LA TRANSICIÓN DEL MANEJO DE BASES DE DATOS ENTRE EL MODELO SQL AL NOSQL EN LA ENSEÑANZA DE CARRERAS DE TECNOLÓGICAS

THE DATABASE MANAGEMENT TRANSITION BETWEEN THE SQL TO NOSQL MODEL IN TEACHING OF TECHNOLOGICAL CAREERS

https://doi.org/10.5281/zenodo.3598497

AUTORES: Harry Saltos Viteri[1]*

Miguel Franco Bayas[2]

DIRECCIÓN PARA CORRESPONDENCIA: hsaltos@utb.edu.ec

Fecha de recepción: 08 / 10 / 2019

Fecha de aceptación: 10 / 12 / 2019

RESUMEN

La familiaridad con las bases de datos durante largo tiempo ha dejado establecido un modelo relacional como el más adecuado, ya que este ha dado resultados por mucho tiempo, y logró una maduración y consistencia que le ha permitido sobre todo la confianza de varias generaciones de desarrolladores, con este modelo se ha utilizado la tecnología *SQL*, con su lógica sencilla y estructurada; además, el análisis permitió determinar que la tendencia en la formación de estudiantes de ingeniería en sistemas de información, es incorporar los dos modelos y arquitecturas de bases de datos, las herramientas que se encuentran disponibles para usar bases de datos documentales no son tan amigables y tienen acceso de administración por consola, y se requiere conocimiento amplio de sus instrucciones para poder explotarlo a conveniencia. Aún se muestra muy apegada en los conocimientos de los estudiantes el modelo relacional de bases de datos, así lo representa un 45% de los entrevistados.

Palabras clave: SQL, NoSQL, DBMS, bases de datos, Big Data

[1]*Magister en Ingeniería y Sistemas de Computación, Universidad Técnica de Babahoyo, hsaltos@utb.edu.ec
[2]Ingeniero en Telecomunicaciones con mención en Gestión Empresarial en Telecomunicaciones, Universidad Técnica de Babahoyo, mfrancob@utb.edu.ec

ABSTRACT

The familiarity with the databases for a long time has established a relational model as the most appropriate, since this has given results for a long time, and achieved maturation and consistency that has allowed it above all the trust of several generations of developers, with this model the *SQL* technology has been used, with its simple and structured logic; In addition, the analysis allowed to determine that the tendency in the formation of engineering students in information systems is to incorporate the two models and architectures of databases, the tools that are available to use documentary databases are not so friendly and They have administration access by console, and extensive knowledge of their instructions is required to be able to exploit it for convenience. The relational database model is still very attached to the students' knowledge, as represented by 45% of the interviewees.

Keywords: SQL, NoSQL, DBMS, Database, Big Data

INTRODUCCIÓN

Las bases de datos podrían tenérselas consideradas como como la televisión, ya que hace un tiempo en la historia de ambos cuando se tenía pocas opciones para elegir y todas estas opciones eran en ocasiones decepcionantes, pero cambiaron los tiempos. Los sistemas manejadores de bases de datos ya no es un sinónimo de bases de datos relacionales, y así mismo la televisión actual ya no se limita únicamente a pocas redes que transmiten programas idénticos (Sullivan, 2015).

Como futuros desarrolladores certificados, según (Díaz, 2019), es común que en algún momento de su formación se hayan topado con conceptos fundamentales de bases de datos, ya que la información o datos que en un futuro o en momentos actuales de los software y sistemas web, apps móviles y otros que han de ser desarrollados y necesitan almacenarse en algún lugar para su posterior consulta o tratamiento de información.

Los conceptos iniciales de bases de datos con los que se han familiarizado de forma general son los apegados al modelo relacional, ya que este fue el que hizo popular a las bases de datos por su forma de organizar los datos y producir o consultar la información

Al usar este modelo relacional de bases de datos, según (López, 2019), estamos estrechamente familiarizados con *SQL (Structure Query Languaje)* y su lógica y forma en

que este permite usar su semántica o lenguaje para definir datos y las consultas de los mismos en sistemas manejadores de bases de datos DBMS.

También Tablas con columnas y registros, relaciones, restricciones, *querys, functions* y *store procedures,* es terminología que se utiliza de forma común al referirse a los datos de los futuros sistemas de información y fueron forjados esos datos como tal en modelos relacionales.

Quienes incursionan en nuevos conocimientos relacionados con bases de datos tienen en su mente configurada ya sus datos expuestos en otros modelos, es el sentido por lo que este artículo se ha desarrollado en el que se hace la investigación acerca de una transición que puede no ser tan sencilla de comprender, porque habría que desaprender en muchos casos para poder adaptarse a los nuevos modelos que demanda la tecnología de los actuales momentos, o aquellos que pretenden o quieran migrar de *SQL a NoSQL* sin tantos inconvenientes.

METODOLOGÍA

Para este artículo, se ha establecido la metodología del análisis-síntesis, porque consiste en el estudio independiente de cada parte de este artículo donde existió investigación de material necesario en cuanto a lo tecnológico, lo educacional y lo funcional (Maya, 2014).

Se utilizaron varias fuentes documentales de lo más necesario, que representa las tecnologías de bases de datos existentes y se agregó mucho contenido relacionado con el pensamiento del autor.

Se realizó conversatorios con varios estudiantes de carreras de ingeniería en sistemas y sistemas de información de la Universidad Técnica de Babahoyo, de la ESPOL y UNIANDES, llegando a un número de 55 personas, en todos los casos han mantenido contacto con bases de datos con tecnologías de modelo relacional y documental.

Se procesó información con modalidad de encuesta y se procedió a realizar un análisis profundo en lo que relaciona a la parte documental y lo opinado por los participantes de esta investigación.

RESULTADOS

Un concepto fundamental cuando se trata del modelo relacional es el de la relación, que fue postulado por un matemático y científico británico Edgar F. Codd donde menciona que una

relación, según (León, 2018) está representada por un conjunto de entidades con las mismas propiedades. Estas relaciones están compuestas por registros a las que también se las puede conocer como filas o tuplas, cuyos valores dependen de sus atributos o columnas.

Al observar los gestores más usados hasta los actuales momentos, teniendo en cuenta que el manejo electrónico de datos y generación de información ha estado regido por el conocido modelo relacional, a continuación, se muestran los más utilizados por su confiabilidad y rapidez:

Microsoft SQL Server: Está disponible con una licencia Microsoft de pago y además en los últimos años se lo puede usar en Linux.

Db2: Un SGBD relacional propietario de la empresa IBM, que tiene mucho tiempo en el mercado de las bases de datos.

MySQL: De código abierto, últimamente es el más utilizado a nivel global. Este pasó a manos de *Oracle*, *MySQL* distribuyéndose con una licencia dual. Los primeros desarrolladores siguen encargándose del proyecto, que tiene como nombre actual *MariaDB.*

Oracle Database: uno de los colosos de la industria de las bases de datos, se distribuye como software licenciado.

PostgreSQL: Es un sistema relacional libre y orientado a objetos tiene una comunidad que lo sigue desarrollando por ser *open source*.

SQLite: Constituye una biblioteca de librerías y programas con licencia de dominio público que contiene un motor de gestión de bases de datos relacionales.

Todos estos sistemas gestores de bases de datos, están basados con una lógica tabular de organización de la información.

Las bases de datos relacionales, según (Silberschatz, F. Korth, & Sudarshan, 2002) funcionan estructuradas por tablas que constituyen en sí su principal relación porque es donde se juntan de forma vinculante los registros; el sistema relacional define su estructura y gestiona además los permisos de escritura y lectura para poder interactuar con las diferentes tablas y así ejercer operaciones de todo tipo.

Los usuarios utilizan un lenguaje de base de datos muchas veces propio del gestor o estandarizado pues al menos todos estos gestores relacionales, según (Silberschatz, F.

Korth, & Sudarshan, 2002) soportan al menos un lenguaje que permite que se puedan ejecutar las siguientes operaciones:

DDL Es una forma de definición de Estructura de datos donde se almacena en forma de metadatos de la estructura definida Cuando se hace una operación de creación de tabla nueva en un diccionario de datos se guarda el correspondiente esquema a este se le denomina *Data Definition Languaje*.

DCL Donde se proporciona la sintaxis correspondiente a gestionar los permisos es un vocabulario integrado que poseen las bases de datos relacionales, se denomina *Data Control Language*.

DCI Permite las Condiciones de Integridad referentes a las bases de datos Está le proporcionan condiciones de integridad para garantizar en todo momento de los datos al momento que se está operando con ellos pues obliga a cumplir condiciones básicas por ejemplo que cada registro o tupla pueda identificarse de forma inequívoca.

DT Se definen transacciones también, Es decir se prepara a la base de datos para que pase de un estado consistente a otro diferente, son instrucciones programadas que deben ejecutarse siempre de forma íntegra y con controles de interrupción para garantizar que los datos sean almacenados de forma correcta es decir que todo llegué a guardarse o no guardarse de ser el caso si es que existe una interrupción en la comunicación ya sea originada por algún apagón o por un desperfecto del *Front end.*

Cuando se interrumpe una transacción puede hacerse la volver a su estado original conociéndose esto como un *rollback*, Estás transacciones pueden ir viajando en un orden y creando conexiones con la base de datos, así como también se dispone de un paso de comprobación denominado *Commit* que asegura la integridad de los datos Siempre que éstos hayan llegado a su destino de forma completa.

DV La base de datos también tienen la posibilidad de definir vistas Qué son como Disponer virtualmente de una consulta previamente compilada.

En un modelo relacional de bases de datos se utiliza de forma estandarizada el lenguaje *SQL* para todas las operaciones denominadas sentencias, basadas en algebra relacional.

En el modelo relacional, las operaciones de consultar, insertar, actualizar o borrar datos, se las realiza por medio de sentencias *SQL* que es un lenguaje con órdenes y comandos en ingles con semántica fácil de comprender, como se muestra a continuación.

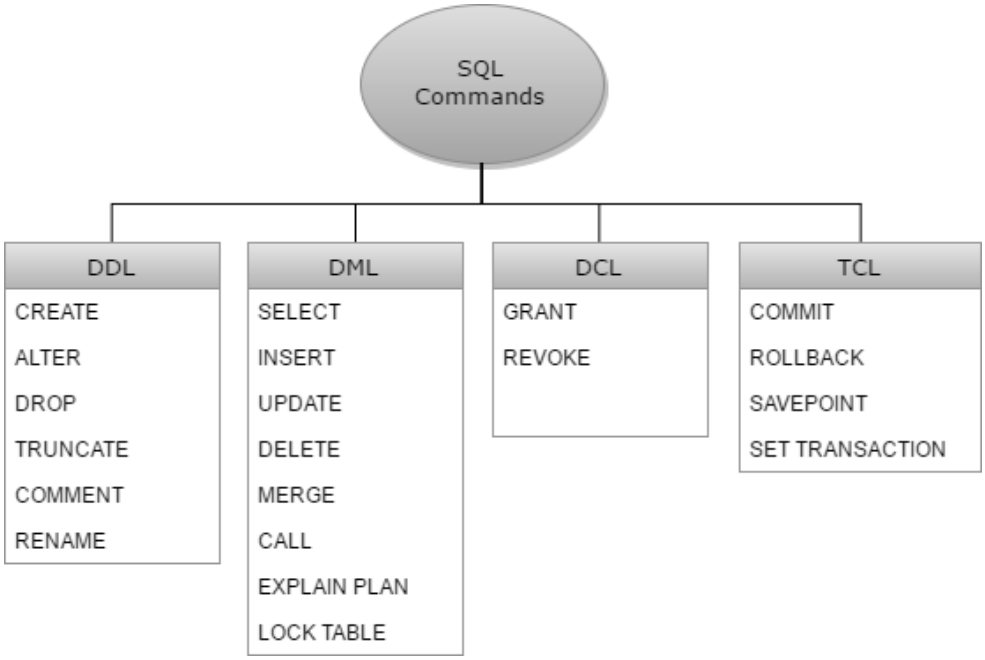

Figura 1: *Operaciones y Sentencias de SQL* (León, 2018)

Comandos básicos como los de creación de tablas:

```
CREATE TABLE IF NOT EXISTS `clientes` (
  `codigo` int(10) NOT NULL AUTO_INCREMENT COMMENT 'Código
Clave primaria',
  `nombre` varchar(40) NOT NULL COMMENT 'nombre del cliente',
  `apellido` varchar(40) NOT NULL COMMENT 'Apellidos del
cliente',
  `telefono` varchar(15) NOT NULL COMMENT 'móvil',
  `edadcte` int(3) DEFAULT NULL,
  `sexcte` char(1) NOT NULL,
  `dedicacion` text NOT NULL,
  PRIMARY KEY (`codigo`),
  UNIQUE KEY `telefono` (`telefono`),
  KEY `nombre` (`nombre`),
  FULLTEXT KEY `apellido` (`apellido`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 COMMENT='tabla de
clientes';
DE CONSULTA DE REGISTROS
```

Con el ejemplo que se muestra a continuación, se selecciona las columnas 'nombre y apellido' de **2 tablas diferentes**, esta consulta devolverán como resultado todas las filas, ya que no se tiene ninguna restricción o condicionante con *WHERE*

```
SELECT nombre, apellido FROM usuarios, clientes
```

El ejemplo siguiente selecciona la columna nombre de la tabla usuarios, aquí ha de devolverse todos los nombres de usuario donde su edad sea igual a 22.

```
SELECT nombre FROM usuarios WHERE edad = 22
```

En el ejemplo que se muestra a continuación, se ha seleccionado el nombre y apellido de los clientes que cumplan con la condición de tener 48 años o más, los que tienen 48 años también saldrán en los resultados.

```
SELECT nombre, apellido FROM empleados

WHERE edad >= 48
```

Estos comandos de consulta son muy utilizados en el modelo relacional, se pueden extender además con operaciones más complejas, para lo cual se puede agrupar condiciones para que se hagan más fáciles de entender o manipular con usa semántica clara que. En el ejemplo que se muestra a continuación se debe cumplir una de las 2 condiciones planteadas, que sí sea de Portugal o tenga más de 40 años, del resultado que se obtiene se mostrarán además los que tengan más de 10 años en la empresa como antigüedad.

```
SELECT nombre, apellidos

FROM empleados

WHERE (pais = 'Portugal' OR edad > 40) AND antiguedad > 10
```

Como se habla de un modelo relacional, se pueden dar además consultas por intersecciones entre tablas que cuentan con campos compatibles que están pensados para ser enlaces o relación entre dos tablas, se puede generar una consulta que obtenga datos de más de una tabla, pudiendo además al mismo tiempo establecer criterios sobre otras.

Como ejemplo se plantea, mostrar los clientes de un concesionario y las ventas de autos a clientes en los diversos concesionarios,

Se puede obtener una consulta que muestre los datos del cliente y de la venta.

Por ejemplo, nombre, apellido, codvehiculo y color, mediante una consulta que comúnmente en un modelo relacional es utilizada con SQL del tipo:

```
SELECT nombre, apellido, codvehiculo, color FROM CLIENTE,
VENTA WHERE CLIENTE.cifcl = VENTA.idCliente
```

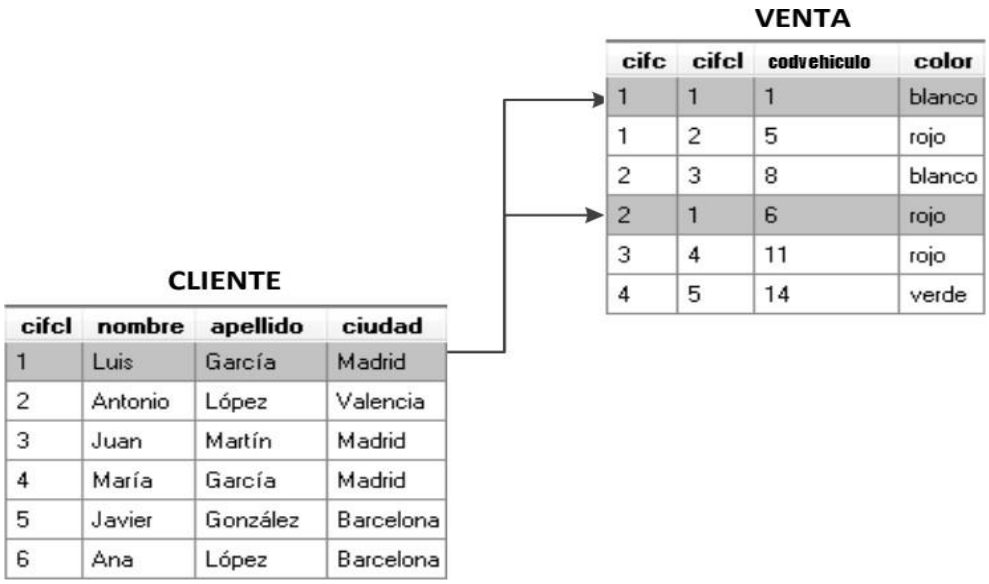

Figura 2: ***Ejemplo de Modelo Relacional con 2 Tablas***

Los comandos de eliminación permiten eliminar filas de tablas, como ejemplo directo se pretende eliminar registros de la tabla usuarios, donde el valor del campo edad sea superior a 32, ordenando los registros por la columna edad y limitando la eliminación a solamente 10 registros

```
DELETE FROM usuarios

WHERE edad >32

ORDER BY edad
```

```
LIMIT 10
```

Los comandos de actualización, permiten modificar registros de una tabla de forma general o basados en algún criterio, por ejemplo, si se quiere actualizar la tabla empleados de una empresa, se podría hacer como a continuación se detalla:

```
UPDATE empleados
SET sueldo_bruto = '70000',
prima_objetivos = '4000'
WHERE sueldo_bruto < 48000 AND sueldo_bruto > 42000
ORDER BY antiguedad DESC LIMIT 20
```

La explicación al detalle de esta consulta de actualización sería, se establece el sueldo bruto anual a 70.000 dólares y la prima de objetivos a 4.000., afectando solamente a los empleados que cobren entre 48.000 y 42.000 dólares, además se actualizarán los primeros 50 empleados por su parámetro *LIMIT* 20 ordenados de mayor a menor antigüedad en la empresa.

Las tablas poseen un identificador de registros que permite manipularlos de forma inequívoca, a estas, como hace referencia (Ricardo, 2004) se las conoce como claves que permiten consultar valores o registros de diferentes formas acelerando además en ciertos casos los procesos de búsqueda. En el modelo relacional se conoce como claves a los atributos de un campo o conjunto de campos que sirven para identificar un registro de forma única.

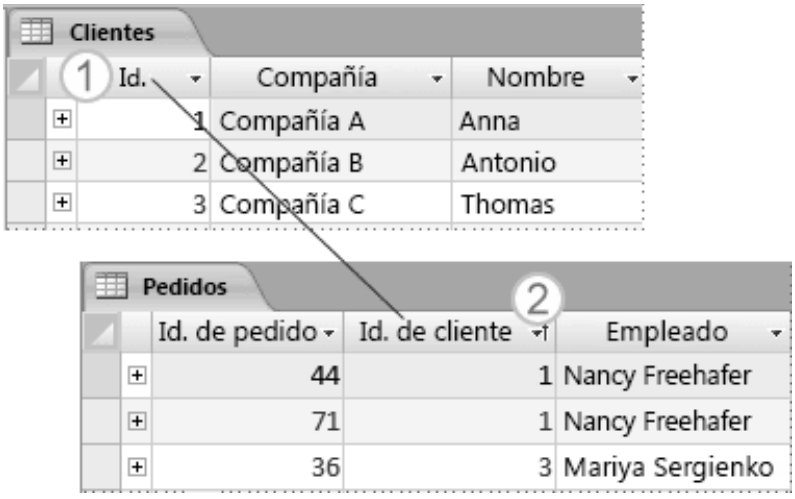

Figura 3 Ejemplo de identificación de campos claves con 2 Tablas

En el ejemplo de la imagen anterior, el campo "Id" de la tabla clientes, permite incorporarse en cada registro de la tabla Pedidos, para posteriormente ser un enlace y que exista una referencia de vínculo entre las dos tablas

DISCUSION

Las bases de datos relacionales que se basan en el lenguaje *SQL*, se diferencian de otros conceptos que persiguen un enfoque alternativo al rato de organizar sus datos. Entre los más importantes que se conocen se tienen a las **bases de datos orientadas a objetos y** los **sistemas de bases de datos basados en documentos.**

Las bases de datos orientadas a objetos (BDOO) entran en escena a finales de 1980, siendo un nuevo modelo que se fundamente en la programación orientada a objetos y permite el almacenamiento de los datos en forma de objetos. Aunque esta funcionalidad no alcanzó a lograr consolidarse, algunas de sus ideas principales se encuentran incluidas en el desarrollo de sistemas de bases de datos relacionales. El resultado de estas son extensiones objeto-relacionales que permiten el almacenamiento fácil de tipos abstractos de datos en el modelo relacional.

Con los cambios y la avanzada de la tecnología y el uso de internet con sus grandes cantidades de datos dispersados, trajo consigo la *web 2.0*, el modelo relacional nuevamente fue objeto de críticas (León, 2018). En el marco del **movimiento *NoSQL*** (*Not only SQL*), se da inicio al desarrollo de nuevos modelos alternativos como son las bases de datos

orientadas a documentos (*BDOD*), cuya finalidad era crear bases de datos de gran rendimiento y con aptitudes para aplicaciones de uso masivo de datos.

Estos dos modelos, los orientados a objetos y orientados a documentos, se diferencian del modelo relacional por la forma como se almacenan y se accede a sus datos.

Por otro lado, las bases de datos orientadas a objetos en su modelo de utilización, consideran el almacenamiento de datos como objetos, muy similares a la programación orientada a objetos, estos definen a una entidad y contiene:

Atributos necesarios para describir a la entidad,

Conexiones o relaciones entre objetos,

Funcionalidades que permiten acceder a los datos almacenados, vendrían a parecerse a los métodos.

Los SGBD orientado a objetos asignan automáticamente una ID a cada objeto, con el cual los identifica de forma única y se dirige a él usando métodos. Según (Kleppmann, 2017), esto le permite asignar ID diferentes a dos objetos que cuentan con los datos similares, con un mismo estado. Este modelo se aleja en si del relacional, ya que en el que cada registro puede identificarse a partir de sus datos con una clave primaria.

Otra de las características del modelo orientado a objetos es la representación de sus datos, ya que puede trabajarlos con **aislamiento**, pues la única forma de sus datos guardados es utilizando los métodos que se han definido con anterioridad. Esto permite la protección y seguridad de datos de cambios no permitidos.

Las estructuras de bases de datos aquí en este modelo se definen por medio de un **sistema de clases jerárquico**. Si identificamos podría expresarse parecido a la programación orientada a objetos, donde una clase permitiría definir un conjunto de objetos capaces de poseer las mismas propiedades y funcionalidades y a cada clase de objetos le subyace una definición de clases.

Aquí también se interactúa con el sistema gestor de estas bases de datos, utilizando un lenguaje de consultas inspirado en *SQL*, ***OQL*** *(Object Query Language).* Donde el resultado de una consulta con el *Object Query Language* no es, como en *SQL*, un espacio de resultados, sino una lista de todos los objetos que logran satisfacer una condición.

Algunos sistemas conocidos del modelo de *BDOO* son *ZODB, Realm y Perst*.

Las *BDOO* nos mueven a los principios de tecnología de orientación a objetos a la y por ello son muy adecuadas sobre todo en la programación de aplicaciones desarrolladas con el paradigma orientado a objetos, sin embargo, los sistemas de bases de datos de este tipo son poco frecuentes y aún carecen de madurez para el mercado.

Bases de datos mixtas objeto-relacionales, estos son sistemas de bases de datos relacionales que se han mejorado y se han reorganizado, ampliándose **con conceptos y algunas características del modelo orientado a objetos (León, 2018)** . De esta manera el modelo probado de años, esto es el relacional se amplía a tipos abstractos de datos como objetos.

Estas bases de datos permiten trabajar con tipos de datos complejos, mientras que las bases de datos relacionales pueden solamente procesar datos alfanuméricos, con los tipos de datos personalizables también se pueden gestionar archivos multimedia complejos.

Permiten derivar tipos nuevos de datos a partir de los tipos básicos ya existentes a través de constructores.

Como el *SQL* no permite crear funciones, los sistemas objeto-relacionales pueden proveer extensiones con las cuales se pueden realizar o definirse funciones de acceso a datos complejos

Luego del cambio de siglo se incluyeron extensiones o mejoras objeto-relacionales, como los tipos estructurados, en las últimas versiones del *ANSI SQL*, aunque no son soportadas por todos los SGBD (León, 2018). Los que disponen de estas extensiones son *Oracle Database*, *IBM Db2*, y *Microsoft SQL Server*.

Las bases de datos orientadas a documentos, son sistemas de almacenamiento BD, que están construidos para ofrecer prestaciones de alto rendimiento ideales el manejo de volúmenes masivos de información. Son una tecnología diferente a las bases de datos relacionales porque, cada documento puede ser almacenado sin seguir un esquema prefijado de datos (León, 2018). En ocasiones puede contener unos datos y otras veces estos pueden ser de diferente naturaleza, o inclusive pueden contener todo el contenido relativo a un documento en un mismo registro, en lugar de almacenar información en diferentes tablas como suele suceder al tener *SQL* (Pramodkumar & Fowler, 2012). Esto evitar que se hagan sentencias *JOINs* que son costosas en gran medida en materia de rendimiento y consumo de recursos. Esta tecnología de bases de datos *noSQL* son

c*lusterizables***,** por lo que nos da la posibilidad de añadirle nodos según su volumen de datos vaya creciendo. Estos nodos pueden además ser consultados al mismo tiempo.

El modelo orientado a documentos está basado en un conjunto heterogéneo de datos compuesto por documentos, a diferencia de las bases de datos relacionales, donde los datos son almacenados en tablas; (Kleppmann, 2017) *indica que, e*l conjunto de datos del modelo orientado a documentos puede ser estructurados, como archivos *XML, JSON o YAML,* o no estructurados, como *BLOB* donde se podría almacenar archivos de imagen, vídeo o audio.

En relación al almacenamiento de los documentos estructurados, estos tienen lugar por medio de pares clave/valor, donde se le es asignado un valor a cada clave. Estos vendrían a ser los atributos "es decir las claves", que no tienen nada que ver con la terminología de claves del modelo relacional.

Los datos o valores que se guardan, equivalen a cualquier tipo de información, así mismo, las matrices o listas pueden ser considerados valores.

A continuación, se muestra una plantilla para almacenar datos de un documento tipo *JSON:*

```
{
  "Codigo" :  333,
  "Apellido" : "Oviedo",
  "Nombre" : "Marcos",
  "Cedula" : " 1233445678 ",
  "Direccion" : "Juan x Marcos 564 y Mejia",
  "Ciudad" : "Babahoyo"
}
```

Varios documentos pueden ser agrupados en una *collection.* de documentos, este documento con datos personales podría ser unido a otra parte de la colección empleados.

Se puede generar consultas utilizando funciones, estas se las puede manejar con *JavaScript*.

Aquí en este modelo, **no existe esquema alguno que permita abarcar a la base de datos por completo.** En una base de datos basada en documentos no existen las formas normales ni propiedades estructurales que deben cumplirse en todos los documentos para manipular integridad de datos (Mayer-Schönberger & Cukier, 2014). Cada documento en principio, puede estar estructurado de forma diferente. En tal sentido, cuando se desarrolle aplicaciones, es recomendable crear los documentos en esquemas adecuados a la aplicación para para poder realizar consultas.

Los Sistemas de gestión de bases de datos orientados a documentos **no ofrecen los *JOIN***, estas posibilidades de hacer consultas deben programarse explícitamente (Mayer-Schönberger & Cukier, 2014), pues las relaciones, así como las inter conexiones de tablas de bases de datos en el modelo relacional, no es permitido aquí. Aunque brinda la posibilidad de añadir manualmente el ID de un documento que sirve de referencia en documentos diferentes.

Definitivamente a la hora de procesar grandes cantidades de documentos, con grandes cantidades de datos y estructuras heterogéneas y muy pocas interconexiones, las bases de datos orientadas a documentos son la mejor opción y por tales razones esto hace a este modelo de bases de datos apto para el trabajo con *Big Data*.

A continuación, se muestran algunas herramientas de bases de datos orientadas a documentos: *MongoDB y RavenDB*, *BaseX*, *CouchDB y eXist.*

Para fortalecer este análisis, se le realizó una encuesta a estudiantes que han tenido la transición o han experimentado el cambio entre bases de datos *SQL* y *NoSQL*, para lo cual se les consultó lo siguiente, y además sus respuestas fueron analizadas:

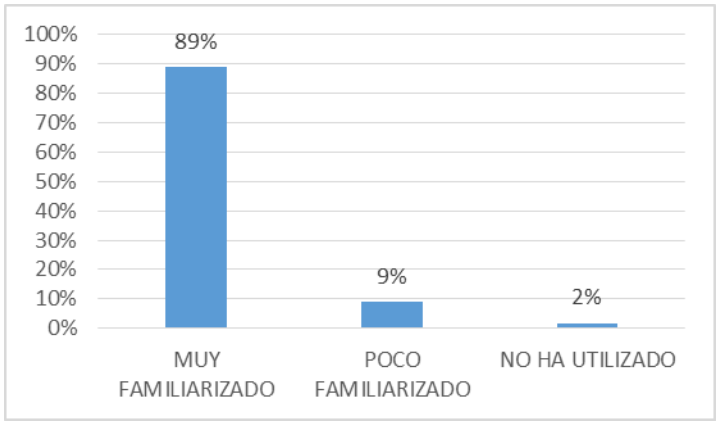

Figura 4: *Que tan familiarizado con el uso de bases de datos relacionales*

Como se puede evidenciar en la gráfica, la mayoría contesto estar muy familiarizado con el tipo de bases de datos relacionales, es decir donde se utiliza aun sentencias SQL, esto se debe a que su formación tuvo este contenido de forma regular.

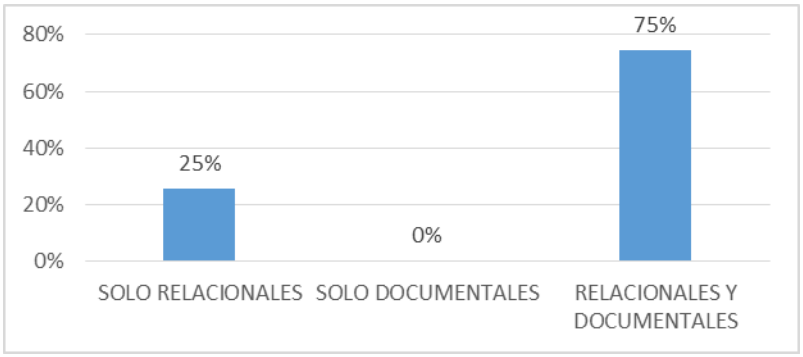

Figura 5: *Qué tipo de base de datos ha utilizado para las operaciones o soluciones que ha desarrollado durante su formación*

Se refleja claramente que la tendencia en la formación de estudiantes de ingeniería en sistemas de información, es incorporar los dos modelos y arquitecturas de bases de datos, así lo refleja su mayoría, que han realizado ya operaciones y soluciones, incorporando ambas metodologías, sin embargo, también existen quienes aún no desarrollan nada relacionado con bases de datos tipo *NoSql*, pero no significa que no las hayan experimentado, aunque sea para alguna prueba básica.

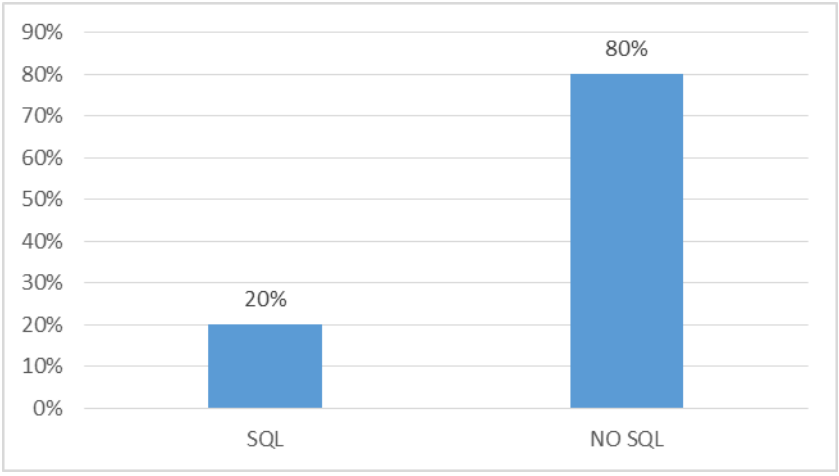

Figura 6: *Durante los contenidos prácticos de su formación, cuál de estas tecnologías considera que es menos útil*

Aparentemente lo que les ha resultado menos útil a los estudiantes consultados, ha sido *NoSql* de bases de datos orientadas a documentos, esto se debe a que no se han desarrollado tantas aplicaciones que requieran de este tipo de tecnologías, lo que se ha explotado comúnmente es el desarrollo de sistemas de información tipo *ERP*, donde comúnmente funcionan de manera perfecta las bases de datos relacionales, por lo que se considera menos útil, sin embargo, si fue creada es para un propósito específico, donde esta tecnología es más útil.

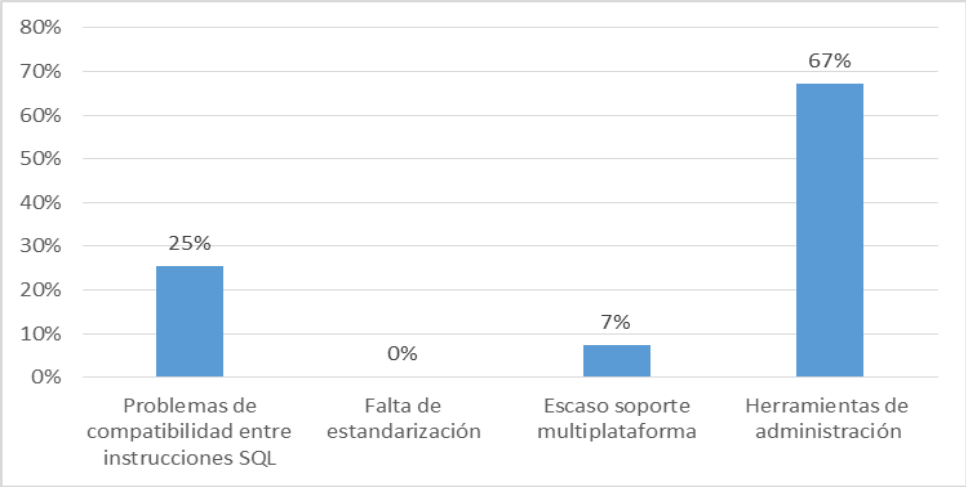

Figura 7: *Cuál de estos inconvenientes ha tenido al momento de utilizar un modelo NoSQL relacionados con su usabilidad.*

Las herramientas que se encuentran disponibles no son tan amigables con quienes están familiarizados con interfaz gráfica, suelen tener acceso de administración por consola, y se requiere conocimiento amplio de sus instrucciones para poder explotarlo a conveniencia. Esta ha sido en un 67% el inconveniente mayormente enmarcado, así como la compatibilidad de instrucciones que no se asemejan a las de *SQL*.

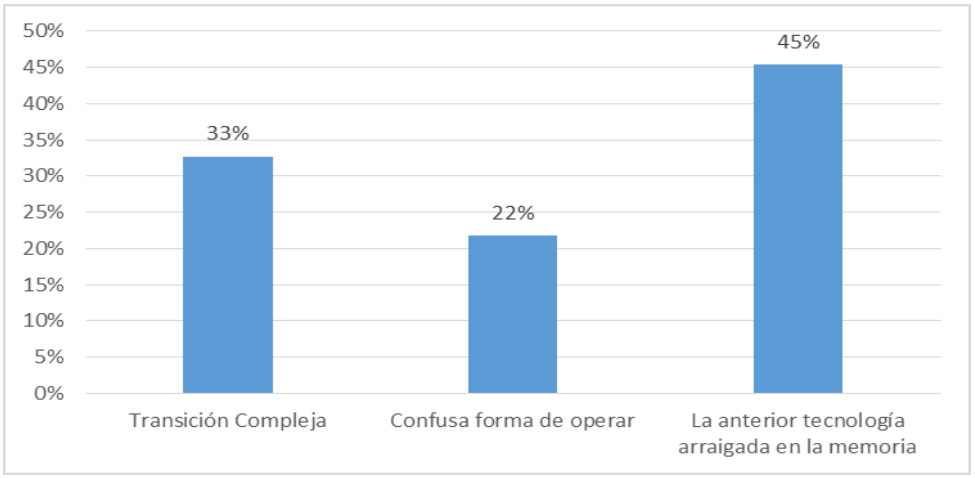

Figura 8: *Que dificultades ha tenido al momento de moverse a la tecnología NoSQL*

El tema de la costumbre a anterior tecnología aún se muestra muy apegada en los conocimientos de los estudiantes, así lo representa un 45% de los entrevistados, además de confusión y complejidad en la transición son la forma común al referirse a las dificultades que se ha tenido al momento de que ha tocado utilizar esta tecnología de bases de datos documentales.

Saber desaprender según (Vidal Ledo & Bertha, 2015), es hacer las cosas de una forma distinta a la que normalmente hacemos, muchas veces creemos que no existe otra manera de realizar algo o de solucionar alguna cosa. Y puede que, en medios de enseñanza el término desaprender se escuche totalmente fuera del objetivo de estos medios, no es así, pues también se puede aprender cosas nuevas a través de desaprender.

Desaprender se puede definir según (Vidal Ledo & Bertha, 2015), como la tendencia que tiene una persona de actualizarse por medio de un proceso que permite unir patrones de significado y catalogar experiencias. Desaprender es estar dispuesto a realizar cambios y desconocer patrones establecidos dejando a un lado la zona de confort intelectual. Reaprendes es despreciar e inclusive eliminar de una manera consciente lo que ya no sirve, lo viejo y aprender algo diferente, mejorado y más eficiente de lo que se ha realizado en años.

Desaprender no es retroceder, al contrario, es avanzar. abandonando todo aquello que ya no es provechoso, es olvidar conocimientos anticuados, borrar creencias y temores que nos

detienen. Desaprender sintetizando la opinión de (Benito & Pilar, 2018), es iniciar de nuevo, refrescar nuestra mente, llenarla de ideas nuevas, valores o cualquier otro aspecto que se requiera para ser mejores y seguir adelante. Aprender a desaprender para aprender de nuevo trae consigo alejarnos de lo fácil, lo seguro y lo previsible es así como se obtendrá buenos resultados. Si bien los conocimientos antes adquiridos fueron de utilidad en su momento, pero en un mundo que evoluciona constantemente ya no funcionan

Es importante recalcar que para los adultos este es un proceso que requiere de su voluntad, de que quieran hacerlo y se esfuercen. Para las nuevas exigencias del mercado laboral de la misma forma debe ser una decisión muy bien administrada.

CONCLUSIONES

El modelo de bases de datos relacionales ha influido mucho en el desarrollo transaccional, con su probada eficacia durante ya más de 41 años, siendo el pilar fundamental para el desarrollo de software aplicado a las empresas y a múltiples otros propósitos que se ha venido utilizando durante estas últimas 4 décadas, sin embargo, aún no se ha adaptado a las exigencias actuales de procesar grandes contenidos de documentos y de grandes volúmenes de datos para ejecución de análisis tipo Big Data, ya que estos podrían saturarse y los motores de bases de datos llegar a requerir más poder de procesamiento.

En tal sentido, son necesarios los sistemas especializados modernos, como las bases de datos desarrollados para la gestión del tipo NoSQL, ya que estas muestran una superioridad, sin embargo, no es posible dejar de lado aun al modelo relacional por completo, ya que siguen siendo el gran potencial tecnológico de almacenamiento en ámbitos corporativos comerciales protagonizados por procesamientos de datos transacciones**.**

La transición del aprendizaje del puede requerir de desaprender muchas costumbres del modelo relacional, por lo que desaprender es podrá refrescar nuestra mente, llenarla de ideas nuevas, valores o cualquier otra dinámica que se requiera para adaptarse a una nueva tecnología o forma de utilizarla como lo involucra en el modelo de bases de datos orientadas a documentos. Aprender a desaprender posiblemente permitirá obtener buenos resultados, pues en su momento los conocimientos antes adquiridos fueron de utilidad, pero en un mundo que evoluciona puede no siempre servir.

NoSql es una buena opción, y posee herramientas como *MongoDB*, que es ideal si se quiere comenzar con la gestión de bases de datos orientadas a documentos, todo esto con los cuidados debidos y un buen diseño se puede lograr estructurar y eficientes volúmenes de datos.

Si se tiene la herramienta adecuada cuando se es un desarrollador, y se encuentra en labor de migración, el usar herramientas de fácil manejo, permitirá garantizar un buen modelado de datos para lograr una manipulación más sencilla, gracias a los API que incorpora.

Como propuesta de trabajos futuros, se podrían realizar mediciones sobre el impacto de la buena gestión de las bases de datos de gran tamaño para toma de decisiones en las organizaciones.

REFERENCIAS BIBLIOGRÁFICAS

Benito, E., & Pilar, M. (2018). Revolución 4.0, Competencias, Educación y Orientación. *REVISTA DIGITAL DE INVESTIGACIÓN EN DOCENCIA UNIVERSITARIA*.

Díaz, C. A. (2019). *Programación en JAVA IV: Estructuras dinámicas - Bases de datos – Android Studio – Búsqueda y ordenamiento.* RedUsers.

Kleppmann, M. (2017). *Designing Data-Intensive Applications.* San Francisco: O'Reilly.

León, S. T. (28 de 03 de 2018). *Modelos de datos y visión conceptual de una base de datos.* Elearning, S.L.

López, C. P. (2019). *PROGRAMACIÓN DE BASES DE DATOS CON MySQL.* Independently Published.

Maya, E. (2014). *Métodos y técnicas de investigación Una propuesta ágil para la presentación de trabajos científicos en las áreas de arquitectura, urbanismo y disciplinas afines.* México: Editorial de la Facultad de Arquitectura UNAM.

Mayer-Schönberger, V., & Cukier, K. (2014). *A Revolution That Will Transform How We Live, Work, and Think.* Londres: John Murray.

Pramodkumar , J., & Fowler, M. (2012). *A Brief Guide to the Emerging World of Polyglot Persistence.* Indiana: Addison-Wesley Professional.

Ricardo, C. M. (2004). *Bases de datos.* Mexico: McGRAW-HILL INTERAMERICANA EDITORES, S.A. de C.V.

Silberschatz, A., F. Korth, H., & Sudarshan, S. (2002). *FUNDAMENTOS DE BASES DE DATOS.* Madrid: McGRAW-HILL/INTERAMERICANA DE ESPAÑA, S. A. U.

Sullivan, D. (2015). *NoSQl for Mere Mortals.* Portland, Oregon: PEARSON EDUCATION INC.

Vidal Ledo, M., & Bertha, F. (2015). Aprender, desaprender, reaprender. *Revista cubana de Educacion*.