DWR2NoSQL: Herramienta para transformar *Data Warehouse*s Relacionales a *Data Warehouse*s *NoSQL*.

DWR2NoSQL: Tool to transform Data Warehouses Relational to Data Warehouses NoSQL.

Francisco Javier Cartujano Escobar* (1).
Tecnológico Nacional de México / Instituto Tecnológico de Zacatepec.
francisco.ce@itzacatepec.edu.mx.

Leticia Santa Olalla Ocampo (2). Tecnológico Nacional de México / Instituto Tecnológico de Zacatepec.
leticia.so@itzacatepec.edu.mx.

José Pedro Aragón Hernández (3). Tecnológico Nacional de México / Instituto Tecnológico de Zacatepec.
jpedroah@live.com.mx.

María Fernanda Silva Reyna (4). Tecnológico Nacional de México / Instituto Tecnológico de Zacatepec.
mafersilva16.ms@gmail.com.

*corresponding author.

Artículo recibido en junio 14, 2019; aceptado en julio 01, 2019.

Resumen.

Una de las tecnologías más importantes en el área de los sistemas de información, es la tecnología referente a los Data Warehouses, también conocidos como almacenes de datos o bodegas de datos. Como es sabido, un Data Warehouse es una enorme base de datos con datos históricos, generalmente de lectura, cuyo propósito principal es apoyar el proceso de toma de decisiones en las organizaciones. El modelo relacional ha sido el modelo de datos tradicionalmente usado para implementar la base de datos de un Data Warehouse, pero debido a que hoy en día se generan grandes cantidades de datos que demandan gran capacidad de almacenamiento y de computo, dichos Data Warehouses relacionales han sido superados. Por esta razón, se hace necesario que nuevos paradigmas para implementación de Data Warehouses sean explorados. La tecnología de bases de datos NoSQL orientada a documentos se presenta como una opción a esta problemática. En este artículo, se describe el desarrollo y funcionalidad de una herramienta computacional, que ha sido bautizada con el nombre de DWR2NoSQL, cuyo propósito principal es transformar de manera automática un Data Warehouse relacional a un Data Warehouse orientado a documentos. Esta herramienta computacional forma parte de los resultados de un proyecto de investigación financiado por el Tecnológico Nacional de México.

Palabras clave: Almacén de Datos, Bases de datos *NoSQL*, Transformación automática de esquemas.

Abstract.

Data warehouses is one of the most important technology of information systems. As it is known, a Data Warehouse is huge database with historical data, mainly for reading processes, whose principal purpose is to support the decision-making process in organizations. Traditionally, the relational data model has been the model used to implement the data warehouse database but due to the large amounts of data currently being generated, which demand a large storage and computing capacity, such relational Data Warehouses have been overcome. For this

reason, it is necessary that new paradigms for the implementation of Data Warehouses be explored. The NoSQL document-oriented database technology is an option to this problem. In this paper, we describe the development and functionality of a software, which has been named DWR2NoSQL, whose main purpose is to automatically transform a relational Data Warehouse to a document-oriented Data Warehouse. DWR2NoSQL is part of the results of a research project funded by Tecnológico Nacional de México.

Keywords: Data Warehouse, NoSQL Database, Schemes Automatic Transformation.

1. Introducción.

Una de las tecnologías computacionales mayormente utilizada en el apoyo a la toma de decisiones son los *Data Warehouses*, cuyo término en español es almacén de datos o también bodega de datos, y es como su nombre lo dice, el contenedor de datos que permite a las corporaciones almacenar toda lo información que requieran para apoyar su toma de decisiones.

Tradicionalmente los *Data Warehouse* han sido modelados y operados con tecnología relacional, pero debido a que actualmente hay una generación excesiva de datos y una gran variedad de los mismos, dichos *Data Warehouse*s tradicionales enfrentan problemas para satisfacer la capacidad y velocidad de procesamiento que demandan esa ola masiva de datos. La tecnología de bases de datos *NoSQL* se presenta como una opción a esta problemática.

Una característica muy importante de las bases de datos *NoSQL* es la capacidad de escalarse horizontalmente, tanto en almacenamiento como en procesamiento. Y es debido a esta característica que los sistemas *NoSQL* ofrecen ventajas sobre los sistemas relacionales, particularmente en lo que se refiere a los *Data Warehouse*s, ya que estos implementados mediante bases de datos *NoSQL* no tendrían problemas de escalarse horizontalmente al momento de requerir más espacio o mayor poder de cómputo.

En particular las bases de datos *NoSQL* Orientadas a documentos se presentan como una de las opciones más viables para implementar un *Data Warehouse* no relacional.

El presente artículo tiene como objetivo principal describir el desarrollo y utilización de una herramienta computacional que permite transformar de manera automática cualquier *Data Warehouse* relacional implementado bajo el modelo de estrella a diferentes modelos equivalentes *NoSQL* orientados a documentos. La herramienta computacional desarrollada ha sido bautizada con el nombre de *DWR2NoSQL*.

La herramienta descrita en este artículo forma parte de los resultados obtenidos en el desarrollo de un proyecto autorizado y financiado por el Tecnológico Nacional de México (TecNM).

Antecedentes.

A la fecha se han realizado pocos trabajos relacionados a la problemática expuesta. En los siguientes párrafos se describen algunos de dichos trabajos.

En (Yangui et al., 2016) se presenta un conjunto de reglas de transformación del esquema de estrella de un *Data Warehouse* a un modelo de datos *NoSQL* orientado a documentos, pero considera solamente una sola posible opción de transformación.

(Chevalier et al., 2015) presenta un esquema de transformación de la malla de datos de un *Data Warehouse* relacional a una malla de datos *NoSQL* orientada a documentos, al igual que (Yangui et al., 2016), considera solamente una sola posible opción de transformación.

Por su parte en (Rocha, 2015) se describe un marco de trabajo para transformar bases de datos relacionales a bases de datos *NoSQL* orientada a documentos. Implementa una herramienta que permite migrar una base de datos relacional

a una base de datos orientada a documentos. Dicho trabajo no contempla la transformación de los modelos específicos de un *Data Warehouse* (estrella, copo de nieve, constelación de hechos)

Otros trabajos como (Dehdouh, 2015) y (Dehdouh, 2014) describen la transformación de un *Data Warehouse* relacional a un modelo *NoSQL* orientado a columnas, es decir un modelo *NoSQL* diferente al que nosotros proponemos.

2. Métodos.

2.1 Fundamento teórico.

Un componente esencial de un *Data Warehouse* lo constituye una enorme base de datos, ejecutando principalmente procesos de consulta y diseñada tradicionalmente bajo el modelo de datos relacional.

A nivel conceptual, la base de datos de un *Data Warehouse*, se diseña mediante un modelo multidimensional, en el cual existen dos elementos esenciales (Kimball, 2008):

a) Dimensiones: Son las entidades bajo las cuales se desea analizar la información, por ejemplo: clientes, productos, sucursales. Una dimensión muy importante y casi presente en todo modelo multidimensional, es la dimensión tiempo, con la finalidad de analizar por diferentes periodos de tiempo.

b) Hechos: Son los datos que se desean analizar, normalmente son valores numéricos, ya que un procesamiento común sobre ellos es sumarlos, promediarlos, obtener el máximo, el mínimo, entre otros. Ejemplos de hechos son: cantidad vendida por vendedor, existencia en el almacén de cierto producto.

A nivel lógico, existen tres modelos de datos relacionales para representar el modelo multidimensional de un *Data Warehouse*: a) modelo de estrella, b) modelo de copo de nieve y c) constelación de hechos (Mishra 2008). El más utilizado de ellos es el modelo de estrella, por lo que en este artículo es el que se referencia. En este modelo existe una sola tabla que almacena a los hechos y varias tablas dimensiones no normalizadas. Todas las tablas dimensiones se relacionan a la tabla de hechos, de tal forma que la tabla de hechos tiene una llave compuesta por las llaves heredadas de todas las tablas dimensiones. Una instancia de este modelo es mostrada en la figura 1.

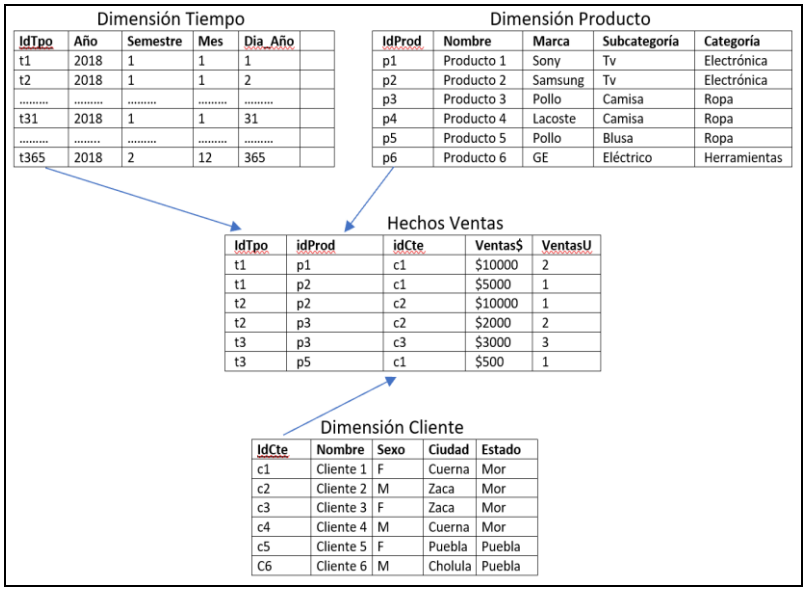


Figura 1. Instancia de un *Data Warehouse* relacional bajo el modelo de estrella.

Actualmente, existen 4 tipos de modelos de bases de datos *NoSQL*. Uno de ellos es el modelo soportado por las bases de datos orientadas a documentos que tienen las siguientes características:

- Almacenan los datos en documentos (en su forma más común: texto simple).
- Los documentos en su forma más simple están compuestos por duplas de clave-valor y pueden tener cualquier número de duplas sin necesidad de que los documentos tengan la misma estructura.
- Una colección es un conjunto de documentos. Los documentos dentro de una colección pueden tener estructuras diferentes.
- Haciendo alusión a las bases de datos relacionales, una colección sería una tabla, los documentos serían las tuplas, y las duplas clave-valor sería el campo con su respectivo valor.

2.2 Análisis.

Se ha realizado un análisis de las posibles formas en que un Data Warehouse puede ser modelado mediante bases de datos NoSQL orientadas a documentos. En este artículo se describen cuatro posibles representaciones y únicamente considera el modelo de estrella, pero las reglas establecidas para este modelo podrían sentar las bases para transformar los modelos de copo de nieve y de constelación de hechos. Para cada representación propuesta se establece el algoritmo, es decir el conjunto de reglas, para mapear un modelo de estrella relacional a cada modelo multidimensional NoSQL orientado a documentos generado por esta investigación.

2.2.1 Modelo Aglomerado.

En este modelo toda la información referente a un hecho se encuentra en un documento con atributos simples. Para generar este modelo aplicar el algoritmo 1 de transformación:

Algoritmo 1. Transformación de Estrella Relacional a Modelo Aglomerado *NoSQL*.

a) Representar a todo el modelo multidimensional por una colección *C*.
b) Para cada tupla de la tabla de hechos, denotada por T^H realizar lo siguiente:
   i. Crear un documento, denotado por D^H.
   ii. Para cada dimensión *i*, con llave K_i, referenciada en T^H, anidar en el documento D^H todos los pares <atributo, valor> de la tupla en la tabla dimensión *i* cuya llave es K_i
   iii. Anidar en D^H cada hecho contenido T^H. Cada hecho es anidado en D^H por el par <atributo,valor>
   iv. Anidar D^H en C.

2.2.2 Modelo por Subdocumentos.

Una de estas posibles representaciones la hemos llamado “Modelado por subdocumentos”. En este modelo se aplica el algoritmo 2 de transformación:

Algoritmo 2. Transformación de Estrella Relacional a Modelo por subdocumentos *NoSQL*

a) Representar a todo el modelo multidimensional por una colección *C*.
b) Para cada tupla de la tabla de hechos, denotada por T^H realizar lo siguiente:
   i. Crear un documento, denotado por D^H
   ii. Crear un atributo cuyo nombre es igual al nombre de la tabla de hechos, denotemos a dicho atributo por A^H. El valor de A^H es un documento que anida a todos los hechos de T^H, cada hecho será representado con el par <atributo,valor> anidado en A^H. Anidar a A^H dentro de D^H.
   iii. Para cada dimensión *i* referenciada en T^H con llave K_i crear un atributo cuyo nombre es igual al nombre de la tabla dimensión, denotemos a dicho atributo A^D. El valor de A^D es un documento que anida a todos los pares <atributo, valo> de la tupla en la dimensión *i* cuya llave es *Ki*. Anidar a A^D dentro de D^H.

iv. Anidar D^H en C.

2.2.3 Modelo de Arreglos.

En este modelo de arreglos, todas las tuplas de una tabla en particular del modelo se almacenan en un arreglo, anidado dentro de un documento que representa a dicha tabla en particular, de esta forma, la colección tiene un documento por cada tabla del modelo multidimensional. Este modelo orientado a documentos se genera de acuerdo al algoritmo 3 de transformación

Algoritmo 3. Transformación de Estrella Relacional a Modelo de Arreglos *NoSQL*.

a) Representar a todo el modelo multidimensional por una colección C
b) Para cada dimensión i, realizar lo siguiente:
   i. Crear un documento, denotado por D^D
   ii. Anidar en D^D un atributo cuyo nombre es igual de la tabla dimensión i; denotemos a dicho atributo por AD_i
   iii. El valor del atributo AD_i es un arreglo que contiene en cada posición un documento, denotado por D^T, que representa a cada tupla de la tabla dimensión i. El documento D^T está formado por pares simples de <atributo, valor> que representan a los atributos de una tupla de la dimensión i.
   iv. Anidar D^D en C
c) Para la tabla de hechos, realizar lo siguiente:
   i. Crear un documento, denotado por D^H
   ii. Anidar en D^H un atributo cuyo nombre es igual de la tabla de hechos; denotemos a dicho atributo por A^H
   iii. El valor del atributo A^H es un arreglo, que contiene en cada posición un documento, denotado por D^T, que representa a cada tupla de la tabla de hechos. El documento D^T está formado por pares simples de <atributo, valor> que representan a los atributos de una tupla de la tabla de hechos.
   iv. Anidar D^H en C

2.2.4 Modelo Multi-Colecciones.

En este modelo que le hemos llamado Multi-Colecciones, se crea una colección para cada tabla del modelo multidimensional y en cada una de dichas colecciones se anida un documento por cada tupla de la tabla en cuestión. Así que para transformar un modelo multidimensional relacional al modelo Multi-Colecciones se aplica el algoritmo 4 de transformación:

Algoritmo 4. Transformación de Estrella Relacional a Modelo Multicolecciones *NoSQL*.

a) Representar a cada tabla dimensión Di cuyo nombre es N por una colección denotada por C^{DN}.
b) Para cada tupla, denotada por T^D, de la dimensión Di:
   i. Crear un documento, denotado por D^D
   ii. Cada par <atributo,valor> de T^D, anidarlo dentro de D^D
   iii. Anidar el documento D^D dentro de C^{DN}.
c) Representar la tabla de hechos cuyo nombre es N por una colección denotada por C^{HN}.
d) Para cada tupla, denotada por T^H, de la tabla de Hechos:
   i. Crear un documento, denotado por D^H
   ii. Cada par <atributo,valor> de T^H, anidarlo dentro de D^H
   iii. Anidar el documento D^H dentro de C^{HN}.

3. Desarrollo.

3.1 Arquitectura del Sistema.

En la figura 2 se muestra la arquitectura del sistema desarrollado.

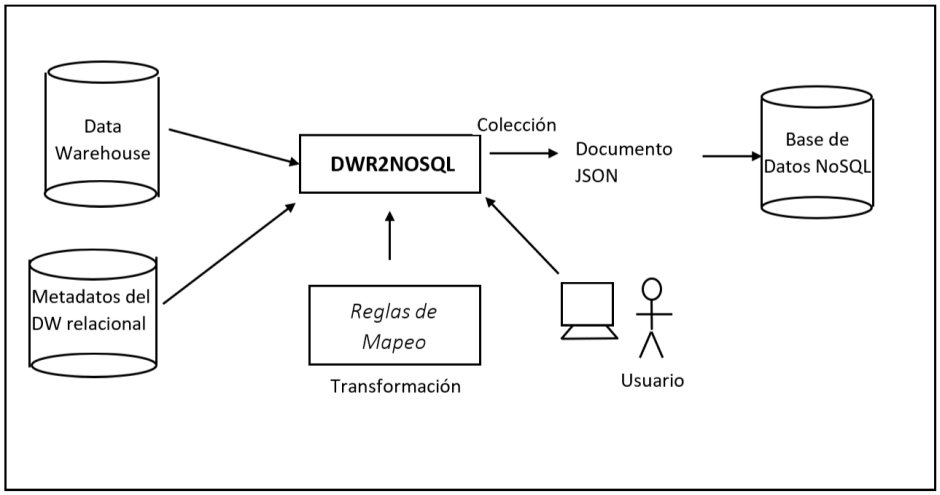


Figura 2. Arquitectura del sistema.

De acuerdo a la figura 2, el sistema opera de acuerdo a las siguientes actividades:

Paso 1: *DWR2NoSQL* accede al diccionario de datos para obtener información de las bases de datos que están definidas en el *DBMS* que almacena el *Data Warehouse* que se desea transformar a una base de datos multidimensional *NoSQL* orientadas a documentos.
Paso 2: El usuario selecciona el *Data Warehouse* relacional a transformar.
Paso 3: *DWR2NoSQL* aplica las reglas de transformación al *Data Warehouse* relacional seleccionado para generar el modelo multidimensional *NoSQL* deseado.
Paso 4: Una vez transformado el modelo, los documentos generados en formato *JSON* (JSON, 2019) se almacenan en la base de datos *NoSQL* orientado a documentos.

3.2 Diseño del Sistema.

DWR2NoSQL principalmente está formado por tres módulos los cuales se explican a continuación:

3.2.1 Módulo de Conexiones a bases de datos.

Este módulo establece la conexión al diccionario de datos del gestor de bases de datos relacional con el fin de recuperar información de las bases de datos almacenadas en él. Tal información comprende: nombre de la base de datos, nombres de tablas que conforman a la base de datos, atributos que conforman a cada tabla, atributo(s) llave(s) de cada tabla, relaciones entre tablas, entre otras cosas.

Una vez recuperada la información del diccionario de datos y seleccionada la base de datos del *Data Warehouse* a transformar, este módulo detecta de manera automática las tablas dimensiones y la tabla de hechos asociadas al modelo multidimensional de estrella seleccionado, así como las llaves de cada tabla y las relaciones entre ellas. Esta información es necesaria para poder transformar el *Data Warehouse* relacional al modelo *NoSQL* orientado a documentos deseado.

El manejador de bases de datos relacional utilizado en este proyecto fue MySQL y el manejador *NoSQL* orientado a documentos fue MongoDB.

3.2.2 Módulo de Transformación.

El segundo módulo, el cual es el principal en esta herramienta, es el que transforma de manera automática el contenido del *Data Warehouse* relacional a una base de datos equivalente *NoSQL* Orientada a documentos en formato *JSON*. La herramienta permite transformar a cualquiera de los cuatro modelos *NoSQL* orientado a documentos descritos previamente en la sección 2.2, aplicando el algoritmo de transformación pertinente definido en dicha sección.

3.2.3 Módulo de exportación a la base de datos *NoSQL*.

Por último, el tercer módulo de esta herramienta, permite que una vez generado todo el *Data Warehouse* orientado a documentos en formato *JSON*, se exporte dicho archivo *JSON* al manejador de bases de datos orientado a documentos *MongoDB* o se almacene en el sistema de archivos del sistema computacional.

3.3 Implementación del Sistema.

Para la implementación de la herramienta *DWR2NoSQL* se utilizó el siguiente software:

- Manejador de base de datos *MySQL* versión 5.7
- Manejador de base de datos *NoSQL* orientadas a documentos *MongoDB* versión 4.0
- Cliente gráfico para *MySQL*, *Workbench* versión 8.0
- *Netbeans IDE* versión 8.2. para el desarrollo del sistema en java.
- Java SE versión 8 para la implementación en código java de los modelos *NoSQL* orientados a documentos propuestos en este artículo.
- *BSON* versión 3.6.3 y *GSON* versión 2.6.2 (ambos paquetes *Jar*) utilizados para el manejo de los documentos en formato *JSON*.
- *mongo-driver* versión 3.6.3 y *mongo-driver-core* 3.6.3 (ambos paquetes jar) para almacenar en la base de datos *MongoDB* los documentos *JSON* generados.
- Conector JDBS de *MySQL JDBC*, *Driver-mysql-connector-java-5.1.23-bin.jar*, para la conexión al gestor de bases de datos de *MySQL*

Es conveniente comentar que se desarrollaron una serie de generadores de datos para insertar automáticamente información en las diferentes tablas del *Data Warehouse* relacional, esto con la finalidad de generar diferentes versiones del mismo *Data Warehouse*, pero con diferentes números de tuplas en cada una de las tablas dimensiones y de hechos del *Data Warehouse*.

3.4 Interfaz del Sistema.

En la figura 3 se muestra la pantalla principal de *DWR2NoSQL*.

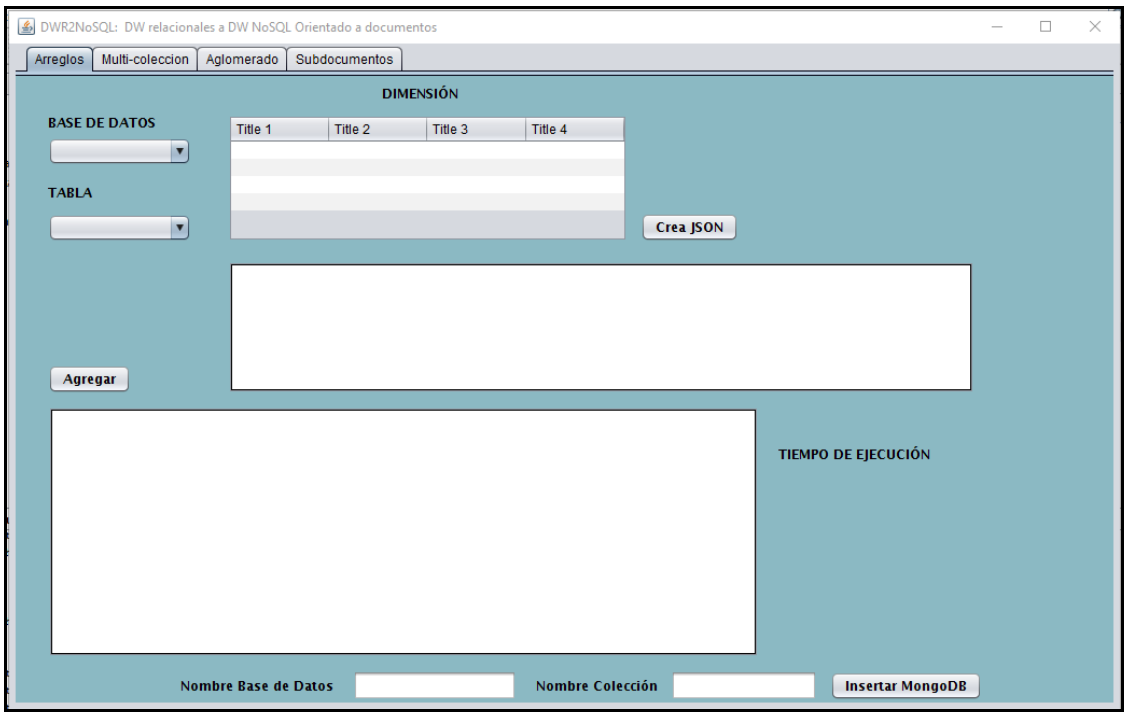


Figura 3. Interfaz de la herramienta *DWR2NoSQL*

Como se ve en la figura 3, en la parte superior de la interfaz se muestra un menú en forma de pestañas con las diferentes opciones para generar los cuatro diferentes modelos multidimensionales *NoSQL*.

De acuerdo al modelo seleccionado, la interfaz cambia ligeramente para mostrar las opciones pertinentes para cada uno de ellos, pero de manera general el sistema se comporta de acuerdo a lo descrito en los siguientes párrafos.

La interfaz muestra un *Combo Box* (debajo de la etiqueta *Base de Datos*) para seleccionar la base de datos del *Data Warehouse* relacional a transformar. Una vez seleccionada dicha base de datos se muestran, en el *Combo Box* correspondiente (debajo de la etiqueta *Tablas*), las tablas que componen a dicho modelo dimensional, es decir las tablas dimensiones y la de hechos.

El botón *Crear JSON* transforma las tablas del *Data Warehouse* relacional al formato *JSON* de acuerdo al modelo seleccionado por el usuario y dicho formato *JSON* se muestra en el área de texto de la parte inferior de la pantalla.

El botón *Insertar MongoDB* exporta el modelo multidimensional *NoSQL* orientado a documentos generado, a la base de datos y a la colección definidas en las áreas correspondientes visualizadas en la parte inferior de la pantalla.

3.5 Pruebas realizadas al sistema.

Con el propósito de evaluar la funcionalidad del transformador de *Data Warehouse* relacional a *Data Warehouse* orientada a documentos, *DWR2NoSQL*, se han ejecutado una serie de pruebas sobre los cuatro modelos de transformación mencionados en la sección 2.2. Debido a la limitante de espacio que tiene el presente artículo, únicamente se presentan algunas de ellas. Las pruebas reportadas en esta sección, se realizan considerando el modelo de estrella mostrado en la figura 1.

Prueba 1: Transformar el *Data Warehouse* relacional de la figura 1 al modelo Aglomerado. Ver figura 4 para el resultado de esta prueba. En la misma figura podemos ver el tiempo que consumió el proceso de transformación a este modelo.

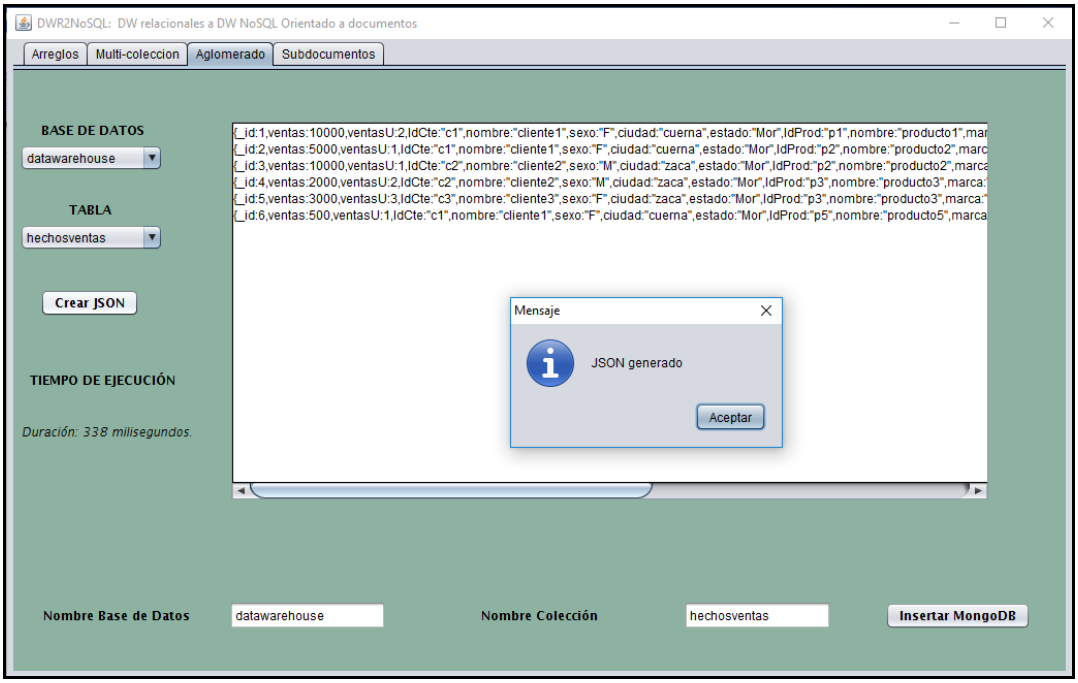
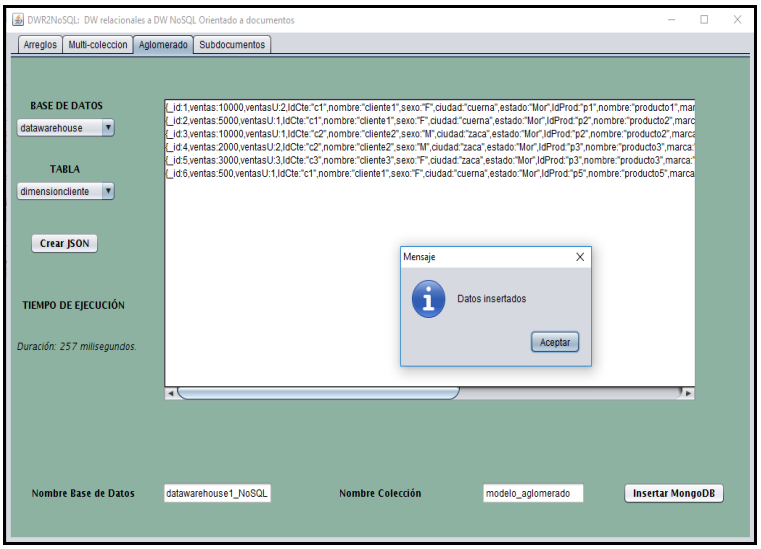


Figura 4. Resultados de la Prueba 1.

Prueba 2: En esta prueba se tomó el modelo Aglomerado generado en la Prueba 1 y se procedió a cargarlo a *MongoDB*, para esto se establece tanto el nombre de base de datos, así como el nombre de la colección. El proceso de carga de este modelo finaliza de acuerdo a la figura 5 (sección izquierda), mostrando el tiempo involucrado en la carga. Además, se muestra que dicha base de datos y dicha colección se crearon en el manejador *MongoDB*, ver figura 5 (sección derecha).



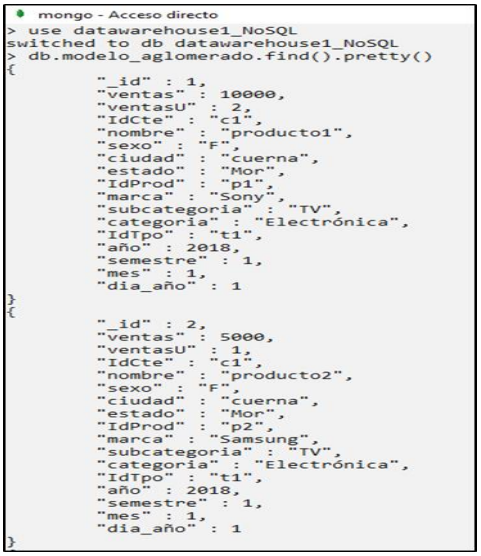


Figura 5. Resultado de la Prueba 2.

Prueba 3: Similar a la prueba 2 pero ahora para el modelo de Subdocumentos generado. Se procede a cargar el JSON generado en MongoDB, estableciendo nombre de base de datos y colección, Ver figura 6 (sección izquierda). También, se muestra el tiempo involucrado en la carga. Además, se muestra que dicha base de datos y dicha colección se crearon en el manejador *MongoDB*, ver figura 6 (sección derecha).

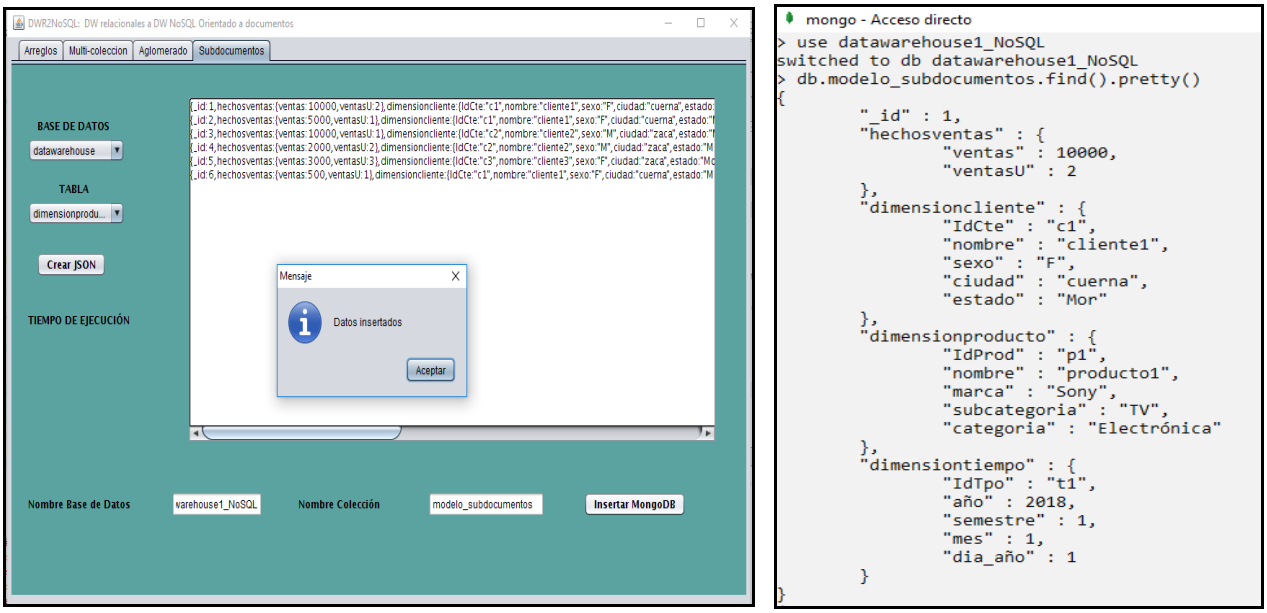


Figura 6. Resultado de la Prueba 3.

Prueba 4: De acuerdo al modelo de Arreglos generado se procedió a cargarlo a *MongoDB*, para esto se establece tanto el nombre de base de datos, así como el nombre de la colección. El proceso de carga de este modelo finaliza de acuerdo a la figura 7 (sección izquierda) mostrando el tiempo involucrado en la carga. Además, se muestra que la base de datos y respectiva colección se crearon en el manejador *MongoDB*, ver figura 7 (sección derecha).

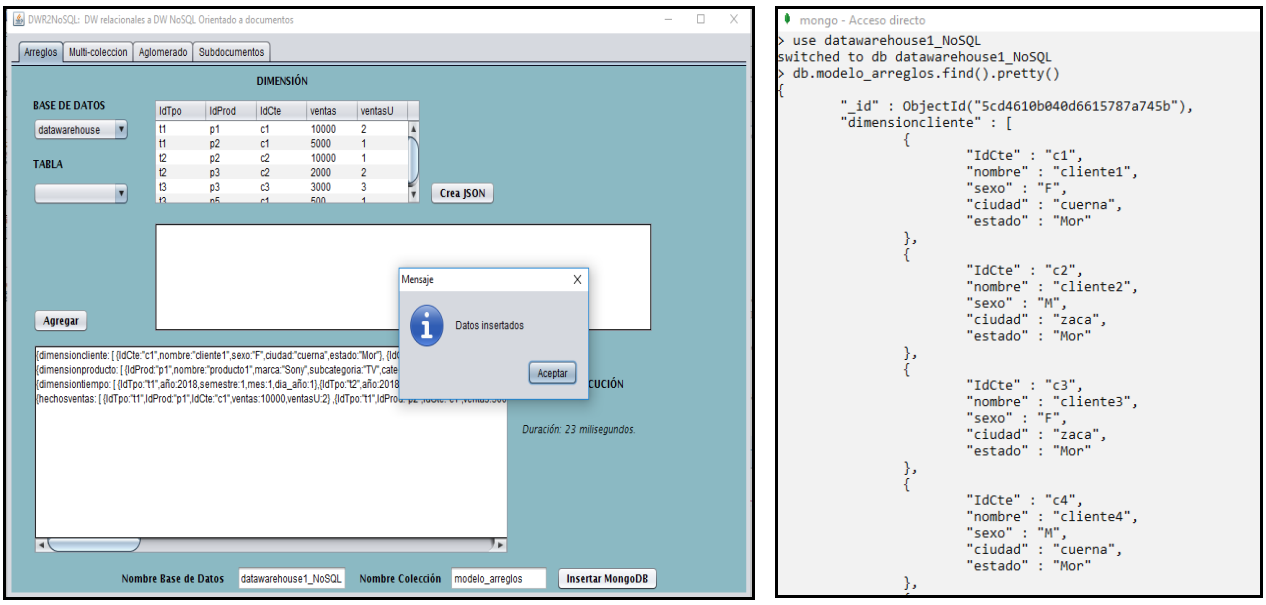


Figura 7. Resultado de la Prueba 4.

3.6 Tiempos de ejecución.

En esta sección se reportan los tiempos de ejecución involucrados al generar cada uno de los modelos multidimensionales *NoSQL* orientado a documentos, así como los tiempos que se consumieron al cargarlos en la base de datos *MongoDB*. Se considera el *Data Warehouse* relacional mostrado en la figura 1, pero realizando diferentes pruebas con un diferente número de tuplas en la tabla de hechos. Para el primer caso se consideraron 100,000 tuplas, para el segundo caso 500,000 tuplas y para el tercer caso 1,000,00 tuplas.

3.6.1 Tiempo de procesamiento en la generación de los modelos multidimensionales *NoSQL*.

En la Tabla 1 se muestran los tiempos de ejecución involucrados por *DWR2NoSQL* en la generación de los cuatro modelos multidimensionales *NoSQL* definidos en este artículo.

Tabla 1. Tiempos de ejecución en la generación de Modelos Multidimensionales *NoSQL*.

ModeloNoSQL\ No Tuplas	**100,000 tuplas**	**500,000 tuplas**	**1,000,00 tuplas**
Multi-Colecciones	**DimensiónCliente:** 342 ms **DimensiónProducto:** 223 ms **DimensiónTiempo**: 20 ms **Tabla Hechos:** 1770 ms ***Total:** 2,355 milisegundos*	**DimensiónCliente:** 350 ms **DimensiónProducto:** 230 ms **DimensiónTiempo:** 21 ms **Tabla Hechos:** 7,834ms **Total:** 8,435 milisegundos	**DimensiónlCliente:** 330 ms **DimensiónProducto:** 240 ms **DimensiónTiempo**: 24 ms **Tabla Hechos**:20,125 ms **Total:** 20,719 mili segundos
Arreglos	**DimensiónCliente:** 196 ms **DimensiónProducto:** 89 ms **DimensiónTiempo:** 29 ms **Tabla Hechos:** 940 ms **Total:** 1,254 milisegundos	**DimensiónCliente:** 200 ms **DimensiónProducto:** 78 ms **DimensiónTiempo:** 25 ms **Tabla Hechos:** 5,068 ms **Total:** 5,371 milisegundos	**DimensiónCliente:** 198 ms **DimensiónProducto**: 85 ms **DimensiónTiempo:** 28 ms **Tabla Hechos:** 10,998 ms **Total:** 11,274 milisegundos
Aglomerado	**Todo el modelo:** 7,529 milisegundos	**Todo el modelo**: 180,739 milisegundos	**Todo el modelo**: 544,080 milisegundos
Sub-Documentos	**Todo el modelo:** 6,550 milisegundos	**Todo el modelo:** 173,806 milisegundos	**Todo el modelo:** 346,529 milisegundos

Como puede verse en la Tabla1, los modelos que involucraron los tiempos de ejecución más bajos, son los modelos de Arreglos y Multi-Colección. La razón por la que dichos modelos son los más rápidos de transformar, se debe a que son modelos similares con una estructura simple. Por el contrario, los modelos que involucraron más tiempo son el Aglomerado y Subdocumentos, esto debido a que son los que tienen mayor redundancia de información.

3.6.2 Tiempo de procesamiento en la exportación de los modelos *NoSQL* a *MongoDB*.

En la Tabla 2 se muestran los tiempos de ejecución involucrados en la exportación de los cuatro modelos multidimensionales *NoSQL* a la base de datos en *MongoDB*.

Tabla 2. Tiempos de exportación de Modelos Multidimensionales *NoSQL* a MongoDB.

ModeloNoSQL\ No Tuplas	**100,000 tuplas**	**500,000 tuplas**	**1,000,00 tuplas**
Multi-Colecciones	**ColecciónDimensiónCliente:** 5,403 ms **DimensiónProducto:** 4,922 ms **DimensiónTiempo**: 2,014 ms **Tabla Hechos:** 21,600 ms ***Total:** 33,939 milisegundos*	**DimensiónCliente:** 5,490 ms **DimensiónProducto:** 4,840 ms **DimensiónTiempo:** 2,110 ms **Tabla Hechos:** 95,205 ms **Total:** 107,645 milisegundos	**DimensiónlCliente:** 5,761 ms **DimensiónProducto:** 4,942 ms **DimensiónTiempo**: 2,409 ms **Tabla Hechos**: 202,210 ms **Total:** 215,322 milisegundos
Arreglos	**Colección Multidimensional** **Total:** 30,110 milisegundos	**Colección Multidimensional** **Total:** 98,416 milisegundos	**Colección Multidimensional** **Total:** 199,241 milisegundos
Aglomerado	**Colección Multidimensional** **Total:** 34,517 milisegundos	**Colección Multidimensional** **Total:** 105,269 milisegundos	**Colección Multidimensional** **Total:** 260,249 milisegundos
Sub-Documentos	**Colección Multidimensional** **Total:** 48,184 milisegundos	**Colección Multidimensional** **Total:** 157,700 milisegundos	**Colección Multidimensional** **Total:** 350,434 milisegundos

Como puede verse en la Tabla 2, los modelos que involucraron los tiempos de ejecución más bajos fueron los modelos Multi-Colecciones y Arreglos. La razón por la que dichos modelos son los más rápidos de exportar, se debe a que son los modelos con la estructura más sencilla y con menos información redundante. Por el contrario, los modelos que involucraron más tiempo al momento de la exportación son el de Sub-Documentos y Aglomerado, ya que estos modelos tienen estructura más compleja y mayor cantidad de información redundante.

Conclusiones.

Las principales aportaciones y beneficios logrados con este trabajo de investigación son las siguientes:

- La herramienta desarrollada, llamada DWR2NoSQL, permite transformar y exportar de manera automática un *Data Warehouse* relacional a un *Data Warehouse NoSQL* orientado a documentos. El proceso de transformación y carga es muy rápido y requiere de un esfuerzo mínimo por parte del usuario.
- Se establecieron cuatro posibles formas para modelar e implementar *Data Warehouses* por medio de bases de datos *NoSQL* orientadas a documentos, los cuales son una opción viable como solución al problema de almacenar, procesar y analizar la enorme cantidad de datos generados por los llamados *Big Data.*
- Como consecuencia de tener un *Data Warehouse NoSQL* se tendría la ventaja de hacer uso del escalamiento horizontal, otorgando de esta manera mejores tiempos de respuesta y una mayor capacidad de almacenamiento.
- La modelación de *Data Warehouse* mediante modelos *NoSQL* es un tema de interés actual en el ámbito tecnológico-científico y la presente investigación ha abordado esta temática con el fin de aportar a dicho tópico.

Los resultados de esta investigación pueden sentar las bases para la realización de los siguientes trabajos futuros.

- Desarrollar una herramienta *OLAP* para manipular *Data Warehouses* orientados a documentos.
- Considerar la transformación un *Data Warehouse* relacional bajo los modelos de Copo de Nieve y de Constelación de Hechos a los cuatro modelos *NoSQL* propuestos en este trabajo.
- Realizar un trabajo de investigación semejante al presentado en este artículo, pero utilizando un modelo *NoSQL* diferente al de documentos, pudiendo ser el más factible, el modelo *NoSQL* orientado a columnas.

Agradecimientos.

Agradecemos al Tecnológico Nacional de México por el apoyo financiero otorgado para la realización del proyecto de investigación reportado en este artículo.

Referencias Bibliográficas.

Chevalier M., Malki M., Kopliku A., Teste O., Tournier T. **(2015)**. *Implementing Multidimensional Data Warehouses into NoSQL*. International Conference on Enterprise Information Systems, 2015, p172-183.

Dayley Brad **(2015)**. *NoSQL with MongoDB*. SAMS Publishing, January 2015.

Dehdouh K., Boussaid O., Bentayeb F. **(2015)**. *Using the column oriented NoSQL model for implementing big data warehouses.* The 21st International Conference on Parallel and Distributed Processing Techniques and Applications, 2015, p469-475.

Dehdouh K., Boussaid O., Bentayeb F. **(2014)**. *Columnar NoSQL star schema benchmark*. Model and Data Engineering, LNCS 8748, Springer, 2014, p281-288.

JSON (2019). Acerca de JSON. Recuperado de https://www.json.org. Acedido:2019-06-09.

Kimball R., Ross M **(2008)**. *The Data Warehouse Lifecycle Toolkit.* Second Edition. John Wiley & Sons, Inc. January 10, 2008.

Mishra D., Yazici A., Bazaran B **(2008)**. *A Case Study of Data Models in Data Warehousing*. Applications of Digital Information and Web Technologies, ICADIWT, Aug 2008.

Rocha L., Vale F., Cirilo E., Barbosa D., Mourao F. **(2015)**. *A Framework for Migrating Relational Datasets to NoSQL*. International Conference On Computational Science, ICCS 2015, Volume 51, p2593-2602.

Yangui R., Nabli A., Gargouri F. **(2016)**. *Automatic Transformation of Data Warehouse Schema To NoSQL Data Base: Comparative Study*. 20th International Conference on Knowledge Based and Intelligent Information and Engineering Systems, KES2016, September 2016, p255-264.

Información de los autores.

Francisco Javier Cartujano Escobar, Doctor en Administración con Especialidad en Sistemas de Información egresado del ITESM, Campus Ciudad de México; Maestro en Ciencias de la Computación e Ingeniero en Sistemas Computacionales, ambos títulos otorgados por el ITESM, Campus Cuernavaca. Obtuvo mención honorífica al terminar la carrera de Ingeniería. Se ha desempeñado en el sector privado como gerente de sistemas y como profesor investigador del Departamento de Computación del ITESM, Campus Ciudad de México. Ha pertenecido al SNI del CONACYT. Actualmente catedrático de la carrera de Ingeniería en Sistemas Computacionales y de la Maestría en Ingeniería en el Instituto Tecnológico de Zacatepec.

Leticia Santa Olalla Ocampo graduada como Licenciada en Informática en 1992 por el Instituto Tecnológico de Zacatepec (ITZ), graduada como Maestra en Ciencias en Ciencias de la Computación 2008 por el Centro Nacional de Investigación y Desarrollo Tecnológico (CENIDET). Actualmente catedrática de la carrera de Ingeniería en Sistemas Computacionales y de la Maestría en Ingeniería en el Instituto Tecnológico de Zacatepec. Área de Interés Ingeniería de Software.

José Pedro Aragón Hernández graduado como Ingeniero en Sistemas Computacionales en 2002 y como Maestro en Tecnologías de la Información en 2006 por el Instituto Tecnológico de Zacatepec (ITZ). Actualmente profesor de la carrera de Ingeniería en Sistemas Computacionales y de la Maestría en Ingeniería en el Instituto Tecnológico de Zacatepec.

María Fernanda Silva Reyna, graduada como Ingeniera en Sistemas Computacionales en mayo de 2019 por el Tecnológico Nacional de México, campus Instituto tecnológico de Zacatepec (ITZ).