Generación de datos NoSQL usando esquemas de bases de datos inferidos*

Alberto Hernández Chillón, Diego Sevilla Ruiz, Jesús García-Molina

Facultad de Informática, Universidad de Murcia
Campus Espinardo, Murcia, Spain
{alberto.hernandez1,dsevilla,jmolina}@um.es

Resumen La generación automática de datos resulta muy adecuada para realizar pruebas sobre bases de datos. Para sistemas relacionales se han definido diferentes enfoques y existe un buen número de herramientas. Sin embargo, todavía se ha prestado escasa atención a este problema para sistemas NoSQL. En este trabajo se presenta un primer prototipo de una herramienta desarrollada para generar datos NoSQL a partir de esquemas inferidos y representados como modelos que son conformes al metamodelo NoSQLSchema. Se describe el proceso de generación y la validación realizados, y se comenta el trabajo futuro.

Keywords: Sistemas NoSQL, Generación de bases de datos, MongoDB

1. Introducción

La generación automática de bases de datos es de gran interés para la validación de resultados de investigación en ingeniería de datos o la realización de *testing* de aplicaciones de uso intensivo de datos. Las bases de datos reales no ofrecen la posibilidad de experimentación requerida para analizar diferentes características. Algunos trabajos de investigación han abordado este problema para sistemas relacionales y han propuesto soluciones como el uso de lenguajes con los que expresar las restricciones [2,5] y varias técnicas se describen en [1]. También existen herramientas como *DTM Data Generator*[1] que ofrece soporte para la generación automática de datos para *testing*. Esta herramienta incluye una utilidad de generación de objetos JSON que pueden ser almacenados en algunos sistemas NoSQL basados en documentos como *MongoDB* o *CouchDB*.

Los enfoques y herramientas existentes tienen serias limitaciones para su uso con sistemas NoSQL. En el caso de generación de bases de datos de documentos, no soportan características básicas como la generación de objetos embebidos, referencias entre objetos y tuplas. Además, el versionado y la variabilidad de objetos de una misma entidad tampoco puede expresarse fácilmente. Por ello,

* Trabajo financiado en parte por la Cátedra SAES de la Universidad de Murcia (http://www.catedrasaes.org), un laboratorio de investigación financiado por la empresa SAES (http://www.electronica-submarina.com/).

[1] http://sqledit.com.

para nuestro trabajo de investigación en ingeniería de datos NoSQL [4,3] hemos tenido que enfrentarnos al problema de construir nuestra propia herramienta. Entre los requisitos que hemos considerado se incluye: (i) soportar diferentes sistemas de base de datos, aunque hasta ahora sólo hemos tratado los sistemas documentales, (ii) soportar las carencias antes mencionadas para las soluciones existentes, (iii) ser altamente configurable, y (iv) exportación de datos generados a varios formatos. Los datos generados pueden ser usados para tareas como: *testing* de operaciones *map-reduce*, estudio de la evolución o *provenance*, probar utilidades, o estudiar la optimización del diseño y consultas.

En este trabajo se presenta el estado actual de nuestro desarrollo. Primero presentaremos el metamodelo usado, luego describiremos el proceso de generación y su validación, y finalmente se comentarán los siguientes pasos a abordar.

2. Metamodelo de esquemas NoSQL

La generación de datos se realiza a partir de modelos que son conformes al metamodelo *NoSQLSchema* presentado en [4] para representar esquemas NoSQL. Como se muestra en la figura 1, un esquema está formado por *Entities* y cada entidad tiene una o más versiones o variaciones (*EntityVersions*). Cada versión se caracteriza por sus propiedades (*Properties*), que pueden ser atributos (*Attribute*) si su valor es de tipo primitivo o tuplas, o bien relaciones entre objetos (*Association*) que pueden ser objetos embebidos (*Aggregate*) o referencias (*Reference*).

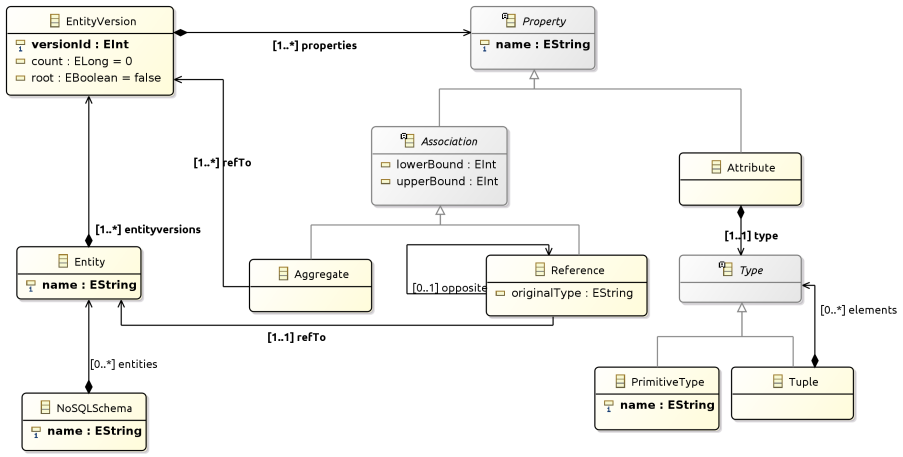

Figura 1: Metamodelo *NoSQLSchema* que representa esquemas NoSQL.

3. Generación de objetos NoSQL

La figura 2 presenta un esquema de la herramienta creada. El componente *Generador de datos* recibe como entrada un modelo *NoSQLSchema*, esto es, un esquema, y un archivo YAML de configuración, y los datos que genera son enviados al *Conector de salida* que los transfiere a una base de datos.

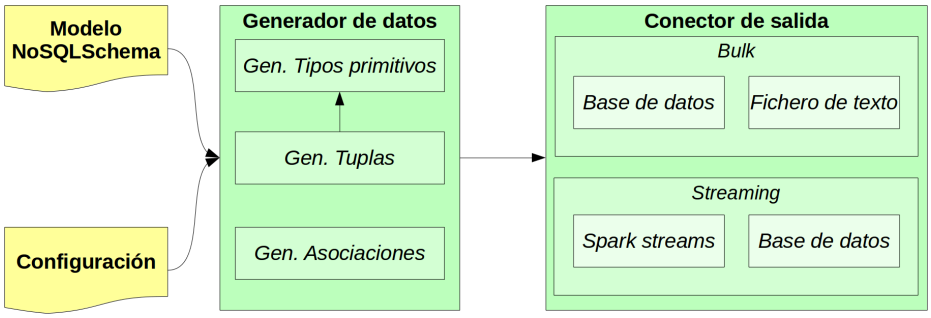

Figura 2: Esquema del proceso de generación de objetos.

El ***Generador de datos*** realiza su tarea en dos pasos secuenciales. Primero se recorren todas las versiones de cada entidad, y para cada versión se generan objetos con sus atributos. En un segundo paso se generan agregados y referencias mediante la selección aleatoria de objetos creados en la etapa anterior. El número de objetos generados por versión se define mediante parámetros en el fichero de configuración.

El ***Generador de tipos primitivos*** genera de forma aleatoria valores *Integer*, *Double*, *Boolean*, *String*, y *ObjectId*s (en formato *BSON*). Varios patrones se usan para formar strings. El ***generador de tuplas*** invoca sucesivamente al anterior para formar *tupla*s como *array*s de valores primitivos. El ***generador de asociaciones*** analiza referencias y agregaciones (cardinalidad y entidad/versión destino) para seleccionar los elementos origen y destino.

Se ofrecen opciones de configuración como: tipo de *String* (nombres, palabras, frases de longitud variable, fragmentos de texto, etc); rangos de los valores *Integer* y *Double*; número de decimales, y el rango de tamaño de tuplas. También se pueden definir probabilidades para generar datos nulos o tipos distintos a los esperados. Así se puede introducir variabilidad y errores en la base de datos, y esto resulta útil para validar procesos de *testing* y consistencia de bases de datos.

El ***Conector de salida***, en la implementación actual, hace cargas *bulk* de los objetos JSON generados a bases de datos *MongoDB* y *CouchDB*. No obstante, la arquitectura ideada permite soportar otros tipos de bases de datos NoSQL, así como la generación de *streams* de datos para su uso en herramientas como *Apache Spark Streaming*[2].

[2] `https://spark.apache.org/streaming/`.

Cabe destacar que al representar los esquemas como modelos se facilita aplicar variaciones sobre ellos para estudiar sus efectos en alguna característica, p.e. cambiar relaciones de agregación por referencias.

4. Validación del proceso de generación

Se ha validado que el proceso genera objetos cuyo esquema (modelo *NoSQLSchema*) es equivalente al representado por el modelo de entrada. Para ello se ha aplicado el proceso de inferencia descrito en [4] a los datos generados, y se ha comprobado que el modelo obtenido es equivalente.

Se ha utilizado el *dataset* de *StackOverflow*[3] para la validación. Se infirió un esquema que incluía 7 entidades y se generaron treinta millones de objetos en MongoDB repartidos entre esas entidades (colecciones): *Posts*, *Comments*, *Users*, *Votes*, *Badges*, *Tags* y *Postlinks*. Al aplicar de nuevo el proceso de inferencia para los datos generados se obtuvo un modelo equivalente al original. Este proceso de validación ha facilitado los cambios a lo largo del desarrollo. La herramienta está disponible en Github[4].

5. Trabajo futuro

Se definirá un lenguaje específico de dominio para establecer restricciones sobre el esquema original y la configuración del proceso. Algunas extensiones previstas al proceso presentado son: (i) poder elegir el número de objetos deseados para cada entidad; (ii) generar *ObjectIds* a partir de *timestamps* para estudiar la evolución del esquema; (iii) generar operaciones aleatorias sobre objetos como *updates* aleatorios, *deletes*, etc. Actualmente la implementación se realiza en iteraciones (*splits*). En un futuro se paralelizará la generación de estos *splits*.

Referencias

1. Binnig, C., Kossmann, D., Lo, E.: Towards automatic test database generation. IEEE Data Eng. Bull. 31(1), 28–35 (2008)
2. Bruno, N., Chaudhuri, S.: Flexible database generators. In: Proc. of the 31st international conference on VLDB. pp. 1097–1107 (2005)
3. Chillón, A.H., Feliciano, S., Sevilla, D., Molina, J.G.: Exploring the Visualization of Schemas for Aggregate-Oriented NoSQL Databases. In: Proc. of the ER Forum 2017, 36th Int. Conf. on Conceptual Modelling (ER 2017). pp. 72–85 (2017)
4. Sevilla Ruiz, D., Morales, S.F., Molina, J.G.: Inferring Versioned Schemas from NoSQL Databases and its Applications. In: ER, International Conference on Conceptual Modeling. pp. 467–480 (September 2015)
5. Smaragdakis, Y., et al: Scalable Satisfiability Checking and Test Data Generation From Modeling Diagrams. Automated Software Engineering 16(1), 73 (2009)

[3] StackOverflow Dataset, `https://archive.org/details/stackexchange`.
[4] `https://github.com/catedrasaes-umu/NoSQLDataEngineering`.