

Verificación del mantenimiento de la consistencia lógica en bases de datos Cassandra

Pablo Suárez-Otero, María José Suárez-Cabal, Javier Tuya

Departamento de Informática, Universidad de Oviedo, Gijón, España
{suarezgpablo, cabal, tuya}@uniovi.es

Resumen. En anteriores trabajos habíamos desarrollado un método para prevenir la producción de inconsistencias en bases de datos Cassandra. En el actual trabajo tenemos como objetivo la verificación de dicho método para lo cual hemos definido un oráculo que nos permite comprobar que las operaciones determinadas por nuestro método mantienen la consistencia. Este oráculo consta de un proceso donde se inserta la tupla en el modelo conceptual de forma paralela a su inserción en el modelo lógico que representa las tablas Cassandra y compara los datos almacenados en ambos. Si tras insertarse la tupla en ambos modelos, éstos contienen los mismos datos, se verifica que se mantiene la consistencia. Este oráculo ha sido aplicado en diversos casos de prueba seleccionados de forma sistemática, verificando que se mantuvo la consistencia en Cassandra en cada uno de ellos.

Palabras clave: Cassandra, consistencia, pruebas dinámicas, modelo conceptual

1 Introducción

Las técnicas de pruebas para garantizar la consistencia de los datos en bases de datos han estado tradicionalmente centradas en las BBDD relacionales. Ejemplos de estas técnicas incluyen la comprobación de la correcta implementación de las integridades referenciales [1] o el estudio de como éstas son implementadas en diferentes DBSM [2]. Sin embargo, la necesidad de nuevos entornos de bases de datos para gestionar Big Data ha hecho emerger nuevas tecnologías como las BBDD NoSQL y con ello también la necesidad de crear nuevas técnicas para comprobar la consistencia de los datos en ellas. Una de estas BBDD es Cassandra, donde las tablas son creadas para satisfacer consultas específicas [4]. Esto significa que, si varias consultas extraen un mismo dato, cada una de las tablas creadas para satisfacerlas almacenará ese mismo dato, manteniéndose su integridad en la aplicación cliente que trabaje con la BD. Esto difiere de las BBDD relacionales normalizadas, en las que a través de las restricciones de integridad y la normalización del modelo la integridad se controla en la propia base de datos.

Este trabajo es una continuación de trabajos previos en los que se prevenía la introducción de inconsistencias en BBDD Cassandra. En [4] se presentó un método de análisis estático preventivo para evitar la producción de inconsistencias en BD Cassandra ante inserciones de tuplas en un modelo conceptual y en [5] se realizó la evaluación de dicho método sobre un caso de estudio. La contribución de este trabajo es verificar este método, para lo cual hemos definido un oráculo que indica si los datos almacenados en

el modelo conceptual y en Cassandra son equivalentes. Este oráculo lo aplicamos en varios casos de prueba en los que insertamos tuplas obtenidas de forma sistemática.

2 Verificación del método para mantener la consistencia

En este apartado detallamos la verificación de nuestro método. En el primer subapartado detallamos el diseño de los casos de prueba, en el segundo describimos el oráculo para verificar el método y en el tercero discutimos los resultados de la experimentación.

a) Diseño de los casos de prueba

Por cada caso de prueba tenemos en cuenta dos componentes: la tupla a insertar y el estado inicial de la base de datos. Con respecto a las tuplas, éstas son una colección de pares atributo-valor de entidades o relaciones obtenidas a través de una combinación sistemática. Con respecto al estado inicial de la BD, por cada tupla se definen dos casos de prueba: uno en la que la BD parta de un estado vacío y otro en la que parta con datos que serán usados para garantizar que se mantiene la consistencia.

Las tuplas a insertar se han obtenido sistemáticamente combinando los posibles pares atributo-valor de las entidades o relaciones (atributos de las dos entidades relacionadas). En estas últimas, si alguna de las entidades relacionadas es detalle de otra entidad (relación 1:n), también se incluirá la relación entre las entidades detalle y maestro. Las tuplas obtenidas por cada entidad y relación son las siguientes:

- Una tupla en la que haya un par atributo-valor para cada atributo.
- Varias tuplas (tantas como atributos no clave haya) en las que en cada una no haya un par atributo-valor para un solo atributo no clave distinto.
- Solo para las relaciones: varias tuplas (tantas como entidades relacionadas en la tupla) en las que en cada una no haya pares atributo-valor para todos los atributos no clave de una de las entidades relacionadas.
- Una tupla en la que solo haya pares atributo-valor para los atributos clave.

Cada tupla es insertada en dos casos de pruebas en los que varía el estado inicial de la BD: en uno estará vacía y en el otro tendrá almacenados los pares atributo-valor no especificados en la tupla (los indicados en la lista anterior). Las tuplas del primer tipo se insertan en un solo caso con la BD vacía, ya que todos los atributos de la entidad o relación tienen pares atributo-valor en la tupla.

b) Oráculo para verificar método de mantenimiento de la consistencia

En este apartado definimos el oráculo que hemos creado para verificar nuestro método de mantenimiento de la consistencia, el cual hemos aplicado en cada caso de prueba de la experimentación. Este oráculo es un mecanismo de verificación que consta de tres fases: (1) Tras insertar los datos de inicialización de la BD en ambos modelos se comprueba mediante consultas sobre el modelo conceptual que éstas devuelven los mismos datos que los almacenados en las tablas del modelo lógico. (2) Inserta en el modelo conceptual la tupla que es insertada en el lógico a través de nuestro método. (3) Realiza otra vez la comprobación de la consistencia realizada en la fase 1. Para realizar estas tres fases, el modelo conceptual se ha implementado en una base de datos relacional mientras que el modelo lógico ha sido implementado en una BD Cassandra.

En la fase 1 se comprueba que la inicialización de la BD en ambas BBDD no ha creado inconsistencias a través de la ejecución de dos consultas por cada tabla Cassandra, una sobre Cassandra y la otra sobre la BD relacional. En la consulta sobre Cassandra se extraen todos los datos almacenados en la tabla mientras que en la de la BD relacional se extrae una vista de estos datos desnormalizados que representa la tabla Cassandra. El mecanismo de verificación es la comparación de lo extraído por ambas consultas, determinándose que la consistencia se mantiene si son los mismos datos.

En la fase 2 se inserta la tupla en la BD relacional insertando los valores asociados a los atributos en las entidades o relaciones correspondientes. De forma análoga a través de nuestro método se ejecutan en Cassandra las operaciones determinadas por éste.

Finalmente, en la fase 3 se realiza la misma verificación realizada en la fase 1 para comprobar que las operaciones ejecutadas en la segunda fase han mantenido la consistencia, verificándose el mantenimiento de la consistencia si ésta tiene éxito.

c) Resultados de la experimentación

En este apartado discutimos los resultados de la ejecución de varios casos de prueba en los que en cada uno insertamos una tupla sobre el modelo conceptual de un caso estudio [3] a través de nuestro método. Este caso consta de un modelo conceptual, con 4 entidades y 5 relaciones, y de un modelo lógico con 9 tablas. En Tabla 1 se muestra un resumen de estos resultados, en la que la primera columna indica sobre qué entidad o relaciones tiene la tupla pares atributo-valor. En ‘Inserciones’ se muestra el total de inserciones de tuplas con información de esa entidad o relaciones. En ‘Éxito’ se muestra el total de inserciones exitosas y en ‘Bloqueadas’ las bloqueadas. Estas inserciones bloqueadas son en las que nuestro método detecta que en una tabla donde se debería insertar la tupla, ésta no puede insertarse debido a que no se proporcionan valores para columnas clave. Esto conduciría a un estado inconsistente en el que las BD relacional y Cassandra no almacenarían la misma información, bloqueando su inserción para evitarlo. Finalmente se muestra el total de operaciones SELECT e INSERT determinadas por nuestro método que se deben ejecutar en Cassandra. A su vez, se muestra una división en cada una de estas columnas de los casos en los que el estado inicial de la BD es vacío (Vacío) y en los que parte de datos no especificados en la tupla (Datos).

Tabla 1 Resumen casos de prueba realizados

Entidad / Relación	Inserciones		Éxito		Bloqueada		INSERT		SELECT	
	Vacío	Datos	Vacío	Datos	Vacío	Datos	Vacío	Datos	Vacío	Datos
Review/User/Venue	15	12	15	12	0	0	0	0	0	0
Artifact	5	4	5	4	0	0	5	4	6	6
Features	10	9	7	9	3	0	21	27	6	27
Posts-Rates	14	13	9	13	5	0	36	37	9	9
LikesV	10	9	10	9	0	0	10	9	6	5
LikesA-Features	14	13	11	13	3	0	63	63	26	45
Total	68	60	57	60	11	0	135	140	53	92

En total se han ejecutado 128 casos de prueba (68 + 60). El oráculo ha verificado que nuestro método ha mantenido la consistencia en todos estos casos. En la primera fila se representa de forma conjunta las inserciones de tuplas que contienen información de las entidades Review, User o Venue debido a que en todas se obtuvo una salida vacía de nuestro método. Para cada una de estas entidades se han insertado 5 tuplas con la

BD vacía y 4 en la que parte con datos. Volvemos a comprobar, tal y como se había observado anteriormente [5], que la desnormalización del modelo obliga a ejecutar varios SELECT e INSERT en varias tablas para mantener la consistencia ante la inserción de una tupla. En cuanto a las inserciones bloqueadas, éstas solo se producen en los casos donde no se han insertado datos iniciales en la BD, mostrando como el estado final de la BD puede variar según el estado inicial de ésta insertando la misma tupla.

3 Conclusiones y trabajo futuro

En trabajos previos, se había definido un método para prevenir la producción de inconsistencias en una BD Cassandra ante la inserción de tuplas en un modelo conceptual [4], así como la evaluación de dicho método [5]. En este trabajo, se ha avanzado en este método aportando su verificación mediante la definición de un oráculo que hemos aplicado en varios casos de prueba. Este oráculo inserta en el modelo conceptual la tupla que nuestro método inserta en el modelo lógico con el objetivo de verificar que ambos modelos almacenan la misma información. En la experimentación donde hemos insertado varias tuplas a través de nuestro método, hemos aplicado este oráculo obteniendo el resultado de que nuestro método mantuvo la consistencia en cada inserción.

Como trabajo futuro se abordará el mantenimiento de la consistencia de filas directamente insertadas en una tabla Cassandra, el cual se había definido como el enfoque bottom-up en un trabajo previo [4]. Para darle una solución a este enfoque se definirá un nuevo método que se integre con el explicado en este y anteriores trabajos.

Agradecimientos

Este trabajo ha sido realizado bajo los proyectos de investigación TIN2013-46928-C3-1-R y TIN2016-76956-C3-1-R/-2-R, financiados por el Ministerio de Economía y Competitividad, y fondos FEDER. También ha sido realizado bajo el proyecto GRUPIN14-007, financiado por el Principado de Asturias y fondos FEDER.

Referencias

1. McMinn, P., Wright, C. J., McCurdy, C. J., & Kapfhammer, G. (2017). Automatic Detection and Removal of Ineffective Mutants for the Mutation Analysis of Relational Database Schemas. *IEEE Transactions on Software Engineering*
2. McMinn, P., Wright, C. J., & Kapfhammer, G. M. (2015). The effectiveness of test coverage criteria for relational database schema integrity constraints. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 25(1), 8.
3. Chebotko, A., Kashlev, A., Lu, S.: A Big Data Modeling Methodology for Apache Cassandra. *Proc. IEEE Int. Congress on Big Data*, pp. 238–245 (2015)
4. Suárez-Otero, P., Gutierrez, J., de la Riva, C., & Tuya, J. *Mantenimiento de la Consistencia Lógica en Cassandra*. JISBD 2017
5. Suárez-Otero, P., Suárez-Cabal, M.J. & Tuya, J. *Evaluación del Mantenimiento de la Consistencia Lógica en Cassandra*. JISBD 2018