Mantenimiento de la Consistencia Lógica en Cassandra

Pablo Suárez-Otero[1], Javier Gutierrez[2], Claudio de la Riva[1], Javier Tuya[1]

[1]Departamento de Informática, Universidad de Oviedo, Gijón, España
{suarezgpablo, claudio, tuya}@uniovi.es
[2]Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, España
javierj@lsi.us.es

Resumen. A diferencia de las bases de datos relacionales, en bases de datos NoSQL como Cassandra es muy común que exista duplicidad de los datos en diferentes tablas. Esto es debido a que, normalmente, las tablas son diseñadas en base a las consultas y a la ausencia de relaciones entre ellas para primar el rendimiento en las consultas. Por tanto, si los datos no se actualizan convenientemente, se pueden producir inconsistencias en la información almacenada. Es relativamente fácil introducir defectos que originan inconsistencia de datos en Cassandra, sobre todo durante la evolución de un sistema en el que se crean nuevas tablas, y éstos son difíciles de detectar utilizando técnicas convencionales de pruebas dinámicas. El desarrollador es quien debe preocuparse de mantener esta consistencia incluyendo y actualizando los procedimientos adecuados. Este trabajo propone un enfoque preventivo a estos problemas, estableciendo los procesos necesarios para asegurar la calidad de los datos desde el punto de vista de su consistencia, facilitando así las tareas del desarrollador. Estos procesos incluyen: (1) un análisis estático considerando el modelo conceptual, las consultas y el modelo lógico de la aplicación, para identificar qué elementos (tablas o columnas) de la base de datos se ven afectados por un cambio, y (2) la determinación y ejecución de las operaciones que aseguren la consistencia de la información.

Palabras clave: Apache Cassandra, consistencia lógica, pruebas estáticas

1 Introducción

Apache Cassandra [1]es una base de datos NoSQL distribuida, escalable y de alta disponibilidad. Las principales diferencias de Cassandra con respecto a los sistemas de bases de datos relacionales son la inexistencia de restricciones de integridad y la ausencia de relaciones entre tablas. Debido en parte a estas diferencias y a que las tablas son creadas para consultar datos específicos (query-driven) [2], los datos pueden estar duplicados en la misma o en diferentes tablas. La responsabilidad de mantener la consistencia de estos datos recae en el desarrollador. La duplicidad de los datos y la posible producción de inconsistencias se reflejan en la Fig. **1** con un ejemplo simple. Se tienen dos entidades ("Autor" y "Libro") y dos tablas Cassandra en las que consultar información de libros: "Libros" para obtener información de un libro localizándolo por su identificador y "Libros_por_autor" para consultar la información de libros por su autor. Si

la inserción de un libro se realiza solo en "Libros_por_autor", se produciría una inconsistencia en los datos ya que la tabla "Libros" no contendría el nuevo libro.

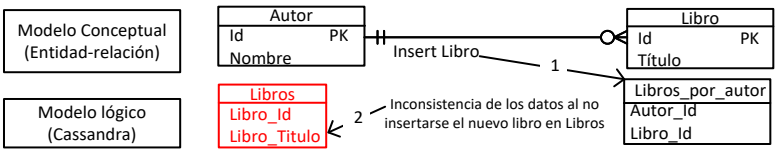

Fig. 1. Inserción en una sola tabla produciendo inconsistencia

Durante el desarrollo de una aplicación, pueden producirse diferentes cambios que afectarán a la consistencia de los datos. En general, se pueden clasificar en dos categorías:

1. Modificación del modelo lógico: Afectan al diseño de la base de datos.
2. Modificación de datos: Afectan a los datos almacenados en las tablas.

Este trabajo se centrará en establecer el proceso para evitar la inconsistencia de los datos que se podría producir en el segundo tipo. Esta aportación será de utilidad para el desarrollador permitiéndole asegurar la consistencia de la información y previniendo posibles defectos en las salidas de las consultas que acceden a las tablas.

El resto del artículo es como sigue: en la sección 2 se describen el estado del arte así como el marco general y el detalle de este trabajo. El artículo finaliza en la sección 3 con las conclusiones y las posibles líneas de trabajo futuro.

2 Estado del arte y descripción del estudio

Cuando se menciona el término "consistencia" en Cassandra, éste suele referirse a la consistencia física [3], la cual se define en función de cómo de actualizada y sincronizada está una fila de una tabla Cassandra en todas sus réplicas [4]. Sin embargo, el objetivo de esta investigación es asegurar lo que denominaremos "consistencia lógica". La consistencia lógica de los datos en Cassandra consiste en conseguir que los datos duplicados en una o varias tablas sean consistentes entre sí.

Otros estudios [5] [6] se han centrado en la automatización del diseño del modelo lógico para bases de datos Cassandra. Los diseños que generan se basan, además de en las consultas (queries), en un modelo conceptual (representado con un modelo entidad-relación) de la aplicación.

En nuestro trabajo, también manejaremos el modelo conceptual, las queries y el correspondiente modelo lógico (tablas Cassandra) obtenidos aplicando los conceptos presentados en [5], pero no para abordar el diseño de la base de datos. En este caso, se trata de mantener la consistencia lógica ante cambios en los datos, que se pueden producir a al nivel del modelo conceptual o del modelo lógico. Se han identificado dos posibles enfoques para mantener esta consistencia, como se muestran en la Fig. **2** (este trabajo se centrará en el primero de ellos):

1. Top-down: Partiendo de un cambio en un dato correspondiente a una entidad (modelo conceptual), determinar qué datos en tablas y columnas Cassandra (modelo lógico) se deben modificar, a través de un mapeo atributo/columna, y cómo.

2. Bottom-up: Partiendo de un cambio en un dato en tablas y columnas Cassandra (modelo lógico), determinar qué otros datos en Cassandra se deben modificar y cómo. Para ello se determinará primero qué entidades y atributos se verían afectados por el cambio y, partiendo de ellos, se aplicaría el enfoque top-down.

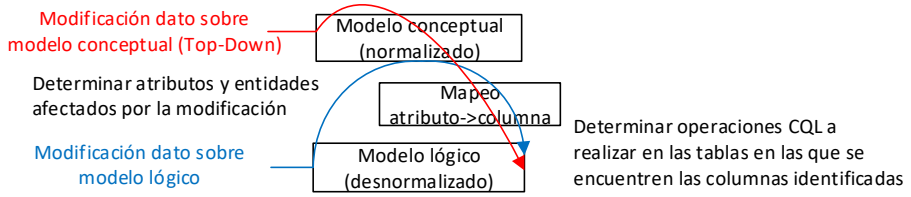

Fig. 2. Enfoques para el mantenimiento de la consistencia lógica de datos

Dado el cambio de un dato en el modelo conceptual (tupla o atributo) se realiza un análisis estático, ilustrado en la Fig. **3**, para determinar en qué tablas están estos datos. En primer lugar (paso 1), se identifican las entidades y atributos afectados por el cambio. Posteriormente (paso 2), se realiza el mapeo de a qué columnas se corresponden estos atributos (los nombre de las columnas siguen el convenio "entidad + atributo"). Finalmente (paso 3), se determinan las tablas a partir de las columnas identificadas.

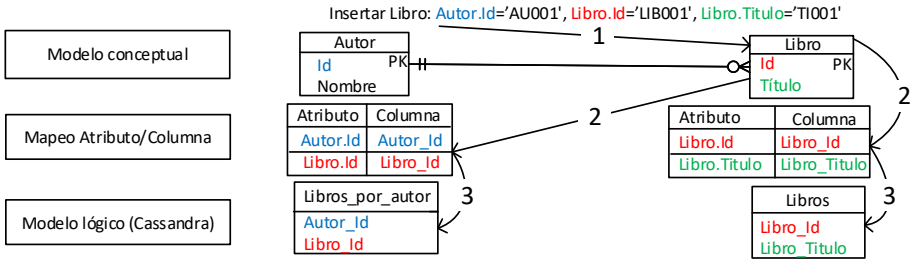

Fig. 3. Proceso de análisis estático

Tras el análisis estático, se determinan y ejecutan las operaciones para mantener la consistencia lógica. En la Fig. **4** se ejemplifica este proceso: en el paso 1 se determinan las operaciones y en el paso 2 se ejecutan.

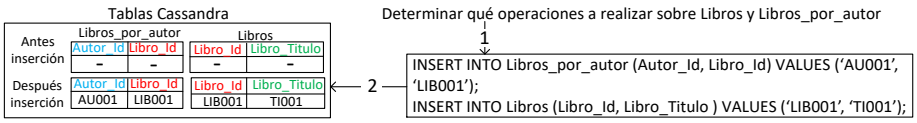

Fig. 4. Proceso de determinación y ejecución de operaciones

En este ejemplo, las operaciones son ejecutadas directamente ya que en las tablas solo se almacena la información proporcionada como entrada para el nuevo libro. No siempre será así: existen casos en los que para ejecutar las operaciones se necesita información adicional que se encuentra dispersa en la base de datos. Para obtener esta información, se deberán ejecutar consultas sobre tablas que serán localizadas siguiendo un proceso similar al análisis estático comentado.

3 Conclusiones y trabajo futuro

Los procesos para prevenir la inconsistencia lógica de datos descritos en este trabajo se han aplicado a diferentes combinaciones de modelo conceptual, modelo lógico y cambio de datos. En concreto, se han identificado 47 casuísticas diferentes. Los modelos conceptuales considerados presentan relaciones 1:n (y sus derivados como 1:n:n), n:m y combinaciones de estos. Respecto a las modificaciones, se han estudiado inserciones y borrados de datos a nivel del modelo conceptual. Además, por cada combinación de modelo conceptual y modificación, se han considerado varios modelos lógicos para disponer de situaciones en las que las operaciones se puedan ejecutar y situaciones en las que se debe extraer información de la base de datos para ejecutarlas. Estos procesos han sido automatizados en una herramienta que contempla todas estas casuísticas.

El uso de esta herramienta proporciona una ayuda al desarrollador para la verificación temprana de potenciales problemas de inconsistencias. Adicionalmente, proporciona las operaciones necesarias para la resolución de estas inconsistencias.

Como trabajo futuro se estudiarán las diferentes casuísticas para la modificación de datos desde el modelo lógico (enfoque bottom-up). Se estudiará también la posibilidad de crear modelos conceptuales a partir del modelo lógico, con el objetivo de que los resultados de este trabajo puedan ser aplicados en aplicaciones cuyo diseño y desarrollo se han basado únicamente en las consultas.

Agradecimientos

Este trabajo ha sido realizado bajo los proyectos de investigación 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. Cassandra (último acceso: Febrero 2017), http://cassandra.apache.org
2. Data Modeling Concepts (último acceso: Febrero, 2017), http://docs.datastax.com/en/cql/3.3/cql/ddl/dataModelingApproach.html.
3. Fan, H., Mckenzie, M., Wong, B.: Understanding the Causes of Consistency Anomalies in Apache Cassandra. Proc. VLDB Endow. 8, pp. 810–813 (2015)
4. About Data Consistency (último acceso: Febrero, 2017), http://docs.datastax.com/en/cassandra/2.1/cassandra/dml/dmlAboutDataConsistency.html
5. 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)
6. Diego Sevilla Ruiz, Severino Feliciano Morales, Jesús García Molina: Inferring Versioned Schemas from NoSQL Databases and Its Applications. ER 2015: 467-480