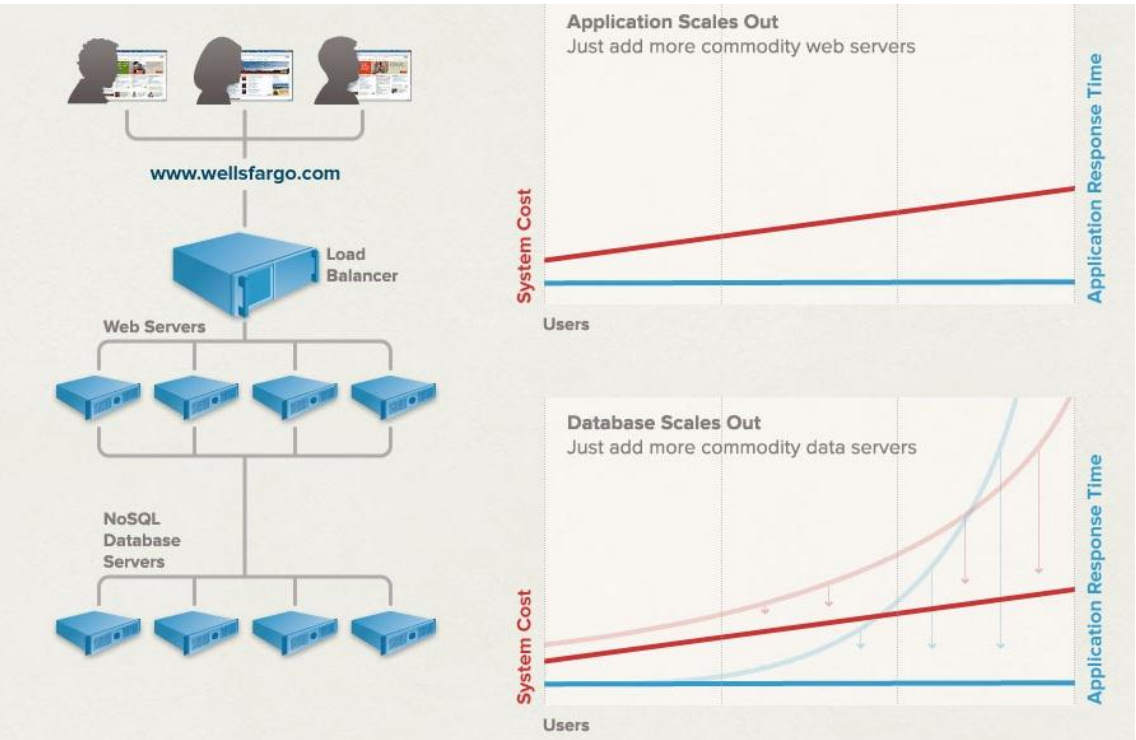

Figura 6: Facilidades de escalado de NoSQL frente a SGBDR [3]

Las gráficas vistas anteriormente inciden en la idea de que NoSQL puede ofrecer un servicio rápido y de bajo coste para soluciones en las que el cliente necesite almacenar una gran cantidad de datos.

NoSQL propone una estructura de almacenamiento más flexible, que relaja las exigencias de las robustas bases de datos SQL. NoSQL no es en sí misma una tecnología, sino que agrupa un conjunto de tecnologías que tienen en común lo que no son: bases de datos relacionales. Esta tendencia olvida los esquemas estructurados en tablas y trabaja con datos desnormalizados, siendo pensados para conseguir escalabilidad horizontal sencilla, así como una velocidad de procesado muy alta.

Cuando los datos no siguen un esquema uniforme es cuando aparece NoSQL; no es necesario hacer JOIN, (forma que tienen las bases de datos relacionales de hacer consultas que involucran varias tablas), cada vez que se quieran relacionar datos, ya que cada registro puede contener diferentes tipos de información en cada ocasión, utilizando sólo aquellos que nos sean de interés.

Estas bases de datos se apoyan en el teorema CAP de Brewer, que establece que es imposible para un sistema distribuido dar simultáneamente las siguientes tres garantías:

- Consistencia: Todos los nodos ven la misma información al mismo tiempo
- Disponibilidad: El fallo de un nodo no impedirá al resto seguir funcionando
- Tolerancia a fallos: El sistema seguirá funcionando a pesar de la existencia de pérdidas de información o fallos parciales del sistema.

Según este teorema, de las tres garantías anteriores, en los sistemas de bases de datos distribuidos sólo podremos asegurar dos al mismo tiempo. Los sistemas No-SQL buscan un compromiso entre estas tres garantías, siendo el más habitual el modelo de consistencia eventual, en el que se garantizan la disponibilidad y la tolerancia a fallos, mientras que se intenta dar el mayor grado de consistencia posible.

La escalabilidad horizontal es básica en este caso, ya que facilita la integración de nuevos nodos al sistema de bases de datos, de forma que aumentamos el rendimiento del mismo sin una gran complejidad operativa.

Una característica destacable de estos sistemas es que realizan operaciones directamente en memoria, transfiriendo solamente los datos a disco cada cierto tiempo, lo que otorga una rapidez de ejecución notable, a costa de perder consistencia en los datos, ya que en caso de apagón o caída de servicio pueden perderse la información.

Por tanto, podemos concluir que este conjunto de tecnologías de bases de datos no relacionales tienen como finalidad tratar con los datos a muy alta velocidad, lo que permite utilizarlos como servicios de almacenamiento primario, sistemas de persistencia para almacenar cachés o analíticas de uso, entre otros.

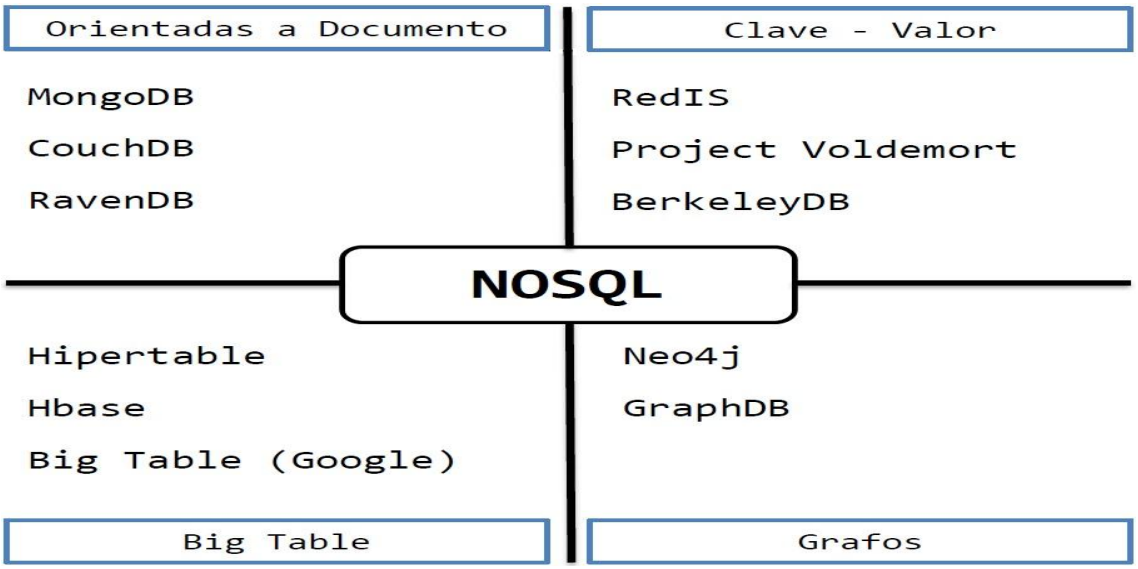

Figura 7: Tipos de Bases de Datos NoSQL

A continuación se describen los tipos de bases de datos NoSQL más destacados, y que aparecen en la gráfica anterior:

2.3.1 Clave-Valor

Las bases de datos NoSQL más simples de utilizar son aquellas que se basan en el paradigma clave-valor, (key-value en inglés).

Para añadir un dato a la base de datos, se ha de indicar cuál es la clave que se va a asociar al mismo. Igual ocurre cuando queremos recuperar un dato de la misma, para lo cual se usan objetos binarios (BLOB). Esto facilita operaciones más rápidas, que redundan en un uso eficaz del sistema. El gran beneficio que ofrecen estas bases de datos radica en el rendimiento anteriormente mencionado, ya que resulta más sencillo de escalar que otras soluciones, y esto es muy importante para compañías que manejen cantidades de datos demasiado grandes. Son muy habituales a la hora de tratar con perfiles de usuarios, sus sesiones, o incluso carros de la compra típicos de empresas online, como Amazon, que utiliza la solución Amazon Dinamo.

2.3.2 Orientadas a Documento

Las bases de datos orientadas a documentos son posiblemente las más extendidas, dada su cercanía hacia las bases de datos relacionales, que supone menos esfuerzo al usuario a la hora de utilizarlas.

Estas bases de datos se encuentran libres de esquemas, cada documento puede tener su propia estructura, incluso varios documentos dentro de la misma colección pueden diferir en sus atributos, y más allá, se pueden

actualizar los documentos quitando o añadiendo campos sin el mayor problema.

Internamente también maneja un sistema de clave-valor, de manera que puede relacionar documentos independientes haciendo uso de sus claves asociadas, que pueden añadirse como un campo más del documento al que se quiere refenciar. A pesar de trabajar con clave-valor, las consultas no sólo son basadas en clave, sino que la gran ventaja de estos sistemas es que puede llevarse a cabo una consulta sobre cualquier campo que pueda contener un documento, quitando las limitaciones que en ese punto ofrecían las bases de datos NoSQL clave-valor vistas anteriormente.

Manejan datos en formato JSON, (JavaScript Object Notation), que consiste en un formato ligero para el intercambio de datos que se presenta como alternativa a XML. En el caso de la base de datos orientada a documentos MongoDB, BSON, (Binary JavaScript Object Notation), es el formato utilizado.

2.3.3 Grafos

Las bases de datos NoSQL basadas en grafos son mayormente utilizadas cuando aparecen datos con muchas relaciones entre sí.

Estas bases de datos relacionan información de muy distinta manera, y en varias ocasiones si fuera necesario. Estas relaciones pueden ser unidireccionales o bidireccionales, y además pueden llevar consigo información asociada.

Muchas de las redes sociales utilizan en cierta forma este tipo de bases de datos, ya que trabajan de forma más intuitiva con las relaciones de los registros, como por ejemplo las relaciones que existen entre personas. También se usan asiduamente para controlar y resolver problemas en redes interconectadas.

2.3.4 BigTable

Estas bases de datos son realmente basadas en columnas, pero reciben este nombre por el impulso que dio Google con su base de datos Google Bigtable. En cierta forma tiene similitudes con las bases de datos relacionales, aunque en el caso que nos ocupa el almacenamiento de datos se hace por columnas, en lugar de hacerlo por filas.

La organización de los datos en columnas tiene algunas particularidades que se exponen seguidamente:

- Familias de columnas: Una familia de columnas es cómo se almacena la información de manera física. Todos los datos de una familia de columnas irá almacenado en el mismo registro. Pueden contener Súper Columnas, o Columnas simples
- Súper Columnas: Se trata de columnas que contienen otras columnas.
- Columnas: Son el conjunto básico de datos

Tipo de Base de Datos NoSQL	Ventajas	Inconvenientes
Clave-Valor	- Modelo simple - Alto rendimiento - Optimizado para gran cantidad de accesos	- No permite consultas más allá del valor de la clave
Orientada a Documentos	- Sin esquema - Todo tipo de consultas - Alto rendimiento general	- Si no se usan índices bajan el rendimiento
Grafos	- Rápido para consultas de relaciones entre miembros	- Menos adecuado para uso general como base de datos
Basadas en Columnas	- Optimizadas para guardar grandes volúmenes de datos - Útiles para data mining	- Latencia y tiempo de respuestas no demasiado buenos

Figura 8: Ventajas e inconvenientes de las diferentes bases de datos NoSQL

Una vez vistos los tipos de bases de datos NoSQL más destacados, se presentan algunos datos en cuanto al mercado que ocupan actualmente, y su comparación con otras tecnologías.

Algunas bases de datos NoSQL empiezan a ocupar una buena parte del mercado de las bases de datos de libre distribución; si bien su porcentaje es aún reducido respecto a alternativas relacionales de carácter comercial, el hecho de que se trate de tecnología relativamente joven, y con una introducción aún lenta

debido a la amplia acogida de las bases de datos de Oracle, IBM o Microsoft, hacen pensar que el camino que aún le queda por recorrer a este conjunto de tecnologías puede ser exitoso.

Se muestra a continuación estadísticas del mes de abril de 2012 en la que puede observarse los porcentajes de uso según los datos del sitio web de hospedaje Jelastic.com

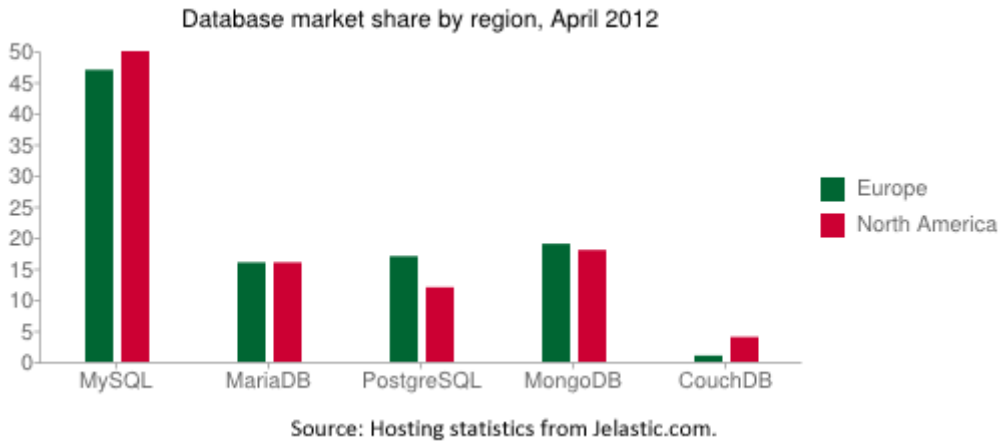

Figura 9: Porcentaje de uso de bases de datos OpenSource en sitio de hospedaje Jelastic.com

La gráfica anterior muestra los resultados porcentuales de las bases de datos OpenSource que usan los usuarios del sitio web de hospedaje Jelastic.com. Puede observarse cómo MongoDB, de tipo NoSQL tiene porcentajes algo superiores a algunas soluciones clásicas como PostgreSQL o MariaDB.

Concluida la introducción a NoSQL, en la que se han visto los distintos tipos de arquitecturas utilizadas, y donde se han ofrecido ventajas e inconvenientes de uso, se pasa a estudiar el concepto de escalabilidad, clave para tratar con volúmenes grandes de datos y que indica la facilidad de un sistema para adaptarse a la variabilidad de accesos y de información.

2.4 Escalabilidad

En anteriores apartados se ha hecho hincapié en la importancia de tratar eficazmente grandes cantidades de datos. De la arquitectura utilizada por cada solución para dar respuesta a este problema ya se ha hablado, y a continuación se desarrolla el concepto de escalabilidad, clave para entender cómo de bien se adaptan las distintas bases de datos a un aumento lineal o desmesurado de datos en el tiempo.

Los datos tratados por los sistemas informáticos actuales se cuentan por cientos, miles o millones dependiendo del contexto en el que se trabaje. Como se ha indicado en este proyecto, las necesidades de las organizaciones serán muy diferentes unas a otras, y es necesario adoptar soluciones válidas para cada caso particular, ya que una solución generalista terminará por ser ineficiente.

Los datos no son estáticos, sino que crecen en número, o disminuyen, si bien la norma indica que los datos crecen en cada sistema de información día a día. Esto nos lleva a la necesidad de escalar nuestra solución, que no es más que aumentar las capacidades, recursos y rendimiento de aquello con lo que trabajamos, que de forma fluida nos ayuda a hacer más grandes nuestros sistemas.

Se presentan dos formas de escalabilidad: Vertical y Horizontal

2.4.1 Escalabilidad Vertical

Escalar verticalmente un sistema consiste en aumentar la capacidad hardware de un nodo, (equipo donde se alojará el sistema de base de datos), con el fin de mejorar las características del ya existente y obtener mejores resultados en el propósito que se le asigne.

Normalmente la escalabilidad vertical supondrá comprar nuevos componentes que agregar a nuestro sistema, o sustituirlo por uno nuevo y más potente. Suele resultar costoso, en tanto en cuanto trabajar con tecnología punta siempre acarrea un precio superior, y además cada cierto tiempo habrá que volver a realizar la misma operación de escalabilidad vertical cuando los equipos se vuelvan obsoletos.

2.4.2 Escalabilidad Horizontal

Escalar horizontalmente un sistema consiste en aumentar la capacidad de un sistema a base de aumentar el número de nodos. La idea central es conseguir un rendimiento igual o superior que con la escalabilidad vertical, pero usando

varios equipos de prestaciones inferiores, lo que bien ejecutado puede suponer un ahorro de costes.

Las empresas cargan más datos cada cierto tiempo, mucho más aun las que tienen base en Internet y suman cientos o miles de usuarios al día, o publican constantemente información que debe ser almacenada. Esto hace pensar que escalar verticalmente no va a ser una solución adecuada, ya que no podemos estar modificando nuestro hardware casi a diario, por ello, una escalabilidad horizontal que permita a estas organizaciones aumentar su rendimiento agregando nuevos nodos a su red resulta muy ventajosa.

Con una optimización adecuada del sistema, y una previsión adecuada que contemple cómo escalarlo, se puede conseguir un aumento de nuestra capacidad con un esfuerzo no excesivo, que dependerá principalmente de la arquitectura propia del sistema y del capital humano que trabaje con él.

Las bases de datos relacionales tienen ciertos problemas para la escalabilidad horizontal, resulta complejo y necesitan muchas horas de trabajo, mientras que las bases de datos NoSQL cuentan como uno de sus principales atractivos con la facilidad de escalar sus soluciones, que de forma nativa implementan de forma eficiente.

Recordar que en el bloque 1 se han visto los objetivos de este proyecto y una panorámica introductoria a las bases de datos, ofreciendo una imagen global de las diferentes tecnologías existentes y viendo qué cuota de mercado ocupa cada una de ellas. También se ha visto el concepto de escalabilidad, que determinará lo bien que un sistema se adapta al incremento de datos y accesos. Vistos estos puntos, en el bloque 2 se va a profundizar en una solución para cada tecnología vista en el bloque 1, mostrando los aspectos más destacables en cuanto a arquitectura y tratamiento de datos de cada una. Por tanto, los siguientes puntos serán:

3.Base de datos relacional: MySQL
4.Base de datos XML nativa: Marklogic
5.Base de datos NoSQL: MongoDB