Ciencia Digital
www.cienciadigital.org

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Análisis de impacto y medición de confiabilidad y tiempo en la migración de bases de datos Sql A Nosql.

Impact analysis and measurement of reliability and time in the migration of SQL databases to Nosql.

Patricio Navas Moya.[1], Tatiana Mayorga Soria.[2], Santiago Viteri Arias.[3] & Carlos Casa Guayta [4]

Recibido: 05-03-2017 / Revisado: 11-05-2017 Aceptado: 01-06-2018/ Publicado: 01-07-2018

DOI: https://doi.org/10.33262/cienciadigital.v2i3.138

Abstract.

The present investigation shows a previous analysis to the migration of data between relational database engines and a No SQL. To carry out this article, a financial application was developed as a scenario in which migration would be more notable to investigate the limitations and advantages of a comparative way in the basic operations of a database. The impact analysis will focus on the actual financial application since it handles large amounts of data using a relational database, migrating it to a non-relational database; additionally, the impact produced in terms of administration capacity, instruction execution times and execution in machines with few resources will be analyzed.

The application is developed in Visual Studio, the relational database used will be MySql and the migration will be done in Apache Cassandra, as far as measurements are concerned, the help of Visual Studio instructions will be used to capture the execution time, at the end, identifying advantages and disadvantages of the same application with a relational and non-relational database; it will be possible to determine the best option in terms of database for a financial application with large amounts of data.

[1] Universidad de las Fuerzas Armadas ESPE, Cotopaxi, Ecuador, mpnavas@espe.edu.ec
[2] Universidad de las Fuerzas Armadas ESPE, Cotopaxi, Ecuador, ptmayorga@espe.edu.ec
[3] Universidad de las Fuerzas Armadas ESPE, Cotopaxi , Ecuador, csviteri1@espe.edu.ec
[4] Universidad de las Fuerzas Armadas ESPE, Cotopaxi, Ecuador, cwcasa@espe.edu.ec

.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Keywords: Migration, Databases, Sql, Nosql

Resumen.

La presente investigación muestra un análisis previo a la migración de datos entre motores de bases de datos relacional y un No SQL. Para llevar a cabo este artículo se desarrolló una aplicación financiera como escenario en el cual la migración sería más notable para investigar las limitaciones y ventajas de una manera comparativa en las operaciones básicas de una base de datos. Se enfocará el análisis de impacto en la aplicación financiera real ya que maneja grandes cantidades de datos utilizando una base de datos relacional, migrándola hacia una base de datos no relacional; adicionalmente se analizará el impacto producido en cuanto a la capacidad de administración, tiempos de ejecución de instrucciones y ejecución en máquinas con pocos recursos.

La aplicación es desarrollada en Visual Studio, la base de datos relacional utilizada será MySql y la migración se la realizará en Apache Cassandra, en lo que respecta a las mediciones se utilizará la ayuda de las instrucciones de Visual Studio para capturar el tiempo de ejecución, al finalizar, identificando ventajas y desventajas de la misma aplicación con base de datos relacional y no relacional; se logrará determinar la mejor opción en cuanto a base de datos para una aplicación financiera con grandes cantidades de datos.

Palabras Claves: Migración, Bases De Datos, Sql, Nosql

Introducción .

El bien más importante que tiene toda organización es la información, y para esto se debe tener actualizado donde se lo va a almacenar para garantizar la integridad de la misma, para lograr este objetivo se requiere procesar y almacenar grandes cantidades de datos de forma rápida y eficiente por lo que las bases de datos necesitan estar en constante evolución. [1]La demanda de alto rendimiento en la lectura y escritura, hace que las bases de datos relacionales enfrenten muchos nuevos desafíos. Utilizar la base de datos relacional para almacenar y consultar datos de usuario dinámicas ha resultado ser insuficiente.[1]

El almacenamiento de históricos se ha realizado desde tiempo atrás, fundamentalmente con tecnologías de bases de datos relacionales, sin embargo, el modelo relacional introduce limitantes derivadas de su propia concepción que coadyuvan a un comportamiento ineficiente cuando los sistemas crecen, que se manifiesta con la reducción significativa del rendimiento y la disponibilidad para la gestión de series de tiempo. [2]

Los sistemas que manejan grandes cantidades de datos deben seguir características como escalabilidad, fiabilidad, durabilidad, tiempo de respuesta, interfaz de consulta, estructura

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

de los datos almacenados (o carencia de la misma) y esquemas de particionamiento de datos.[2]

La demanda de los usuarios y la llegada de nuevas tecnologías que impulsan el uso de las tecnologías NoSQL; con millones de usuarios requiriendo sistemas completos, pero sobre todo veloces, que principalmente realizan en almacenamientos y búsqueda de la información, hace necesario un paradigma de bases de datos orientado a la velocidad.[3][4]

Estado del arte.

Una base de datos es un conjunto de información estructurada en registros y almacenada es un soporte electrónico legible desde un ordenador. Cada registro constituye una unidad autónoma de información que puede estar a su vez estructurada en diferentes campos o tipos de datos que se recogen en dicha base de datos. [5][6]

El almacenamiento de datos se ha venido desarrollando de manera convencional por bases de datos relacionales desde 1970 Edgar Fram Codd, las bases de datos relacionales son una colección de datos almacenados en forma de tablas relacionadas.[7]

Modelo relacional.

Es un modelo de organización y gestión de bases de datos el cual tiene como premisa el almacenamiento de datos en tablas compuestas por filas y columnas tipo hojas de cálculos, las filas que son conocidas como tuplas y las columnas o campos. [5]

Se distingue de otros modelos por ser más comprensible que cualquier otro, se basa en la lógica de predicados para establecer relaciones entre distintos datos, en la actualidad es una solución para la creciente variedad de los datos. [5][8]

MySql.

Es un sistema gestor de bases de datos relacional (RDBMS, Sistemas Gestores de Bases de Datos Relacional) de código abierto, el cual su base fundamental es el lenguaje de consultas estructurado (SQL). Entre las características más importantes es que se ejecuta en todas las plataformas incluyendo Windows, Linux y Unix, tiene muchas cualidades, a las que suman la facilidad para desarrollo de aplicaciones de escritorio, pero lo más relevante son las aplicaciones web, en la cual se potencia con herramientas como Apache que es el servidor web.[3][9]

Originalmente Mysql fue desarrollado en la compañía sueca MySQL AB, pero en el año 2008 fue adquirida por Oracle, sin embargo, los desarrolladores pueden seguir utilizando el MySQL bajo la Licencia Pública General (GPL),[3] pero las empresas deben adquirir la licencia comercial a Oracle.[10] Una vez que el código de MySQL es privativo aparecieron los conocidos hijos o llamados también como FORKS, en los que se incluyen a Drizzle

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

cuyo código abierto se baso en MySQL 6.0. MariaDB que es el considerado reemplazo popular "drop-in" desarrollado en la comunidad MySQL el mismo que utiliza las API y los comandos de MySQL.[9]

EL Percona Server XtraDB que es una versión mejorada de MySQL conocido por su escalabilidad horizontal.

Modelo no relacional o Bases de datos NoSQL.

Son bases de datos no relacionales para información o datos sin esquemas y que por sus características su desempeño es escalable, particularmente son conocidas por la facilidad en el desarrollo baja latencia y su alta fiabilidad. Utilizan una variedad de modelos de datos, como los almacenes de valor clave en memorias de gráficos de documentos y columnas en fin todo lo que tiene que ver con datos documentales. [7][11][12]

Los sistemas de bases de datos de este tipo son las más comunes en aplicaciones web y para app móviles que requieren de gran almacenamiento y que sean eficientes a la respuesta, son más fáciles de escalar que una base de datos SQL.[3]

Cassandra.

Es una base de datos NoSQL de alto rendimiento, tolerante a fallos y escalable, tiene la característica denominada Column-Family. Cassandra combina los beneficios de Google Bigtable y Amazon Dynamo que esto no ocurre en las bases de datos relacionales.[3][10]

Este Motor de bases de datos se lo puede encontrar en algunas empresas tales como: Cisco, Rackspace, Netflix, Twitter, Urban Airship, entre otras.[4]

Características.

Esta diseñada para el almacenamiento de una gran cantidad de datos estructurados y esta albergado dentro del licenciamiento de Apache. Tiene la propiedad de escalar tanto linealmente como elásticamente. El mejor rendimiento se obtiene conforme se aumentan los nodos en los clusters. Como cualquier base de datos soporta la propiedad ACID(Atomicidad, Aislamiento, Consistencia, Aislamiento, Durabilidad) y tiene una velocidad de ingreso de información altamente rápido. Soporta una fácil de distribución de datos mediante replicado a través de varios centros de datos. Fue desarrollado en Java, el cluster más grande que se tiene en estadística tiene 300TB de datos distribuidos en unas 400 máquinas.[3][4]

Relacional vs. NoSQL. El modelo relacional permite tener una gran cantidad de tablas y poder interrelacionar entre ellas de tal manera que puedan acceder a la información mucho más rápido, mientras que en las NoSQL no se puede hacer ya que es solo un documento.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

El modelo NoSQL permita una escalabilidad horizontal y elástica mientras que en el relacional esto es más rígido ya que crece en registros y serian de forma vertical.

Experimentos y resultados.

Para una adecuada interpretación de resultados se plantea una serie de pruebas de rendimiento entre los dos motores de bases de datos y como estos pueden responder a las operaciones previo a la migración de la información. Se desarrollo dos aplicaciones en Visual Studio, con la finalidad de medir el tiempo de respuesta en cada una de las bases de datos, de igual manera se planteo 3 veces para un distinto tamaño.

Se propone la inserción y consukta de de un conjunto de datos que fueron en este orden 500, 1000, y 1500 registros que para el caso de Cassandra se lo tomo como columnas/documentos, y se midió el promedio del tiempo de ejecución para cada caso plateado.

Para la realización de las operaciones de consulta en las bases de datos de MySQL se tomo en cuenta los siguientes campos:

Id_cliente, id_direccion, numer_cuenta, id_estado, id_tipo_cuenta, id_transaccion, id_tipo_transaccion.

Para la base de datos de Cassandra se tomó en cuenta los siguientes atributos:

Ids y los índices: tipotransaccion e idcuenta para la tabla de controltransaccion.

Primero prueba se realizó la inserccion de datos, para la cual se tomó un conjunto de 500, 1.000 y 5.000 columnas/documentos, después se utilizó un comando apropiado para insertar en una base de datos vacía, después se tomó el tiempo de cuanto se demoró la operación, estas operaciones se repitieron 3 veces y al final se promediaron los 3 tiempos resultantes.

Para la prueba en Mysql, se utilizó en siguiente comando para insertar datos:

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

```
INSERT INTO
(`CLIENTE`
(`ID_CLIENTE`,
`CEDULA_CLIENTE`,
`ID_DIRECCION`,
`NOMBRES_CLIENTE`,
`APELLIDOS_CLIENTE`)
VALUES
(1, '1693042073999', 34,
'ANNE',
'GILMORE'),(..),(..),….,(..);
```

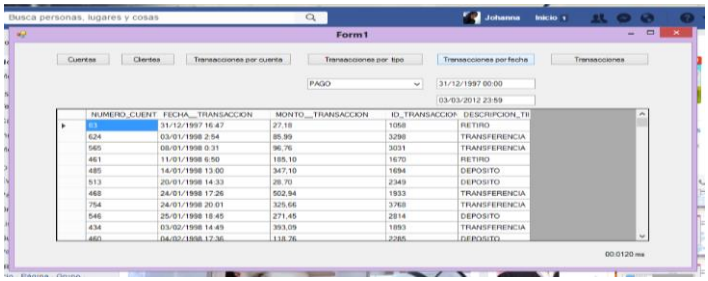
Tabla 1. Resultados de la operación de inserción en Mysql

	TIEMPO EN SEGUNDOS			
REGISTROS	**Prueba 1**	**Prueba 2**	**Prueba 3**	**Promedio**
500	6.324	5.969	6.112	**6,135**
1000	12.158	11.947	11.785	**11,963**
5000	45.857	43.941	46.194	**45,336**

Elaborado por: Grupo de Investigación.

La tabla 1 muestra el tiempo resultante en segundos de las 3 pruebas, incluyendo el promedio final de la operación de insertar registros en MySQL.

Figura 1. Pantalla de inserción de datos en Mysql



Elaborado por: Grupo de Investigación.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Para el ingreso de información en la base de datos de Mysql se realizó un front end en Visual Studio el cual ayudo en medir los tiempos en milisegundos ya que la propia base de datos no tenía esa propiedad.

Para la prueba en Cassandra se debió ejecutar el comando de inserción uno por uno, se utilizó el siguiente comando para insertar datos ejecutados desde la aplicación:

SESSION.EXECUTE("INSERT INTO CONTROLCUENTA (IDCUENTA, ESTADOCUENTA,TIPOCUENTA,IDCLIENTE,SALDOCUENTA,DETALLECUENTA) VALUES(200, TRUE, 'DEBITO', 199, 1.74, 'ULTRICES. VIVAMUS RHONCUS. DONEC EST.');

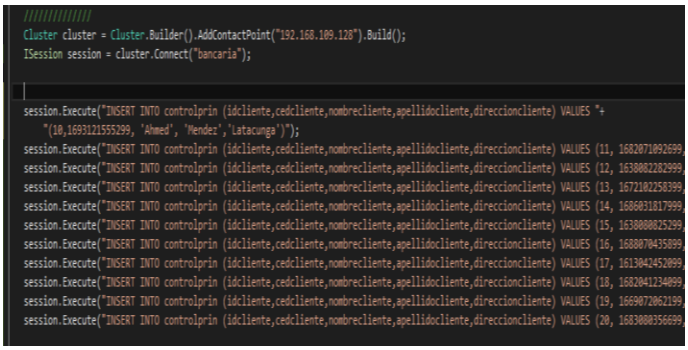
Tabla 2. Resultados de la operación inserción en Cassandra.

	TIEMPO EN SEGUNDOS			
REGISTROS	**Prueba 1**	**Prueba 2**	**Prueba 3**	**Promedio**
500	2.005	2.010	2.047	**2.05**
1000	4.265	4.456	4.624	**4.448**
5000	23.534	23.113	22.846	**23.164**

Elaborado por: Grupo de Investigación.

En la tabla 2 se obtuvo el tiempo resultante en segundos de las 3 pruebas planteadas incluyendo el promedio final de la operación de insertar en Cassandra.

Figura 2. Comandos para consultas de inserción en Cassandra.



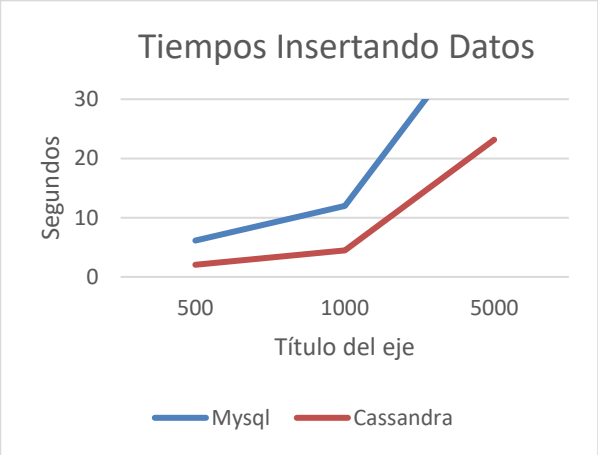
Elaborado por: Grupo de Investigación.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

En la figura anterior muestra la secuencia de ingreso de datos en la consola de Cassandra, para cumplir con lo propuesto dentro de la investigación planteada-

Para las operaciones de inserción los resultados se muestran en la Figura 1. Existe una gran diferencia en tiempos promedio entre las dos bases de datos, por lo tanto, los resultados obtenidos favorecieron a Cassandra.

Figura 3. Resultado de las pruebas de inserción



Elaborado por: Grupo de Investigación.

En la segunda prueba se realizó la función de consultar, para la cual se tomaron los mismos datos ya insertados en la primera prueba.

Para la prueba en Mysql, se utilizó en siguiente comando para consultar datos:

```
SELECT C.ID_CLIENTE,C.CEDULA_CLIENTE,
        CONCAT(
CONCAT(C.NOMBRES_CLIENTE,'                ),
C.APELLIDOS_CLIENTE) AS CLIENTE,
        CONCAT(
CONCAT(D.CIUDAD_DIRECCION,'       -       '),
D.PROVINCIA_DIRECCION)                    AS
DIRECCION,D.TELEFONO1_DIRECCION       AS
TELEFONO1,D.TELEFONO2_DIRECCION       AS
TELEFONO2
        FROM CLIENTE C,DIRECCION D WHERE
C.ID_DIRECCION=D.ID_DIRECCION
    ORDER BY C.ID_CLIENTE;
```

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Tabla 3. Resultados de la operación de MySQL

REGISTROS	TIEMPO
500	0.0035
1000	0.0048
5000	0.0080

Elaborado por: Grupo de Investigación.

En la tabla 3 se puede observar la consulta de los registros que para este caso los tiempos estuvieron dados en milisegundos, por lo que se requiere de una alternativa de medición.

Para la prueba en Cassandra, se utilizó en siguiente comando para consultar datos:

```
ROWSET RESULT = SESSION.EXECUTE("SELECT * FROM CONTROLPRIN LIMIT 500");
```

Usando este código con la cláusula (limit) la base de datos únicamente nos devuelve el limite de resultados establecidos

Tabla 4. Resultados de la operación consultar en Cassandra

REGISTROS	TIEMPO
500	0.0015
1000	0.0015
5000	0.0063

Elaborado por: Grupo de Investigación.

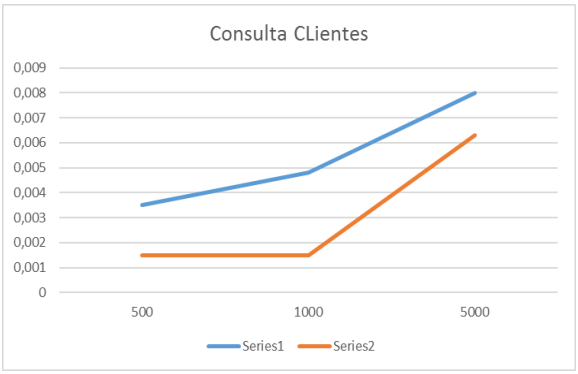
En la table anterior muestra los tiempos que muestra el motor de Cassandra que de igual manera esta dado en milisegundos notándose que son mucho más bajos que con los de

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

MySQL lo que sugiere mayor y mejor eficiencia.

.

Figura 4. Resultados de las pruebas de consulta de Clientes



Elaborado por: Grupo de Investigación.

Como se puede observar en la figura anterior los tiempos son muy eficientes, pero siempre la base de datos NoSQL son mucho mejores por lo que se puede concluir que siempre serán mejores que las relacionales.

En la tercera prueba se realizó la función de consultar en la tabla de transacciones que es la que contiene más campos, para la cual se tomaron los mismos datos ya insertados en la primera prueba. Es decir que tendremos 5000 registros originalmente, pero para el caso del experimento vamos a proceder en el mismo orden de inserción lo que concluirá con 500, 1000 y 5000 registros en MySQL y los mismos valores, pero con columnas/documentos en Cassandra se realizó una consulta.

Para la prueba en Mysql, se utilizó en siguiente comando para actualizar datos:

SELECT * FROM TRANSACCION;

Tabla 5. Resultados de la operación consulta transacciones en MySQL

REGISTROS	TIEMPO
500	0.0060
1000	0.0120
5000	0.0376

Elaborado por: Grupo de Investigación.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

La tabla anterior muestra los tiempos resultantes en segundos en la operación de verificar todos los registros que fueron ingresados, y que son bajos para el número de atributos que tienen y el tipo de datos del mismo.

Para la prueba en Cassandra, se utilizó en siguiente comando para actualizar datos:

```
ROWSET RESULT = SESSION.EXECUTE("SELECT * FROM CONTROLTRANSACCION LIMIT 500");
```

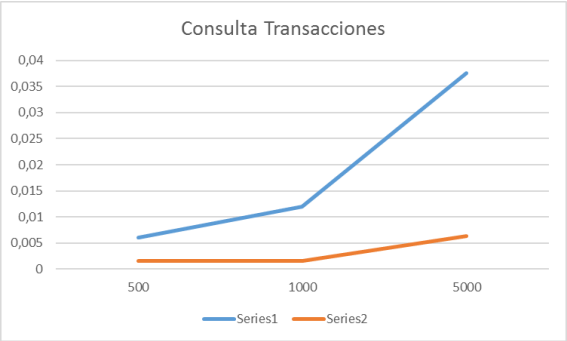
Tabla 6. Resultados de la operación consulta de transacciones en Cassandra

REGISTROS	TIEMPO
500	0.0015
1000	0.0020
5000	0.0070

Elaborado por: Grupo de Investigación.

Los resultados al igual que en las operaciones anteriores muestran ser muy bajos pero lo que más llama la atención es que guarda relación de acuerdo al número de registros que se tienen dentro de las bases de datos.

Figura 5. Resultados de la consulta de transacciones



Elaborado por: Grupo de Investigación.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Para las operaciones de consulta de las transacciones los resultados se muestran en la Figura anterior. Existe una gran diferencia en tiempos promedio entre las dos bases de datos, por lo tanto, los resultados obtenidos favorecieron a Cassandra.

De forma general los resultados de todas las operaciones planteadas previa a una migración sugieren que Cassandra es más rápido que Mysql, tanto para consultas de inserción y consulta de datos. Estos resultados fueron razonables, ya que estas pruebas se realización en un computador sin mayor prestación como un servidor, se tomó en cuenta a un computador de escritorio Core i7 con 8Gb de memoria RAM y un disco duro de 1Tb .

Conclusiones.

- En este trabajo se analizó la diferencia de respuesta que ofrece una base de datos Sql a una noSql , cuando se las usa en una aplicación, debido a que comúnmente las aplicaciones se las realiza únicamente con bases de datos Sql.

- En las consultas realizadas, los tiempos de respuesta hacia la aplicación que se registraron demuestran claramente que Cassandra(noSql) al tener un código más simplificado realiza las consultas mucho más rápido que Mysql(Sql).

- En el caso de codificación en la aplicación, igualmente Cassandra presenta un código menor respecto al código necesario para poder conectar la aplicación con la base de datos.

- En conclusión, Cassandra demostró que una base de datos noSql es más eficiente con respecto a tiempos de respuesta y líneas de código en una aplicación, a diferencia de una base de datos Sql.

Referencias bibliográficas.

S. Ag, “Sistemas para,” *Micro*, vol. 13, pp. 17–28, 2009.

A. C. Romero, J. S. G. Sanabria, and M. C. Cuervo, “Utilidad y funcionamiento de las bases de datos NoSQL,” *Rev. Fac. Ing. UPTC*, vol. 21, no. 33, pp. 21–32, 2012.

M. Giridhara Gopalan, C. Prasanna, Y. Srihari Krishna, B. Shanthini, and A. Arulkumar, “MYSQL to cassandra conversion engine,” *Proc. 2017 3rd IEEE Int. Conf. Sensing, Signal Process. Secur. ICSSS 2017*, pp. 503–508, 2017.

D. J. Dean, P. Wang, X. Gu, W. Enck, and G. Jin, “Automatic server hang bug diagnosis:

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Feasible reality or pipe dream?," *Proc. - IEEE Int. Conf. Auton. Comput. ICAC 2015*, pp. 127–132, 2015.

M. Franklin, A. Halevy, and D. Maier, "From Databases to Dataspaces: A New Abstraction for Information Management," *ACM SIGMOD Rec.*, vol. 34, no. 4, pp. 27–33, 2005.

D. Ramakanth and K. Vinod, "SQL Injection - Database Attack Revolution And Prevention," *J. Int. Commer. Law Technol.*, vol. 6, no. 4, pp. 224–231, 2011.

Y. Li and S. Manoharan, "A performance comparison of SQL and NoSQL databases," *IEEE Pacific RIM Conf. Commun. Comput. Signal Process. - Proc.*, no. August 2013, pp. 15–19, 2013.

C. A. L. Pena, "Análisis De Las Bases De Datos Nosql Como Alternativa a Las Bases De Datos Sql," p. 58, 2012.

D. J. Dean, H. Nguyen, P. Wang, X. Gu, A. Sailer, and A. Kochut, "PerfCompass: Online Performance Anomaly Fault Localization and Inference in Infrastructure-as-a-Service Clouds," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 6, pp. 1742–1755, 2016.

T. D. Le *et al.*, "EPC information services with No-SQL datastore for the Internet of hings," *2014 IEEE Int. Conf. RFID, IEEE RFID 2014*, pp. 47–54, 2014.

A. Rodríguez Pérez, D. Rodríguez Hernández, and E. Elizabeth Martínez, "Selección de base de datos NoSQL para almacenamiento de históricos en sistemas de supervisión," *Rev. Cuba. Ciencias Informáticas*, vol. 10, no. 3, pp. 85–97, 2016.

A. C. M. Antaño, J. M. M. Castro, and R. E. C. Valencia, "Migracion de Bases de Datos SQL a NoSQL," *Rev. Tlamati*, vol. Especial 3, no. February 2015, pp. 144–148, 2014.

ISSN: 2602-8085
Vol. 2, N°3, p. 74-87, Julio - Septiembre, 2018

Para citar el artículo indexado.

Navas P., Mayorga T., Viteri S. & Casa C. . (2018). Análisis de impacto y medición de confiabilidad y tiempo en la migración de bases de datos Sql a Nosql. *Revista electrónica Ciencia Digital* ***2***(3), 74-87. Recuperado desde: http://cienciadigital.org/revistacienciadigital2/index.php/CienciaDigital/article/view/138/123